

Multi-Scale Simulation of Cerebral Blood Flow and Oxygen Exchange for the Entire Mouse Brain

BY

Grant Hartung

B.S., University of Illinois at Chicago, 2012

M.S., University of Illinois at Chicago, 2015

THESIS

Submitted as partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Bioengineering
in the Graduate College of the
University of Illinois at Chicago, 2020

Chicago, Illinois

Defense Committee:

Andreas Linninger, Chair and Advisor
Ali Alaraj, Neurosurgery
Simon Alford, Anatomy and Cell Biology
Dieter Klatt, Bioengineering
Richard Magin, Bioengineering

This thesis is dedicated to my father (V. Adam Hartung) and to the memory of my brother Alexander M. Hartung. Without your influence and support throughout my life, I would never have attempted, nor made it through this work.

Acknowledgements

I would like to thank my thesis committee - Dr. Alaraj, Dr. Alford, Dr. Magin and Dr. Klatt - for all their assistance and guidance in accomplishing this research. They helped me to progress through the research and enjoy the process of academic enlightenment. I would also like to thank other advisors, Dr. Hetling, Dr. Patton, and Dr. Khetani, for offering advice on graduate studies and research. I would also like to thank all the members of the lab past and present who assisted with all this work. Without the great effort of my colleagues, this work may never have come to fruition.

I would like to specifically thank Ryan Morley, Shoale Badr, and Claudia Vesel for their help. Their contributions were instrumental in streamlining the data curation process.

I would like to thank Ian Gould for laying the groundwork and establishing the fundamentals on which this thesis was built. His methods began a shift from small simulations of the brain (only a few segments at a time) to large simulations of massive microcirculatory structures. Without his breakthroughs, this work could not have been accomplished.

I would most importantly like to thank my PI, Dr. Andreas Linninger, for all his guidance and teachings throughout the duration of my dissertation. The countless hours spent programming, deriving, and designing approaches to solve new problems were essential to the success of this project.

Contribution of Authors

Section 1, 2 and 4 of this manuscript is an introduction to the topics and relevance of the thesis written entirely by me.

Chapter 3 is a reprinting of a manuscript (Mathematical synthesis of the cortical circulation for the whole mouse brain-part I. theory and image integration. A Linninger, G Hartung, S Badr, R Morley, Computers in Biology and Medicine, 2019) for which I was the second author and held the duties of data curation, algorithm advancement, figure design, and data analysis. Shoale and Ryan (3rd and 4th authors) assisted in data curation by executing codes generated by me and Dr. Linninger.

Chapter 5 was previously published (as Simulations of blood as a suspension predicts a depth dependent hematocrit in the circulation throughout the cerebral cortex G Hartung, C Vesel, R Morley, A Alaraj, J Sled, D Kleinfeld, A Linninger, PLoS Computational Biology, 14(11), e1006549, 2018). I am the first author and am responsible for the code implementation, procurement of data, data analysis and figure/text design. Claudia, the second author, was primarily responsible for assisting in literature research related to comparable models and Ryan was responsible for executing codes generated by me. Dr. Kleinfeld and Dr. Sled generously shared empirical data to which all synthetic structures were compared (as ground truth) and Dr. Alaraj assisted in the design of the manuscript.

Portions of Section 33-34 were printed (as Modeling the diffusion of D-2-hydroxyglutarate from IDH1 mutant gliomas in the central nervous system. A Linninger, GA Hartung, BP Liu, S Mirkov, K Tangen, RV Lukas, D Unruh, CD James, JN Sarkaria, C Horbinski, Neuro-Oncology,

20(9), 1197–1206, 2018) to which I am the 2nd author. Only the portion of the manuscript generated by me was added to this thesis.

Portions of Section 23 were printed (as Starling forces drive intracranial water exchange during normal and pathological states. A Linninger, C Xu, K Tangen, G Hartung, Croatian Medical Journal, 58, 384-394, 2017.) but only the portions constructed by me are included in this thesis.

Portions of Section 30 are included in a manuscript (Quantification of blood flow patterns in the cerebral arterial circulation of individual (human) subjects. Chang Sub Park, Grant Hartung, Ali Alaraj, Xinjian Du, Fady T Charbel, Andreas A. Linninger. *in print*) however only the portions generated by me are included in this thesis.

Table of Contents

1	INTRODUCTION	1
2	IMAGE-BASED CIRCULATORY NETWORK (ICNS) SYNTHESIS PART I: THEORY AND IMAGE INTEGRATION. 9	
2.1	INTRODUCTION.....	9
2.2	METHODOLOGY.....	11
2.2.1	<i>The formal optimization problem.....</i>	<i>12</i>
2.2.2	<i>Image-guided segment addition</i>	<i>20</i>
2.2.3	<i>Synthesis of closed networks (=microvascular closure).....</i>	<i>21</i>
2.2.4	<i>Neuroimage data for validation of synthetic growth.....</i>	<i>22</i>
2.3	IMPLEMENTATION	22
2.4	APPLICATIONS AND RESULTS	25
2.4.1	<i>Simple application of open arterial trees.</i>	<i>25</i>
2.4.2	<i>Synthesis of closed cortical blood supply with microcirculatory closure</i>	<i>25</i>
2.4.3	<i>Mathematical network model of the entire MCA territory</i>	<i>30</i>
2.4.4	<i>Cortical blood supply (whole brain circulation)</i>	<i>32</i>
2.5	DISCUSSION	35
2.6	LIMITATIONS.....	38
2.7	CONCLUSIONS	38
2.8	DERIVATION OF FLOW RATIOS IN BALANCED TREE	39
2.9	MICROVASCULAR CLOSURE PSEUDOCODE	42
2.10	SAMPLE GENERATOR PSEUDOCODES	42
3	IMAGE-BASED CIRCULATORY NETWORK (ICNS) SYNTHESIS PART II: AUTOMATED MATCHING OF NETWORK TOPOLOGY.....	43
3.1	INTRODUCTION.....	43
3.2	METHODS	44
3.2.1	<i>Automated topological characteristic matching.....</i>	<i>44</i>
3.2.2	<i>Automated matching of two cumulative density functions</i>	<i>52</i>
3.2.2.1	<i>Using a non-constant modifier to match CDFs</i>	<i>54</i>
3.3	APPLICATIONS	56
3.3.1	<i>Microcirculatory subsections.....</i>	<i>56</i>
3.3.2	<i>MCA/hemisphere.....</i>	<i>60</i>
3.3.3	<i>Human</i>	<i>62</i>
3.4	DISCUSSION	64
4	LARGE MICROVASCULAR NETWORKS REVEAL DEPTH DEPENDENT HEMATOCRIT GRADIENT	66
4.1	INTRODUCTION.....	66
4.1.1	<i>Regulation.</i>	<i>67</i>
4.2	MATERIALS AND METHODS	69
4.2.1	<i>Pial surface data acquisition</i>	<i>71</i>
4.2.2	<i>Microcirculatory data acquisition</i>	<i>71</i>
4.2.3	<i>Synthesis of large circulatory networks.....</i>	<i>72</i>
4.2.4	<i>Blood flow.....</i>	<i>72</i>
4.3	RESULTS.....	74
4.3.1	<i>Morphometrics.....</i>	<i>74</i>
4.3.2	<i>Synthetic data sets</i>	<i>75</i>
4.3.3	<i>Branching patterns.....</i>	<i>76</i>
4.3.4	<i>Network Effects in large-scale models</i>	<i>78</i>
4.3.5	<i>Path analysis of hematocrit and layer dependence</i>	<i>79</i>
4.3.6	<i>Extension to brain-wide hemodynamic simulations.....</i>	<i>83</i>
4.3.7	<i>Complete circulation of the MCA territory including arterial and venous side.....</i>	<i>84</i>
4.3.8	<i>Blood flow.....</i>	<i>86</i>
4.4	DISCUSSION	90

Table of Contents (continued)

4.4.1	<i>Morphometrics</i>	90
4.4.2	<i>Blood Flow</i>	91
4.4.3	<i>Hematocrit</i>	92
4.4.4	<i>Synthesis</i>	94
4.4.5	<i>MCA</i>	95
4.4.6	<i>Boundary conditions</i>	96
4.4.7	<i>Limitations</i>	97
4.4.8	<i>Conclusions</i>	98
4.5	DEFICITS IN DIFFERENT HEMATOCRIT SPLIT RULES	100
4.6	IMPLEMENTATION OF THE BIPHASIC BLOOD FLOW.....	103
4.7	HEMATOCRIT DEPENDENCE ON DIAMETER.....	106
5	IMPROVED OXYGEN SIMULATION FOR LARGE MICROVASCULAR NETWORKS	107
5.1	INTRODUCTION.....	107
5.1.1	<i>Computational paradigms</i>	108
5.2	METHODS	110
5.2.1	<i>Oxygen tension measurement acquisition in young and aged brain</i>	110
5.2.2	<i>Vascular structure acquisition</i>	110
5.2.3	<i>Mathematical blood flow and oxygen model</i>	111
5.2.4	<i>Vascular masking for a Cartesian mesh</i>	112
5.2.5	<i>Implementation</i>	115
5.3	RESULTS.....	116
5.4	DISCUSSION	124
5.5	APPENDIX A: VESSEL IDENTIFICATION IN 3D	128
6	DISCUSSION.....	131
7	APPENDICES	133
7.1	APPENDIX A: CEREBROVASCULAR SYNTHESIS IMPLEMENTATION	133
7.1.1	<i>Segment addition</i>	133
7.1.2	<i>Validating the optimal bifurcation point</i>	135
7.1.3	<i>Assigning diameter</i>	137
7.1.4	<i>Staged growth algorithm implementation</i>	138
7.1.4.1	Pial Surface	139
7.1.4.1.1	Projecting bifurcation point to surface	141
7.1.4.2	Penetrating arterioles and ascending venules.....	143
7.1.4.3	Capillaries	146
7.1.4.4	Microcirculatory closure.....	148
7.1.4.5	Hemisphere implementation notes.....	151
7.2	APPENDIX B: SAMPLE GENERATOR IMPLEMENTATION AND VALIDATION	153
7.2.1	<i>Analytic X-Y plane sampler</i>	153
7.2.2	<i>Analytic hexahedral sampling</i>	154
7.2.3	<i>Triangle sample generator</i>	154
7.2.4	<i>Triangle prism sample generator</i>	159
7.2.5	<i>Tetrahedral sampling</i>	159
7.2.5.1	Method 1: using the vector norms	160
7.2.5.2	Method 2: using the random length sampling	161
7.2.5.3	Method 3: mirroring.....	163
7.3	APPENDIX C: CALCULATING CORTICAL DEPTH FROM A MESH	165
7.4	APPENDIX D: RAW DATASET TOPOLOGICAL ANALYSIS	166
7.4.1	<i>Calculating Murray coefficient</i>	166
7.4.2	<i>Spectra of Murray coefficient from μCT data</i>	169
7.4.3	<i>The spectra of Murray coefficient from 2PLSM microcirculation</i>	171

Table of Contents (continued)

7.4.3.1	Anatomical characteristics of cerebral microcirculatory networks	176
7.5	APPENDIX E: PERPENDICULAR LINE CONNECTIONS, ALTERNATIVE METHODS.....	183
7.6	APPENDIX F: MODIFYING TORTUOSITY OF A SPLINE.....	188
7.6.1	<i>Adding tortuosity</i>	188
7.6.2	<i>Reducing Tortuosity</i>	190
7.7	APPENDIX G: USING A QUADRATIC MODEL WITH Y-OFFSET OF L_0 TO MATCH CDFs	194
7.7.1	<i>Relevant codes</i>	197
7.7.1.1	Code for bifurcation volume optimization evaluation	197
7.7.1.2	Code for tetrahedral sample generation evaluation	199
7.7.1.3	Code for intersecting a point with a line at a 90 degree angle in Matlab	202
7.7.1.4	CDF matching case study and code in Matlab	202
7.8	APPENDIX H: PIAL GROWTH MODEL FOR HUMAN CORTEX	222
7.8.1	<i>Growth from expansion</i>	223
7.8.1.1	Case Studies:.....	224
7.8.2	<i>Using ridge detection (paraboloid fitting method)</i>	229
7.8.2.1	Case studies	240
7.8.2.2	Growth along the valleys	242
7.9	APPENDIX I: RECONSTRUCTING NETWORK GEOMETRY	245
7.9.1	<i>Overview</i>	245
7.9.2	<i>Reconstructing with VMTK</i>	247
7.9.2.1	Step 1. run vesselness filter	247
7.9.2.2	Step 2. Run VMTK	248
7.9.2.3	Step 3. run VTP-to-case/nwk converter.....	252
7.10	APPENDIX J: MESH RECONSTRUCTION.....	254
7.11	APPENDIX K: NETWORK ANALYSIS	256
7.11.1	<i>Modeling with networks</i>	256
7.11.2	<i>Problem Formulation and solving</i>	257
7.11.3	<i>Tellegen theorem</i>	260
7.11.4	<i>Linear flow (Hagen Poiseuille)</i>	261
7.11.4.1	Accounting for turbulent flow in HP.....	264
7.11.4.2	Problem simplification	267
7.11.5	<i>Examples of Network Analysis</i>	267
7.12	APPENDIX L: ALTERNATIVE MODELS OF COMPUTING HEMATOCRIT AND BIPHASIC VISCOSITY.....	277
7.12.1	<i>Viscosity</i>	277
7.12.1.1	Pries In Vitro and In Vitro Modified models	281
7.12.1.2	Pries In Vivo model	283
7.12.1.3	Charm and Kurland, marginal zone layer.....	286
7.12.1.4	Kiani and Hudetz model.....	288
7.12.1.5	Viscosity trends.....	290
7.12.2	<i>Plasma skimming</i>	291
7.12.2.1	Pries and Secomb model	293
7.12.2.2	Linninger – KPSM (2015) and Beta Model (2017):.....	294
7.12.2.3	Dellimore	296
7.12.2.4	Fenton	297
7.12.2.5	Plouraboue	298
7.12.2.6	Balogh and Bagchi (3D immersive boundary method)	299
7.12.2.7	Plasma skimming implementation	302
7.12.2.8	Modified m-coefficient model	302
7.12.3	<i>Application to networks</i>	304
7.12.4	<i>Comparison of plasma skimming models to empirical data</i>	306
7.12.5	<i>Closer look: variation of viscosity with diameter</i>	308
7.12.6	<i>Closer look: variation of viscosity with hematocrit level</i>	309
7.12.7	<i>Closer look: Variation in plasma skimming trends with diameter</i>	311
7.12.8	<i>Closer look: Balogh and Bagchi (3D immersive boundary method)</i>	312
7.13	APPENDIX M: METHODS FOR INTERROGATING MODEL PREDICTIONS	321

Table of Contents (continued)

7.13.1	<i>Measuring goodness of fit</i>	321
7.13.1.1	The coefficient of determination (R^2) for linear fit:	321
7.13.1.2	Adjusted coefficient of determination (for use with higher order fits):	323
7.13.1.3	Nonlinear goodness of fit Akaike Information Criterion (AIC):	324
7.13.1.4	Linear Regression Examples:.....	325
7.13.1.5	Validation by direct minimization of residual error surface	329
7.13.2	<i>Investigating simulation results</i>	330
7.13.2.1	3D immersive visualizations	330
7.13.2.2	Calculating Perfusion	331
7.13.2.3	Path analysis	332
7.13.3	<i>Logarithmic and power scaling</i>	336
7.13.3.1	Surface flow divergence visualization.....	337
7.13.3.2	Volume flow divergence visualization	341
7.13.3.3	Displaying lognormal network trends.....	341
7.13.3.4	Case Study 1: Flow in a Microcirculatory KF dataset	343
7.13.3.5	Case study 2: calculation of mean and standard deviation	346
7.13.3.6	Case study3: calculation of median, interquartile range and median absolute deviations	347
7.14	APPENDIX N: DISCRETIZATION SCHEMES.....	351
7.14.1	<i>Translating between analytic and discretized form in 1D</i>	351
7.14.1.1	Graphical example	353
7.14.2	<i>Two dimensions</i>	355
7.14.2.1	Graphical example of numerical discretization	355
7.14.3	<i>Three dimensions</i>	356
7.14.3.1	Graphical example.....	357
7.14.4	<i>Finite mesh discretization</i>	359
7.15	APPENDIX O: BOUNDARY CONDITIONS.....	361
7.15.1	<i>Dirichlet and Neumann</i>	362
7.15.2	<i>Proportional flow</i>	362
7.15.3	<i>Periodic</i>	363
7.15.4	<i>Constant Gain</i>	364
7.15.5	<i>Implementation</i>	364
7.15.5.1	Choosing bulk flow BCs.....	365
7.15.5.2	Choosing hematocrit BCs.....	366
7.15.5.3	Choosing oxygen BCs	366
7.15.6	<i>BCs from empirical measurements and optimization</i>	367
7.15.6.1	Mathematical formulation of the optimization equation.....	368
7.15.6.2	Using the Fourier Transform to produce time-dependent BCs	374
7.15.7	<i>Case studies</i>	380
7.15.7.1	Intermediate network	384
7.15.7.2	Steady-state simple brain with circle of Willis.....	385
7.15.7.3	Steady-state patient	386
7.15.8	<i>Dynamic inversion case studies</i>	387
7.15.8.1	Simple tube.....	387
7.15.8.2	Time-Dependent Patient	393
7.15.8.3	Optimization from measurements that are out of phase	396
7.15.9	<i>Findings</i>	398
7.15.10	<i>Fourier filtering and Gibbs phenomena</i>	398
7.15.11	<i>Considering phase shifts</i>	400
7.15.12	<i>Hand calculated symbolic validation:</i>	405
7.15.12.1	Manual Implementation	410
7.15.13	<i>Case study for frequency independence in the Fourier domain representation of the optimization problem</i>	410
7.16	APPENDIX P: VALIDATION OF MATRIX IDENTITIES	413
7.17	APPENDIX Q: DISCRETE FOURIER SERIES.....	416
7.17.1	<i>Discrete Fourier Transform (DFT) - complex arithmetic</i>	417
7.17.1.1	Truncation of Order N.....	417

Table of Contents (continued)

7.17.1.2	DFT in Matrix, Component and Index Format	418
7.17.2	<i>Inverse DFT (IDFT) in Matrix, Component and Index Form</i>	419
7.17.2.1	Reconstruction.....	421
7.17.2.2	Illustration of the Principal Roots	422
7.17.3	<i>Applications</i>	425
7.17.3.1	Reconstruction and noise removal for a heartbeat signal	427
7.17.4	<i>Special topic: different indexing scheme of the DFT with implementation</i>	430
7.17.4.1	Quarteroni implementation	433
7.17.4.2	Matlab/Trefethen implementation	433
7.17.4.3	Conventional implementation	434
7.17.5	<i>Discrete Fourier Transform (DFT) – real arithmetic</i>	439
7.17.6	<i>Matlab implementation of DFT and iDFT</i>	443
7.17.7	<i>Matlab implementation of case study in section 31.3.1 with different methods for solving for even coefficients</i>	445
7.17.8	<i>Matlab implementation of case study in section 31.3.2</i>	447
7.17.9	<i>Matlab implementation of Figure 8.145</i>	448
7.17.10	<i>Implementation notes</i>	450
7.17.11	<i>Non-Uniform Sampling Rates</i>	451
7.17.12	<i>Code for generating and sampling wavelengths of different frequencies</i>	453
7.17.13	<i>evaluation of the DFT matrices using real algebra</i>	456
7.18	APPENDIX R: COPYRIGHT PERMISSIONS FOR PREVIOUSLY PUBLISHED MATERIAL	457
7.19	APPENDIX S: VALIDATING OXYGEN MODEL GENERATION	460
7.19.1	<i>Introduction</i>	461
7.19.2	<i>Linear flow</i>	462
7.19.3	<i>Linear flow + convection</i>	463
7.19.4	<i>Linear flow + convection + rxn</i>	465
7.19.5	<i>Linear flow + convection + tissue diffusion (no reaction)</i>	467
7.19.6	<i>Linear flow + convection + tissue diffusion + vascular reaction</i>	469
7.19.7	<i>Linear flow + convection + tissue diffusion + vascular rxn(0th) + tissue rxn(0th)</i>	471
7.19.8	<i>Flow + convection + tissue diffusion + tissue rxn(0th) + MT (Dirichlet BC)</i>	473
7.19.9	<i>Tissue diffusion + Neumann BC</i>	475
7.19.10	<i>Tissue diffusion + Neumann BC + rxn(0th)</i>	476
7.19.11	<i>Flow + convection + tissue diffusion + tissue rxn(0th) + MT (Neumann BC)</i>	477
7.19.12	<i>Flow + convection + tissue diffusion + tissue rxn(1st) + MT (Neumann BC)</i>	479
7.19.13	<i>Conclusion</i>	481
7.20	APPENDIX T: PETSC IMPLEMENTATION AND VALIDATION	482
7.20.1	<i>Summary</i>	482
7.20.2	<i>Introduction</i>	484
7.20.3	<i>Working precision of a matrix</i>	485
7.20.4	<i>Using BCGS and GMRES</i>	487
7.20.4.1	Solver Validation.....	487
7.20.4.2	Validating Binary reading/writing.....	490
7.20.4.3	Validating Using Initialization Vector	494
7.20.4.4	Solving Networks using ma48 Solution to Initialize and Converge Diverging Solvers.....	497
7.20.5	<i>Simple diagonal matrix conditioning</i>	500
7.20.5.1	Matrix Conditioning.....	501
7.20.6	<i>PETSc block Jacobi preconditioner</i>	505
7.20.7	<i>Time to solve networks</i>	509
7.20.8	<i>Using a two-step solving method for lowering residuals</i>	510
7.20.9	<i>Solving Meshes</i>	512
7.20.9.1	Diffusion Problem with Coarse-Grid Interpolation	512
7.20.10	<i>Large meshes</i>	513
7.20.11	<i>Solving Dual-mesh problems</i>	514
7.20.11.1	Mass Transfer Problem with Coarse-Grid Interpolation	514

Table of Contents (continued)

7.20.11.2	Splines	516
7.20.11.3	Using connection vs boundary conditions.....	516
7.20.11.4	Investigation of units on solver	517
7.20.12	Testing Different Solvers.....	518
7.21	APPENDIX U: CARTESIAN MESH.....	519
7.21.1	Creating a Cartesian mesh	519
7.21.2	Creating a Cartesian mesh in memory	521
7.21.3	Making flux balances without mesh data structures	521
7.21.4	Connecting a 2-point network with a mesh	522
7.21.5	Splitting the network to match length scale of mesh (vascular re-segmentation).....	523
7.21.6	Benefits from using mesh without data structures	525
7.21.7	Validation of mesh from memory compared to dense mesh	526
7.21.8	Interpolating a coarse-mesh guess for initializing solvers for fine-grid results	526
7.22	APPENDIX V: ILLUSTRATION OF DIFFUSION	529
7.23	APPENDIX W: STARLING LAW AND REVISED STARLING LAW	531
7.23.1	Summary	531
7.23.2	Introduction.....	532
7.23.3	Methods	533
7.23.4	Results	536
7.23.4.1	Considerations of the basolateral concentrations:.....	541
7.23.4.2	Parametric study of water and solute fluxes for different hydrostatic and osmotic pressure gradients 542	
7.23.5	Discussion:.....	544
7.23.6	Sample Matlab codes.....	546
7.23.7	Derivation of the Patlack Equation from conservation	552
7.23.8	Deriving revised Starling law from Patlack equation	557
7.23.9	Analytic solution to linear ODE of 2 nd order	559
7.23.10	Example problem.....	562
7.23.10.1	Solution of diffusion/convection through a membrane.....	567
7.23.10.2	Solutions Prepared by Grant Hartung, 9/2018.....	572
7.23.10.3	Code Listing	577
7.24	APPENDIX X: NEWTON METHOD FOR SOLVING NONLINEAR EQUATIONS	580
7.24.1	Theory.....	580
7.24.2	Implementation	582
7.24.2.1	Stepsize control	583
7.24.3	Case Study 1 - Two Nonlinear Equations:.....	585
7.24.4	Case Study 2, 2 nonlinear equations	588
7.24.5	Matlab N-dimensional Newton code.....	590
7.24.6	Derivation of Armijo linesearch optimal stepsize	593
7.25	APPENDIX Y: PROPERTIES OF LINEAR ALGEBRAIC SET OF EQUATIONS.....	597
7.25.1	Residual	597
7.25.2	Gradient	599
7.25.3	Energy.....	601
7.26	APPENDIX Z: IDEALIZED BRAIN GEOMETRY	603
7.27	APPENDIX AA: VALIDATION OF SPHERICAL GEOMETRY FOR DIFFUSION REACTION SYSTEM	608
7.27.1	1D symmetric simulation with a range of production rates and diffusivity rates	609
7.27.1.1	Consistency test of the production rate	609
7.27.2	Validation	612
7.27.2.1	Finite Volume Method compared to Finite Element Analysis	614
7.27.3	Mesh Independence	617
7.27.4	Material properties	618
7.27.5	The validation of flux balance	621
7.28	APPENDIX AB: OXYGEN IN THE BRAIN	623
7.28.1	Introduction.....	623

Table of Contents (continued)

7.28.2	<i>Problem formulation</i>	625
7.28.3	<i>Boundary conditions and material properties</i>	626
7.28.4	<i>Numerical Implementation</i>	628
7.28.4.1	Solving a 3D block using successive 2D sweeps for initialization routine	629
7.28.5	<i>1D problem</i>	629
7.28.5.1	1D diffusion only	630
7.28.5.2	1D problem with reaction and mass transfer	637
7.28.6	<i>2D problem</i>	647
7.28.6.1	Diffusion in cylindrical coordinates	647
7.28.6.2	Problem formulation	648
7.28.6.3	Implementation	653
7.28.7	<i>3D problem</i>	662
7.28.7.1	3D blood flow (Darcy flow) and 3D convection-diffusion-rxn	664
7.28.7.2	1D blood flow projected to 3D mesh	673
7.28.7.3	1D blood flow, 1D convection, 3D diffusion-rxn, mass transfer	674
7.29	APPENDIX AC: MODELS OF AGING BRAIN	680
7.29.1	<i>Physiological background to aging brain</i>	681
7.29.2	<i>Implementation</i>	685
7.30	APPENDIX AD: PROPOSAL OF A FUNCTIONAL HYPEREMIA MODEL	687
7.30.1	<i>Summary</i>	687
7.30.2	<i>Strategy</i>	691
7.30.3	<i>Implementation</i>	693
7.30.4	<i>Dynamically solving a steady problem</i>	693
7.30.4.1	Validation	693
7.30.5	<i>Dynamic solving of constant NO generation, vasculature unchanged</i>	695
7.30.6	<i>Dynamic solving of 0.04s impulse, 1 second simulation, NO generation, vasculature unchanged</i>	696
7.30.7	<i>Dynamic solving of impulse NO generation, vasculature expansion</i>	699
7.30.8	<i>Dynamic solving of impulse NO generation, vasculature signaling</i>	702
7.30.9	<i>Dynamic biphasic blood flow</i>	703
7.31	APPENDIX AE: TIME INTEGRATION SOLVERS	706
7.31.1	<i>Introduction</i>	706
7.31.2	<i>Methods</i>	707
7.31.3	<i>Case study 1: cyclic reaction system</i>	708
7.31.4	<i>Matlab implementation of different solvers on the case study</i>	712
7.32	APPENDIX AF: EDGE DETECTION IN A CARTESIAN MESH	718
7.32.1	<i>Vessel edge detection in 2D Cartesian mesh</i>	718
7.32.2	<i>Validating implementation with diffusion</i>	721
7.32.3	<i>Identifying vessel edge in 3D</i>	723
7.32.3.1	Case Studies	724
7.32.3.2	Simulation case studies	727
7.33	APPENDIX AG: PARAMETRIC STUDIES (PARAMETER SENSITIVITY) ON 1D-3D COUPLING	729
7.33.1	<i>Mass Transfer</i>	730
7.33.2	<i>Reaction (kk)</i>	732
7.33.3	<i>Diffusivity (DD)</i>	734
7.33.4	<i>Volumes</i>	736
7.34	APPENDIX AH: PHYSICAL CHEMISTRY BEHIND OXYGEN COMPOSITION IN BRAIN:	737
7.34.1	<i>Background</i>	737
7.34.2	<i>Unit conversion between concentration and partial pressure</i>	738
7.34.3	<i>Conversion of CMRO₂ from medical units to moles/s</i>	743
7.35	APPENDIX AI: CALCULATING FREE OXYGEN CONCENTRATION HEMOGLOBIN BINDING KINETICS	745
7.36	APPENDIX AJ: THE HILL EQUATION	749
7.36.1	<i>Effects of catalyzed oxygen unbinding on Hill equation</i>	750
7.37	APPENDIX AK: MODELS OF THE NEUROVASCULAR UNIT	754

Table of Contents (continued)

7.37.1	<i>Idealized networks</i>	754
7.37.2	<i>Realistic vasculature</i>	763
7.37.3	<i>Further information: building of the Greens functional parts in full</i>	780
7.38	APPENDIX AL: MODELS FOR SYNTHESIZING VASCULAR NETWORKS	797
7.39	APPENDIX AM: DATA INVENTORY	803
7.39.1	<i>Empirical Kleinfeld networks</i>	803
7.39.2	<i>Generation 1, biphasic blood flow paper data</i>	804
7.39.3	<i>Synthetic Kleinfeld second generation, 100-series, paper 2 (growth paper 1)</i>	809
7.39.4	<i>Empirical Boas</i>	818
7.39.5	<i>Synthetic Boas second generation, 100-series</i>	819
7.39.6	<i>Empirical Dunn</i>	820
7.39.7	<i>Synthetic Dunn second generation, 100-series</i>	821
7.39.8	<i>Mouse 1 (Sled reconstruction)</i>	822
7.39.9	<i>Mouse 2 hemispheres (reconstructed from Allen Brain Institute images)</i>	825
7.39.10	<i>Horowitz heart reconstructions</i>	827
7.39.11	<i>Human pial growth</i>	829
8	CITED LITERATURE	836
9	VITA	853

LIST OF TABLES

Number	Title	Page
TABLE 2.1.	REFERENCE TABLE FOR THE SUPPLEMENTAL DATA IN THIS DOCUMENT	7
TABLE 3.1.	NOMENCLATURE	13
TABLE 3.2.	PHYSIOLOGICAL PARAMETERS NECESSARY FOR ANATOMICAL GROWTH IN MOUSE.....	28
TABLE 3.3.	STATISTICS FOR FOUR LARGE SYNTHETIC ANATOMICAL NETWORK STRUCTURES, SPECIFICALLY TWO MOUSE MCA TERRITORIES (SMCA1.101, SMCA2.101) AND TWO COMPLETE HEMISPHERES (SH1.101, SH2.101).....	35
TABLE 3.4.	PSEUDOCODE FOR MICROVASCULAR CLOSURE ALGORITHM	42
TABLE 3.5.	PSEUDOCODE FOR GENERATING SAMPLE POINTS IN A TRIANGULAR MESH	42
TABLE 3.6.	PSEUDOCODE FOR GENERATING SAMPLE POINTS IN A TRIANGULAR MESH	42
TABLE 3.7.	PSEUDOCODE FOR GENERATING SAMPLE POINTS INSIDE A CARTESIAN BOUNDING BOX	42
TABLE 4.1.	PSEUDOCODE FOR GROUPING FACES FROM A 2-POINT NETWORK BETWEEN BIFURCATIONS	48
TABLE 4.2.	PSEUDOCODE FOR CONVERTING A 2-POINT NETWORK TO SPLINED NETWORK.....	48
TABLE 4.3.	PSEUDOCODE FOR CALCULATING A CDF FROM A SORTED VALUE ARRAY	50
TABLE 5.1.	SUMMARY OF BOUNDARY CONDITIONS	73
TABLE 5.2.	TOPOLOGICAL FEATURE COMPARISON BETWEEN EXPERIMENTAL 2PLSM AND SYNTHETIC DATA SETS	74
TABLE 5.3.	PIAL NETWORK PARAMETERS USED IN THIS WORK IN COMPARISON TO PRIOR RESEARCH	87
TABLE 5.4.	PHYSIOLOGICAL ASSESSMENT OF PLASMA SKIMMING MODELS.....	103
TABLE 6.1.	PARAMETERS USED IN THE PREDICTION OF OXYGEN TENSION THROUGHOUT THE MURINE CORTEX	112
TABLE 8.1.	PSEUDOCODE FOR THE SEGMENT ADDITION WITH VOLUME OPTIMIZATION	134
TABLE 8.2.	COMPARISON FOR TOPOLOGICAL VALUES OF A BIFURCATION BEFORE AND AFTER OPTIMIZATION	135
TABLE 8.3.	TOPOLOGICAL VALUES OF A BIFURCATION AFTER TWO DIFFERENT DIAMETER ASSIGNMENTS	135
TABLE 8.4.	COMPARISON FOR TOPOLOGICAL VALUES OF A BIFURCATION BEFORE AND AFTER OPTIMIZATION	136
TABLE 8.5.	TOPOLOGICAL VALUES OF A BIFURCATION AFTER TWO DIFFERENT DIAMETER ASSIGNMENTS	137
TABLE 8.6.	PSEUDOCODE FOR STAGED GROWTH.....	139
TABLE 8.7.	PSEUDOCODE FOR GROWING PENETRATING VESSELS	145
TABLE 8.8.	PSEUDOCODE FOR GENERATING A SAMPLE POINT INSIDE A TRIANGULAR PRISM.....	145
TABLE 8.9.	PSEUDOCODE FOR TETRAHEDRALSAMPLEGENERATOR1	160
TABLE 8.10.	PSEUDOCODE FOR TETRAHEDRALSAMPLEGENERATOR2	162
TABLE 8.11.	PSEUDOCODE FOR CALCULATING PERPENDICULAR DISTANCE FROM A TRIANGLE	165
TABLE 8.12.	PSEUDOCODE FOR ADDING TORTUOSITY TO A GROUP OF SPLINES.....	189
TABLE 8.13.	PSEUDOCODE FOR MAKING A SPLINE LONGER	190
TABLE 8.14.	SUMMARY OF FRICTIONAL LOSSES DUE TO TURBULENT FLOW IN A TUBE	267
TABLE 8.15.	SEGMENT RESISTANCE FOR SIMULATING THE NETWORK IN FIGURE 8.69	268
TABLE 8.16.	INVENTORY OF EQUATIONS USED FOR SIMULATING THE NETWORK IN FIGURE 8.69	268
TABLE 8.17.	HAGEN POISEUILLE LAW APPLIED ON EACH FACE IN THE CoW	271
TABLE 8.18.	FLOW BALANCES AND DIRICHLET BOUNDARY FOR EACH POINT IN THE CoW.....	272
TABLE 8.19.	RESISTANCE VECTOR USED IN FLOW COMPUTATIONS FOR THE CoW (MMHG MIN/ML)	272
TABLE 8.20.	RESULTS OF SOLVING LINEAR FLOW EQUATIONS FOR SIMPLIFIED CIRCLE OF WILLIS MODEL	274
TABLE 8.21.	SIMULATION RESULTS FOR AN OCCLUSION USING DIFFERENT PROBLEM FORMULATIONS	275
TABLE 8.22.	SIMULATION RESULTS FOR A VERY SHORT VESSEL (RESISTANCE = 0) USING DIFFERENT PROBLEM FORMULATIONS	276
TABLE 8.23.	OVERVIEW VISCOSITY MODELS	281
TABLE 8.24.	TABLE OF VARIABLES FOR PRIES'S IN VITRO VISCOSITY MODELS	282
TABLE 8.25.	VARIABLES FOR PRIES'S IN VIVO VISCOSITY MODEL.....	286
TABLE 8.26.	VARIABLES FOR CHARM AND KURLAND'S VISCOSITY MODEL	288
TABLE 8.27.	VARIABLES FOR KIANI AND HUDETZ'S VISCOSITY MODEL	289
TABLE 8.28.	COEFFICIENTS USED FOR PRIES'S PLASMA SKIMMING MODEL	294
TABLE 8.29.	VARIABLES USED IN PLOURABOUÉ'S PLASMA SKIMMING MODEL.....	299
TABLE 8.30.	STATISTICAL ANALYSIS OF NETWORKS USED FOR BIPHASIC BLOOD FLOW SIMULATIONS	305
TABLE 8.31.	DATA FOR LINEAR FIRST ORDER FITTING PROBLEM.....	325

TABLE 8.32. DATA FOR LINEAR SECOND ORDER FITTING PROBLEM.....	327
TABLE 8.33. DATA FOR LINEAR FITTING PROBLEM IN 3D	328
TABLE 8.34: OPTIMIZED VALUES FOR TERMINAL PRESSURES AND OVERALL MASS CONSERVATION	385
TABLE 8.35: MEASURED VALUES (f) FROM NOVA REPORT	394
TABLE 8.36. DFT FOURIER TRANSFORMATION MATRIX T AND IDFT TRANSFORMATION MATRIX C FOR 2, 4, AND 6 DATA POINTS.	424
TABLE 8.37. FOURIER POLYNOMIAL AND ITS COEFFICIENTS TO RECONSTRUCT SIGNAL WITHOUT NOISE USING QUATERONI INDEXING (N=84).....	428
TABLE 8.38. COMPARISON OF INDEXING SCHEMES. QUATERONI ($k-N/2$), TREFETHEN (k), AND MATLAB ($k-1$), N=4.....	435
TABLE 8.39. DFT FOURIER TRANSFORMATION MATRIX T AND IDFT TRANSFORMATION MATRIX C FOR 2, 4, AND 6 DATA POINTS USING TREFETHEN INDEXING.....	439
TABLE 8.40. COEFFICIENT C AND T MATRICES FOR THE DFT AND IDFT USING REAL ALGEBRA.....	443
TABLE 8.41: SUGGESTED PARAMETERS IN HUMAN FOR THE CALCULATIONS OF THE RSL	535
TABLE 8.42: PARAMETERS FOR CALCULATION BASED ON RAT KIDNEY. HERE, Π_P STANDS FOR Π_C	536
TABLE 8.43: PARAMETERS USED FOR MY RECREATION OF THE CURVES FROM THE MANUSCRIPT.	536
TABLE 8.44: PARAMETERS USED FOR MY RECREATION OF THE CURVES FROM THE MANUSCRIPT.	541
TABLE 8.45: VALUES USED FOR THE COMPUTATION OF SCENARIO 1-5 IN FIGURE 8.166.....	542
TABLE 8.46: RESULTING VALUES COMPUTED USING EQUATION (8.379) AND (8.382) USING VALUES IN TABLE 8.45.	543
TABLE 8.47. THIS SECTION DETAILS A DERIVATION OF THE REVISED STARLING LAW.	553
TABLE 8.48: VARIABLES USED FOR SECTION 8.23.3	553
TABLE 8.49: PARAMETERS USED IN THIS CASE STUDY, EMPIRICALLY DERIVED	565
TABLE 8.50. SOLUTION FOR j_w AND Pe USING NEWTON METHOD	575
TABLE 8.51: THE GIVEN VALUES USED TO PREDICT THE GENERATION RATE OF THE PROTEIN	605
TABLE 8.52. THE GIVEN VALUES USED TO PREDICT THE GENERATION RATE OF THE PROTEIN	609
TABLE 8.53. CORRESPONDING DIFFUSIVITIES TO THE PLOTTED VARIABLES	611
TABLE 8.54. DIFFERENTIAL PRODUCTION RATES (R_A) USED IN BIOMETRIC STUDY. (MOL/CM ³ /s)	611
TABLE 8.55. COMPARATIVE ERROR AND COMPUTATION TIME BETWEEN DIFFERENT DISCRETIZATION METHODS.....	618
TABLE 8.56. SIMULATION PARAMETERS NECESSARY FOR QUANTIFYING THE CONCENTRATION PROFILE.....	619
TABLE 8.57. DATA CONVERSION FROM ORIGINAL UNITS TO STANDARDIZED UNITS AND SIMULATED UNITS. RED INDICATES A SECOND VALUE FOR MINIMUM VALUE.	620
TABLE 8.58. PARAMETER AND BOUNDARY CONDITION CHOICES FOR THIS MODEL.....	628
TABLE 8.59. TIME LAPSE FOR SOLVING A 3D PROBLEM WITH SUCCESSIVE 2D SWEEPS.....	629
TABLE 8.60. VALUES USED TO GRAPHICALLY EVALUATE THE ANALYTIC SOLUTION TO DIFFUSION EQUATION IN CYLINDRICAL COORDINATES.	652
TABLE 8.61. RESULTING FLOWS AT THE INITIAL AND TERMINAL SEGMENTS WHEN VARYING THE VESSEL RADIUS (A).....	655
TABLE 8.62. RESULTING FLOWS AT THE INITIAL AND TERMINAL SEGMENTS WHEN VARYING THE VESSEL RADIUS (A).....	656
TABLE 8.63. COMPARISON OF VASCULAR TOPOLOGICAL CHANGES OBSERVED IN THE AGED BRAIN	681
TABLE 8.64. COMPARATIVE ERROR BETWEEN NUMERICAL INTEGRATORS.	711
TABLE 8.65. COMPARATIVE PERFORMANCE BETWEEN NUMERICAL INTEGRATORS.....	711
TABLE 8.66. PSEUDOCODE FOR VASCULAR LABELING OF 2D CARTESIAN MESH	719
TABLE 8.67. PSEUDOCODE FOR GETTING CELL INDEX SURROUNDING A POINT IN 2D CARTESIAN MESH	719
TABLE 8.68. MOLECULAR PROPERTIES OF OXYGEN AND WATER	739

LIST OF FIGURES

FIGURE 3.1. OVERVIEW OF CCO METHOD BY SCHREINER AND KARCH STARTING WITH ONE SEGMENT.	13
FIGURE 3.2. RECURSIVE TREE RESISTANCE COMPUTATIONS.	19
FIGURE 3.3. SYNTHESIS OF OPEN ARTERIAL STRUCTURES WITH SAMPLE GENERATORS.	27
FIGURE 3.4. A COLLECTION OF SYNTHETIC MICROCIRCULATORY NETWORKS OF THE SOMATOSENSORY CORTEX IN MOUSE.	29
FIGURE 3.5. FOUR STAGES OF THE PIAL LEPTOMENEGIAL ARTERIAL NETWORK GROWTH IN MOUSE.	31
FIGURE 3.6. COMPARISON OF GROWTHS WITH VASCULAR ATLAS.	33
FIGURE 3.7. ILLUSTRATION OF THE PIAL VASCULAR GROWTH FOR THE MOUSE HEMISPHERE.	34
FIGURE 3.8. NETWORK STRUCTURE AND SELECT SIMULATION RESULTS FROM BIPHASIC BLOOD FLOW FOR THE COMPLETE CEREBRAL HEMISPHERE IN MOUSE.	37
FIGURE 4.1. WORKFLOW DIAGRAM FOR MATCHING THE LENGTH AND TORTUOSITY SPECTRA OF A SYNTHETIC NETWORK TO EMPIRICAL NETWORK.	45
FIGURE 4.2. COMPARISON BETWEEN EMPIRICAL AND SYNTHETIC NETWORKS BEFORE AND AFTER ADDING TORTUOSITY.	46
FIGURE 4.3. A DIAGRAM EXEMPLIFYING TWO CASES WHERE A LINE IS SEGMENTED INTO AN ARBITRARY NUMBER OF SECTIONS.	47
FIGURE 4.4. GRAPHICAL REPRESENTATION OF THE TORTUOSITY OF A LINE.	51
FIGURE 4.5. TWO CUMULATIVE DENSITY FUNCTIONS THAT DO NOT MATCH.	53
FIGURE 4.6. CDF MATCHING BETWEEN TWO DATASETS ARE NOT THE SAME NUMBER OF VALUES.	55
FIGURE 4.7. TOPOLOGICAL COMPARISON BETWEEN 1 NETWORK FROM THE DUNN GROUP, 5 NETWORKS FROM THE BOAS GROUP AND 4 NETWORKS FROM THE KLEINFELD GROUP.	57
FIGURE 4.8. PROBABILITY DENSITY FUNCTIONS OF THE TORTUOSITY AS A FUNCTION OF NEUROANAL LAYER IN THE EMPIRICALLY-DERIVED NETWORKS OF BOAS, KLEINFELD, AND DUNN.	57
FIGURE 4.9. COMPARISON BETWEEN EMPIRICAL AND SYNTHETIC KLEINFELD NETWORKS.	58
FIGURE 4.10. COMPARISON BETWEEN SYNTHETIC AND EMPIRICAL BOAS NETWORKS.	59
FIGURE 4.11. COMPARISON BETWEEN SYNTHETIC AND EMPIRICAL DUNN NETWORKS.	59
FIGURE 4.12. WORKFLOW DIAGRAM OF THE ANATOMICAL GROWTH ALGORITHM.	61
FIGURE 4.13. NETWORK STRUCTURE AND SELECT SIMULATION RESULTS FROM BIPHASIC BLOOD FLOW FOR THE COMPLETE CEREBRAL HEMISPHERE IN MOUSE.	62
FIGURE 4.14. THE PENETRATING ARTERIOLES APPLIED ACROSS AN ENTIRE REGION OF THE HUMAN BRAIN.	63
FIGURE 4.15. FURTHER VISUALIZATIONS OF CAPILLARY SYNTHESIS IN HUMAN CORTEX.	64
FIGURE 5.1. MULTI-MODAL IMAGING DATA USED TO CONSTRUCT REALISTIC MODELS OF CEREBRAL CIRCULATION FOR ENTIRE MOUSE BRAIN.	70
FIGURE 5.2. MORPHOMETRIC COMPARISON BETWEEN EXPERIMENTAL AND SYNTHETIC MICROCIRCULATORY NETWORKS FROM THE MURINE VIBRISSA PRIMARY SENSORY CORTEX.	77
FIGURE 5.3. PREDICTIONS OF HEMODYNAMIC STATES IN PRIMARY CORTEX SIMULATIONS SHOW LARGE VARIATIONS DUE TO NETWORK ARCHITECTURE.	79
FIGURE 5.4. DEPTH DEPENDENT PATH ANALYSIS OF HEMATOCRIT TRAJECTORIES THROUGH THE CORTEX.	81
FIGURE 5.5. STATISTICS OF HEMATOCRIT DISTRIBUTION AND RBC FLUXES IN CORTICAL LAYERS OF CEREBRAL MICROCIRCULATORY NETWORKS.	82
FIGURE 5.6. SCHEMATIC OF MULTISCALE BIPHASIC BLOOD FLOW SIMULATIONS IN THE ARTERIAL SIDE OF THE MCA TERRITORY.	85
FIGURE 5.7. BLOOD FLOW OF THE COMPLETE ARTERIAL AND VENOUS CIRCULATION FOR THE MCA TERRITORY IN MOUSE.	88
FIGURE 5.8. DEPTH DEPENDENCE OF HEMATOCRIT ON TOTAL BLOOD FLOW IN THE LEFT MCA TERRITORY.	89
FIGURE 5.9. SCHEMATIC OF THE DEPTH DEPENDENT HEMATOCRIT NETWORK EFFECT.	94
FIGURE 5.10. HEMATOCRIT FIELDS PORTRAYED ON TWO SIMPLIFIED MICROCIRCULATORY MODELS.	101
FIGURE 5.11. FLOW DIAGRAM FOR MAIN STEPS IN THE FIXED POINT ITERATION FOR BIPHASIC BLOOD FLOW COMPUTATIONS.	105
FIGURE 5.12. DIAMETER DEPENDENCE OF HEMATOCRIT IN MICROCIRCULATORY NETWORKS.	106
FIGURE 6.1. VASCULAR MASKING OF THE CARTESIAN MESH AND VESSEL SEGMENTATION.	114
FIGURE 6.2. MATRIX STRUCTURE OF DIFFUSION WITH DIFFERENT MESH TYPES.	117
FIGURE 6.3. EXAMPLE LABELING OF A CARTESIAN MESH AT DIFFERENT DENSITY LEVELS.	118
FIGURE 6.4. VALIDATION OF SIMULATION PARADIGM AGAINST THREE SETS OF EXPERIMENTAL DATA.	119
FIGURE 6.5. DETAILED ANALYSIS OF EXTRAVASCULAR SPACE IN AN OXYGEN SIMULATION FOR A SINGLE EMPIRICAL NETWORK.	120
FIGURE 6.6. DETAILED ANALYSIS OF EXTRAVASCULAR SPACE IN AN OXYGEN SIMULATION FOR A SECOND EMPIRICAL NETWORK.	121
FIGURE 6.7. OXYGEN PREDICTIONS THROUGHOUT THE EXTRAVASCULAR SPACE.	122

FIGURE 6.8. OXYGEN PREDICTIONS THROUGHOUT THE EXTRAVASCULAR SPACE IN A SECOND CASE.	123
FIGURE 6.9. EXAMPLES OF OXYGEN TENSION AT DIFFERENT CORTICAL DEPTHS IN YOUNG AND AGED MOUSE.	125
FIGURE 6.10. COMPARISON BETWEEN OXYGEN TENSION IN YOUNG AND AGED MOUSE.	126
FIGURE 6.11. SCHEMATIC DIAGRAM OF A CYLINDER.	129
FIGURE 8.1. WORKFLOW DIAGRAM FOR GROWING A SINGLE SEGMENT.	134
FIGURE 8.2. OPTIMIZATION SPACE FOR BIFURCATION LOCATION.	136
FIGURE 8.3. OPTIMIZATION SPACE FOR BIFURCATION LOCATION.	137
FIGURE 8.4. WORKFLOW DIAGRAM OF THE ANATOMICAL GROWTH ALGORITHM.	138
FIGURE 8.5. WORKFLOW DIAGRAM FOR PIAL SURFACE GROWTH.	140
FIGURE 8.6. PIAL GROWTH ALGORITHM AT DIFFERENT STAGES OF GROWTH.	141
FIGURE 8.7. UNITY IMAGES OF SLED MOUSE. IMAGES OF AN ARTERIAL TREE, VENOUS TREE, AND THE SLED MOUSE SURFACE MESH LOADED IN UNITY.	141
FIGURE 8.8. EXPLANATORY DIAGRAM OF THE PROJECTION OF A POINT TO THE SURFACE MESH.	142
FIGURE 8.9. THE GRAPHICAL REPRESENTATION OF A TRIANGLE WITH REFERENCE TO A NEW POINT (P_N).	143
FIGURE 8.10. (A) WORKFLOW DIAGRAM FOR GROWING PENETRATING VESSELS AND (B-C) EXAMPLE VIEWS OF ASCENDING VEINS IN AN MCA TERRITORY.	146
FIGURE 8.11. WORKFLOW DIAGRAM FOR CAPILLARY GROWTH.	147
FIGURE 8.12. DEMONSTRATION OF ONE PASS OF THE CLOSURE ALGORITHM.	149
FIGURE 8.13. WORKFLOW DIAGRAM FOR CLOSURE GROWTH.	150
FIGURE 8.14. ANATOMICAL GROWTH OF A KLEINFELD-LIKE MICROCIRCULATORY NETWORK.	150
FIGURE 8.15. VALIDATION OF SAMPLE PLANAR SAMPLE GENERATOR.	154
FIGURE 8.16. VALIDATION OF DEPTH SAMPLE GENERATOR.	154
FIGURE 8.17. THE GRAPHICAL REPRESENTATION OF A TRIANGLE.	156
FIGURE 8.18. VISUALIZATION OF THE MIRRORING EFFECT WHEN A SAMPLE IS THROWN OUTSIDE A TRIANGLE.	157
FIGURE 8.19. DIFFERENT TRIANGLE SAMPLE GENERATOR BIAS EVALUATIONS.	158
FIGURE 8.20. TRIANGULAR PRISM USED IN ANATOMICAL GROWTH.	159
FIGURE 8.21. IN THIS CASE, THE PROJECTION INTO THE DOMAIN IS VALIDATED.	161
FIGURE 8.22. VISUALIZATION OF HOW THREE RANDOM VARIABLES CREATE A VECTOR OUTSIDE A TETRAHEDRON.	161
FIGURE 8.23. THE SECOND METHOD FOR TETRAHEDRAL SAMPLING.	163
FIGURE 8.24. THE SECOND METHOD FOR TETRAHEDRAL SAMPLING.	164
FIGURE 8.25. GRAPHICAL REPRESENTATION OF CALCULATING DEPTH FROM A SURFACE TRIANGLE.	165
FIGURE 8.26. THE SOLUTION OPF CASE STUDY 1 IS VALIDATED BY BOTH MATLAB AND DELPHI AT $\gamma=0.87$	167
FIGURE 8.27. SOLUTION TO CASE STUDY 2 IS $\gamma=2.165$ AS VERIFIED BY MATLAB AND DELPHI.	168
FIGURE 8.28. CASE STUDY 3 DOES NOT HAVE A SOLUTION. IN THIS CASE, THE DOUBLE-ROOT OCCURS WITH A VERY LARGE RESIDUAL VALUE, MEANING IT IS NOT A SOLUTION.	169
FIGURE 8.29. HISTOGRAMS OF THE MURRAY COEFFICIENT FOR THE 20 MICRON PIAL SURFACE, SOLVED FOR USING NEWTON METHOD.	170
FIGURE 8.30. DATASET PREDICTIONS OF PARENT DIAMETER (WITH VARYING COEFFICIENT, γ) COMPARED TO EXPERIMENTAL PARENT DIAMETER.	170
FIGURE 8.31. HISTOGRAMS OF THE MURRAY COEFFICIENT FOR THE 7 MICRON PIAL SURFACE, SOLVED FOR USING NEWTON METHOD.	171
FIGURE 8.32. MICRON DATASET PREDICTIONS OF PARENT DIAMETER (WITH VARYING COEFFICIENT, γ) COMPARED TO EXPERIMENTAL PARENT DIAMETER.	171
FIGURE 8.33. THE COMPARISON OF THE 4 EMPIRICAL DATASETS AND THEIR RESPECTIVE MURRAY COEFFICIENT AS DERIVED USING THE NEWTON METHOD.	172
FIGURE 8.34. THE COMPARISON OF THE FIRST 2 EMPIRICAL DATASETS AND THEIR RESPECTIVE MURRAY COEFFICIENT AS DERIVED USING THE NEWTON METHOD.	173
FIGURE 8.35. THE COMPARISON OF THE LAST 2 EMPIRICAL DATASETS AND THEIR RESPECTIVE MURRAY COEFFICIENT AS DERIVED USING THE NEWTON METHOD.	174
FIGURE 8.36. HISTOGRAMS OF THE MURRAY COEFFICIENT FOR ACCUMULATED 4 EMPIRICAL MICROCIRCULATORY DATASETS.	175
FIGURE 8.37. THE COMPARISON OF THE CUMULATIVE EMPIRICAL DATASETS AND THEIR RESPECTIVE MURRAY COEFFICIENT AS DERIVED USING THE NEWTON METHOD.	175
FIGURE 8.38. VISUALIZATION OF THE AUTOMATICALLY LABELED PENETRATING ARTERIES AND ASCENDING VEINS IN THE EMPIRICAL NETWORKS.	176
FIGURE 8.39. VISUALIZATION OF THE WHOLE E1.1 NETWORK WITH PENETRATING VESSELS HIGHLIGHTED.	177

FIGURE 8.40. VISUALIZATION OF SHORT PENETRATING VESSELS THAT WERE INITIALLY OVERLOOKED BY THE LABELING ALGORITHM.	178
FIGURE 8.41. PICTORAL REPRESENTATION OF THE TWO CLASSIFICATIONS OF PENETRATING VESSELS ON THE ARTERIAL SIDE.	178
FIGURE 8.42. THE GRAPHICAL REPRESENTATION OF A FACE (AS VECTOR V IN BLUE) IN AN X, Y, AND Z COORDINATE SYSTEM.	179
FIGURE 8.43. PROBABILITY DENSITY FUNCTION OF THE VERTICAL ALIGNMENT OF EMPIRICAL NETWORKS.	180
FIGURE 8.44. VISUALIZATION OF VERTICAL ALIGNMENT OF EMPIRICAL NETWORK EKF1.1.	181
FIGURE 8.45. SCHEMATIC REPRESENTATION OF A NEW SAMPLE (P_4) AND THE INTERSECTION POINT WITH THE EXISTING FACE (P_1 - P_3) AT POINT P_2	183
FIGURE 8.46. SIMPLE TEST OF PERPENDICULAR DISTANCE BETWEEN A POINT AND A LINE.	185
FIGURE 8.47. SCHEMATIC REPRESENTATION OF EVALUATION IF NEW BIFURCATION POINT (P_2) LIES ALONG THE VECTOR $\langle P_1 - P_3 \rangle$ BY EVALUATING THE AREA OF THE TRIANGLE MADE BY 3 POINTS $\langle P_1, P_2, P_3 \rangle$	185
FIGURE 8.48. SCHEMATIC REPRESENTATION OF VARIABLES NECESSARY FOR ADDING TORTUOSITY TO A SPLINED SEGMENT.	188
FIGURE 8.49. WORKFLOW DIAGRAM FOR ADDING TORTUOSITY TO A SPLINED NETWORK.	190
FIGURE 8.50. VISUALIZATION OF A CLOSURE SEGMENT THAT CONNECTS ARTERIES TO VEINS THROUGH AN ACUTE ANGLE.	191
FIGURE 8.51. THE CONTROL AND TERMINAL POINTS TO TWO BEZIER CURVES OF 3 RD ORDER LINKED TOGETHER TO MAKE A SPLINED SEGMENT.	192
FIGURE 8.52. THE CONTROL AND TERMINAL POINTS TO TWO BEZIER CURVES OF 3 RD ORDER LINKED TOGETHER TO MAKE A SPLINED SEGMENT.	193
FIGURE 8.53. THE FINISHED PRODUCT OF REDUCING TORTUOSITY CHANGES THE LONG, SMOOTH CURVE INTO THE SHORTER SMOOTH CURVE.	193
FIGURE 8.54. COMPLETELY DIFFERENT NETWORKS (GENERATED WITH SIMILAR ALGORITHMS BUT AT DIFFERENT SCALES AND DIFFERENT DENSITIES) MATCHED WITH 2, 5, 10, AND 100 BINS.	196
FIGURE 8.55. MATCHING CDFs BETWEEN EKF1.1 AND SKF1.101.	197
FIGURE 8.56. VISUALIZATION OF AUTOMATED CDF MATCHING USING CASE STUDY 1.	203
FIGURE 8.57. EXAMPLE OF ATTACHING TWO POINTS ON A HUMAN PIAL SURFACE.	222
FIGURE 8.58. PREDICTION OF MAX AND MINIMUM CURVATURE FOR AN ANALYTIC SURFACE DEFINED BY A SERIES OF POINTS USING A PRE-KNOWN FUNCTION.	234
FIGURE 8.59. PREDICTION OF MAXIMUM AND MINIMUM CURVATURE FOR AN ANALYTIC SURFACE DEFINED BY A SERIES OF POINTS USING A PRE-KNOWN FUNCTION USING OFF-CENTER SAMPLING.	235
FIGURE 8.60. THE ANALYTIC CURVE GIVEN BY EQUATION (8.46) IS RECREATED FROM THE POINT COORDINATES OF A TRIANGULAR SURFACE MESH.	236
FIGURE 8.61. THE NEW COORDINATE SYSTEMS FOR EACH TRIANGLE IN THE MESH AS IT SITS ON THE PLANE. FOR SCALING PURPOSES, THE Z-COORDINATE SHOWN HERE IS THE SQUARE ROOT OF THE Z-COORDINATE DEFINED BY EQUATION (8.46). THE BLUE LINES ON EACH TRIANGLE CORRELATE TO e_n , THE RED LINE IS e_1 , AND THE GREEN LINE IS e_2 . THE BLUE LINES REPRESENT e_n , THE GREEN LINES REPRESENT e_2 , AND THE RED LINES REPRESENT e_1	238
FIGURE 8.62. TRANSLATING THE (LEFT) ORIGINAL STRUCTURE IN CARTESIAN COORDINATES TO (RIGHT) THE COORDINATE SYSTEM DEFINED BY THE BLUE TRIANGLE.	239
FIGURE 8.63. VISUALIZATION OF MANY TYPES OF CURVATURE CALCULATIONS ACROSS A SMALL SECTION OF THE HUMAN CORTICAL SURFACE.	241
FIGURE 8.64. CURVATURE CALCULATIONS ON THE HEMISPHERE REVEAL EXCELLENT IDENTIFICATION OF RIDGES AND VALLEYS.	242
FIGURE 8.65. SCREENSHOT OF THE VMTK SOFTWARE AFTER SUCCESSFUL COMPILING OF THE CODE AND EXECUTION OF THE FIRST LINES.	249
FIGURE 8.66. SCREENSHOT OF THE VMTK SOFTWARE AFTER SUCCESSFUL IDENTIFICATION OF THE BA, LCA, AND RCA AND PLACEMENT OF RED SPHERES ON THESE LOCATIONS.	251
FIGURE 8.67. ANATOMICAL RECONSTRUCTION SHOWING A MOUSE BRAIN IN TWO VIEWPOINTS. RECONSTRUCTION WAS PERFORMED WITH ITK-SNAP AND MESHING/LABELING WAS PERFORMED BY ICEM.	255
FIGURE 8.68: A SIMPLIFIED NETWORK OF NODES (P) AND ARCS (F).	259
FIGURE 8.69: SIMPLIFIED CEREBRAL ANGIOARCHITECTURE (FLOW IS FROM LEFT TO RIGHT)	268
FIGURE 8.70: REPRESENTATIONS OF THE HUMAN CIRCLE OF WILLIS (COW).	270
FIGURE 8.71: FAHREUS AND FAHREUS-LINDQVIST EFFECT ILLUSTRATION.	279
FIGURE 8.72: EXPERIMENTAL DATA FOR CHANGES IN APPARENT VISCOSITY WITH DIAMETER.	280
FIGURE 8.73: APPARENT VISCOSITY OF FOR THE (A,B) PRIES IN-VITRO MODEL AND (C,D) PRIES IN-VITRO MODIFIED MODEL.	283
FIGURE 8.74: APPARENT VISCOSITY FOR PRIES'S IN VIVO MODEL.	286
FIGURE 8.75: RELEVANT DIMENSIONS IN A SINGLE TUBE.	288

FIGURE 8.76: APPARENT VISCOSITY AS PREDICTED BY KIANI AND HUDETZ VISCOSITY MODEL.	290
FIGURE 8.77. PARAMETRIC INVESTIGATION OF VISCOSITY OVER A RANGE OF DIAMETER AND HEMATOCRIT.	291
FIGURE 8.78. PLASMA SKIMMING IN SMALL VESSELS. IN VESSELS WITH SMALL DIAMETER, THE RED BLOOD CELLS AGGREGATE AT THE CENTER OF THE CAPILLARY, CREATING A CELL FREE LAYER.	292
FIGURE 8.79: GRAPHIC REPRESENTATION OF A BASIC BIFURCATION.	292
FIGURE 8.80: VISUALIZATION OF THE M VALUE AS A FUNCTION OF HEMATOCRIT.	303
FIGURE 8.81. THE PREDICTED EFFECTS OF CHANGING DIAMETER AND HEMATOCRIT AS A FUNCTION OF DIFFERENT SPLIT RULES APPLIED TO A SINGLE BIFURCATION.	305
FIGURE 8.82. THE NETWORK EFFECT OF HEMATOCRIT DISTRIBUTION.	306
FIGURE 8.83: COMPARING THE VARIATION IN HEMATOCRIT SPLIT (TOP) AND RBC FLUX FRACTION (BOTTOM) WITH THE FLOW SPLIT.....	307
FIGURE 8.84. GENERAL TRENDS OF VISCOSITY OVER DIAMETER FOR A HEMATOCRIT OF 0.4.....	308
FIGURE 8.85. VISCOSITY COMPARISON BETWEEN PRIES IN VITRO MODIFIED AND PRIES IN VIVO MODELS.	309
FIGURE 8.86. PREDICTED VISCOSITY BY ALL MODELS AT HIGH HEMATOCRIT VALUES.	310
FIGURE 8.87. PREDICTED VISCOSITY BY ALL MODELS FOR DIAMETERS <20MM.....	311
FIGURE 8.88. BRANCH HEMATOCRIT AND FLOW VARIATION AS A FUNCTION OF THE DIAMETER FOR ONE BIFURCATION.	312
FIGURE 8.89: LINEAR FIRST ORDER FITTING BETWEEN DATA POINTS (BLUE CIRCLES) AND LINE OF BEST FIT (ORANGE LINE) USING LINEAR REGRESSION.	326
FIGURE 8.90: SECOND ORDER LINEAR FITTING MODEL (BLUE LINE) AND SOURCE DATA (ORANGE POINTS).....	327
FIGURE 8.91: LINEAR FIT OF A PLANE THROUGH 5 POINTS IN 3D.....	328
FIGURE 8.92. BLOOD PRESSURE AND HEMATOCRIT VISUALIZED IN AN IMMERSIVE 3D ENVIRONMENT FOR QUALITATIVE ANALYSIS OF GLOBAL TRENDS.	331
FIGURE 8.93. ANATOMICAL NETWORK OF A COMPLETE CIRCLE OF WILLIS (LENGTHS NOT TO SCALE) WITHOUT DIAMETER INFORMATION.	333
FIGURE 8.94. A WORKFLOW DIAGRAM FOR THE PATH ANALYSIS.	334
FIGURE 8.95. EXAMPLE OF LOOP IN ARTERIAL TREE.	335
FIGURE 8.96. THE COMPARISON OF LINEAR, LOG, AND POWER LAW BETWEEN THE VALUES OF -1 AND 1 SHOWS A POWER LAW CAN OVERCOME THE ASYMPTOTE IN THE LOG PLOT AT $y < 1$	336
FIGURE 8.97. THE COLORATION OF THE PREVIOUSLY PROPOSED LOG PLOTS SHOWS THE POWER LAW HAS A DISTINCT ADVANTAGE OVER THE LOG SCALING.	337
FIGURE 8.98. VISUALIZATION OF SURFACE FLOW VECTORS.	338
FIGURE 8.99. STEP 1 OF THE SURFACE DIVERGENCE CALCULATING ALGORITHM.....	339
FIGURE 8.100. STEP 2 OF THE SURFACE DIVERGENCE CALCULATING ALGORITHM.....	339
FIGURE 8.101. STEP 3 OF THE SURFACE DIVERGENCE CALCULATING ALGORITHM.....	339
FIGURE 8.102. COLORATION COMPARISON BETWEEN LINEAR AND LOG SCALING FOR PIAL SURFACE DIVERGENCE.....	340
FIGURE 8.103. VISUALIZATION OF THE VOLUME DIVERGENCE IMPLEMENTED OVER A NETWORK AND A 10x10x10 CARTESIAN MESH.	341
FIGURE 8.104: PROBABILITY DENSITY FUNCTION OF FLOW IN LAYER 1 AFTER PIAL VESSELS HAVE BEEN REMOVED.	344
FIGURE 8.105: PROBABILITY DENSITY FUNCTION OF FLOW IN LAYER 1 AFTER PIAL VESSELS HAVE BEEN REMOVED WITH A LOGARITHMIC X- AXIS.....	344
FIGURE 8.106: ANNOTATED PDF OF FLOW IN LAYER 1 OF THE MICROCIRCULATORY BED.....	345
FIGURE 8.107: ANNOTATED PDF OF FLOW IN LAYER 1 OF THE MICROCIRCULATORY BED.....	345
FIGURE 8.108: THE GRAPHICAL INTERPRETATION OF STANDARD DEVIATION ON A PDF WITH A NORMAL DISTRIBUTION.	347
FIGURE 8.109: IDENTIFICATION OF THE MEDIAN, 25% AND 75% LOCATION ON A GAUSSIAN DISTRIBUTION.	348
FIGURE 8.110: AN EXAMPLE OF AN ARRAY BEFORE AND AFTER SORTING.	348
FIGURE 8.111: A SORTED ARRAY WITH ANNOTATIONS FOR THE MEDIAN AND INTERQUARTILE RANGE (25% - 75%).	349
FIGURE 8.112: AN EXAMPLE OF A SORTED ARRAY BEFORE AND AFTER SUBTRACTING THE MEDIAN VALUE FROM THE EVERY ELEMENT OF THE VECTOR.	349
FIGURE 8.113: AN EXAMPLE OF FINDING THE MAD VALUE OF A MAD ARRAY (SEE FIGURE 8.112 FOR MORE DETAILS).	350
FIGURE 8.114: AN EXAMPLE OF A MAD ARRAY COMPARED TO THE OUTLIER VALUE, $3 \cdot (\text{MAD VALUE})$	350
FIGURE 8.115. GRAPHICAL REPRESENTATION OF THE DISCRETIZED 1D DOMAIN ON WHICH THE DIFFUSION-REACTION PROBLEM WILL BE FORMULATED. NOTE, Δx IS UNIFORM.	353
FIGURE 8.116. 2-DIMENSIONAL GRID SYSTEM ON WHICH THE TRANSPORT EQUATIONS WILL BE DISCRETIZED.	356
FIGURE 8.117. 2-DIMENSIONAL GRID SYSTEM ON WHICH THE TRANSPORT EQUATIONS WILL BE DISCRETIZED.	358
FIGURE 8.118. SCHEMATIC OF DOMAIN ON WHICH THE CURRENT 1D IMPLEMENTATION IS BEING SOLVED.	359

FIGURE 8.119. CASE STUDY 1.....	381
FIGURE 8.120. CASE STUDY 2.....	382
FIGURE 8.121. VISUALIZATION OF CASE STUDY 3.....	383
FIGURE 8.122. THE NETWORK USED FOR THE PRESENT CASE STUDY.....	384
FIGURE 8.123. SIMPLIFIED CEREBROVASCULAR NETWORK USED FOR OPTIMIZING REAL-WORLD DATA TO SIMULATION CONSTRAINTS.....	386
FIGURE 8.124. OPTIMIZATION GIVES BEST BC TO MATCH MEASURED FLOWS.	386
FIGURE 8.125. A STEADY STATE SIMULATION USING THE NOMINAL VALUES FROM NOVA.	387
FIGURE 8.126. THE DYNAMIC INLET AND OUTLET PRESSURES AND FLOW MEASUREMENTS.	388
FIGURE 8.127. THE DYNAMIC INLET AND OUTLET PRESSURES AND FLOW MEASUREMENTS.	389
FIGURE 8.128. THE DYNAMIC INLET AND OUTLET PRESSURES AND FLOW FROM MEASUREMENTS.	390
FIGURE 8.129. THE DYNAMIC INLET AND OUTLET PRESSURES AND FLOW FROM MEASUREMENTS.	391
FIGURE 8.130. THE DYNAMIC INLET AND OUTLET PRESSURES AND FLOW FROM MEASUREMENTS.	392
FIGURE 8.131. THE ALIGNMENT OF THE NOVA RESULTS AND PREDICTED FLOW PATTERNS USING FOURIER SERIES OPTIMIZATION.....	395
FIGURE 8.132. THE RESULTING FLOW OF TWO SEGMENTS WHEN THE INLET AND OUTLET BCs ARE NOT IN PHASE WITH ONE ANOTHER AND THE MEASUREMENTS HAVE KNOWN ERROR.....	396
FIGURE 8.133. OPTIMIZATION ON A SIMPLE BIFURCATION WITH KNOWN ERROR IN AMPLITUDE AND PHASE OF MEASUREMENTS. T	397
FIGURE 8.134. THE RECONSTRUCTED BA FLOWS USING A MULTITUDE OF FREQUENCIES.	399
FIGURE 8.135. THE RESULTING PRESSURE AND FLOW CURVES IN TIME AFTER BEING OPTIMIZED WITH NO ERROR.	400
FIGURE 8.136. THE DYNAMIC INLET AND OUTLET PRESSURES AND FLOW MEASUREMENTS.	401
FIGURE 8.137. THE RESULTING PRESSURE AND FLOW WHEN THE INLET AND OUTLET BCs ARE NOT IN PHASE WITH ONE ANOTHER AND THE MEASUREMENTS ARE COMMENSURATE WITH THE FORWARD SIMULATION.	403
FIGURE 8.138. THE RESULTING PRESSURE AND FLOW WHEN THE INLET AND OUTLET BCs ARE NOT IN PHASE WITH ONE ANOTHER AND THE MEASUREMENTS HAVE KNOWN ERROR.....	404
FIGURE 8.139. VISUALIZATION OF THE FIRST 4 PRINCIPLE ROOTS IN THE DISCRETE FOURIER SERIES.	422
FIGURE 8.140. VISUALIZATION OF THE POWERS OF THE FOURTH PRINCIPLE ROOT NEEDED IN DFT FOR DATA SETS WITH N=4 POINTS.	423
FIGURE 8.141. GRAPHICAL REPRESENTATION OF THE PRINCIPLE ROOT BEING ROTATED (RAISED TO DIFFERENT POWERS) AND SCALED BY $F(x_i)$	425
FIGURE 8.142. (BLUE) ORIGINAL SIGNAL AND (ORANGE) RECONSTRUCTION USING DISCRETE FOURIER SERIES AND INVERSE DISCRETE FOURIER SERIES.	427
FIGURE 8.143. A) NOISY PERIODIC SIGNAL. B) BLUE CURVE IS CONSTRUCTED BY USING ALL COEFFICIENTS IN IDFT WITH A LOW-PASS FILTER APPLIED.....	429
FIGURE 8.144. COMPARISON OF INDEXING SCHEMES BETWEEN 3 DIFFERENT DFT INDEXING PARADIGMS.	431
FIGURE 8.145. COMPARISON OF INDEXING SCHEMES BETWEEN TWO DIFFERENT DFT INDEXING PARADIGMS.	432
FIGURE 8.146. GRAPHICAL REPRESENTATION OF INTERPOLATION VECTORS CALCULATED WITH INDEXING SCHEMES OF TREFETHEN (LEFT) AND QUARTERONI (RIGHT).	436
FIGURE 8.147. A GRAPHICAL REPRESENTATION OF TWO MIXED MESH SYSTEMS (WITH REACTION, DIFFUSION, MASS TRANSFER, AND CONVECTION).	460
FIGURE 8.148. THE INTERFACE USED TO RUN ALL CASE STUDIES.....	462
FIGURE 8.149. THE COMPARISON OF DIFFERENT ITERATIVE SOLVING ALGORITHMS WITH SIMPLE DIAGONAL PRECONDITIONER.	493
FIGURE 8.150. THE COMPARISON OF DIFFERENT ITERATIVE SOLVING ALGORITHMS WITH SIMPLE DIAGONAL PRECONDITIONER.	504
FIGURE 8.151. THE COMPARISON OF DIFFERENT ITERATIVE SOLVING ALGORITHMS WITH PETSC BLOCK JACOBI PRECONDITIONER.	509
FIGURE 8.152. COMPARISON OF THE RUNTIMES FOR DIFFERENT SIZE SYSTEMS WITH DIFFERENT SOLVERS.	510
FIGURE 8.153. DESIGN OF THE RECURSIVE ALGORITHM THAT SPLITS THE NETWORK INTO SMALLER SEGMENTS UNTIL THE CRITERIA OF BEING ON THE SAME SCALE AS THE MESH IS CONFIRMED.....	523
FIGURE 8.154. THE RE-SEGMENTATION ALGORITHM CLEARLY INCREASES THE VASCULAR SEGMENT DENSITY WITHOUT ALTERING THE SEGMENT STRUCTURE.....	524
FIGURE 8.155. COMPARISON OF COMPUTATIONAL EFFORT TO SOLVE A SIMPLE DIFFUSION PROBLEM WITH ALL MESH DATA STRUCTURES (DENSE MESH) AND WITH ONLY MINIMAL DATA STRUCTURES (WITHOUT MESH).	525
FIGURE 8.156. GLOBAL AND LOCAL CO-ORDINATE SYSTEM FOR THE HEXAHEDRON ELEMENT.	527
FIGURE 8.157. EVALUATION OF TRANSPORT FLUX AND DIVERGENCE ON UNSTEADY CONCENTRATION PROFILE IN 1-DIMENSION.	530
FIGURE 8.158. OPPOSING PRESSURES THAT EXIST IN A SYSTEM WITH A SEMI-PERMEABLE MEMBRANE AND A PRESSURE GRADIENT.	532
FIGURE 8.159. CARTOON OF THE A SECTION OF THE BBB AND THE RELEVANT STATES.....	535
FIGURE 8.160. RECREATED CURVE FROM [252].....	537

FIGURE 8.161. PICTOGRAPHIC REPRESENTATION OF THE TWO SYSTEMS BEING EVALUATED.	538
FIGURE 8.162. VISUALIZATION OF TRENDS IN DIFFERENT MODELS IN RELATIONSHIP TO WATER FLUX AND INTERSTITIAL PRESSURE.	539
FIGURE 8.163. THE SCHEMATIC REPRESENTATION OF THE SIMULATED SYSTEM.	540
FIGURE 8.164. THE EFFECT OF PECLET NUMBER ON THE CONCENTRATION PROFILE.	540
FIGURE 8.165. THE EFFECT OF ENFORCING POSITIVE DEFINITE CONCENTRATION FOR C_i	542
FIGURE 8.166. CLASSIC AND REVISED STARLING'S LAW ACROSS A MEMBRANE WITH HYDROSTATIC (P_1 - P_2) AND OSMOTIC (C_1 - C_2) PRESSURE DRIVING FORCES.	544
FIGURE 8.167. (A) A CARTOON REPRESENTATION OF THE PRESSURE GRADIENTS THAT OCCUR ACROSS THE BBB THROUGH A SINGLE PORE.	563
FIGURE 8.168. CONCENTRATION PROFILE, $C(x)$ IN $\mu\text{MOL}/\mu\text{M}^3$, AS A FUNCTION OF LENGTH, x IN μM , BETWEEN $x=0$ AND $x=L$	574
FIGURE 8.169. RESIDUAL ERROR LINE PLOT (1D) AND SURFACE (2D) COMPUTED WITH EXHAUSTIVE ENUMERATION.	575
FIGURE 8.170. PLOT OF RESULTING FLOW WHEN CHANGING PARAMETRICALLY VARYING THE ARTERIAL PRESSURE (FROM $P_C=1$ - 100MMHG).	576
FIGURE 8.171. INFORMATION FLOW DIAGRAM AND PSEUDO-CODE FOR THE NEWTON METHOD USING STEPSIZE CONTROL.	582
FIGURE 8.172. THE VISUALIZATION OF ERROR OF THE NONLINEAR FUNCTION IN THE NEWTON DIRECTION.	584
FIGURE 8.173. RESIDUAL ERROR SURFACE FOR EQUATION (8.497).	585
FIGURE 8.174. FIRST NEWTON STEP WITH ARMIJO LINESEARCH EMPLOYED.	586
FIGURE 8.175. THE SECOND NEWTON STEP WITH ARMIJO LINESEARCH EMPLOYED.	586
FIGURE 8.176. THIRD NEWTON STEP WITH ARMIJO LINESEARCH EMPLOYED.	587
FIGURE 8.177. SOLUTION TO THE NONLINEAR EQUATIONS IS FOUND AFTER A FEW MORE SMALL UPDATES (NOT VISIBLE IN THE FIGURE).	587
FIGURE 8.178. FIRST NEWTON STEP WITH ARMIJO LINESEARCH EMPLOYED. IN THIS STEP, THE STEPSIZE NEEDS TO BE REDUCED IN ORDER TO GIVE A REASONABLE UPDATE.	588
FIGURE 8.179. SECOND NEWTON STEP WITH ARMIJO LINESEARCH EMPLOYED. IN THIS STEP, THE STEPSIZE NEEDS TO BE REDUCED IN ORDER TO GIVE A REASONABLE UPDATE.	589
FIGURE 8.180. THIRD NEWTON STEP WITH ARMIJO LINESEARCH EMPLOYED. IN THIS STEP, THE STEPSIZE NEEDS TO BE REDUCED IN ORDER TO GIVE A REASONABLE UPDATE.	589
FIGURE 8.181. SOLUTION TO THE NONLINEAR EQUATIONS IS FOUND AFTER A FEW MORE SMALL UPDATES (NOT VISIBLE IN THE FIGURE).	590
FIGURE 8.182. VISUALIZATION OF THE RESIDUAL ERROR ON A 2X2 SYSTEM OF EQUATIONS.	599
FIGURE 8.183. GRAPHIC REPRESENTATION OF THE COORDINATE TRANSFORMATION EXPRESSING THE GRADIENT OF THE FUNCTION IN TERMS OF THE MATRIX A AND VECTORS B AND X GUESS.	601
FIGURE 8.184. SCHEMATIC REPRESENTATION OF THE IDEALIZED SYSTEM.	604
FIGURE 8.185. BIOMETRIC STUDY OF DIFFERING D2HG RELEASE RATES AND DIFFUSIVITIES.	606
FIGURE 8.186. SCHEMATIC REPRESENTATION OF THE IDEALIZED SYSTEM.	608
FIGURE 8.187. BIOMETRIC STUDY OF DIFFERING MATERIAL DIFFUSIVITIES AND TUMOR PRODUCTION RATES.	611
FIGURE 8.188. THE COMPARISON OF THE FINITE VOLUME METHOD AGAINST THE ANALYTIC SOLUTION SHOWS EXCELLENT AGREEMENT.	613
FIGURE 8.189. COMPARISON BETWEEN THE FEM AND FVM ON THE SPHERICAL GEOMETRY SYSTEM.	614
FIGURE 8.190. THE COMPARISON OF THE FINITE VOLUME METHOD WITH THE FINITE ELEMENT METHOD.	616
FIGURE 8.191. THE COMPARISON OF THE FINITE VOLUME METHOD WITH THE FINITE ELEMENT METHOD WHERE EACH HAS A SUFFICIENT MESH DENSITY TO PROVIDE MESH INDEPENDENCE.	617
FIGURE 8.192. COMPARATIVE ERROR AS A FUNCTION OF MESH DENSITY.	618
FIGURE 8.193. OVERVIEW OF RELEVANT TRANSPORT PHENOMENA RELATED TO THE LIFESPAN OF OXYGEN IN THE BRAIN.	626
FIGURE 8.194. SCHEMATIC OF DOMAIN ON WHICH THE CURRENT 1D IMPLEMENTATION IS BEING SOLVED.	630
FIGURE 8.195. GRAPHICAL REPRESENTATION OF TWO DISCRETIZATION TECHNIQUES IN 1-DIMENSION.	634
FIGURE 8.196. COMPARISON OF DIFFERENT MESH SIZES WITH THE ANALYTIC SOLUTION OF THE PEAK CONCENTRATION (COMPUTED ABOVE).	634
FIGURE 8.197. COMPARISON OF DIFFERENT MESH SIZES WITH THE ANALYTIC SOLUTION OF THE PEAK CONCENTRATION (COMPUTED ABOVE).	635
FIGURE 8.198. COMPARISON OF DIFFERENT MESH SIZES.	637
FIGURE 8.199. OVERVIEW OF DOMAIN ON WHICH THE CURRENT 1D IMPLEMENTATION IS BEING SOLVED.	638
FIGURE 8.200. SCHEMATIC LAYOUT OF THE 1D SIMPLIFIED PROBLEM.	638
FIGURE 8.201. PROFILE SHAPE COMPARISON OF DIFFERENT BOUNDARY CONDITIONS AND REACTION MODELS FOR A 1D DIFFUSION/REACTION PROBLEM.	640
FIGURE 8.202. COMPARISON BETWEEN DIFFERENT REACTION RATES AND BOUNDARY CONDITIONS IN THE 1D OXYGEN SIMULATION.	641

FIGURE 8.203. THREE METHODS (MODELS) FOR MASS TRANSFER BETWEEN VASCULATURE AND TISSUE.	642
FIGURE 8.204. COMPARISON OF DIFFERENT MASS TRANSFER MODELS ON MESH INDEPENDENCE CONVERGENCE BETWEEN DIFFERENT BOUNDARY CONDITIONS.	643
FIGURE 8.205. COMPARISON BETWEEN DIFFERENT METHODS OF MASS TRANSFER IN A 1D MODEL.	644
FIGURE 8.206. COMPARISON OF MESH INDEPENDENCE FOR INSULATED TISSUE BOUNDARIES (FLUX=0) AND 1 ST ORDER REACTION).....	645
FIGURE 8.207. COMPARISON OF MESH INDEPENDENCE FOR PERIODIC TISSUE BOUNDARY CONDITIONS AND 1 ST ORDER REACTION.	646
FIGURE 8.208. VALIDATION OF THE IMPLEMENTATION OF THE RADIAL DIFFUSION PROBLEM BY VERIFYING BOUNDARY CONDITIONS	652
FIGURE 8.209. VARYING THE RADIUS OF THE CYLINDER SOURCE (A) AND ITS EFFECT ON THE FLUX THROUGH THE SYSTEM.....	655
FIGURE 8.210. VARYING THE RADIUS OF THE CYLINDER SOURCE (A) AND ITS EFFECT ON THE FLUX THROUGH THE SYSTEM USING DISCRETIZED IMPLEMENTATION.	656
FIGURE 8.211. COMPARISON OF DIFFERENT DISCRETIZATION SCHEMES.	658
FIGURE 8.212. COMPARISON OF ANALYTIC TO DISCRETE SIMULATIONS OF CONCENTRATION PROFILES WITH VARYING THE SOURCE VESSEL RADIUS (a).	660
FIGURE 8.213. 3D PROFILES OF THE CONCENTRATION SIMULATED WITH NUMERICAL IMPLEMENTATION WHILE VARYING THE MESH DENSITY.....	661
FIGURE 8.214. 3D PROFILES OF THE CONCENTRATION SIMULATED WITH NUMERICAL IMPLEMENTATION WHILE VARYING THE VESSEL RADIUS, A.	662
FIGURE 8.215. VASCULAR FLOW AND PRESSURE SIMULATED IN A FULL 3D VOXEL MATRIX USING DARCY'S LAW.	668
FIGURE 8.216. EXAMPLES OF VESSEL INTERIOR CELLS THAT ONLY SHARE A SINGLE NEIGHBORING INTERIOR CELL.	669
FIGURE 8.217. EXAMPLES OF CONVECTION IN NETWORKS THAT USE A DIFFUSIVE-LIKE EQUATION TO ACCOUNT FOR ORPHANED CELLS. ..	670
FIGURE 8.218. SAMPLE CASE STUDY OF CONVECTION WITH A VERY LOW DARCY CONSTANT, K, (=VERY LOW FLOW).	671
FIGURE 8.219. CASE STUDY FOR CONVECTION - MASS TRANSFER - DIFFUSION — REACTION SYSTEM PERFORMED USING ONLY A 3D MESH.	672
FIGURE 8.220. SCHEMATIC DIAGRAM OF A NETWORK CYLINDER (WITH EMBEDDED FLOW VECTOR) REGISTERED TO A VOXEL MATRIX IN 2 DIMENSIONS.....	673
FIGURE 8.221. GRAPHIC DEPICTION OF FOUR SCENARIOS WHEN CLASSIFYING MASK ON MESH TO DETERMINE EQUATIONS.	677
FIGURE 8.222. THE SLICE (LEFT) AND TRANSPARENCY (MIDDLE) VIEW OF THE OXYGEN DISTRIBUTION AROUND THE SIMPLIFIED SYNTHETIC NETWORK SHOWS REASONABLE AND CONSISTENT TRENDS BETWEEN THE DENSE MESH (MIDDLE) AND COARSE MESH (RIGHT).	679
FIGURE 8.223. FIRST ORDER IMPULSE RESPONSE MODEL AS IN EQUATION (8.572).....	689
FIGURE 8.224. SCHEMATIC REPRESENTATION OF THE TWO MODELS FOR FUNCTIONAL HYPEREMIA.	690
FIGURE 8.225. WORKFLOW DIAGRAM FOR FUNCTIONAL HYPEREMIA FOR SIMPLIFIED VASCULAR OXYGEN BINDING.	690
FIGURE 8.226. WORKFLOW DIAGRAM FOR FUNCTIONAL HYPEREMIA FOR SIMPLIFIED VASCULAR OXYGEN BINDING.	691
FIGURE 8.227. THE CHEMICAL REACTION SYSTEM OF A,B AND C SPECIES USED FOR THIS CASE STUDY.	709
FIGURE 8.228. COMPARISON BETWEEN TOP LEFT) IMPLICIT EULER (IE), TOP RIGHT) ADAMS BASHFORD (AB), BOTTOM LEFT) RUNGE- KUTTA 4, AND BOTTOM RIGHT) RUNGE-KUTTA 5 INTEGRATORS AT DIFFERENT TIMESTEP SIZES.	710
FIGURE 8.229. EXAMPLE OF EDGE DETECTION ALGORITHM VISUALIZED WITH MATLAB.	720
FIGURE 8.230. EXAMPLES OF EDGE DETECTION AT VARYING MESH DENSITIES FROM 10x10 (TOP LEFT) TO 500x500 (BOTTOM RIGHT). ..	721
FIGURE 8.231. VISUALIZATION OF THE DISCRETE LABELING ON TWO MESHES.	722
FIGURE 8.232. VISUALIZATION OF THE AMALGAMATED MESHES INTO A SINGLE LABELED MATRIX.	723
FIGURE 8.233. CONCENTRATION PROFILE OF THE CIRCULAR DOMAIN SIMULATION.	723
FIGURE 8.234. PICTORAL REPRESENTATION OF THE VESSEL LABELING ALGORITHM IN 3 DIMENSIONS.	725
FIGURE 8.235. VISUALIZATION OF NETWORK EDGE DETECTION USING DICOM VIEWER TOOLS.	726
FIGURE 8.236. VISUALIZATION OF SIMULATION FOR TWO NETWORKS USING VESSEL EDGE DETECTION VISUALIZED WITH DICOM IMAGE REVIEWING TOOLS.	727
FIGURE 8.237. VISUALIZATION OF OXYGEN SIMULATION IN MANY DICOM REVIEW TOOLS.	728
FIGURE 8.238. RAYTRACES FOR DIFFERENT RAYS THROUGH THE 3D BLOCK WITH VARYING MASS TRANSFER COEFFICIENT CORRESPONDING TO THE ABOVE TABLE.....	731
FIGURE 8.239. RAYTRACES FOR DIFFERENT RAYS THROUGH THE 3D BLOCK WITH VARYING THE METABOLIC RATE (CMRO ₂) CORRESPONDING TO THE ABOVE TABLE.	733
FIGURE 8.240. RAYTRACES FOR DIFFERENT RAYS THROUGH THE 3D BLOCK WITH VARYING DIFFUSIVITY CORRESPONDING TO THE ABOVE TABLE.....	735
FIGURE 8.241. PARAMETRIC STUDY OF MOLAR FRACTION OF OXYGEN AND THE CALCULATION OF (LEFT) MOLES OF OXYGEN AND (RIGHT) PARTIAL PRESSURE OF OXYGEN.	742

FIGURE 8.242. EFFECT OF SHIFTING THE EQUILIBRIUM CONCENTRATION ON THE RESULTS OF THE HILL EQUATION.	753
FIGURE 8.243. IMPLEMENTATION OF GREEN'S FUNCTION WITH A SHORT SEGMENT (YELLOW REGION) THAT TERMINATES MIDWAY THROUGH THE DOMAIN.	772

LIST OF ABBREVIATIONS

HP:	Hagen Poiseuille
RBC:	Red blood cell
RHS:	Right hand side
CCO:	Constrained constructive optimization
mCCO :	Modified constrained constructive optimization
MCA:	Medial cerebral artery
ACA	Anterior cerebral artery
PCA	Posterior cerebral artery
ICA	Internal carotid artery
BA	Basilar artery
FSI:	Fluid-structure interaction
SNR:	Signal-to-noise-ratio
STL:	Stereo lithography
BBB:	Blood brain barrier
MCA:	Middle cerebral artery
ACA:	Anterior cerebral artery
PCA:	Posterior cerebral artery
SSS:	Superior sagittal sinus
KPSM:	Kinetic plasma skimming model
iCNS:	Image-based circulatory network synthesis
FEA:	Finite element analysis
FVM:	Finite volume method
FEM:	Finite element method
BS:	Boundary condition
PDF:	Probability density function
CDF:	Cumulative density function
AD:	Alzheimer's Disease
2PLSM	Two-photon laser scanning microscopy
μCT	Micro-computed tomography
MRI	Magnetic resonance imaging
GB	Gigabyte(s)
fL	Femtoliter(s)
nL	Nanoliter(s)
s	Second(s)
min	Minute(s)
m	Meter(s)
L	Liter(s)
mm	Micrometer(s)
mm	Millimeter(s)
g	Gram(s)
kg	Kilogram(s)
Pa	Pascals
2D	Two-dimension(al)
3D	Three-dimension(al)
MINLP	Mixed-integer nonlinear programming

NP	Non-polynomial
mmHg	Millimeters of mercury
nsgm	Number of segments
RAM	Random access memory
CPU	Computer
SMCA	Synthetic MCA
SH	Synthetic hemisphere
Trs	Transverse sinuses
Crhv	Caudal rhinal vein
Rrhv	Rhostral rhinal vein
SL	Starling law
RSL	Revised Starling law
BOLD	Blood oxygen level-dependent
fMRI	Functional MRI
CBF	Cerebral blood flow
ANOVA	Analysis of variance
A	Arteries
PA	Penetrating arteries
C	Capillaries
AV	Ascending venules
V	Veins
Hct	Hematocrit
SS	Steady-state

SUMMARY

This thesis discusses novel breakthroughs in mathematical modeling of cerebral vasculature and extravascular space in mouse. Previous limitations in modeling this *neurovascular unit* were overcome with improvements to (i) vascular synthesis, leading to creation of vascular models spanning a much larger domain than image reconstructions, (ii) robust solving techniques and analysis for stable convergence of nonlinear biphasic blood flow on such large networks, and (iii) novel 1D-3D coupling between the compartments of the neurovascular unit effectively removing the limitations exhibited by previous models.

The improvements to vascular synthesis models incorporated (i) growth from pial surface reconstruction meshes, (ii) microvascular closure (for closing an arterial tree and a venous tree through a realistic capillary bed, and (iii) automatic matching of topological properties to produce realistic vascular structures. These models were validated against empirical reconstructions of microvasculature from 2-Photon laser-scanning microscopy by comparing length, diameter, surface area, volume and tortuosity spectra. These models were also capable of constructing complete cerebrovascular models for the entire MCA territory and entire hemisphere in mouse.

Robust fixed-point solving methods were implemented for consistent simulation convergence for the nonlinear equations of biphasic blood flow (red blood cells + plasma). These improved simulations, when interrogated, revealed a depth-dependent hematocrit (=red blood cell volume fraction) gradient in the microcirculation which was previously unknown. This finding was verified between empirical and synthetic vascular networks including the MCA territory and hemisphere.

A novel 1D-3D coupling methodology was also created to resolve the interface between the vascular network and extravascular tissue space in the brain with greatly enhanced solvability. This method derives from a Cartesian mesh masking logic greatly enhancing simulation

solvability. Validation of the model was performed against 2-photon phosphorescence lifetime microscopy measurements and was expanded to large portions of the mouse cortex. The model was also capable of simulating a larger subsection of the cortex than previously performed ($3 \times 3 \times 1 \text{ mm}^3$) with a mesh density of ~95 million elements, significantly larger than other models of the same type.

These new adaptations lay the groundwork for simulating the neurovascular unit at the scale of the entire mouse brain (hemisphere) and even the human brain. Included appendices propose detailed models for adaptation of the work in this thesis to the aged brain, functional hyperemia, and human vascular synthesis.

1 Introduction

Many late-onset neurodegenerative diseases involve degradation of the dynamic coordination between vascular blood flow and neural tissue oxygenation. These diseases are hallmarked by the generation of hypoxia-related events, such as an increase in beta-amyloid production in Alzheimer's Disease (AD)[1,2]. Many forms of dementia suffer from decreased oxygen tension while also exhibiting significant morphological changes in vascular structure. Unfortunately, healthy aging (aging without dementia) also suffers from similar morphological changes to vascular structure and oxygen tension [3]. While the link between vascular restructuring and changes in cerebral oxygen content have been established, no single intervention methodology can treat all changes simultaneously. Moreover, the healthy aging brain changes morphology as well, so some changes to vascular topology may be necessary to maintain homeostasis. In order develop methods for treating these age-related dementia, it is imperative to quantitatively explore changes to the vascular topology and the corresponding alteration to oxygen tension in the brain.

Animal studies have attempted to elucidate the detailed neurovascular coupling and quantify age-related neurodegeneration[4–8] including investigations investigating the link between hypoxia, β -Amyloid production, and plaque formation leading to AD [1,2]. A recent study, for example, investigated oxygen distribution in neurological tissue to identify age-related formation of hypoxic pockets[4]. Other studies were able to identify changes in the rate of temporary blockage of capillary vessels in older subjects [5–8]. Despite all the neuroimaging progress, simultaneous data for detailed microcirculatory structure during age-related impaired oxygen delivery is still missing. This gap in knowledge can be bridged by computational paradigms.

In order to investigate the causal link between cerebral blood flow, oxygen distribution, and oxygen metabolism, many computational studies have been proposed [9–20]. Models with

simplified vasculature give great insight into the role of vascular structure on changes to oxygen tension, yet suffer from the drawback that they are incapable of resolving the complex heterogeneous oxygen distribution as in the cerebral micro-environment. In light of this, anatomically consistent neurovascular models have been proposed [9,21–25].

In order to predict the oxygen tension throughout the cerebral domain, a computational model of the vasculature and extravascular space (meshing) must be provided. The spatial density of the cerebrovasculature in mouse is $\sim 11,000$ segments/mm³ [26,27] in mouse spanning ~ 300 mm³ (estimate from reconstructed images [28]) which gives 3,300,000 segments in just the small mouse brain. The human brain ($\sim 1.45 \cdot 10^6$ mm³) is roughly 4800x the size of the mouse brain (by volume), generating ~ 16 billion vessels. To create an unstructured mesh out of such a large network would create an astronomical number of elements. Even with the most advanced Cartesian mesh simplifications of such a domain, as presented by Ghaffari et al [29], the estimated number of mesh elements is ~ 64 trillion volume elements ($64 \cdot 10^{12}$ volume elements) for the vasculature alone (Equation (2.13)).

$$\frac{1.958M \text{ elements}}{482 \text{ segments}} = 4062 \text{ elements/segment} \quad (1.1)$$

Compounding this number with the four unknown values per element when solving blood flow using the Navier-Stokes equations gives 247 trillion unknowns ($247 \cdot 10^{12}$). With the upper limit of published linear algebraic solvers in the mere hundreds of millions (10^{11}) [30], this scale of simulation is infeasible.

One way to drastically reduce the number of equations is to model linear segments as cylinders in 3D space. This model assumes radial and azimuth symmetry (uniform plug flow), leaving only

the axial dimension for spatial variation in state. This simplifies to a 1D simulation (in a 3D domain) represented as a graph where the lines represent the arcs (each arc is endowed with a radius value) and the junctions in the network are represented by points. This mesh will preserve the number of equations for a blood flow simulation to the originally anticipated 3.3 million equations, which is well within reason for linear algebraic solvability. The theory and implementation of network analysis for 1D linear flow networks can be found in Section 7.11.

To model the extravascular oxygen tension, many models have attempted the coupling of oxygen transport through the vasculature and extravascular transport. Models that propose a 1D-3D analytic or semi-analytic solution to oxygen distribution are computationally prohibitive for models larger than a few hundred segments [25,31–33]. The traditional approach for such a simulation is to apply a body-fitted tetrahedral mesh of extra- or intra-vascular space, due to the plethora of professional toolboxes for these types of meshes, however the amount of mesh elements to accurately simulate large, dense microcirculatory networks ($>11,000$ vessels/mm³) is computationally prohibitive. Fortunately, Cartesian vascular structures have recently been developed to offer a significant computational simplification of the 3D blood vessel domain, yet these methods have not yet been applied to the dense microcirculatory networks [29,34,35].

In light of these drawbacks, a few groups recently proposed homogenization methods for the microcirculatory domain [36,37]. These methods proposed to represent the anisotropy of generalized capillary flow by reducing the problem to a series of Cartesian mesh volumes and an anisotropic diffusion tensor. Unfortunately, while these methods offer great scalability, this method is not amenable to the discrete network of segments that the blood must adhere to while traversing the cerebral microcirculation.

A final method for 1D-3D coupling was proposed using a mass transfer flux between each 1D network element and the closest 3D mesh volume. This method was robust and capable of solving very large vascular structures due to the simplicity of the coupling mechanism.

Unfortunately, none of the proposed methods are stable for scalability to an entire mouse brain. This means boundary effects remain apparent throughout the current simulation community. These effects can cast doubt over the results, given that they are artificial (do not exist in the actual brain). This thesis will build the framework for a whole-brain scale investigation of the neurovascular unit. A method is proposed for informed anatomical synthesis of cerebral microvasculature at the whole-brain scale uses the information of dense microcirculatory network reconstructions to generate every vessel in a mouse hemisphere with matching topological properties. Linear and biphasic blood flow are predicted on these vast networks to reveal a depth-dependent trend in red blood cell concentration. Finally, a method is proposed for a stable, scalable neurovascular mesh coupling that has the potential for scaling to the scale of the entire hemisphere.

To overcome the limited spatial extent as in 2PLSM reconstructions or limited reconstruction resolution as in μ CT image reconstructions, an image-based circulatory network synthesis (iCNS) methodology was created. This algorithm uses an information from 2PLSM networks and scales the growth to the scale of an entire mouse brain. The synthesis of every vessel in an anatomically-consistent hemisphere (~ 1 million vessels) took less than 2 hours and 6GB of memory on a personal computer. This breakthrough marks a significant contribution to anatomically-accurate cerebrovascular modeling.

The advanced implementation of large, sparse linear algebraic solvers with a robust successive over-relaxation (SOR) and physically-consistent plasma skimming formula allowed the prediction of the nonlinear relationship between red blood cell (RBC) distribution and network geometry

throughout the entire hemisphere in mouse. The investigation of the RBC distribution in smaller microcirculatory networks (both image-based and synthetic) led to the discovery of an RBC depth-dependent gradient in the brain. This finding was also confirmed by the larger simulations of the hemisphere, showing this is not simply a boundary effect but a systemic characteristic of the network structure. This discovery was later validated with animal experiments from another group [38].

To improve the scalability of full 3D meshing while preserving its ability to resolve the distributed endothelial layer mass transfer at the scale of the point-volume coupling, we have developed a novel Cartesian mesh masking technique with vessel detection. This method combines the benefits of 3D endothelial resolution for larger vessels while allowing significant expandability to large microcirculatory sections of the cerebral cortex. This new model can also be applied to the models of the aging brain to investigate independent contribution of vascular structure changes on the cortical oxygen tension.

The proposed methodology will bridge the knowledge gap by removing the artificial boundary edge thus improving the accuracy of the results. In order to expand current methodologies for stable convergence at the whole-brain scale novel methods were derived for generating physiologically-consistent cerebrovascular networks at the whole-brain scale, solvability was improved for computations of biphasic blood flow, and a new, highly stable and scalable paradigm for coupling a 1-dimensional (1D) vascular network with a 3-dimensional tissue domain was created. These advancements will propel the next level of simulations to the whole-brain scale without sacrificing anatomical detail.

The main manuscript (Sections 0-0) describes the major advancements of this thesis, although a significant amount of derivation, validation, and implementation was necessary for achieving

these goals. While this information is highly necessary for the development of the models used for the main manuscript, the details were too lengthy to appear in the main body. This is exemplified by the high number of supplemental chapters in this thesis (Sections 7.1-7.38). To assist the reader with traversing this long list of available information contained within this thesis a reference table is offered:

Table 1.1. Reference table for the supplemental data in this document

Section number	Relevance/Topic	Relevant Thesis Section
7.1	Vascular synthesis implementation notes	2-3
7.2	Detailed analysis of sample generator implementation and sample bias	2-3
7.3	Calculation of cortical depth from a triangulated mesh surface	2-4
7.4	Extended topological analysis of empirical networks	3
7.5	An alternative method for connecting a point to a nearby line segment using an orthogonal connection	2-3
7.6	A detailed method for reducing the tortuosity of a spline to a finite number	3
7.7	Derivation and implementation of an alternative model for automated CDF matching	3
7.8	Next-generation methods for expanding pial vascular network synthesis to the human cortex	2
7.9	Methods for reconstructing vascular networks from images	2-5
7.10	Methods for reconstructing brain meshes	2-4
7.11	An introduction to network analysis of blood flow	5
7.12	History and description of biphasic nature of blood flow, namely models of plasma skimming and hematocrit-dependent viscosity	4
7.13	Methods for visualizing and interrogating model predictions	2-5
7.14	Methods for discretizing a mesh or network during computational analysis	4-5
7.15	Implementation and derivation of different boundary condition choices	4-5
7.16	Matrix derivative validation	22
7.17	Discrete Fourier series derivation and implementation notes	22
7.19	Validation of oxygen simulation implementation with case studies	5
7.20	Validation of PETSc implementation with case studies	4-5
7.21	Cartesian mesh implementation and benefit	5
7.22	Diffusion: an illustrative description and example	5
7.23	Starling law and revised Starling law derivation and notes	5
7.24	Newton method for solving nonlinear equations	4
7.25	Properties of linear algebraic sets of equations	4-5
7.26	Simulations of idealized brain geometry	5
7.27	Validation of spherical geometry for diffusion-reaction system	32
7.28	Detailed implementation of oxygen transport in the neurovascular unit in 1D, 2D and 3D	5
7.29	Proposed methods for modeling the aged brain	5
7.30	Proposed functional hyperemia model	5
7.31	Implementation and comparison of different time integration schemes	36

7.32	Implementation and derivation of edge detection in a Cartesian mesh	5
7.33	Parametric studies of oxygen transport in the brain	5
7.34	Derivation of calculating molar concentration of oxygen given partial pressure oxygen tension	5
7.35	Calculating free oxygen tension using hemoglobin binding kinetics	36
7.36	The Hill equation derivation and implementation	41
7.37	Description of competing models for the neurovascular unit	5
7.38	Description of alternative models for vascular synthesis	2-3

2 Image-based circulatory network (iCNS) synthesis Part I: theory and image integration

Parts of this chapter were previously published as Linninger, Andreas, Grant Hartung, Shoale Badr, and Ryan Morley. "Mathematical synthesis of the cortical circulation for the whole mouse brain-part I. theory and image integration." *Computers in Biology and Medicine* (2019).

2.1 Introduction

Detailed anatomical models of cerebral circulation can serve as virtual surrogates enabling a quantitative analysis of functional mechanisms in the brain. To ensure consistency, digital vascular models should match physiological and anatomical topology of in vivo cerebral angioarchitecture. Image-segmentation is an essential technique to acquire necessary physiological information such as the number, position, and connectivity of arterial and venous segments. Specific implementations of many algorithms for anatomical reconstructed are outlined in Section 7.9. Accordingly, several groups have created microcirculatory models from neuroimages [25,39–42]. However, segmentation of image data faces several challenges. The number of microvascular segments is staggering; the human brain is estimated to have more than 10 billion capillaries, or $\sim 8,000$ segments/mm³ [42]. Morphometric studies on cadaver brains [42,43] are also problematic, because blood flow ceases and capillary networks may collapse post-mortem. In vivo imaging acquisition looks at microcirculation through a cranial window that affords only a narrow glimpse of the brain covering a range of hundreds of microns to a millimeter. At the edges of the imaging window, it is unavoidable that all pial connections of larger arteries, deeper arterioles, and capillaries are severed. These artificial cut-off boundaries expose mathematical models to boundary effects, whose detrimental impact on reliable predictions has been pointed out by Lorthois [40,44]. Moreover, raw neuroimaging data require extensive post-processing to fill gaps,

remove dangling segments, or reconcile noisy or missing information at the imaging threshold [45]. No single imaging modality can directly image all blood vessels in the brain at the macro- and micro-anatomical scale reliably, although there is progress towards this goal [46–48]. This limitation leaves gaps in data acquired at the macro[49–52] and micro [25,41,42] anatomical scale. More details on reconstructions can be found in Section 7.9.

Synthetic vascular models offer an alternative to purely image-based approaches that may suffer from uncertainty in the microscale [36,37,53,54]. Bui et al. used a fractal tree model with a level set distance function [55]. Schreiner and Karch [56–61] generated coronary arterial trees artificially by combining the principle of volume minimization (=minimum blood lumen) with random segment addition. Their *constrained constructive optimization* (CCO) algorithm is capable of synthesizing branched structures that resemble natural arterial trees. They successfully created physiologically sound arterial trees in flat sheets (2D) and slabs (3D). The algorithm also synthesized trees whose morphometrics closely matched coronary arteries. However, constructive synthesis of tree-like structures breaks down for microcirculatory networks, because they have loops and anastomoses, thus they are not binary trees. It is therefore not possible to perform tree lumen minimization by recursion, thus rendering the required combinatorial optimization intractable. Accordingly, the most critical limitation of classical CCO is its inability to create circulatory networks which connect the arterial side to the venous circulation through a physiologically consistent capillary bed. More information on previous vascular synthesis models can be found in Section 7.38 .

Recently, Linninger presented an alternative method for building realistic microcirculatory beds using Voronoi tessellation [62]. They synthesized the cortical blood supply in a sizable section of the human cortex with physiologically sound multi-scale representation of arteries, the

capillary bed and the venous circulation [62]. However, that solution requires expensive Delaunay and Voronoi tessellations followed by diameter smoothing procedures. This article introduces a novel microvascular closure that employs construction principles directly, so that closed networks encompassing an arterial and a venous tree with capillary connection can be synthesized with a single algorithm.

This section presents a novel methodology entitled image-based Circulatory Network Synthesis (iCNS) that combines the critical advantages of image-based models with synthetic vascular growth. The methodology and implementation sections will present mathematical background and procedures. Section 2.4 will demonstrate the creation of realistic models for the circulation in major vascular territories (MCA territory) and present the successful application of the methodology to synthesize a realistic model of the cortical circulation for the entire mouse brain.

2.2 Methodology

Image-based Circulatory Network Synthesis (iCNS) is also based on constrained constructive optimization principles developed by the pioneering work by Karch and Schreiner [56–61]. Foundation of vascular synthesis within a mathematical programming (=optimization) theory has been implied by their work, but not been presented formally. Therefore, we derive constrained constructive optimization principles from a formal mixed integer nonlinear programming (MINLP) framework in Section 2.2.1. We generalize the method of Karch and Schreiner in two critical aspects: First, a novel scale-invariant nonlinear programming formulation enables the incorporation of anatomical features from image data (Section 2.2.2). Second, a novel *microcirculatory closure* algorithm for synthesizing capillary networks provides the critically missing connection between arterial and venous circulation (Section 2.3). Note, more application

details for the microcirculatory closure and anatomical topological matching are offered in the next chapter and in Section 7.1.

2.2.1 The formal optimization problem.

A vascular tree can be created artificially by starting with a single cylindrical segment (=root branch), see Figure 2.1. The tree is expanded by spawning additional segments until a desired number of branches is reached. Branches are added by connecting a close segment from an existing branch to a new terminal node at a random location. Branch addition must obey geometrical and physiological constraints to ensure that the artificial tree matches real vascular topology. This is ensured by imposing the objective of minimizing the overall tree volume with the side constraint that blood conveyed through the tree segments perfuse the domain evenly. For each segment addition step, tree volume minimization subject to hemodynamic constraints can be formally expressed as a mixed integer non-linear program (MINLP) given by system (2.1). It is a global minimization problem with binary variables, y , that decide the location (=existing branch) where the new segment should be attached. Only one connection can be made at each step as expressed by the logical constraint in (1b). Moreover, morphometric parameters (=segment length and diameters, for nomenclature see Table 2.1) appear in the highly non-linear resistance computations of fluid flow equations, $F(x, y) = 0$, in Equation (1a).

$$\min_{x,y} V(x, y) \tag{2.1}$$

$$s. t. \quad F(x, y) = 0 \tag{1a}$$

$$\sum_{i=1}^N y = 1 \tag{1b}$$

$$x = \{\alpha, p, q\}, \quad y \in \{0,1\}^N$$

Table 2.1. Nomenclature

Symbol	Description
A	Diagonal resistance matrix
a	Accumulated downstream resistance
$\text{Bif}(\xi^*, \eta^*)$	Unknown bifurcation position
C_1	Connectivity matrix
C_2	Incidence matrix
d, d_0	Segment diameter, root segment diameter
$F(x)$	Flow equations (mass, momentum, conservation)
N	Number of possible choices for connecting new terminal
p, p_0, p_t	Vascular nodal pressure, inlet pressure, terminal pressure
Q, Q_0	Vascular flow, root segment flow
r	Segment reduced resistance*
r_0	root segment reduced resistance=total reduced resistance of tree*
V	Vascular tree volume
x	Vector of unknowns (resistances, flows, and pressures)
y	Binary decision to choose connection
α	Vascular segment resistance
α, α_0	Single segment resistance, root segment resistance
β	Diameter ratio
ξ	x coordinate of bifurcation
κ	Power law parameter (set to 3)
η	y coordinate of bifurcation
ρ	Accumulated downstream reduced resistance*

*Reduced resistance is the length-dependent portion of the resistance that does not contain diameter dependence

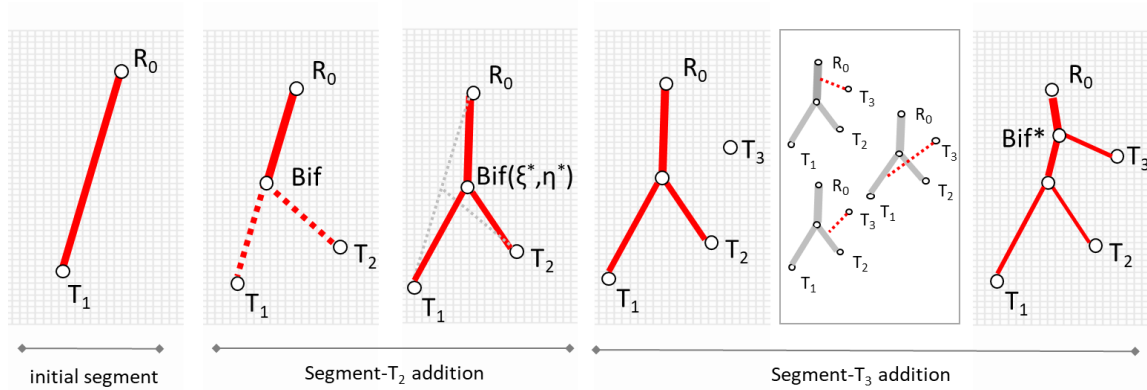


Figure 2.1. Overview of CCO method by Schreiner and Karch starting with one segment. Arterial tree growth begins with an initial segment, R_0 - T_1 . Segment addition at a random point, T_2 , defines a new segment, Bif - T_2 . The optimal coordinates in the bifurcation plane (ξ^*, η^*) minimize the tree volume and balance the tree by adjusting all segment diameters. Segment- T_3 addition entails a logical decision to select the globally smallest tree out of $N = 3$ possible segment connections. In this example, segment addition at the segment root is optimal. At each stage, the *balanced* tree discharges equal blood flow to all terminal nodes (T_1 - T_3).

Each volume minimization step is a *non-polynomial (NP) hard* problem, because it contains logical (=new possible connection to multiple existing segments) and parametric decisions (=optimal positions $\text{Bif}(\xi^*, \eta^*)$ which in turn determine segment lengths, diameters, resistances, α , blood flows, q , and pressures, p , along the vascular tree). Two *admissible* heuristics help prune the search:

- Only consider N segments closest to the new terminal as possible connections (*near-vicinity heuristic*)
- Search bifurcation locations in the two dimensional subspace spanned by the bifurcation topology (parent node and two child nodes - *planar bifurcation heuristic*).

The near vicinity heuristic allows problem decomposition into independent nonlinear subproblems given in system (2.2), one *volume minimization* problem, $V(x)$, for each possible connection, y , in the N nearest neighborhood of the new terminal node. The first constraint relates the flow, q , to pressure drops, p , which is compactly expressed in matrix form using the diagonal resistance matrix, $A = \text{diag}(\alpha(\xi, \eta))$, which holds the segment resistances, $\alpha(\xi, \eta)$, as functions of the unknown bifurcation position, (ξ, η) . The incidence matrix C_1 stores the node connectivity for each segment. The second constraint, $C_2 q = 0$, enforces flow conservation for each node. More details on the use of incidence matrices C_1 and C_2 for formulating network flow problems can be found elsewhere [63–65]. The planar bifurcation heuristic limits the position of new connections to two independent spatial coordinates (ξ, η) inside the bifurcation plane, which reduces the computational burden. The desired solution is the globally best connection with planar bifurcation

coordinates (ξ^*, η^*) that determine associated tree metrics (=length, diameters) which in turn fixes fluid flows and pressures.

For N segments close to new terminal T do (for each possible connection y do)

$$\begin{aligned} & \min_x V(x) \\ & \text{s. t. } \begin{bmatrix} A(\xi, \eta) & C_1^T \\ C_2 & 0 \end{bmatrix} \begin{pmatrix} q \\ p \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad x(\alpha(\xi, \eta), p(\xi, \eta), q(\xi, \eta)) \end{aligned} \quad (2.2)$$

Problem reduction. Each constrained NLP in Equation (2.2) can be further simplified into a single unconstrained nonlinear function minimization problem in merely two variables (=the bifurcation position, (ξ, η)). The volume in a balanced tree can be precisely determined with the help of Equation (2.3)-(2.5) for a desired perfusion ratio $Q_0/\Delta P$ and segment lengths determined by the bifurcation position (ξ, η) . *Balancing* the tree brings the substantial benefit that instead of solving for all pressures, p , and flows, q , only the *total tree resistance*, a_0 , needs to be computed. Accordingly, the total tree resistance is a function of only the root radius and the total accumulated reduced resistance. In effect, all flows and pressures as well as all branch diameters (except the root diameter, d_0) can be eliminated by recursive formulae. It will be shown that *total tree resistance*, a_0 , is only a function of *total diameter-independent resistance* r_0 and the root diameter, d_0 .

$$\min_{\xi, \eta} V(d_0(\xi, \eta)) \quad (2.3)$$

$$a_0 Q_0 = \Delta P = P_0 - P_t$$

$$a_0 = \frac{r_0}{d_0^4} \quad (2.4)$$

$$d_0 = \left(r_0 \frac{Q_0}{\Delta P} \right)^{1/4} \quad (2.5)$$

Recursive total tree resistance computations. Each connected tree segment i has resistance, α_i . The *cumulative resistance*, a_i , sums the accumulated resistances of its subtree, as in Equation (2.6), where the subscripts i, j , and k indicate the parent, the existing branch of the tree at a given bifurcation, and the newly added daughter branch, see Figure 2.2. Terminal segments with no subtrees have $a_i = \alpha_i$.

$$a_i = \alpha_i + \frac{1}{\frac{1}{a_j} + \frac{1}{a_k}} \quad (2.6)$$

To obtain cumulative tree resistance, a_i , it is advantageous to split it into two separate contributions: the diameter-independent *reduced resistance*, r_i , and the segment diameter, d_i . This choice will bring the benefit of enabling the expression of the entire tree resistance in terms of the root diameter d_0 . *Reduced resistances* also have a cumulative, r_i , and segment component, ρ_i , as defined in Equation (2.7).

$$r_i = \frac{a_i}{d_i^4} \quad \rho_i = \frac{\alpha_i}{d_i^4} \quad (2.7)$$

Fortunately, *cumulative* tree resistances in a balanced tree can be expressed in terms of diameter ratios that only depend on the bifurcation point coordinates, $r_1(\xi, \eta)$. Specifically, a recursive relation for the *reduced resistance*, r_i , can be obtained in terms of diameter ratios β_j, β_k as in Equation (2.8).

$$\begin{aligned}
a_i &= \frac{r_i}{d_i^4} = \alpha_i + \frac{1}{\frac{1}{a_j} + \frac{1}{a_k}} = \frac{\rho_i}{d_i^4} + \left(\frac{d_j^4}{r_j} + \frac{d_k^4}{r_k} \right)^{-1} \\
r_i &= d_i^4 \left(\frac{\rho_i}{d_i^4} + \left(\frac{d_j^4}{r_j} + \frac{d_k^4}{r_k} \right)^{-1} \right) = \rho_i + \left(\frac{d_j^4}{d_i^4} \frac{1}{r_j} + \frac{d_k^4}{d_i^4} \frac{1}{r_k} \right)^{-1} \quad \Bigg| \quad \beta_j = \frac{d_j}{d_i}, \beta_k = \frac{d_k}{d_i} \\
r_i &= \rho_i + \left(\frac{\beta_j^4}{r_j} + \frac{\beta_k^4}{r_k} \right)^{-1}
\end{aligned} \tag{2.8}$$

Next we need a method to compute daughter branch ratios. Parent to daughter branch ratios should obey Murray's law expressed in Equation (2.9). We set $\kappa = 3$ following physiological ranges given in the literature [56,66] and is supported by our data analysis in Section 7.4.

$$d_i^\kappa = d_k^\kappa + d_j^\kappa \tag{2.9}$$

Moreover, tree balancing imposes the condition in Equation (2.10) on the diameter ratios of the daughter branch β_j, β_k . The derivation in Section 2.8 proves that the scalar ratio m is known, because cumulative *diameter-independent* resistances and the number of terminals connected to each daughter branch (N_j and N_k) are set for a segment addition at position (ξ, η) .

$$\begin{aligned}
\beta_j &= (1 + m^\kappa)^{-1/\kappa} \\
\beta_k &= (1 + m^{-\kappa})^{-1/\kappa}
\end{aligned} \tag{2.10}$$

where

$$m = \left(\frac{\rho_k N_k}{\rho_j N_j} \right)^{1/4}$$

With diameter ratios for all segments now known, the cumulative *diameter-independent* resistance for the entire tree, $r_0(\xi, \eta)$, can be recursively computed. Finally, the tree root diameter, d_0 , is found from Equation (2.5) for a desired perfusion flow rate Q_0 , and perfusion pressure, ΔP . The total tree volume, V , is computed by adding up cylindrical segment volumes, using Equation (2.10). Actual daughter segment diameters, d_i , are recursively calculated by multiplying the diameter ratios from the root downwards to segment i as in Equation (2.11), where the index set j signifies the *path* leading from the root segment to segment i .

$$d_i = d_0 \prod_{j=0}^i \beta_j \quad j \in path(i) \quad (2.11)$$

$$\min_{\xi, \eta} V(d_0(\xi, \eta)) = \sum_0^{nSegments} l_i \pi d_i^2 \quad (2.12)$$

The optimal position (ξ^*, η^*) in the bifurcation plane gives the minimum tree volume for a possible segment connection. The tree with the smallest volume among the N trees with structurally different segment connections gives the *optimal segment addition*. Repeated segment additions, each one performing the global optimization process described in Equation (3-12) yields a synthetic blood flow network with the desired number of segments, $nSegments$.

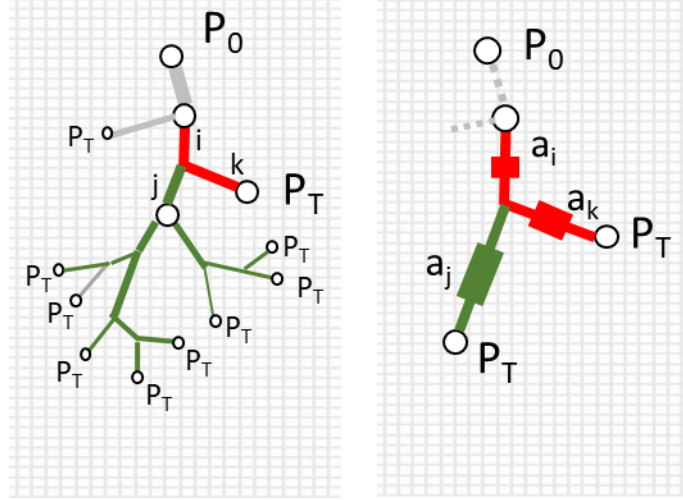


Figure 2.2. Recursive tree resistance computations.

For a segment addition to the existing segment i , the accumulated subtree resistance (a_j) of the split segment, j , can be recursively computed because all terminal nodes discharge at the terminal pressure P_T . (Nomenclature: parent node, i ; existing branch of the tree, j ; newly added daughter branch, k)

Illustrated example of the construction principle for generating vascular trees. Constrained constructive segment addition discovered by Karch and Schreiner is illustrated with the help of Figure 2.1. *Initiation.* Define a cylindrical segment R_0-T_1 with vascular lumen determined by its length and diameter. Next, randomly chose end point coordinates for a new branch, (terminal node T_2). Tentatively connect T_2 to the middle of the initial segment at location Bif. The simple vascular tree has now three segments (R_0 -Bif, Bif- T_1 , and Bif- T_2). All diameters are precisely computed to *balance* the tree, a physiological condition that will ensure uniform tissue perfusion by discharging exactly the same blood flow in the terminal nodes (T_1 and T_2). Moreover, Murray's law sets parent to daughter diameter ratios [67]. Next, move the bifurcation coordinates, $\text{Bif}(\xi, \eta)$, to minimize the tree volume, while maintaining the tree *balanced* by segment radii adjustments. The optimal position, $\text{Bif}(\xi^*, \eta^*)$, gives the desired minimal volume tree.

Continuation. Next, begin another segment conveying blood to a new random terminal location, T_2 . From T_2 , three structurally different connections each one to another existing tree segment are possible. From among the three optimized candidate trees, the global minimum is chosen. Repeated segment addition gradually builds large *balanced* vascular trees with minimum volume.

2.2.2 Image-guided segment addition

The original CCO is well suited for synthesizing space-filling trees, but specific anatomical configurations for organs with complex anatomy are not amenable to constructive growth. For example, the special arrangement of arteries in the Circle of Willis need not be *synthesized*. Instead, it is more practical to incorporate specific anatomical configurations directly from neuroimage or anatomical atlas data. Our new construction methodology has two options to integrate anatomical information:

- Growth from backbone
- Physiological constraints from neuroimages (sample-guided segment addition)

Growth from backbone. This paradigm enables the initiation of vascular synthesis from an existing binary tree (*backbone*) with known connectivity and dimensions (point coordinates, segment length and diameters obtained from an image). Fine-grained additional segments can be added by tree minimization with backbone geometry remaining fixed. If desired, original diameter measurements obtained from image segmentation software [68,69] can be imposed on the backbone segments after the growth algorithm has completed.

Physiological constraints from neuroimages (sample-guided segment addition). We propose to enforce physiological constraints by *SampleGenerator* operators that precisely control subspaces where anatomical segment growth should occur. Using well-established computational meshing methods, arbitrarily complex shapes can be precisely delineated.

2.2.3 Synthesis of closed networks (=microvascular closure)

Constructive synthesis of tree-like structures breaks down for microcirculatory networks, because loops and anastomoses break the binary tree logic. The required tree lumen minimization cannot be performed recursively, so that segment addition becomes an intractable combinatorial optimization problem. Thus, CCO by Karch and Schreiner is not able to create circulatory networks, which connect the arterial side to the venous circulation.

To overcome the limitation of CCO, we introduce a novel microvascular closure. We create *circulatory networks* whose arterial side connects to the venous circulation through a physiologically consistent capillary bed. A *TerminalNodeSampleGenerator* directs segment formation between each open arterial terminal (=nodes at the precapillary and capillary level) to a nearby venous segment (=near terminal venous segments). The pseudocode listed in Section 2.9 naturally “grows” contiguous connections between the arterial and venous trees. The microcirculatory closure produces circulatory models with arterial and venous trees linked by a network-like capillary water shed region. When all arterial terminals are attached to the venous segments (=connected), the process is repeated for open venous terminals.

2.2.4 Neuroimage data for validation of synthetic growth

We based main anatomical features of the large arteries and veins on data acquired with μ CT [63,70] and mouse atlases [71]. Microvascular network growth was validated with metrics from two-photon laser scanning microscopy (2PLSM) data acquired previously [41]. More details on data acquisition are given in Section 4.2.2.

2.3 Implementation

This section provides guidelines for implementing the proposed algorithmic framework on existing computer hardware. An overview of the information flow diagram and pseudocodes are given in Section 7.1.4.

Binary tree representation. Vascular trees can be conveniently encoded as binary graphs. An object-oriented implementation should include two key attributes: An *adjacency matrix* (faceMx, C₁) whose row indices correspond to vascular segment indices, and row entries holding segment point indices (=two element integer array of point indices). The *point coordinate matrix* (ptCoordMx) is a double precision matrix where each row encodes the point coordinates of the corresponding point (=three element array of double precision). The constructor allocates contiguous memory blocks for a desired number of segment additions, with *Npoints* and *Nsegments* serving as counters for the number of node and vascular segments, respectively. The binary graph class also has iterators to navigate to binary tree structures recursively: traverse tree upwards from terminal node, traverse tree downwards from root node.

Sample generators. Sample generators are *functors* that control terminal nodal positions during segment addition. A base class implementation of a *TriangleSampleGenerator* is given in Section 2.10. *GeometricGenerators* produce random coordinates confined to 2D surfaces or three-

dimensional volumes (analytical functions, Rectangle, Circle, Slab, Cube, Cylinder sampler). *MeshSampleGenerator* issues sample points from anatomical surfaces (STL files), or within the cortical volume (3D Cartesian, tetrahedral unstructured or hexahedral structured meshes [34]). Computational meshes can be reconstructed from neuroimage data [68,69]. *SampleGenerators* can also hold a hard coded (predefined), editable list of sample coordinates. For example, the *TerminalNodeSampleGenerator* supplies coordinates of terminals of arterial and venous sections of an emerging (=still unconnected) vascular network. Pseudocodes for a selection of sample generators are listed in Section 2.10.

Strategy factory class. The vascular strategy follows the *factory* design pattern. It owns all the data structures (binary arterial and venous trees) and provides the application with control over different stages of growth (sample generators, and constraint *functors* [72]). It also keeps counters for the current and desired number of segments additions.

Anatomical constraints in arterial and venous generation. Arterial trees can be synthesized from a backbone, unsupervised growth on an anatomical surface (cortical surface), or a tissue space (subcortical gray matter) based on 3D image data. In each case, *SampleGenerators* can be customized to the needs of the respective anatomy. Venous trees can be created with the same methods as arterial trees when observing lower perfusion pressures ($\Delta P_A=40-60$ mmHg, $\Delta P_V=10-30$ mmHg) which produces thicker (=lower resistance) branches for the same perfusion rate, Q_0 . Since arterial and venous trees inhabit the same tissue space, collisions between the main branches or individual sections should be avoided.

In case a new sample is too close to a prior terminal, it can be discarded and replaced with a new random point. Specifically, terminal node duplication between the arterial and venous sides is prevented by a vicinity test (*is-close procedure*). Significant performance enhancements can be

achieved when supplying properly spaced sample sets a-priori (*SampleGeneratorFromPointList*), instead of performing vicinity search during each step of the evolving network.

Topological constraints. Constraint enforcement provides control over the candidate tree topology after structural optimization. In case the best candidate tree violates a structural constraint (for example, segment intersection), it is removed from the candidate list and the next best solution in the stack is examined. Multiple exclusion constraints can be enforced or combined with AND or OR logic. Topological constraints avoid segment collisions between previously generated branches or to all segments of a complimentary venous tree. Useful implementations include “segments too long”, “too short”, “angle too blunt”, “too acute”, etc...

Microcirculatory closure strategy. The crucial innovation of the proposed microvascular closure rests on two advantageous implementation ideas: (i) deployment of *TerminalNodeSampleGenerator* that guarantees closure segments find all open terminals in arterial and venous trees and (ii) maintenance of binary tree logic of arterial and venous networks during closure. The *TerminalNodeSampleGenerator* uses an emptying index list to avoid repetitious connections to the same terminal. We usually prefer to connect one segment of the arterial tree followed by one segment in venous tree, so that tree growth and diameter updates are spatially balanced. If desired, arterial and venous trees can be fused into a single network for graphical display or further computational purposes.

For a realistic microcirculatory network, it is necessary to adjust tortuosity, which can be achieved by a Bezier Spline approach introduced in previous work [62]. More details on microcirculation can be found in part II of this series.

2.4 Applications and Results

2.4.1 Simple application of open arterial trees.

Figure 2.3 demonstrates the versatility of confining vascular growth to arbitrary shapes. For example, Figure 2.3A shows the letters LPPD - the acronym of our lab - serving as a highly irregular template to grow balanced arterial trees covering the letter-shaped domain. Note that synthetic arterial trees entwining each letter are perfectly balanced, so that all terminals discharge equal amounts of blood. The circular triangular surface mesh sample generator in Figure 2.3B confines growth to a flat disc. Figure 2.3C depicts a cubic tissue sample with a partial view of the pial surface arteries, the penetrating arterioles, and a few hierarchies of branching arterioles.

2.4.2 Synthesis of closed cortical blood supply with microcirculatory closure

Most prior work produced arterial trees without physiological connection to the microcirculation or the venous drainage. This case study illustrates the synthesis of the complete circulation of the somatosensory cortex in mouse. It proceeds through five stages with outcomes presented in Figure 2.4 and pseudocode given in Section 7.1.4. In step-1, the leptomeningeal arteries and veins are synthesized with methods described in Section 2.2. To ensure anatomically consistent surface coverage by leptomeningeal blood vessels, we control the number of penetrating arteries [41] in the strategy (here 12-14 penetrating arteries per mm^2). The number of veins was set between 13 and 39 veins per mm^2 in accordance to morphometric data acquired from the 2PLSM data, which also agrees with prior literature values [41,73]. The cortical synthesis is informed by a database of values obtained from the open literature with key entries listed in Table 2.5. Step-2 ensures that the territories of the main leptomeningeal trunks are orientated perpendicular to the cortical surface. We use *LineNumberGenerator* that create endpoints vertical

to the cortical surface up to a depth of 1mm. The *addSegment* procedure attaches a straight vertical segment at a specific arterial pial terminal. In step-3, subcortical arterial segments grow off main penetrating vessels by volume minimization until a segment count of about half of the total segment number is reached. Note, the penetrating arterioles and ascending venules branch in a tree-like fashion into the capillary bed. In step-4, the incomplete, not yet confluent arterial and venous sides of the network are sequentially connected by the microvascular closure. From each terminal arterial node, exactly one segment is grown to a close micro-segment of the venous side. In analogue fashion, venous terminals are grown to near arterial capillary segments, step-5. Thus, microcirculatory closure adds $N_{\text{terminals}} \times 2$ additional microvascular segments.

Tortuosity. Many microvascular segments do not conform to straight cylindrical shapes but exhibit significant tortuosity. We imposed a tortuosity of $\tau = 1.1-1.81$ for segments in the $d < 150\mu\text{m}$ diameter range to create realistic microvessels. Anatomical consistency was imposed by matching the cumulative density functions of the synthetic and corresponding empirical network. Tortuosity was added to the synthetic networks within each bin until the two CDFs agreed.

Validation. Visual inspection of Figure 2.4 shows a striking structural similarity between synthetic networks and microcirculatory images (2PLSM) at all hierarchical levels for length, orientation and diameters of pial arteries, to number and direction of penetrating arteries and the connectivity of the tortuous capillary network. Also the venous subnetworks seem almost indistinguishable.

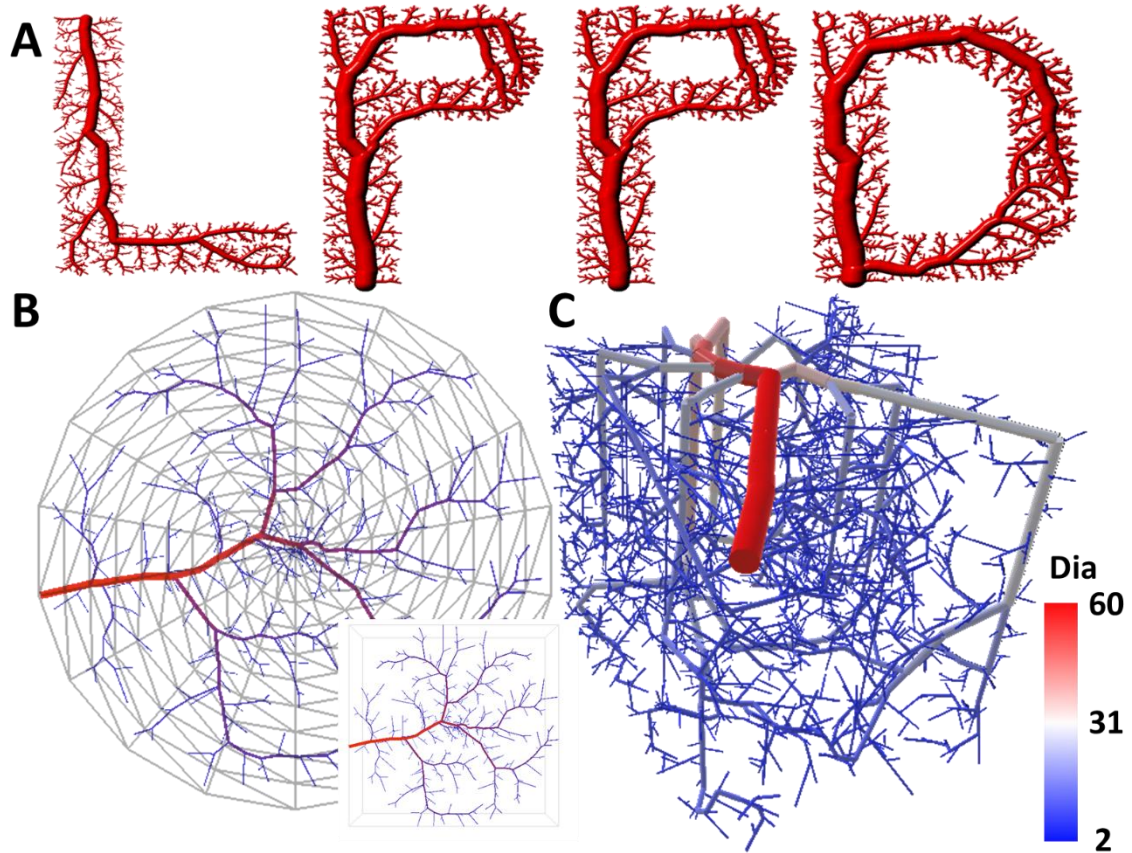


Figure 2.3. Synthesis of open arterial structures with sample generators.

(A) To show the versatility of the algorithm, the letters LPPD- the acronym of our lab – served as a highly irregular template to grow balanced arterial trees covering the letter-shaped domain. Note that each terminal node discharges exactly the same amount of fluid. (B) A flat arterial tree was created on a disk using a two-dimensional triangular mesh. (C) A snapshot of an incomplete subcortical arterial network grown with the help of a 3D unstructured *CarthesianMeshSampleGenerator*.

Table 2.2. Physiological parameters necessary for anatomical growth in mouse.

Parameter	Value	Units	Source
Penetrating arterioles (PA) density	12-13	Nsgm/mm ²	[41]
Ascending venule (AV) density	13-39	Nsgm/mm ²	[41]
PA/AV ratio	3.0	--	[41]
MCA root diameter	143 ± 8	μm	[74]
ACA root diameter	138 ± 9	μm	[74]
PCA root diameter	121 ± 6	μm	[74]
Brain volume	453 ± 19	mm ³	[75]
	509 ± 23	mm ³	[76]
	415 ± 24	mm ³	[77]
Sagittal Length	10-14	mm	[77-80]
Axial Height	8-10	mm	[77-79]
Coronal Width	5-6	mm	[77-80]
Cortical surface area	380 ± 20	mm ²	[75]
	348 ± 3	mm ²	[76]
Density of splined segments (Nsgm)	11,474 ± 1,216	Nsgm/mm ³	[41]
Cortical Thickness	1139	μm	[41]
	1154 ± 7	μm	[81]
	1210	μm	[82]

We also performed detailed statistical analysis of critical morphometric properties in the 2PLSM and synthetic networks. Figure 2.4 show very close statistical agreement between 80 synthetic networks and four 2PLSM data as measured by the cumulative density functions (CDF) of diameter, length, volume and surface area. In fact, the variation between the four experimental datasets is larger than the difference between synthetic and matching experimental datasets. Moreover, the total count of segments (Nsgm) belonging to pial surface arteries, arterioles, capillaries, venules, and veins are virtually the same. Figure 2.4H summarizes additional similarity metrics characterizing critical properties of the entire network (=cumulative structural properties). Specifically, the total vascular length, cumulative vascular volume, and total endothelial surface area show excellent agreement between the synthetic and empirical networks.

Taken together, network connectivity (=assessed by visual inspection) as well as statistical comparison at the segment level (= CDF for segment-to-segment comparison) and total network level (= cumulative properties overall sample comparison) indicate that the iCNS is capable of synthesizing artificial microcirculatory networks that match morphometrics of microimaging

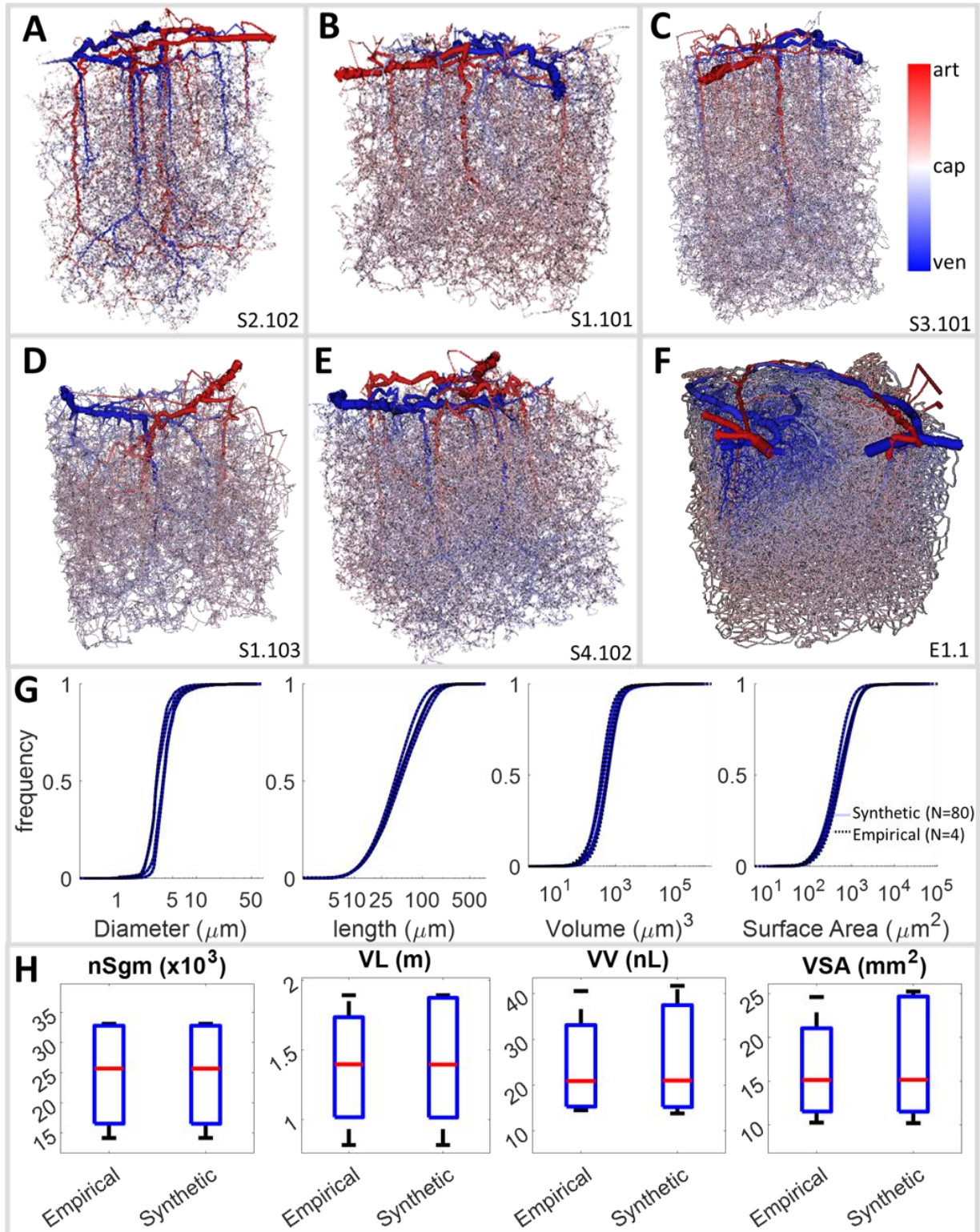


Figure 2.4. A collection of synthetic microcirculatory networks of the somatosensory cortex in mouse.

(A)-(E) show synthetic structures (S2.102, S1.101, S3.101, S1.103 and S4.102) that were generated to match 2PLSM samples with similar morphometric statistics [41]. Note, red indicates arteries, blue corresponds to veins and white shows capillaries. (G) The statistical

properties (CDFs of diameter, length, volume and surface area) of experimental (N=4, dotted black line) and synthetic (N=80, transparent blue line) datasets are reasonably aligned. For comparison, frame F shows an original 2PLSM dataset (E1.1), which looks similar to the synthetic ones. (H) The cumulative statistics for the empirical and synthetic networks are comparable (comparison between number of total segment count, nSgm, accumulated vascular length, VL, total vascular volume, VV, and total endothelial surface area, VSA).

(=2PLSM) counterparts. The statistical analysis suggest that the 80 synthetic datasets are statistically equivalent to the four 2PLSM counterparts.

2.4.3 Mathematical network model of the entire MCA territory

Arterial circulation. The cortical surface growth was achieved by a *SurfaceMeshSampler* customized to delineate the mouse cortex (STL surface mesh generated from a mouse atlas). First, the M1 segment of the middle cerebral artery and the main orientation of the main branches were established using *growth from backbone* as depicted in Figure 2.5. Initially, new arteries were restricted to a close neighborhood of existing arterial branches. This strategy avoided pial arteries cutting into the curved cortical surface, which is not physiological. Optionally, surface adherence of the leptomeningeal segments can be improved by normal projection of the bifurcation points (Bif) to the cortical surface. We precisely controlled the number of pial terminals [41] that connect to penetrating arteries by generating 12-14 terminals per mm^2 on the triangular elements of the cortical *SurfaceMeshSampler*. Note, all sample generators with the exception of the tetrahedral sample generator (not used in the present work) are shown to exhibit no signs of sample bias (see Section 7.2 for more details).

For the venous circulation of the MCA territory, we departed from the superior sagittal sinus backbone and traversed in reverse direction of the blood flow towards cortical bridging veins. The backbone also contained anatomical information of the Transverse sinuses (Trs), Caudal rhinal

vein (Crhv), and Rhostral rhinal vein (Rrhv), see Figure 2.6A. During segment addition, collision avoidance was enforced with *Topological constraint functors*.

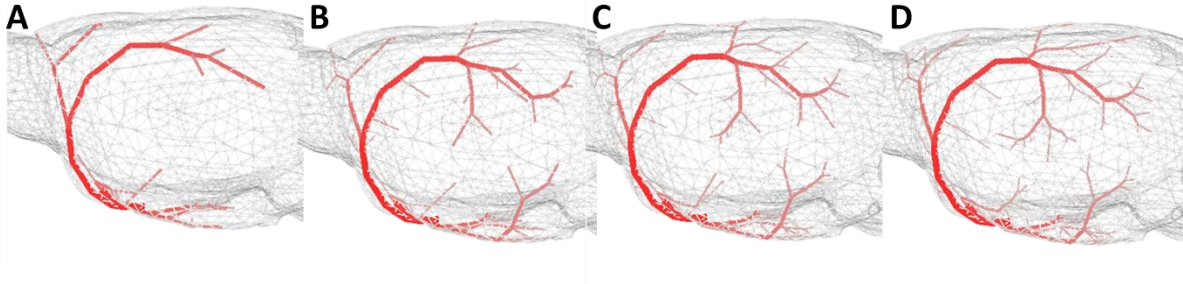


Figure 2.5. Four stages of the pial leptomenengial arterial network growth in mouse. The arterial growth was guided by a triangular mesh surface sample generator that delineated the cortical surface, which was reconstructed from in vivo microCT images [63,70]. The *SurfaceMeshSampleGenerator* uses integer random numbers to generate terminal node samples at the center of STL surface mesh triangles. The smallest pial arteries feeding into penetrating arteries are evenly distributed and discharge the same amount of flow.

The cortical microvasculature was synthesized as described in Section 4.2, but with microcirculatory branch terminal positions drawn by a *VolumeSampler* which penetrated 1 mm into the subcortical tissue. The final network encompasses the complete arterial and venous left middle cerebral artery (LMCA) territory with microcirculatory closure. It covers an area 40 times larger than the 2PLSM data sets. This network spanning the entire MCA territory of the first sample, SMCA1.101, totaled 11,092 arteries, 11,206 veins, and 433,042 capillaries (11,537 segments per mm^3). The statistics of a second MCA territory, SMCA2.101, is given in Table 2.3. Segment numbers and dimensions agree with our 2PLSM data as well as prior literature data listed in Table 2.5. The synthesis took less than 2 CPU hours and 6 GB of RAM on a personal computer.

2.4.4 Cortical blood supply (whole brain circulation)

Figure 2.6A displays the intricate circulatory connections from a detailed mouse atlas [71], which we aimed to recreate with a concise mathematical network model. Arterial trees. For the arterial side, orientation and anatomical connectivity of the first few segments of the anterior (LACA), posterior (LPCA) and middle (LMCA) cerebral arteries were supplied as arterial *backbones*, see Figure 2.7A. The surface mesh generators were encoded by an STL surface mesh shown in Figure 2.6B. The completed synthetic arterial networks are depicted in Figure 2.6C and D. More detailed views of the expanded LACA, LMCA, LPCA territories are depicted in Figure 2.7B. The pial networks were constructed with the methods of Section 2.2, but with *SurfaceSampleGenerators* guiding growth along the tilted surfaces of the ACA, MCA and PCA territories. Venous trees. Venous system synthesis for the whole brain required a few adaptations. The venous trees were grown in reverse flow direction. The completed venous networks are depicted in Figure 6D. Several backbones of the venous circulation depicted in Figure 2.7C were assembled from segmented image data. Microcirculation. The microcirculation was created as discussed on Section 2.3. At the smallest length scale, the thinnest microvessels and closure segments are relatively short (of $d=1-3\mu\text{m}$, $L<50\mu\text{m}$). For the microscale, it is possible to accelerate performance by replacing the volume optimization objective with a simple nearest segment objective.

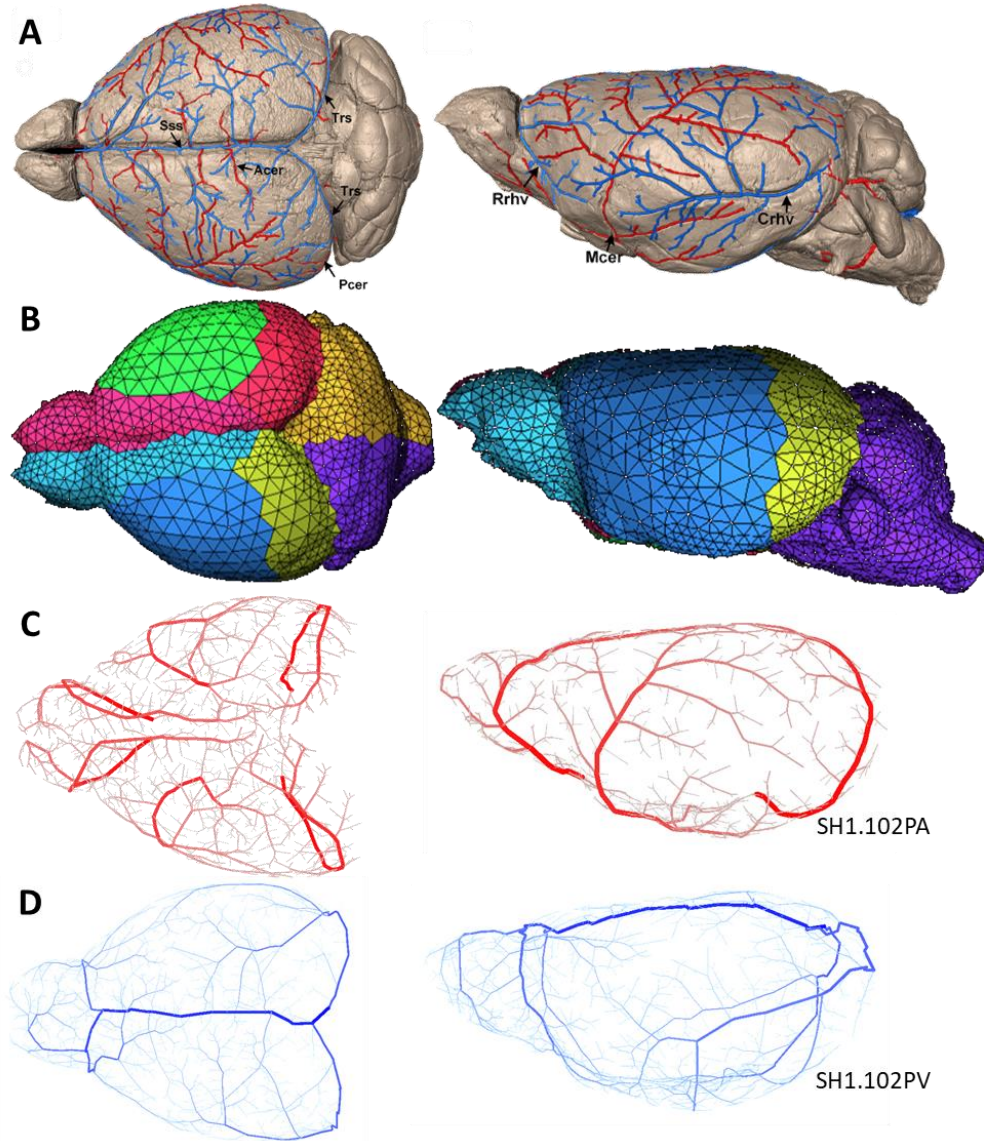


Figure 2.6. Comparison of Growths with Vascular Atlas.

(A) Arteries (red) and veins from the Xiong mouse atlas [71]. (B) Cortical surfaces supplied by main cerebral arteries. The colors correspond to different brain territories; RMCA (green), RACA (pink), RPCA (red), LACA (light blue), LMCA (dark blue), and LPCA (yellow). (C) Synthetic arterial networks and (D) venous circulation for the entire mouse brain (abbreviations: Transverse sinuses, Trs, Caudal rhinal vein, Crhv, and Rhostral rhinal vein, Rrhv). The denomination of SH1.102 signifies the synthetic hemisphere version 102 grown from mouse 1.

Algorithmic complexity of the methodology. For two mice specimen, the cerebral circulation for the left hemispheres with complete cortical circulation were synthesized. The synthetic growth performed on two specimen-specific pial surfaces required approximately 8 CPU hrs on a personal

computer and less than 8 GB of memory. Anatomical data for the automatic construction and statistical information for the massive computer model are provided in Table 2.3.

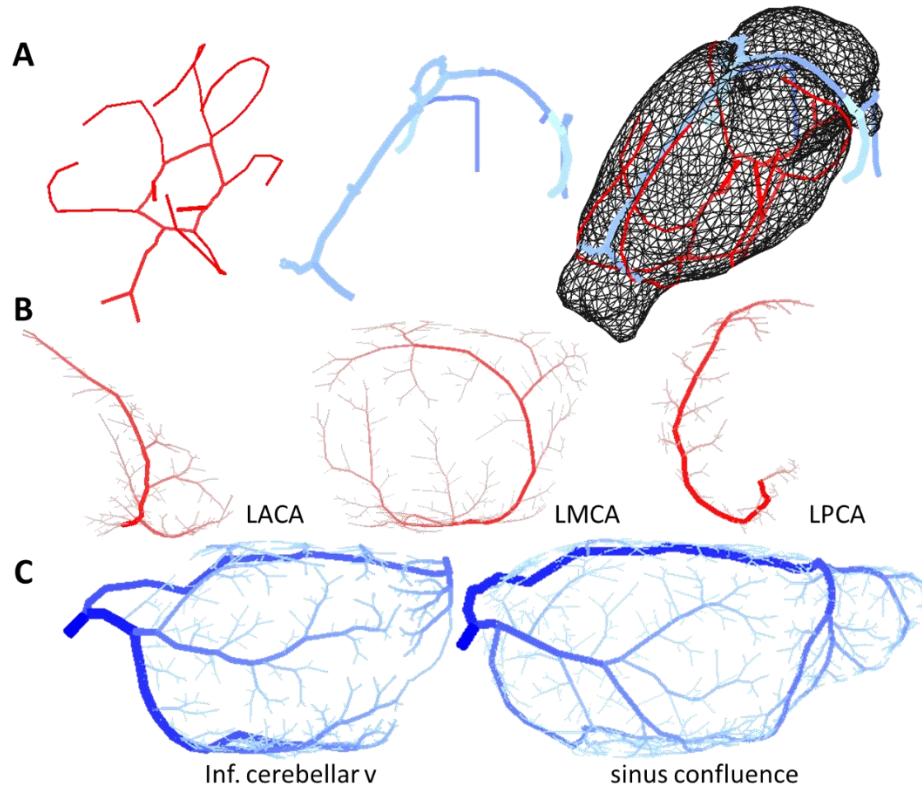


Figure 2.7. Illustration of the pial vascular growth for the mouse hemisphere. (A) Cortical surface and initial arterial/venous backbones. (B) Partially completed pial networks grown from backbones for the LACA, LMCA, and LPCA regions. (C) Synthetically grown territory of the inferior cerebellar vein (partial venous network draining the MCA arteries), and the sinus confluence including the superior sagittal and transverse sinuses. For the arteries of the complete brain, six different backbones are used and later connected to the Circle of Willis.

Table 2.3. Statistics for four large synthetic anatomical network structures, specifically two mouse MCA territories (SMCA1.101, SMCA2.101) and two complete hemispheres (SH1.101, SH2.101)

Parameter	MCAs		Hemispheres		Units	Ref
	SMCA1.101	SMCA2.101	SH1.101	SH2.101		
Penetrating arterioles density	13.00	13.03	13.01	13.03	Nsgm/mm ²	[41]
MCA root diameter	142.5	142.5	142.5	142.5	μm	[74]
ACA root diameter	--	--	138.3	138.3	μm	[74]
PCA root diameter	--	--	120.5	120.5	μm	[74]
SSS root diameter	142.5	142.5	250	250	μm	--
Mouse brain volume	238.2	297.9	238.2	297.9	mm ³	--
Sagittal Length	13.7	13.7	13.7	13.7	mm	[77–80]
Axial Height	8.0	7.9	8.0	7.9	mm	[77–79]
Coronal Width	5.5	5.5	5.5	5.5	mm	[77–80]
Cortical surface area	236.8	281.7	236.8	281.7	mm ²	--
Cortical Depth	1000	1000	1000	1000	μm	--
Density of splined segments (Nsgm)	11,537	11,545	11,518	11,541	Nsgm/mm ³	[41]

2.5 Discussion

We offered a derivation of network synthesis which placed the construction algorithm into a framework of mathematical programming. We introduced a novel methodology to recreate the cortical circulation in the entire mouse brain on a computer. Because anatomical data have limited spatial coverage or resolution, they are insufficient for synthesizing complete mathematical models of functional circulatory networks. This shortcoming was overcome by combining anatomical data with synthetic construction principles. The proposed methodology fills the gaps between data from different length scales pertaining to diverse imaging modalities. Network synthesis can also be used to complement in vivo data sets in the smallest diameter range where imaging data may be unreliable or simply missing. In addition, one can envision applications in which the center of the simulation domain is populated with actual image data but are linked with synthetic microcirculatory models at the boundaries to avoid boundary effects.

We also introduced a novel microvascular closure to connect arterial and venous trees with a realistic capillary bed. A novel *microcirculatory closure* was seamlessly integrated with tree

generation principles. We point out that microcirculatory closure can further be accelerated by replacing the volume minimization objective with the nearest neighbor heuristic.

While this paper focuses on image-based circulatory network synthesis (iCNS), its main purpose is not limited to faithful reproduction of anatomical structures, but aims at supporting mathematical modeling of hemodynamic and metabolic functions in the brain. The synthetic anatomical networks are ideal for performing blood flow simulations with some computational results shown in Figure 3.13. Biphasic simulations of the cerebral blood flow in the left hemisphere took less than 30 CPU min using preconditioned GMRES and required 13.5GB of memory. A description of the mathematical modeling of biphasic blood flow and oxygen exchange is beyond the scope of this paper, but is discussed elsewhere [25,63,83–85]. It is important to note that construction principles enforce simplified hemodynamic constraints as a side condition, which leads to remarkably realistic vascular network structures. Simplified hemodynamic principles used during the synthesis phase do not preclude choices for rigorous blood flow simulations after the vascular networks have been generated. For example, Coutey et al. [86] used constrained constructive optimization with diameter-dependent viscosity.

All synthesis algorithms were implemented on inexpensive computer hardware in serial execution. The run time could be drastically reduced by parallel processing. Parallelization is most beneficial in at least two locations: (i) delegate one candidate tree optimization (=volume minimization) of N -closest segment connections to N processors, (ii) run the list of topological constraints, especially collision tests, in separate threads.

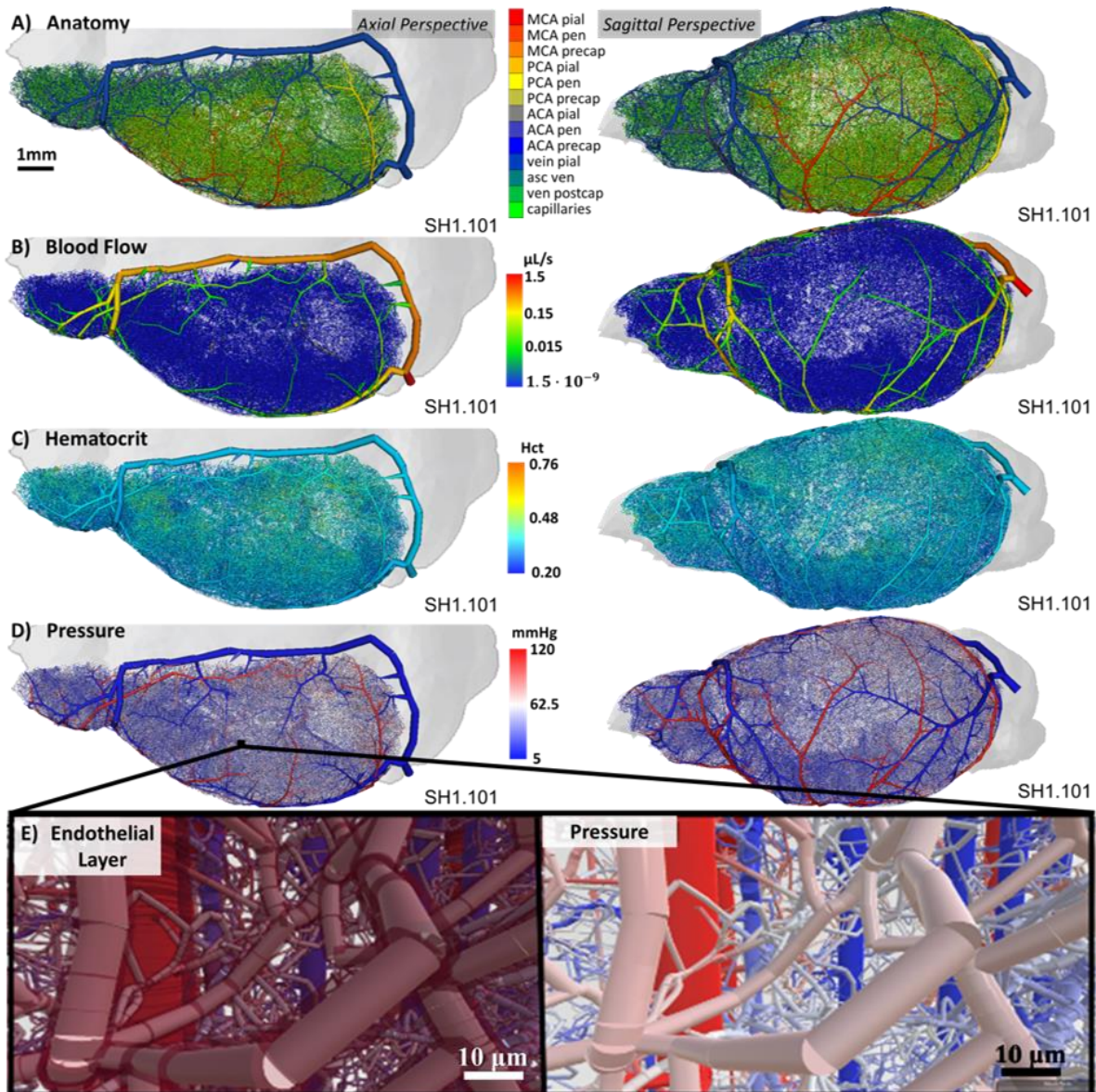


Figure 2.8. Network structure and select simulation results from biphasic blood flow for the complete cerebral hemisphere in mouse.

(A) The anatomical hierarchy of the network encompasses blood vessels including pial arteries, penetrating arterioles, precapillaries, capillaries, post-capillaries, ascending venules and final drainage through the pial veins and sinuses. (abbreviations: MCA pial/pen/precap indicates pial vessels, penetrating arterioles and precapillaries belonging to the MCA territory, respectively. Likewise for ACA and PCA. Ven postcap, asc ven, and ven pial indicate venous post-capillaries, ascending venules and pial veins respectively). (B) Blood flow was simulated by solving the hemodynamic flow-pressure equations for the entire hemisphere in less than 30 CPU min for biphasic blood flow. (C) Simulating blood as a biphasic suspension shows cortical depth dependent hematocrit distribution [63]. (D) The pressure distribution across all territories confirms earlier findings that the largest resistance to blood flow occurs in the microcirculation (E) A zoomed view shows tortuous microvasculature (red-blue) with detailed rendering of the endothelial cell layer (vessel wall)

and perivascular spaces (pink). (Note, the segments are rendered as open cylinders because closed parametric surface meshing is computationally very expensive).

2.6 Limitations.

In the case studies, morphometric parameters of the growth algorithm of the entire cortical surface were set equal to somatosensory cortex statistics, for which accurate 2PLSM data were available. Several other groups have also produced microanatomical data, but these datasets are not publically available [25,85]. Regional differences in brain anatomy are expected, but these can readily be incorporated by adjusting vessel density and topological constraints, thus not limiting the proposed methodology.

We also noticed the occurrence of very small diameters close to the imaging threshold of the four 2PLSM data sets [41] (some segments have $d < 3$ μm); other authors working with the same data scale the original diameter information to avoid very small diameter occurrence [39]. Since this is a limitation of the diameter reconstruction in the original 2PLSM acquisition and not of the synthesis method, we chose not to alter the original diameters.

For the growth algorithm, we chose constant perfusion pressure and equal terminal node pressure, which caused uniform perfusion of the tissue. Previous studies experimented with variable terminal pressures or flows using probabilistic arguments [56], but the relationship between these more elaborate choices and the impact they had on the structure was not obvious.

2.7 Conclusions

High resolution image data and construction principles were incorporated in an algorithmic framework for synthesizing anatomically detailed models of the cortical blood supply in the mouse brain. The approach combined the advantages of image segmentation with synthetic network

generation. The proposed principles are adaptable for circulatory network generation in other organs. The flexibility for incorporating image data, controlling topological growth with sample generators and the ability to synthesize microcirculatory closures should make the proposed iCNS methodology suitable for other complex anatomical spaces that occur in lung, heart, or kidney.

Mathematical models of cerebral circulation that incorporate anatomically sound morphometric properties are expected to help address open questions regarding blood flow control after neuronal firing (=functional hyperemia), resilience exhibited after stroke (=vascular reserve and collateral blood supply) and autoregulation. The brain-wide scope of the synthetic cortical circulatory networks with anatomically consistent representation of blood vessels spanning multiple length scales is an essential milestone towards computer models able to render mechanistic insights concerning the brain's remarkable adaptability. Based on the encouraging results for the mouse brain, the final goal of creating predictive mathematical models of the *human brain* seems within the grasp of approaches presented in this paper.

2.8 Derivation of flow ratios in balanced tree

Recall that the definition of cumulative resistances in Equation (2.13) and Murray's law in Equation (2.14)

$$r_i = \rho_i + \frac{1}{\frac{\beta_j^4}{r_j} + \frac{\beta_k^4}{r_k}} \quad (2.13)$$

$$d_i^\kappa = d_k^\kappa + d_j^\kappa \quad (2.14)$$

$$1 = \beta_k^\kappa + \beta_j^\kappa$$

At a given bifurcation, the following relationship holds between the flow rates in each daughter branch of a balanced tree.

$$\begin{aligned} a_j Q_j &= P_i - P_T \\ a_k Q_k &= P_i - P_T \end{aligned} \tag{2.15}$$

Since each daughter branch may have a subtree supplying N terminal nodes of even terminal flow, where N_i is the number of terminal segments downstream from the i^{th} segment, the ratio expressed in Equation (2.17) between the cumulative resistances must hold.

$$Q_j = N_j q \tag{2.16}$$

$$Q_k = N_k q$$

$$\frac{a_j N_j}{a_k N_k} = 1 \tag{2.17}$$

This required ratio imposes a condition on diameter ratios in the two daughter branches as in Equation (2.18).

$$\frac{r_j d_k^4 N_j}{d_j^4 r_k N_k} = 1 \tag{2.18}$$

$$\frac{\beta_k^4 r_j N_j}{\beta_j^4 r_k N_k} = 1$$

We introduce the scalar, m , which is known for a tree with given segment lengths (=in terms of its diameter-independent resistances and number of nodes in terminal subtrees). Its value dictates the necessary diameter ratios in any bifurcation of the balanced tree as in Equation (2.19).

$$\beta_k = m\beta_j$$

Where

$$m = \left(\frac{\rho_k N_k}{\rho_j N_j} \right)^{1/4} \quad (2.19)$$

Substituting this daughter branch relation into Murray's law gives the desired formula for relative diameters, β_j and β_k in Equation (2.20).

$$\begin{aligned} 1 &= \beta_j^\kappa + (m\beta_j)^\kappa \\ \beta_j &= (1 + m^\kappa)^{-1/\kappa} \end{aligned} \quad (2.20)$$

For the second branch, we obtain

$$\begin{aligned} \beta_k &= m\beta_j = m(1 + m^\kappa)^{-1/\kappa} = (m^{-\kappa}[1 + m^\kappa])^{-1/\kappa} = (m^{-\kappa} + 1)^{-1/\kappa} \\ \beta_k &= (1 + m^{-\kappa})^{-1/\kappa} \end{aligned} \quad (2.21)$$

2.9 Microvascular closure pseudocode

Table 2.4. Pseudocode for microvascular closure algorithm

1. FOR all terminals aT in TerminalSampleGeneratorList do
2. Choose closest vS \in close_segment_list(venous_tree)
3. add_fork (venous_tree, aT, vS) // with or without optimizing
4. remove(TerminalSampleGeneratorList , aT)
5. balanceTree(venous_tree, d0) // update venous_tree diameter ratios
6. ENDFOR

2.10 Sample generator pseudocodes

Table 2.5. Pseudocode for generating sample points in a triangular mesh

1. function ooTriangularMeshSampler.getSamplePoint(aFaceIdx:integer)
2. u := randomFloat(0,1); v:=randomFloat(0,1)
3. IF u+v > 1 THEN temp = u; u = 1-v; v = 1- temp; // mirror pt if outside triangle
4. [p0,p1,p2] := SurfaceMesh.getPointsTriangle(aFaceIdx)
5. result := plus([p0, scale(u,Vector(p0,p1)), scale(v,Vector(p0,p2))])

Table 2.6. Pseudocode for generating sample points in a triangular mesh

1. function ooTriangularMeshSampler.getSamplePoint(aFaceIdx:integer)
2. u := randomFloat(0,1); v:=randomFloat(0,1)
3. IF u+v > 1 THEN temp = u; u = 1-v; v = 1- temp; // mirror pt if outside triangle
4. [p0,p1,p2] := SurfaceMesh.getPointsTriangle(aFaceIdx)
5. result := plus([p0, scale(u,Vector(p0,p1)), scale(v,Vector(p0,p2))])

Table 2.7. Pseudocode for generating sample points inside a Carthesian bounding box

1. function ooVolumeSampler.getSamplePoint(abb:BoundingBox)
2. xMin=abb[0]; xMax=abb[1]; yMin=abb[2]; yMax=abb[3]; zMin=abb[4];zMax = abb[5];
3. u := randomFloat(0,1); v := randomFloat(0,1); w := randomFloat(0,1);
4. xVal = (xMax-xMin)*u + xMin;
5. yVal = (yMax-yMin)*v + yMin;
6. zVal = (zMax-zMin)*w + zMin;
7. Result := [xVal, yVal, zVal];

3 Image-based circulatory network (iCNS) synthesis Part II: automated matching of network topology

3.1 Introduction

In order to investigate such topics as the aging brain and functional hyperemia, simulations have been able to overcome shortcomings in imaging paradigms. These simulations have been greatly improved by recent advancements in vascular reconstructions leading to new availability of vascular network structures for large portions of the mouse cortex available for simulation. Unfortunately, these networks do not span the entire cortex, leading to artificial computational boundaries at the edge of the imaging window. At these edges the modeler must make choices of how to model this tissue, and this choice directly affects the predictions. To avoid these artificial boundaries, recent advancements in anatomically accurate vascular synthesis allows extrapolation of these empirically-derived structures to the scale of an entire hemisphere (see Section 2).

The previously proposed image-based cerebrovascular network synthesis (iCNS) algorithm (Section 2) builds the microvascular closure from a space-filling sample generation with a novel closure step to connect arterial and venous trees seamlessly. Also proposed was a method for assigning a diameter approximation (Murray's law and setting root radius of each tree). This is a reasonable diameter approximation with a Murray coefficient of ~ 3 as described by empirical network interrogation offered in Section 7.4. While this network shares comparable vascular density with empirical counterparts, it does not mirror the same tortuosity and diameter spectra of empirical networks. For instance, the diameter spectra of empirical data have peaks at the imaging threshold whereas the diameter approximation from Murray's law enforces a smooth diameter spectrum with no sharp peaks.

To improve the topological characteristics, details of the microvascular growth and closure for smaller (non-pial) vessels will be expanded here. An automated method for matching the topology of two networks is investigated. These additions are shown to automatically produce synthetic networks with matching topology to empirical counterparts.

3.2 Methods

3.2.1 Automated topological characteristic matching

In order to align the topology of the synthetic network to the empirical ones, the spectra of diameter and length must be matched. For best results, the tortuosity spectra were matched and the growth constraints were tuned such that the tortuosity and length matched simultaneously. The list of steps given below and an overview of the algorithm is given in Figure 3.2:

1. Agglomerate segments until all segments between bifurcations are considered a single vessel (splining procedure)
2. Reduce tortuosity in all segments to straight lines (start with clean tortuosity spectra)
3. Calculate length and diameter CDF of synthetic and empirical networks
4. Match large vessel (pial arteries/veins) diameter spectra
5. Match small vessel (arterioles/venules and capillaries) diameter spectra
6. Calculate empirical and synthetic tortuosity spectra
7. Calculate tortuosity necessary to match the length spectra (relative difference in length between synthetic network as-is and synthetic network with matching length spectra)
8. Match tortuosity CDF of empirical and synthetic networks

Matching topology

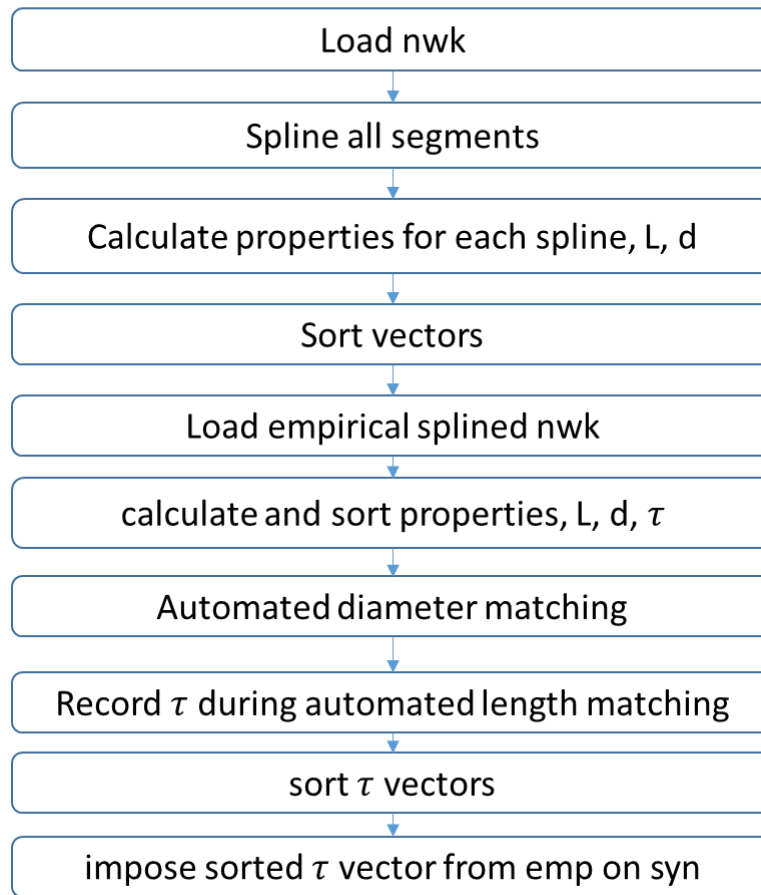


Figure 3.1. Workflow diagram for matching the length and tortuosity spectra of a synthetic network to empirical network.

Note, by matching these properties the length, surface area, and volume spectra also match.

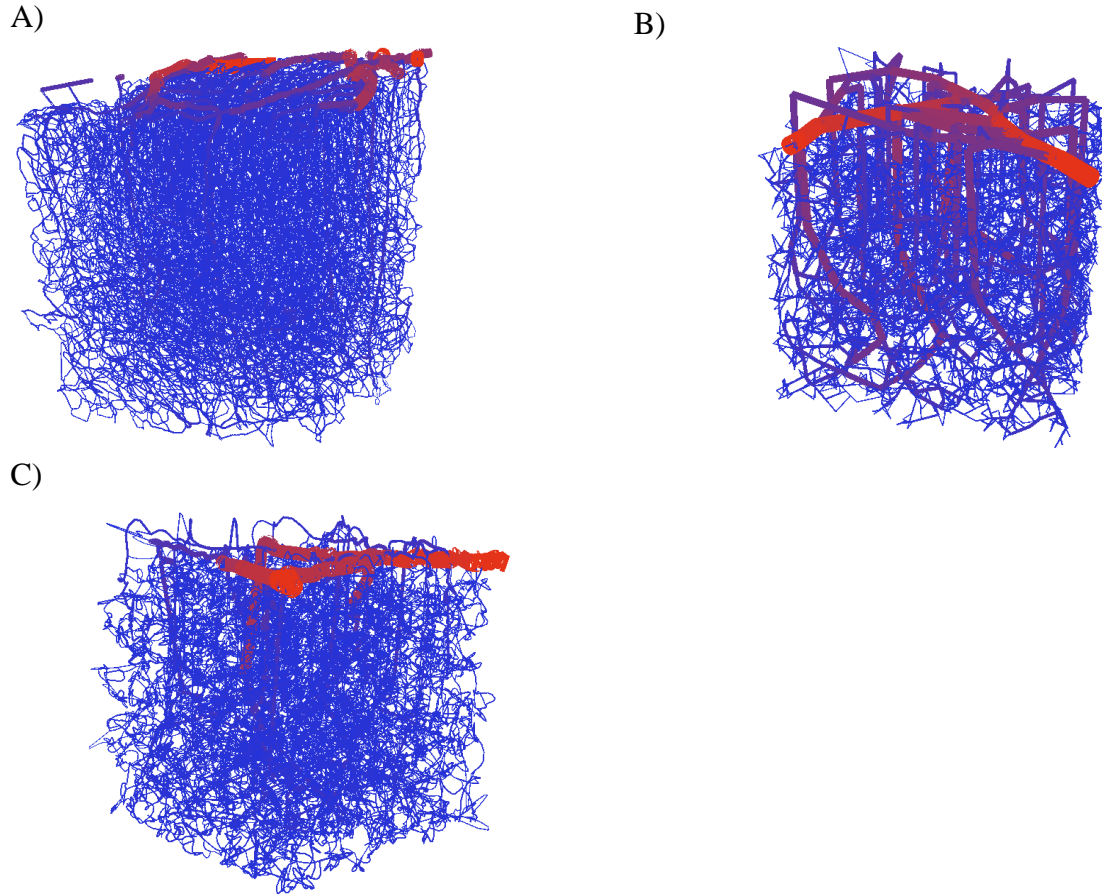


Figure 3.2. Comparison between empirical and synthetic networks before and after adding tortuosity.

(A) empirical, tortuous networks, (B) fully synthetic, straight tree segments and (C) tortuous synthetic networks. Note, tortuosity in this figure was assigned manually.

Step 1, Splining. Splining is the process of grouping all adjacent segments into a single structure between bifurcations. Splining is necessary for comparing vascular structures, as single segments are frequently broken arbitrarily (into more than 1 segment) due to variations in direction or length (see Figure 3.3 for examples). These agglomerated structures can then be represented by a series of 3rd order Bezier curves to give smooth representation of the blood vessel lumen. This new vascular structure is represented by only segments connecting bifurcations to other bifurcations or bifurcations connected to terminals.

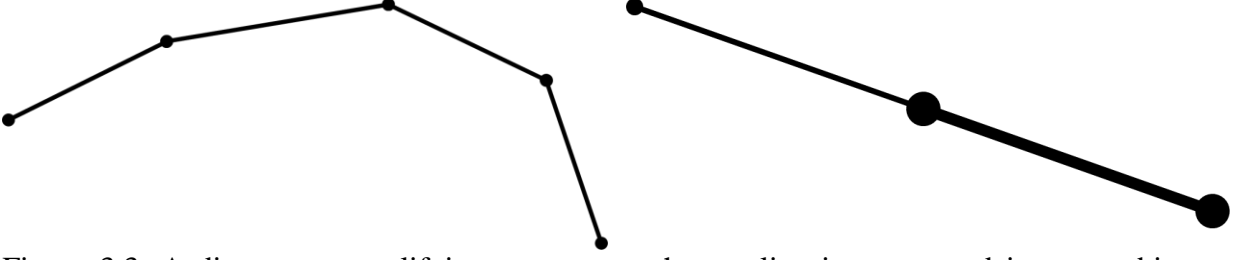


Figure 3.3. A diagram exemplifying two cases where a line is segmented into an arbitrary number of sections.

(Left) a line segment with a bend and (Right) a line segment with varying diameter. While both images can represent a single, continuous blood vessel, they have been cut into an arbitrary number of lines based on diameter and bending (tortuosity) information.

Calculating control/terminal points for each Bezier curve. The Bezier curves must define the control and terminal points from the original network structure. Note, if multiple curves represent the same vessel segment, these curves should adhere to C_0 and C_1 continuity as described elsewhere [34]. These parametric curves are represented by Equation (3.1) with C_0 and C_1 continuity expressed in Equation (3.2) and (3.3), respectively. Note, all equations are given in the x -dimension, but are computed with the same equations in the y and z directions by replacing the x variable with y and z .

$$x(t) = P_0^x t^3 + (1-t)t^2 C_0^x + (1-t)^2 t C_1^x + (1-t)^3 P_1^x \quad t \in 0..1 \quad (3.1)$$

$$x(t)_L = x(t)_r \quad (3.2)$$

$$x(1)_{i-1} = x(0)_i$$

$$x'(t)_L = x'(t)_r \quad (3.3)$$

Here P_0^x , C_0^x , C_1^x , and P_1^x are the origin point, 1st control point, 2nd control point, and terminal point of the curve, respectively. A pseudocode for grouping faces into a single vessel is given in

Table 3.1, while the overarching code for converting an entire network into splines is given in Table 3.2.

Table 3.1. Pseudocode for grouping faces from a 2-point network between bifurcations

```

1.  groupSegmentsIntoSplines(nwk):IntArray;
2.  splineIdx = 1; splineIdxPerFace = iVector(nwk.nFaces+1, 0);
3.  FOR i = 1 TO nwk.nFaces DO
4.      IF splineIdxPerFace[i] = 0 THEN continue; //skip face if already labeled
5.      getPointsForFace(p1,p2);
6.      recurseUpToBifurcationAndLabelFaces(p1, splineIdx);
7.      recurseDownToBifurcationAndLabelFaces(p2, splineIdx);
8.      splineIdx = splineIdx + 1;
9.  ENDFOR;
10. result = groupFacesByIdx(splineIdxPerFace); //group all faces with same spline
    index
11. END;
```

Table 3.2. Pseudocode for converting a 2-point network to splined network

```

1.  Convert2PointNetworkToSplinedNetwork(nwk, diaArray):splinedNWK;
2.  groupedFaceArray = groupSegmentsIntoSplines(nwk);
3.  FOR i = 1 TO length(groupedFaceArray)
4.      faceList = groupedFaceArray[i];
5.      aSpline = createSpilneFromFaceList(i, faceList,p0,c0,c1,p1);
6.      splinedNWK.addFace(aSpline);
7.      splinedNWK.setDia(getSubVector(faceList,diaArray);
8.  ENDFOR;
9.  Result = splinedNWK;
10. END;
```

The function *setDia* can use any model for identifying the diameter from a list of values (=list of diameters for the segments prior to grouping). The diameter model could be a constant (=mean value) or a model parameterized in t where the model coefficients can be obtained through linear regression. This fitting model requires a value of t for each diameter recorded. The value of t can be approximated by finding the cumulative length to the cylinder center for the i^{th} segment as in Equation (3.4) where i denotes the i^{th} segment in the list, n is the total number of faces in the list, and t_i is an approximation of t at length l_i .

$$t_i = \frac{\sum_{j=1}^i l_j}{\sum_{k=1}^n l_k} \quad (3.4)$$

Step 2, Reducing tortuosity of segments to straight lines. The penetrating vessel stage and closure stage of the synthesis algorithm generates segments consisting of acute angles in continuations (for an example, see Section 7.6). These segments can have tortuosity values outside the range of empirical reconstructions. The best value for the maximum tortuosity was observed to be a value of 1.0 (straight lines), although a method for reducing the tortuosity to a fixed value is also given in Section 7.6.

The reduction of all segments to straight lines must not apply to the penetrating arterial-pial vessel interface, as that curve is reflected in the empirical networks and does not exceed physiological values. The remainder of the vessels are reduced to straight lines, effectively flattening the tortuosity spectra of the synthetic network.

Step 3, Calculating length, tortuosity, and diameter cumulative density functions. The discrete cumulative density function (CDF) of a parameter list (such as a list of segment lengths or diameters) can be generated by probing and sorting these values from the network. The length of each vascular segment is approximated by segmenting each spline into N-divisions and summing the linear distance between the discrete points. Once the length and diameter vectors are identified, they are sorted from smallest to largest value.

The vectors can now be simplified to CDFs by cycling through the elements of the vector and tracking the iteration number at every new value. The CDF does not generate a new point in the event the adjacent values in the array are the same (if two indices have the same value, that bin in the CDF has 2 elements). To identify if a new value is significantly different than the previous

value, a tolerance must be set, here taken as $\mu = 1e - 9$. The iterator is the non-normalized y-value in the CDF and the vector value is the x-coordinate. After all x and y values for the CDF are calculated, the y-values are normalized by dividing by the length of the array. A pseudocode is offered:

Table 3.3. Pseudocode for calculating a CDF from a sorted value array

```

1.  FUNCTION getCDF(sortedDia);
2.  Counter = 0;
3.  FOR i = 0 TO sortedDia.Length-1 DO
4.    IF (sortedDia[i+1] - sortedDia[i] < tol) THEN
5.      yVector[counter] = I;  xVector = sortedDia[i];
6.    ENDIF
7.  ENDFOR
8.  yVector = Scale(1/Length(sortedDia), yVector);

```

Step 4-5, Matching diameter spectra. The pial vessels are significantly larger than the other microvessels because they carry large quantities of collateral blood to portions of the brain that are outside the imaging domain. Due to the large disparity between the diameter range for the pial vessels and the smaller vessels (arterioles, venules, and capillaries), the diameter spectra for these two ranges should be matched independent of the microvessels.

Matching the pial diameter spectra follows the CDF matching algorithm proposed in Section 3.2.2 but restricts the matching to only the pial diameter spectra in both networks. In the case of datasets with very large penetrating arteries, the penetrating vessels (penetrating arteries and ascending veins) are matched in the pial phase instead of in the microvessel stage. In the event that the empirical data does not include anatomical labelling, N-largest vessels in the empirical diameter spectra are used for pial matching, where N refers to the number of pial vessels in the synthetic network. The capillary diameter spectra are then matched using the remainder of the spectra.

Step 6, Calculating empirical tortuosity spectra. The tortuosity spectra for the synthetic and empirical networks are calculated as defined in Equation (3.5).

$$\tau = \frac{C}{L} \quad (3.5)$$

Here, C is the length along the curve and L is the Euclidean distance between endpoints of the curve. This is expressed graphically in Figure 3.4.

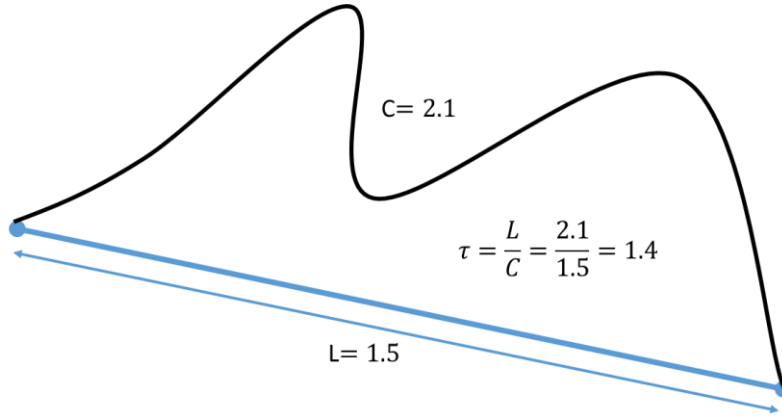


Figure 3.4. Graphical representation of the tortuosity of a line. C represents the distance traversed by the line and L is the Euclidean distance between two points. If the line is a straight line between two points, the value of tortuosity (τ) is 1.

Step 7, Calculate tortuosity necessary to match the length spectra. The method of matching the length spectra of two networks inherently enforces a least-tortuosity matching algorithm, which leads to a non-physiological tortuosity spectrum. A method of informed tortuosity spectra matching was developed to execute the simultaneous matching of length and tortuosity.

When comparing the sorted length spectra (length CDF) between the synthetic and empirical networks, a new variable is created known as the *normalized length difference*. This value is

calculated by finding the difference between the current length (synthetic length) and desired length (empirical length). This value is then divided by the current synthetic length to result in the *normalized length difference*. This gives the amount of tortuosity required to match the length spectra (τ_i):

$$\tau_i = \frac{l_i^e - l_i^s}{l_i^s} + 1 \quad (3.6)$$

Where l_i^e is the empirical face length and l_i^s is the synthetic face length. This new vector (τ_i) is a measure of the necessary tortuosity to match the synthetic network to the empirical one. In order to match the length and tortuosity spectra, this new *required tortuosity* measurement is sorted.

Step 8, matching the tortuosity spectra. After obtaining the sorted empirical tortuosity the synthetic *required tortuosity* spectra, the CDFs of these vectors can be computed. The actual tortuosity spectra can then be matched according to the *required tortuosity* spectra and methods in Section 3.2.2. The method for adding tortuosity to splined segments can be found elsewhere ([62] and in Section 7.6.1).

3.2.2 Automated matching of two cumulative density functions

It is time consuming and inaccurate to match the topological spectra (CDFs) of two networks by hand. Instead, a method for binning two CDFs and matching the values within each bin has been proposed.

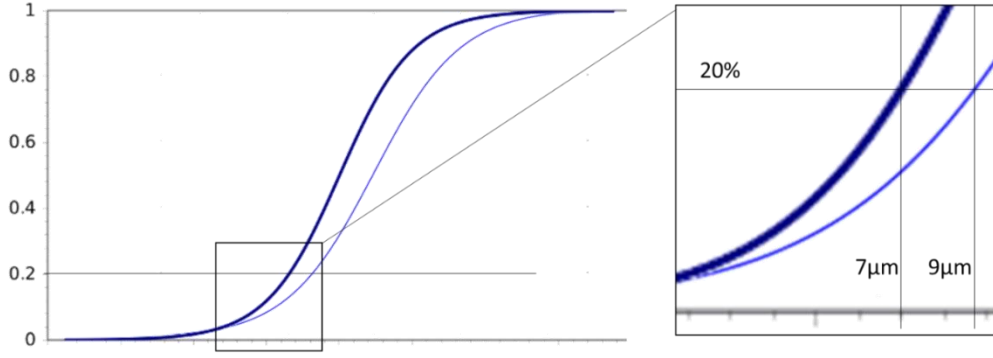


Figure 3.5. Two cumulative density functions that do not match.

The close-up picture highlights 20% of the vessels pertaining to the thick curve have a length of $7\mu\text{m}$, whereas the same number of vessels pertaining to the thin branch have a length of $9\mu\text{m}$. The vessels from the thick-line group would need to be lengthened to match the thin line.

A CDF adjustment can then be applied to every value within the bin to match the original line (thick line in Figure 3.5) to the target line (thin line in Figure 3.5). In this simple example, consider the first 20% of the vessels all within the same bin and consider a constant model for adjustment (single multiplier for every value). To match the CDFs, all vessels in this bin would be multiplied by a factor of $9/7$ in order to make the values larger (to match the thin line). This can be validated by evaluating the test point at the 20% line:

$$\text{error} = \text{source} * \text{modifier} - \text{target}$$

$$\text{error} = 7 \left(\frac{9}{7} \right) - 9 = 0 \quad (3.7)$$

Calculating bins. The bin edges correspond to an index in the sorted array of values of each CDF. The first (v_0) and last (v_1) index of a bin can be found as follows:

$$v_0 = valueA \left[floor \left((iBin - 1) \cdot \frac{nFaces}{nBins} \right) \right] \quad (3.8)$$

$$v_1 = valueA \left[floor \left(iBin \cdot \frac{nFaces}{nBins} \right) \right] \quad (3.9)$$

With the final bin arising from the final value for v_l and ending at $nFaces$.

3.2.2.1 Using a non-constant modifier to match CDFs

To identify the modifier to apply to the elements within a single bin, a first approximation could use a constant factor. Due to the inherent nonlinearity of the sigmoid-like shape that CDFs exhibit, a model with a fixed coefficient is unsuitable. A more advanced method could derive a model that varies within the bin to match all elements within the bin more aggressively. Two models for such a calculation have been identified; (i) using a linear shape function and (ii) using a quadratic function with known y-offset parameter:

$$\phi(x) = m\phi_L + b \quad (3.10)$$

$$\phi(x - \bar{x}) = m\phi_L^2 + \phi_L b + \phi_L \quad (3.11)$$

Here, $\phi(x)$ represents the CDF values (ϕ) at any position along the x -axis. ϕ_L and ϕ_R are the values at the left and right edges of the bin, respectively. The two options give effectively the same result, so method (i) is preferred due to simplicity. Method (ii) is further discussed in Section 7.7. The algorithm begins with two cumulative density functions that initially do not match as in Figure 3.6.

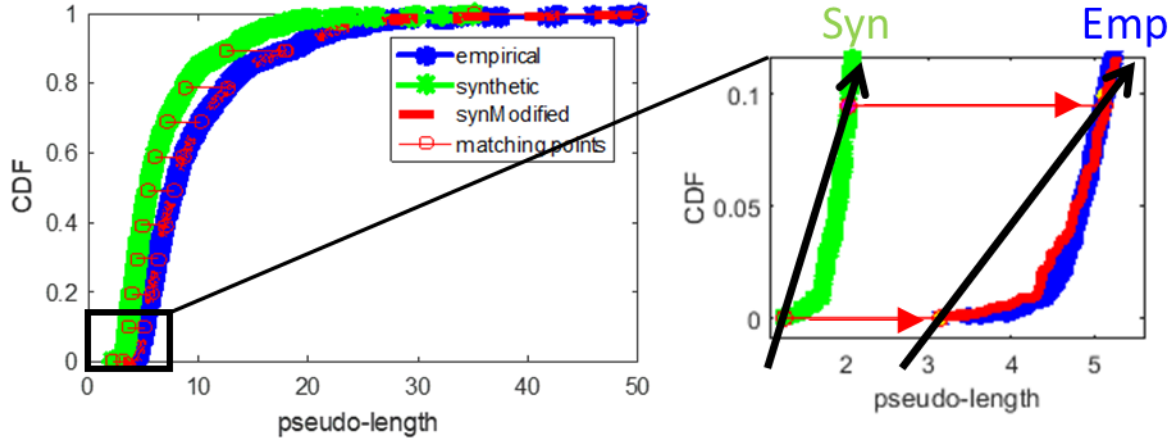


Figure 3.6. CDF matching between two datasets are not the same number of values. The choice of a uniform tortuosity measure is not ideal for this pseudo-length parameter not only because higher order models have a higher number of parameters (automatically giving a better goodness-of-fit) but also because the change of tortuosity within a single bin may not be a fixed value of tortuosity (frequency 0% - 10% is highlighted for reference). This is visualized by black arrows indicating two different slopes (which are bin-dependent). The choice of binning is also important, as the CDF curves are nonlinear, so in areas of high curvature a higher bin density may be necessary.

To update the length CDF using a linear model the following equation must be executed:

$$l' = \tau_0 + \tau_1 \cdot l \quad (3.12)$$

Here, l is the starting length (length of the synthetic network prior to adding tortuosity), and l' is the final length (corresponding length in the empirical network). In this equation, τ_0 and τ_1 are unknowns. As can be seen, there are 2 unknowns, so the system needs 2 equations to be fully determined.

To supply 2 equations, the two bin edges can serve as two independent equations. The start and end of the bin can be derived from Equations (3.8) - (3.9) and serve as the start lengths (l_0 for first index and l_1 for last index in the synthetic bin) and end lengths (l'_0 for first index and l'_1 for last index in the empirical bin).

last index in the empirical bin). When using these 2 data points, the unknown coefficients in Equation (3.12) defines a fully determined system to be solved (assuming the matrix is regular).

$$\begin{aligned} l'_0 &= l_0^{emp} & l'_1 &= l_1^{emp} \\ l_0 &= l_0^{syn} & l_1 &= l_1^{syn} \end{aligned} \tag{3.13}$$

Fortunately, the diameter spectra are easily manipulated. Unfortunately, the length of a Bezier curve is non-analytic. Due to this, the value for the tortuosity factor, α , is not analytically calculable. In order to execute the `setControlPointsForFace(aFace, p0, c0, c1, p1)` procedure, a fixed-point iteration must be used. A code snippet is offered in Matlab in Section 7.7.1.4 along with a case study.

3.3 Applications

3.3.1 Microcirculatory subsections

Characterization of empirical networks. Empirical networks were generously provided for detailed analysis by the Kleinfeld, Boas and Dunn group. These samples showed variability both within the samples from each group and between the groups. The length spectra reflected more similarity between datasets than diameter as reflected in Figure 3.7. The number of segments, tissue domain extent, and volume fractions also showed significant variability between groups. The tortuosity spectra, broken by neuronal layers, reflects similar trends between the datasets as well (Figure 3.8).

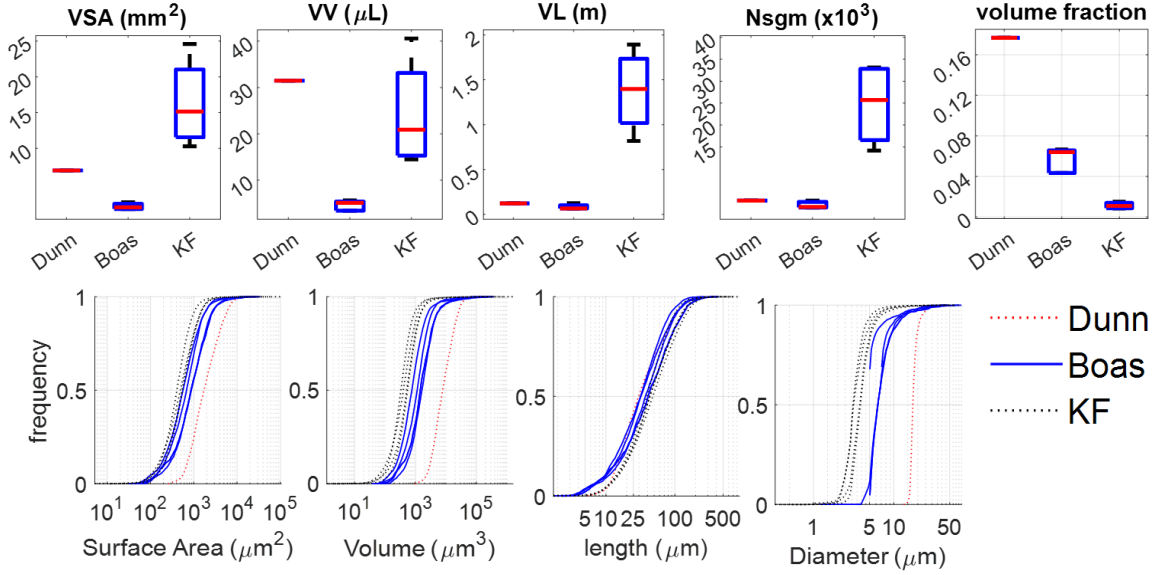


Figure 3.7. Topological comparison between 1 network from the Dunn group, 5 networks from the Boas group and 4 networks from the Kleinfeld group. The sample variability inside each group is large while the sample variability between groups is even larger. Top) overall statistics of the agglomerated network and Bottom) spectra of topological properties are offered.

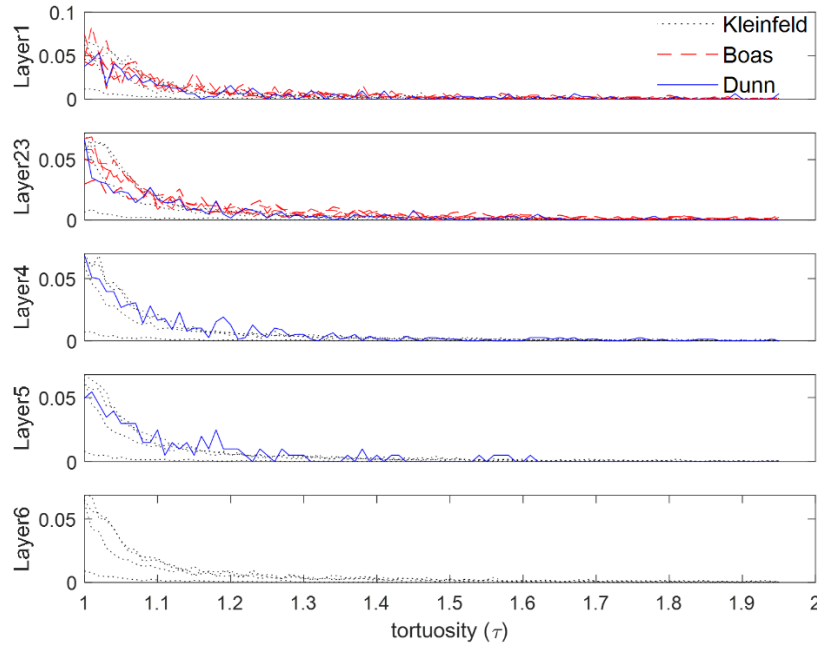


Figure 3.8. Probability density functions of the tortuosity as a function of neuroanl layer in the empirically-derived networks of Boas, Kleinfeld, and Dunn. Not all empirical networks reached the lower layers of the cortical surface and only the Kleinfeld structures penetrated all the way to Layer VI.

The topology of the synthetic structures matches that of the empirical counterparts as expressed by Figure 3.9. The same approach was validated with networks produced by the Boas group (Figure 3.10) and Dunn group (Figure 3.11).

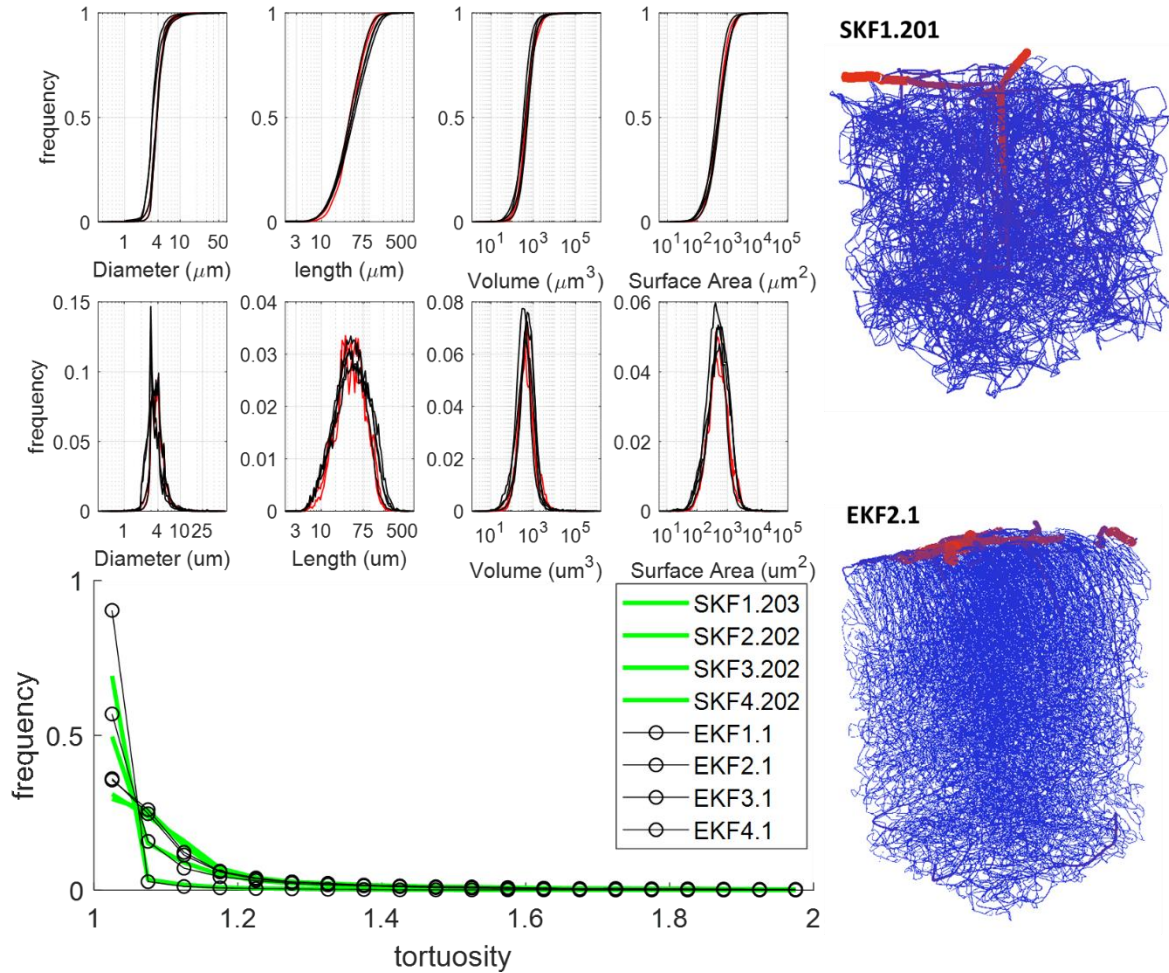


Figure 3.9. Comparison between empirical and synthetic Kleinfeld networks.

The intravariability among the empirical networks is larger than the variability between synthetic and empirical networks. The diameter, length, volume and surface area distributions show excellent alignment between empirical (black lines) and a selection of synthetic (red lines) networks. The tortuosity also shows excellent agreement between empirical (black lines) and synthetic (green lines) datasets. 3D rendering of synthetic and empirical networks are also offered to exemplify similarity.

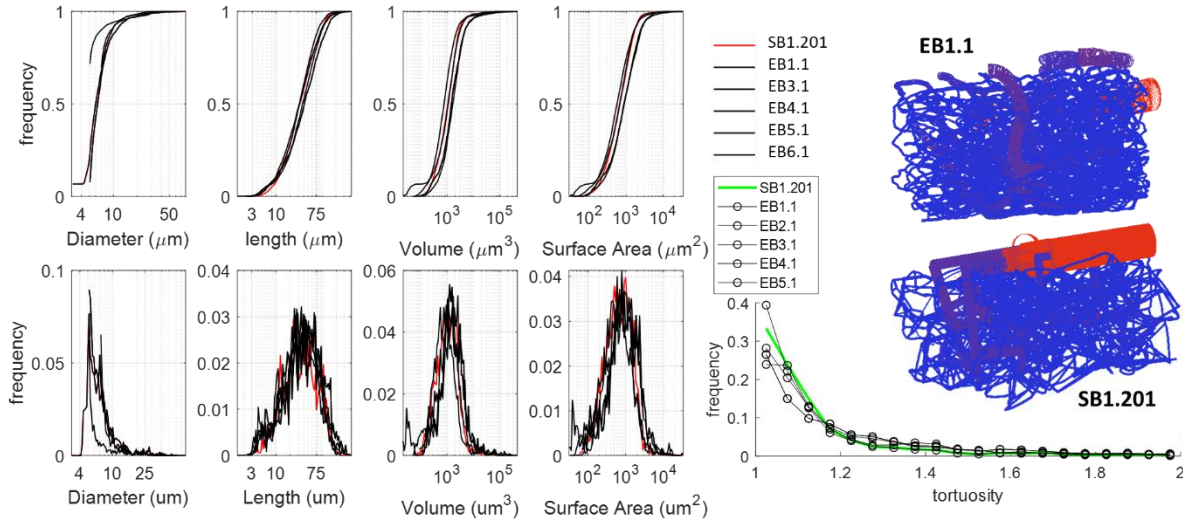


Figure 3.10. Comparison between synthetic and empirical Boas networks.

The intravariability among the empirical networks is larger than the variability between synthetic and empirical networks. The diameter, length, volume and surface area distributions show excellent alignment between empirical (black lines) and a representative synthetic (red lines) networks. The tortuosity also shows excellent agreement between empirical (black lines) and synthetic (green lines) datasets. 3D rendering of synthetic and empirical networks are also offered to exemplify similarity.

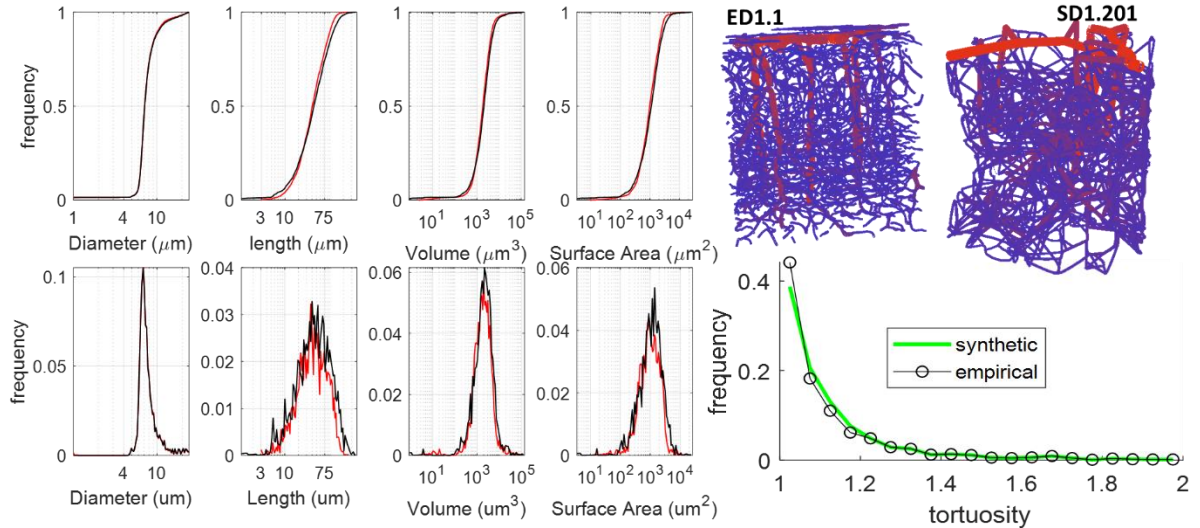


Figure 3.11. Comparison between synthetic and empirical Dunn networks.

The intravariability among the empirical networks is larger than the variability between synthetic and empirical networks. The empirical network is in black while the synthetic network is in red.

3.3.2 MCA/hemisphere

The growth model used for the Kleinfeld-like networks were averaged and applied to the MCA territory of a single mouse and the hemisphere of 2 mice. These growths resulted in the generation of 11 hemispheres and 1 MCA territory using statistics described in Sections 2.4.3 and 4.3.7. These case studies are apt for future investigations of blood flow and collateral reperfusion.

In order to grow such asymmetric trees and territories as exist in the mouse hemisphere, a new method for segment growth was created that relied on growing each tree independently prior to the closure stage, as opposed to simultaneously. This alternative method (called Method 2) grows the arterial tree(s) independently, then grows the venous tree(s) before connecting the two. This is especially important when growing structures like the hemisphere where a single venous tree drains the blood from 3 non-intersecting arterial trees. The overview of Method 2 is offered in Figure 3.12 and a full hemisphere colored by hierarchical labeling is given in Figure 3.13.

To avoid sample bias generated from pial surface mesh indexing, it is imperative to randomly sample from a list of triangles prior to sampling within the triangle. As described elsewhere, this removes inherent bias in the growing procedure [57]. More elaboration on this growth was offered in Section 2.4.4. A full list of completed structures is offered in Section 7.39.8-7.39.9. Implementation details are offered in Section 7.1.

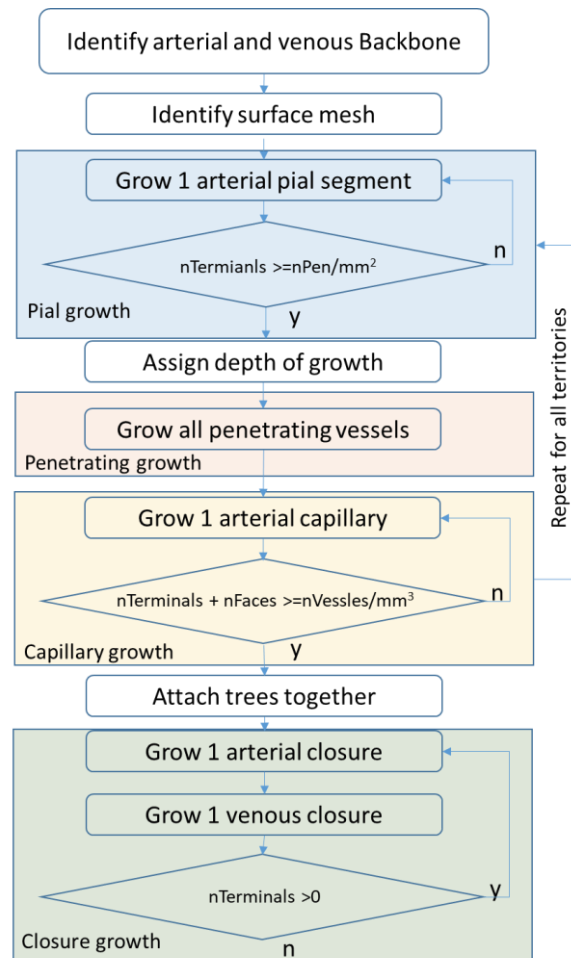


Figure 3.12. Workflow diagram of the anatomical growth algorithm.
The diameter is assigned and the vasculature is saved with a new name at the end of every step.

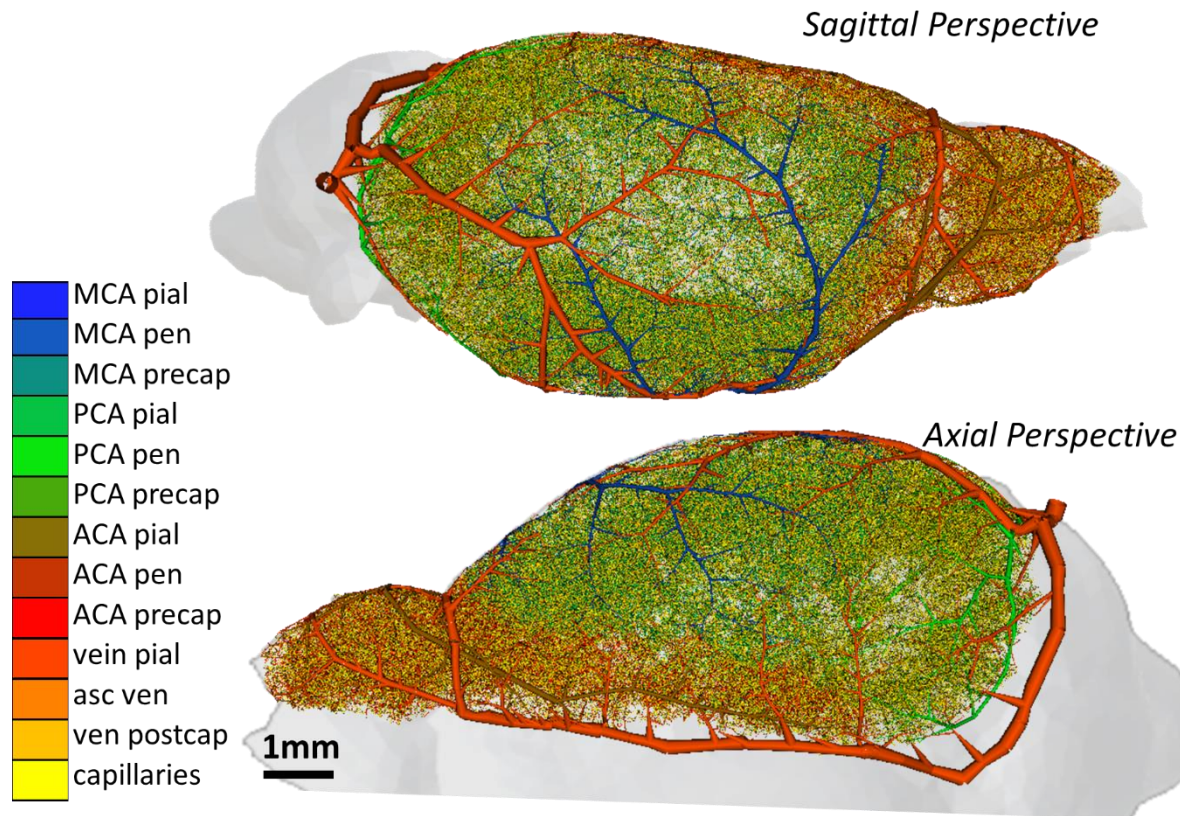


Figure 3.13. Network structure and select simulation results from biphasic blood flow for the complete cerebral hemisphere in mouse.

The anatomical hierarchy of the network encompasses blood vessels including pial arteries, penetrating arterioles, precapillaries, capillaries, post-capillaries, ascending venules and final drainage through the pial veins and sinuses. Abbreviations: MCA pial/pen/precap indicates pial vessels, penetrating arterioles and precapillaries belonging to the MCA territory, respectively. Likewise for ACA and PCA. Ven postcap, asc ven, and ven pial indicate venous post-capillaries, ascending venules and pial veins respectively. File location is:

tShare:\03_Mouse\artificialNetworks\artificialSledMouse\Gen1_BiasSampling\hemisphereV4_16704_density\hemisphereArtVenClosureWithClosurev3.AddedTortuosity.resultsLinninger2015Pries_In_Vitro.cs31

3.3.3 Human

The growth algorithm for the hemisphere was also extended to the arterial tree of the human brain. This preliminary case study exemplifies the robustness of the algorithm when adapted to the highly tortuous gyrations of the human cortex. These gyrations do, however, pose problems when using an un-biased sampling throughout the cortical region-of-interest. Methods for overcoming this dilemma are offered in Section 7.8 where curvature estimation gives rise to ridge detection.

Also offered in the same section is a method for growing via expansion and a paradigm for pial growth identifying continuations and bifurcations using advanced methods. Visualizations of the current methods are offered in Figure 3.14-Figure 3.15.

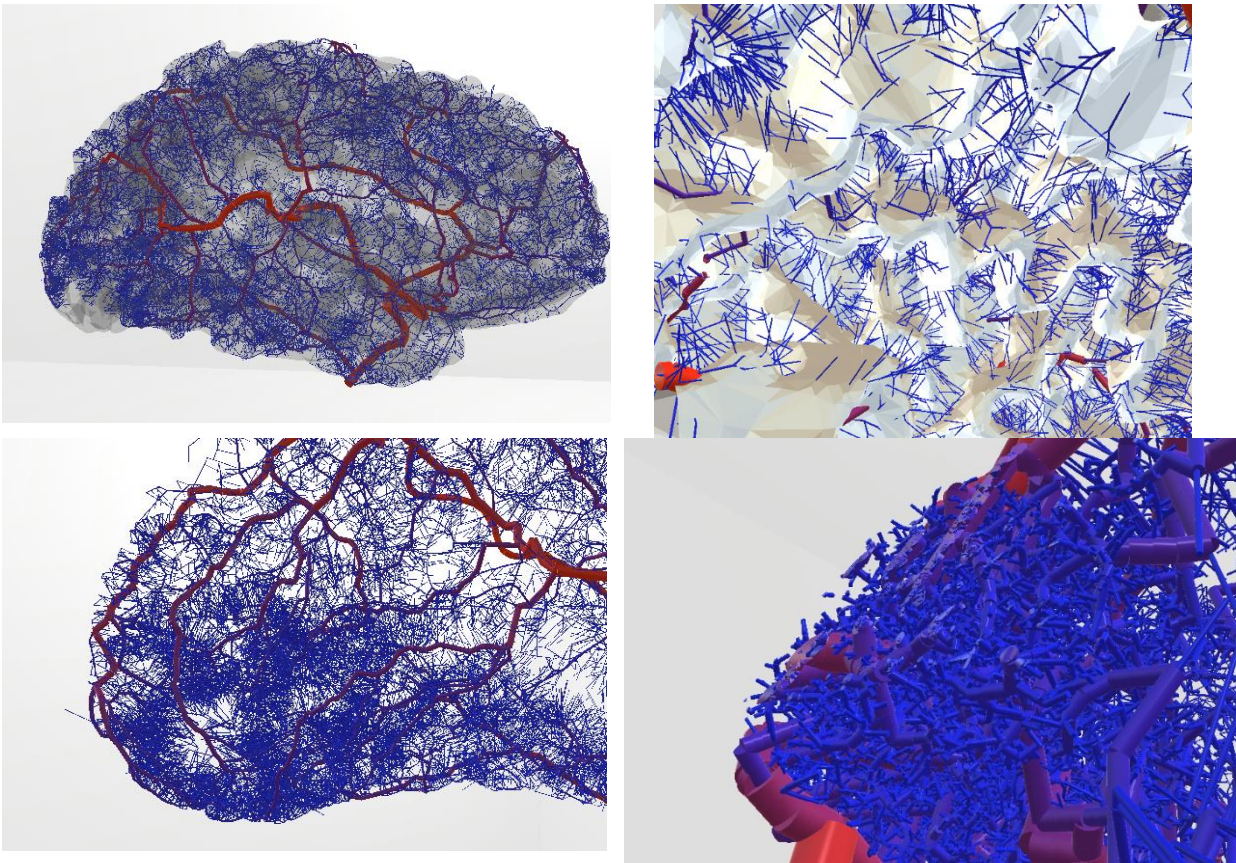


Figure 3.14. The penetrating arterioles applied across an entire region of the human brain. This visualization (MCA territory) shows concentration of vessels around ridges, where all penetrators point towards each other. This is in opposition to the gyrations of the cortex where the penetrators flare outwards from each other.

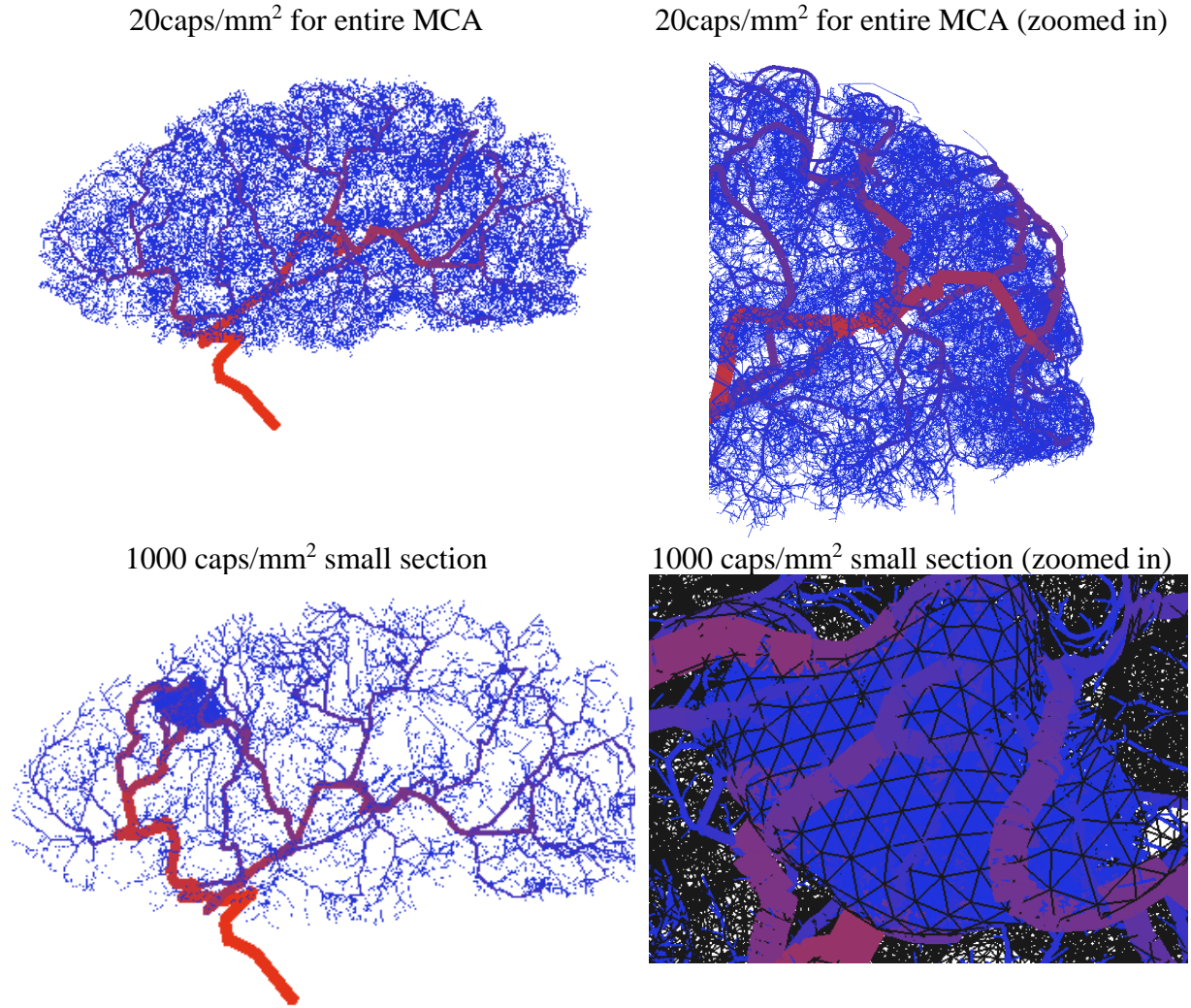


Figure 3.15. Further visualizations of capillary synthesis in human cortex.

The growth with a defined backbone and (from MRI reconstructions) and pial surface growth at different densities and focal areas. The entire MCA region of the surface is grown to a predetermined number of penetrators. Penetrators and capillaries were then grown in Top Row) the entire MCA territory at a density of 20 capillaries/mm² surface area coverage and Bottom Row) a small user-defined section of the cortex to a density of 1000 capillaries/mm².

3.4 Discussion

The automated topology matching algorithm proposed here allows, for the first time, a post-processing alignment of topological property spectra from any synthetic network to an empirical counterpart. This is a pivotal advancement to automatic construction of anatomically-consistent cerebrovascular models. This automation removes the most difficult and time-consuming part of

anatomical growth; tuning the growth constraints, while producing realistically tortuous segments. An expansion of these methods could allow application of different topological profiles to different regions without modifying the core growth algorithm itself (once these data are acquired).

Limitations. While these methods produce topologically consistent structures, our team has identified previously undescribed properties of empirical networks including vertical alignment bias and extraneous unidentified penetrating arterioles (see Section 7.4). The implementation of these new discoveries into the growth paradigm is outside the scope of the current work.

4 Large microvascular networks reveal depth dependent hematocrit gradient

Parts of this chapter was previously published as Hartung, Grant, Claudia Vesel, Ryan Morley, Ali Alaraj, John Sled, David Kleinfeld, and Andreas Linninger. "Simulations of blood as a suspension predicts a depth dependent hematocrit in the circulation throughout the cerebral cortex." *PLoS computational biology* 14, no. 11 (2018): e1006549.

4.1 Introduction

Metabolic activity of the brain is controlled by a complex system of neuroreceptors, small molecular regulators such as nitric oxide, hormones and proteins. The supply, clearance, and balance of metabolites, oxygen, glucose and waste are controlled by the cerebral circulation which is coupled with the cerebrospinal and interstitial fluid (CSF/ISF) subnetworks [87,88]. The coordination between oxygen extraction and increased cerebral blood flow after neuronal firing has garnered intense research interest in blood oxygen-level dependent (BOLD) signal, which is the basis of functional magnetic resonance imaging (fMRI). Recent work [89] has begun to quantify the microvascular origin of the BOLD fMRI signal in a microsection of a mouse brain. The study integrated state-of-the-art neuroimaging of anatomical spaces, tissue oxygen tension measurements and a mechanistic model of blood-bound oxygen supply and converted changes in cerebral blood flow and oxygen extraction into synthetic BOLD signals using Monte Carlo simulations. The main achievement was a successful first principles correlation between measured oxygen and cerebral blood flow (CBF) levels generating fMRI signals.

A recent paper from our group [90] aimed at widening the spatial coverage of coupled blood flow and oxygen simulations. Our model also offered detailed saturation and dissociation kinetics of plasma and red blood cell-bound oxygen, endothelial mass transfer and tissue oxygen extraction. Our study quantified vascular network effects by coupling biphasic (=suspension of red blood cells in plasma) hemodynamics and nonlinear blood rheology with oxygen kinetics. In addition, the number, distribution and position of neuronal and glial cell nuclei were acquired in a sizable section ($\sim 1 \times 1 \times 1 \text{ mm}^3$) of vibrissa primary sensory cortex. We also predicted oxygen saturation in arterioles, capillaries and veins within experimental error bounds. By adopting a probabilistic approach to account for mitochondria respiration associated with specific neuronal and glial somata, the model was used to compute subcellular oxygen gradients between the extracellular matrix, the cytoplasm and individual neuronal/glial mitochondria. The remaining open question concerns the spatiotemporal coordination inside the neurovascular unit.

4.1.1 Regulation.

There is agreement that the neurovascular unit locally controls the cerebral blood flow response. Yet, oxygen supply exceeds the metabolic demand of neuronal activation for reasons that still remain uncertain [91]. Because of the massive size of the mammalian brain with its immense number of neurons and capillaries, the precise temporal and spatial coordination among cellular components still eludes exact physiological description. For example, studies suggest that functional hyperemia causes local neuronal metabolism increase of 5%, which in turn augments local blood flow by 30% up to almost 130% of base line perfusion [92]. However, the exact timing, regulation, and extent of dilation in individual spatially distributed vascular compartments during functional hyperemia are still being investigated [89,93–95].

The cerebral circulation also exercises a second blood flow control mechanism known as cerebral autoregulation [96–103]. Clinical observation [97] suggests that the total CBF remains constant over a wide range in perfusion pressure (± 50 mmHg, ± 6666 Pa). Many excellent contributions [104–106] correctly attribute the constancy of cerebral blood supply to global resistance adjustments. Yet, the involvement of specific vascular compartments, speed and spatial coverage of local vasodilatory/vasoconstrictive districts remains elusive [11,18–20]. Moreover, quantification of network effects and control principles among vascular compartments requires an anatomically accurate mathematical model of the cerebral circulation.

Propelled by the advances in neuroimaging, several groups have begun to integrate medical image data with large-scale computer models [89,90,95,107–109]. Generally, these efforts fall into two types. One type adopts a reductionist approach using simplified networks to highlight global blood flow distribution patterns [93,110–114]. The second type follows a bottom-up strategy which aims at replicating relevant microcirculatory components down to the level of the cellular ensemble. Noteworthy examples include quantifying the neurovascular coupling in functional hyperemia [89], analysis of pressure drop dependence on cortical depth [108], predictions of blood flow control by intra-cortical arterioles [95], and cortical oxygen distribution [115,116]. The ultimate goal of bottom-up models is *a hemodynamic simulation of the entire brain*, yet virtual circulation models of the whole brain have been perceived as intractable due to size and nonlinearity of the mathematical coupling between blood flow and oxygen kinetics [110].

This manuscript will introduce a computational procedure that integrates multimodal neuroimage data covering different length scales into a unified virtual representation of the murine cortical circulation. Two-photon imaging provides data for the reconstruction of capillary networks. High resolution micro computed tomography (μ CT) imaging is used to capture the

connectivity between main arterial branches and pial blood vessels. The morphometrics of the micro, meso- and macro-scale vascular models have been statistically analyzed in order to synthesize virtual blood flow networks with anatomically equivalent statistics, but without being confined to the limited field-of-view or resolution of imaging modalities.

The aim of this paper was to quantify network effects of uneven red blood cell distribution in the cerebral circulation. Although uneven red blood cell distribution also known as plasma skimming can be observed in single bifurcations, neuroimaging of the entire cerebral circulation has so far not been accomplished. To overcome this shortcoming, we integrated physiological data from several neuroimaging modalities covering three different lengths scales. Massive computer simulations of large microcirculatory networks of the murine primary cortex revealed a trend of depth-dependent hematocrit, which is a significant finding indicating that the intricate architecture of the cortical microcirculation serves a self-regulating function to maintain uniform oxygen perfusion.

4.2 Materials and Methods

An overview of the data structures used in this study is presented in Figure 4.10.

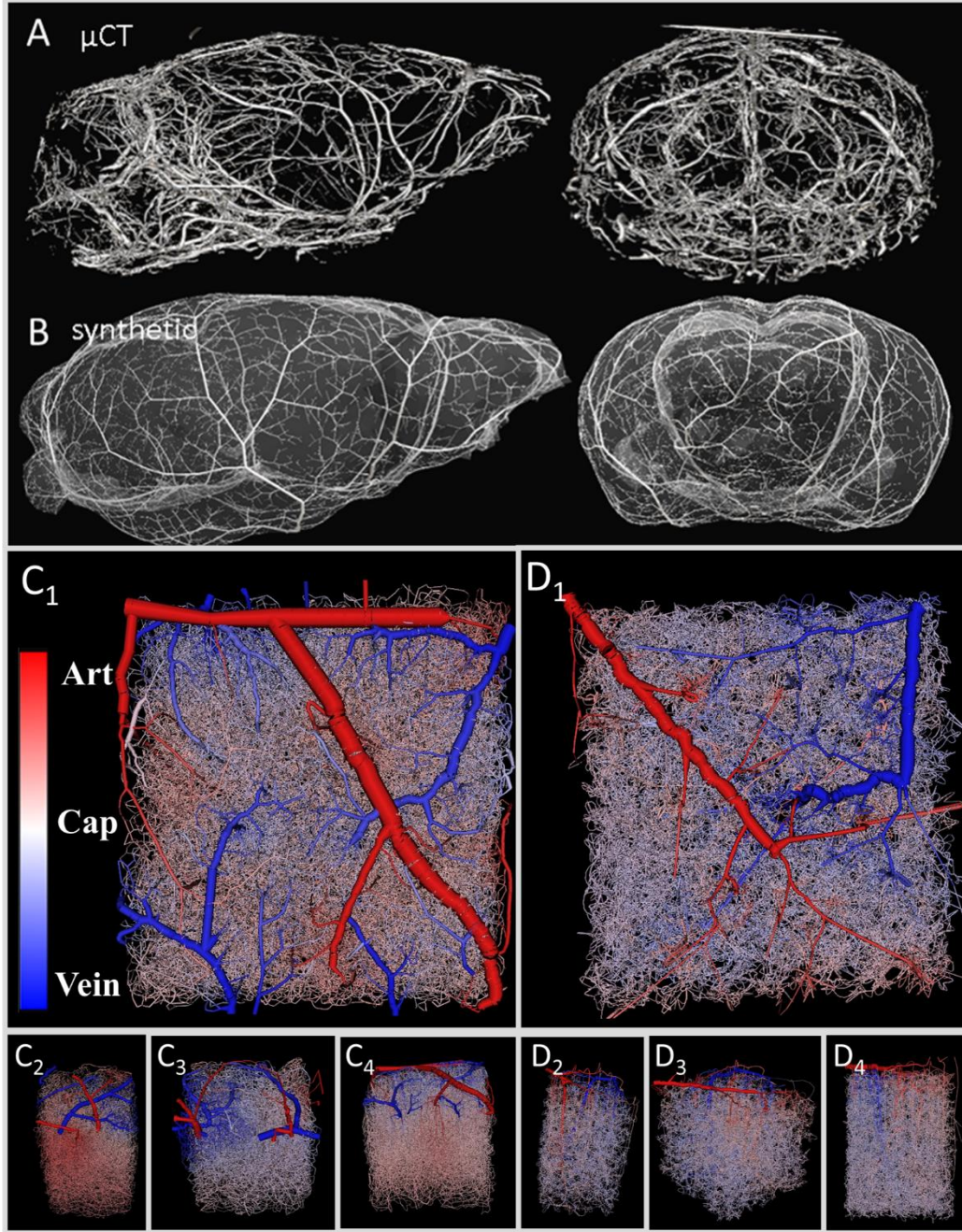


Figure 4.1. Multi-modal imaging data used to construct realistic models of cerebral circulation for entire mouse brain.

(A,B) *Main blood vessels and pial arterial network.* (A) High resolution μ CT image of the vascular tree in mouse. (B) Synthetic pial arterial tree generated by modified constrained constructive optimization and morphological data from the μ CT images. Note that the synthetic circulatory network does not perfectly reproduce the layout of the pial vessels in the μ CT, but merely possesses similar morphometrics. (C,D) *Cortical microcirculation* (C₁-C₄) Experimental microcirculatory networks acquired with two-photon laser imaging [26]. (D₁-D₄) Synthetic microcirculatory data sets. The synthesized data sets are statistically equivalent to the

reconstructed networks from 2PLSM. Arterial (red) and venous side (blue) are color coded for better visibility.

4.2.1 Pial surface data acquisition

Nine female C57BL/6 mice were imaged for pial vascular network structures following intravascular injection of a lead pigment contrast agent as described elsewhere [117–120]. The mice were perfusion fixed prior to micro computed tomography (μ CT) imaging with 7-20 μ m isotropic resolution of the cerebral angioarchitecture. The resulting 3D images were filtered and the vascular lumen reconstructed as previously described [121–123]. Figure 4.10A shows raw μ CT samples of the mouse vasculature from a 20 μ m resolution image. The pial network statistics such as penetrating arteriole density and vessel diameter were compiled with results summarized in Table 4.2.

4.2.2 Microcirculatory data acquisition

Four volumes (N=4) that encompassed the murine vibrissa primary sensory cortex [26] were imaged using two-photon laser scanning microscopy (2PLSM) and are shown in Figure 4.10C. 2PLSM was employed to extract the spatial arrangement, length and orientation of blood vessels in the vibrissa primary sensory cortex [26,124,125]. Blood vessels in four data sets ($\sim 1 \times 1 \times 1$ mm³) were labeled as pial arteries, penetrating arterioles, capillaries, ascending venules, or pial veins. Categorization was based on size and branching level according to Strahler order rather than physiological markers. No effort was made to differentiate pre-capillary arterioles from post-arteriole capillaries because it requires differential labeling of smooth muscle and pericytes. Capillaries were differentiated from ascending venules by a diameter cutoff of 6 μ m and

penetrating venules were differentiated from pial veins for vessels within a depth of 100 μm below the pia and a diameter less than 12 μm . Diameter information was also derived from images. The network information was stored using sparse connectivity matrices. Length, diameter, and tortuosity spectra are depicted in Figure 4.2. More details on image acquisition [26,124,125], image reconstruction [126], as well as the formulation of the network equations [115] can be found elsewhere.

4.2.3 Synthesis of large circulatory networks

Artificial microvascular networks ($N=60$) for large sections of the cortex ($\sim 1 \times 1 \times 1 \text{ mm}^3$) were synthesized using a previously described vascular growth algorithm (Sections 2-3). Four examples are displayed in Figure 4.10D. The algorithm preserved dimensions of the experimentally acquired cortical samples, pattern and dimension of pial arteries, number, orientation and connectivity of penetrating arterioles, and morphometrics of the capillary bed, draining venules and pial veins, as listed in Table 4.3. Statistics and morphometric comparisons of experimental and synthetic data sets are displayed in Figure 4.2.

The arterial network of the entire MCA territory spanning three orders of magnitude in length from large arteries ($\sim 1 \text{ mm}$ range) down to the entire capillary bed ($\sim 1 \mu\text{m}$) was synthesized based on morphometric statistics of source data from multimodal images (μCT and 2PLSM).

4.2.4 Blood flow

Microcirculatory blood flow was modeled as a biphasic suspension comprised of red blood cells and plasma. Bulk blood flow was described by Poiseuille law relating volumetric flow to

pressure drop as a function of resistance which in turn depends on diameter, d , and hematocrit-dependent viscosity [127]. In addition, a kinetic plasma skimming model (KPSM) presented previously [115] accounted for the uneven RBC distribution, known as plasma skimming. An in-depth inventory of alternative plasma skimming models is offered in Section 7.12.

This model has only one adjustable parameter, the skimming coefficient, m . It was set to value of $m=8$ in all microcirculatory models, although this parameter could be refined as shown recently [128–130]. The nonlinear systems of conservation balances in system (1) were solved iteratively to calculate blood pressures, p , flow, Q , and hematocrit, h . Here, R is the resistance matrix, C_1 and C_2 are fundamental connectivity matrices [131] and C_3 is the advection flux matrix. Boundary conditions are summarized in Table 4.1. More details on the mathematical background are given in Section 4.6; implementation details are discussed elsewhere [115].

$$G(Q, p, h) = 0 \begin{cases} R(h, d)Q - C_1 p & = 0 \\ C_2 Q & = 0 \\ C_3(Q, d)h & = 0 \end{cases} \quad (1)$$

Table 4.1. Summary of boundary conditions

Arterial Inlet	$p=120$ mmHg (15,999 Pa)
Venous Outlet	$p=5$ mmHg (667 Pa)
Inlet Hematocrit	$h=0.35$
Outlet Hematocrit	Fully developed, $\nabla h=0$
lower boundaries (GM/WM interface)	confined domain (zero flux, $\nabla Q = 0$)
sides boundaries (vertical tissue boundary)	cyclic boundary conditions or confined domain (zero flux, $\nabla Q = 0$)

4.3 Results

4.3.1 Morphometrics

We first assessed morphometrics of experimental data obtained from murine primary somatosensory cortex samples (N=4, E1.1-E4.1). The indexing and naming scheme for the data sets is listed in Table 4.2. The total microvascular segment count was $24,669 \pm 9,594$ splines. An important property is that all four original two-photon laser scanning microscopy (2PLSM) data sets contained blood vessels that divide into more than two daughter branches (multifurcations). Specifically, the four data sets contained 654, 725, 1686, 1440 multifurcations, respectively.

Table 4.2: Topological feature comparison between experimental 2PLSM and synthetic data sets

	Experimental 2PLSM (N=4)	Synthetic data sets (N=60)
Data name (labels)	E1.1, E2.1, E3.1, E4.1	S{1,2,3,4}.1-15
Number of splined segments (Nsgm)	$24,669 \pm 9594$	$24,679 \pm 8389$
Segments per pial surface (Nsgm/mm ²)	$16,704 \pm 2816$	$16,710 \pm 2464$
Bifurcations	$14,842 \pm 5652$	$16,451 \pm 5592$
Multifurcations	1172 ± 584	<250
Length (m)	1.38 ± 0.47	1.33 ± 0.47
Intravascular volume (nL)	24.3 ± 12.0	27.0 ± 6.1
Blood-brain barrier surface area (mm ²)	16.3 ± 6.3	17.7 ± 5.4
Coverage of pial surface area (mm ²)	1.6 ± 0.3	1.77 ± 0.5
Tissue volume (mm ³)	2.2 ± 0.8	2.2 ± 0.7

Statistics on cumulative metrics including vascular surface area, path length and luminal volume are compiled in Table 4.2. Although the data originate from the same cortical region, there are subject-specific variations between different specimens. There was higher variability in the low end of the vascular diameter spectra, because unavoidable uncertainty affects the thinnest

vessels close to image resolution threshold as observed previously [124]. We also estimated the surface area to tissue volume ratio of the blood-brain-barrier (BBB) of the microvascular network as $8.8 \pm 1.1 \text{ mm}^2 \text{ vasculature/mm}^3 \text{ tissue}$. This number was obtained by summing the (endothelial) surface area of the capillary bed; this estimate compares to experimental values of the BBB surface of about $10\text{--}17 \text{ mm}^2/\text{mm}^3$ in humans [132–134].

4.3.2 Synthetic data sets

A modified constructive growth algorithm (mCCO [116,135], as described in Sections 2-3) was used to create 60 synthetic data sets (S1.1-S4.15) of the murine primary sensory cortex. For each of the four experimental data sets, 15 clones with statistics matching closely their experimental original were created, so that the S1.1-15 series matched the original E1.1, and S4.1-15 matched data set E4.1. Artificial networks smoothly connect arterial vessels through the capillary bed to the veins without gaps or the need to insert artificial segments as observed with other methods [113]. In addition, since blood vessels are not exactly straight, realistic tortuosity values were imposed by a Bezier spline-based technique described previously [116]. Moreover, at the boundaries of the synthetic data sets neither pial surface vessels, nor deeper laying arterioles, capillaries, or venules were severed or had to be pruned thanks to the precise geometric control of the vasogenic growth algorithm. Artificial network growth took less than five minutes for each dataset on a personal computer.

4.3.3 Branching patterns

We also compared morphometrics of experimental (N=4) against synthetic vibrissa primary somatosensory cortex data sets (N=60, S1.1-S4.15). No discernible feature differences can be inferred from visual inspection as shown for three experimental (E2.1, E3.1, E4.1) and six synthetic data sets (S2.5, S3.3, S4.5, S2.3, S3.4, S4.8) in Figure 4.2A. Total count amounted to $24,679 \pm 8389$ spline segments and 16,451 bifurcations. Spline segments were defined as tubular connections (splines) between branching points (bifurcations or multifurcations). This counting method ensured that the final tally is independent of image grid resolution or number of segment sub-partitions. The comparison of cumulative properties and probability density functions shows excellent agreement between the experimental and synthetic networks as seen in the plots Figure 4.2B and C. The synthetic networks are different realizations, but statistically equivalent replica (clones) of the original image samples.

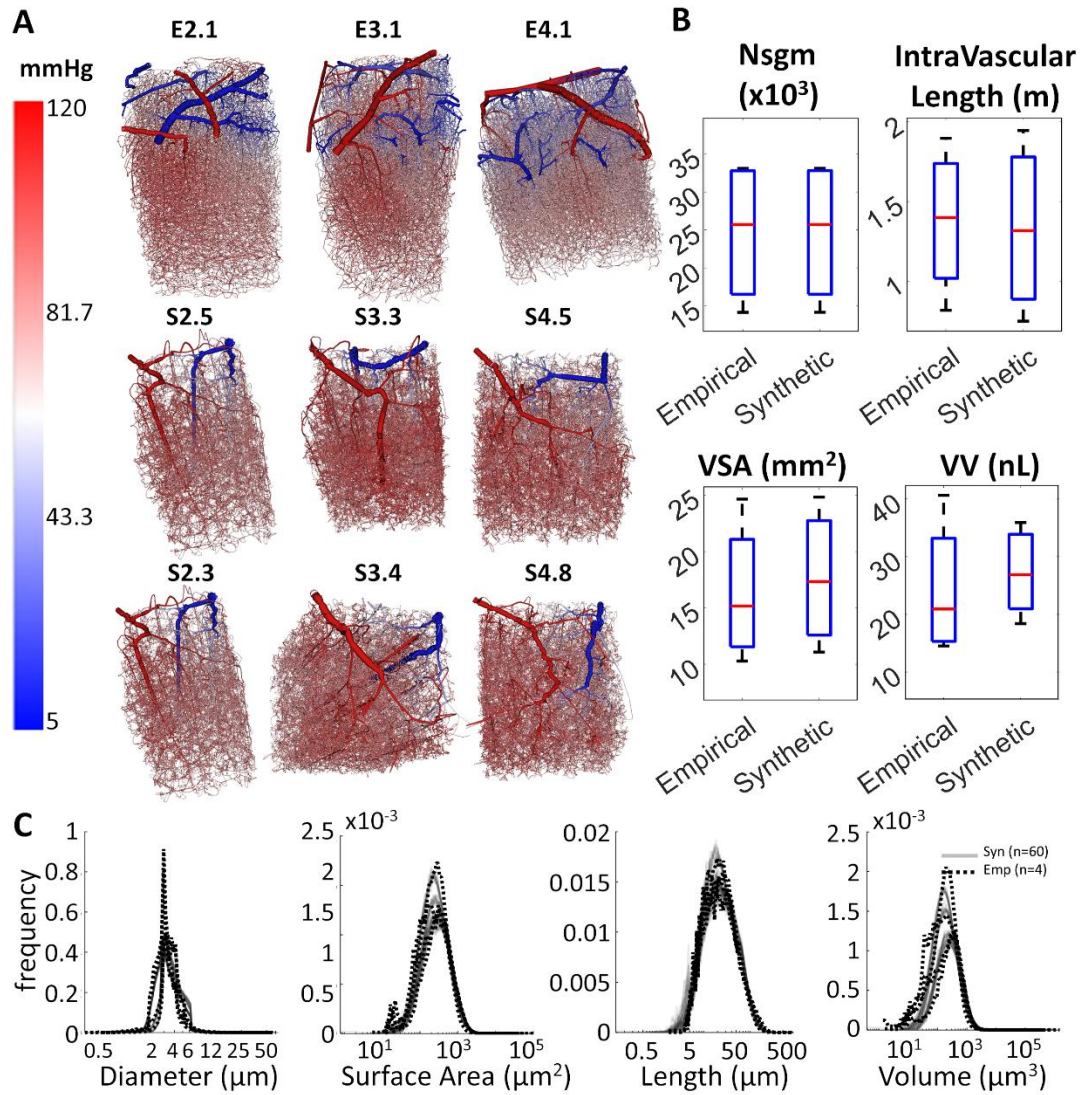


Figure 4.2. Morphometric comparison between experimental and synthetic microcirculatory networks from the murine vibrissa primary sensory cortex.

(A) Three 2PLSM experimental [26] data sets (E2.1, E3.1, E4.1, three shown out of four) are compared to synthetic (S2.5, S3.3, S4.5, S2.3, S3.4, S4.8) microcirculatory networks (six shown, out of sixty total). (B) Cumulative microcirculatory morphometrics for experimental ($N=4$) and synthetic ($N=60$) networks (segment number-Nsgm, intravascular length, vascular surface area-VSA, and vascular volume-VV are statistically similar, $p>0.05$ in all cases using one-way ANOVA). (C) Probability density functions show that the synthetic data sets are not identical, but match the topology of experimental data sets. Taken together, the morphometric analysis shows that experimental and synthetic networks are statistically equivalent.

4.3.4 Network Effects in large-scale models

The nonlinear biphasic blood flow, pressure and hematocrit equations for all four experimental networks and all sixty synthetic networks converged within five minutes [115]. Results were visualized with 3D rendering software Walk-In Brain developed at our institution [136,137]. Path analysis was conducted based on flow trajectories traversing the network along streamlines. Biphasic blood flow and network effects determining blood pressure and hematocrit distribution through large experimental (N=4) and synthetic (N=60) networks perfusing a large portion of the cortex were studied.

Typical pressure distributions along the microcirculatory network hierarchy are shown in Figure 4.3. Pressure drop trajectories through the microcirculation showed patterns consistent with experimental data [138–140]. Results of the path analysis in Figure 4.3 also depict the wide variations of hemodynamic states when blood traverses the dense microcirculatory network from the pial surface vessels through penetrating arterioles into the capillary bed and finally back to the collecting veins. The trajectories of individual paths (green, blue, magenta and yellow) display wide variability of hemodynamic states along the flow direction. Flow analysis reflected that a perfusion pressure drop in the microcirculatory networks from 120 to 5 mmHg (15,999-667 Pa) resulted in a mean tissue perfusion of 68.9 ml/100g/min ($11 \cdot 10^{-6} \text{ m}^3/\text{kg/s}$) which is within experimentally observed ranges [141,142].

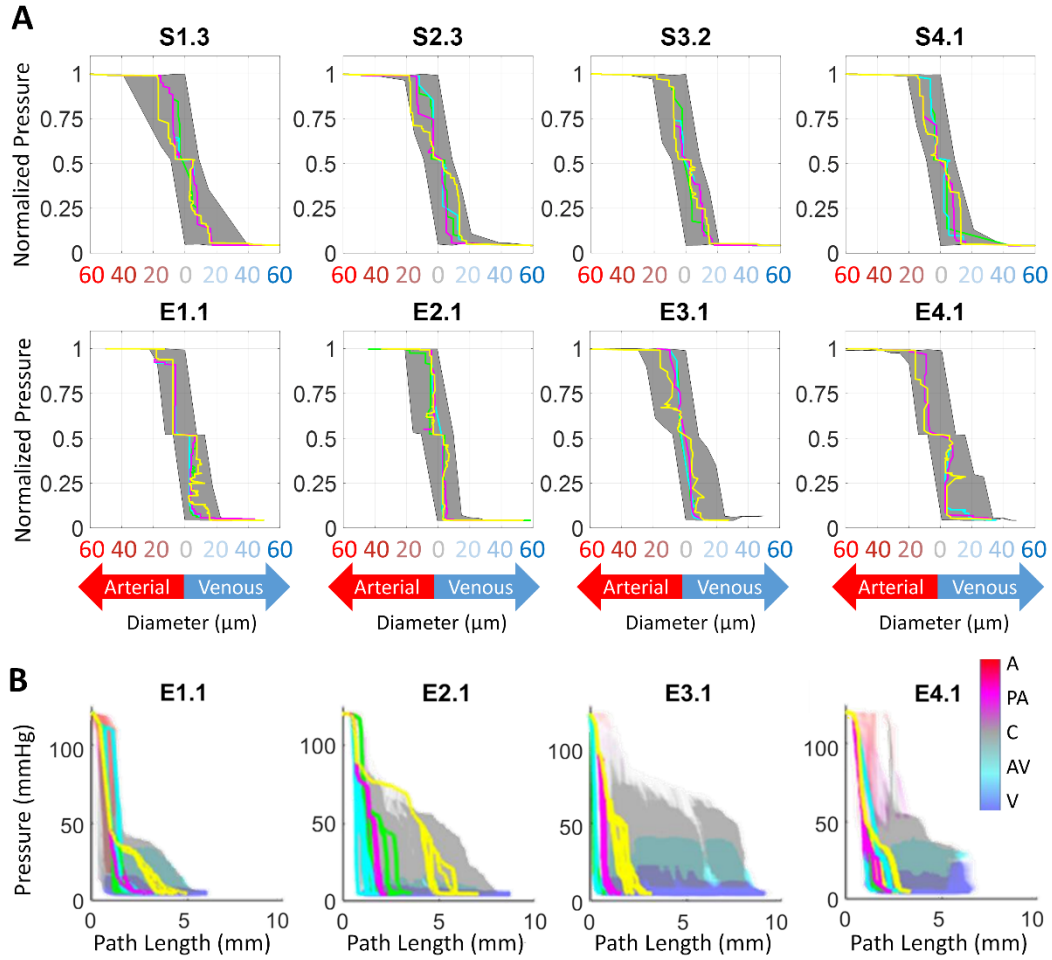


Figure 4.3. Predictions of hemodynamic states in primary cortex simulations show large variations due to network architecture.

(A) Path analysis ($N = 2,300\text{--}22,052$ paths per dataset) of blood pressure as a function of diameter. (It should be noted that some paths in the experimental data sets e.g. E1.1 exhibit zigzagging which is probably due to uncertainty in the diameter information). (B) Blood pressure as a function of path length in four microcirculatory data sets. Representative path trajectories have been plotted in green, blue, magenta and yellow (A-arteries, PA-penetrating arteries, C-capillaries, AV-ascending venules, V-veins).

4.3.5 Path analysis of hematocrit and layer dependence

We further inspected the RBC flux distribution as a function of network hierarchy (=vascular) and position inside the cortical hierarchy (=neuronal). The results were acquired for both empirical and synthetic data sets. Two representative specimens are highlighted in Figure 4.4A and B; eight

more examples are displayed in Figure 4.4C. Typical paths belonging to different cortical layers are color coded in Figure 4.4. Flow paths were generated by tracing the flow from arterial inlet nodes downstream through the capillary bed until reaching a venous outlet. Paths were sorted according to their tissue supply function as follows: a path depth label equal to the cortical depth of the deepest segment was assigned to each flow path. Thus, all paths were uniquely ordered within a spectrum of shallow to deep reaching paths according to the neuronal layer (I-VI) hierarchy in agreement with previously reported values [26,143,144]. Figure 4.4 depicts hematocrit values along representative paths in shallow (layer I-green) and deeply penetrating paths (layer V/VI-yellow). Along each path and between different paths there is high variability along the flow direction. For example, discharge hematocrit in data set S1.1 reaches values as high as $h_{\max} \sim 0.7$, and as low as $h_{\min} \sim 0.18$. However, there is an overall trend of higher hematocrit being carried to lower cortical levels (layer-V/VI paths). The trend of relatively higher hematocrit, h , conveyed to deeper tissue layers (p-value<0.01, using one-way ANOVA test in Matlab) was observed consistently in all experimental and synthetic data sets. The bulk flow, Q , showed the opposite trend; it was reduced in segments of deeper layers which are connected by longer paths as is summarized in Figure 4.5. In contrast to bulk flow and hematocrit, the RBC flux (=volumetric flow rate of the RBC phase) exhibited weak layer dependency, it was almost constant irrespective of the cortical depth. We also observed that the variance of capillary RBC fluxes decreased with cortical depth, thus RBC fluxes in deeper layers show lower variability than paths on the surface. Taken together, biphasic blood rheology and network effects seem to induce depth dependent hematocrit supply to the cerebral cortex which leads to more homogenized RBC fluxes in deeper layers (=lower variance in RBC fluxes). Further analysis of diameter dependence on hematocrit

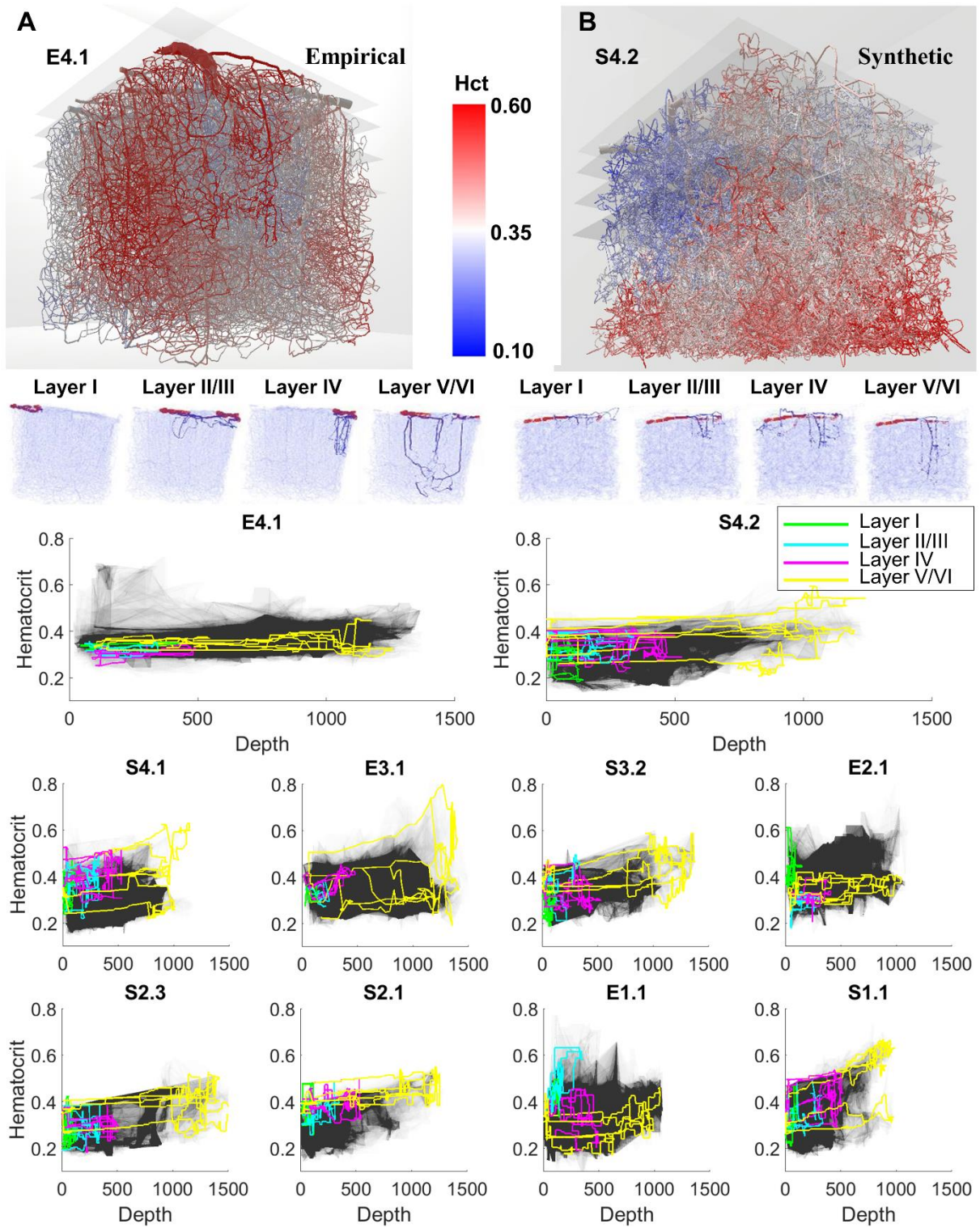


Figure 4.4. Depth dependent path analysis of hematocrit trajectories through the cortex. The visualization of the hematocrit field in three dimensions shows a higher (red) level of discharge hematocrit in the deeper segments than in segments closer to the pial surface. This

trend is observed in experimental (A, E4.1) and synthetic (B, S4.2) networks. Plots for all paths ($N=2,300-22,052$ paths per dataset) trace the grayed region, depicted here for experimental data set E4.1 and synthetic data set S4.2. For better visibility, representative paths descending to different depths are shown with color coding by layer (I-VI). Deeper reaching paths (layer V/VI-yellow) tend to carry higher hematocrit levels than shallower paths (layer I-green). (C) Additional data sets show consistently depth dependent hematocrit in synthetic ($N=5$, S1.1, S2.1, S3.2, S2.3, S4.1) and experimental ($N=3$, E1.1, E2.1, E3.1) networks (eight examples depicted).

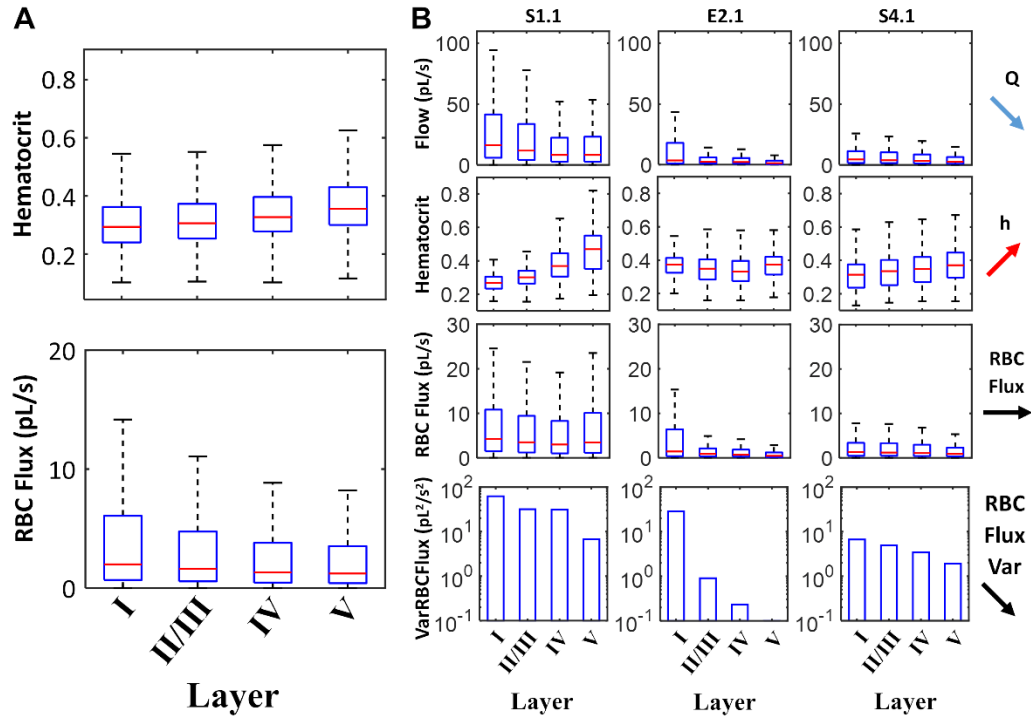


Figure 4.5. Statistics of hematocrit distribution and RBC fluxes in cortical layers of cerebral microcirculatory networks.

(A) Statistics over an ensemble of experimental ($N=4$) and synthetic ($N=12$) data sets show higher discharge hematocrit in deeper segments ($N=1,234,412$ with $p<0.01$ using one-way ANOVA). The median (red line), 25-75th percentile (blue box) and limits (black lines, excluding outliers) of hematocrit in all blood vessels for each layer obtained for all data sets. (B) Statistics in three individual data sets (S1.1, E2.1, S4.1, $N=59830$, 94842, and 108833 respectively, $p<0.01$ in all cases). In all case, layer V has higher hematocrit levels than the layers closer to the cortical surface. In general, shorter surface paths (layer I) tend to have higher flow rate, Q , but lower hematocrit levels, h . The total red blood cell flux (RBC) is rather uniform for all cortical layers, because the flow effect (Q , lower in deep layers) and hematocrit (h , higher in deep layers) balance each other out. The variance of the RBC fluxes, $VarRBCFlux$, decreases with depth; accordingly there is more homogenous RBC flux distribution in deeper layers.

confirmed the high degree of hematocrit variability across the diameter spectra as previously observed [90] (Section 4.7).

The agreement between the simulation results obtained for experimental and synthetic data confirms that the synthetic networks are hemodynamically equivalent to the experimental networks. The satisfactory match in morphometrics and hemodynamics between experimental and synthetic data justifies the extension of network synthesis to large anatomical regions as described next.

4.3.6 Extension to brain-wide hemodynamic simulations

Vascular networks covering the circulation of the entire MCA territory were generated with the help of our modified CCO (mCCO) algorithm as described in Gould et. al [116]. The mCCO algorithm was launched with the MCA M1 as the first segment. The location of the MCA territory within the context of the mouse cortex is shown in Figure 4.6 top-row. Sequentially, more segments were added at the cortical surface depicted in Figure 4.6 top-row, while minimizing the vascular tree volume subject to blood flow constraints. Thus, gradually the algorithm generated all arterial branches of the pial network on the cortical surface. Then, it was directed to proceed with penetrating arterioles and microcirculatory growth to a depth of approximately 1 mm below the pial surface, until a preset vessel density was reached. At each step of the segment generation, connectivity and bifurcation position were optimized to obtain minimum tree volume. The diameters of the network branches were recursively recomputed in accordance with hemodynamically-inspired principles [145]. The total number of splined segments in the artificial MCA territory was 993,185. This was roughly 60 times the number of segments in the cortical samples.

The topology of the synthetic MCA territory resembled maps available in mouse atlases [146,147]. Branching density and pattern of the pial arteries as well as the number of penetrating arterioles was within ranges of the reconstructed sets of μ CT images as listed in Table 4.3. Detailed views in Figure 4.6 show pial, microcirculatory and individual capillary scales illustrating different aspects of the massive network model covering three length scales ranging from the MCA M1 segment with a diameter [74] of 142 μ m down to the capillary bed [26], $d < 6 \mu$ m. Morphometrics of the synthetic MCA networks are summarized in Table 4.3. Figure 4.6A-C depicts the pressure, flow and hematocrit field from the outflow of the Circle of Willis (MCA M1), down to the smallest capillaries in the microcirculation. The anatomical detail and branching pattern is depicted for the highly irregular, tortuous microcirculatory network.

4.3.7 Complete circulation of the MCA territory including arterial and venous side

The simulation of the entire MCA territory included the compartments of pial arteries, penetrating arterioles, pre-capillaries, capillaries, post-capillaries, ascending venules and pial veins. To complete the MCA circulation, the venous tree including venules was synthesized in reverse and connected to the capillary bed as described previously [30]. Figure 4.7 depicts the distribution of pressure, flow and hematocrit throughout the MCA territory. Figure 4.7A shows comprehensive three-dimensional maps of the anatomical hierarchy, pressure distribution, blood flow in the MCA territory, and uneven biphasic hematocrit. Figure 4.7B-E highlights the anatomical grouping, pressure, flow, and hematocrit distribution throughout individual compartments. In these views, explosion diagrams separating the anatomical groups (pial arteries, penetrating arterioles, pre-capillaries, capillaries, post-capillary venules, venules and pial veins) were used to better delineate the hemodynamic states in each group. Visual inspection of the

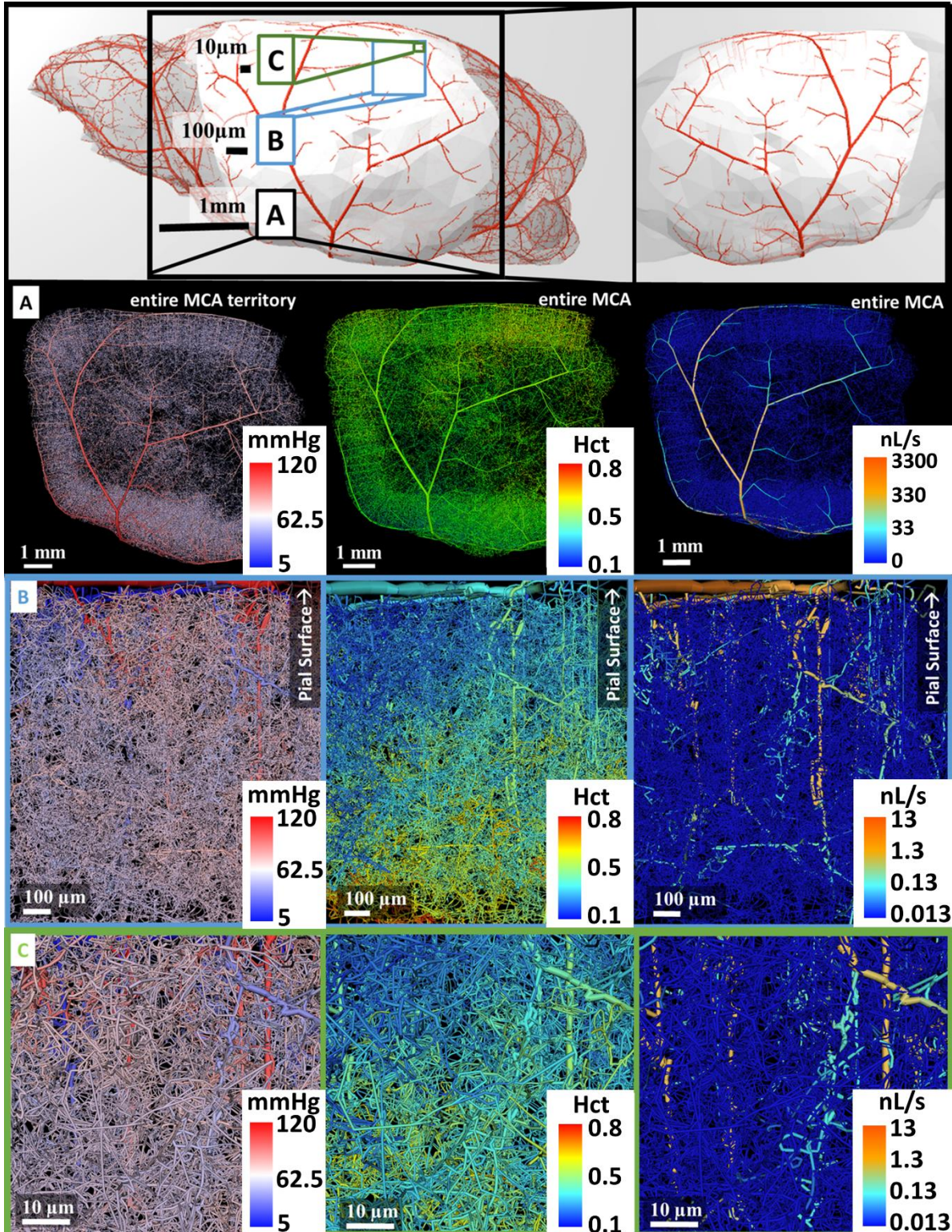


Figure 4.6. Schematic of multiscale biphasic blood flow simulations in the arterial side of the MCA territory.

The large-scale cerebrocirculatory model connects the Circle of Willis to the territory of the middle cerebral artery with its complete pial arterial network and microvasculature. Simulation results show snapshots of pressure distribution, flow rates, and hematocrit at three length scales (1mm, 100 μ m, 10 μ m). The three views roughly correspond to the resolution of several imaging modalities: top layer, A, depicts the major arteries and anatomical features at the millimeter scale as seen in μ CT imaging; the middle layer, B, shows arterioles at the micron range; the bottom layer, C, reaches cellular resolution as seen in 2PLSM or with confocal imaging.

microcirculatory compartments (pre-capillaries, capillaries, and post-capillaries) depicted in Fig 6E reveal higher hematocrit levels in deeper cortical layers than on the surface.

4.3.8 Blood flow

Simulations conducted for the entire circulation on the MCA territory required boundary conditions at only two points; MCA M1 arterial blood pressure ($p=120\text{mmHg}$, $15,999\text{Pa}$), hematocrit level ($h=0.35$), and venous outlet pressure ($p=5\text{mmHg}$, 667Pa). The solution encompassed blood pressure, flow and hematocrit for 5452 pial vessels, 27,374 segments perpendicular to the pial surface, and 960,359 capillaries of the entire left MCA territory, for a total of 993,185. In total, the proposed iterative method succeeded in bringing to convergence a total of 2,648,853 equations for biphasic blood flow.

Table 4.3. Pial network parameters used in this work in comparison to prior research

Parameter	Value	Units	Citation
Penetrating arterioles surface coverage	13 ± 3	Nsgm/mm ²	Nishimura [148]
	13	Nsgm/mm ²	This work (entire MCA territory)
Average penetrating arterioles diameter	11	µm	Blinder [26]
	11	µm	This work (entire MCA territory)
Larger artery diameter	143 ± 8	µm	Kidoguchi [74]
	142	µm	This work (entire MCA territory)
Mouse brain volume	453 ± 19	mm ³	Ma [75]
	509 ± 23	mm ³	Badea [76]
	415 ± 24	mm ³	Kovačević [77]
Sagittal Length	13	mm	Kovačević [77]
	13.7	mm	Diem [149]
	13	mm	Clavaguera [150]
	14	mm	Natt [151]
	13.7	mm	This work (entire MCA territory)
Coronal Height	10	mm	Kovačević [77]
	8	mm	Diem [149]
	9	mm	Natt [151]
	8.0	mm	This work (entire MCA territory)
Coronal Width	5	mm	Kovačević [77]
	5.5	mm	Diem [149]
	5.5	mm	Clavaguera [150]
	6	mm	Natt [151]
	5.7	mm	This work (entire MCA territory)
Cortical surface area	380 ± 20	mm ²	Ma [75] (young mice)
	348 ± 3	mm ²	Badea [76]
Number of splined segments	993,185	Nsgm	This work (entire MCA territory)
Segments per pial surface	25,144	Nsgm/mm ²	This work (entire MCA territory)

*Nsgm – number of splined segments

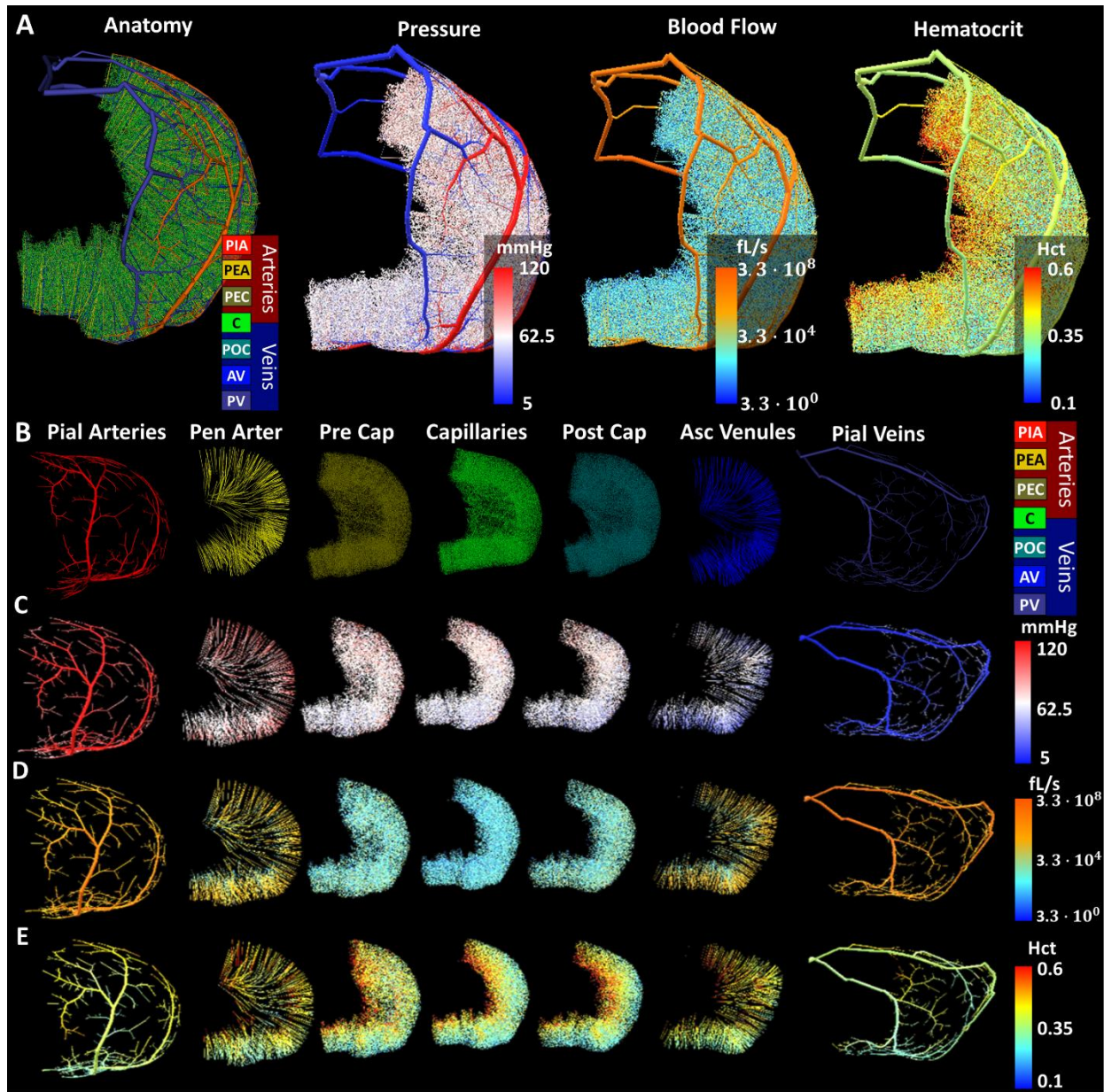


Figure 4.7. Blood flow of the complete arterial and venous circulation for the MCA territory in mouse.

This large-scale model contains the MCA M1 segment branching from the Circle of Willis as inflow and covers the entire territory of the middle cerebral artery with a complete pial network and microvasculature encompassing P_iA-pial arteries, P_eA-penetrating arterioles, P_eC-pre-capillaries, C-capillaries, P_oC-post-capillaries, AV-ascending venules, PV-pial veins. (A) Three dimensional snapshots of the spatial distribution of anatomical grouping, blood pressure, blood flow (perfusion), and hematocrit. Explosion diagram of anatomical compartments of the angioarchitecture in the MCA territory; color-coding depicts (B) anatomical groups, (C) blood pressure, (D) flow (perfusion) and (E) hematocrit distribution. This large-scale model contains 5,452 spline segments of the pial network, 27,374 splines encompassing penetrating arterioles and ascending venules, and 960,359 capillaries. The volume of the pial arteries is 143 nL

(16.5%), penetrating arterioles is 75.1nL (8.7%), precapillary arterioles is 101.9nL (11.8%), capillaries is 96.8nL (11.2%), post-capillary venules is 100.7nL (11.6%), ascending venules is 75.8nL (8.7%) and pial veins including portions of the superior sagittal sinus is 273.8nL (31.5%).

The predicted perfusion rate for the MCA territory was 50 ml/100g/min ($=8.3 \cdot 10^{-6} \text{ m}^3/\text{kg/s}$) which is in agreement to literature ranges [141,142] of 40-163 ml/100g/min ($=6.7\text{-}27.2 \cdot 10^{-6} \text{ m}^3/\text{kg/s}$). The trend of higher hematocrit levels in deeper cortical layers seen in the smaller cortical samples was also confirmed in the massive simulations for the MCA territory as shown in Figure 4.8.

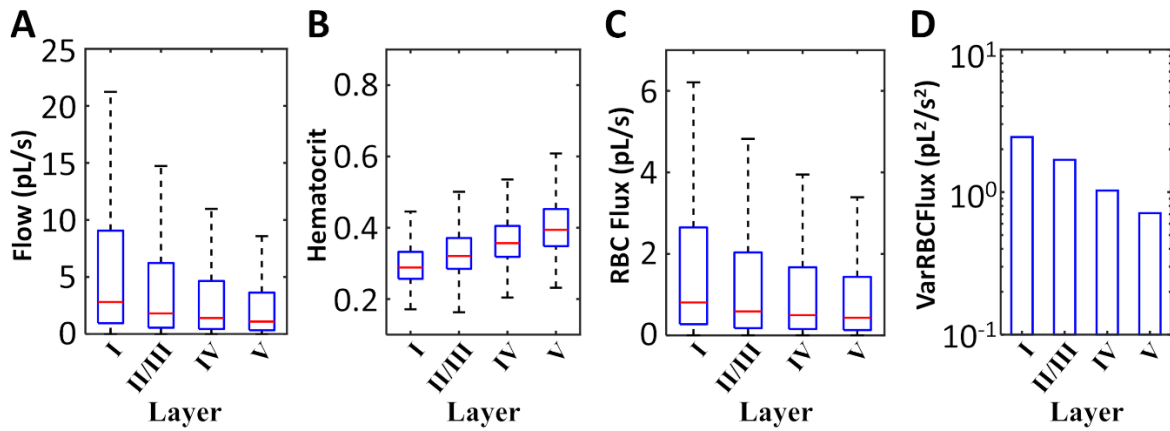


Figure 4.8. Depth dependence of hematocrit on total blood flow in the left MCA territory. Biphasic blood flow simulations for the entire MCA territory were analyzed statistically to illustrate depth dependence of hemodynamic states. In a large subsection cut out of the left MCA cortical vasculature (with volume=4.1 mm³ and surface area=4.2 mm² which equals 11% of the MCA territory), 188,865 microvascular segments in layers I-V were assessed. (A) Blood flow slightly decreases in deeper paths ($p < 0.01$ using one-way ANOVA test). (B) Hematocrit increase along deeper cortical layers ($p < 0.01$, one-way ANOVA test). (C) The product of bulk blood flow and hematocrit gives the RBC flux, which is almost constant with a mild decrease with depth ($p < 0.01$ using one-way ANOVA test). (D) The variance of RBC flux (*VarRBCFlux*) decreases with depth, so RBC flux distribution is more homogenous in deeper layers than close to the surface.

It should be noted that the simulations showed virtually no boundary effects in the center of the MCA territory where the primary sensory cortical samples were located. The suppression of boundary effects that can be achieved by large-scale simulation is extremely important for simulating hemodynamic blood flow control such as it occurs in functional hyperemia or under autoregulatory control. A full simulation of the entire MCA territory (arterial and venous side) required 65 iterations and ~2 hours on multicore workstations.

4.4 Discussion

4.4.1 Morphometrics

We performed multiscale morphometric analysis of the cerebral circulation in mouse over three length scales. On both the macro and the mesoscale, statistical data for the Circle of Willis, the middle cerebral artery and its pial arterial network were extracted from high quality micro-CT (μ CT) data [117]. Microcirculatory morphometrics were acquired by two-photon imaging (2PLSM) delineating the micro-angioarchitecture down to the level of individual capillaries for sizable sections ($\sim 1 \times 1 \times 1 \text{ mm}^3$) of the vibrissa primary sensory cortex. There were statistical differences between the 2PLSM microcirculatory data sets especially in the diameter information as can be expected from a high resolution analysis of cortical microcirculatory networks. However, these variations did not significantly alter hemodynamic flow patterns. The morphometrics (arterial, capillary and venous segment number, connectivity and branching patterns, probability density functions for length, diameter and surface area spectra) informed a synthetic vascular growth algorithm. Because the statistics (e.g. segment numbers) could directly be input into the mCCO algorithm, we were able to create 15 synthetic replica for each of the four data sets. In total,

we synthesized artificial vascular networks ($N=60$) with morphometrics and blood perfusion patterns that are statistically equivalent to the experimental data. The wealth of experimental and synthetic data used in this study provided a testbed for hemodynamic analysis of biphasic blood flow through the cortical microcirculation.

4.4.2 Blood Flow

Hemodynamic simulations were performed using computer algorithms described and tested extensively [115]. We performed biphasic blood flow simulations on both experimental ($N=4$) and synthetic microcirculatory networks ($N=60$). Simulation results predicted patterns of blood flow, pressure and hematocrit within ranges currently known from experiments. Even though our blood flow computations are deterministic [90,115], computed hemodynamics states varied widely within the labyrinth of paths traversing the microcirculation. We pinpointed randomness of the angioarchitecture as the origin of the wide range of predicted hemodynamic states. The finding of variability in hemodynamic states due to network architecture is significant, because it suggests that there are no characteristic properties (e.g. average hematocrit, mean capillary pressure) that would justifiably represent a typical physicochemical state of a microvascular compartment (arterioles, capillary bed, venules). It also explains why idealized trees such as binary ordered hierarchical graphs [112] are unsuitable surrogates for microcirculatory flow networks, because their regular and symmetric branching patterns lack the randomness in network topology seen in the murine anatomy. Specifically, ordered trees have equal states in all branches of a given hierarchy, which leads to even hematocrit splits due to symmetry in daughter branching diameters.

Variability in hemodynamic states reported previously [90] has implications for neuroimaging research. Specifically, even exact measurements at an individual point within the limited

neuroimaging field of view (e.g. $\sim 1 \text{ mm}^2$ surface in two-photon images) would be prone to exhibit wide variations. The patchiness (variability) obtained by image acquisition at a single point cannot be overcome by more accurate imaging. Instead, an effective response to counteract variability due to network randomness is to adopt imaging protocols that emphasize spatially distributed samples over point measurements. In other words, measurements intended to infer global trends necessitate spatially distributed samples. Specifically, point observations acquired for single blood vessels can be expected to exhibit wide variations due to network effects, even if measurements are precise.

4.4.3 Hematocrit

Our large-scale computer simulations suggest a depth dependent hematocrit gradient in the cortical blood supply as summarized conceptually in Figure 4.9. Detailed analysis of the spectrum of individual microcirculatory blood flow paths illuminated a clear trend; namely that deeply penetrating microvessels convey more red blood cells than paths running closer to the pial surface. The observation of higher hematocrit in deeper paths was observed in all simulation experiments for the primary sensory sets (experimental data sets, $N=4$; synthetic microcirculatory networks, $N=60$ as seen in Figure 4.5) as well as for the large-scale blood flow simulations covering the entire MCA territory shown in Figure 4.8. The predicted homogenization effect results in more uniform RBC fluxes, because shorter superficial paths tend to have higher bulk flow, Q , but carry less hematocrit, h . On the other hand, longer deeper penetrating paths have to overcome higher resistance leading to lower flows, but enjoy increased hematocrit as summarized in Figure 4.5 and Figure 4.8. As a consequence, this phenomenon also suggests that shorter surface paths which tap into fresh arterial oxygen supply have fewer RBCs, while deeper paths have higher concentrations

of RBCs which on average carry lower O_2 saturation. Another effect of hematocrit gradient is that net oxygen fluxes conveyed to different cortical layers are more evenly balanced than would be the case if RBCs distributed uniformly (no plasma skimming). We also noticed that the variance of RBC fluxes decreased with cortical depth. Accordingly, the distribution of RBC fluxes in deeper layers is more homogeneous than in surface layers. Random network architecture together with non-uniform hematocrit distribution due to the complex biphasic blood rheology seems to be two synergetic factors for ensuring homogenous oxygen supply irrespective of the cortical tissue depth. Since this homogenization effect needs no external feedback, it is plausible to infer that layer dependency of hematocrit and reduction of RBC flux variance serves a self-regulatory mechanism to balance oxygen supply to all cortical layers.

The plasma skimming effect describes a phenomenon seen in microvascular bifurcations ($d < 300 \mu m$) [152,153] in which thinner side branches syphon disproportionately large amounts of plasma from the parent segment than thicker daughter branches. Our mechanistic simulations illustrate how plasma skimming phenomena apply over thousands of bifurcations and multifurcations in a tortuous vessel network, effectively overcoming the geometrical unavoidability of path length differences as shown in Figure 4.9.

Our recently developed kinetic plasma splitting model (KPSM) was our choice for computing large-scale network effects in this study. The main critical reasons include: (i) the KPSM split rule is able to handle multifurcations that occur in the murine microcirculatory anatomy (7.1%, 5.9%, 8.9%, 6.7% of all segments had multifurcations in experimental data sets), (ii) its predictions fall within physiologically meaningful property ranges. Specifically, it does not lead to predictions of zero or excessive hematocrit, and (iii) its linear and differentiable mathematical properties

guarantee convergence of massive network computations. A full account documenting the KPSM model can be found in Section 4.6.

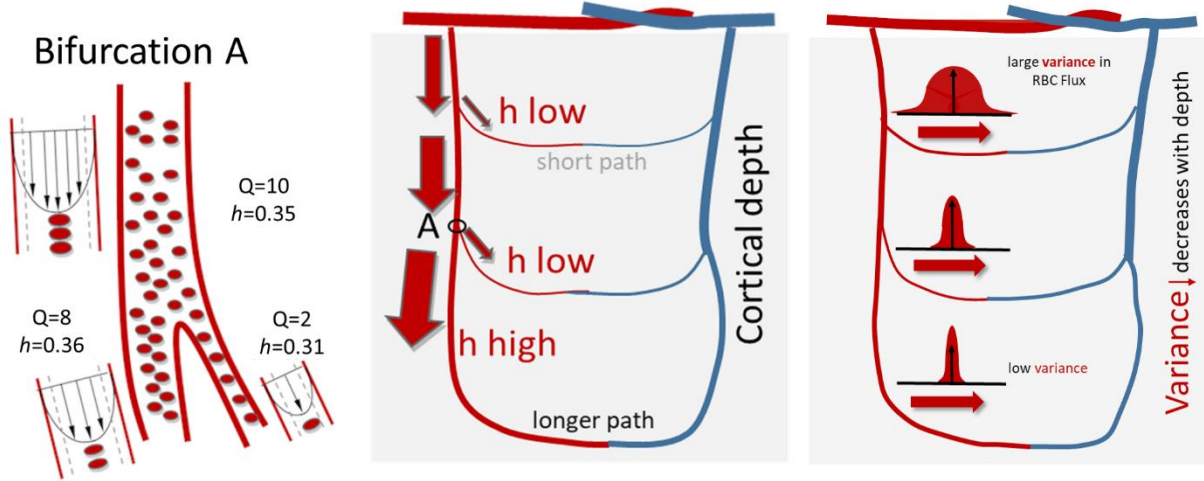


Figure 4.9. Schematic of the depth dependent hematocrit network effect.

At the microcirculatory level, blood is a suspension of red blood cells in plasma with a thin boundary layer close to the vascular wall which contains little or no red blood cells. When red blood cells suspended in plasma flow through a bifurcation of a penetrating artery, they tend to concentrate in the thicker daughter branch, while the thinner side branch syphons a comparatively higher fraction of plasma from the cell free layer near the wall of the parent branch. This effect is known as *plasma skimming*. When plasma skimming repeats over many bifurcations of the cortical microcirculation, deeper reaching paths through the capillary bed tend to have higher hematocrit than surface paths. Longer path length incurs higher flow resistance leading to less bulk flow. In effect, total RBC flux, which is the product of hematocrit times flow, is more balanced than if RBC splits were even. The network effect also reduces variability in RBC fluxes, so that deeper layers are more evenly perfused. Our simulations implicated network effects due to biphasic blood rheology for the predicted hematocrit gradients and increased RBC flux homogenization.

4.4.4 Synthesis

The previously introduced network synthesis used a modified constrained constructive optimization (mCCO) [116] algorithm. The mCCO algorithm originally conceived by Schreiner [145] deploys two very simple principles: (i) minimization of vascular volume, and (ii) hemodynamic flow principle constraints which enforce that the total blood flow entering the

network discharges in exactly equal amounts through the terminal outflow segments. Remarkably, this approach builds network structures whose topology resembles vascular network anatomy observed in vivo. One major task consisted of testing whether realistic network representations with arterial-capillary-venous closures could be synthesized with morphometric and hemodynamic properties matching networks acquired with neuroimaging modalities. The results showed that synthetic data (N=60) created with a modified mCCO algorithm were statistically and hemodynamically equivalent to experimental cortical data sets (N=4). More details on the vascular synthesis algorithm are provided in the previous chapters.

4.4.5 MCA

The hemodynamically inspired vascular growth procedure enabled the construction of realistic representations of the cortical blood supply of the entire MCA territory spanning multiple length scales from the large arteries (mm range) to the smallest capillaries (μm range), and draining through the pial veins (mm range) or three orders of magnitude in length scales. It allowed us to seamlessly integrate state-of-the-art topological data acquired from two entirely different imaging modalities (μCT and 2PLSM) into a single, coherent multiscale representation of the entire MCA territory with unprecedented anatomical detail that includes both the arterial and the venous side of the cerebrocirculation. Because simple, blood flow inspired construction principles are applied at all length scales, the resulting MCA circulation has no discontinuities or gaps between the main cerebral arteries, the pial arterial network, or the microcirculation. Morphometrics, anatomical details such as the shape of the cortical surface and hemodynamic principles, are incorporated at each stage of the growth algorithm. Thus, our proposed methodology may serve as an alternative to the practice of merely stitching together data from different locations or length scales.

The application of biphasic blood flow simulations for the entire MCA territory shows that large-scale blood flow and hematocrit simulations are feasible with existing computer resources. The large-scale simulations confirmed the trend of hematocrit layer dependence predicted for the smaller cortical samples. The massive simulations also elucidate the spatiotemporal coordination between different vascular compartments at different length scales (arteries vs arterioles vs capillary bed). The anatomical detail achieved with the MCA model may serve as a starting point for dynamic simulations that elucidate the involvement of different vascular components in regulating functional hyperemia, autoregulation or collateral blood supply in stroke. Because the network extended over a sizable portion of the mouse cortex, predictions for the center of the primary sensory cortex were free of boundary effects.

4.4.6 Boundary conditions

The synthetic MCA circulatory network also has the critical advantage that boundary conditions, which have been reported to hamper simulations on thin data sets [95], are applied very far away from the area of investigation. For example, Figure 4.6 displays typical subsections comparable in size to the 2PLSM data sets which are located far away from the MCA boundaries (MCA M1 segment and veins of the superior sagittal sinus). Thus, in samples situated at the center of the MCA territory, boundary conditions have negligible impact on hemodynamic predictions. The blood flow simulation for the entire MCA territory required only the arterial inlet pressure at the M1 segment and the blood pressure at the venous side.

We point out three additional reasons why the ability to synthesize morphologically and hemodynamically equivalent data sets is significant. (i) Artificial networks continuously connect the arterial side and the venous side without gaps. In 3D neuroimages assembled from two-

dimensional image stacks, it is easy to miss segment connections or segments running between two slices. (ii) No segments are severed nor is there a need to prune dangling segments at domain bounds (this cleanup is unavoidable in image reconstructions [89,109]). In particular, fragmentation to pial arteries and many microcirculatory segments running perpendicular to the pial surface lead to boundary effects that can substantially affect predictions [95]. (iii) The most important benefit is the ability to expand the scope of data acquired by neuroimages without being confined to the bounded field-of-view or limited resolution of the imaging modality.

The ability to conduct brain-wide simulations would free the modeler from the burden of making uncertain assumptions at the boundaries of the artificial domain (edge of the image or simulation domain boundary). Because our algorithm succeeded in converging blood flow computations with hematocrit split for the entire MCA circulation in about two hours of CPU time, our group is confident that the proposed computational approach will enable blood flow simulations and oxygen transport on a brain-wide level in the near future.

4.4.7 Limitations

Despite the evidence for trends such as depth dependent hematocrit, it should be emphasized that individual flow paths may experience substantially weaker or even reverted trends, as can be expected from the inherent randomness of the microcirculatory network architecture.

The 2PLSM technique provided a very detailed inventory of the cortical microcirculation. The four data sets did not include information about the subcortical blood supply to the white matter. White matter subcortical circulation is physiologically separated from the cortical blood supply. Accordingly, we assumed that the white matter supply is hydraulically separated from the cortical blood supply. However, certainty about this point would require a model of both the cortical and

the subcortical networks (white matter blood supply). This task is intriguing, but is currently beyond the reach of 2PLSM, which is limited to ~ 1 mm depth. This is clearly a point for future research, but is currently outside the scope of this paper.

The main finding of depth dependency of hematocrit supply to the cortical layers is the result of a model prediction whose basis rests on experimental observations about plasma skimming and uneven hematocrit splits observed in capillaries outside the brain [154–156]. Therefore, the next logical step is to experimentally verify layer dependent hematocrit with deep imaging such as adaptive optics (AO) two-photon imaging [157]. If experiments confirm depth dependence and homogenization of RBC flux distribution, it would constitute a remarkable mathematical modeling contribution, which actually predicted, instead of merely explained, cortical blood supply. In the adverse case, the model would have prompted the need to revise our understanding of biphasic blood flow rheology as it relates to the cortical microcirculation (=diameter and hematocrit dependent viscosity laws, and hematocrit split rules), since so far it has been assumed that plasma skimming is active in capillaries throughout the entire circulatory system including the brain.

The conclusions about oxygen supply also need to be verified experimentally and computationally. The methods presented previously might be a first step in this direction [90]. However, oxygen predictions require discretization of the extracellular space which can be done in principle using the methods presented in Gould et. al [115], but is beyond the scope of this paper.

4.4.8 Conclusions

We predicted uneven depth dependent hematocrit distribution due to the complex biphasic blood rheology. Because our simulation did not include external factors such as gravity, we

conclude that the result of depth dependent hematocrit arises from the combination of structural and hemodynamic properties of the network. Our findings suggest that network effects due to biphasic blood rheology and randomness of the network architecture are a controlling factor for ensuring adequate oxygen supply irrespective of the cortical depth. Since the observed homogenization of RBC variability requires no feedback, depth dependent hematocrit gradient may serve an important self-regulatory mechanism to balance oxygen supply to all cortical layers.

Uneven distribution of hemodynamic states in the microcirculation as well as the notion of layer-dependent hematocrit also have implications on the interpretation of the fMRI BOLD signal where it is usually assumed that hemodynamic states and hematocrit are homogeneous and evenly distributed throughout the microcirculation. The predictions in this work suggest that focal analysis of the fMRI BOLD signal would be more relevant than assuming global constants for the entire cortex.

We demonstrated that the modified constrained constructive optimization algorithm (mCCO) is successful in synthesizing artificial microcirculatory networks with topological and hemodynamic properties that are statistically equivalent to experimental data sets from different imaging modalities and length scales.

Simulations of the entire MCA circulation, which until recently would have to be considered intractable, are now becoming accessible to rigorous numerical analysis due to stable, efficient and physiologically consistent plasma skimming algorithms implemented on existing computer hardware. The synthesis of anatomically faithful cerebrocirculatory networks with desired topology closes the gap between large-scale blood flow simulations performed on image-derived data sets on one hand, and simulations on purely synthetic data sets on the other.

The successful synthesis of biphasic blood flow in the entire territory of the MCA constitutes a step towards the ultimate goal of first principle simulations of cerebrocirculatory blood and oxygen distribution patterns for the entire brain.

4.5 Deficits in different hematocrit split rules

We first attempted to make use of a hematocrit splitting rule for biphasic blood flow computations derived from experiments in the rat mesentery. At first, we tried three different versions of the Secomb-Pries split rule developed for small networks of the rat mesentery which do not exhibit multifurcations, and for which numerous different parameter choices can be found in the literature [158,109,159–161]. However, none of the different versions can be applied to in vivo data sets with multifurcations (654, 725, 1868, and 1440 in the four experimentally-derived microcirculatory networks). We also tried to simulate biphasic blood flow of two midsize network of a cortical microcirculation with about 2000 and 10,000 splined segments, see Figure 4.10 and Table 4.4. However, all simulations with the Secomb-Pries splitting rule contained segments with physiologically problematic results. Table 4.4 summarizes convergence results and problematic predictions by their splitting rule. For the large-scale MCA blood flow simulations, the Secomb-Pries model rapidly diverged in all simulation runs.

Zero RBC flow. For example, the 2005 version [158] had numerous segments without RBCs. The 2011 version [1] and the 2015 version [2] did not converge for larger networks (Table 4.4).

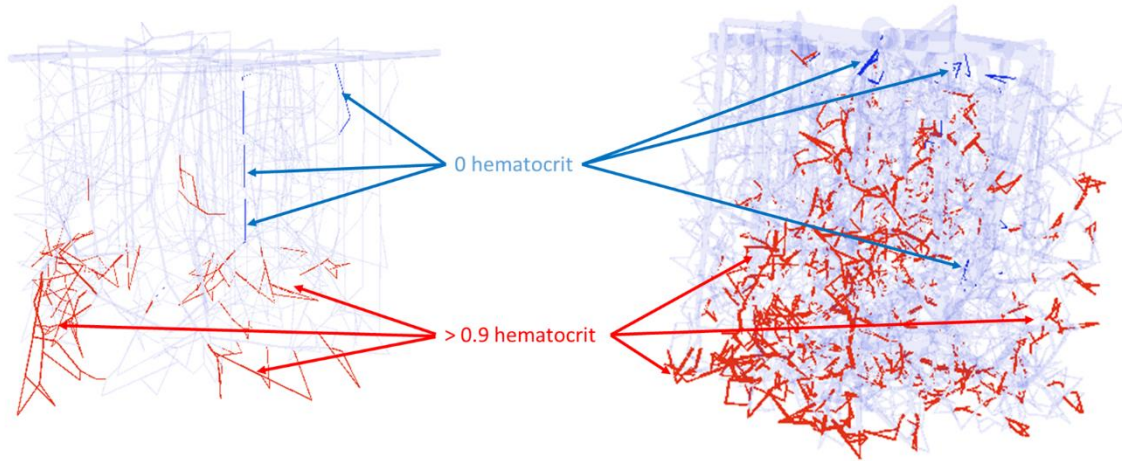


Figure 4.10. Hematocrit fields portrayed on two simplified microcirculatory models. Obtained with the Pries 2005 splitting rule [158] showing many segments with non-physiological results. Many segments had zero RBC flux (Hct ~ 0), other segments reached hematocrit levels that fell outside the definition space of the hematocrit-dependent viscosity (Hct > 0.9). Red indicates excessive hematocrit and dark blue indicates vessels with zero RBC flux. Taken together, these data and more results listed in Table 4.4 explain why the Pries splitting rule, with any combination of parameters, was not applicable to simulate large-scale network effects in the mouse cortex. Instead, the KPSM model with implementation in Section 4.6 was used which gave consistently, physiologically meaningful results in all cases as listed in Table 4.4

Excessive RBC flow. Moreover, all predictions based on the Pries family of models [109,158,159] led to segments with excessive discharge hematocrit greater than 0.9 ($h > 0.9$). This value is outside the validity range of the viscosity law, which is only valid [154–156,162] for hematocrit up to up to 0.7 ($h \leq 0.7$). The 2005 version [158] of coefficients had up to 11.8% vessels above the excessive hematocrit threshold. Numerical complications were also noted in a study [1] which deployed the Pries split model, stating the following comments on the splitting rule:

“... a threshold of 0.8 for the hematocrit in the daughter branches was prescribed ...”

“... hematocrit was set to zero in daughter branches with flow below a given threshold ...”

In a recent report [161], the authors retorted our earlier findings [115] about the inconsistencies of their split rules. They updated their formulae with a new set of three parameters in their highly non-linear functions to reduce the number of occurrences of RBC free vessels. The effort to eliminate RBC free vessels stands in contrast to another recent review [163] which suggests that zero RBC predictions are a necessary feature as expressed in the following quote:

“This approach (KPSM model by Gould et al, 2015) excludes the possibility of zero hematocrit in low-flow branches, whereas such behavior can be observed in vivo...”

In our view, zero RBC flow prediction for a stationary simulation is not physiologically meaningful. The prediction of RBC free blood flow points to a structural problem of the empirical split rule by Pries that cannot be remedied by parameters adjustments. Inspection of the split formula shows a discontinuous “if-statement” which sets one branch of the Secomb-Pries split formula artificially to zero to avoid transgressing the definition space of the *logit* function.

We have no further comment on the applicability of the Secomb-Pries split rule for single bifurcations or small mesentery networks. However, simulations for large cerebral microcirculatory networks as shown in this and in prior studies [90,115] point to two principal reasons why the KPSM model was chosen:

- First, the KPSM model can address multifurcations which occur in realistic data sets (654, 725, 1868, and 1440 in four vibrissa primary sensory cortex networks), while the Secomb-Pries models in all their variants cannot.
- Second, the implementation of the KPSM model with a single parameter, m , predicts hematocrit distribution within physiological ranges for large cerebrocirculatory networks.

Table 4.4. Physiological assessment of plasma skimming models

	Segments with h=0			Segments with discharge h > 0.9		
Simplified Microcirculation 1						
KPSM	✓	✓	✓	✓	✓	✓
Pries [158] 2005	●	✓	2	●	231	217
Lorthois et. al [109] 2011	●	●	●	●	●	●
Chebbi et. al [159] 2015	●	●	●	●	●	●
Simplified Microcirculation 2						
KPSM	✓	✓	✓	✓	✓	✓
Pries [158] 2005	316	●	54	1397	●	2159
Lorthois et. al [109] 2011	●	●	●	●	●	●
Chebbi et. al [159] 2015	●	●	●	●	●	●
Massive MCA Simulation						
KPSM	✓	✓	✓	✓	✓	✓
Pries [158] 2005	●	●	●	●	●	●
Lorthois et. al [109] 2011	●	●	●	●	●	●
Chebbi et. al [159] 2015	●	●	●	●	●	●

V1 = Pries In Vitro, **V2** = Pries In Vitro Modified, **V3** = Pries In Vivo, ●=not converged, ✓=converged

without faces ($h \sim 0$, $h > 0.9$, h discharge hematocrit)

4.6 Implementation of the biphasic blood flow

Biphasic blood flow was solved computationally by enforcing three conservation laws: conservation of mass, linear momentum, and RBC fluxes using a plasma skimming model. For the RBC splitting rule, we used the KPSM model [115] with a constant plasma skimming coefficient,

m , although the accuracy could be further improved by adjusting the m -value as a function of hematocrit and diameter as shown by Yang et al. [128–130].

The system of conservation laws in Eq. (1) in S2 Supplement is highly nonlinear and coupled. We showed previously a beneficial decomposition technique that enables the consecutive but separate convergence of linear algebraic subsystems [115]. The advantages of our implementation include the ability for each subsystem to be solved separately with highly efficient sparse iterative linear algebraic solvers. Additionally, no derivative information is necessary. In effect, we perform the fixed point iteration algorithm depicted in Fig A in S2 Supplement. Given an initial hematocrit field (initially set to systemic hematocrit $h = 0.35$), solve for the pressure in the first equation set using the resistance matrix, R , which incorporates the nonideal hematocrit dependent viscosity law. With the converged pressure field, p , compute the flow field, Q , using the connectivity matrix C_2 . Finally, update the hematocrit field, h , using the kinematic plasma skimming (KPSM) law to determine how the hematocrit in parent segments splits into two or more daughter branches. In order to limit the magnitude of hematocrit updates, we use an underrelaxation parameter $\alpha=0.5$ for stabilization. More details on implementation and stabilization can be found elsewhere [115].

$$G(Q, p, h) = 0 \begin{cases} R(h, d)Q - C_1 p & = 0 \\ C_2 Q & = 0 \\ C_3(Q, d)h & = 0 \end{cases} \quad (1)$$

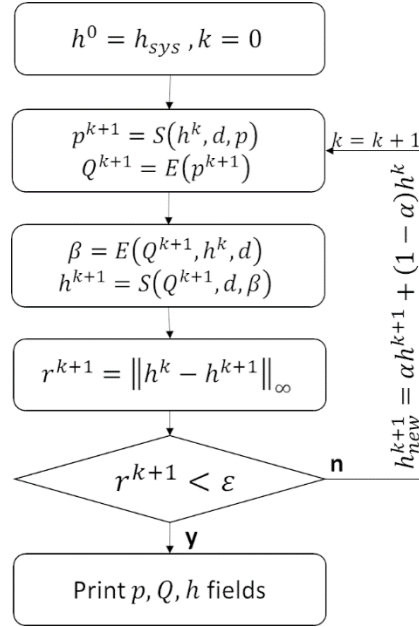


Figure 4.11. Flow diagram for main steps in the fixed point iteration for biphasic blood flow computations.

Here, E means an evaluation step; S signifies simultaneous solution of linear algebraic equation sets.

We also implemented cyclical boundary conditions at side faces perpendicular to the pial surface. This was achieved by connecting boundary segments located near one vertical face of the domain to a corresponding node at an opposite face. Matching boundary blood vessel facets in all cortical layers, the domain boundaries were effectively extended to infinity, without losing blood flow or RBCs. We also implemented no flux boundary conditions and compared the results to cyclic boundary conditions. Because of the large domain size and similar conditions governing opposite faces, simulation results between those two choices were very close (total flow differences less than 0.0001%).

4.7 Hematocrit dependence on diameter

This section plots hematocrit distribution as a function of diameter. Fig A in S1 Supplement reflects wide variability in discharge hematocrit levels. Path analysis discussed in the main paper in Fig 2 and Fig 3 revealed a depth-dependent trend in hematocrit distribution independent of diameter.

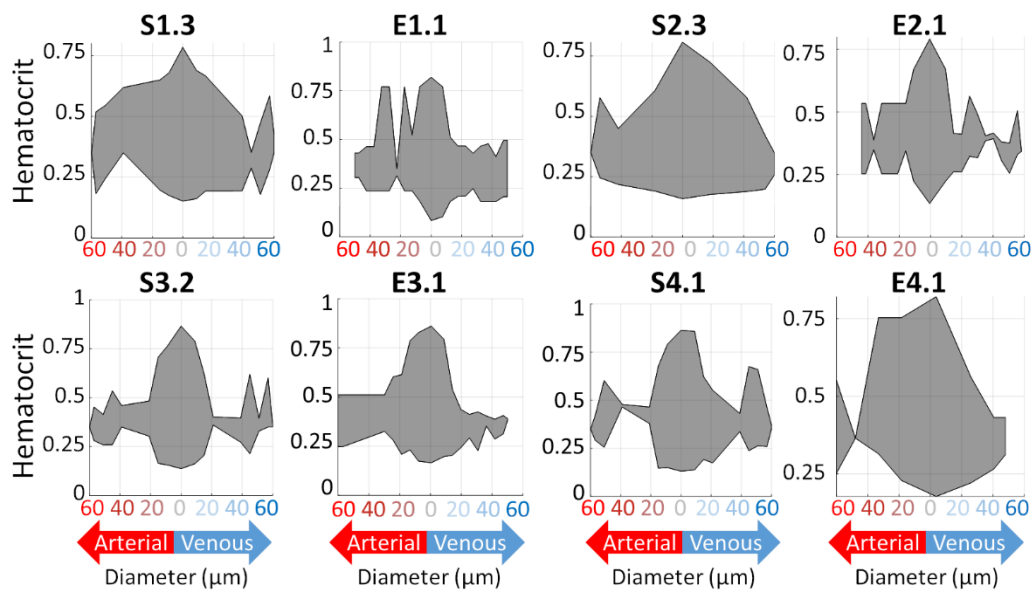


Figure 4.12. Diameter dependence of hematocrit in microcirculatory networks. The analysis confirms the high degree of hematocrit variability along the microcirculatory bed. Diameter hierarchy did not correlate with hematocrit level ($N = 2,300\text{--}22,052$ paths per dataset).

5 Improved oxygen simulation for large microvascular networks

5.1 Introduction

Many late-onset neurodegenerative diseases involve degradation of the coordination between vascular blood flow and neural tissue oxygenation. These diseases are hallmarked by the generation of hypoxia-related events, such as an increase in beta-amyloid production in Alzheimer's Disease (AD) [1,2]. Many forms of dementia suffer from decreased oxygen tension while also exhibiting significant morphological changes in vascular structure. Unfortunately, healthy aging (aging without dementia) also suffers from similar morphological changes [3]. While the link between vascular restructuring and changes in cerebral oxygen content have been established, the contribution of each factor to abnormal oxygen distribution is not understood. In order to develop methods for treating these age-related dementia, it is imperative to quantitatively explore changes to the vascular topology and the corresponding alteration to oxygen tension in the brain.

Multimodal medical images are now capable of acquiring detailed measurements about the microcirculation such as oxygen concentration in the vasculature, in the extravascular space at different positions in the cortex and at different time points, hematocrit measurements, blood flow velocities, etc. [1,2,4–8]. However, multimodal imaging typically collects these data from different specimen or at different times or even different locations or imaging modalities. In order to reconstruct a closed picture of the interesting neurovascular coupling, it is important to not interrogate these measurements one-by-one, but instead to create a close understanding of the physical phenomena pertaining to cerebral autoregulation and functional hyperemia.

Many groups have pursued methods of assembling or integrating measurements from multimodal imaging on the computer to create a mechanistic model of blood flow and oxygen exchange [9–20]. Computational modeling has the advantage that the measurements at different length scales and using different imaging modalities can be combined simultaneously and a causal relationship between blood flow and oxygen relationships can be obtained to test medical hypothesis as well as predictions from a physico-chemical point of view. Therefore, we pursue here a combined approach that uses medical image data from multiple sources but integrates them of a mechanistic model of the mouse cortex.

5.1.1 Computational paradigms

In order to capture the inherent heterogeneity of the neurovascular unit, anatomically consistent neurovascular models have been proposed [9,21–25,31–33,164,165]. Analytic 1D-3D approaches are capable of resolving complex heterogeneous oxygen tension but are computationally prohibitive for models larger than a few hundred segments [25,31–33,164,165].

Body-fitted unstructured meshes offer more expandability than analytic methods, but are limited by the massive number of elements needed to resolve the tortuous extravascular space and interface with the dense vascular structure [9,23,25]. For instance, the extravascular space of a $\sim 4 \times 4 \times 4 \mu\text{m}$ cube was simulated with ~ 82 million tetrahedrons recently [30].

Homogenization methods have been proposed to overcome the spatial limitations [36,37] but unfortunately, these methods does not accurately represent the heterogeneous flows in the realistic microvascular structure. In other words, the homogenous medium, even endowed with an anisotropic diffusion tensor, does not represent the discrete and deliberate paths blood travels in the cerebral microcirculation.

Another approach that we have pursued for large vessels, but is not shown here, is parametric meshing of the vasculature which was shown to drastically reduce the mesh size required for modeling tubular network structures as shown in the cerebroarterial tree simulations stemming from the Circle of Willis and extending to throughout the pial territories of the MCA, PCA and ACAs. This approach is not pursued here but references can be found elsewhere [29,34,35].

Here we propose the use of a parametric meshing technique, specifically a Cartesian mesh, which is capable of drastically reducing the mesh cells that would otherwise be needed, like in an unstructured tetrahedral mesh. The advantages of parametric meshes have been recognized and discussed in prior work, but proposed here for the first time for the interface between the intravascular and extravascular space. The parametric meshing is giving two advantages: (i) the mesh size to resolve the interface between the endothelium and the extravascular space is much smaller than when using tetrahedral meshing and (ii) the connectivity of the Cartesian mesh is fixed to 8 faces, whereas the bandwidth in the tetrahedral mesh is not controllable which impacts not only the size of the model but also the solvability (as visualized in Figure 5.2). Therefore, we advocate the use of a Cartesian mesh that is interfaced with the vasculature through a binary mask of the endothelium. The use of Cartesian mesh for the endothelium and extravascular space has the second advantage that it does not require the introduction of singularity removal techniques because the interfaces are sharply defined and fluxes can be formulated on each face rigorously.

This new methodology will allow simulations of large cortical microcirculatory networks, extending the interior domain whose predicted oxygen tension is primarily a result of network topology, and not boundary effects; a breakthrough necessary for investigating oxygen tension in age-related structural changes to vascular morphology.

5.2 Methods

5.2.1 Oxygen tension measurement acquisition in young and aged brain

A custom two-photon imaging system was used to image oxygen content in the murine cortex through a thinned skull window. The light sequence employed 150fs pulses at 80 MHz with a wavelength of 820nm. The oxygen content was imaged with a focal window of 400x400 μ m up to a depth of 300 μ m using 30-40 μ m intervals. The phosphorescence lifetime microscopy required a slow injection ~300 μ m below the cortical surface of PtP-C343 dye (~150 μ M concentration). The lifetime recordings converted to oxygen tension (pressure, mmHg) using a calibration curve.

5.2.2 Vascular structure acquisition

Four sections of the murine vibrissa primary sensory cortex [26] were reconstructed from two-photon laser scanning microscopy (2PLSM) images. These images represented the length and orientation of the blood vessels [26,124,125]. These structures were then labeled with an automated algorithm as described in the original manuscript. The categorization used size and branching level information (Strahler order) to differentiate between pial vessels, penetrators and capillaries. Capillaries were identified by a diameter cutoff of 6 μ m and penetrating vessels differentiated from pial vessels with depth and diameter thresholds. The final reconstructed network topology and diameter information was stored using sparse connectivity matrices. More details on image acquisition [26,124,125], image reconstruction [126], as well as the formulation of the network equations [115] can be found elsewhere.

5.2.3 Mathematical blood flow and oxygen model

The blood flow will be modeled with Poiseuille flow throughout the vascular structures as given in [21,27] in Equation (7.564). Oxygen enters the domain through the vasculature where it moves via convection through the network and permeates the blood brain barrier (BBB) through a mass transfer flux (=a diffusive-like flux) into the surrounding tissue (=extravascular space) as in Equation (7.565). The boundary conditions follow previously published values [22].

$$\Delta p = \alpha f; \quad 0 = \vec{\nabla} \cdot f \quad (36)$$

$$UA \frac{c_v - c_t}{t} = \vec{\nabla} \cdot (c_v f) \quad (37)$$

Here, f is the bulk flow field derived from Equation (7.564), U is the transmembrane permeability of the endothelial layer, t is the endothelial layer thickness, A is the endothelial layer surface area, c_v is the concentration of oxygen in the vasculature, and c_t is the concentration in the tissue.

The oxygen that enters the tissue domain (through aforementioned mass transfer across the BBB) where it is permitted to diffuse while being metabolized following Equation (7.566). This model is a diffusion-reaction domain with a mass transfer source from the vasculature:

$$\vec{\nabla} \cdot (D \vec{\nabla} c_t) + UA \frac{c_v - c_t}{dx} = -k_{met} c_t V \quad (38)$$

Where D is the isotropic diffusivity, c_o is the oxygen concentration, k_{met} is the 1st order cerebral metabolic rate of oxygen (CMRO₂), and A_{vasc} is the cross sectional area for the vasculature. All

physiological parameters are similar to previously reported values for oxygen in the brain as reflected by Table 5.1. These values were estimated from experimental data and were found within the range of reported values.

Table 5.1. Parameters used in the prediction of oxygen tension throughout the murine cortex

Property	symbol	value	units	ref
Diffusivity	D	18	femtoMoles/fL	[166]
Transmembrane permeability	U	2.4	femtoMoles/fL	[167]
Metabolic rate	k_1	14.17	1/ms	[9,25]
Endothelial layer thickness	t	1	μm	[62]
Pressure drop (BC)	p	115	mmHg	[22,115]
Concentration inlet (BC)	c	68.5	mmHg	[115]

5.2.4 Vascular masking for a Cartesian mesh

To accurately represent the mass transfer between the largest and smallest vessels alike, an edge detection algorithm was identified for resolving the endothelial layer inside a Cartesian mesh. This was achieved through a masking procedure that detects the vessel edge using a fuzzy logic.

The algorithm cycles across every vessel (=every cylinder in network) and for each vessel, it defines the coordinates (i, j, and k indices in Cartesian mesh) of the bounding box encompassing the cylinder. These values are calculated using the radius and the centerline with the help of a *getSurroundingCell* computation, which can be directly computed from the Cartesian mesh.

The distance is then calculated between each mesh element center and the vessel centerline. If the cell center is inside the endothelial layer (defined by a thickness from the vessel edge), then it is labeled as an edge volume. If it is not an edge but is inside the vessel radius, it is labeled with as an interior node. The remainder of nodes are, by default, labeled an extravascular node. A pseudocode is offered in Appendix A. Note, each cylindrical vessel is endowed with a sphere at

each terminal to ensure a smooth connection between adjacent segments. An example of mesh labeling is offered in Figure 5.1A and C.

The fluxes in the mesh are described in one of four methods: (i) diffusion exists between two adjacent extravascular elements, (ii) mass transfer exists between an endothelial element and an extravascular element or mass transfer elements, (iii) a mass transfer flux (hindered diffusion) exists between adjacent endothelial elements, and (iv) no flux between any cell and an intravascular element.

In the event a segment does not span more than 1 mesh element (the diameter is thinner than the mesh edge length), the mass transfer is discharged entirely into the single mesh element. If many mesh elements are assigned to the endothelial layer of a single vessel, the total area is evenly divided amongst the mass transfer mesh elements. When forming the equations for the mesh elements identified as blood vessel elements, a simple equation is used to assign the vascular element value to the mesh element value:

$$c_{tiss}^j = c_{vasc}^i \quad (39)$$

Note, all mesh elements other than vascular (blood) elements observe reactions.

Vascular segmentation. In order to ensure similar characteristic length between the Cartesian mesh and vascular network, we propose a method for re-segmenting the vascular structure dynamically (re-cutting the segments) defined by the Cartesian mesh. The method, similar to the mesh labeling algorithm, can identify the mesh indices (i, j, and k dimensions for x, y, and z dimensions, respectively) of the start and end point in the network. If the difference between start and end indices in any dimension is greater than 1, the vessel is split in half. The process is

recursively completed until the sub-segments all span a maximum of 2 adjacent mesh elements. An example of vascular segmenting is offered in Figure 5.1B and C.

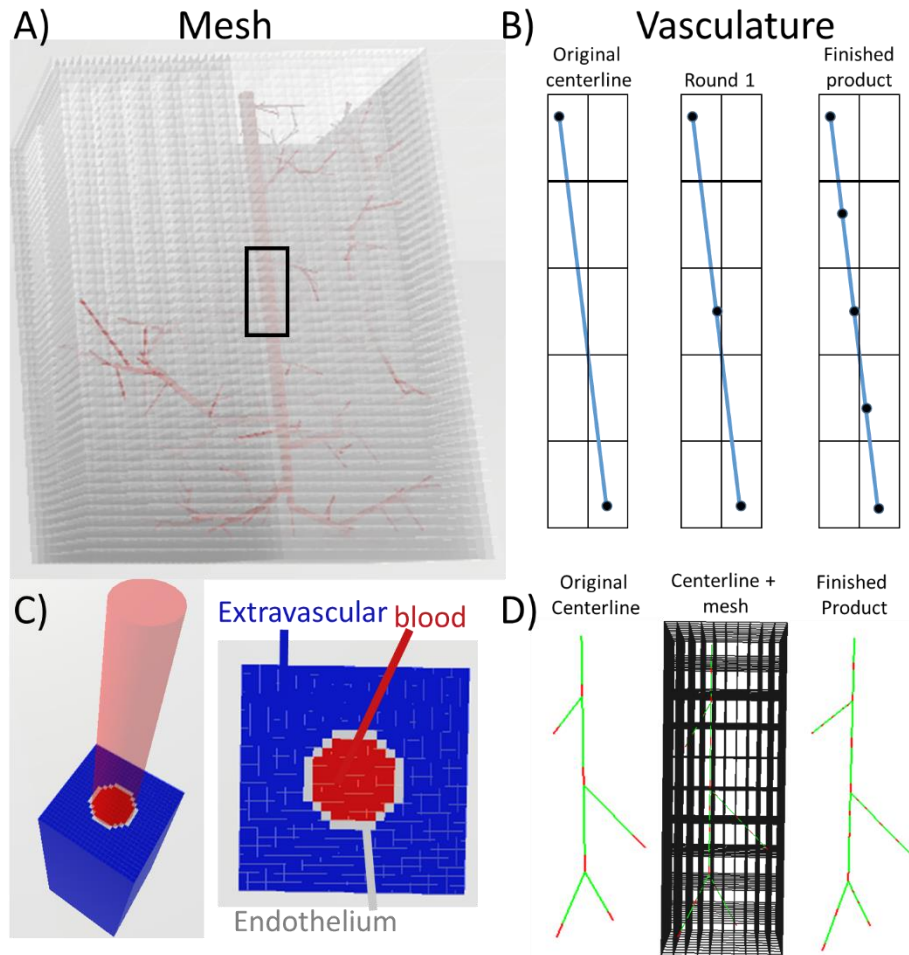


Figure 5.1. Vascular masking of the Cartesian mesh and vessel segmentation.

A) A penetrating arterial tree with the overlay of a Cartesian mesh reflects a network whose straight segments span more than two adjacent volumes. B) The vasculature is segmented (cut) until no single segment spans more than two adjacent volumes. C) A labeled vessel inside a mesh shows the extravascular elements in blue, intravascular segments in red, and endothelial segments in grey.

D) The original centerline before and after segmentation compared to the mesh used for segmentation of a penetrating arteriole sub-tree reveals the network is segmented to a higher density after the algorithm. The coloration of the centerline is green (segment origin) and red (segment terminal).

5.2.5 Implementation

Equation generation was written with proprietary object-oriented codes. All linear algebraic computations were performed in PETSc using the GMRES solver and a block Jacobi preconditioner [168,169]. The linear algebraic system for highly dense meshes (large number of elements) does not directly converge in some cases. To converge these simulations, we offer a simplified mesh multigrid technique where coarse mesh solutions are linearly interpolated to initialize the solver of a denser mesh. Our method differs from the classic multigrid as it does not solve a dual-stage problem using the residual vector to improve the update direction, but rather uses consecutively more refined meshes using an interpolated solution vector to initialize the solver (=initial guess). This method does not improve convergence but instead stabilizes it, preventing divergence by filtering high frequency updates and preventing them from overtaking the solution. We note the user could use deflation or algebraic multigrid to converge such a system, but these methods are outside the scope of this work.

In order to account for the intravascular volume contained within mesh elements who entirely enclose vessels, the volume of the vessel within each mesh element is removed from the respective mesh element volume.

When simulating occlusions, many vessels remain with no flow, causing singularity in the convection equations. To resolve this, these vessels are endowed with special equations replaced by equations of assignment to the same concentration as the upstream vessel. This maintains numerical stability in the equations.

To avoid solving inconsistencies during multigrid solving (due to the vascular segmentation procedure) and enhance numerical stability, the blood flow is solved only once at the coarsest vascular resolution. When segmenting the vasculature, the flow and pressure of each newly-cut segment can be directly calculated from the coarse resolution.

5.3 Results

The Cartesian mesh discretization significantly improves the structure of the diffusion matrix formulation as opposed to the tetrahedral mesh formulation as exemplified by Figure 5.2. The matrix representation of finite volume diffusion in the tetrahedral mesh has a highly irregular structure while that of the Cartesian mesh shares a tri-diagonal block structure. This is more amenable to block preconditioners and, due to the regularity in the coefficient magnitude, a more stable convergence. Moreover, the number of elements in this case study is significantly larger in the unstructured mesh than the Cartesian mesh, revealing a significant problem reduction of at least 8-fold when using the proposed meshing scheme. In other words, a hexahedral mesh is, by default, at least $1/8$ the size of a tetrahedral mesh encompassing the same space with the same characteristic edge length. Specifically, the breaking of a hexahedral domain into tetrahedrals using ANSYS ICEM [Canonsburg, PA] resulted in a ratio of tetrahedral volumes to hexahedral volumes of 9.2:1 when using similar characteristic edge length. Note, for reference the labeling of the Cartesian mesh does not break the diagonal dominance or structured block tri-diagonal format of the matrix (as seen in Figure 5.2, Right).

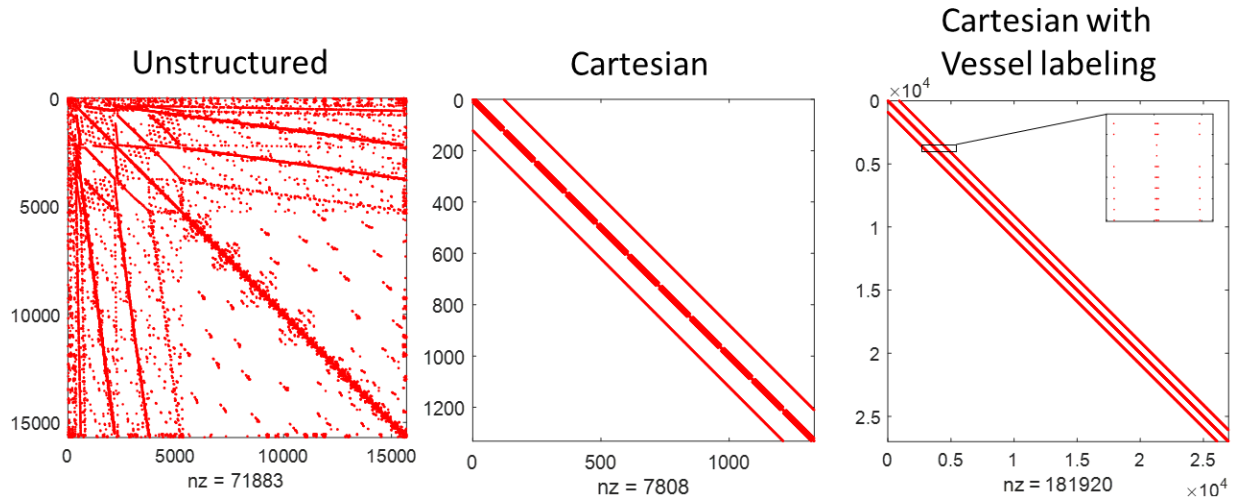


Figure 5.2. Matrix structure of diffusion with different mesh types.

Left) tetrahedral mesh compared to a Middle) Cartesian mesh of the same size and similar characteristic edge length. Right) The labeling procedure does not destroy this structure. Note, the labeled mesh is applied to a realistic vascular structure, resulting in a larger domain and larger mesh size.

The masking procedure has been applied to a mesh surrounding one of the empirical networks. This labeling and the associated network are given in Figure 5.3. The smaller vessels discharge into a single endothelial element (isolated white elements) while larger vessels that span many elements lead to elements labeled as endothelium and vasculature (blood). Note, the majority of the mesh elements are labeled as uninterrupted extravascular elements.

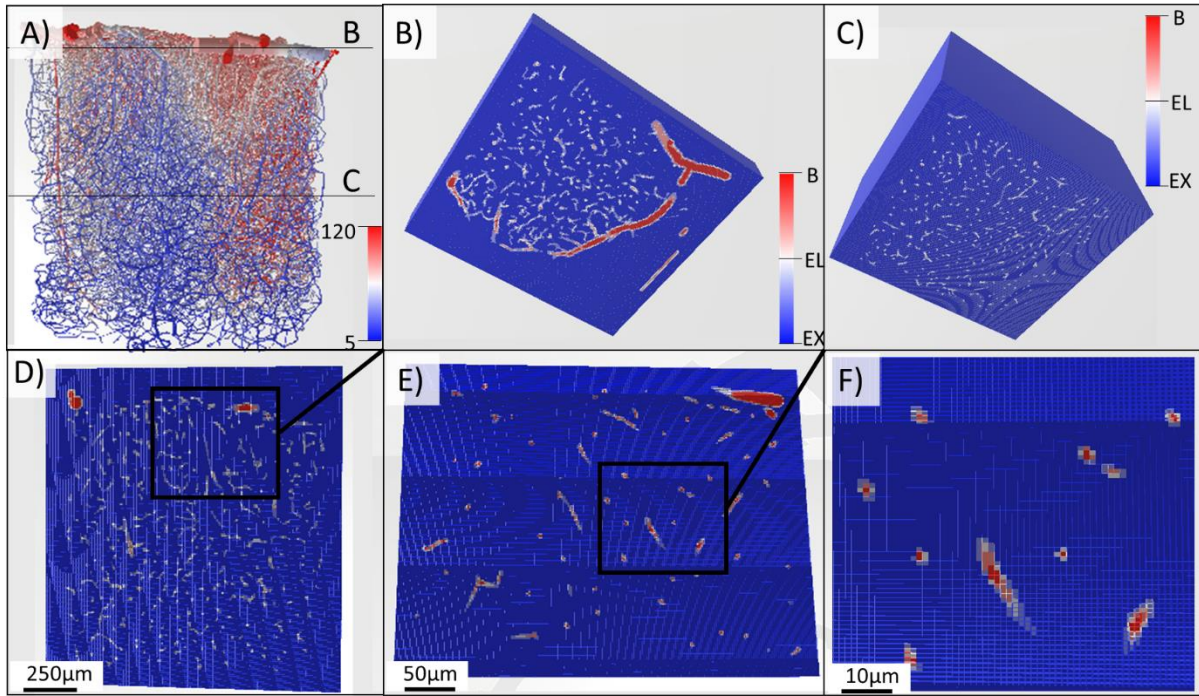


Figure 5.3. Example labeling of a cartesian mesh at different density levels.

A-C) The full E1.1 network has labeled a mesh visualized at many resolutions with the majority of the structure accounting for extravascular space (blue), some intravascular elements (red) and mass transfer elements (grey). D) The vascular structure of E1.1 reflects the blood pressure field flowing from high pressure (red, 120mmHg) to low pressure (blue, 5mmHg). E) The labeling near the surface shows large vessels constituting many intravascular mesh elements while F) Deeper into the cortical surface the structure shows less large vessels and more capillary mass transfer elements.

Predictions of oxygen tension utilizing this mesh masking technique shows qualitative and quantitative similarities to experimental distributions as reflected by Figure 5.4. The simulation distribution shows an excellent agreement with the experimental data with a few exceptions attributed to differences in vascular structure above and below the focal plane. The parameters used to align these two data are listed in Table 5.1 and agree well with empirically derived values.

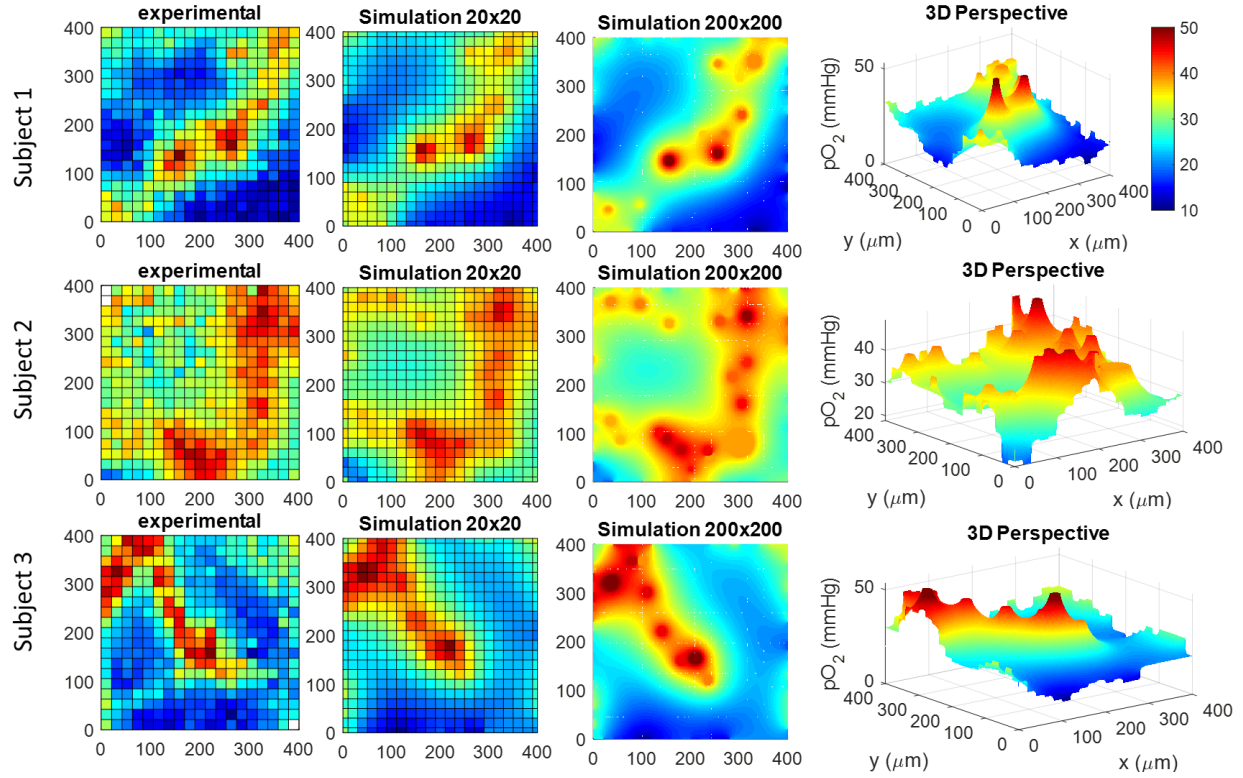


Figure 5.4. Validation of simulation paradigm against three sets of experimental data. The experimental results were obtained with 2-photon phosphorescence lifetime oxygen imaging. The simulations were performed with two different mesh resolutions; a coarse 20x20 and a dense 200x200. The close match between the concentration profiles was used to determine tissue diffusivity, reaction rate, and mass transfer coefficient. The 3D perspective illustrates the steep concentration gradients along two penetrating arterioles.

The investigation mesh convergence revealed excellent stability as visualized in Figure 5.5. The individual plots of rays through the extravascular block reveals a convergence each ray to a unique profile. The rays also show peaks near penetrating arterioles and valleys near ascending venules.

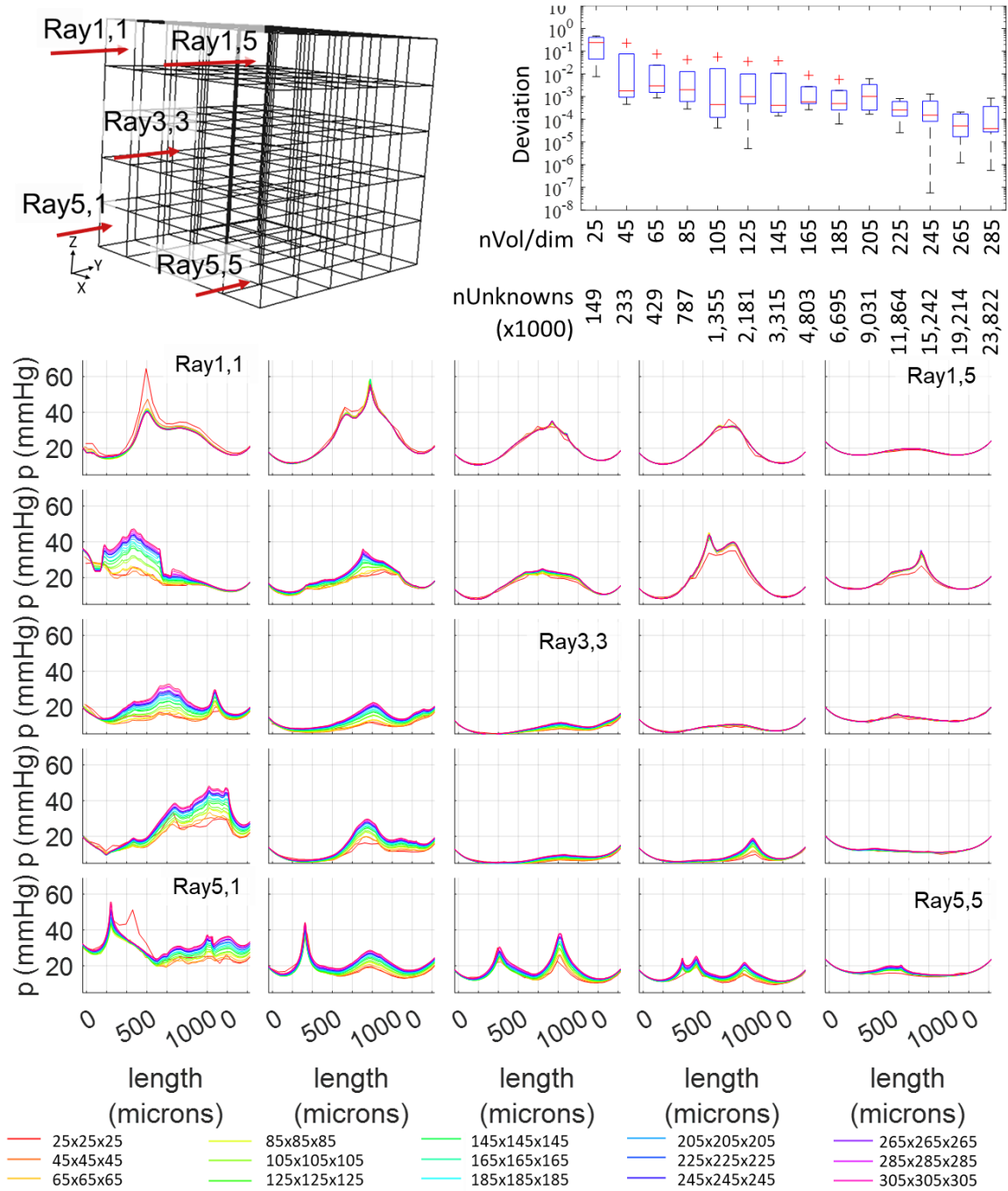


Figure 5.5. Detailed analysis of extravascular space in an oxygen simulation for a single empirical network.

A) The 3D image shows the dispersion of oxygen originating in the vascular inlets and dispersing throughout the tissue domain. B) The convergence of values for many rays at $y=718\mu\text{m}$ shows stable convergence and an acceptable tolerance at ~ 105 mesh elements per dimension. C) Raytraces through the domain exemplify the spatial complexity of oxygen tension. Note, the plateau in ray 6 corresponds to a vessel segment aligned the ray. All results show a deviation from the highest density mesh ($305 \times 305 \times 305$ volumes per dimension).

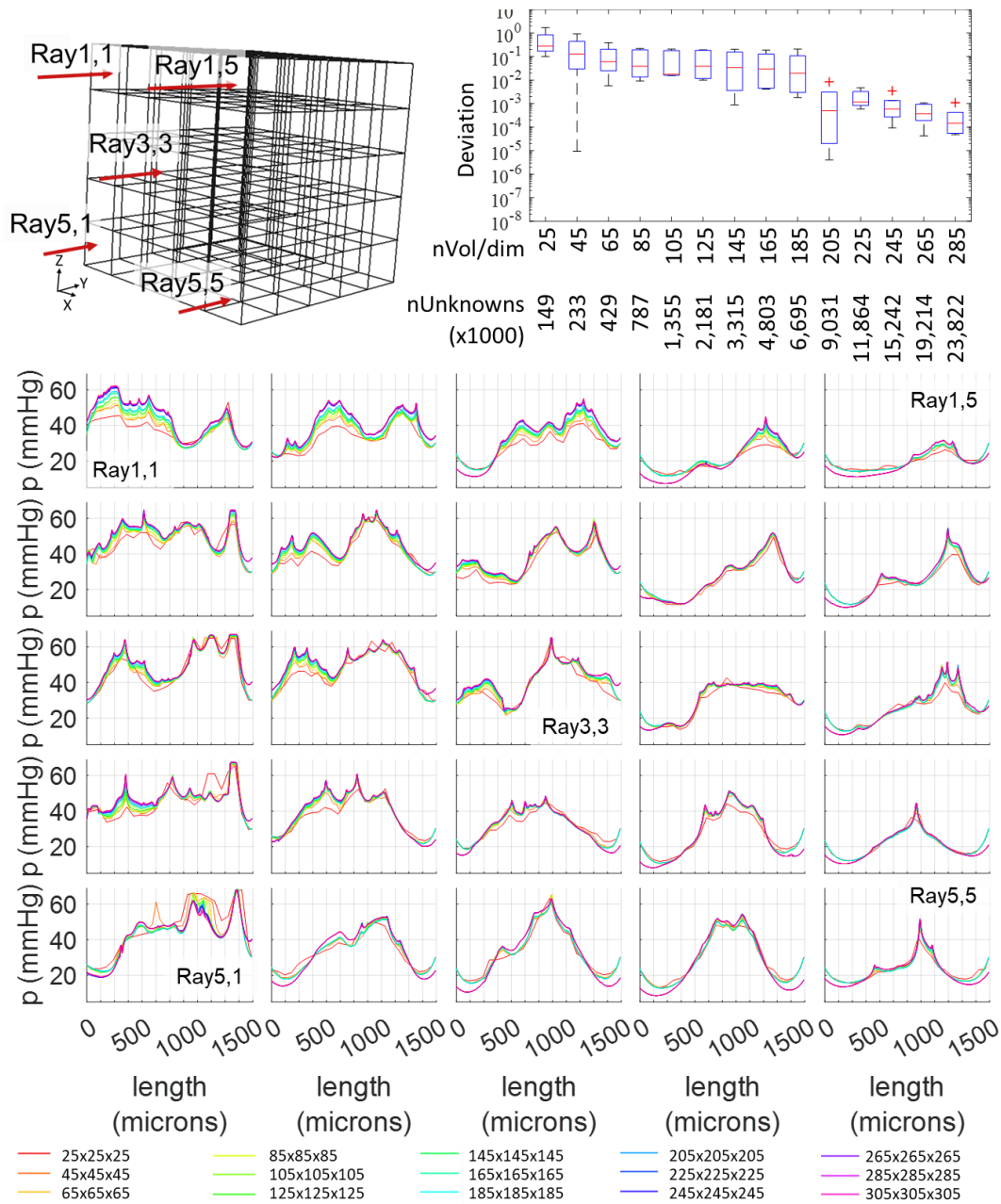


Figure 5.6. Detailed analysis of extravascular space in an oxygen simulation for a second empirical network.

A) The 3D image shows the dispersion of oxygen originating in the vascular inlets and dispersing throughout the tissue domain. B) The convergence of values for many rays at $y=718\mu\text{m}$ shows stable convergence and an acceptable tolerance at ~ 205 mesh elements per dimension. C) Raytraces through the domain exemplify the spatial complexity of oxygen tension. Note, the plateau in ray 6 corresponds to a vessel segment aligned the ray. All results show a deviation from the highest density mesh ($305 \times 305 \times 305$ volumes per dimension).

To further investigate the detailed microgradients of the extravascular space within the framework of a large microcirculatory network, networks were simulated for oxygen exchange. Detailed 3D oxygen maps and 2D planar distributions are shown in Figure 5.7 and Figure 5.8 for two representative oxygen distributions. Tall plateaus can be seen where large penetrating arterioles span more than a few mesh elements and flat basins (upside-down plateaus) can be seen for the venules with comparable width. Near each of these major vessels, the mesh in the vicinity reveals a concentration profile with a steep gradient.

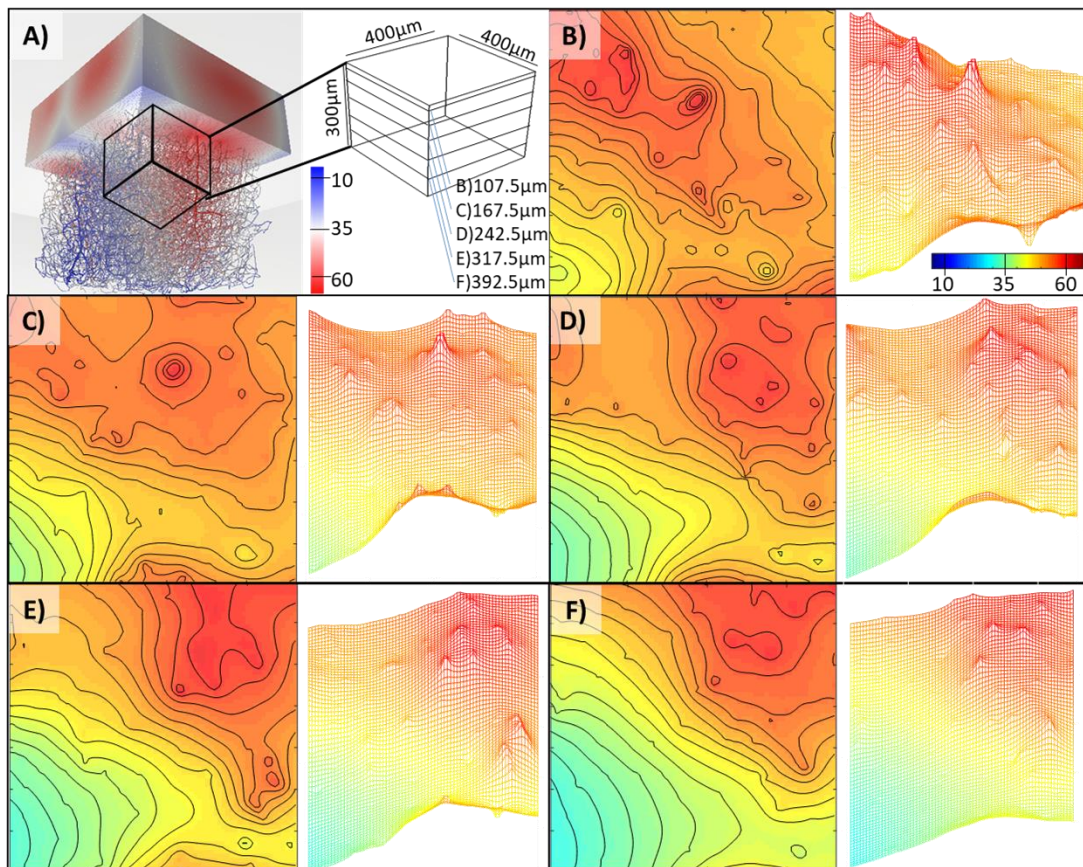


Figure 5.7. Oxygen predictions throughout the extravascular space.

A) Visualization of a portion of the extravascular space with a block cutout defined by 5 observation planes (107.5, 167.5, 242.5, 317.5, and 392.5μm below the cortical surface). B-F) The visualization of oxygen at different cortical layers shows peaks and steep microgradients surrounding main feeding and draining vessels.

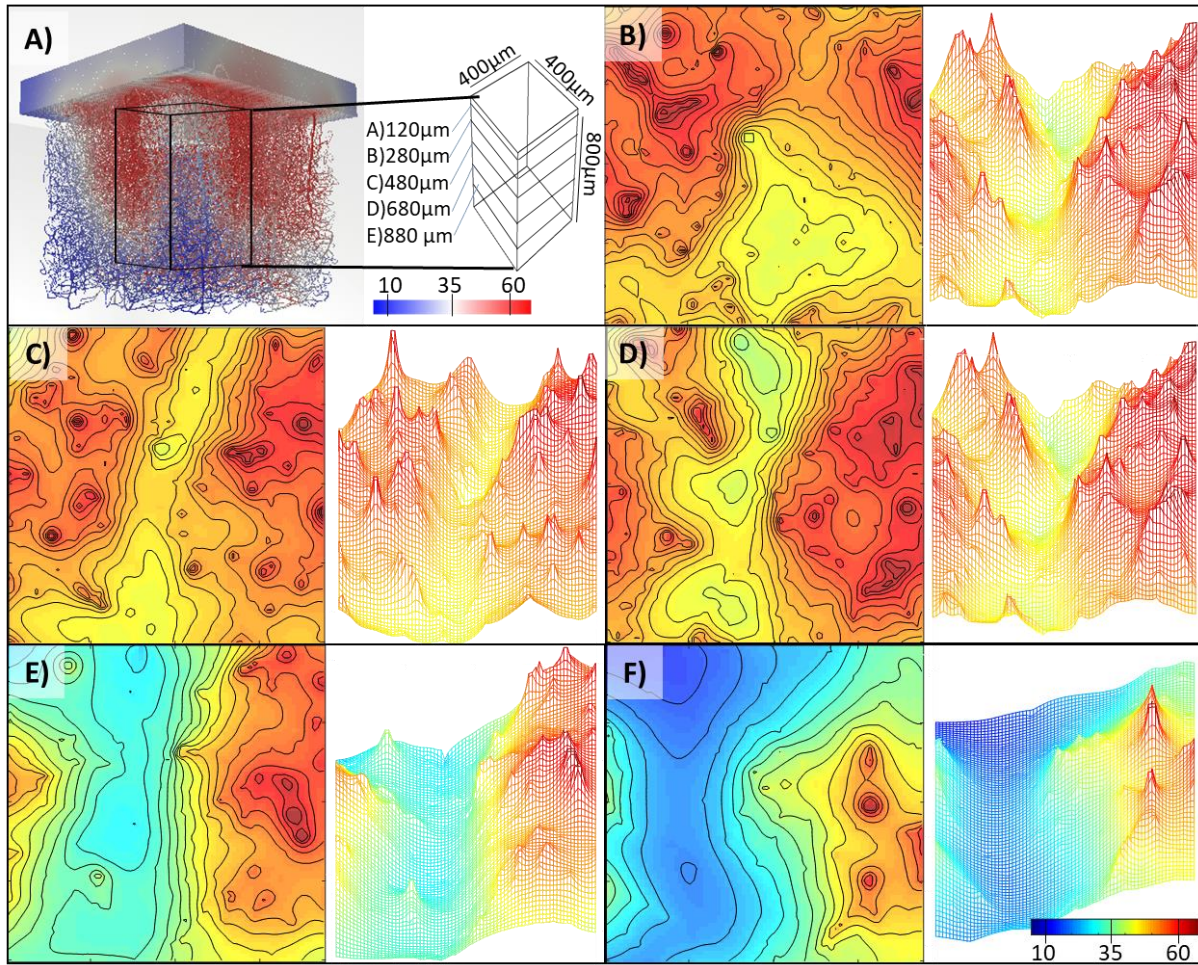


Figure 5.8. Oxygen predictions throughout the extravascular space in a second case.

A) Visualization of a portion of the extravascular space with a block cutout defined by 5 observation planes (120, 280, 480, 680, and 880 μm below the cortical surface). B-F) The visualization of oxygen at different cortical layers shows peaks and steep microgradients surrounding main feeding and draining vessels.

These distributions show excellent resolution of the microgradients surrounding larger vessels while maintaining stable convergence. The normalized infinity norm of the residual is on the order of 0.1×10^{-6} and the normalized mass transfer residual is on the order of 1×10^{-15} in all simulations.

5.4 Discussion

This methodology offers a novel and robust approach for stable convergence on very large meshes. This method has proven capable of simulating massive microcirculatory networks with stable convergence at an unprecedented mesh density. With recent advancements in microcirculatory network construction [27,170], this highly-scalable oxygen simulating paradigm offers a timely method for scaling these simulations to much larger networks. Moreover, the stable mesh convergence showed limitations only at the local memory bandwidth of local workstation personal computers (>95 million equations).

Applications to larger networks. The stable convergence of this methodology offers the far-reaching impact of making predictions of the neurovascular unit on larger vascular domains. This stability can be exemplified by simulating a topologically equivalent cerebroanatomical vascular network much larger than ever simulated before spanning $\sim 9\text{mm}^2$ of the cortical surface of a mouse. This model (seen in Figure 5.10 related to the aging brain) exemplifies the scalability of the proposed methodology to such large structures.

Applications to the aging brain. Another application for such a robust system is to attempt to recreate experimental observations of hypoxic micro-pockets in the brain. The aged brain suffers from a variety of vascular changes. These include; (i) a decrease vessel density (nSgm/mm^3) of $\sim 20\%$ [3,8,171–174], (ii) a decreased hematocrit by $\sim 30\%$ [8], and (iii) an increase in micro-occlusion occurrence $\sim 30\%$ [6]. These changes can be modeled by modifying the vascular structure (vessel reduction and occlusion) or boundary conditions (hematocrit) and investigate the resulting change in oxygen distribution that may lead to hypoxic micro pockets. This type of

simulation requires high resolution simulations capable of resolving such microgradients while maintaining stable solvability. An example of such resolution has been offered in Figure 5.9.

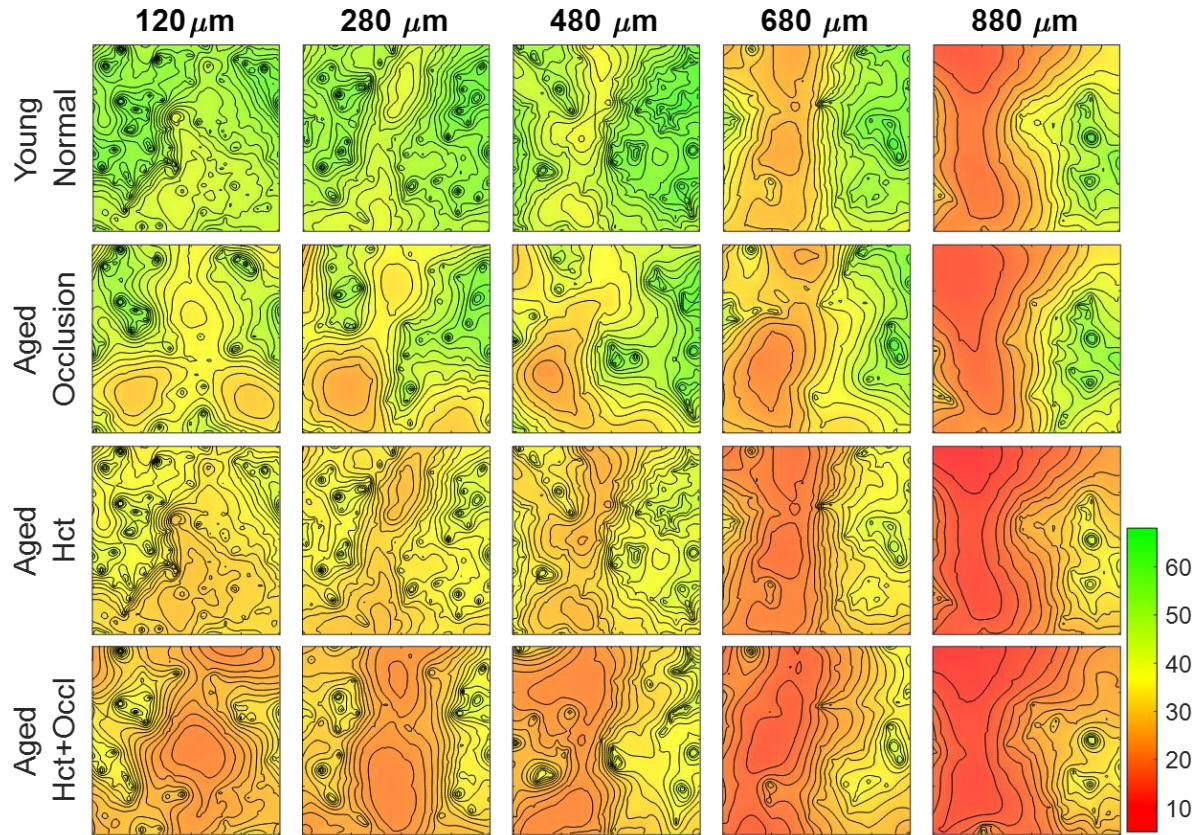


Figure 5.9. Examples of oxygen tension at different cortical depths in young and aged mouse. Top Row) normal young conditions, Second Row) aged brain with capillary micro-occlusions, Third Row) aged brain with reduced systemic hematocrit, and Bottom Row) aged brain with reduced hematocrit and microocclusions. The region containing tissue regions of low oxygen content greatly increase in the aged models with the largest hypoxic regions created in lower regions with both aging models.

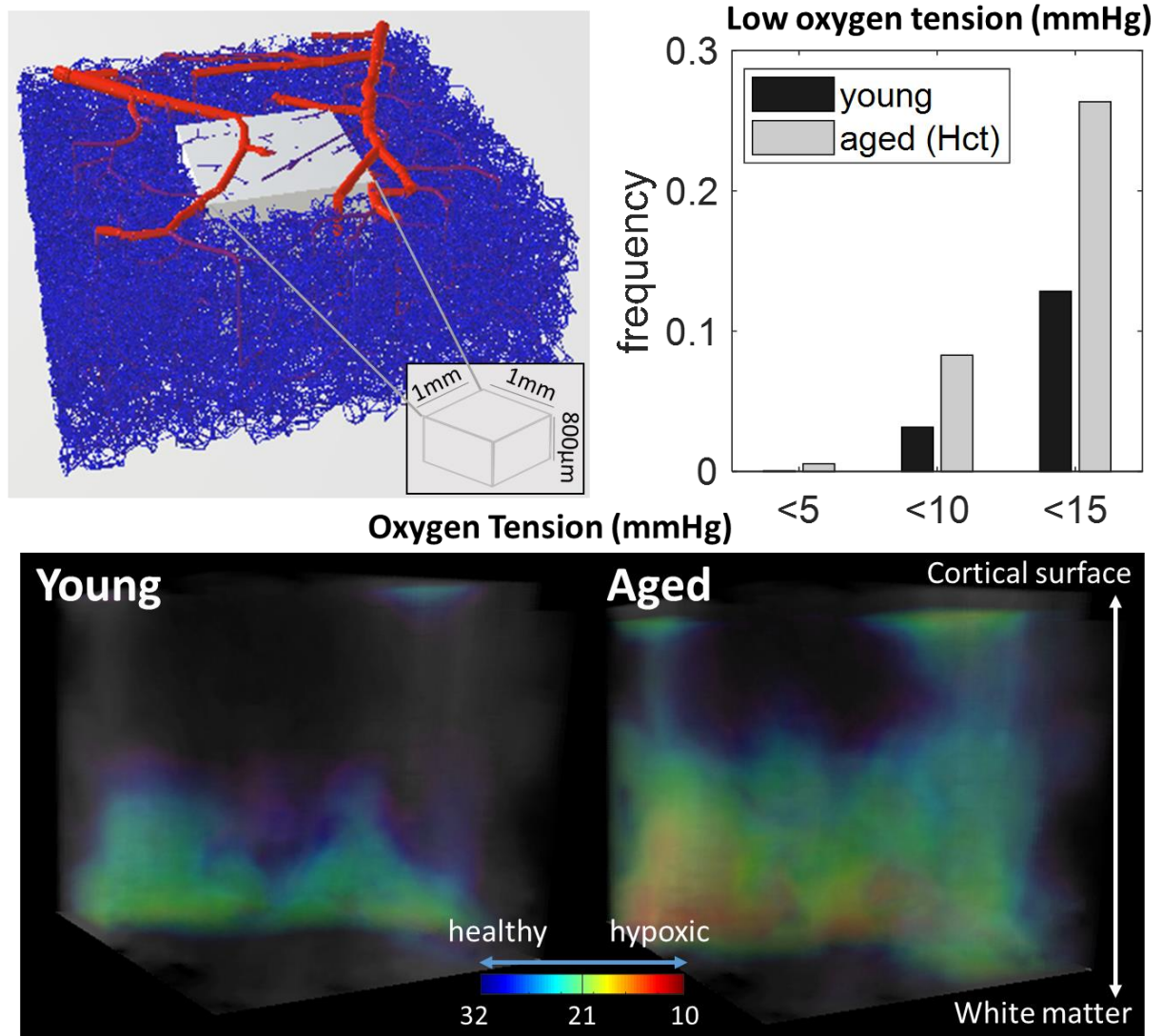


Figure 5.10. Comparison between oxygen tension in young and aged mouse. (Left) young mouse and (Right) aged mouse suffering from reduced hematocrit in a synthetic vascular network spanning $3 \times 3 \times 1 \text{ mm}^3$ of the mouse cortex. The oxygen levels are significantly reduced in the aged brain and hypoxic pockets (red) are created as shown by the volume occurrence rate of tissue that were <5 , <10 , and <15 mmHg of oxygen tension. Note, the mesh density was $455 \times 455 \times 455$ dimensions, correlating to over 95M equations.

These predictions help to compare the different models of aging on the cerebrovasculature. Such an application can be explored in future work, along other simulations, to investigate the mechanistic sources of age-related oxygen tension changes.

Another advantage of this method is that it uses a hexahedral cell-based logic. This implies a simulation could be generated from a voxelized set of data, such as a direct medical image. In other words, this simulation could be applied directly to image stacks without the need of first reconstructing a mesh.

Limitations. This method can be improved with the use of profile assumptions [165] using the centerline distance as the source, or by using partial volume approximation for removing the partial volumes inside the mass transfer elements. These methods would advance the implementation convergence.

5.5 Appendix A: vessel identification in 3D

If a new point is determined to be between the first and second point, axially, then the perpendicular distance between the point and the segment characterizes whether or not the point is within the cylinder (distance < radius). Due to the gaps between adjacent segments modeled as perfect cylinders, the ends of the cylinder should be endowed with a sphere. A pseudocode has been offered:

```
1.  FUNCTION Label3DMesh(mesh,nwk):PDb1Array
2.      setLength(result,mesh.nVolumes+1);
3.      FOR iFace = 1 TO nwk.nFaces DO
4.          getPointsForFace(iFace, p1Idx, p2Idx);
5.          dia = nwk.dia[iFace];
6.          getMinMaxIJKFromFace(p1,p2,mesh, dia/2, minI,minJ,minK,maxI,maxJ,maxK);
7.          FOR i = minI TO maxI DO
8.              FOR j = minJ TO maxJ DO
9.                  FOR k = minK TO maxK DO
10.                     cellIdx = mesh.getGlobalIdxFromIJK(i,j,k);
11.                     cellCenter = mesh.getCellCenter(cellIdx);
12.                     IF isCellInCylinder(p1,p2,cellCenter,dia) THEN result[cellIdx] = interior;
13.                 ENDFOR
14.             ENDFOR
15.         ENDFOR

1.  FUNCTION getMinMaxIJKFromFace(p1,p2,mesh,r,minI,minJ,minK,maxI,maxJ,maxK)
2.      setLength(iA,4); setLength(jA,4); setLength(kA,4);
3.      Mesh.getIJKOfSurroundingBox(plus(r, p1), iA[0]. jA[0], kA[0]);
4.      Mesh.getIJKOfSurroundingBox(plus(r, p2), iA[1]. jA[1], kA[1]);
5.      Mesh.getIJKOfSurroundingBox(plus(-r, p1), iA[2]. jA[2], kA[2]);
6.      Mesh.getIJKOfSurroundingBox(plus(-r, p2), iA[3]. jA[3], kA[3]);
7.      minI = getMin(iA);  minJ = getMin(jA);  minK = getMin(kA);
8.      maxI = getMax(iA);  maxJ = getMax(jA);  maxK = getMax(kA);
```

Identifying whether a point lies in a cylinder or not can be performed with vector operations. This efficient algorithm simply calculates the centerline of the cylinder as a vector, and uses sine and cosine operations to evaluate the distance (see Figure 5.11).

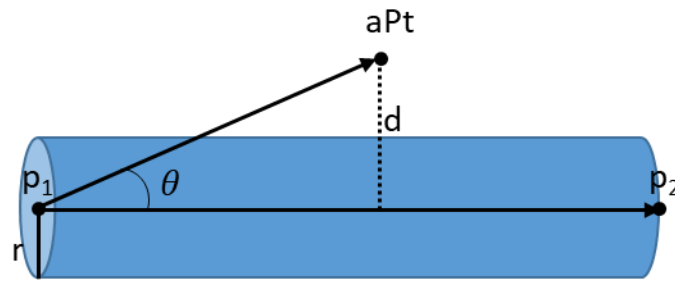


Figure 5.11. Schematic diagram of a cylinder. The cylinder is defined by 2 points (p_1 and p_2) and a radius (r). A new point (aPt) will be evaluated for whether or not it lies within the cylinder.

The minimum distance between a new point (cell center) and a segment centerline is given by the following pseudocode:

```

1.  FUNCTION isCellInCylinder(p1,p2,cellCenter,dia):boolean;
2.      v1 = getAsVector(p1,p2);  v2 = getAsVector(p1, cellCenter);
3.      c = dot(v1,v2)/(norm(v1)*norm(v2));    H = norm(v2);
4.      d = H*power(1-c*c, 0.5);
5.      IF (c == 1) AND (norm(v1) + r > H) THEN          result=true;
6.      ELSEIF (abs(c-1) > 0) AND (H*c < norm(v1)) AND (d < r) THEN  result = true;
7.      ELSEIF (abs(c-1) > 0) AND (d < r) THEN              result = true;
8.      ELSEIF  (d < r) THEN                                result = true;
9.      ELSE  result = false;

```

Where line 5 evaluates a point that is parallel to the centerline and line 6 checks the perpendicular distance to cylinder centerline. Line 7 and 8 evaluate whether the point lies in the sphere at the first point of the face or the last point of the face, respectively.

6 Discussion

Mathematical modelling is necessary to piece together experimental measurements (made across different imaging modalities, different specimen, and at different times). These models, however, are frequently simplified to small sub-sections or simplified geometric structures to enhance implementation. This limits all insights, as the simplifications often create artifacts in the simulation domain that heavily impact predictions. In other words, the simplifications skew the simulation results. In order to create mathematical models capable of investigating mechanistic relationships in the cerebral anatomy, such assumptions must be avoided.

To date, the largest simulations of anatomically consistent microvascular structures were limited to $\sim 1\text{mm}^3$ and oxygen distribution in and around these vascular structures were not readily expandable to the whole-brain scale. In this thesis, a methodology was proposed to synthesize vascular structures with topology matching microvascular reconstructions. A methodology was also proposed for automated matching of the topology between synthesized networks and empirical counterparts by matching tortuosity and diameter spectra.

The mathematical implementation was expanded from recent models to converge the KPSM model of nonlinear biphasic blood flow on a massive hemisphere and MCA network. This (along with predictions on empirical networks and synthetic clones) led to the discovery of a depth-dependent gradient in RBC density which was later validated by experimentation by an independent lab.

A novel paradigm for resolving the complex 3D structure of the microvascular network offers stable simulation of the neurovascular coupling at much higher densities than simulated previously. This simulation not only offers investigation of the steep micro-gradients of oxygen

surrounding large feeding vessels, but also offers stable scalability to larger structures than ever simulated before.

This work marks a major advancement towards simulating the neurovascular unit at the scale of the whole-brain. This kind of simulation will eliminate artificial boundary effects which will be replaced by real boundary effects from the real-life boundaries.

7 Appendices

7.1 Appendix A: Cerebrovascular synthesis implementation

While the main algorithm of cerebrovascular synthesis is offered in Section 2-3, the implementation is not straightforward. This section offers further clarification for implementation of the vascular synthesis and topological matching algorithms.

7.1.1 Segment addition

The most core algorithm used in vascular synthesis is the segment addition procedure as each step of growth requires the addition of new segments. The generalized workflow for segment addition is given in Figure 7.1. Note, if none of the trees satisfies all constraints, the segment addition procedure fails, a new sample is generated, and the procedure runs again. This cycle repeats until a suitable candidate is identified.

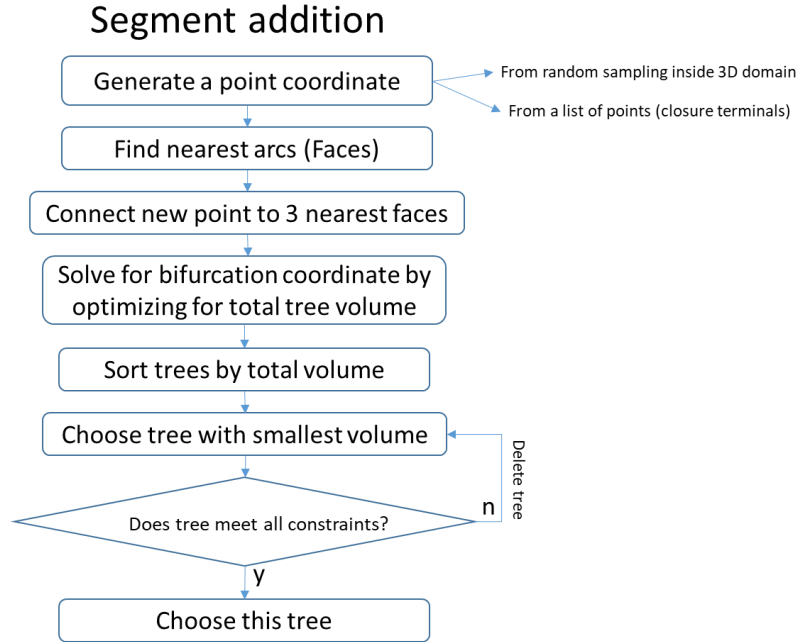


Figure 7.1. Workflow diagram for growing a single segment.

Note, the segment addition assumes only 3 near-segments for each addition. This value can be modified to include more options if the user finds it necessary.

The general implementation of this code can be seen below:

Table 7.1. Pseudocode for the segment addition with volume optimization

1.	Algorithm 1- SegmentAddition with Volume Optimization
2.	WHILE $i \leq n\text{Segments}$ do
3.	$xP := \text{SampleGenerator.getNextSample}(\text{tree})$
4.	$\text{closeSegmentsList} := \text{findClosestSegments}(n\text{MaxSegments}, xP)$
5.	FOR all s in closeSegmentsList do begin
6.	$\text{candidateTree} := \text{getShallowCopy}(\text{tree})$
7.	$x0 := \text{segmentsCenter}(s)$
8.	$\text{candidateTree.bifC} := \text{findOptimalBifuractionPointCoord}(x0, \text{candidateTree})$
9.	$\text{candidateTree.recAdjustDiameterAndComputeVolume}(\text{rootRadius})$
10.	IF $\text{passesConstraints}(\text{candidateTree})$ THEN $\text{SolutionList.addSorted}(\text{candidateTree})$
11.	ENDFOR
12.	IF $\text{SolutionList} \neq \text{empty}$ then continue // sample point is invalid
13.	$i++$ // tree has grown one segment
14.	$\text{tree} := \text{SolutionList}[0]$ // first element is smallest feasible volume tree
15.	ENDWHILE

7.1.2 Validating the optimal bifurcation point

The derivation of the optimal bifurcation point with minimum volume is offered in Section 2.2.1 however this section validates the optimal bifurcation point with case studies.

One way to validate the solution to the optimization of bifurcation location is to exhaustively enumerate the region around the bifurcation and show that the volume is the lowest point. This was performed with the results presented in 2 case studies. Note, the method for calculating the point on original segment that makes a perpendicular connection is given in Section 5.5.

Case study 1. The original network was balanced and the results reported. For comparison, the same values were reported post-optimization (Table 7.2). The code for this case study is offered in Section 7.7.1.1.

Table 7.2. Comparison for topological values of a bifurcation before and after optimization

	Root D	Total V	Total α	Inlet Flow	Left flow	Right flow
Original Tree	9.2137	336.946	1e-4	10,000.00	5,000.0	5,000.0
Optimized Tree	9.0149	302.714	1e-4	10,000.00	5,000.0	5,000.0

After the bifurcation point was identified by exhaustive enumeration, the root diameter was reassigned to a fixed value and the topological statistics were revisited. The results in Table 7.3 reflect a change in all parameters.

Table 7.3. Topological values of a bifurcation after two different diameter assignments

	Root D	Total Volume	Total α	Inlet Flow	Left branch flow	Right branch flow
D_{opt}	9.0129	302.714	1e-4	10,000.00	5,000.0	5,000.0
D_{set}	10	372.651	6.599e-5	15,154.45	7,577.2	7,577.2

Source data:

ptCoordMx = [-1.0001 1.0001; 0.00001 0.49999; 3.0001 -1.0001; 2.0001 2.0001];
faceMx = [1 2; 2 3; 2 4];

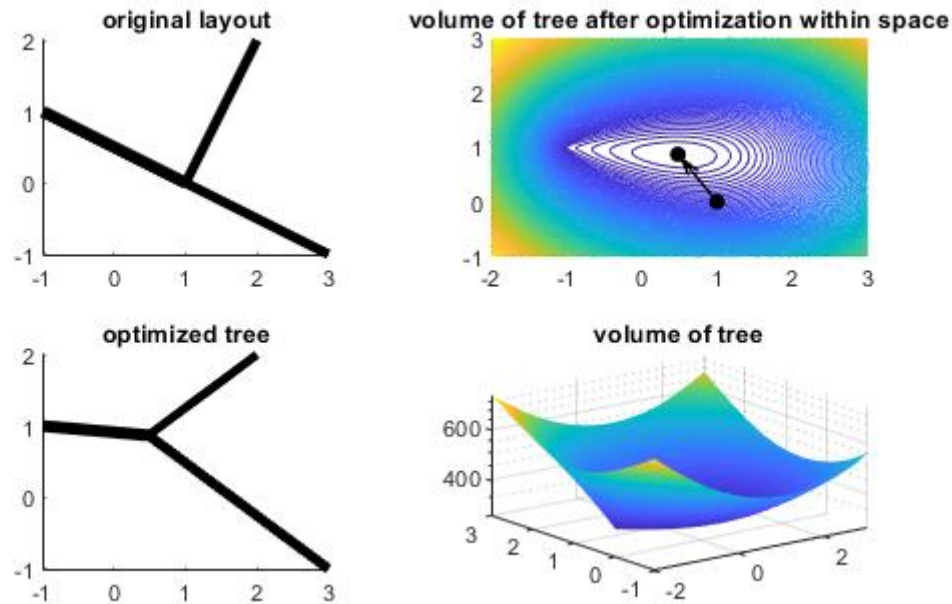


Figure 7.2. Optimization space for bifurcation location.

(Top Left) when the root diameter is variable. The solution space for bifurcation location ($f_{Bif}(x,y)$) is shown in (Top Right) contours and (Bottom Left) surface plots. The optimal location ($f_{Bif}(x^*,y^*)$) is chosen for the bifurcation point and diameters are assigned (Bottom Right).

Case study 2. The second case study follows the same procedure but with a different geometry.

Table 7.4. Comparison for topological values of a bifurcation before and after optimization

	Root D	Total V	Total α	Inlet Flow	Left flow	Right flow
Original Tree	9.2616	348.975	1e-4	10,000.00	5,000.0	5,000.0
Optimized Tree	9.0905	325.032	1e-4	10,000.00	5,000.0	5,000.0

Table 7.5. Topological values of a bifurcation after two different diameter assignments

	Root D	Total Volume	Total α	Inlet Flow	Left branch flow	Right branch flow
D_{opt}	9.088	325.065	0.1e-3	10,000.00	5,000.0	5,000.0
D_{set}	10	393.598	6.82e-5	14,661.03	7,330.5	73,30.5

Source data:

ptCoordMx = [-1.0001 3.0001; 0.00001 0.49999; 2.0001 -1.0001; 2.0001 1.0001]; %modify 2nd point
faceMx = [1 2; 2 3; 2 4];

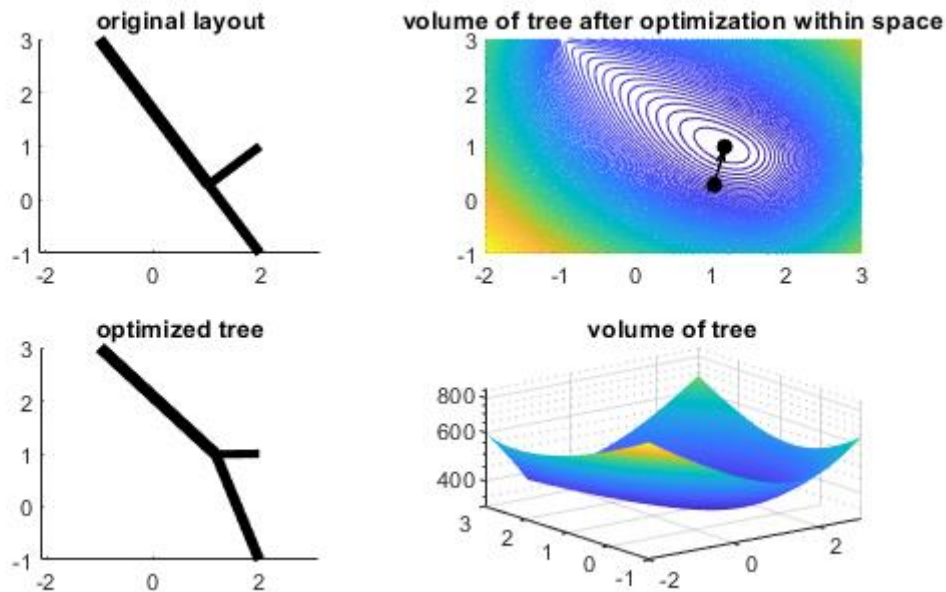


Figure 7.3. Optimization space for bifurcation location.

(Top Left) when the root diameter is variable for a second tree. The solution space for bifurcation location ($fBif(x,y)$) is shown in (top right) contours and (bottom left) surface plots. The optimal location ($fBif(x^*,y^*)$) is chosen for the bifurcation point and diameters are assigned (bottom right).

7.1.3 Assigning diameter

Once a tree is completed it must be assigned a root diameter (either with constant tree resistance as in the original implementation or from empirically-derived root radii, whichever the modeler has more confidence in) which can recursively be used to compute downstream diameters for all

segments in the tree. This can be accomplished by using the same values for β computed in Section 7.1.2 and successively assigning daughter diameters of each downstream bifurcation.

7.1.4 Staged growth algorithm implementation

This section will explain more details on the implementation of staged growth for a cerebrocirculatory structure. The overview of the anatomical growth algorithm is given in Figure 7.4.

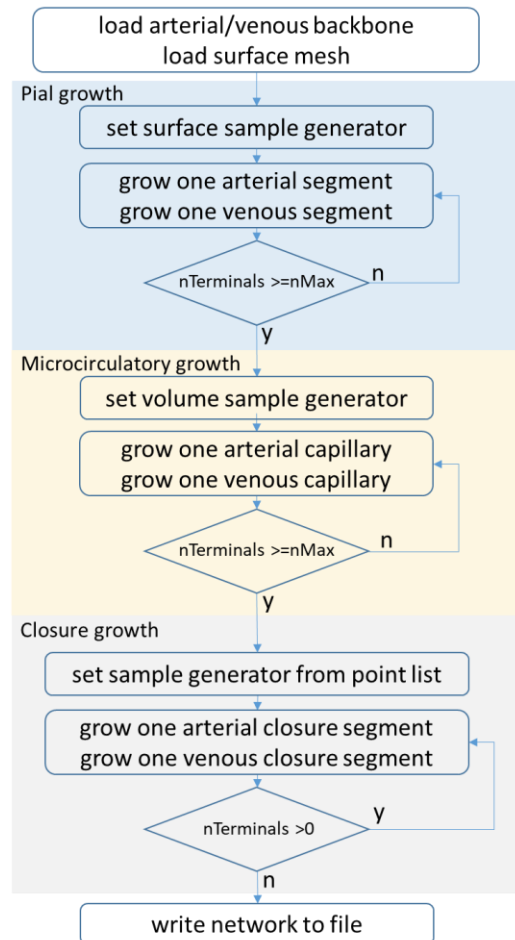


Figure 7.4. Workflow diagram of the anatomical growth algorithm.

A pseudocode is also offered:

Table 7.6. Pseudocode for staged growth

1.	Algorithm – Vascular Strategy for Cortical Slab
2.	maxAdditions := 20000
3.	artTree := ooPropertyTree.create(maxAdditions)
4.	venTree := ooPropertyTree.create(maxAdditions)
5.	makePialNetwork (surfaceSampleGenerator)
6.	makePenetratingVessels; // adds penetrating Arteries at random length
7.	makeMicroVessels(CubeSampleGenerator)
8.	END
9.	Algorithm PialNetwork - oKFCapillaryApp.makePialNetwork;
10.	sampleGenerator := PlanarSampler.create(SetSurfaceSlab (X0, X1))
11.	// First arterial segment only
12.	pts := growPialNetworkAutomatic(artTree, artB1,artB2, [artS1, artS2])
13.	pts := growPialNetworkAutomatic(venTree,venB1,venB2, [venS1, venS2])
14.	// 2D arterial tree on cortex
15.	FOR ii:= 1 to nPialTerminals do begin
16.	setPialConstraintsForArteries;
17.	pts := growPialNetworkAutomatic(artTree,1,artB1,artB2)
18.	setPialConstraintsForVeins
19.	pts2 := growPialNetworkAutomatic(venTree,1,venB1,venB2)
20.	ENDFOR

7.1.4.1 Pial Surface

The pial surface growth requires a surface mesh to identify locations of new points to add to the network. The mesh is first grouped into territories manually. The user selects a group name (integer value used as ID for the group, denoted GroupID) and the mesh faces corresponding to that group are put into a list. This list is fed into a sampleGenerator to generate a list of point coordinate samples on the surface mesh (see Section 7.2.3 for more information on triangle sample generators). The bifurcation point may not lie on the surface mesh after optimizing its location, so a post-optimization step is employed that projects the point to the nearest surface triangle (see Section 7.1.4.1.1 for projecting a point to a surface). The workflow for this can be seen in Figure 7.5.

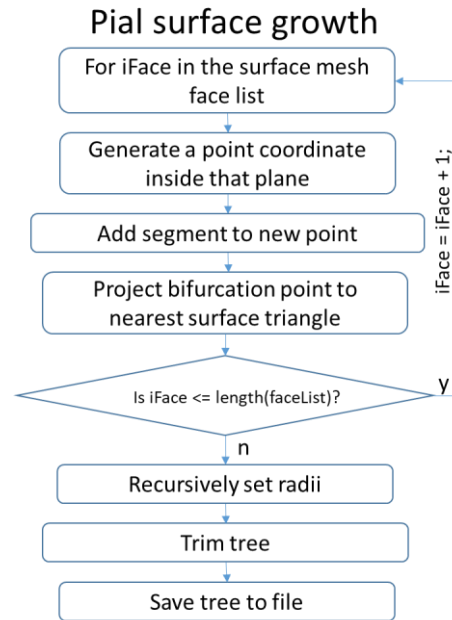


Figure 7.5. Workflow diagram for pial surface growth.

Note that the surface growth hits every surface element in the group before it returns for a second addition. This allows the tree to grow evenly distributed without sample bias. A time-lapse of the pial growth and the finished pial surface can be seen in Figure 7.6. Finished products of pial growth on a surface mesh are offered in Figure 7.7.

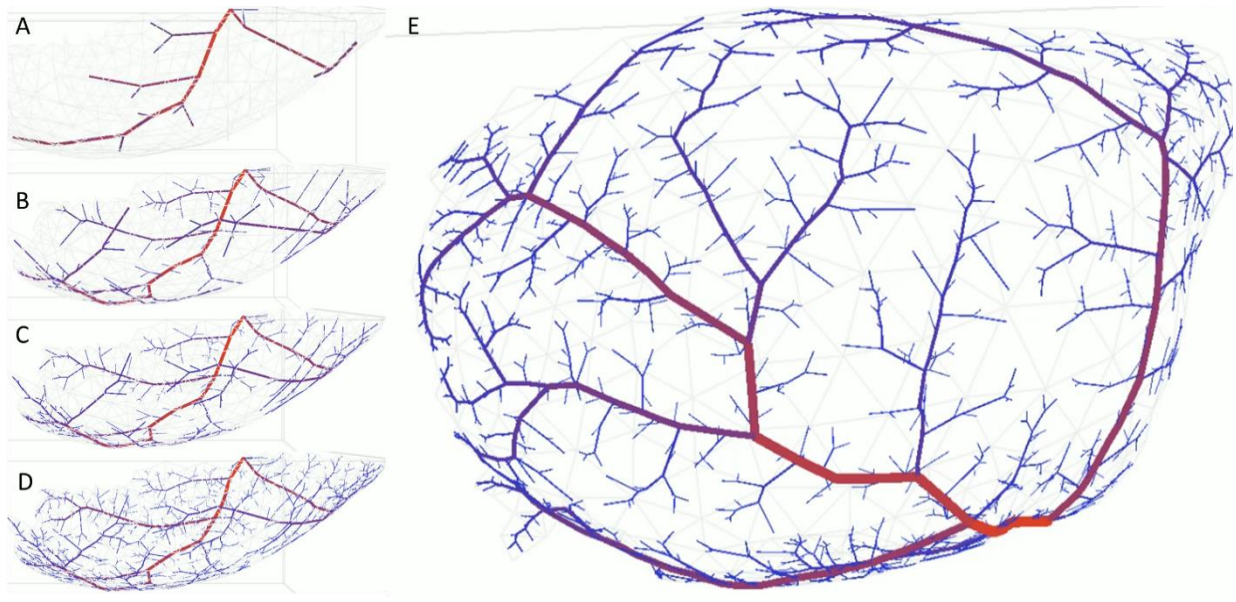


Figure 7.6. Pial growth algorithm at different stages of growth. A-D) Different stages along the growth where the pial network is increasing in size with A as the most sparse and D is the most dense network. E) Rotated visual of the final network.

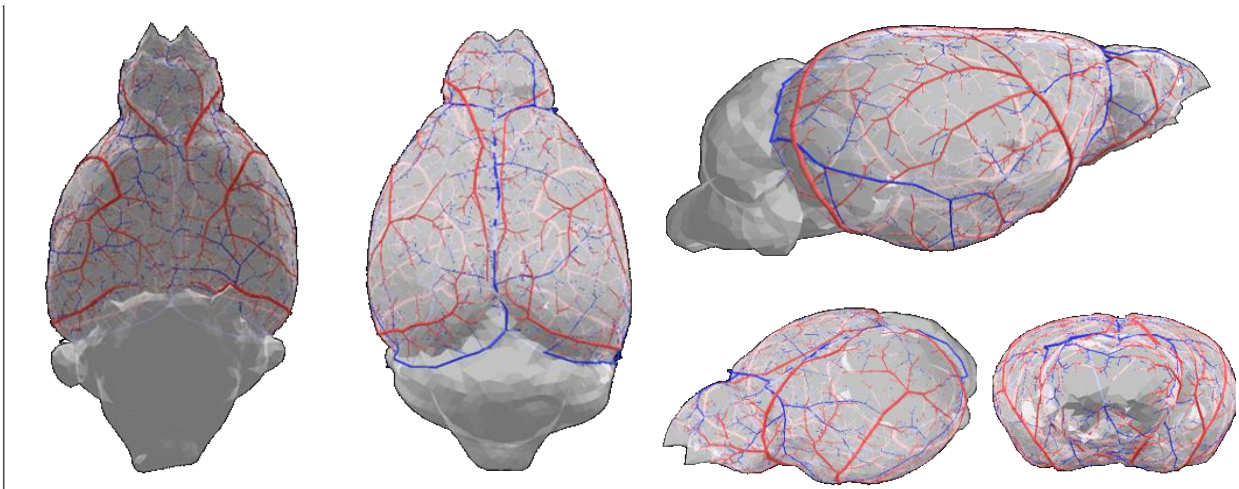


Figure 7.7. Unity Images of Sled Mouse. Images of an arterial tree, venous tree, and the sled mouse surface mesh loaded in Unity.

7.1.4.1.1 Projecting bifurcation point to surface

The cortex of a mouse brain is generally a convex shape. This means that any line between two points on the surface give cuts through the surface. When the bifurcation point is identified, it is

coplanar with the three points on the mesh surface, meaning the bifurcation point also lies inside this convex shape. In order to rectify this, and force all pial vessels to follow the pial surface, all bifurcations points are projected to the pial surface after optimization as illustrated in Figure 7.8.

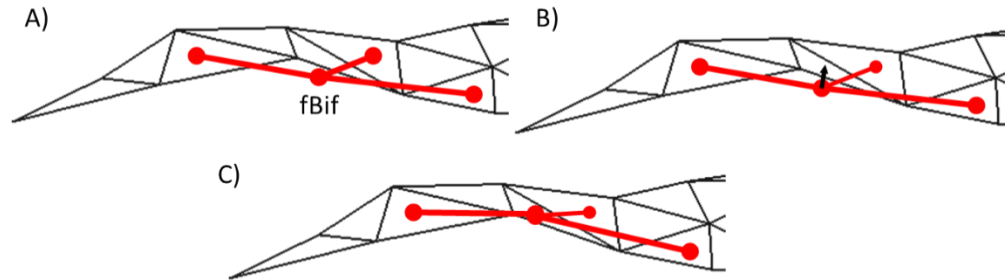


Figure 7.8. Explanatory diagram of the projection of a point to the surface mesh. Notice the in-plane bifurcation (fBif) is placed inside the convex structure. The final, projected point will lie inside a surface triangle maintaining the surface adherence of the pial vessel structure.

In order to ensure all points lie on the surface, the bifurcation point needs to be relocated (projected) to the surface. This procedure can be executed as depicted in Figure 7.9 and mathematically expressed in Equations (7.1) - (7.2).

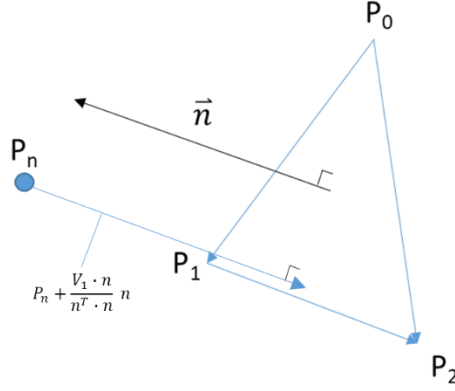


Figure 7.9. The graphical representation of a triangle with reference to a new point (P_n). These symbols are consistent with the mathematical description of the projection procedure given below. n is the normal vector to the triangle and P_n is the new point to be projected to the surface.

$$V_1 = \langle P_1 - P_n \rangle^T \quad (7.1)$$

$$p_p = P_n + \frac{V_1 \cdot n}{n^T \cdot n} n \quad (7.2)$$

Where P_p is the projected point coordinate (the new point inside the triangle). The appropriate surface mesh element (triangle) used for projection is identified by closest distance to the point P_n .

7.1.4.2 Penetrating arterioles and ascending venules

The penetrating arterioles and ascending venules, as opposed to the surface growth, penetrate beneath the surface with a relevant depth parameter. The user defines the maximum depth (distance from the surface) and possibly a minimum depth that the penetrating vessel can reach inside the cortex.

The first step to identify the depth-dependent random sample is to identifies a random depth between the minimum depth and the maximum depth. This depth is then used to scale the normal

vector, Equation (7.3), and is added to the triangle sample point (in the case of penetrating vessels, this sample point is the pial terminal node as in Equation (7.4)). Of course, this requires identifying and maintaining a list of the available pial terminals. The Kleinfeld datasets have prescribed depth to use, but in the reconstructed mouse brain growth, a default value of 1mm is chosen. The workflow for this can be seen in Figure 7.10.

$$V = \frac{d}{|n|} \cdot n \quad (7.3)$$

$$p_n = p_t + V \quad (7.4)$$

Where p_t is the point coordinate of the pial network terminal node, n is the normal to the mesh surface triangle, d is the depth generated from the random number generator, V is a new vector, and p_n is the new terminal point. The new terminal point and the old terminal point are connected by a single arc to make the penetrating arteriole or ascending venule. One way to improve the horizontal bias of attachments to the penetrators (make more horizontal segments) is to generate the penetrating vessel with many divisions (10 or 20). The workflow and finished product are offered in Figure 7.10. A pseudocode is also offered for adding penetrators and generating sample points:

Table 7.7. Pseudocode for growing penetrating vessels

```

1. FUNCTION growPenetrators(previousCaseFileName:string;   groupId:integer;
   rootRadius, depth:double; aMesh:ooMesh);
2. aTree := CreateTreeFromBackbone(fileName);
3. facIdxList := getIdxListFromGroup(aMesh,groupId);
4. aSmplGen := sampleGenerator.Create(aMesh,facIdxList,depth);
5. setConstraints(nil);
6. terminals := aTree.findOutletTerminalPorts();
7. GrowNIterationsInSection(nRemainderFaces, facIdxList);
8. GrowNIterationsInSection(nRemainderFaces, facIdxList);
9. FOR i = low(terminals) TO high(terminals) DO
10.   aPoint = generateSamplePoint(aTree,terminals[i]);
11.   aTree.addPolyLineAtKnownPoint(aTree,terminals[i], aPoint);
12. ENFOR
13. END

```

Table 7.8. Pseudocode for generating a sample point inside a triangular prism

```

1. FUNCTION generateSamplePoint(aPointIdx:integer):PDbIArray;
2. aPtCoord := aTree.getPointCoord(aPointIdx);
3. n := getFaceNormal(aMesh,nearestSurfaceFacIdx);
4. n := scale(depth/norm(n), n); //need to scale the norm to the depth we want
5. result := plus(aPtCoord,n); //new point coordinate
6. END

```

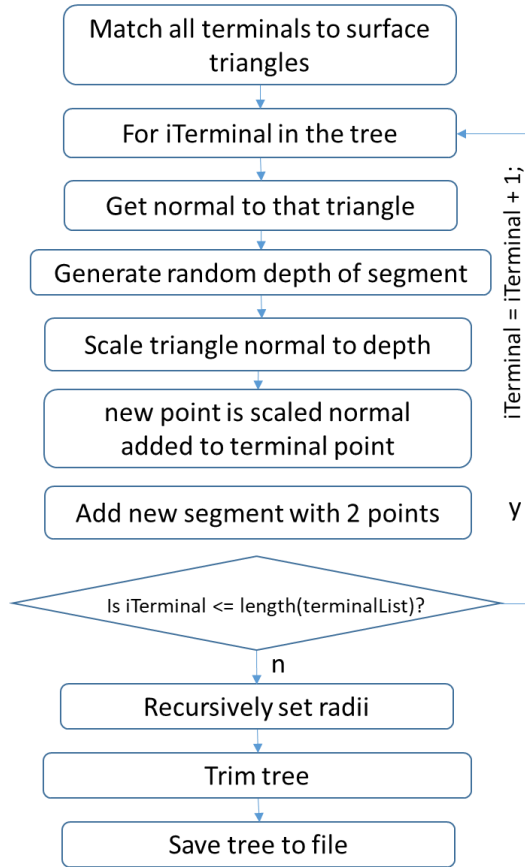
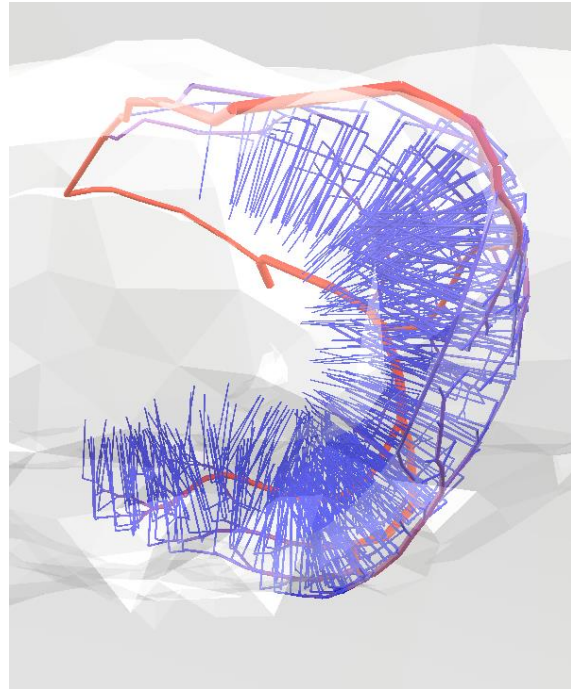
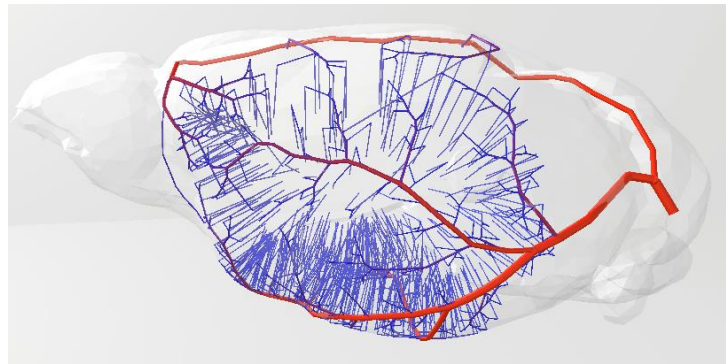
A)**B)****C)**

Figure 7.10. (A) Workflow diagram for growing penetrating vessels and (B-C) example views of ascending veins in an MCA territory.

7.1.4.3 Capillaries

The capillary bed is a highly dense structure. This structure has two unique criteria that are not present in the previous stages of growth; (i) it is very dense leading to many short segments and

(ii) the optimization of the tree for bifurcation location does not substantially impact the topology. These qualities allow the capillary growth to work with more liberal constraints such as replacing the objective function with a minimum distance heuristic with constraints such as minimum/maximum segment length. One important constraint is the *newVesselMinimumLength* which requires a different length (longer) than the existing vessel lengths. This new constraint allows generation of many segments without shortening the entire vascular structure spectra. Without needing to optimize the bifurcation location, all sort routines and recursive computing routines are no longer necessary, tremendously reducing growth time. The workflow diagram for capillary growth is expressed in Figure 7.11.

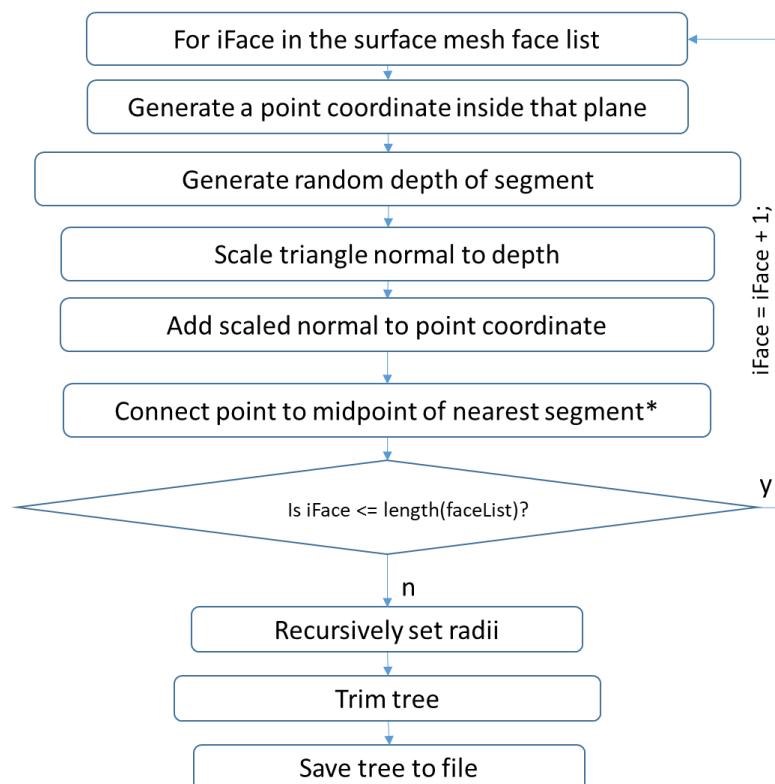


Figure 7.11. Workflow diagram for capillary growth.

*denotes that only certain group ID's are allowed for connection, giving more control over bifurcating depth.

7.1.4.4 Microcirculatory closure

Closure segments are a key advancement of the anatomical growth developed by our lab. As the growth of a new segment uses a randomly generated sample, it can simply use a sample from a list of point coordinates. Such a sample point coordinate list is created using the terminal list of the opposite tree after all capillaries have finished growing. The growth then continues to add segments to the tree (except from the new sample generator instead of a random sample generator) until all terminals of each tree have been attached to the other tree and vice versa. An example of one cycle of attaching terminals is offered in Figure 7.12. The steps of the closure algorithm are offered below:

1. Grow a new venous branch to every arterial terminal
2. Grow a new arterial branch to every venous terminal
3. Rebalance trees
4. Merge trees
5. Remove duplicate points in merged network
6. Add boundary conditions
7. Save to file

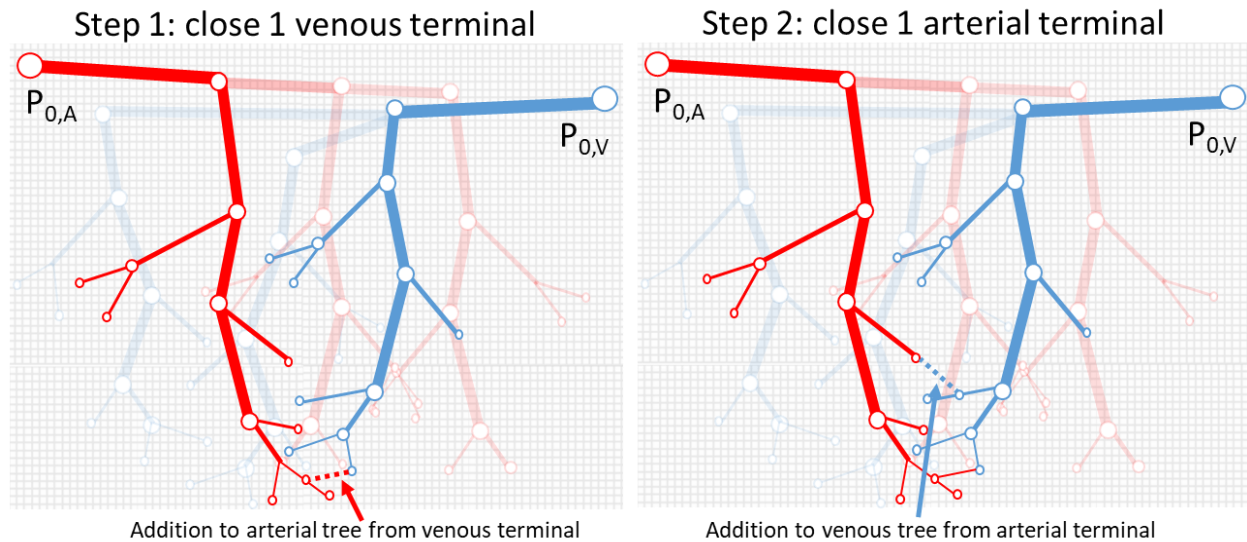


Figure 7.12. Demonstration of one pass of the closure algorithm.

Left) The algorithm first eliminates a single venous terminal by using it as a new terminal to be attached to the arterial tree. Right) The second step uses an arterial terminal as the newest terminal to be connected to the venous tree.

When the trees become excessively large ($>200,000$ points), the merging of the steps and removal of duplicating points becomes time prohibitive. In order to circumvent this problem, all trees are attached first and the closure is performed on the connected network using anatomical group information to identify points and relevant segments. Again, constraints have not been employed and diameter is merely assigned during each step as seen in Figure 7.13. Some constraints can be implemented here, but all constraints must be flexible, as no sample can be disregarded if it fails all constraints. A hierarchical “loosening” of the constraints may be necessary for adding more weight to some constraints than others. An image of the trees prior to closure and after closure segments have been added is offered in Figure 7.14.

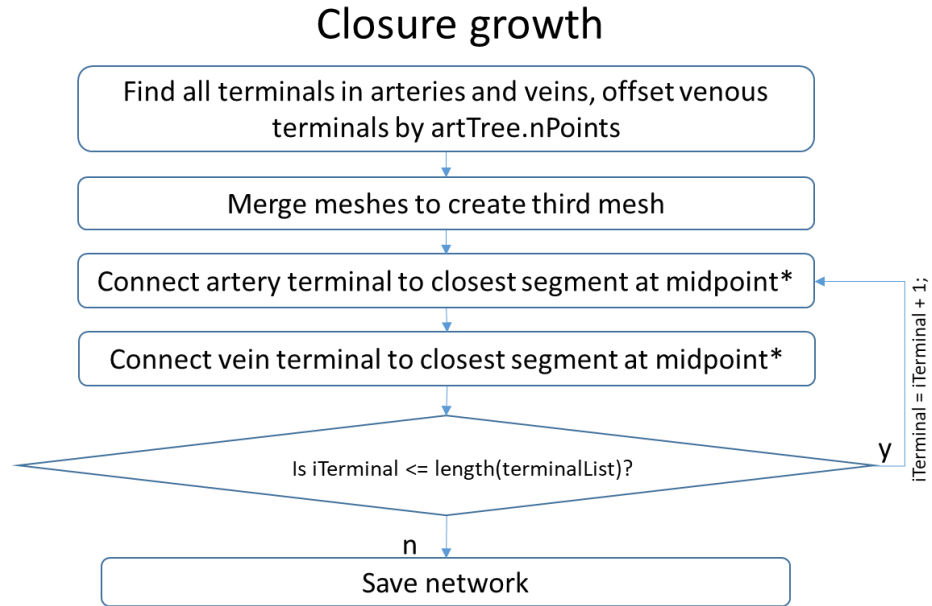


Figure 7.13. Workflow diagram for closure growth.

*denotes that only certain group ID's are allowed to be used for closest segment.

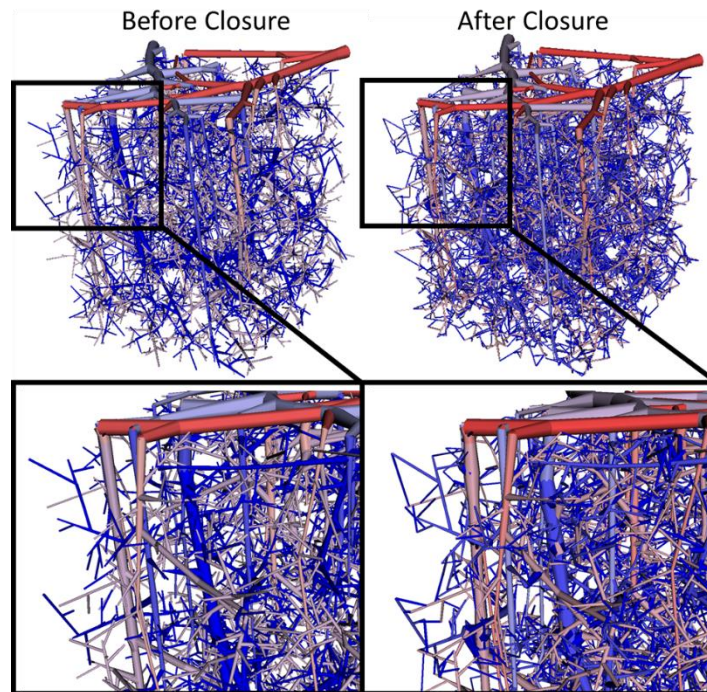


Figure 7.14. Anatomical growth of a Kleinfield-like microcirculatory network.

(Left) Before and (Right) after closure segments have been added. On the zoomed in panels (lower panels), the closure is more apparent. Note, blue indicates veins, arteries in red. Closure segments are also colored in dark blue.

After both trees have grown new segments to every terminal of the opposite tree, the balanced trees are merged together. This simply adds the points and faces of the venous tree to the end of the point coordinate list and face list of the arterial tree, respectively. This, of course, duplicates the points used for closure (they already existed on the opposite tree and were added as new terminals to the new tree). A simple *duplicatePointRemoval* algorithm can remove these and reattach the two connected faces to the same point. Note, an efficient version of this method has been implemented by the author which is necessary for any structure larger than a Kleinfeld-like structure (such as MCA or hemisphere).

7.1.4.5 Hemisphere implementation notes

In order to generate the two networks (arteries and veins), a backbone for each network was provided accompanying each of the two meshes (a reconstruction from images provided by the Sled group and the open-source Allan brain institute images). The files necessary to instantiate growth are a triangular pial surface mesh (with corresponding anatomical grouping), an arterial backbone, and a venous backbone.

In order to grow an entire mouse territory, the algorithm loads the backbone, calculates the pial density based on how many penetrating arterioles or ascending venules are needed (hardcoded to 13 terminals/mm² as defined by previous work [26]). Once the pial surface is complete, the interface visualizes this network so the user can review it and ensure it is physiologically meaningful. This avoids growing microcirculatory structures on a pial network that *cuts* through the cortex. Upon closing the visualization window, the user is prompted to accept or reject the network. If the user rejects the network, a new one is generated and the process is repeated until a

successful pial surface has been created. If the user accepts the pial surface, the growth can continue. The next step involves loading the pial surface and growing penetrating arterioles from every terminal in the binary tree (excluding the inlet) and the network is saved again. The pials + penetrating arterioles structure is then loaded and the capillaries are grown until the desired density is reached. Once this has been completed for both arterial and venous networks, the two networks are loaded and the closure is grown onto the structure.

Once the capillaries are grown, the three arterial networks are connected to the venous network and the closure is grown between the arterial trees and the venous tree using anatomical group indexing for connections (instead of connecting 2 independent trees). Because the networks are not symmetrical, there is a finishing step which closes the last terminals of the arterial or venous tree (whichever has more terminals).

7.2 Appendix B: Sample generator implementation and validation

In order to ensure no sampling bias in the sample generators, many samples were generated and the probability density functions calculated. The results indicate the sample generators do not have any inherent bias except the tetrahedral sampler (not used in any of the current growth algorithms).

7.2.1 Analytic X-Y plane sampler

The simplest sample generator is the X-Y planar sample generator. This samples in the analytic domain defined by the minimum point coordinate (p^o) and base vectors ($\overrightarrow{v_x}$ and $\overrightarrow{v_y}$) which are the vectors $\langle x_{\max} - x_{\min} \rangle$ and $\langle y_{\max} - y_{\min} \rangle$ respectively. These vectors each simplify to two independent scalar values that represent the edge length of the hexahedral domain in x, δx , and y, δy . The result shows no bias as visualized in Figure 7.15. The generation of a sample point (p^n) follows:

$$u = rand\{0,1\} \quad (7.5)$$

$$w = rand\{0,1\} \quad (7.6)$$

$$p^n = p^o + u \overrightarrow{v_x} + w \overrightarrow{v_y} \quad (7.7)$$

Which simplifies to:

$$p^n = [p_x^o + u \delta x, \quad p_y^o + w \delta y]^T \quad (7.8)$$

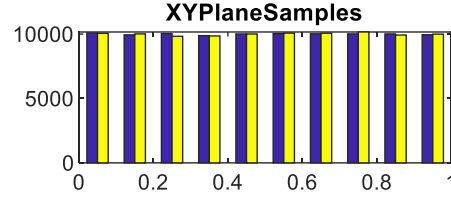


Figure 7.15. Validation of sample planar sample generator. 100,000 samples are binned into 10 bins by Matlab. The sample generator has a uniform probability distribution.

7.2.2 Analytic hexahedral sampling

A small variation to the X-Y planar sample generator is the addition of a depth parameter. This is a third random number and the calculation of a 3rd dimension which also shows no sample bias (Figure 7.16).

$$k = rand\{0,1\} \quad (7.9)$$

$$p^n = [p_x^o + u \delta x, \quad p_y^o + w \delta y, \quad p_z^o + k \delta z]^T \quad (7.10)$$

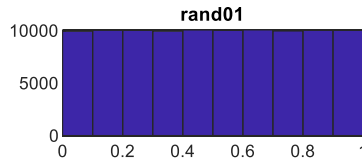


Figure 7.16. Validation of depth sample generator. 100,000 samples are binned into 10 bins by Matlab. The sample generator reveals a uniform probability distribution.

7.2.3 Triangle sample generator

Most sample generators only sample from a 1-dimensional PDF of either normal or Gaussian distribution. This has been shown capable of representing a hexahedral domain yet this method

must be modified in order to sample inside an arbitrarily defined triangulated shape. This kind of sample generator will define any random sample by characterizing a single triangle and creating a random sample inside the triangle. Such a sample generator has been designed and implemented as in Figure 7.18.

Note, this procedure follows previously published methods of staged growth [57]. In short, a PDF is generated with uniform distribution among the possible sample triangles:

$$\lambda \subseteq \{1, n\} \quad (7.11)$$

Where λ is the set of samples (list of triangles in the surface structure) and n is the number of mesh triangles in the given list. The PDF for this is a uniform distribution following:

$$p(\lambda) = U(1, n) \quad (7.12)$$

$$\int_1^n p(\lambda) = 1 \quad (7.13)$$

As described in previous work [57], sampling the subdomain (surface mesh triangle list) randomly before prescribing a random coordinate within the subdomain (within the triangle) is important to remove growth bias. This bias would be apparent when growing consecutively through the domains, as the tree structure would always look the same (always attach new segments from triangles in the same order). A sample triangle is depicted in Figure 7.17 with labeled point coordinates.

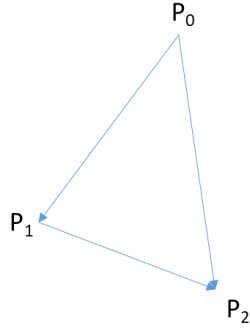


Figure 7.17. The graphical representation of a triangle.
The symbols are used in the mathematical expression below.

The first step is to define the base vectors of the triangle:

$$\alpha = \langle p_2 - p_1 \rangle, \quad \beta = \langle p_3 - p_1 \rangle \quad (7.14)$$

A random point is located inside the triangle if the scale of the 2 base vectors adds to 1:

$$pt = \alpha u + \beta v \quad (7.15)$$

$$pt = (u - v)p_1 + up_2 + vp_3 \quad (7.16)$$

Where u and v are scalars between 1 and 0. Given that $u + v = 1$:

$$t = (u - (1 - u))p_1 + up_2 + vp_3 \quad (7.17)$$

$$pt = p_1 + up_2 + vp_3$$

This shows that the update vector is just the source point (p_1) offset by a linear combination of p_2 and p_3 . In the event $u + v = 1$, the combinations of u and v describe the line $\langle p_3 - p_2 \rangle$. To define a point within the triangle, the evaluation becomes an inequality $u + v \leq 1$. In the event the point is outside the triangle, it can be mirrored over the vector $\langle p_3 - p_2 \rangle$ by a reassignment as in Equation (7.18). The mirroring is visualized in Figure 7.18.

$$u = 1 - v; \quad v = 1 - u \quad (7.18)$$

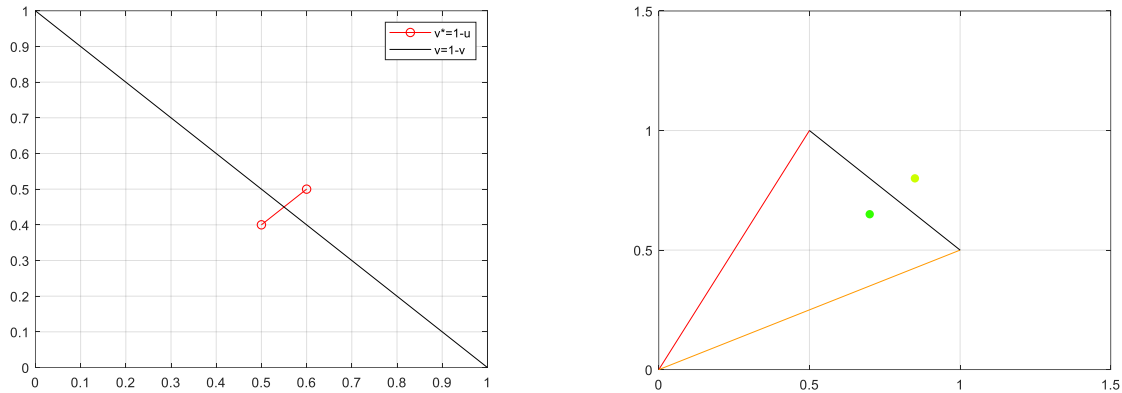


Figure 7.18. Visualization of the mirroring effect when a sample is thrown outside a triangle. The mirroring procedure to enforce all samples in an analytic domain (represented by base vectors of 2 lines in a triangle) are always within the bounds of the triangle. Left) The right triangle example shows how the point is perfectly mirrored across the triangle edge using a grid for reference. Right) The procedure also works with an oblique triangle, only that the sampling domain must be shifted from $0..L$ in two dimensions to $p_0..{\vec{v}}$ and $p_0..{\vec{u}}$ where p_0 is the source point (vertex) and \vec{v} and \vec{u} are the base vectors (red and orange).

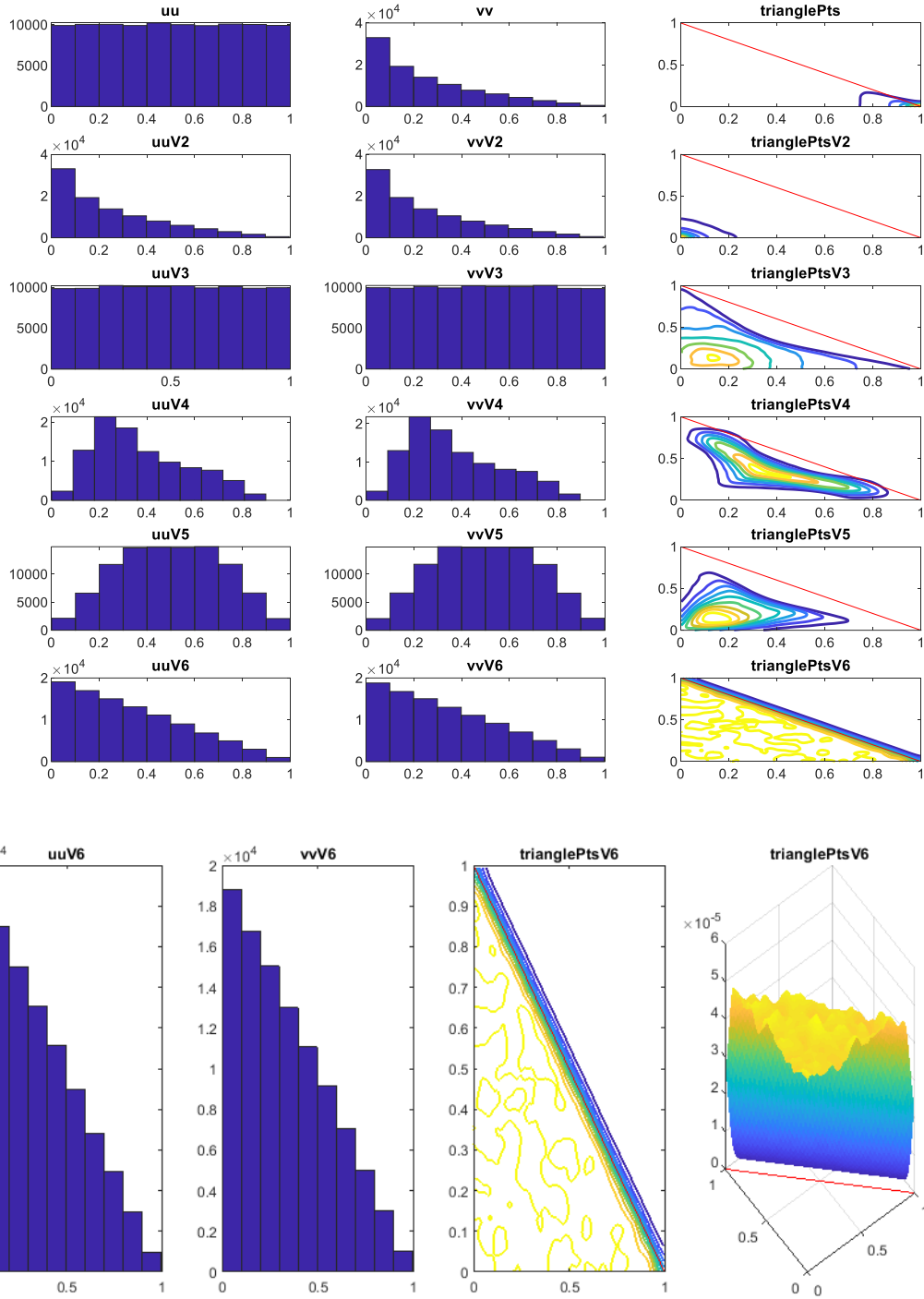


Figure 7.19. Different triangle sample generator bias evaluations. Different sample generators (termed V for 2-6 new versions) all had unique trends in the base vector scaling coefficients (u , v) leading to different distributions of samples in the triangle (right column, triangle is below red line). Only the final version (V6) shows a uniform distribution.

The distribution of different triangular sampling methods are compared in Figure 7.19. Only the mirroring technique (V6) gives unbiased samples. Note, the biased generators are not described here.

7.2.4 Triangle prism sample generator

A triangular prism is simply a triangle with a depth as shown in Figure 7.20. As such, the triangular prism generator (used for penetrating vessels and capillary growth when growing from a surface mesh) uses a triangle sample generator and applies a depth by offsetting the triangle surface sample by a scaled version of the triangle normal vector. Note, this method requires all normal vectors are unified and pointing towards the center of the mesh. The scaling of the normal vector is defined by a random value between 0 and maximum depth as described in Section 7.2.2.

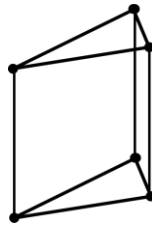


Figure 7.20. Triangular prism used in anatomical growth.
This is used to project a triangulated surface into a 3D depth easily and inexpensively.

7.2.5 Tetrahedral sampling

Tetrahedral sampling in an analytic domain is not as simple as in a 2-dimensional triangle, as the mirroring of a point across a face can result in being placed outside the domain on the other side of the tetrahedral. To avoid this, two methods have been implemented. Code is given in Section 7.7.1.2. Note, the probability is geometrically biased towards the center of the tetrahedral

because there is a natural bias away from the corners (1,0,0), (0,1,0) and (0,0,1) in the base vector length (u, v, w). This bias arises similar to the low probability of rolling 2 die and obtaining a 2 (only 1 permutation, 1/36 probability) vs rolling a 6 (5 permutations, 5/36 probability).

7.2.5.1 Method 1: using the vector norms

The ad-hoc method for sampling within a tetrahedral uses the norm of the base vectors to rescale in the event that the length of the resulting vector is too long. This method has a bias towards the middle of the region as it does not address the bias discussed previously. The results visualized in Figure 7.21 reveal a bias towards the center of the tetrahedron. A pseudocode is offered:

Table 7.9. Pseudocode for tetrahedralSampleGenerator1

1. *translatePtCoordToWithinTetrahedron*(newPt,aVolIdx,u);
2. *getFacesAsVectors*(v1,v2,v3,v4,aVolIdx,newPt);
3. vNewA = *scale*(v1,u[0], v2,u[1], v3,u[2]);
4. IF *sum*(u) > 1 THEN DO
5. temp[0] = norm(u) – norm(u[1],u[2]); temp[1] = norm(u) – norm(u[0],u[2])
6. temp[2] = norm(u) – norm(u[0],u[1]); u = temp;
7. END;
8. END;

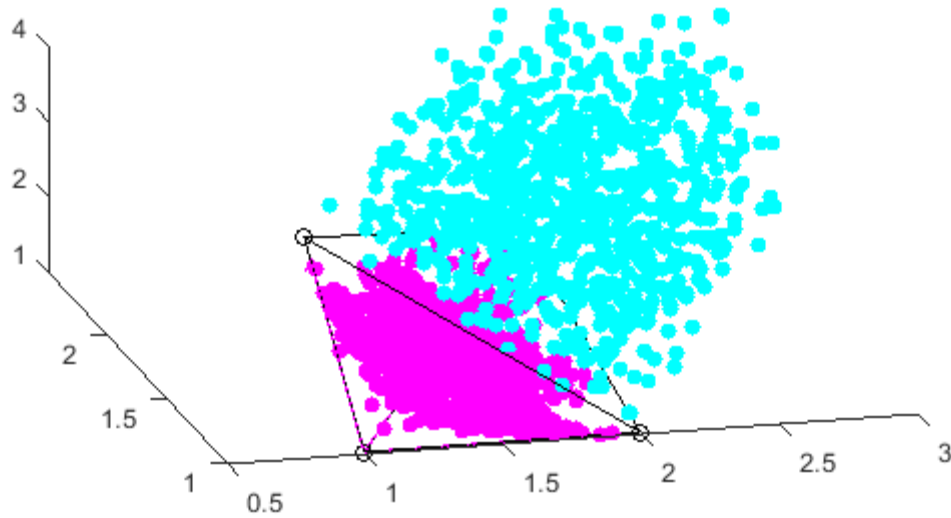


Figure 7.21. In this case, the projection into the domain is validated. Original points in blue and final, translated points in pink. All final points lie within the tetrahedron.

7.2.5.2 Method 2: using the random length sampling

An alternative method uses a random length variable applied to all vectors only when the sum of the scaling parameters is greater than 1. The 3 coordinate vectors determine the direction and another random variable calculates the length. This has been shown graphically:

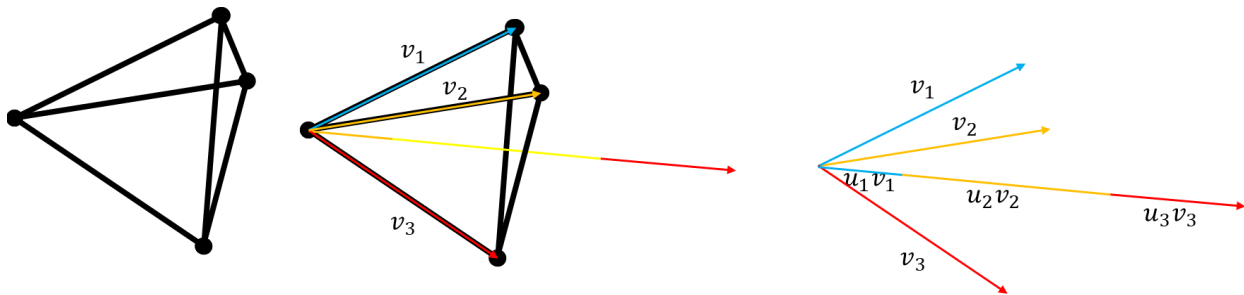


Figure 7.22. Visualization of how three random variables create a vector outside a tetrahedron. The original tetrahedron compared to the sampling vectors (v_1 , v_2 , and v_3) scaled by the corresponding randomly selected weighting variables (u_1 , u_2 , and u_3). In this case, the sum of weighting variables is greater than 1, so the new vector is outside the tetrahedron.

The triangle ($T(x,y,z)$) opposite the origin point can be characterized by all permutations of u_1, u_2 , and u_3 that sum to a value of 1:

$$T(x, y, z) = \sum_{i=1}^3 u_i v_i(x, y, z) \quad (7.19)$$

Where

$$1 = \sum_{i=1}^3 u_i$$

The sum of u_1, u_2 , and u_3 describes the length of the vector, where $\sum_{i=1}^3 u_i = 0..1$ designates a point inside the tetrahedron. In the event that the value is greater than 1 (or less than 0 which does not occur in any of the calculations because the random numbers are bounded at 0), the length can be set by an independent random variable (i.e. $L = \text{rand}(0,1)$). This method fixes the bias towards the tetrahedral corner of origin, but does not fix the bias away from the other corners of the tetrahedral as seen in Figure 7.23. A pseudocode for this method is offered:

Table 7.10. Pseudocode for tetrahedralSampleGenerator2

1. *translatePtCoordToWithinTetrahedron*(newPt,aVolIdx,u);
2. *getFacesAsVectors*(v1,v2,v3,v4,aVolIdx,newPt);
3. vNewA = *scale*(v1,u[0], v2,u[1], v3,u[2]);
4. IF *sum*(u) > 1 THEN DO
5. L = *rand*(0,1); delta = L/*sum*(u); u = *scale*(delta,u);
6. ENDIF;
7. END;

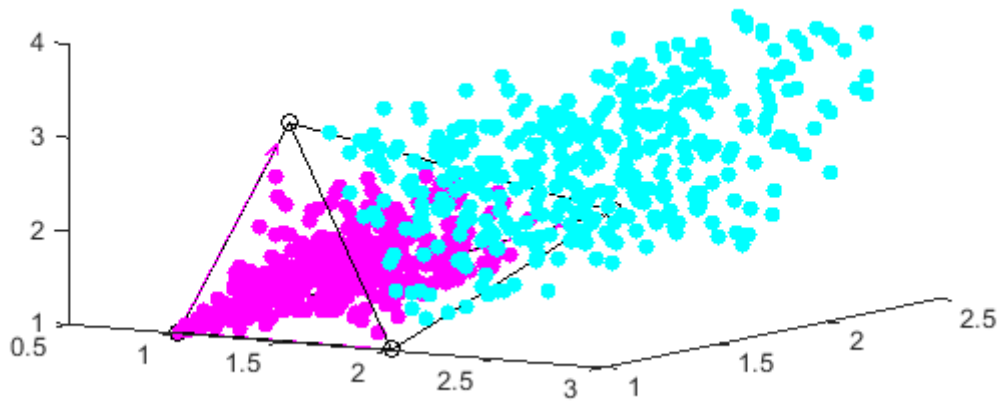


Figure 7.23. The second method for tetrahedral sampling. Original points in blue and final, translated points in pink. All final points lie within the tetrahedron.

7.2.5.3 Method 3: mirroring

A method inspired by the mirroring developed for the triangle sample generator also found a bias towards the center of the tetrahedron as seen in Figure 7.24. The method first checks that the value is inside all 3 base triangles independently. This method has a bias towards the origin, as the majority of the space in the analytic domain (hexahedron surrounding the tetrahedral) is very large. The method is implemented by repetitively mirroring across the triangular sides of the tetrahedral until the point lies within the structure.

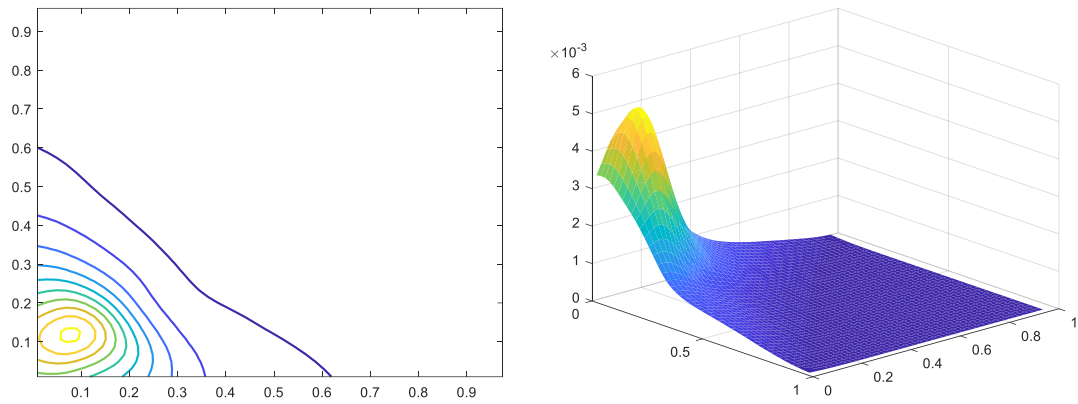


Figure 7.24. The second method for tetrahedral sampling. Original points in blue and final, translated points in pink. All final points lie within the tetrahedron.

7.3 Appendix C: Calculating cortical depth from a mesh

The steps to calculating the cortical depth follows the variables outlined in Figure 7.25.

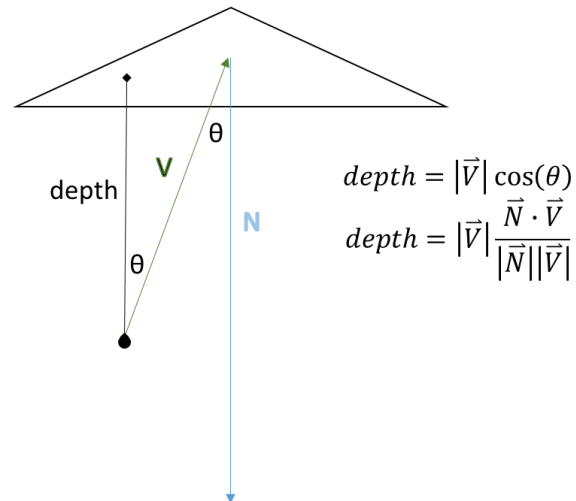


Figure 7.25. Graphical representation of calculating depth from a surface triangle. This is computed by calculating the perpendicular distance to that triangle. This is the calculation used to identify depth in the MCA territory.

Pseudocode for calculating depth for all segments in a network is offered:

- Table 7.11. Pseudocode for calculating perpendicular distance from a triangle
1. *calculateDepthForAllFacesInNetwork*(nwk, mesh, faceMatchingList):DBLArray;
 2. FOR iFace =1 TO nwk.nFaces DO
 3. meshFace = faceMatchingList[iFace]; n = *getFaceNormal*(mesh, meshFace);
 4. faceCenter = *getFaceCenter*(nwk, iFace); meshCenter = *getFaceCenter*(mesh, meshFace);
 5. V = *getAsVector*(faceCenter, meshCenter);
 6. Depth[iFace] = *norm*(V)**dot*(n,V)/(*norm*(V)**norm*(n));
 7. ENDFOR;
 8. END;

7.4 Appendix D: Raw dataset topological analysis

7.4.1 Calculating Murray coefficient

Murray's law (Equation (7.20)) is a nonlinear relationship between parent diameter and daughter diameter. Finding the unknown coefficient (γ) can be formulated as a least square optimization problem (Equation (7.21)). Under the assumption the minimum value is $Z=0$, the minimization becomes a nonlinear equation (Equation (7.22)) which can be solved using the Newton method with the gradient defined in Equation (7.23).

$$d_p^\gamma = d_1^\gamma + d_2^\gamma \quad (7.20)$$

$$\min_{\gamma} Z(\gamma)$$

$$\text{Where} \quad (7.21)$$

$$Z(\gamma) = [d_p^\gamma - (d_1^\gamma + d_2^\gamma)]^2$$

$$[d_p^\gamma - (d_1^\gamma + d_2^\gamma)]^2 = 0 \quad (7.22)$$

$$\frac{dZ}{d\gamma} = 2[d_p^\gamma - (d_1^\gamma + d_2^\gamma)] [\ln(d_p) d_p^\gamma - (\ln(d_1) d_1^\gamma + \ln(d_2) d_2^\gamma)] \quad (7.23)$$

Validation of the implementation can be seen graphically where the residual error nonlinear 1-dimensional equation (Equation (7.22)) gives optical identification of the solution region (Figure 7.26). This value is identified by the Newton method for two case studies; an idealized geometry (not from images) and an image-based reconstructed bifurcation for which the Murray coefficient is a positive number. The following case studies were examined:

#	Type	Parent diameter (μm)	D_1 diameter (μm)	D_2 diameter (μm)	coefficient
1	Idealized	4	2	1.5	0.8371
2	μCT reconstruction	37.721925	32.6994	20.46435	2.165

The findings show a solution (if one exists) occurs in the positive domain, yet the Newton method sometimes converges to an asymptotic value of 0 in the negative domain (no solution exists). In other words, the solution takes the form of Figure 7.27 Top Left where a solution may not exist and a numerical method may converge to a false solution in the negative domain ($0+0=0$).

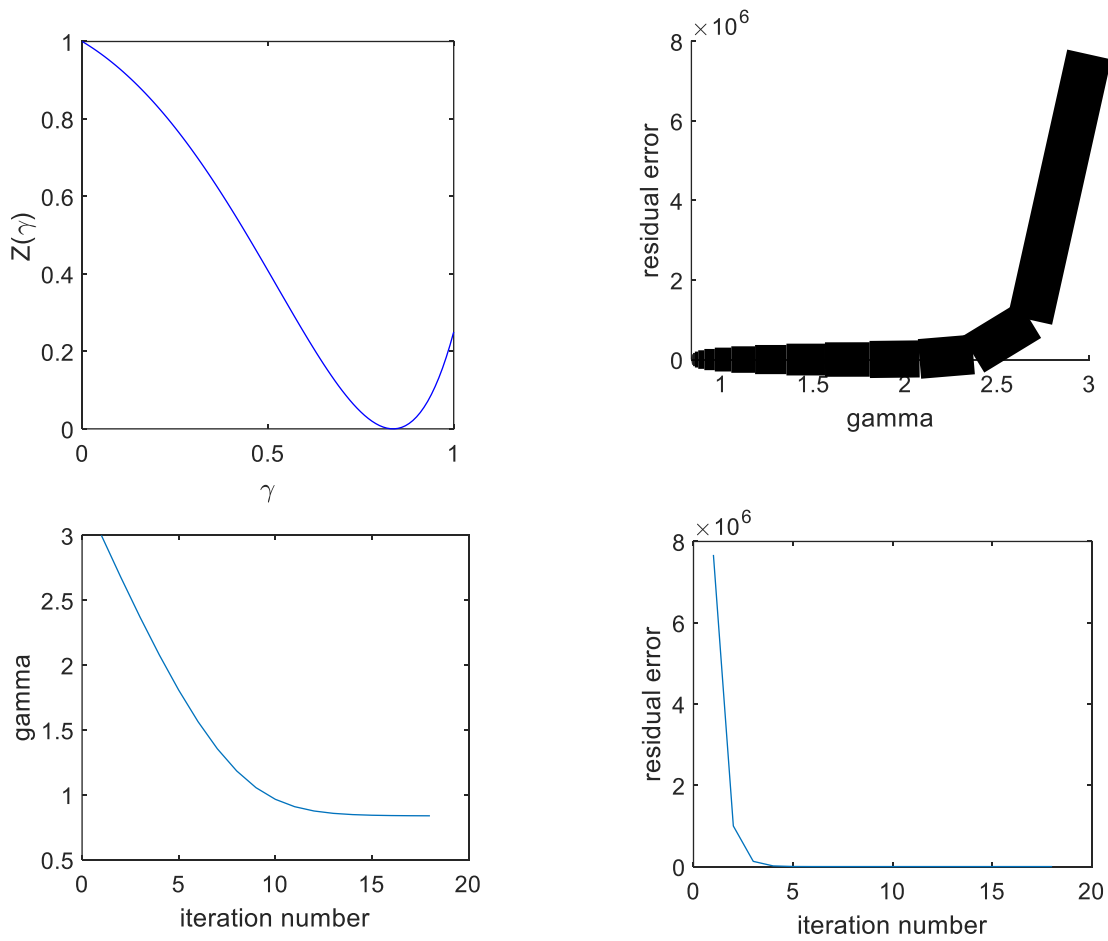


Figure 7.26. The solution opf case study 1 is validated by both Matlab and Delphi at $\gamma=0.87$.

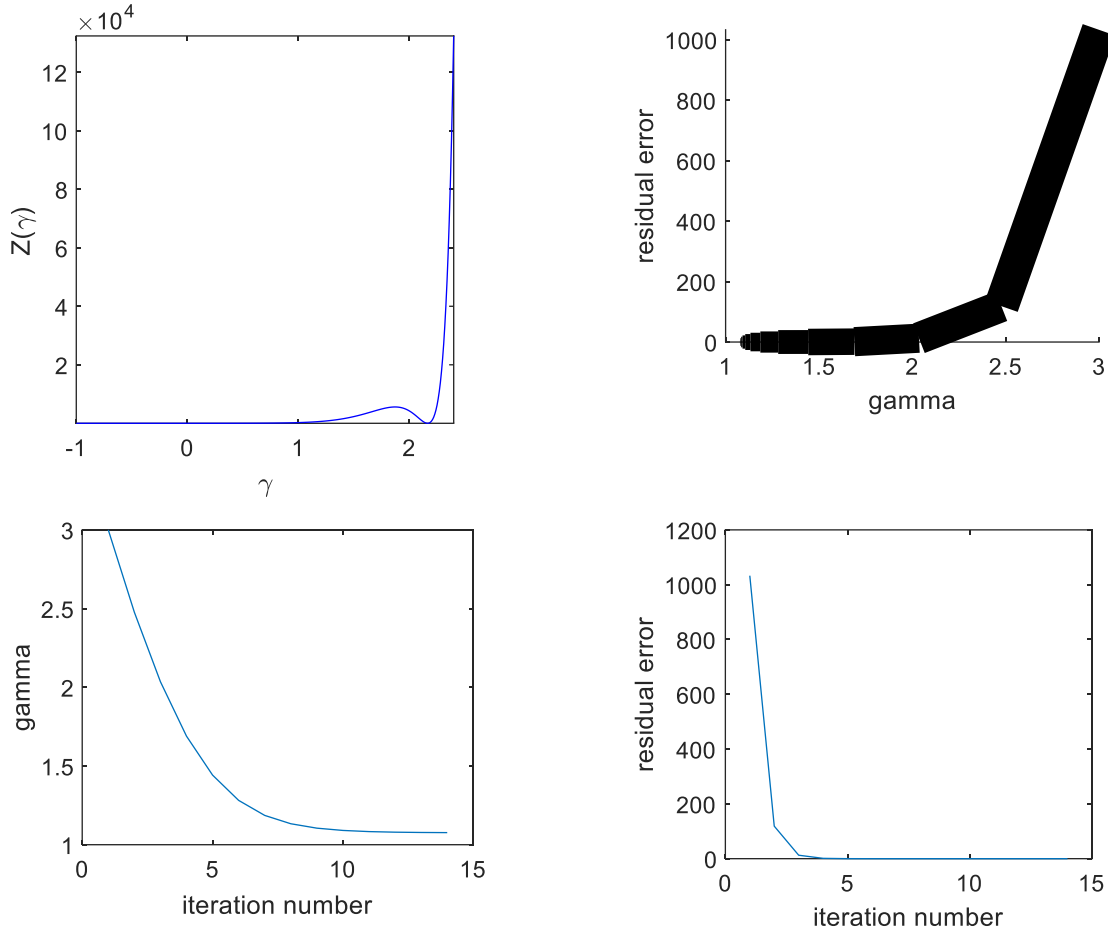


Figure 7.27. Solution to case study 2 is $\gamma=2.165$ as verified by Matlab and Delphi.

A case study with no solution. A case study was generated from a reconstructed bifurcation that iteratively converged to a negative Murray coefficient. In the cases where a negative value is obtained, the reasoning is that the minimum of the Murray residual equation (Equation (7.22)) is a negative number. This results in a significant residual magnitude for all meaningful regions for the Murray coefficient (Figure 7.28 Left).

#	Type	Parent diameter (μm)	D ₁ diameter (μm)	D ₂ diameter (μm)	coefficient
3	μCT reconstruction	37.721925	36.75	21.947925	-2.3581

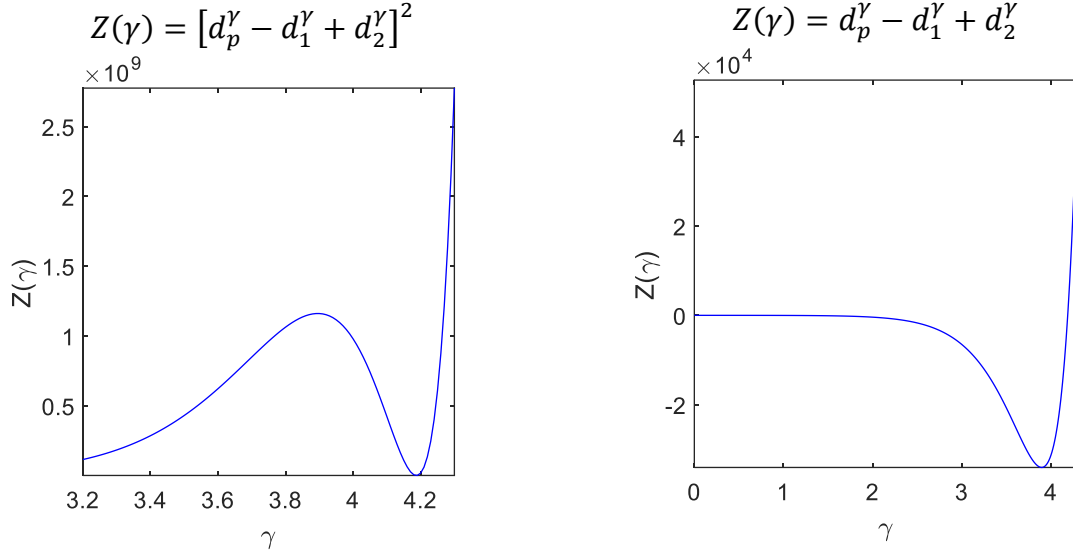


Figure 7.28. Case study 3 does not have a solution. In this case, the double-root occurs with a very large residual value, meaning it is not a solution.

7.4.2 Spectra of Murray coefficient from μ CT data

Anatomical reconstructions of μ CT images of mouse vasculature were performed using previously published methods [68–70]. This network is apt to calculating the Murray coefficient. The distribution is offered in Figure 7.29. The diameter coefficients are offered with the lines for different values of γ in Figure 7.30. The values obtained from a reconstruction at an imaging resolution of $20\mu\text{m}$ is shown in Figure 7.29 - Figure 7.30. The values obtained from a reconstruction at an imaging resolution of $7\mu\text{m}$ is shown in Figure 7.31 - Figure 7.32.

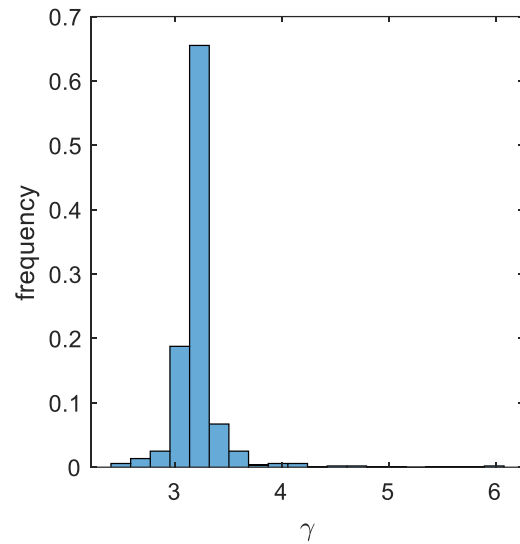


Figure 7.29. Histograms of the Murray coefficient for the 20 micron pial surface, solved for using newton method.

Note, range is 2.465-6.069, in many cases the newton method did not converge (but more converged than did not).

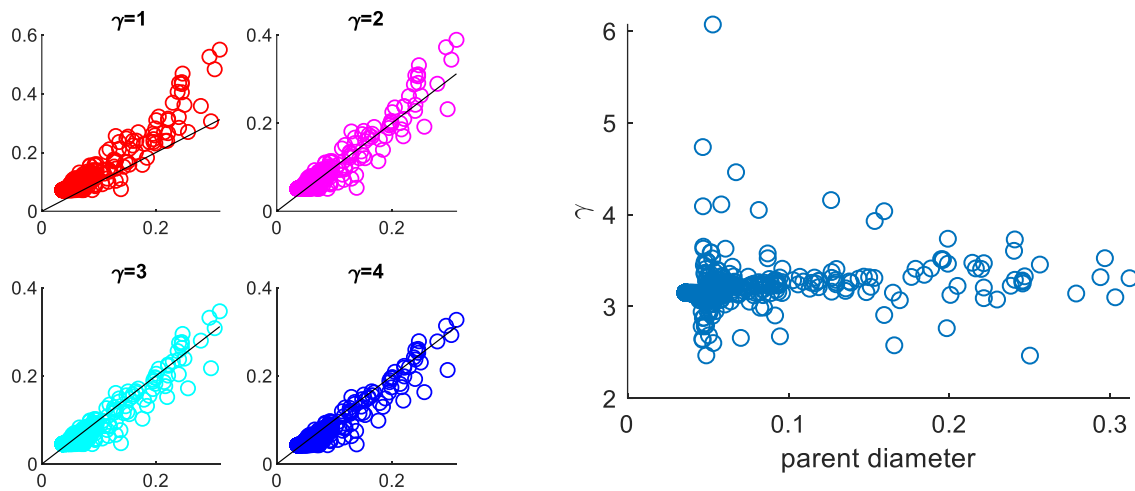


Figure 7.30. Dataset predictions of parent diameter (with varying coefficient, γ) compared to experimental parent diameter.

The most reasonable coefficient lies between 0.8 and 1.0.

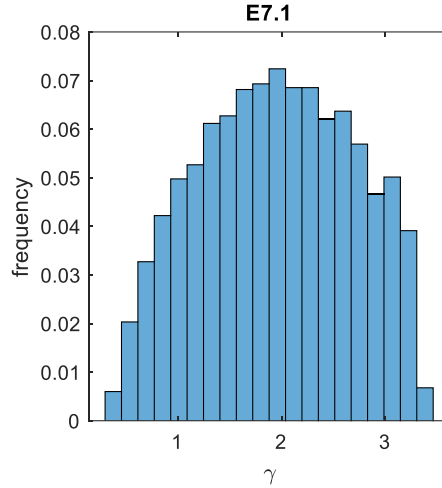


Figure 7.31. Histograms of the murray coefficient for the 7 micron pial surface, solved for using newton method.

Note, range is -120 to 20, in many cases the newton method did not converge (but more converged than did not).

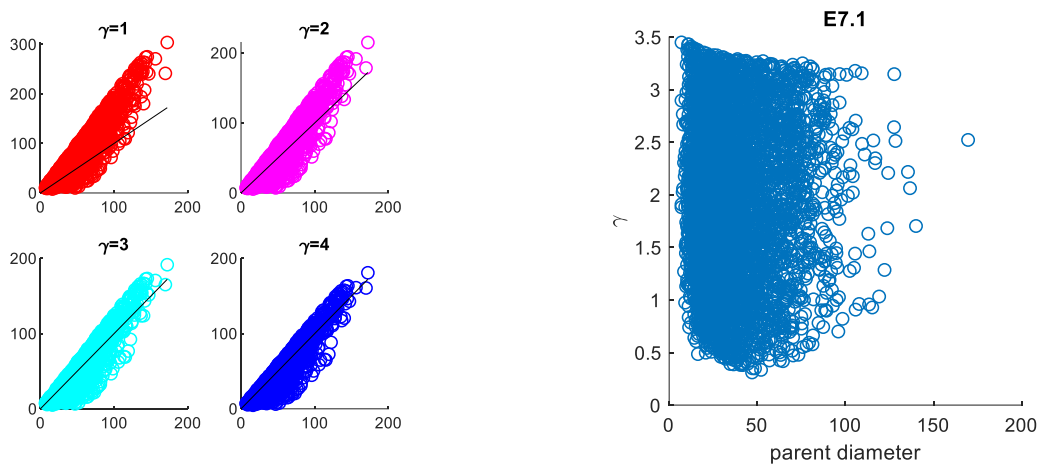


Figure 7.32. micron dataset predictions of parent diameter (with varying coefficient, γ) compared to experimental parent diameter.

The most reasonable coefficient lies between 0.8 and 1.0.

7.4.3 The spectra of Murray coefficient from 2PLSM microcirculation

Four anatomical reconstruction of 2 photon laser scanning microscopy (2PLSM) images of mouse microvasculature was performed using previously published methods [26]. The distribution

of Murray coefficients is offered in Figure 7.33. The diameter coefficients are offered with the lines for different values of γ in Figure 7.34 - Figure 7.35 for individual datasets. The cumulative distributions are offered in Figure 7.36 - Figure 7.37.

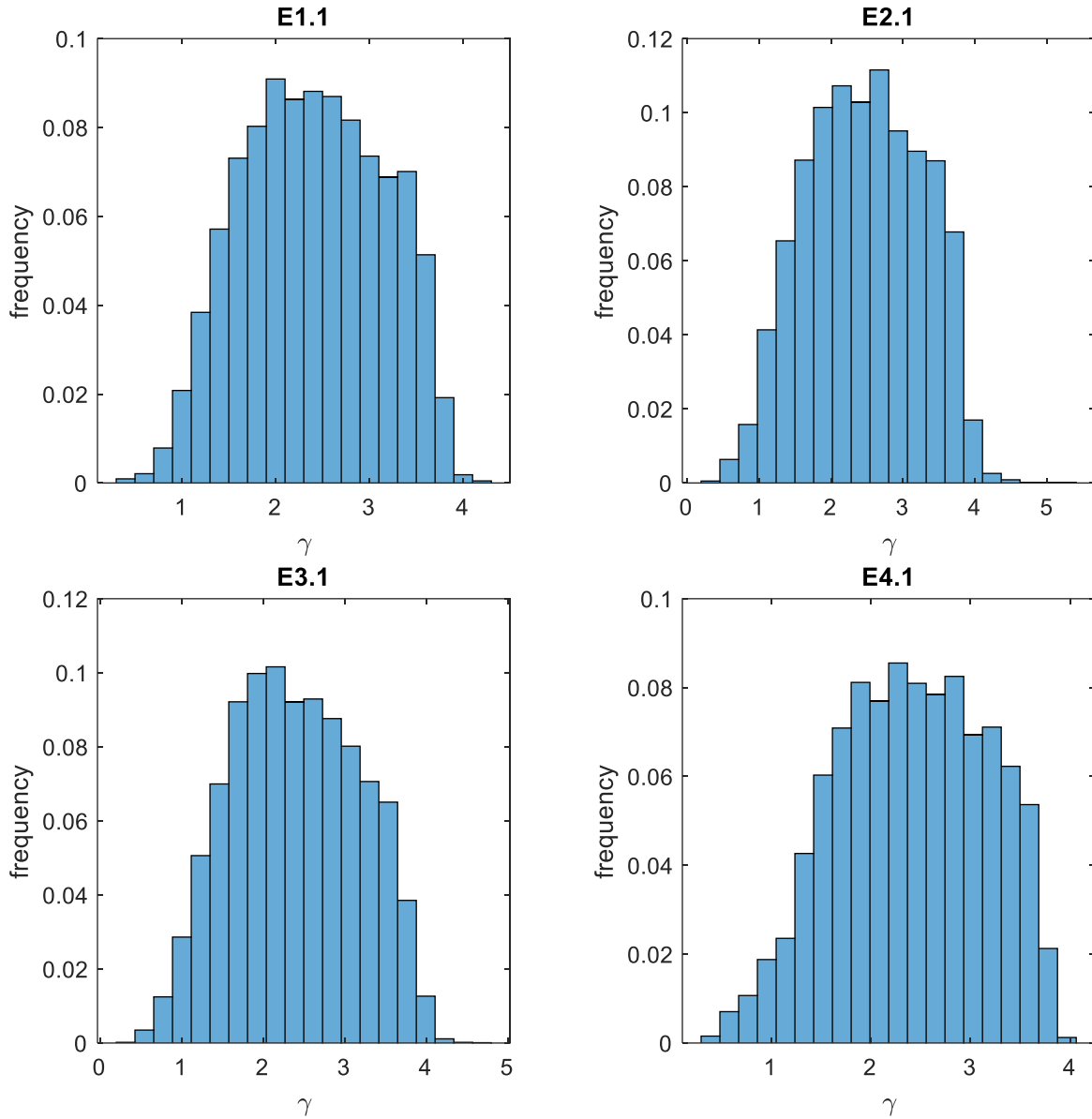


Figure 7.33. The comparison of the 4 empirical datasets and their respective Murray coefficient as derived using the Newton method.

Note, all negative values are a result of non-convergence of the Newton method (See section 7.4.1).

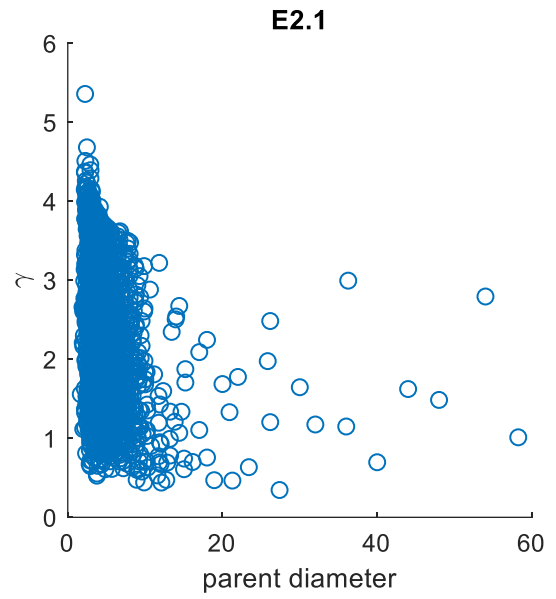
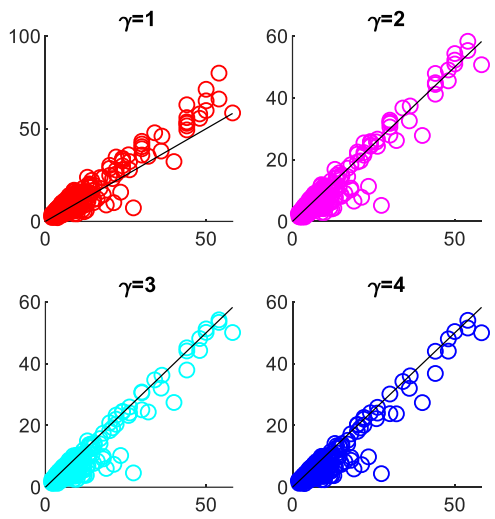
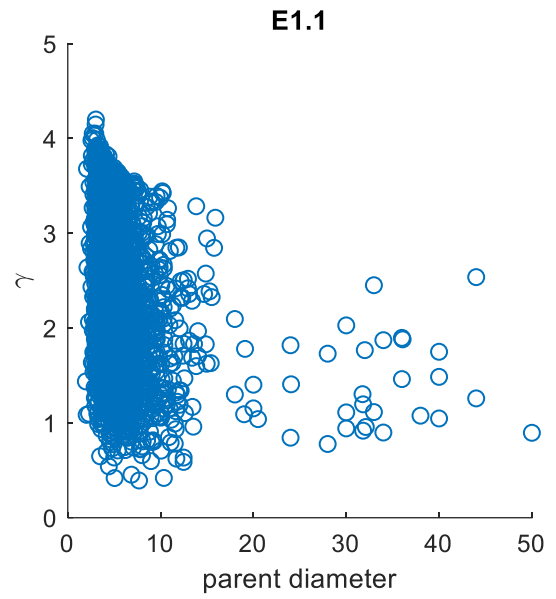
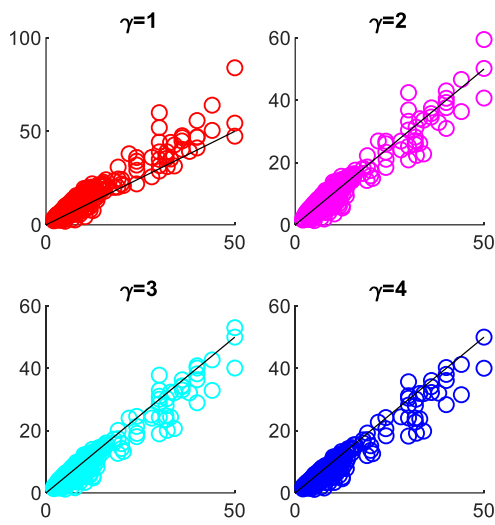


Figure 7.34. The comparison of the first 2 empirical datasets and their respective Murray coefficient as derived using the Newton method.

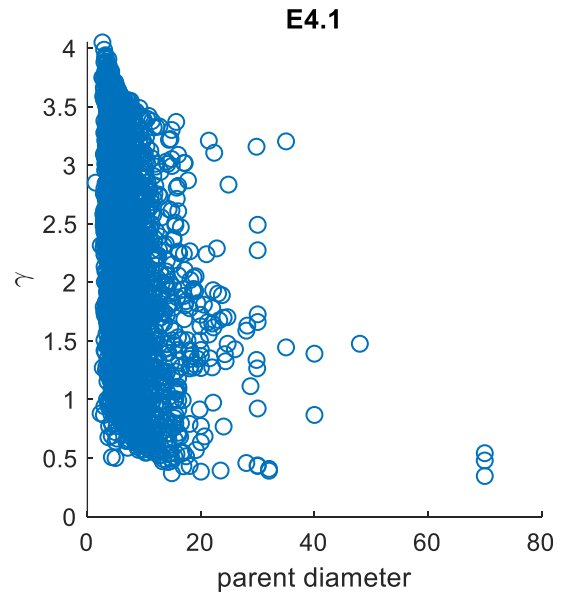
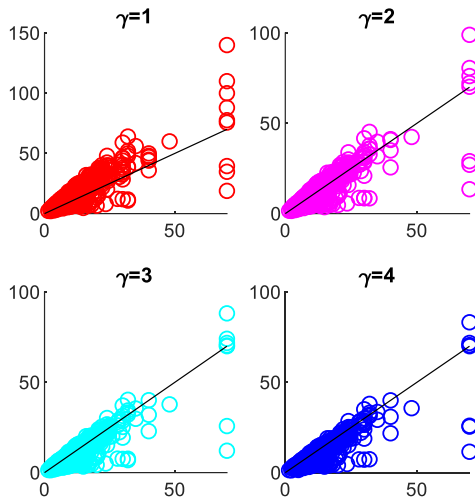
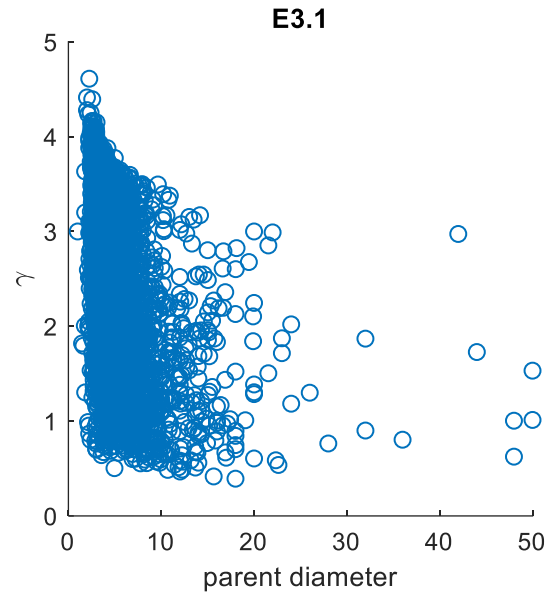
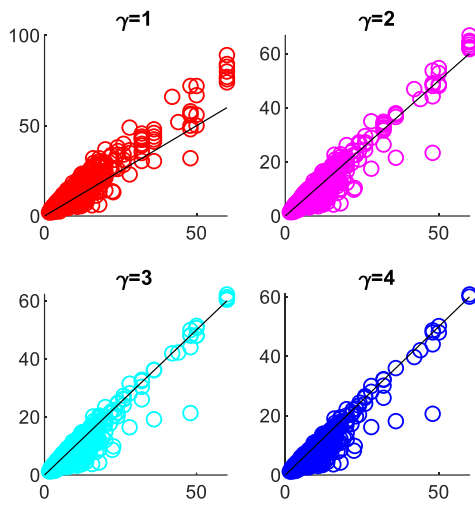


Figure 7.35. The comparison of the last 2 empirical datasets and their respective Murray coefficient as derived using the Newton method.

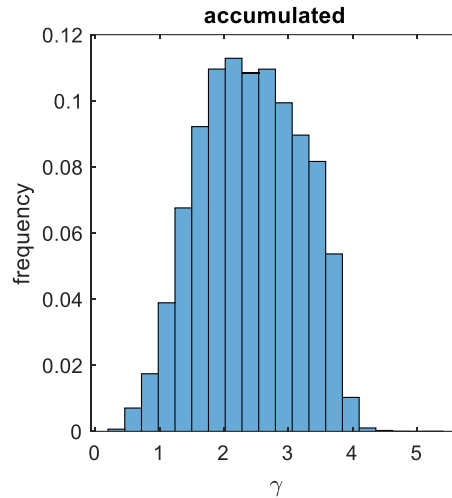


Figure 7.36. Histograms of the murray coefficient for accumulated 4 empirical microcirculatory datasets.

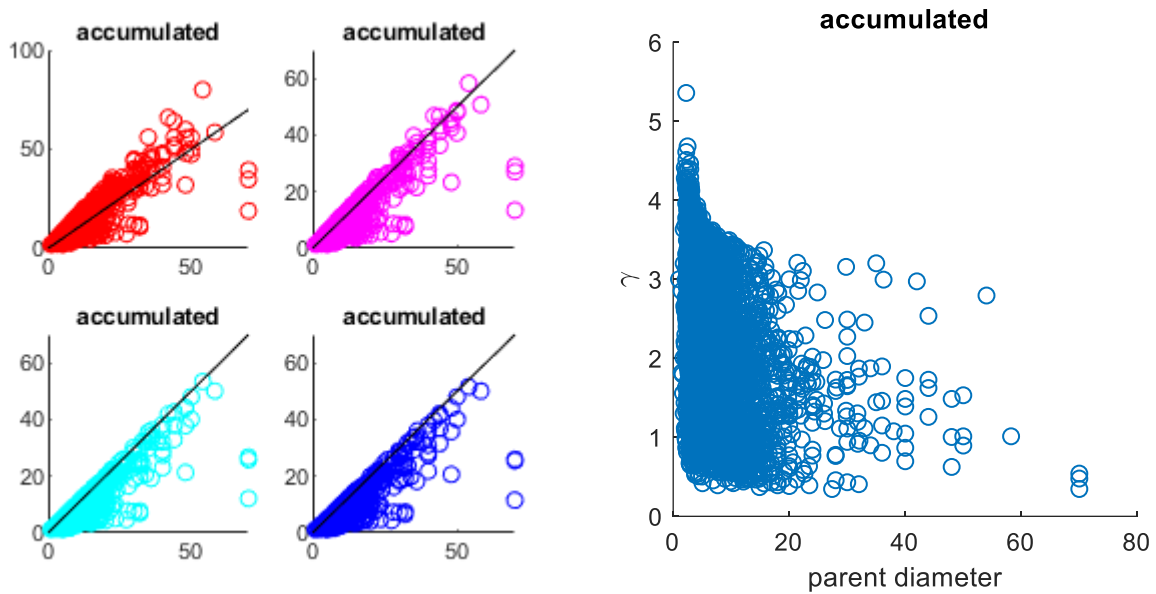


Figure 7.37. The comparison of the cumulative empirical datasets and their respective Murray coefficient as derived using the Newton method. The plotted lines indicate $\gamma = 1, 2, 3, 4$ for scatter plots in red, magenta, light blue, and dark blue respectively.

7.4.3.1 Anatomical characteristics of cerebral microcirculatory networks

Small penetrators. The primary dataset for investigation in this section is the Kleinfeld datasets [41], as they are significantly larger than other datasets [24,25] and are anatomically labeled. These datasets came prescribed with large penetrating vessels pre-labeled:

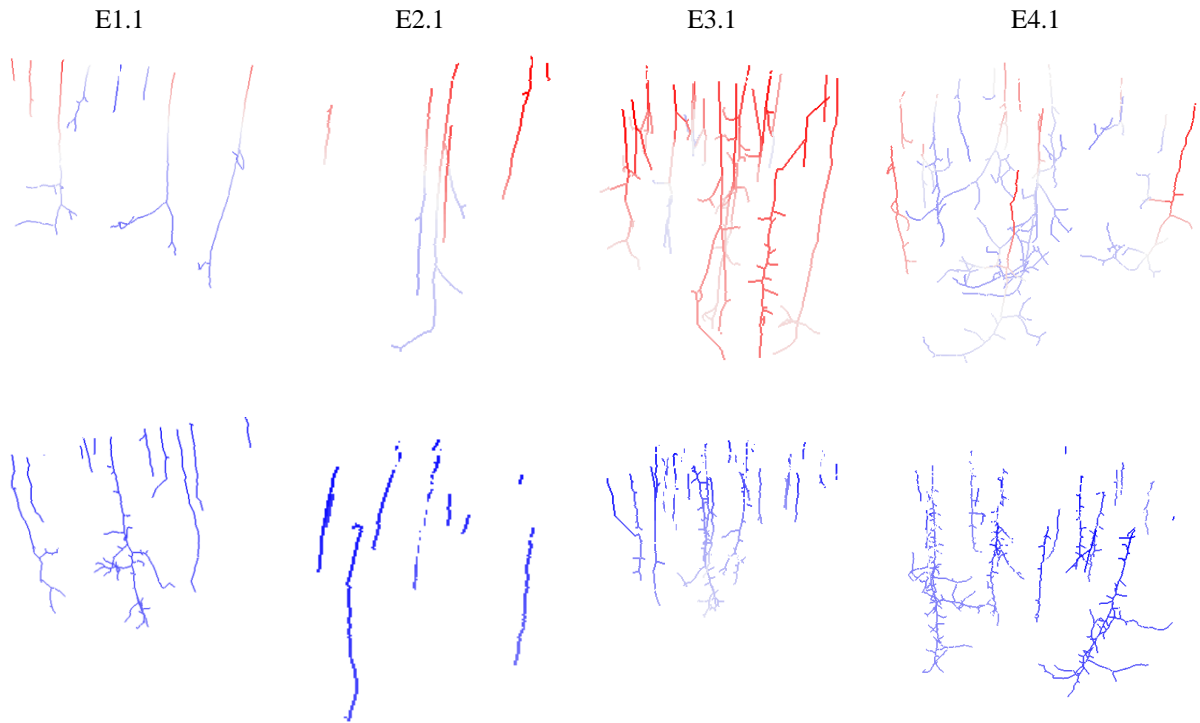


Figure 7.38. Visualization of the automatically labeled penetrating arteries and ascending veins in the empirical networks.

(Top) the penetrating arterial trees show higher pressure (red) than the (Bottom) ascending venules (blue). Note, not all penetrating vessels of the same group exhibit the same pressure.

When interrogating these trees of vessels, the relevant topological parameter for vascular synthesis is the number of purely vertically-aligned vessels. These have been counted in the labeled structures to reveal the ratio of penetrating arteries to ascending venules is in the range=0.45-0.72 with a mean of 0.66 and a standard deviation of 0.16. Moreover, the longer penetrating vessels

were noted to bifurcate primarily in the bottom $\frac{1}{2}$ of the domain (not many branches near the pial surface).

During this investigation, many small vessels appeared that stemmed from the pial vasculature but only penetrated into the top $\frac{1}{3}$ of the dataset. Images of these short penetrators can be seen in Figure 7.39 with close-up views in Figure 7.40. These small pial vessels also were noted to have different bifurcating patterns; namely, they branch fairly uniformly along their length. A schematic of these branching patterns has been drawn in Figure 7.41 to exemplify this finding.

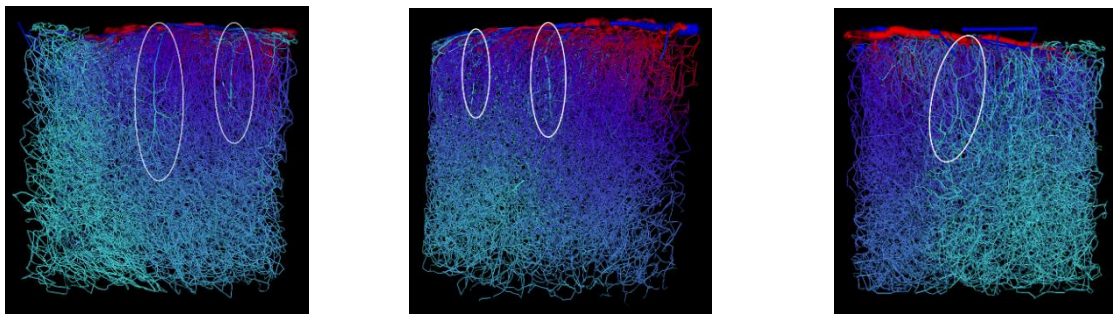


Figure 7.39. Visualization of the whole E1.1 network with penetrating vessels highlighted. These vessels penetrate $\frac{1}{4}$ - $\frac{1}{3}$ into the depth of the network. With these new penetrators, an estimated ~ 80 total penetrating vessels/ mm^2 surface area coverage is estimated.

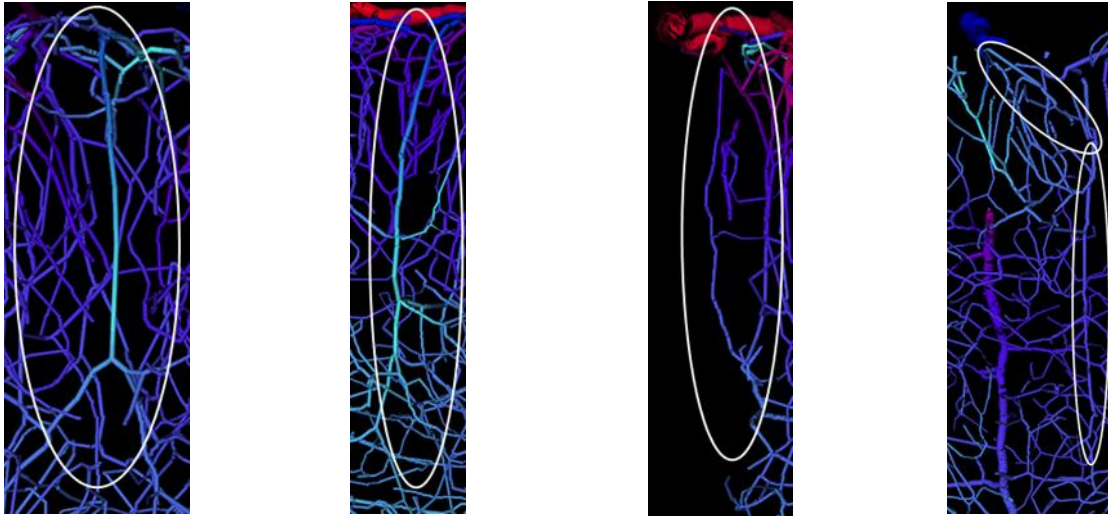


Figure 7.40. Visualization of short penetrating vessels that were initially overlooked by the labeling algorithm.

Note, many of these are short and narrow, so they were originally classified as capillaries.

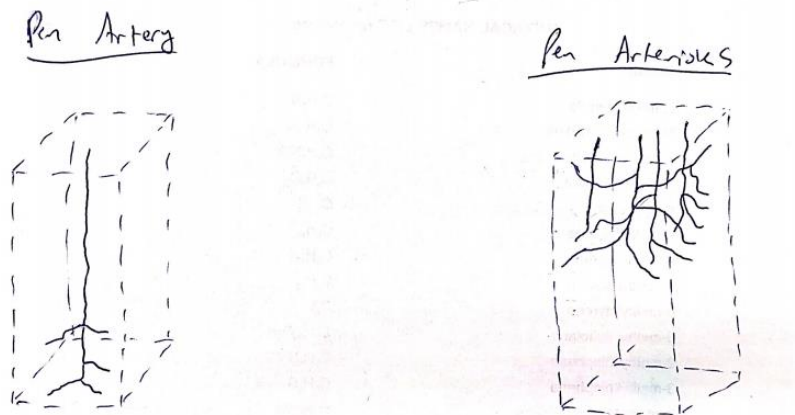


Figure 7.41. Pictorial representation of the two classifications of penetrating vessels on the arterial side.

Left) the penetrating arteries are larger in diameter, penetrate deeper into the tissue, and do not branch until reaching the lower portions of the dataset. Right) the penetrating arterioles are shallow and branch throughout the depth they cover.

A total of ~80 penetrators have been identified by manual inspection. To account for this, the model could be adapted to grow the deep penetrators (~30) allowing the penetrators to angle up to 10 degrees from direct-vertical alignment. The method would then grow $\frac{1}{2}$ the capillary bed before

growing the additional 50 small, short penetrators. Then the remaining capillary bed would be finished before continuing with connection and topological agreement. The results of this adaptation are outside the scope of this work.

Vertical alignment bias. Another notable characteristic of the empirical networks is a propensity of vessels to align parallel or perpendicular to the Z-axis. The centerline of every vessel can be characterized by a vector (v) has 3 components, the x, y, and z component as pictured in Figure 7.42.

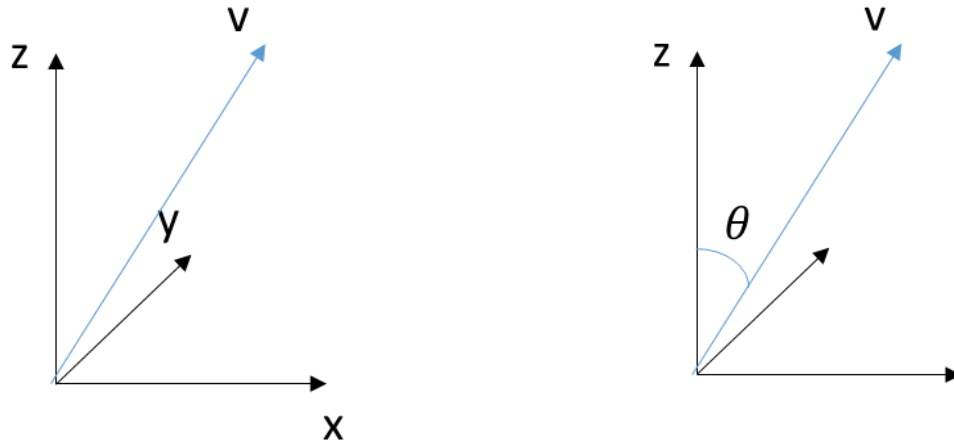


Figure 7.42. The graphical representation of a face (as vector v in blue) in an x, y, and z coordinate system.

The proportional magnitude of the z component (v_z) can be calculated using:

$$\alpha = \frac{|v_z|}{|v|} \quad (7.24)$$

And the angle from the z-axis can be calculated following:

$$\theta = \arccos(\alpha) \quad (7.25)$$

And the resulting distribution for empirical and synthetic networks is offered:

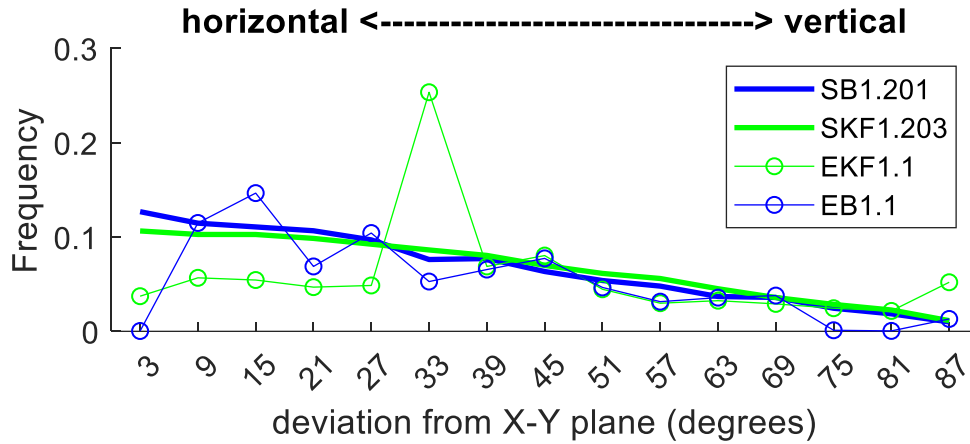


Figure 7.43. Probability density function of the vertical alignment of empirical networks. Note, the values are plotted at the center of each bin.

The network segments that share an alignment with the z-axis can be defined by the cosine of the angle between the z-axis and the segment centerline being close to 1 (>0.9). The segments perpendicular to this axis have a small cosine (<0.1). The vessels of network E1.1 (also designated by EKF1.1) have been characterized by these parameters and visualized in Figure 7.44.

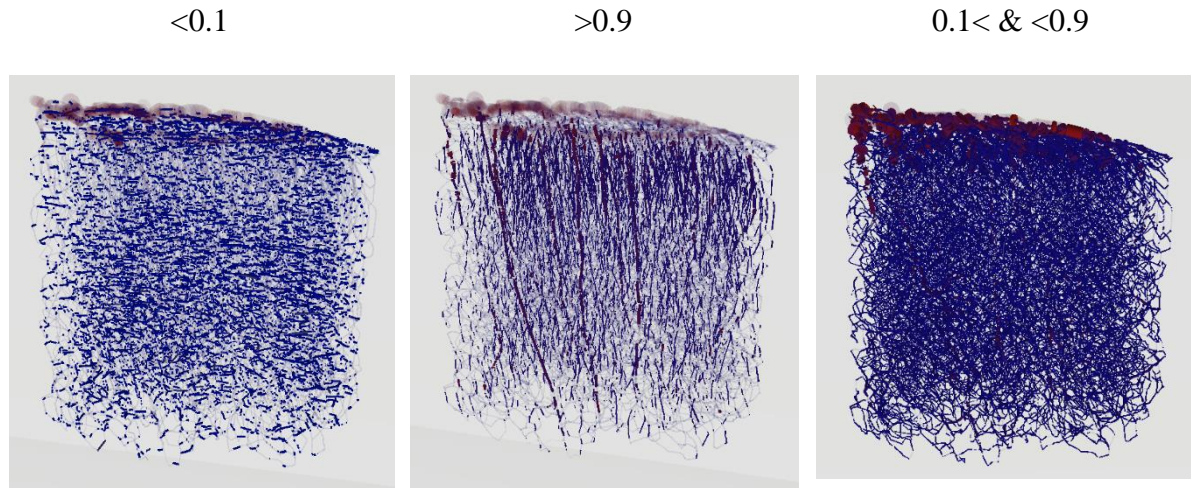


Figure 7.44. Visualization of vertical alignment of empirical network EKF1.1. Left) horizontal, Middle) vertical and Right) all other segments. Note even with very tight criteria, a very large number of segments are identified by the alignment to horizontal or vertical angles.

Proposed adaptation to algorithm. In order to synthesize anatomically consistent cerebrovascular networks, it is not sufficient to use a random space-filling algorithm alone, as this algorithm will not account for inherent bias in the empirical network. In order to incorporate these topologically consistent characteristics of cerebral microcirculatory networks into the previously proposed iCNS algorithm, the penetrator growth stage should only grow the deep penetrators, then grow $\sim 1/2$ the capillary bed (using multiple stages of consecutively relaxing constraints). The shorter penetrators ($\sim 1/2$ of the total ~ 78 penetrators/mm² pial surface coverage) are then added and the remainder of the capillaries grown. This will help limit the random sample generating bias of the segments near the pial vessels towards the small penetrators and ensure these vessels attach to the short penetrators and trees from larger penetrators alike (as happens in empirical networks).

Another modification is to add the large penetrators as a series of 10 extensions, as opposed to a single vessel. This will assist in the horizontal bias when attaching to these vessels. The penetrating vessel should begin straight and have some degree of *angle* from the Z-axis by

allowing the x-y coordinate of the penetrator to vary by a random number giving 0-10 degrees of skewedness to the new terminal point coordinate. After the coordinate is set and the addition of 10 segments (signifying a single penetrator) takes place, some vessels (~4 per penetrator) should be grown within a cylinder surrounding the vessel (~100-125 μ m).

7.5 Appendix E: Perpendicular line connections, alternative methods

To characterize the distance between a new sample and a selected face in the network, a calculation of the perpendicular distance between a point and a line is necessary. To do this, the intersection point must be defined along the segment arc that guarantees the two vectors are perpendicular (dot product is 0) as in Equation (7.27):

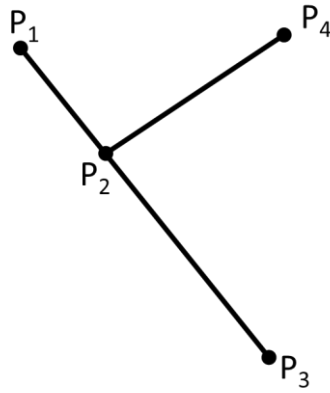


Figure 7.45. Schematic representation of a new sample (P₄) and the intersection point with the existing face (P₁-P₃) at point P₂.

$$0 = [x_3 - x_1, y_3 - y_1] \cdot \begin{bmatrix} x_2 - x_4 \\ y_2 - y_4 \end{bmatrix} \quad (7.26)$$

This equation is written in the x-dimension but can likewise be written in y and z. The solution to this dot product can be written in the unknown value (x₂):

$$0 = (x_3x_2 - x_1x_2 - x_3x_4 + x_1x_4) + (y_3y_2 - y_1y_2 - y_3y_4 + y_1y_4) \quad (7.27)$$

$$(x_3x_4 - x_1x_4) + (y_3y_4 - y_1y_4) = [x_3 - x_1 \quad y_3 - y_1] \begin{bmatrix} x_2 \\ y_2 \end{bmatrix}$$

Which gives 1 equation with 2 unknowns. A second equation should enforce that the second point lies on the line $\langle p_1-p_3 \rangle$. This can be accomplished by fitting an analytic line encompassing both p_1 and p_3 and ensuring the new point obeys to that line. This can be accomplished by first fitting a line to the two points in the original network face:

$$\begin{bmatrix} y_1 \\ y_3 \end{bmatrix} = \begin{bmatrix} x_1 & 1 \\ x_3 & 1 \end{bmatrix} \cdot \begin{bmatrix} m \\ b \end{bmatrix} \quad (7.28)$$

Which will result in a value of m and b . This can be used to ensure:

$$y_2 = mx_2 + b$$

Which can be rewritten as: (7.29)

$$b = -mx_2 + y_2$$

Which can be added to the original equation (Equation (7.27)) resulting in a final set of equations:

$$\begin{bmatrix} (x_3x_4 - x_1x_4) + (y_3y_4 - y_1y_4) \\ b \end{bmatrix} = \begin{bmatrix} x_3 - x_1 & y_3 - y_1 \\ -m & 1 \end{bmatrix} \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} \quad (7.30)$$

This has been implemented in Matlab for validation. Code is offered in Section 7.7.1.3.

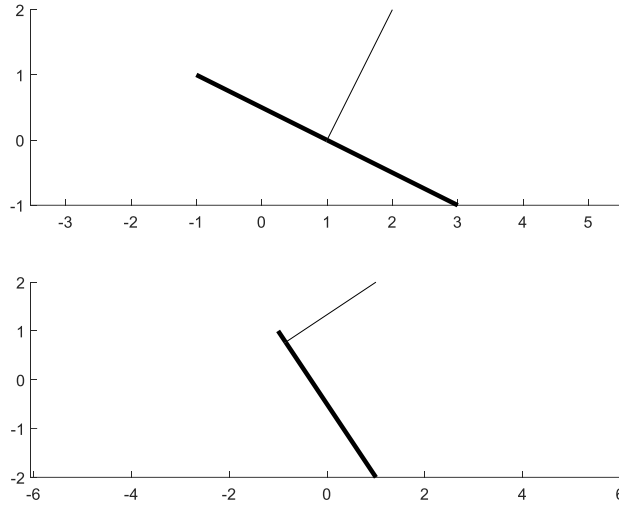


Figure 7.46. Simple test of perpendicular distance between a point and a line.
The new line (thin line) intersects the original line (thick line) at a 90 degree angle in both cases.

Another method for finding the second equation (that the point lies on the line prescribed by the segment arc) can be proposed. This second method would guarantee the area of the triangle made by connecting the new point to both terminals of the segment is 0. A graphic depiction of this new triangle is offered:

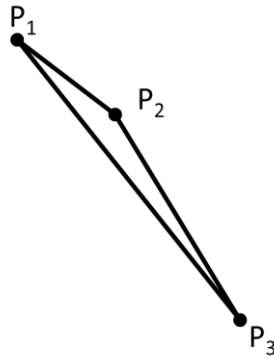


Figure 7.47. Schematic representation of evaluation if new bifurcation point (P_2) lies along the vector $\langle P_1 - P_3 \rangle$ by evaluating the area of the triangle made by 3 points $\langle P_1, P_2, P_3 \rangle$.

The area is defined using a cross product of the base vectors of the triangle:

$$\begin{aligned}
v_1 &= p_2 - p_1; & v_2 &= p_3 - p_1 \\
0 = A &= \frac{|v_1 \times v_2|}{2}
\end{aligned} \tag{7.31}$$

Which implies:

$$|v_1 \times v_2| = 0 \tag{7.32}$$

Which needs to be written in terms of the unknowns:

$$v_1 \times v_2 = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} v_1(y)v_2(z) - v_1(z)v_2(y) \\ v_1(z)v_2(x) - v_1(x)v_2(z) \\ v_1(x)v_2(y) - v_1(y)v_2(x) \end{bmatrix} \tag{7.33}$$

Which simplifies to:

$$\begin{aligned}
v_1 \times v_2 &= \begin{bmatrix} (p_2 - p_1)(y)v_2(z) - (p_2 - p_1)(z)v_2(y) \\ (p_2 - p_1)(z)v_2(x) - (p_2 - p_1)(x)v_2(z) \\ (p_2 - p_1)(x)v_2(y) - (p_2 - p_1)(y)v_2(x) \end{bmatrix} \\
|v_1 \times v_2| &= \sqrt{[(p_2 - p_1)(y)v_2(z) - (p_2 - p_1)(z)v_2(y)]^2 + \\
&\quad [(p_2 - p_1)(z)v_2(x) - (p_2 - p_1)(x)v_2(z)]^2 + \\
&\quad [(p_2 - p_1)(x)v_2(y) - (p_2 - p_1)(y)v_2(x)]^2}
\end{aligned} \tag{7.34}$$

Because this value is 0, the square root can be simplified:

$$\begin{aligned} 0 = & [(p_2 - p_1)(y)v_2(z) - (p_2 - p_1)(z)v_2(y)]^2 \\ & + [(p_2 - p_1)(z)v_2(x) - (p_2 - p_1)(x)v_2(z)]^2 \\ & + [(p_2 - p_1)(x)v_2(y) - (p_2 - p_1)(y)v_2(x)]^2 \end{aligned} \quad (7.35)$$

Which is a nonlinear set of equations, so it is not a desirable approach. This method will not be used.

7.6 Appendix F: Modifying tortuosity of a spline

In many instances, it is necessary to add or remove tortuosity of a spline. This section describes how to achieve both of these modifications.

7.6.1 Adding tortuosity

In order to increase tortuosity in a spline (a Bezier spline of 3rd order will be used for this explanation), the control points can be moved and the curve will have a longer or shorter arc length.

The control point in a network is assigned using the topological connectivity:

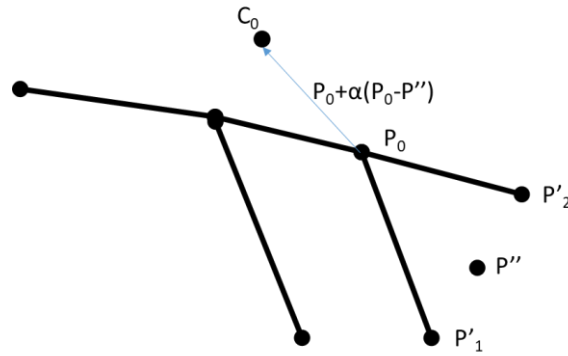


Figure 7.48. Schematic representation of variables necessary for adding tortuosity to a splined segment.

C_0 is the control point for the P_0 terminal point. P'_1 and P'_2 are points connected to splines sharing the terminal point P_0 .

The calculation for point C_0 are as follows:

$$p'' = \frac{1}{nCon} \sum_{i=1}^{nCon} p'_i \quad (7.36)$$

$$c_0 = p_0 + \alpha(p_0 - p'') \quad (7.37)$$

Where p_0 is the point coordinate of the current terminal point of a face, c_0 is the control point of interest (the control point on the current spline closest to this point), $nCon$ is the number of connected faces to the current point, p'_i denotes downstream points connected to i^{th} bifurcation point, and p'' is the average of all p'_i coordinates. This process is repeated for both control points in the spline.

The length of a Bezier curve, however, is not linearly proportional to the tortuosity. With no analytic expression for the length of a Bezier spline of 3rd order as a function of control point location, a fixed-point iterative scheme was developed that iteratively adds length and checks tortuosity until the length of the spline matches the desired length. This scheme begins at the original length (l_0) and adds tortuosity until a final length (αl_0) is achieved. A pseudocode is offered:

Table 7.12. Pseudocode for adding tortuosity to a group of splines

```

1.  FUNCTION addTortuosity(aSynSplinedNwk, anEmpSplinedNWK:ooSplinedTubeMesh);
2.  FOR iBin := 1 TO nBins DO
3.     $\alpha := \text{matchCDF}(\text{sortedSynNwkLL}, \text{sortedSynNwkLL}, iBin)$ 
4.    FOR iFace := 1 TO nFacesInBin DO
5.      connectedFaces := getConnectedFaces(iFace);
6.      WHILE  $\text{abs}(\text{aSynSplinedNWK.getL}(iFace) - \alpha * \text{SynNwkLL}[iFace]) > \text{tolerance}$  DO
7.        stretch(aSynSplinedNWK[iFace].c0,  $\alpha$ , connectedFaces, c0, c1)
8.      ENDWHILE
9.    ENDFOR
10. ENDFOR
11. END

```

Table 7.13. Pseudocode for making a spline longer

1. FUNCTION *Stretch*(aSpline, alfa, lNew, vNew);
2. controlPointVector= scale(alfa, vNew);
3. aSpline.C0 = add(controlPointVector, aSpline .P0);
4. aSpline.updateLengthIteratively;
5. END

The *matchCDF* algorithm is described in Section 3.2.2. The stretch procedure involves moving the control points (C_0 and C_1) following Equation (7.37). Note, both control points in a curve must be stretched, so the total length matches the desired value.

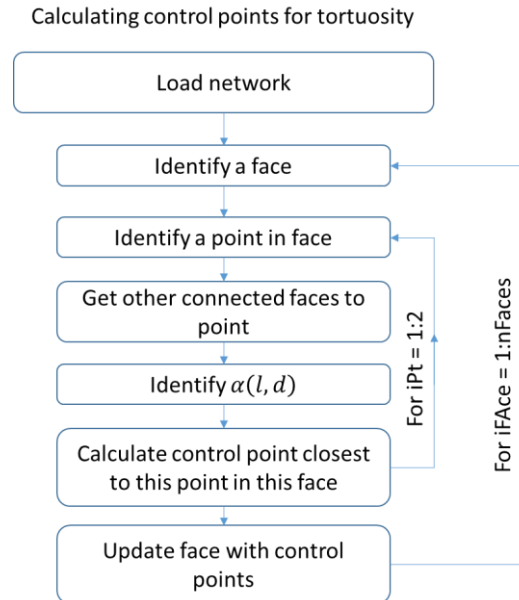


Figure 7.49. Workflow diagram for adding tortuosity to a splined network.

This workflow overwrites the old control points with new control points. The terminal points of each spline remain unchanged.

7.6.2 Reducing Tortuosity

The closed network from Section 7.1.4 gives a network of splines each with two control points and two terminal points. The closure stage of growth, however, can create vessels with highly

acute angles, which creates splined segments with tortuosity outside of the empirical range (1 - 8) as visualized in Figure 7.50. This arises from an enforcement of minimum/maximum length conditions outright, while an acute angle constraint is relaxed if no suitable candidate segment can be found for a given terminal.

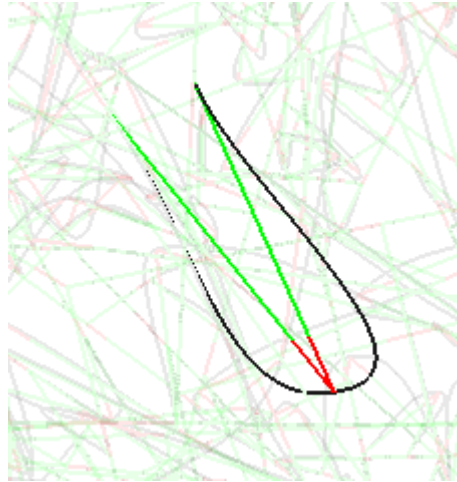


Figure 7.50. Visualization of a closure segment that connects arteries to veins through an acute angle.

This segment has a high degree of tortuosity.

These segments and terminal points are synthesized, and not reconstructed, allowing the freedom to modify the location of the control and terminal points in the spline as needed. The tortuosity can be reduced by simply reducing the amount of linear length the segment has. This can be accomplished by shrinking the spline towards the center between terminal points of the segment:

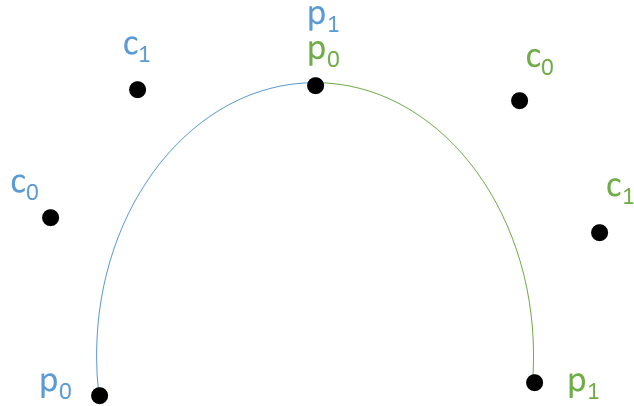


Figure 7.51. The control and terminal points to two bezier curves of 3rd order linked together to make a splined segment.

The first curve is in blue and the second curve is in green. Note the p_1 point for the first curve is the same as the p_0 point for the second curve.

The midpoint between the first and last terminal points in the spline dictate an origin point (pt_c). From this point, vectors can be defined for all intermediate points (all except p_0 from the first curve and p_1 from the second curve in this case). These vectors can be scaled by a constant value to generate a smaller curve:

Another option is to set these segments to a tortuosity value of 1 which helps offset the lack of extremely short segments that appear in empirical data (near the image-threshold range). The option to reduce the tortuosity to 1 has shown the best results among all values for *maximum tortuosity*.

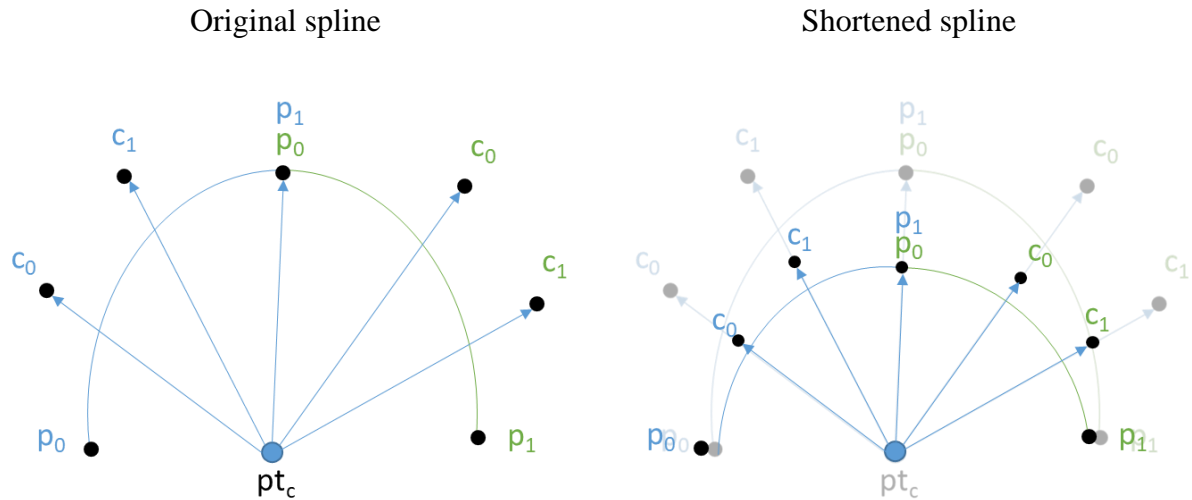


Figure 7.52. The control and terminal points to two bezier curves of 3rd order linked together to make a splined segment. The first curve is in blue and the second curve is in green. Note, the spline is shortened by simply shortening the vectors between pt_c and the control/terminal points of each Bezier curve.

An application to the case study in Figure 7.50 can be seen for full and reduced tortuosity:

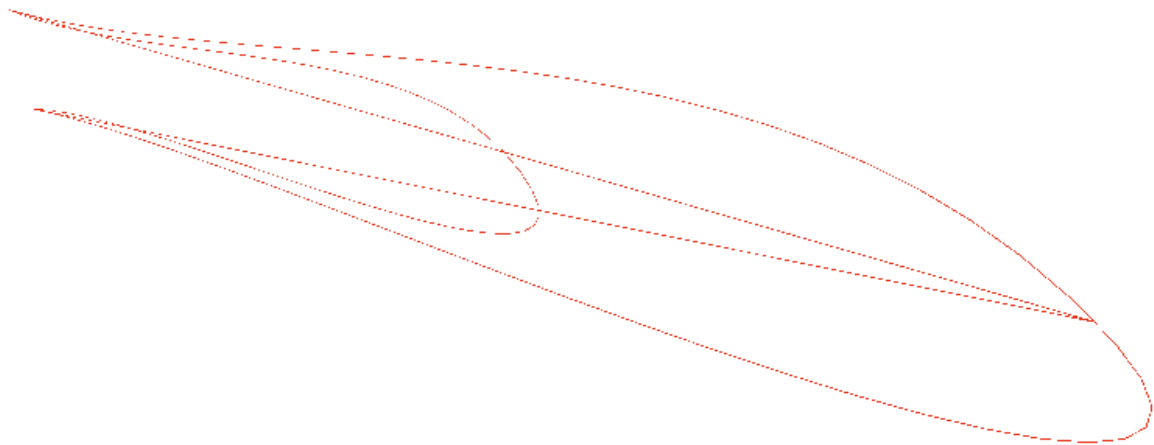


Figure 7.53. The finished product of reducing tortuosity changes the long, smooth curve into the shorter smooth curve.

7.7 Appendix G: Using a quadratic model with y-offset of L_0 to match CDFs

Another model for matching two CDF functions is to assume a quadratic model where the vertical offset is the starting length. This model can be formulated as:

$$\tau = \tau_0 + \tau_1 \cdot l \quad (7.38)$$

$$l_n = l(1 + \tau) \quad (7.39)$$

By combining (7.38) and (7.39) into a single equation:

$$l' = l(1 + \tau_0 + \tau_1 \cdot l) \quad (7.40)$$

Which takes the form of:

$$y = c + bx + ax^2 \quad (7.41)$$

As:

$$l' = l + \tau_0 l + \tau_1 \cdot l^2 \quad (7.42)$$

This problem can be solved for unknowns τ_0 and τ_1 using linear regression. Here, the coefficients a and b are unknown (corresponding to τ_0 and τ_1 respectively) and the coefficient c is already known to be l_0 . Here, l is the starting length (length of the synthetic network prior to

adding tortuosity), and l' is the final length (corresponding length at the same bin edge in the empirical network). To provide two pieces of information and solve for the unique solution for the unknowns, the start and end of the bin can be sampled from the sorted array for empirical and synthetic networks (for example, l_0 to l'_0 and l_1 to l'_1 for start and end bin lengths for synthetic and empirical network lengths, respectively):

$$\begin{aligned} l'_0 &= l_0^{emp} & l'_1 &= l_1^{emp} \\ l_0 &= l_0^{syn} & l_1 &= l_1^{syn} \end{aligned} \tag{7.43}$$

When using these 2 data points, the unknown coefficients in (7.40) constitute a perfectly defined system to be solved:

$$\begin{bmatrix} l_0 & l_0^2 \\ l_1 & l_1^2 \end{bmatrix} \begin{pmatrix} \tau_0 \\ \tau_1 \end{pmatrix} = \begin{pmatrix} l'_0 - l_0 \\ l'_1 - l_1 \end{pmatrix} \tag{7.44}$$

This will give a quadratic fit to align τ_0 and τ_1 . The algorithm can then apply this tortuosity (Equation (7.38)) to the faces with indices in the sortedIdxA with a preexisting procedure.

Case study 1: small networks. Implementation of the quadratic model of CDF fitting on two sample datasets renders excellent alignment of the CDFs:

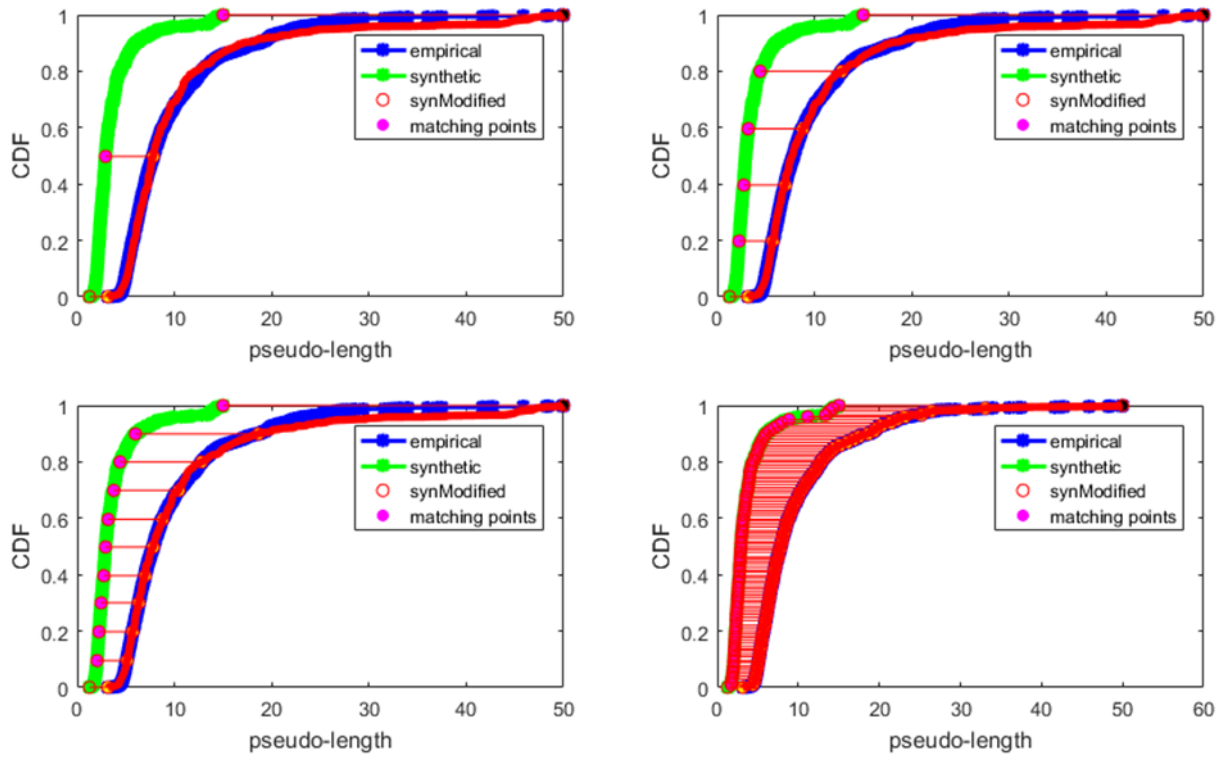


Figure 7.54. Completely different networks (generated with similar algorithms but at different scales and different densities) matched with 2, 5, 10, and 100 bins. Green corresponds to initial synthetic network, blue is empirical network, and red is aligned synthetic (after tortuosity added). In this case, the results are meaningless, but it is a good test for proof of concept.

Case study 2: synthetic KFI. Implementation of the quadratic model of CDF fitting on a full KF dataset renders excellent alignment of the CDFs:

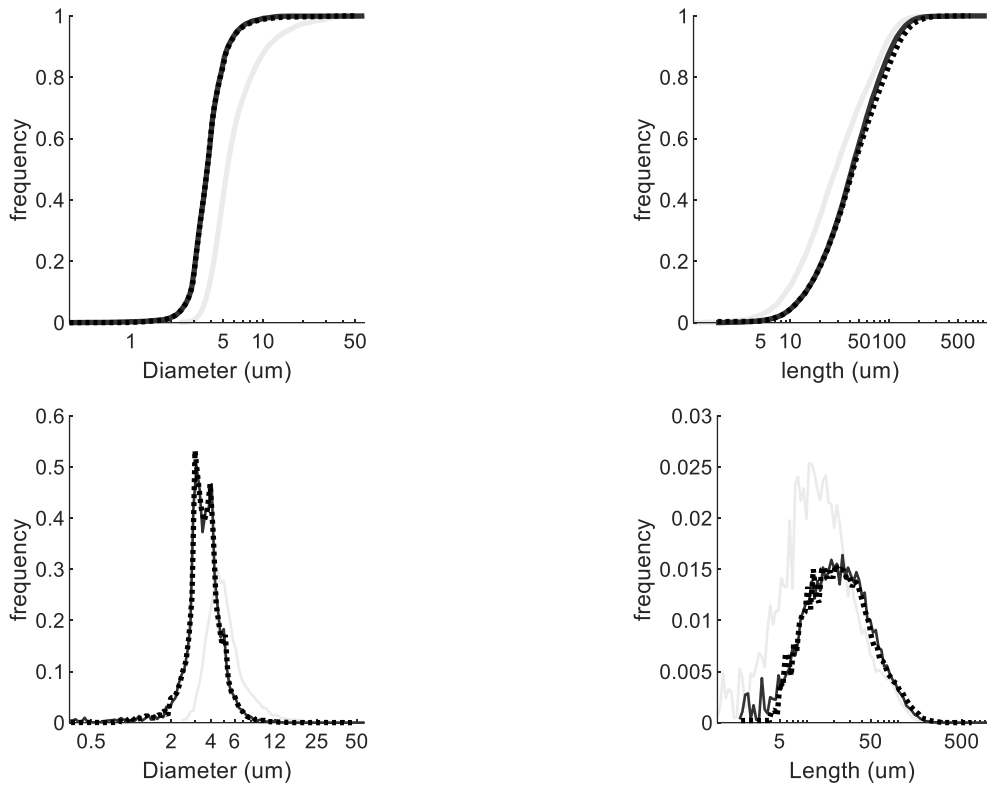


Figure 7.55. Matching CDFs between EKF1.1 and SKF1.101.

Statistical cumulative density functions and probability density functions for automatically matching raw synthetic (grey lines) to empirical (black dots) statistics. Note the shifted empirical (black line) aligns nicely with the empirical datasets.

7.7.1 Relevant codes

7.7.1.1 Code for bifurcation volume optimization evaluation

The code provided here visualizes optimal point of bifurcation location in 2D.

```

% % GH 2/16/2019 -- a procedure to generate statistics figures for growth paper
function showOptimizedBifurcations
close all;
global MY,
MY = 3.5e-3;

ptCoordMx = [-1.0001 1.0001; 0.00001 0.49999; 3.0001 -1.0001; 2.0001 2.0001]; %modify 2nd point
plotOptimizationSpace(ptCoordMx,100);
ptCoordMx = [-1.0001 3.0001; 0.00001 0.49999; 2.0001 -1.0001; 2.0001 1.0001]; %modify 2nd point
plotOptimizationSpace(ptCoordMx,15);
end

function plotOptimizationSpace(ptCoordMx,d0)
figure,
disp([' ']);
p2Orig = ptCoordMx(2,:);
xRange = -2:0.1:3;
yRange = -1:0.1:3;
totalVolume = zeros(length(xRange),length(yRange)); %allocate memory
subplot(2,2,1), hold on,
[L1,L2,L3] = getL(ptCoordMx);
plotNetwork(ptCoordMx,calculateDiameter(L1,L2,L3),'k'); title('original layout');
xlim([min(xRange),max(xRange)]); ylim([min(yRange),max(yRange)]);
disp('original tree'); makeReport(L1,L2,L3,calculateDiameter(L1,L2,L3)); disp(' ');

% % use optimized diameter
for x = 1:length(xRange)
for y = 1:length(yRange)
ptCoordMx(2,:) = [xRange(x) yRange(y)]; [L1,L2,L3] = getL(ptCoordMx);
dia = calculateDiameter(L1,L2,L3); totalVolume(x,y) = V(L1,dia(1)) + V(L2,dia(2))
+ V(L3,dia(3));
end
end
subplot(2,2,2), hold on, contour(xRange,yRange,log(totalVolume'),100),
title('volume of tree after optimization within space'); xlim([min(xRange),max(xRange)]);
ylim([min(yRange),max(yRange)]);

subplot(2,2,3), surf(xRange,yRange,totalVolume','edgecolor','none'), set(gca,'zScale','log')
title('volume of tree'); xlim([min(xRange),max(xRange)]);
ylim([min(yRange),max(yRange)]);

subplot(2,2,2); hold on,
val = min(min(totalVolume));
ptCoordMx(2,:) = [xRange(x) yRange(y)]; [x,y] = find(totalVolume == val);
finalDia = calculateDiameter(L1,L2,L3); [L1,L2,L3] = getL(ptCoordMx);
xlim([min(xRange),max(xRange)]);
ylim([min(yRange),max(yRange)]);
scatter([ptCoordMx(2,1) p2Orig(1,1)], [ptCoordMx(2,2) p2Orig(1,2)], 'k','filled');
quiver(p2Orig(1),p2Orig(2), ptCoordMx(2,1)-p2Orig(1),ptCoordMx(2,2)-p2Orig(2), 'k', 'markersize',10, 'maxheadsiz',20, 'linewidth',1);
subplot(2,2,4), hold on, plotNetwork(ptCoordMx,finalDia,'k'), title('optimized tree');
disp('optimized tree'); makeReport(L1,L2,L3,finalDia);
dia = recSetRootRadius(L1,L2,L3,10); makeReport(L1,L2,L3,dia);

end

function makeReport(L1,L2,L3,dia)
disp(' '); [p,f] = simulate(dia,L1,L2,L3); disp(['root dia: ' num2str(dia(1))]);
disp(['inlet flow: ' num2str(f(1))]); disp(['left branch sum: ' num2str(f(2))]);
disp(['right branch sum: ' num2str(f(3))]);
alfa = getAlfa(dia,[L1,L2,L3]); alfaT = alfa(1) + alfa(2)*alfa(3)/(alfa(2)+alfa(3));
disp(['total alfa: ' num2str(alfaT)]);
VTotal = V(L1,dia(1)) + V(L2,dia(2)) + V(L3,dia(3));
disp(['total Volume: ' num2str(VTotal)]);
end

function alfaV = getAlfa(dia,L)
alfaV = [r(L(1))/(dia(1)^4); r(L(2))/(dia(2)^4); r(L(3))/(dia(3)^4)];
end

function [p,f,alfa] = simulate(dia,L1,L2,L3)
d1 = dia(1); d2 = dia(2); d3 = dia(3);
alfa = getAlfa(dia,[L1,L2,L3]);
A(1,:) = [1 0 0 0]; b(1) = 1;

```

```

A(2,:) = [1/alfa(1) -1/alfa(1)-1/alfa(2)-1/alfa(3) 1/alfa(2) 1/alfa(3)];
A(3,:) = [0 0 1 0];      b(3) = 0;
A(4,:) = [0 0 0 1];      b(4) = 0;
p = A\b';
f = [(p(1)-p(2))/alfa(1); (p(2)-p(3))/alfa(2); (p(2)-p(4))/alfa(3)];
end

function [L1,L2,L3] = getL(ptCoordMx)
L1 = norm(ptCoordMx(1,:)-ptCoordMx(2,:));
L2 = norm(ptCoordMx(2,:)-ptCoordMx(3,:));
L3 = norm(ptCoordMx(2,:)-ptCoordMx(4,:));
end

function dia = calculateDiameter(L1,L2,L3)
q0 = 10000; %ml/min - total tree flow
delP = 1; %mmHg
kappa = 3;
m = (r(L2)/r(L3))^(1/4);
beta2 = (1 + m^(-kappa))^(1/kappa); %can be reduced in this problem
beta3 = (1 + m^kappa)^(-1/kappa); %can be reduced in this problem
rTot = r(L1) + (beta3^4/r(L3) + beta2^4/r(L2))^(-1);
d1 = (rTot*q0/delP)^(1/4);
d2 = beta2*d1;      d3 = beta3*d1;      dia = [d1;d2;d3];

% d1^4/rTot
% alfaT1 = r(L1)/(d1^4) + ((r(L2)/(d2^4))*(r(L3)/(d3^4)))/(r(L2)/(d2^4)+r(L3)/(d3^4));
% alfa = getAlfa(dia,[L1,L2,L3]);
% alfaT2 = alfa(1) + alfa(2)*alfa(3)/(alfa(2)+alfa(3));
% [p,f,alfa] = simulate(dia,L1,L2,L3)
% alfaT3 = alfa(1) + alfa(2)*alfa(3)/(alfa(2)+alfa(3));
end

function dia = recSetRootRadius(L1,L2,L3,d0)
q0 = 10000; %ml/min - total tree flow
delP = 1; %mmHg
kappa = 3;
m = (r(L2)/r(L3))^(1/4);
beta2 = (1 + m^(-kappa))^(1/kappa); %can be reduced in this problem
beta3 = (1 + m^kappa)^(-1/kappa); %can be reduced in this problem
d1 = d0;      d2 = beta2*d1;      d3 = beta3*d1;      dia = [d1;d2;d3];
end

function R = r(L),      global MY,      R = (128*MY*L/(pi));      end

function plotNetwork(ptCoordMx,dia,clr)
faceMx = [1 2; 2 3; 2 4];
for i=1:length(faceMx)
    plot([ptCoordMx(faceMx(i,1),1)      ptCoordMx(faceMx(i,2),1)], [ptCoordMx(faceMx(i,1),2)
ptCoordMx(faceMx(i,2),2)],...
        clr,'linewidth',dia(i)/2);
end
end

function vol = V(L,dia)
vol = pi*(dia/2)^2*L;
end

```

7.7.1.2 Code for tetrahedral sample generation evaluation

The code provided in this section evaluates polyhedron sampling bias.


```

% % GHartung 2/26/2019 -- a program to develop and test a tetrahedral
% sample generator
function testPolyhedralMirroring
close all
global faceMx
faceMx = [1 2 3; 1 3 4; 3 4 2; 2 4 1]; i = 1;

% % % case study 3
figure, hold on
ptCoordMx = [1 1 1; 1 1.5 3; 2 2 2; 2 1 1]; newPt = [0.6 0.5 0.6];

%to get statistics
figure, hold on
for i = 1:1000
    movePtIntoTetrahedralAndPlotV3(ptCoordMx,rand(1,3), {},i,false)
end
end

function movePtIntoTetrahedralAndPlotV2(ptCoordMx,L,names,i, useLgd)
v1 = ptCoordMx(2,:) - ptCoordMx(1,:); v2 = ptCoordMx(3,:) - ptCoordMx(1,:);
v3 = ptCoordMx(4,:) - ptCoordMx(1,:);
newPt = ptCoordMx(1,:) + L(1)*v1 + L(2)*v2 + L(3)*v3;
v4 = newPt - ptCoordMx(1,:);
scatter3(newPt(1),newPt(2),newPt(3),'oc','filled'); names{i} = 'original pt'; i = i+1;
drawTetrahedral(ptCoordMx); %put last so that legend makes sense
plotBaseVectors(v1,v2,v3,ptCoordMx(1,:))

if sum(L) > 1
    r = rand(1,1); L = r/sum(L)*L;
end
newPt = ptCoordMx(1,:) + L(1)*v1 + L(2)*v2 + L(3)*v3;
scatter3(newPt(1),newPt(2),newPt(3),'om','filled'); names{i} = 'new pt'; i = i+1;
drawTetrahedral(ptCoordMx); %put last so that legend makes sense
if useLgd, legend(names{:}); end
end

function movePtIntoTetrahedralAndPlotV3(ptCoordMx,L,names,i, useLgd)
v1 = ptCoordMx(2,:) - ptCoordMx(1,:); v2 = ptCoordMx(3,:) - ptCoordMx(1,:);
v3 = ptCoordMx(4,:) - ptCoordMx(1,:);
newPt = ptCoordMx(1,:) + L(1)*v1 + L(2)*v2 + L(3)*v3;
v4 = newPt - ptCoordMx(1,:);
scatter3(newPt(1),newPt(2),newPt(3),'oc','filled'); names{i} = 'original pt'; i = i+1;
drawTetrahedral(ptCoordMx); %put last so that legend makes sense
plotBaseVectors(v1,v2,v3,ptCoordMx(1,:))

if sum(L) > 1
    r = rand(1,1); L = 1/(2*sum(L))*L;
end
newPt = ptCoordMx(1,:) + L(1)*v1 + L(2)*v2 + L(3)*v3;
scatter3(newPt(1),newPt(2),newPt(3),'om','filled'); names{i} = 'new pt'; i = i+1;
drawTetrahedral(ptCoordMx); %put last so that legend makes sense
if useLgd, legend(names{:}); end
end

function movePtIntoTetrahedralAndPlot(ptCoordMx,scalingValues,names,i, useLgd)
v1 = ptCoordMx(2,:) - ptCoordMx(1,:); v2 = ptCoordMx(3,:) - ptCoordMx(1,:);
v3 = ptCoordMx(4,:) - ptCoordMx(1,:);
newPt = ptCoordMx(1,:) + scalingValues(1)*v1 + scalingValues(2)*v2 + scalingValues(3)*v3;
v4 = newPt - ptCoordMx(1,:);
scatter3(newPt(1),newPt(2),newPt(3),'oc','filled'); names{i} = 'original pt'; i = i+1;
L = getL(ptCoordMx,newPt); Lx = L(1); Ly = L(2); Lz = L(3);
drawTetrahedral(ptCoordMx); %put last so that legend makes sense
plotBaseVectors(v1,v2,v3,ptCoordMx(1,:))

if Lx+Ly+Lz > 1
    temp1 = Lx; temp2 = Ly; temp3 = Lz;
    Lx = norm([temp1,temp2,temp3]) - norm([temp2,temp3]);
    Ly = norm([temp1,temp2,temp3]) - norm([temp1,temp3]);
    Lz = norm([temp1,temp2,temp3]) - norm([temp1,temp2]);
end

```

```

end
    newPt = ptCoordMx(1,:) + Lx*v1 + Ly*v2 + Lz*v3;
    L = getL(ptCoordMx,newPt);
    scatter3(newPt(1),newPt(2),newPt(3),'om','filled');    names{i} = 'new pt';    i = i+1;
drawTetrahedral(ptCoordMx);    %put last so that legend makes sense
if useLgd, legend(names{:});    end
end

function L = getL(ptCoordMx,newPt)
v1 = ptCoordMx(2,:) - ptCoordMx(1,:);    v2 = ptCoordMx(3,:) - ptCoordMx(1,:);
v3 = ptCoordMx(4,:) - ptCoordMx(1,:);    %plotBaseVectors(v1,v2,v3,ptCoordMx(1,:));
v4 = newPt - ptCoordMx(1,:);
A = [v1(1) v2(1) v3(1); v1(2) v2(2) v3(2); v1(3) v2(3) v3(3)];
b = [v4(1); v4(2); v4(3)];
L = A\b;
end

function plotBaseVectors(v1,v2,v3,aSrcPt)
quiver3(aSrcPt(1),aSrcPt(2),aSrcPt(3), v1(1),v1(2),v1(3),'m','markersize',10,'linewidth',1);
quiver3(aSrcPt(1),aSrcPt(2),aSrcPt(3), v2(1),v2(2),v2(3),'m','markersize',10,'linewidth',1);
quiver3(aSrcPt(1),aSrcPt(2),aSrcPt(3), v3(1),v3(2),v3(3),'m','markersize',10,'linewidth',1);
end

function distanceToFace = getDistanceToFace(ptCoordMx,newPt,aFaceIdx)
N = getFaceNormal(aFaceIdx,ptCoordMx);
faceCenter = getFaceCenter(ptCoordMx,aFaceIdx);
V = newPt-faceCenter;
distanceToFace = norm(V) * (N*V') / (norm(V) * norm(N));
end

function faceN = getFaceNormal(closestFace,ptCoordMx)
global faceMx
pt1 = ptCoordMx(faceMx(closestFace,1),:);    pt2 = ptCoordMx(faceMx(closestFace,2),:);
pt3 = ptCoordMx(faceMx(closestFace,3),:);
u = pt2 - pt1;    v = pt3 - pt1;
faceN = cross(u,v);
end

function x = cross(a,b)
x(1)=a(2)*b(3)-a(3)*b(2);
x(2)=-a(1)*b(3)-a(3)*b(1);
x(3)=a(1)*b(2)-a(2)*b(1);
end

function center = getFaceCenter(ptCoordMx,faceIdx)
global faceMx
pt1 = ptCoordMx(faceMx(faceIdx,1),:);
pt2 = ptCoordMx(faceMx(faceIdx,2),:);
pt3 = ptCoordMx(faceMx(faceIdx,3),:);
center = (pt1+pt2+pt3)./3;
end

function result = getClosestFace(ptCoordMx,aPt)
global faceMx
for i = 1:4
    faceCenter(i,:) = getFaceCenter(ptCoordMx,i);
    length(i) = norm(faceCenter(i,:) - aPt);
end
[val, result] = min(length);
scatter3(faceCenter(result,1),faceCenter(result,2),faceCenter(result,3),'or')
end

function drawTetrahedral(ptCoordMx)
global faceMx
for i = 1:4
    pt1 = ptCoordMx(faceMx(i,1),:);
    pt2 = ptCoordMx(faceMx(i,2),:);
    pt3 = ptCoordMx(faceMx(i,3),:);

```

```

    plot3([pt1(1) pt2(1) pt3(1) pt1(1)], [pt1(2) pt2(2) pt3(2) pt1(2)], [pt1(3) pt2(3) pt3(3)
    pt1(3)], '-ok');
end
view(125,45)
end

```

7.7.1.3 Code for intersecting a point with a line at a 90 degree angle in Matlab

```

% % GH 2/16/2019 -- a procedure to generate statistics figures for growth paper
function plotReverseStrahler
close all;
ptCoordMx = [-1.0001 1.0001; 0.00001 0.49999; 3.0001 -1.0001; 2.0001 2.0001]; %modify 2nd point
p2 = findP2(ptCoordMx); ptCoordMx(2,:) = [p2(1) p2(2)];
subplot(2,1,1), hold on, plotNetwork(ptCoordMx, [5;5;1], 'k'), axis equal
ptCoordMx = [-1.0001 1.0001; 0.00001 0.49999; 1.0001 -2.0001; 1.0001 2.0001]; %modify 2nd point
p2 = findP2(ptCoordMx); ptCoordMx(2,:) = [p2(1) p2(2)];
subplot(2,1,2), hold on, plotNetwork(ptCoordMx, [5;5;1], 'k'), axis equal
end

function p2 = findP2(ptCoordMx)
x1 = ptCoordMx(1,1); x3 = ptCoordMx(3,1); x4 = ptCoordMx(4,1);
y1 = ptCoordMx(1,2); y3 = ptCoordMx(3,2); y4 = ptCoordMx(4,2);
xx = [x1 1; x3 1]\[y1;y3]; mm = xx(1); bb=xx(2); %linear fit of first line
b = [(x4*x3-x1*x4)+(y4*y3-y1*y4); bb];
A = [(x3-x1) (y3-y1); -mm 1];
p2 = A\b;
end

function R = r(L), global MY, R = (128*MY*L/(pi)); end

function plotNetwork(ptCoordMx,dia,clr)
faceMx = [1 2; 2 3; 2 4];
for i=1:length(faceMx)
    plot([ptCoordMx(faceMx(i,1),1) ptCoordMx(faceMx(i,2),1)], [ptCoordMx(faceMx(i,1),2)
    ptCoordMx(faceMx(i,2),2)], ...
    clr, 'linewidth', dia(i)/2);
end
ylim([min(ptCoordMx(:,2)) max(ptCoordMx(:,2))]); xlim([min(ptCoordMx(:,1))
max(ptCoordMx(:,1))]);
end

```

7.7.1.4 CDF matching case study and code in Matlab

Case study. The length spectra of two simple networks were generated for comparing bin size and automated CDF matching accuracy. The networks show reasonable alignment even with only 2 bins, however increasing the number of bins is computationally inexpensive and results in an excellent agreement (Figure 7.56).

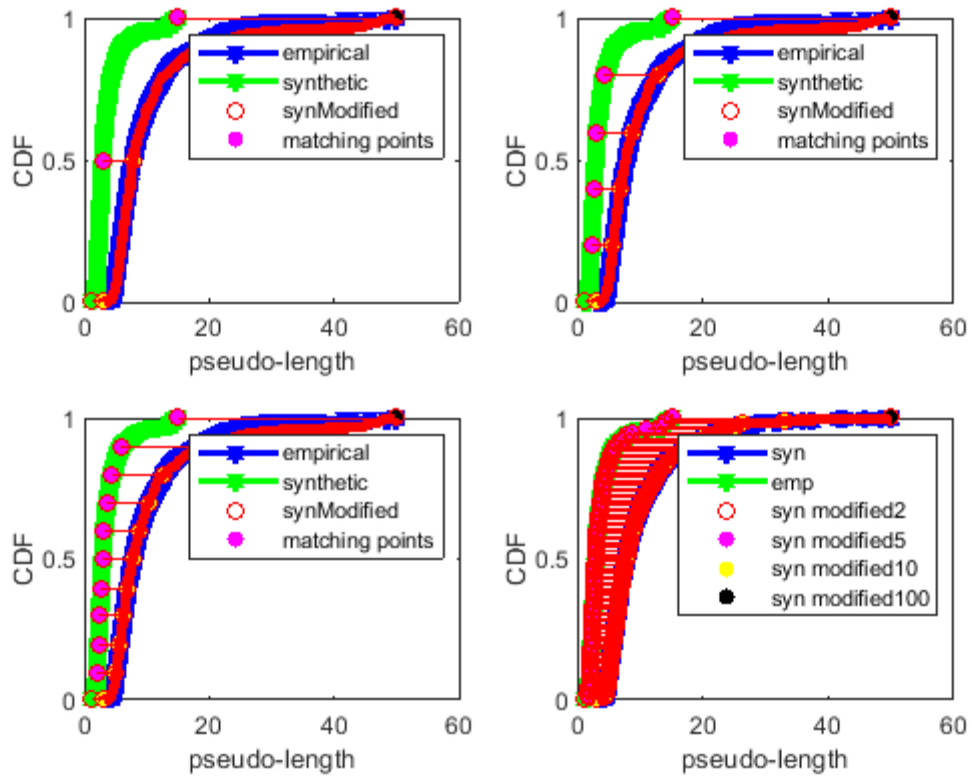


Figure 7.56. Visualization of automated CDF matching using case study 1. Completely different networks (generated with similar algorithms but at different scales and different densities) matched with 2, 5, 10, and 100 bins. Green corresponds to initial synthetic network, blue is empirical network, and red is aligned synthetic (after tortuosity added).

The code in this section shows CDF matching using Matlab.

```

% % GH 2/20/2019 -- a program that tests automated CDF matching
function testCDFMatching
%% enough fucking around -- time to use a real fucking cdf
close all
[xEmp,cdfEmp] = getCDF(1); %long
[xSyn,cdfSyn] = getCDF2; %short

figure,
subplot(2,2,1), matchCDFAndPlot(2,cdfEmp,cdfSyn,xEmp,xSyn)
subplot(2,2,2), matchCDFAndPlot(5,cdfEmp,cdfSyn,xEmp,xSyn)
subplot(2,2,3), matchCDFAndPlot(10,cdfEmp,cdfSyn,xEmp,xSyn)
subplot(2,2,4), matchCDFAndPlot(100,cdfEmp,cdfSyn,xEmp,xSyn)
legend('syn','emp','syn modified2','syn modified5','syn modified10','syn modified100')

figure,
subplot(2,2,1), matchCDFAndPlotV2(2,cdfEmp,cdfSyn,xEmp,xSyn)
subplot(2,2,2), matchCDFAndPlotV2(5,cdfEmp,cdfSyn,xEmp,xSyn)
subplot(2,2,3), matchCDFAndPlotV2(10,cdfEmp,cdfSyn,xEmp,xSyn)
subplot(2,2,4), matchCDFAndPlotV2(100,cdfEmp,cdfSyn,xEmp,xSyn)
legend('syn','emp','syn modified2','syn modified5','syn modified10','syn modified100')
end

% % % % % % % % version 2
function matchCDFAndPlotV2(nBins,cdfEmp,cdfSyn,xEmp,xSyn)
nFacesEmp = length(xEmp); nFacesSyn = length(xSyn);
synSortedLengthIdx = xSyn; empSortedLengthIdx = xEmp;
plot(xEmp,cdfEmp,'-b','linewidth',2), hold on, plot(xSyn,cdfSyn,'-g','linewidth',2)

binEdgesSyn = getBinEdges(cdfSyn,nBins);
binEdgesEmp = getBinEdges(cdfEmp,nBins);

for i = 1:nBins
    startIdxSyn = binEdgesSyn(i); endIdxSyn = binEdgesSyn(i+1);
    startIdxEmp = binEdgesEmp(i); endIdxEmp = binEdgesEmp(i+1);
    l0 = synSortedLengthIdx(startIdxSyn); l1 = synSortedLengthIdx(endIdxSyn);
    l0Prime = empSortedLengthIdx(startIdxEmp); l1Prime = empSortedLengthIdx(endIdxEmp);
    A = [l0 l1]; b = [l0Prime l1Prime];
    scatter([l0], [cdfSyn(startIdxSyn)], 'r');
scatter([l1], [cdfSyn(endIdxSyn)], 'm', 'filled')
scatter([l0Prime], [cdfEmp(startIdxEmp)], 'y', 'filled');
scatter([l1Prime], [cdfEmp(endIdxEmp)], 'k', 'filled')
    alfaV = A\b; alfa0 = alfaV(1); alfa1 = alfaV(2);
    for j = startIdxSyn:endIdxSyn
        x3(j) = alfa0 + alfa1*synSortedLengthIdx(j);
    end
end
plot(x3,cdfSyn,'-r','linewidth',4),
% show the direction of shift
for i = 1:nBins
    startIdx = binEdgesSyn(i); endIdx = binEdgesSyn(i+1);
    plot([synSortedLengthIdx(startIdx) x3(startIdx)], [cdfSyn(startIdx) cdfSyn(startIdx)], '-or')
    %show line from start to finish for startIdx
    plot([synSortedLengthIdx(endIdx) x3(endIdx)], [cdfSyn(endIdx) cdfSyn(endIdx)], '-or')
    %show line from start to finish for startIdx
end
legend('empirical','synthetic','synModified','matching points');
xlabel('pseudo-length'); ylabel('CDF')
end

% right now, assumes same length array for each
function matchCDFAndPlot(nBins,cdfEmp,cdfSyn,xEmp,xSyn)
nFacesEmp = length(xEmp); nFacesSyn = length(xSyn);
synSortedLengthIdx = xSyn; empSortedLengthIdx = xEmp;
plot(xEmp,cdfEmp,'-b','linewidth',2), hold on, plot(xSyn,cdfSyn,'-g','linewidth',2)

binEdgesSyn = getBinEdges(cdfSyn,nBins);
binEdgesEmp = getBinEdges(cdfEmp,nBins);

for i = 1:nBins

```


5.00479732300000	5.01175284500000	5.01642347900000	5.01757976200000	5.01869644700000
5.01922974300000	5.02600011400000	5.02766389600000	5.03415917700000	5.04564407600000
5.04610396700000	5.04662426600000	5.04680842800000	5.04775797000000	5.04991856400000
5.05040052300000	5.05149930800000	5.06376128900000	5.06643715300000	5.07187898800000
5.07224621100000	5.07312186800000	5.07766504300000	5.07771706900000	5.07845071700000
5.08066962500000	5.08093006400000	5.08710972800000	5.09066797900000	5.09751469900000
5.10176410000000	5.10215343600000	5.10241904400000	5.10430217100000	5.10589383500000
5.11001159500000	5.11832660100000	5.12313956400000	5.12470282100000	5.12473571600000
5.12922163500000	5.13158071900000	5.13191188700000	5.13495806700000	5.13934496300000
5.14201931700000	5.14237693200000	5.14615232900000	5.14623386300000	5.14798333000000
5.14811361100000	5.15656803600000	5.16146201600000	5.16442166000000	5.16512823500000
5.17066122800000	5.17070330000000	5.17342830100000	5.17376605900000	5.17438315000000
5.17553127000000	5.17977454900000	5.18286918000000	5.18673000600000	5.18738217600000
5.19321876500000	5.19791405900000	5.20073778400000	5.20415027400000	5.20588731300000
5.21891992400000	5.22654100200000	5.22804484300000	5.22879990500000	5.23319680400000
5.23513860900000	5.23736151600000	5.23809042000000	5.23965638300000	5.24047063200000
5.24222419100000	5.24363200300000	5.24414813000000	5.25307822000000	5.25611961300000
5.25642195000000	5.26027819000000	5.26156610800000	5.26335247200000	5.26714402900000
5.26861688600000	5.27482259900000	5.27743560600000	5.27915391800000	5.28024835800000
5.28202928900000	5.28901300600000	5.28953283000000	5.29017180800000	5.29281735800000
5.29465423300000	5.29855017900000	5.29926630700000	5.30210804400000	5.30322683000000
5.30589709600000	5.30639128500000	5.30819948000000	5.30926878500000	5.31373600900000
5.32114441600000	5.32336581400000	5.32624630700000	5.32649632200000	5.32847449100000
5.33043323000000	5.33242890200000	5.33951846100000	5.34278754500000	5.34360261900000
5.34424488800000	5.34576435700000	5.34753985200000	5.34854353200000	5.34902723700000
5.35295006100000	5.35346769100000	5.35796173600000	5.36380377600000	5.36550722400000
5.36993563600000	5.38407672000000	5.39465889200000	5.39494185700000	5.39711285100000
5.40730383500000	5.41002018700000	5.41241109400000	5.42008839100000	5.42062559000000
5.42070890100000	5.42177700000000	5.42235726200000	5.42644999000000	5.42665400600000
5.43000249400000	5.43352190300000	5.43562883800000	5.43617708700000	5.45286876000000
5.45564279700000	5.46487629200000	5.46635368100000	5.46711478900000	5.46756408900000
5.47138928900000	5.47242641900000	5.47447147500000	5.47601550200000	5.48746682400000
5.49242889900000	5.49327456400000	5.49398540200000	5.49615814300000	5.49624287300000
5.49662358300000	5.49998837700000	5.50346127300000	5.50384140200000	5.50674744100000
5.50741332000000	5.50936956600000	5.51019189900000	5.51028727400000	5.51289722000000
5.51760331300000	5.51894151800000	5.52131162200000	5.52442866600000	5.52797723900000
5.52842757300000	5.53096637400000	5.53321408400000	5.53464370600000	5.53924585900000
5.53970065800000	5.54369817500000	5.54475659500000	5.54742092500000	5.54869172800000
5.54912394300000	5.54914640800000	5.55196340300000	5.55727336300000	5.56486767400000
5.56552920800000	5.56628841400000	5.56913204800000	5.56960817200000	5.57002284100000
5.57126421700000	5.57758606700000	5.57781877900000	5.58770505900000	5.58886333200000
5.59280735200000	5.59609741100000	5.60110792900000	5.60338267600000	5.60870237300000
5.60941804600000	5.61178439400000	5.61386045700000	5.61625937900000	5.63189507200000
5.63489737500000	5.63649541300000	5.63667269100000	5.63803019200000	5.63897747300000
5.64140040100000	5.64728096000000	5.64741697700000	5.65665273700000	5.66042375700000
5.66710276400000	5.66730527900000	5.66927013100000	5.67126338400000	5.67416029100000
5.68263464100000	5.68275315500000	5.68737890300000	5.68775797200000	5.69815304000000
5.70155050300000	5.70426497200000	5.70749577200000	5.70926497400000	5.71523810200000
5.71713289200000	5.72126705100000	5.72332666000000	5.72721241400000	5.73402387000000
5.73859473700000	5.74062414800000	5.75552774400000	5.75981936400000	5.76322380000000
5.76439305500000	5.76993625800000	5.77189475000000	5.77605572600000	5.77930831700000
5.78005834400000	5.78387271400000	5.78926511500000	5.79546648100000	5.79850801300000
5.79863925800000	5.80132582100000	5.80408384500000	5.80418611700000	5.80994087600000
5.82035727800000	5.82460937400000	5.82625987500000	5.82809991600000	5.82878368100000
5.83763495200000	5.83960614300000	5.84616190400000	5.85051088700000	5.85629488600000
5.85930114100000	5.86309880300000	5.86921313700000	5.87704610700000	5.87719622300000
5.87786373700000	5.88006786900000	5.88033094300000	5.88063008500000	5.88166227000000
5.88706339700000	5.88720314900000	5.88946675800000	5.89337380600000	5.89590274100000
5.90412730100000	5.90415017600000	5.90494278200000	5.90539289500000	5.90606366100000
5.90832199600000	5.91029121200000	5.91293371900000	5.91352744000000	5.91883600200000
5.92263526800000	5.92418918500000	5.92435807500000	5.92563741100000	5.92594468300000
5.93020450400000	5.93266421100000	5.93703021400000	5.94089999200000	5.94577564100000
5.95241698600000	5.95330951600000	5.95644359100000	5.96208955800000	5.96704361200000
5.96948852300000	5.97317319100000	5.97374677900000	5.97594084700000	5.97712387900000
5.98548796400000	5.98841665400000	5.98960892500000	5.99161162700000	6.00105568400000
6.00265715600000	6.00650270100000	6.00859208600000	6.01237173300000	6.02655029700000
6.02775549000000	6.03025988400000	6.03087880100000	6.03686685300000	6.04575802700000
6.05268936200000	6.05516284000000	6.05834486000000	6.05900628100000	6.06018337300000
6.06209054000000	6.06339178700000	6.06354179600000	6.07110048500000	6.07296678300000

6.07564621000000	6.07917886100000	6.08495973200000	6.08514749000000	6.08861719300000
6.09102434400000	6.09740177600000	6.09992226600000	6.10076577600000	6.10186427900000
6.10223126000000	6.10441423700000	6.10477912700000	6.10570890000000	6.10711833300000
6.11243544700000	6.11469767800000	6.12172252900000	6.13274735900000	6.13500561100000
6.14056827200000	6.14081199800000	6.14245328800000	6.14384565000000	6.14828220100000
6.14829966500000	6.15268329000000	6.15628807000000	6.16059138000000	6.16087743200000
6.16428454500000	6.16445227200000	6.16490791700000	6.16752042300000	6.17247230600000
6.17727201300000	6.17736737700000	6.18554453900000	6.18784301000000	6.19548063600000
6.19773621200000	6.19798440000000	6.20043467400000	6.20242939800000	6.20711120600000
6.21070527500000	6.21250942400000	6.21362235800000	6.22107281700000	6.22538582800000
6.22885492500000	6.23318921700000	6.23331745600000	6.23365921700000	6.23867954400000
6.23952282300000	6.24098185300000	6.24338086000000	6.24424563200000	6.24427406600000
6.24437710600000	6.24488366100000	6.24691219700000	6.24840920500000	6.25357185700000
6.26123983900000	6.26455183500000	6.26515846800000	6.27295611700000	6.27331072600000
6.27623161500000	6.27723098300000	6.28232093300000	6.29037771700000	6.29372748800000
6.29380851300000	6.29606938000000	6.29771707800000	6.30172138700000	6.30453722100000
6.30737865500000	6.30747609700000	6.31367360200000	6.32181782400000	6.32406323500000
6.32919707600000	6.32948179900000	6.33076235600000	6.33232855300000	6.33835000400000
6.33887671700000	6.34094824400000	6.34560874600000	6.34749534800000	6.35276368200000
6.35895037100000	6.35921245700000	6.36084060400000	6.36112083000000	6.36392218000000
6.36953164700000	6.37016941700000	6.37374935000000	6.37581939500000	6.37659939600000
6.38057485000000	6.38201589100000	6.38429147300000	6.38553637600000	6.38614237900000
6.38678252000000	6.39127262200000	6.39329207300000	6.39712290100000	6.40162468700000
6.40835894600000	6.40974204500000	6.42695235200000	6.43032448100000	6.43039934700000
6.44075959600000	6.44244705300000	6.44655037800000	6.45154529500000	6.45315345800000
6.45604870200000	6.45783021800000	6.45811327100000	6.46461459600000	6.46684958900000
6.46781263800000	6.46893830100000	6.47443903400000	6.47494974000000	6.47645200300000
6.47913124500000	6.48224781600000	6.48828745700000	6.49034007900000	6.49431023200000
6.49995355200000	6.50109563500000	6.50157079500000	6.50412163300000	6.50520236400000
6.50906298300000	6.50951451800000	6.50981147200000	6.51925912200000	6.52397801500000
6.52588460600000	6.52797338300000	6.52916107500000	6.53032557400000	6.53406381000000
6.53473082100000	6.53498457900000	6.53760482300000	6.53760482300000	6.53801134600000
6.54060347800000	6.54492007500000	6.54611170800000	6.54658202500000	6.54680508700000
6.55186488400000	6.56069206600000	6.56457787900000	6.56611178500000	6.56927822100000
6.56999278900000	6.57777366900000	6.58245020500000	6.59283141700000	6.59386052900000
6.59738974200000	6.60102507600000	6.60534132500000	6.61023994300000	6.61546012600000
6.61849625100000	6.61932565200000	6.61980279100000	6.62451616700000	6.62545084300000
6.62600087600000	6.62627352000000	6.62807398800000	6.62828053800000	6.63683533700000
6.63775468800000	6.64290104600000	6.65083579400000	6.65210549000000	6.66468780400000
6.66525953200000	6.66615301400000	6.66682476900000	6.66695113100000	6.67077405200000
6.67380626500000	6.67523507900000	6.67543575700000	6.68070795800000	6.68205879500000
6.68829326500000	6.69538842000000	6.69912186900000	6.70004193200000	6.70270458300000
6.70988119400000	6.72515793800000	6.72823227400000	6.72979308300000	6.73195994100000
6.73618425300000	6.73998948800000	6.74385584000000	6.74715411100000	6.75260619800000
6.75290111200000	6.75354358400000	6.76018437900000	6.76439399600000	6.76576556500000
6.77210090200000	6.77268972600000	6.77333401200000	6.77360412000000	6.77865894600000
6.80035176800000	6.80363386000000	6.80486285600000	6.80620118900000	6.81097561600000
6.81725772200000	6.82873142900000	6.82895217500000	6.83971310600000	6.84266431600000
6.84497481600000	6.84811176000000	6.85356791800000	6.85404848000000	6.85569206200000
6.85724640900000	6.86373985100000	6.87037197100000	6.87607566700000	6.87927313500000
6.88344227600000	6.88379788400000	6.89095646600000	6.89654851500000	6.89721120500000
6.90212072400000	6.90261392800000	6.90886269600000	6.91191146300000	6.91716564100000
6.91717006800000	6.92414748100000	6.92655041000000	6.93115071500000	6.93446808800000
6.93816932500000	6.94337123100000	6.94362173900000	6.95088279800000	6.95288582000000
6.95382113800000	6.95719750600000	6.96032087400000	6.96046391600000	6.96058273500000
6.96112502600000	6.96310912500000	6.96324353100000	6.96956566900000	6.97239347500000
6.98293482400000	6.98397492000000	6.98873380400000	6.98912321800000	6.99186244200000
6.99306519900000	6.99340670200000	6.99514337600000	6.99572839900000	6.99658890800000
6.99967362900000	7.00392184400000	7.00498360600000	7.00757652700000	7.01499212400000
7.01717236600000	7.01756819000000	7.01923867700000	7.01942443600000	7.02220316000000
7.02342847000000	7.02448800200000	7.02547257800000	7.03705446800000	7.03758018600000
7.04267913300000	7.04785169500000	7.05667248700000	7.05995451200000	7.06293452900000
7.06379228500000	7.06510624100000	7.07019010000000	7.07794783800000	7.07935457100000
7.08804676700000	7.10025394900000	7.10064413500000	7.10138313000000	7.10166130400000
7.10459672000000	7.10922508400000	7.11277814500000	7.11548128500000	7.11872184900000
7.11902084700000	7.12879666800000	7.13311958500000	7.13427720400000	7.13917353900000
7.14463097700000	7.14481472500000	7.14612236100000	7.15330308300000	7.15868696300000
7.16604101600000	7.16808211400000	7.17139125900000	7.17227506000000	7.17733167700000
7.17995996200000	7.18424955700000	7.18919745800000	7.19070922800000	7.19145021000000

7.19502832100000	7.19924410100000	7.20793417400000	7.20977848500000	7.21235191900000
7.22101992200000	7.22609734100000	7.22799584500000	7.23566846000000	7.23841724700000
7.24862395400000	7.24957970800000	7.25176988600000	7.25235385700000	7.25445539900000
7.25899207600000	7.25987912900000	7.26119817200000	7.26953706800000	7.27384731600000
7.27491228300000	7.27831822600000	7.27868057000000	7.28373838000000	7.28594181200000
7.29349708500000	7.29464661200000	7.29478373700000	7.29994530800000	7.30274224700000
7.30959897900000	7.31358820600000	7.32834830900000	7.33110988400000	7.34008153600000
7.34216730900000	7.34581810500000	7.34997964600000	7.35251677400000	7.35256052600000
7.35475267900000	7.35985133300000	7.38662973500000	7.39925716600000	7.40102802700000
7.40412948800000	7.40717819000000	7.41646901100000	7.41987446000000	7.42054581500000
7.42254333100000	7.43034498200000	7.43589894900000	7.44442966600000	7.44764680100000
7.45055770400000	7.45543718100000	7.45914909600000	7.46838291600000	7.47317775400000
7.47513586200000	7.48116922000000	7.48162678600000	7.48398891200000	7.48461119000000
7.48469396200000	7.48850205700000	7.49590792700000	7.50310605400000	7.50524314300000
7.50804289500000	7.51113118400000	7.51637956000000	7.51768983700000	7.52248475800000
7.52389202750000	7.53079144000000	7.53893904700000	7.54579627700000	7.54986401800000
7.55291076100000	7.56001150900000	7.56640963700000	7.56852058300000	7.56927690700000
7.56986058700000	7.57249934700000	7.57494577100000	7.58780242100000	7.59317327200000
7.60169867700000	7.60899284900000	7.61615274800000	7.62661930000000	7.62695763600000
7.62969770400000	7.63693576800000	7.64098542000000	7.64223613700000	7.64463444600000
7.64606188900000	7.64626404800000	7.65791392700000	7.65929935200000	7.66042852200000
7.66184146500000	7.66334426500000	7.67096087900000	7.67314698600000	7.68680225800000
7.68705749100000	7.70094933600000	7.71630405800000	7.72260909400000	7.73499556500000
7.74520310300000	7.74654471500000	7.75108541100000	7.75197463700000	7.75389647700000
7.75787783900000	7.76571620200000	7.77041515900000	7.78134517000000	7.78667769600000
7.78819461800000	7.78858267300000	7.79412474500000	7.79826935900000	7.80154897000000
7.80251549000000	7.80396765100000	7.80951129500000	7.81826364400000	7.82430056400000
7.82883480500000	7.83591564200000	7.83993662000000	7.84192717600000	7.84257444600000
7.84276976500000	7.84381460400000	7.84840053600000	7.87118311800000	7.87120465200000
7.87739770600000	7.88135564400000	7.88412446200000	7.88876392000000	7.88879772800000
7.88882662300000	7.89029499200000	7.89569948400000	7.89657218900000	7.90698767400000
7.91065318600000	7.91746540500000	7.92032043700000	7.92039428700000	7.92480034900000
7.92899612900000	7.93171388100000	7.93260401600000	7.93904823300000	7.94373363600000
7.94672838100000	7.96127607800000	7.96714070800000	7.96841837400000	7.97019085300000
7.97180407300000	7.97621142400000	7.97818138900000	7.99167639400000	7.99346469500000
7.99601436800000	8.00974601900000	8.01234705700000	8.01646400900000	8.01715950700000
8.01763626600000	8.01819580800000	8.02432073600000	8.02493050800000	8.02683610100000
8.02764983300000	8.02789076400000	8.02893570300000	8.03429222500000	8.03634248900000
8.04382682800000	8.04643042800000	8.04718172600000	8.05330930600000	8.05386866100000
8.05423356920000	8.05492010400000	8.05554310900000	8.05587498500000	8.05608994200000
8.05729809700000	8.06769787000000	8.07478315900000	8.10691341400000	8.11098471400000
8.11141213100000	8.11532789400000	8.11685650600000	8.13215241700000	8.15913105600000
8.17159578800000	8.17517374300000	8.17656414100000	8.17870017300000	8.17961739200000
8.18021830000000	8.18522544200000	8.18579209300000	8.18602873000000	8.20014872800000
8.20244740900000	8.20348552300000	8.20363131100000	8.20508751700000	8.21767992200000
8.22813631900000	8.23660962800000	8.23797541000000	8.24341576500000	8.24387530400000
8.24901372500000	8.25437749300000	8.26243266400000	8.26595261900000	8.26869191100000
8.27780078300000	8.28014949900000	8.28441338800000	8.29175839000000	8.30894675100000
8.31206186300000	8.31506568900000	8.32256169900000	8.32323239900000	8.32356507000000
8.32775732800000	8.32842927900000	8.34171464000000	8.35679865300000	8.36089222100000
8.36481572800000	8.36572302800000	8.36684283900000	8.36707763000000	8.36714269300000
8.37914424900000	8.38041582800000	8.38313381100000	8.38488776500000	8.39095307600000
8.40535169600000	8.40979217900000	8.41270068200000	8.42204750600000	8.43080075900000
8.43532720500000	8.43534978600000	8.43730030500000	8.44120022500000	8.44717278400000
8.45774445000000	8.45970756000000	8.46285949400000	8.46992296700000	8.48431455600000
8.48629453900000	8.48805869200000	8.48908789300000	8.49087517400000	8.49743196400000
8.49994489600000	8.50812749400000	8.50999702400000	8.51620831700000	8.52315134500000
8.52400587200000	8.53865518100000	8.53936569600000	8.55284234700000	8.55896784900000
8.55906723500000	8.55964086100000	8.56715023200000	8.56813601400000	8.57031313900000
8.57139049700000	8.57263921800000	8.58282001200000	8.58450307500000	8.58641859000000
8.58721463600000	8.58974691600000	8.59264398800000	8.59373627300000	8.59452730900000
8.60306175600000	8.60932165200000	8.61648669000000	8.61724311800000	8.62263302000000
8.63241998100000	8.63504079200000	8.63823396000000	8.65300883600000	8.65486854100000
8.66712313600000	8.68779170600000	8.69052618100000	8.70573741600000	8.70645962200000
8.70743337000000	8.72820143900000	8.73922781600000	8.74057606300000	8.75089645300000
8.75374295900000	8.75788743200000	8.76048996800000	8.76173619800000	8.77123214000000
8.79963998800000	8.80156282700000	8.80728742100000	8.81857338900000	8.84826473600000
8.86673532500000	8.87007347500000	8.88095562100000	8.88467236800000	8.89640622600000
8.90089601900000	8.90530991800000	8.90727896300000	8.91455172300000	8.92084727800000

8.92732196600000	8.93363144200000	8.94202479100000	8.94374158900000	8.95569209800000
8.95697281400000	8.96393603700000	8.96630119500000	8.97302429600000	8.97634915800000
8.98685279500000	8.98756237300000	9.00357603200000	9.01717577300000	9.04094294500000
9.04545250600000	9.05436419000000	9.05469788100000	9.05882298000000	9.06296030500000
9.06819543300000	9.07676103000000	9.08373299400000	9.09525330100000	9.10073208800000
9.10179198600000	9.10229909700000	9.10971431200000	9.11245339000000	9.11284917200000
9.11312160300000	9.11645646800000	9.12014569300000	9.12385613100000	9.13040572100000
9.13181299800000	9.13900238900000	9.14182908200000	9.14854374500000	9.16307593300000
9.16639255600000	9.17124225800000	9.18406772100000	9.18797364300000	9.20301986200000
9.22675354900000	9.23727668600000	9.23817864800000	9.25186442700000	9.25629110700000
9.25985121400000	9.26098714000000	9.26599660600000	9.26687206800000	9.29207675400000
9.30273999200000	9.33799543900000	9.34762585200000	9.35689838000000	9.36904075300000
9.37791731100000	9.38614296200000	9.38623421900000	9.38791657800000	9.38908007600000
9.39474513200000	9.40706659400000	9.40797119600000	9.43000582600000	9.43467062800000
9.43783947300000	9.45007336000000	9.45559102200000	9.45613397300000	9.45941978300000
9.46163702600000	9.46602499000000	9.47090345800000	9.47471044200000	9.47711855200000
9.48248939200000	9.49923645600000	9.53853970900000	9.55775120200000	9.56712474200000
9.57379754200000	9.60326332200000	9.61564730900000	9.63007894400000	9.63150032300000
9.65548861700000	9.66534994300000	9.66706122000000	9.66978612300000	9.67072324600000
9.67293174400000	9.67356901700000	9.68080825400000	9.70552524000000	9.72956363700000
9.76286781500000	9.77127517500000	9.77442239800000	9.77827499100000	9.77915680600000
9.78332410000000	9.78838704400000	9.79201828600000	9.79440997500000	9.79525808300000
9.79857524700000	9.80867989300000	9.82285233400000	9.83136821200000	9.83593168600000
9.83714492400000	9.85177781100000	9.85858334300000	9.89070317400000	9.89478136700000
9.89866337600000	9.89872831200000	9.91415681400000	9.91940874900000	9.92668131500000
9.92963659400000	9.93782046600000	9.94627352300000	9.95116291000000	9.95172415300000
9.95544729700000	9.95595782600000	9.96899477400000	9.98859534200000	10.00099645000000
10.00512001000000	10.01101684000000	10.03189996000000	10.03824784000000	10.04814461000000
10.05418799000000	10.05585602000000	10.06036406000000	10.06826287000000	10.09019515000000
10.09167816000000	10.09431526000000	10.09708647000000	10.10242427000000	10.11226008000000
10.12727997000000	10.13456325000000	10.13636726000000	10.14064808000000	10.14616578000000
10.14623738000000	10.16910779000000	10.17220421000000	10.17829660000000	10.20143904000000
10.21184165000000	10.25532063000000	10.27482517000000	10.27627070000000	10.28733285000000
10.29436933000000	10.29863038000000	10.31179230000000	10.33119376000000	10.34760436000000
10.37541684000000	10.38588757000000	10.40176079000000	10.40400435000000	10.41372977000000
10.43123762000000	10.43464709000000	10.43953701000000	10.44240107000000	10.46041838000000
10.46718736000000	10.48931910000000	10.49281508000000	10.49805699000000	10.50207218000000
10.51037435000000	10.55918626000000	10.56542626000000	10.57233789000000	10.57599138000000
10.57856269000000	10.58050427000000	10.58526816000000	10.60786779000000	10.61070142000000
10.61807392000000	10.63649540000000	10.64401371000000	10.64750902000000	10.65650193000000
10.68256844000000	10.69902913000000	10.70257505000000	10.70317604000000	10.71126914000000
10.71189442000000	10.73254460000000	10.76994111000000	10.77022315000000	10.78744034000000
10.79264599000000	10.79549277000000	10.81002106000000	10.81727276000000	10.83074234000000
10.83928465000000	10.88903768000000	10.91428725000000	10.91911822000000	10.92285470000000
10.92736033000000	10.93082172000000	10.94550416000000	10.98451866000000	10.98605642000000
10.99933438000000	11.00449701000000	11.03258498000000	11.08049103000000	11.08457451000000
11.09313855000000	11.09438513000000	11.09538262000000	11.10328976000000	11.10501136000000
11.10541223000000	11.11661391000000	11.13834270000000	11.14365135000000	11.15087909000000
11.16470225000000	11.18075845000000	11.18726517000000	11.19669400000000	11.20109625000000
11.21284015000000	11.21673232000000	11.23104393000000	11.25345267000000	11.27203713000000
11.27975571000000	11.32114228000000	11.32524907000000	11.34238193000000	11.34257049000000
11.36857496000000	11.37590313000000	11.37844297000000	11.38098563000000	11.38557622000000
11.39140717000000	11.40538080000000	11.42411429000000	11.42919298000000	11.45850918000000
11.47121439000000	11.47328482000000	11.48501268000000	11.48679187000000	11.49914517000000
11.50110624000000	11.51530241000000	11.53184953000000	11.56649083000000	11.56966856000000
11.57708218000000	11.60909032000000	11.61742172000000	11.62229905000000	11.65329834000000
11.65940804000000	11.67540169000000	11.69055260000000	11.69426854000000	11.70304949000000
11.73662278000000	11.74694820000000	11.76391305000000	11.77418513000000	11.78135692000000
11.81400449000000	11.82879427000000	11.86634686000000	11.86793429000000	11.87326929000000
11.87610647000000	11.88167648000000	11.90119653000000	11.90946862000000	11.92631169000000
11.99499377000000	12.02702076000000	12.04487954000000	12.05508222000000	12.05525369000000
12.06787191000000	12.07787508000000	12.09401712000000	12.09507268000000	12.09732113000000
12.11195911000000	12.11282736000000	12.13195668000000	12.13269466000000	12.17527279000000
12.17698052000000	12.21006025000000	12.21621920000000	12.21851671000000	12.23305855000000
12.24346856000000	12.24854523000000	12.29144225000000	12.29903586000000	12.30589676000000
12.31459062000000	12.33915635000000	12.34151083000000	12.34785627000000	12.35933084000000
12.36859900000000	12.36943724000000	12.37415111000000	12.37572013000000	12.37617287000000
12.38371874000000	12.38584552000000	12.39659970000000	12.41738434000000	12.42061377000000
12.42920562000000	12.45249406000000	12.45375759000000	12.49036831000000	12.49042891000000

12.5252937700000	12.5255377200000	12.5379498100000	12.5419651600000	12.5855424300000
12.5962926500000	12.5990860100000	12.6031577700000	12.6163892000000	12.6396196800000
12.6438633900000	12.6460223300000	12.6651832200000	12.6780125100000	12.6848442700000
12.6875830700000	12.7026491000000	12.7182820300000	12.7343091200000	12.7509412100000
12.7694832200000	12.7738684200000	12.7801828000000	12.7804139000000	12.7864545100000
12.7945399600000	12.7967915000000	12.7985885600000	12.8082343900000	12.8333064500000
12.8748969400000	12.8870508400000	12.9145091200000	12.9182689900000	12.9243982300000
12.9649910800000	12.9799186300000	12.9939266900000	12.9945204400000	13.0086451900000
13.0127124600000	13.0285575800000	13.0395633900000	13.0501333800000	13.0821895500000
13.1076949400000	13.1349538000000	13.1383774400000	13.1392548900000	13.1463478800000
13.1488106100000	13.1493206600000	13.1539787100000	13.1709541400000	13.2000996200000
13.2320226600000	13.2436871600000	13.2760001300000	13.3371286900000	13.3643119700000
13.3788902300000	13.3895753000000	13.4213768100000	13.4308047900000	13.4345133500000
13.4687000500000	13.4857340200000	13.4894830500000	13.5027841500000	13.5675898400000
13.5835528600000	13.5882359600000	13.6162917500000	13.6406638900000	13.6482133100000
13.6829525000000	13.6943921600000	13.6975634000000	13.7314980000000	13.7478556800000
13.7480065100000	13.7505301000000	13.7674321400000	13.7851854500000	13.7860050400000
13.7930192600000	13.7994580400000	13.8043469600000	13.8629941400000	13.9148367400000
13.9229062600000	13.9899274200000	13.9982233400000	14.0223564000000	14.0650638100000
14.0888615210000	14.0984529400000	14.1127396100000	14.1268654300000	14.1309432900000
14.1630484200000	14.1698119400000	14.2160447200000	14.2560474900000	14.2710105000000
14.2720560000000	14.3118409300000	14.3237803300000	14.3611379500000	14.3730214300000
14.3757116100000	14.3900865700000	14.4188944400000	14.4360226300000	14.4689297000000
14.4847862900000	14.5316687600000	14.5571304700000	14.5878758600000	14.6154125600000
14.6513144400000	14.6657813000000	14.6757710600000	14.6968637500000	14.7291209900000
14.7304382700000	14.8375252100000	14.8447722600000	14.8549508600000	14.9505178900000
14.9945474900000	15.0157456800000	15.0363974000000	15.1243522700000	15.1268078700000
15.1804374600000	15.1839076400000	15.2230478500000	15.2896637000000	15.3990759300000
15.4442771300000	15.4496080300000	15.4967728800000	15.5290392100000	15.5591799700000
15.6691282100000	15.6773682800000	15.6794036100000	15.6860641500000	15.7591698100000
15.8911740600000	15.9926791100000	16.0228619800000	16.0598395000000	16.0638632300000
16.0663152100000	16.2081911800000	16.2112482400000	16.2345742000000	16.3757793700000
16.3880825800000	16.4195535300000	16.4355350000000	16.5134174000000	16.5329213800000
16.5895427900000	16.6547459800000	16.6585859400000	16.6987383100000	16.7617687400000
16.7987609100000	16.8568485900000	16.8845196800000	16.9499808900000	16.9829532000000
16.9962806900000	17.0420975700000	17.0472981600000	17.0865455600000	17.1993933000000
17.2061769500000	17.2090102600000	17.2528678300000	17.2906944500000	17.3160615600000
17.3520216600000	17.3568249000000	17.4034573500000	17.4045274600000	17.4912017900000
17.4984129200000	17.5420290500000	17.5448868200000	17.5478943200000	17.5808059900000
17.6654629100000	17.7771791700000	17.8965867100000	17.8979597600000	17.9681277900000
18.0012586900000	18.0105018400000	18.1327670200000	18.1751817200000	18.2276345800000
18.3139160100000	18.3392016200000	18.4287033300000	18.4753907000000	18.5729354900000
18.6283169400000	18.6762159500000	18.7594103800000	18.7736788800000	18.8470143000000
18.8942911200000	18.9344428100000	18.9843382200000	18.9895975400000	19.0059583600000
19.0214329600000	19.0531623100000	19.0923493000000	19.1036296300000	19.1294529100000
19.1330077100000	19.1784277300000	19.3120800200000	19.3274322100000	19.3334677100000
19.3621820800000	19.4079642500000	19.4192937100000	19.4471426500000	19.4549381200000
19.4704833700000	19.4732580600000	19.4810469600000	19.5141877600000	19.5422117800000
19.5569242600000	19.6271792700000	19.6358285000000	19.6448880900000	19.6477484900000
19.6816614200000	19.7043219700000	19.7131585400000	19.7241521700000	19.7951877800000
19.7968028000000	19.8020482200000	19.8176104300000	19.8401327100000	19.8781368200000
19.8792298300000	19.8950217300000	19.9204289200000	19.9398863400000	19.9484829600000
20.0256016100000	20.0621643300000	20.1743708900000	20.2097600300000	20.2396165300000
20.2716697300000	20.3925968600000	20.5021506800000	20.5143759200000	20.5561579100000
20.6290231100000	20.6367622700000	20.6842104600000	20.7909748000000	20.8136375300000
20.8946119600000	21.0081067800000	21.0946326800000	21.1426189000000	21.2338378200000
21.2760801800000	21.3519484800000	21.4002161600000	21.4682925800000	21.4952012100000
21.5532981200000	21.6807199400000	21.6837510400000	21.7628703000000	21.7688526500000
21.8383046100000	21.8468855700000	21.8770683900000	21.9364578100000	21.9384582100000
21.9931931200000	22.0353676200000	22.0385692500000	22.0897205500000	22.1266817800000
22.1660825500000	22.1792883400000	22.2088924200000	22.2098440600000	22.2814805100000
22.2868987300000	22.5318184900000	22.5353185500000	22.5889632600000	22.6748623200000
22.8035301700000	22.8709241800000	22.8872914300000	22.9844841300000	22.9849511200000
23.0763943000000	23.1076600400000	23.1527160100000	23.1531752900000	23.4473956300000
23.8072484900000	23.8578840900000	23.9361504200000	24.0122216200000	24.2059412500000
24.2439052300000	24.4217423700000	24.4393587100000	24.5355457700000	24.8924099400000
24.9308892900000	25.0209648300000	25.0215672700000	25.0800839300000	25.1407112200000
25.1515035800000	25.2084549900000	25.2268798300000	25.2308285000000	25.2537310800000
25.2864002700000	25.4031852600000	25.4706311200000	25.5257988600000	25.8071853600000

25.8197001100000	25.8664989300000	25.8774986800000	26.1782555600000	26.2316746900000
26.2628409300000	26.3133465100000	26.4271464600000	26.8371625000000	26.8648572200000
26.9567626500000	27.0196912400000	27.0718617000000	27.1354142800000	27.2274720400000
27.4968353000000	27.6527167900000	28.2757838700000	28.5725343400000	29.3965578300000
29.4130749100000	31.5469206800000	31.9821147900000	32.4035854000000	32.5900642400000
33.0236893800000	33.0438207000000	33.1946815600000	33.9540847600000	33.9785921600000
36.0012540200000	37.0678674900000	37.5049695700000	41.6222704900000	41.6603388200000
42.3568765100000	42.3725722100000	42.7555267800000	45.8217166000000	45.8694732300000
48.2254364300000	48.2447011000000	48.4651680300000	49.8419387300000	49.9867247700000
50];				
val =	[0.000508646999000000	0.00101729399800000	0.00152594099700000	0.00203458799600000
0.00254323499500000	0.00305188199400000	0.00356052899300000	0.00406917599200000	
0.00457782299100000	0.00508646999000000	0.00559511698900000	0.00610376398800000	
0.00661241098700000	0.00712105798600000	0.00762970498500000	0.00813835198400000	
0.00864699898300000	0.00915564598200000	0.00966429298100000	0.0101729399800000	
0.01068158698000000	0.01119023398000000	0.01169888098000000	0.01220752798000000	
0.01271617497000000	0.01322482197000000	0.01373346897000000	0.01424211597000000	
0.01475076297000000	0.01525940997000000	0.01576805697000000	0.01627670397000000	
0.01678535097000000	0.01729399797000000	0.01780264496000000	0.01831129196000000	
0.01881993896000000	0.01932858596000000	0.01983723296000000	0.02034587996000000	
0.02085452696000000	0.02136317396000000	0.02187182096000000	0.02238046796000000	
0.02288911495000000	0.02339776195000000	0.02390640895000000	0.02441505595000000	
0.02492370295000000	0.02543234995000000	0.02594099695000000	0.02644964395000000	
0.02695829095000000	0.02746693795000000	0.02797558494000000	0.02848423194000000	
0.02899287894000000	0.02950152594000000	0.03001017294000000	0.03051881994000000	
0.03102746694000000	0.03153611394000000	0.03204476094000000	0.03255340793000000	
0.03306205493000000	0.03357070193000000	0.03407934893000000	0.03458799593000000	
0.03509664293000000	0.03560528993000000	0.03611393693000000	0.03662258393000000	
0.03713123093000000	0.03763987792000000	0.03814852492000000	0.03865717192000000	
0.03916581892000000	0.03967446592000000	0.04018311292000000	0.04069175992000000	
0.04120040692000000	0.04170905392000000	0.04221770092000000	0.04272634791000000	
0.04323499491000000	0.04374364191000000	0.04425228891000000	0.04476093591000000	
0.04526958291000000	0.04577822991000000	0.04628687691000000	0.04679552391000000	
0.04730417091000000	0.04781281790000000	0.04832146490000000	0.04883011190000000	
0.04933875890000000	0.04984740590000000	0.05035605290000000	0.05086469990000000	
0.05137334690000000	0.05188199390000000	0.05239064090000000	0.05289928789000000	
0.05340793489000000	0.05391658189000000	0.05442522889000000	0.05493387589000000	
0.05544252289000000	0.05595116989000000	0.05645981689000000	0.05696846389000000	
0.05747711089000000	0.05798575788000000	0.05849440488000000	0.05900305188000000	
0.05951169888000000	0.06002034588000000	0.06052899288000000	0.06103763988000000	
0.06154628688000000	0.06205493388000000	0.06256358087000000	0.06307222787000000	
0.06358087487000000	0.06408952187000000	0.06459816887000000	0.06510681587000000	
0.06561546287000000	0.06612410987000000	0.06663275687000000	0.06714140387000000	
0.06765005086000000	0.06815869786000000	0.06866734486000000	0.06917599186000000	
0.06968463886000000	0.07019328586000000	0.07070193286000000	0.07121057986000000	
0.07171922686000000	0.07222787386000000	0.07273652085000000	0.07324516785000000	
0.07375381485000000	0.07426246185000000	0.07477110885000000	0.07527975585000000	
0.07578840285000000	0.07629704985000000	0.07680569685000000	0.07731434385000000	
0.07782299084000000	0.07833163784000000	0.07884028484000000	0.07934893184000000	
0.07985757884000000	0.08036622584000000	0.08087487284000000	0.08138351984000000	
0.08189216684000000	0.08240081384000000	0.08290946083000000	0.08341810783000000	
0.08392675483000000	0.08443540183000000	0.08494404883000000	0.08545269583000000	
0.08596134283000000	0.08646998983000000	0.08697863683000000	0.08748728383000000	
0.08799593082000000	0.08850457782000000	0.08901322482000000	0.08952187182000000	
0.09003051882000000	0.09053916582000000	0.09104781282000000	0.09155645982000000	
0.09206510682000000	0.09257375381000000	0.09308240081000000	0.09359104781000000	
0.09409969481000000	0.09460834181000000	0.09511698881000000	0.09562563581000000	
0.09613428281000000	0.09664292981000000	0.09715157681000000	0.09766022380000000	
0.09816887080000000	0.09867751780000000	0.09918616480000000	0.09969481180000000	0.10020345880000000
0.10071210580000000	0.10122075280000000	0.10172939980000000	0.10223804680000000	0.10274669380000000
0.10325534080000000	0.10376398780000000	0.10427263480000000	0.10478128180000000	0.10528992880000000
0.10579857580000000	0.10630722280000000	0.10681586980000000	0.10732451680000000	0.10783316380000000
0.10834181080000000	0.10885045780000000	0.10935910480000000	0.10986775180000000	0.11037639880000000
0.11088504580000000	0.11139369280000000	0.11190233980000000	0.11241098680000000	0.11291963380000000
0.11342828080000000	0.11393692780000000	0.11444557480000000	0.11495422180000000	0.11546286880000000
0.11597151580000000	0.11648016280000000	0.11698880980000000	0.11749745680000000	0.11800610380000000
0.11851475080000000	0.11902339780000000	0.11953204480000000	0.12004069180000000	0.12054933880000000
0.12105798580000000	0.12156663280000000	0.12207527980000000	0.12258392680000000	0.12309257380000000
0.12360122080000000	0.12410986780000000	0.12461851480000000	0.12512716170000000	0.12563580870000000

0.126144455700000	0.126653102700000	0.127161749700000	0.127670396700000	0.128179043700000
0.128687690700000	0.129196337700000	0.129704984700000	0.130213631700000	0.130722278700000
0.131230925700000	0.131739572700000	0.132248219700000	0.132756866700000	0.133265513700000
0.133774160700000	0.134282807700000	0.134791454700000	0.135300101700000	0.135808748700000
0.136317395700000	0.136826042700000	0.137334689700000	0.137843336700000	0.138351983700000
0.138860630700000	0.139369277700000	0.139877924700000	0.140386571700000	0.140895218700000
0.141403865700000	0.141912512700000	0.142421159700000	0.142929806700000	0.143438453700000
0.143947100700000	0.144455747700000	0.144964394700000	0.145473041700000	0.145981688700000
0.146490335700000	0.146998982700000	0.147507629700000	0.148016276700000	0.148524923700000
0.149033570700000	0.149542217700000	0.150050864700000	0.150559511700000	0.151068158700000
0.151576805700000	0.152085452700000	0.152594099700000	0.153102746700000	0.153611393700000
0.154120040700000	0.154628687700000	0.155137334700000	0.155645981700000	0.156154628700000
0.156663275700000	0.157171922700000	0.157680569700000	0.158189216700000	0.158697863700000
0.159206510700000	0.159715157700000	0.160223804700000	0.160732451700000	0.161241098700000
0.161749745700000	0.162258392700000	0.162767039700000	0.163275686700000	0.163784333700000
0.164292980700000	0.164801627700000	0.165310274700000	0.165818921700000	0.166327568700000
0.166836215700000	0.167344862700000	0.167853509700000	0.168362156700000	0.168870803700000
0.169379450700000	0.169888097700000	0.170396744700000	0.170905391700000	0.171414038700000
0.171922685700000	0.172431332700000	0.172939979700000	0.173448626700000	0.173957273700000
0.174465920700000	0.174974567700000	0.175483214700000	0.175991861600000	0.176500586000000
0.177009155600000	0.177517802600000	0.178026449600000	0.178535096600000	0.179043743600000
0.179552390600000	0.180061037600000	0.180569684600000	0.181078331600000	0.181586978600000
0.182095625600000	0.182604272600000	0.183112919600000	0.183621566600000	0.184130213600000
0.184638860600000	0.185147507600000	0.185656154600000	0.186164801600000	0.186673448600000
0.187182095600000	0.187690742600000	0.188199389600000	0.188708036600000	0.189216683600000
0.189725330600000	0.190233977600000	0.190742624600000	0.191251271600000	0.191759918600000
0.192268565600000	0.192777212600000	0.193285859600000	0.193794506600000	0.194303153600000
0.194811800600000	0.195320447600000	0.195829094600000	0.196337741600000	0.196846388600000
0.197355035600000	0.197863682600000	0.198372329600000	0.198880976600000	0.199389623600000
0.199898270600000	0.200406917600000	0.200915564600000	0.201424211600000	0.201932858600000
0.202441505600000	0.202950152600000	0.203458799600000	0.203967446600000	0.204476093600000
0.204984740600000	0.205493387600000	0.206002034600000	0.206510681600000	0.207019328600000
0.207527975600000	0.208036622600000	0.208545269600000	0.209053916600000	0.209562563600000
0.210071210600000	0.210579857600000	0.211088504600000	0.211597151600000	0.212105798600000
0.212614445600000	0.213123092600000	0.213631739600000	0.214140386600000	0.214649033600000
0.215157680600000	0.215666327600000	0.216174974600000	0.216683621600000	0.217192268600000
0.217700915600000	0.218209562600000	0.218718209600000	0.219226856600000	0.219735503600000
0.220244150600000	0.220752797600000	0.221261444600000	0.221770091600000	0.222278738600000
0.222787385600000	0.223296032600000	0.223804679600000	0.224313326600000	0.224821973600000
0.225330620500000	0.225839267500000	0.226347914500000	0.226856561500000	0.227365208600000
0.227873855500000	0.228382502500000	0.228891149500000	0.229399796500000	0.229908443500000
0.230417090500000	0.230925737500000	0.231434384500000	0.231943031500000	0.232451678500000
0.232960325500000	0.233468972500000	0.233977619500000	0.234486266500000	0.234994913500000
0.235503560500000	0.236012207500000	0.236520854500000	0.237029501500000	0.237538148500000
0.238046795500000	0.238555442500000	0.239064089500000	0.239572736500000	0.240081383500000
0.240590030500000	0.241098677500000	0.241607324500000	0.242115971500000	0.242624618500000
0.243133265500000	0.243641912500000	0.244150559500000	0.244659206500000	0.245167853500000
0.245676500500000	0.246185147500000	0.246693794500000	0.247202441500000	0.247711088500000
0.248219735500000	0.248728382500000	0.249237029500000	0.249745676500000	0.250254323500000
0.250762970500000	0.251271617500000	0.251780264500000	0.252288911500000	0.252797558500000
0.253306205500000	0.253814852500000	0.254323499500000	0.254832146500000	0.255340793500000
0.255849440500000	0.256358087500000	0.256866734500000	0.257375381500000	0.257884028500000
0.258392675500000	0.258901322500000	0.259409969500000	0.259918616500000	0.260427263500000
0.260935910500000	0.261444557500000	0.261953204500000	0.262461851500000	0.262970498500000
0.263479145500000	0.263987792500000	0.264496439500000	0.265005086500000	0.265513733500000
0.266022380500000	0.266531027500000	0.267039674500000	0.267548321500000	0.268056685000000
0.268565615500000	0.269074262500000	0.269582909500000	0.270091556500000	0.270600203500000
0.271108850500000	0.271617497500000	0.272126144500000	0.272634791500000	0.273143438500000
0.273652085500000	0.274160732500000	0.274669379500000	0.275178026400000	0.275686673400000
0.276195320400000	0.276703967400000	0.277212614400000	0.277721261400000	0.278229908400000
0.278738555400000	0.279247202400000	0.279755849400000	0.280264496400000	0.280773143400000
0.281281790400000	0.281790437400000	0.282299084400000	0.282807731400000	0.283316378400000
0.283825025400000	0.284333672400000	0.284842319400000	0.285350966400000	0.285859613400000
0.286368260400000	0.286876907400000	0.287385554400000	0.287894201400000	0.288402848400000
0.288911495400000	0.289420142400000	0.289928789400000	0.290437436400000	0.290946083400000
0.291454730400000	0.291963377400000	0.292472024400000	0.292980671400000	0.293489318400000
0.293997965400000	0.294506612400000	0.295015259400000	0.295523906400000	0.296032553400000
0.296541200400000	0.297049847400000	0.297558494400000	0.298067141400000	0.298575788400000
0.299084435400000	0.299593082400000	0.300101729400000	0.300610376400000	0.301119023400000

0.301627670400000	0.302136317400000	0.302644964400000	0.303153611400000	0.303662258400000
0.304170905400000	0.304679552400000	0.305188199400000	0.305696846400000	0.306205493400000
0.306714140400000	0.307222787400000	0.307731434400000	0.308240081400000	0.308748728400000
0.309257375400000	0.309766022400000	0.310274669400000	0.310783316400000	0.311291963400000
0.311800610400000	0.312309257400000	0.312817904400000	0.313326551400000	0.313835198400000
0.314343845400000	0.314852492400000	0.315361139400000	0.315869786400000	0.316378433400000
0.316887080400000	0.317395727400000	0.317904374400000	0.318413021400000	0.318921668400000
0.319430315400000	0.319938962400000	0.320447609400000	0.320956256400000	0.321464903400000
0.321973550400000	0.322482197400000	0.322990844400000	0.323499491400000	0.324008138400000
0.324516785400000	0.325025432300000	0.325534079300000	0.326042726300000	0.326551373300000
0.327060020300000	0.327568667300000	0.328077314300000	0.328585961300000	0.329094608300000
0.329603255300000	0.330111902300000	0.330620549300000	0.331129196300000	0.331637843300000
0.332146490300000	0.332655137300000	0.333163784300000	0.333672431300000	0.334181078300000
0.334689725300000	0.335198372300000	0.335707019300000	0.336215666300000	0.336724313300000
0.337232960300000	0.337741607300000	0.338250254300000	0.338758901300000	0.339267548300000
0.339776195300000	0.340284842300000	0.340793489300000	0.341302136300000	0.34181078300000
0.342319430300000	0.342828077300000	0.343336724300000	0.343845371300000	0.344354018300000
0.344862665300000	0.345371312300000	0.345879959300000	0.346388606300000	0.346897253300000
0.347405900300000	0.347914547300000	0.348423194300000	0.348931841300000	0.349440488300000
0.349949135300000	0.350457782300000	0.350966429300000	0.351475076300000	0.351983784300000
0.352492370300000	0.353001017300000	0.353509664300000	0.354018311300000	0.354526958300000
0.355035605300000	0.355544252300000	0.356052899300000	0.356561546300000	0.357070193300000
0.357578840300000	0.358087487300000	0.358596134300000	0.359104781300000	0.359613428300000
0.360122075300000	0.360630722300000	0.361139369300000	0.361648016300000	0.362156663300000
0.362665310300000	0.363173957300000	0.363682604300000	0.364191251300000	0.364699898300000
0.365208545300000	0.365717192300000	0.366225839300000	0.366734486300000	0.367243133300000
0.367751780300000	0.368260427300000	0.368769074300000	0.369277721300000	0.369786368300000
0.370295015300000	0.370803662300000	0.371312309300000	0.371820956300000	0.372329603300000
0.372838250300000	0.373346897300000	0.373855544300000	0.374364191300000	0.374872838300000
0.375381485200000	0.375890132200000	0.376398779200000	0.376907426200000	0.377416073200000
0.377924720200000	0.378433367200000	0.378942014200000	0.379450661200000	0.379959308200000
0.380467955200000	0.380976602200000	0.381485249200000	0.381993896200000	0.382502543200000
0.383011190200000	0.383519837200000	0.384028484200000	0.384537131200000	0.385045778200000
0.385554425200000	0.386063072200000	0.386571719200000	0.387080366200000	0.387589013200000
0.388097660200000	0.388606307200000	0.389114954200000	0.389623601200000	0.390132248200000
0.390640895200000	0.391149542200000	0.391658189200000	0.392166836200000	0.392675483200000
0.393184130200000	0.393692777200000	0.394201424200000	0.394710071200000	0.395218718200000
0.395727365200000	0.396236012200000	0.396744659200000	0.397253306200000	0.397761953200000
0.398270600200000	0.398779247200000	0.399287894200000	0.399796541200000	0.400305188200000
0.400813835200000	0.401322482200000	0.401831129200000	0.402339776200000	0.402848623200000
0.403357070200000	0.403865717200000	0.404374364200000	0.404883011200000	0.405391658200000
0.405900305200000	0.406408952200000	0.406917599200000	0.407426246200000	0.407934893200000
0.408443540200000	0.408952187200000	0.409460834200000	0.409969481200000	0.410478128200000
0.410986775200000	0.411495422200000	0.412004069200000	0.412512716200000	0.413021363200000
0.413530010200000	0.414038657200000	0.414547304200000	0.415055951200000	0.415564598200000
0.416073245200000	0.416581892200000	0.417090539200000	0.417599186200000	0.418107833200000
0.418616480200000	0.419125127200000	0.419633774200000	0.420142421200000	0.420651068200000
0.421159715200000	0.421668362200000	0.422177009200000	0.422685656200000	0.423194303200000
0.423702950200000	0.424211597200000	0.424720244200000	0.425228891100000	0.425737538100000
0.426246185100000	0.426754832100000	0.427263479100000	0.427772126100000	0.428280773100000
0.428789420100000	0.429298067100000	0.429806714100000	0.430315361100000	0.430824008100000
0.431332655100000	0.431841302100000	0.432349949100000	0.432858596100000	0.433367243100000
0.433875890100000	0.434384537100000	0.434893184100000	0.435401831100000	0.435910478100000
0.436419125100000	0.436927772100000	0.437436419100000	0.437945066100000	0.438453713100000
0.438962360100000	0.439471007100000	0.439979654100000	0.440488301100000	0.440996948100000
0.441505595100000	0.442014242100000	0.442522889100000	0.443031536100000	0.443540183100000
0.444048830100000	0.445066124100000	0.445574771100000	0.446083418100000	0.446592065100000
0.447100712100000	0.447609359100000	0.448118006100000	0.448626653100000	0.449135300100000
0.449643947100000	0.450152594100000	0.450661241100000	0.451169888100000	0.451678535100000
0.452187182100000	0.452695829100000	0.453204476100000	0.453713123100000	0.454221770100000
0.454730417100000	0.455239064100000	0.455747711100000	0.456256358100000	0.456765005100000
0.457273652100000	0.457782299100000	0.458290946100000	0.458799593100000	0.459308240100000
0.459816887100000	0.460325534100000	0.460834181100000	0.461342828100000	0.461851475100000
0.462360122100000	0.462868769100000	0.463377416100000	0.463886063100000	0.464394710100000
0.464903357100000	0.465412004100000	0.465920651100000	0.466429298100000	0.466937945100000
0.467446592100000	0.467955239100000	0.468463886100000	0.468972533100000	0.469481180100000
0.469989827100000	0.470498474100000	0.471007121100000	0.471515768100000	0.472024415100000
0.472533062100000	0.473041709100000	0.473550356100000	0.474059003100000	0.474567650100000
0.475076297000000	0.475584944000000	0.476093591000000	0.476602238000000	0.477110885000000

0.477619532000000	0.478128179000000	0.478636826000000	0.479145473000000	0.479654120000000
0.480162767000000	0.480671414000000	0.481180061000000	0.481688708000000	0.482197355000000
0.482706002000000	0.483214649000000	0.483723296000000	0.484231943000000	0.484740590000000
0.485249237000000	0.485757884000000	0.486266531000000	0.486775178000000	0.487283825000000
0.487792472000000	0.488301119000000	0.488809766000000	0.489318413000000	0.489827060000000
0.490335707000000	0.490844354000000	0.491353001000000	0.491861648000000	0.492370295000000
0.492878942000000	0.493387589000000	0.493896236000000	0.494404883000000	0.494913530000000
0.495422177000000	0.495930824000000	0.496439471000000	0.496948118000000	0.497456765000000
0.497965412000000	0.498474059000000	0.498982706000000	0.499491353000000	0.500000000000000
0.500508647000000	0.501017294000000	0.501525941000000	0.502034588000000	0.502543235000000
0.503051882000000	0.503560529000000	0.504069176000000	0.504577823000000	0.505086470000000
0.505595117000000	0.506103764000000	0.506612411000000	0.507121058000000	0.507629705000000
0.508138352000000	0.508646999000000	0.509155646000000	0.509664293000000	0.510172940000000
0.510681587000000	0.511190234000000	0.511698881000000	0.512207528000000	0.512716175000000
0.513224822000000	0.513733469000000	0.514242116000000	0.514750763000000	0.515259410000000
0.515768057000000	0.516276704000000	0.516785351000000	0.517293998000000	0.517803011900000
0.518311292000000	0.518819939000000	0.519328586000000	0.519837233000000	0.520345880000000
0.520854527000000	0.521363174000000	0.521871821000000	0.522380468000000	0.522889115000000
0.523397762000000	0.523906409000000	0.524415056000000	0.524923703000000	0.525432349900000
0.525940996000000	0.526449643900000	0.527466937900000	0.527975584900000	0.528483011900000
0.528992878900000	0.529501525900000	0.530010172900000	0.530518819900000	0.531027466900000
0.531536113900000	0.532044760900000	0.532553407900000	0.533062054900000	0.533570701900000
0.534079348900000	0.534587995900000	0.535096642900000	0.535605289900000	0.536113936900000
0.536622583900000	0.537131230900000	0.537639877900000	0.538148524900000	0.538657171900000
0.539165818900000	0.539674465900000	0.540183112900000	0.540691759900000	0.541200406900000
0.541709053900000	0.542217700900000	0.542726347900000	0.543234994900000	0.543743641900000
0.544252288900000	0.544760935900000	0.545269582900000	0.545778229900000	0.546286876900000
0.546795523900000	0.547304170900000	0.547812817900000	0.548321464900000	0.548830111900000
0.549338758900000	0.549847405900000	0.550356052900000	0.550864699900000	0.551373346900000
0.551881993900000	0.552390640900000	0.552899287900000	0.553407934900000	0.553916581900000
0.554425228900000	0.554933875900000	0.555442522900000	0.555951169900000	0.556459816900000
0.556968463900000	0.557477110900000	0.557985757900000	0.558494404900000	0.559003051900000
0.559511698900000	0.560020345900000	0.560528992900000	0.561037639900000	0.561546286900000
0.562054933900000	0.563072227900000	0.563580874900000	0.564089521900000	0.564598168900000
0.565106815900000	0.565615462900000	0.566124109900000	0.566632756900000	0.567141403900000
0.567650509000000	0.568158697900000	0.568667344900000	0.569175991900000	0.569684638900000
0.570193285900000	0.570701932900000	0.571210579900000	0.571719226900000	0.572227873900000
0.572736520900000	0.573245167900000	0.573753814900000	0.574262461900000	0.574771108900000
0.575279755800000	0.575788402800000	0.576297049800000	0.576805696800000	0.577314343800000
0.577822990800000	0.578331637800000	0.578840284800000	0.579348931800000	0.579857578800000
0.580366225800000	0.580874872800000	0.581383519800000	0.581892166800000	0.582400813800000
0.582909460800000	0.583418107800000	0.583926754800000	0.584435401800000	0.584944048800000
0.585452695800000	0.585961342800000	0.586469989800000	0.586978636800000	0.587487283800000
0.587995930800000	0.588504577800000	0.589013224800000	0.589521871800000	0.590030518800000
0.590539165800000	0.591047812800000	0.591556459800000	0.592065106800000	0.592573753800000
0.593082400800000	0.593591047800000	0.594099694800000	0.594608341800000	0.595116988800000
0.595625635800000	0.596134282800000	0.596642929800000	0.597151576800000	0.597660223800000
0.598168870800000	0.598677517800000	0.599186164800000	0.599694811800000	0.600203458800000
0.600712105800000	0.601220752800000	0.601729399800000	0.602238046800000	0.602746693800000
0.603255340800000	0.603763987800000	0.604272634800000	0.604781281800000	0.605289928800000
0.605798575800000	0.606307222800000	0.606815869800000	0.607324516800000	0.607833163800000
0.608341810800000	0.608850457800000	0.609359104800000	0.609867751800000	0.610376398800000
0.610885045800000	0.611393692800000	0.611902339800000	0.612410986800000	0.612919633800000
0.613428280800000	0.613936927800000	0.614445574800000	0.614954221800000	0.615462868800000
0.616480162800000	0.616988809800000	0.617497456800000	0.618006103800000	0.618514750800000
0.619023397800000	0.619532044800000	0.620040691800000	0.620549338800000	0.621057985800000
0.621566632800000	0.622075279800000	0.622583926800000	0.623092573800000	0.623601220800000
0.624109867800000	0.624618514800000	0.625127161700000	0.625635808700000	0.626144455700000
0.626653102700000	0.627161749700000	0.627670396700000	0.628179043700000	0.628687690700000
0.629196337700000	0.629704984700000	0.630213631700000	0.630722278700000	0.631230925700000
0.631739572700000	0.632248219700000	0.632756866700000	0.633265513700000	0.633774160700000
0.634282807700000	0.634791454700000	0.635300101700000	0.636317395700000	0.636826042700000
0.637334689700000	0.637843336700000	0.638860630700000	0.639369277700000	0.639877924700000
0.640386571700000	0.640895218700000	0.641403865700000	0.641912512700000	0.642421159700000
0.642929806700000	0.643438453700000	0.643947100700000	0.644455747700000	0.644964394700000
0.645473041700000	0.645981688700000	0.646490335700000	0.646998982700000	0.647507629700000
0.648016276700000	0.648524923700000	0.649033570700000	0.649542217700000	0.650050864700000
0.650559511700000	0.651068158700000	0.651576805700000	0.652085452700000	0.652594099700000
0.653102746700000	0.653611393700000	0.654120040700000	0.654628687700000	0.655137334700000

0.655645981700000	0.656154628700000	0.656663275700000	0.657171922700000	0.657680569700000
0.658189216700000	0.658697863700000	0.659206510700000	0.659715157700000	0.660223804700000
0.660732451700000	0.661241098700000	0.661749745700000	0.662258392700000	0.662767039700000
0.663275686700000	0.663784333700000	0.664292980700000	0.664801627700000	0.665310274700000
0.665818921700000	0.666327568700000	0.666836215700000	0.667344862700000	0.667853509700000
0.668362156700000	0.668870803700000	0.669379450700000	0.669888097700000	0.670396744700000
0.670905391700000	0.671414038700000	0.671922685700000	0.672939979700000	0.673448626700000
0.673957273700000	0.674465920700000	0.674974567700000	0.675483214600000	0.675991861600000
0.676500508600000	0.677009155600000	0.677517802600000	0.678026449600000	0.678535096600000
0.679043743600000	0.679552390600000	0.680061037600000	0.680569684600000	0.681078331600000
0.681586978600000	0.682095625600000	0.682604272600000	0.683112919600000	0.683621566600000
0.684130213600000	0.684638860600000	0.685147507600000	0.685656154600000	0.686164801600000
0.686673448600000	0.687182095600000	0.687690742600000	0.688199389600000	0.688708036600000
0.689216683600000	0.689725330600000	0.690233977600000	0.690742624600000	0.691251271600000
0.691759918600000	0.692268565600000	0.692777212600000	0.693285859600000	0.693794506600000
0.694303153600000	0.694811800600000	0.695320447600000	0.695829094600000	0.696337741600000
0.696846388600000	0.697355035600000	0.697863682600000	0.698372329600000	0.698880976600000
0.699389623600000	0.699898270600000	0.700406917600000	0.700915564600000	0.701424211600000
0.701932858600000	0.702441505600000	0.702950152600000	0.703458799600000	0.703967446600000
0.704476093600000	0.704984740600000	0.705493287600000	0.706002034600000	0.706516480160000
0.707019328600000	0.707527975600000	0.708036622600000	0.708545269600000	0.709053916600000
0.709562563600000	0.710071210600000	0.710579857600000	0.711088504600000	0.711597151600000
0.712105798600000	0.712614445600000	0.713123092600000	0.713631739600000	0.714140386600000
0.714649033600000	0.715157680600000	0.715666327600000	0.716174974600000	0.716683621600000
0.717192268600000	0.717700915600000	0.718209562600000	0.718718209600000	0.719226856600000
0.719735503600000	0.720752797600000	0.721261444600000	0.721770091600000	0.722278738600000
0.722787385600000	0.723296032600000	0.723804679600000	0.724313326600000	0.724821973600000
0.725330620500000	0.725839267500000	0.726347914500000	0.726856561500000	0.727365208500000
0.727873855500000	0.728382502500000	0.728891149500000	0.729399796500000	0.729908443500000
0.730417090500000	0.730925737500000	0.731434384500000	0.731943031500000	0.732451678500000
0.732960325500000	0.733468972500000	0.733977619500000	0.734486266500000	0.734994913500000
0.735503560500000	0.736012207500000	0.736520854500000	0.737029501500000	0.737538148500000
0.738046795500000	0.738555442500000	0.739064089500000	0.739572736500000	0.740081383500000
0.740590030500000	0.741098677500000	0.741607324500000	0.742115971500000	0.742624618500000
0.743133265500000	0.743641912500000	0.744150559500000	0.744659206500000	0.745167853500000
0.745676050050000	0.746185147500000	0.746693794500000	0.747202441500000	0.747711088500000
0.748219735500000	0.748728382500000	0.749237029500000	0.749745676500000	0.750254323500000
0.750762970500000	0.751271617500000	0.751780264500000	0.752288911500000	0.752797558500000
0.753306205500000	0.753814852500000	0.754323499500000	0.754832146500000	0.755340793500000
0.755849440500000	0.756358087500000	0.756866734500000	0.757375381500000	0.757884028500000
0.758392675500000	0.758901322500000	0.759409969500000	0.759918616500000	0.760427263500000
0.760935910500000	0.761444557500000	0.761953204500000	0.762461851500000	0.762970498500000
0.763479145500000	0.763987792500000	0.764496439500000	0.765005086500000	0.765513733500000
0.766022380500000	0.766531027500000	0.767039674500000	0.767548321500000	0.768056968500000
0.768565615500000	0.769074262500000	0.769582909500000	0.770091556500000	0.771108850500000
0.771617497500000	0.772126144500000	0.772634791500000	0.773143438500000	0.773652085500000
0.774160732500000	0.774669379500000	0.775178026400000	0.775686673400000	0.776195320400000
0.776703967400000	0.777212614400000	0.777721261400000	0.778229908400000	0.778738555400000
0.779247202400000	0.779755849400000	0.780264496400000	0.781281790400000	0.781790437400000
0.782299084400000	0.782807731400000	0.783316378400000	0.784333672400000	0.784842319400000
0.785350966400000	0.785859613400000	0.786368260400000	0.786876907400000	0.787385554400000
0.787894201400000	0.788402848400000	0.788911495400000	0.789420142400000	0.789928789400000
0.790437436400000	0.790946083400000	0.791454730400000	0.791963377400000	0.792472024400000
0.792980671400000	0.793489318400000	0.793997965400000	0.794506612400000	0.795015259400000
0.795523906400000	0.796032553400000	0.796541200400000	0.797049847400000	0.797558494400000
0.798067141400000	0.798575788400000	0.799084435400000	0.799593082400000	0.800101729400000
0.800610376400000	0.801627670400000	0.802136317400000	0.802644964400000	0.803153611400000
0.803662258400000	0.804679552400000	0.805188199400000	0.805696846400000	0.806205493400000
0.806714140400000	0.807222787400000	0.807731434400000	0.808240081400000	0.809257375400000
0.809766022400000	0.810274669400000	0.810783316400000	0.811291963400000	0.811800610400000
0.812309257400000	0.812817904400000	0.813326551400000	0.813835198400000	0.814343845400000
0.814852492400000	0.815361139400000	0.815869786400000	0.816378433400000	0.816887080400000
0.817395727400000	0.817904374400000	0.818921668400000	0.819430315400000	0.820447609400000
0.820956256400000	0.821464903400000	0.821973550400000	0.822482197400000	0.822990844400000
0.823499491400000	0.824008138400000	0.824516785400000	0.825025432300000	0.825534079300000
0.826042726300000	0.826551373300000	0.827060020300000	0.828077314300000	0.828585961300000
0.829094608300000	0.829603255300000	0.830111902300000	0.830620549300000	0.831129196300000
0.831637843300000	0.832146490300000	0.833163784300000	0.834181078300000	0.834689725300000
0.835198372300000	0.835707019300000	0.836215666300000	0.837232960300000	0.837741607300000

0.838758901300000	0.839267548300000	0.839776195300000	0.840284842300000	0.840793489300000
0.841302136300000	0.841810783300000	0.842319430300000	0.842828077300000	0.843336724300000
0.843845371300000	0.844354018300000	0.844862665300000	0.845371312300000	0.845879959300000
0.846388606300000	0.846897253300000	0.847405900300000	0.847914547300000	0.848423194300000
0.848931841300000	0.849440488300000	0.849949135300000	0.850457782300000	0.850966429300000
0.851475076300000	0.851983723300000	0.852492370300000	0.853001017300000	0.853509664300000
0.854018311300000	0.855035605300000	0.855544252300000	0.856052899300000	0.856561546300000
0.857070193300000	0.857578840300000	0.858087487300000	0.858596134300000	0.859104781300000
0.859613428300000	0.860122075300000	0.860630722300000	0.861139369300000	0.861648016300000
0.862156663300000	0.862665310300000	0.863173957300000	0.863682604300000	0.864191251300000
0.864699898300000	0.865208545300000	0.865717192300000	0.866225839300000	0.866734486300000
0.867243133300000	0.867751780300000	0.868260427300000	0.868769074300000	0.869277721300000
0.869786368300000	0.870295015300000	0.870803662300000	0.871820956300000	0.872329603300000
0.872838250300000	0.873346897300000	0.873855544300000	0.874364191300000	0.874872838300000
0.875381485200000	0.875890132200000	0.876398779200000	0.877416073200000	0.877924720200000
0.878433367200000	0.878942014200000	0.879450661200000	0.879959308200000	0.880426752000000
0.880976602200000	0.881485249200000	0.881993896200000	0.882502543200000	0.883011190200000
0.883519837200000	0.884028484200000	0.884537131200000	0.885045778200000	0.885554425200000
0.886063072200000	0.886571719200000	0.887080366200000	0.887589013200000	0.888097660200000
0.888606307200000	0.889114954200000	0.889623661200000	0.890132248200000	0.890640895200000
0.891149542200000	0.891658189200000	0.892166836200000	0.892675483200000	0.893184130200000
0.893692777200000	0.894201424200000	0.894710071200000	0.895218718200000	0.895727365200000
0.896236012200000	0.896744659200000	0.897253306200000	0.897761953200000	0.898270600200000
0.898779247200000	0.899287894200000	0.899796541200000	0.900305188200000	0.900815835200000
0.901322482200000	0.901831129200000	0.902339776200000	0.902848423200000	0.903357070200000
0.903865717200000	0.904374364200000	0.904883011200000	0.905391658200000	0.905900305200000
0.906408952200000	0.906917599200000	0.907426246200000	0.907934893200000	0.908443540200000
0.908952187200000	0.909460834200000	0.909969481200000	0.910478128200000	0.910986775200000
0.911495422200000	0.912004069200000	0.912512716200000	0.913021363200000	0.913530010200000
0.914547304200000	0.915055951200000	0.915564598200000	0.916073245200000	0.916581892200000
0.917599186200000	0.918616480200000	0.919125127200000	0.919633774200000	0.920142421200000
0.920651068200000	0.921159715200000	0.921668362200000	0.922177009200000	0.922685656200000
0.923194303200000	0.923702950200000	0.924211597200000	0.924720244200000	0.925228891100000
0.925737538100000	0.926246185100000	0.926754832100000	0.927263479100000	0.927772126100000
0.928789420100000	0.929298067100000	0.929806714100000	0.930824008100000	0.931332655100000
0.931841302100000	0.932858596100000	0.933367243100000	0.933875890100000	0.934384537100000
0.934893184100000	0.935401831100000	0.935910478100000	0.936419125100000	0.937436419100000
0.937945066100000	0.938453713100000	0.938962360100000	0.939471007100000	0.939979654100000
0.940488301100000	0.940996948100000	0.941505595100000	0.942014242100000	0.942522889100000
0.943031536100000	0.943540183100000	0.944048830100000	0.944557477100000	0.945066124100000
0.945574771100000	0.946083418100000	0.946592065100000	0.947100712100000	0.947609359100000
0.948118006100000	0.948626653100000	0.949135300100000	0.949643947100000	0.950152594100000
0.950661241100000	0.951169888100000	0.952187182100000	0.952695829100000	0.953204476100000
0.954221770100000	0.954730417100000	0.955239064100000	0.955747711100000	0.956256358100000
0.956765005100000	0.957273652100000	0.957782299100000	0.958290946100000	0.958799593100000
0.959308240100000	0.960325534100000	0.960834181100000	0.961342828100000	0.961851475100000
0.962360122100000	0.962868769100000	0.963886063100000	0.964394710100000	0.964903357100000
0.965412004100000	0.965920651100000	0.966429298100000	0.966937945100000	0.967445692100000
0.967955239100000	0.968463886100000	0.968972533100000	0.969489827100000	0.970498474100000
0.971007121100000	0.972024415100000	0.972533062100000	0.973550356100000	0.974059003100000
0.974567650100000	0.975076297000000	0.975584944000000	0.976022380000000	0.977110885000000
0.977619532000000	0.978128179000000	0.978636826000000	0.979145473000000	0.979654120000000
0.980162767000000	0.980671414000000	0.981180061000000	0.981688708000000	0.982197355000000
0.982706002000000	0.983723296000000	0.984231943000000	0.984740590000000	0.985249237000000
0.985757884000000	0.986266531000000	0.986775178000000	0.987283825000000	0.987792472000000
0.988301119000000	0.988809766000000	0.989318413000000	0.989827060000000	0.990335707000000
0.990844354000000	0.991353001000000	0.991861648000000	0.992370295000000	0.992878942000000
0.993387589000000	0.993896236000000	0.994404883000000	0.994913530000000	0.995422177000000
0.995930824000000	0.996439471000000	0.996948118000000	0.997456765000000	0.997965412000000
0.998474059000000	0.998982706000000	1];		
x = multiplier*x;				
end				
function [x,val] = getCDF2				
val				
[[0.000635324015200000,0.00127064803000000,0.00190597204600000,0.00381194409100000,0.0044472681				
0700000,0.00508259212200000,0.00571791613700000,0.00635324015200000,0.00698856416800000,0.00762				
388818300000,0.00825921219800000,0.00889453621300000,0.00952986022900000,0.0101651842400000,0.0				
108005082600000,0.0114358322700000,0.0133418043200000,0.016518424400000,0.0184243964400000,0.0				

203303684900000,0.0209656925000000,0.0216010165200000,0.0222363405300000,0.0228716645500000,0.0
235069885600000,0.0254129606100000,0.0260482846300000,0.0279542566700000,0.0285895806900000,0.0
292249047000000,0.0298602287200000,0.0304955527300000,0.0311308767500000,0.0343074968200000,0.0
349428208400000,0.0355781448500000,0.0362134688700000,0.0368487928800000,0.0374841169000000,0.0
381194409100000,0.0387547649300000,0.0393900889500000,0.0425667090200000,0.0432020330400000,0.0
463786531100000,0.0470139771300000,0.0476493011400000,0.0482846251600000,0.0489199491700000,0.0
495552731900000,0.0501905972000000,0.0508259212200000,0.0514612452400000,0.0520965692500000,0.0
527318932700000,0.0533672172800000,0.0540025413000000,0.0546378653100000,0.0552731893300000,0.0
571791613700000,0.0578144853900000,0.0584498094000000,0.0590851334200000,0.0609911054600000,0.0
616264294800000,0.0622617534900000,0.0641677255400000,0.0648030495600000,0.0654383735700000,0.0
660736975900000,0.0667090216000000,0.0673443456200000,0.0679796696300000,0.0686149936500000,0.0
705209656900000,0.0762388818300000,0.0832274460000000,0.0838627700100000,0.0844980940300000,0.0
876747141000000,0.0883100381200000,0.0889453621300000,0.0895806861500000,0.0927573062300000,0.0
946632782700000,0.0952986022900000,0.101016518400000,0.102922490500000,0.103557814500000,0.1041
93138500000,0.106099110500000,0.106734434600000,0.107369758600000,0.109275730600000,0.109911054
600000,0.113087674700000,0.114993646800000,0.115628970800000,0.116264294800000,0.11689961880000
0,0.117534942800000,0.118170266800000,0.121346886900000,0.121982210900000,0.123888183000000,0.1
24523507000000,0.128970775100000,0.129606099100000,0.130241423100000,0.132147395200000,0.132782
719200000,0.133418043200000,0.135324015200000,0.135959339300000,0.136594663300000,0.14231257940
0000,0.144218551500000,0.144853875500000,0.145489199500000,0.153748411700000,0.154383735700000,
0.155019059700000,0.155654383700000,0.156289707800000,0.156925031800000,0.158831003800000,0.160
736975900000,0.162642947900000,0.164548919900000,0.165184244000000,0.167090216000000,0.16772554
0000000,0.168360864000000,0.168996188100000,0.169631512100000,0.172808132100000,0.1734435462000
00,0.174078780200000,0.174714104200000,0.175349428200000,0.175984752200000,0.179161372300000,0.
183608640400000,0.184243964400000,0.184879288400000,0.185514612500000,0.186149936500000,0.18678
5260500000,0.187420584500000,0.188055908500000,0.188691232500000,0.190597204600000,0.1912325286
00000,0.195679796700000,0.196315120700000,0.196950444700000,0.197585768700000,0.200762388800000
,0.203939008900000,0.204574332900000,0.205209656900000,0.205844980900000,0.206480305000000,0.20
7115629000000,0.207750953000000,0.208386277000000,0.209021601000000,0.210927573100000,0.2115628
97100000,0.212198221100000,0.212833545100000,0.213468869100000,0.215374841200000,0.216010165200
000,0.216645489200000,0.219822109300000,0.220457433300000,0.233799237600000,0.235705209700000,0.
236340533700000,0.236975857700000,0.237611181700000,0.243329097800000,0.247776365900000,0.2484
11690000000,0.250317662000000,0.252223634100000,0.252858958100000,0.253494282100000,0.254129606
100000,0.254764930100000,0.256670902200000,0.258576874200000,0.261753494300000,0.26238881830000
0,0.263024142300000,0.263659466300000,0.264294790300000,0.264930114400000,0.266836086400000,0.2
67471410400000,0.26816734400000,0.275095298600000,0.275730622600000,0.277636594700000,0.278271
918700000,0.280177890700000,0.282083862800000,0.283989834800000,0.284625158800000,0.28526048280
0000,0.285895806900000,0.286531130900000,0.287166454900000,0.290343075000000,0.290978399000000,
0.291613723000000,0.292249047000000,0.292884371000000,0.293519695000000,0.294155019100000,0.294
790343100000,0.296696315100000,0.299872935200000,0.300508259200000,0.301143583200000,0.30304955
5300000,0.303684879300000,0.304320203300000,0.304955527300000,0.305590851300000,0.3062261753000
00,0.306861499400000,0.307496823400000,0.308132147400000,0.311308767500000,0.311944091500000,0.
313850063500000,0.317026683600000,0.317662007600000,0.319567979700000,0.320203303700000,0.32210
9275700000,0.322744599700000,0.327191867900000,0.327827191900000,0.328462515900000,0.3303684879
00000,0.331003811900000,0.331639136000000,0.332274460000000,0.332909784000000,0.333545108000000
,0.335451080100000,0.336086404100000,0.336721728100000,0.337357052100000,0.337992376100000,0.34
1168996200000,0.344345616300000,0.344980940300000,0.348157560400000,0.348792884400000,0.3506988
56400000,0.352604828500000,0.353240152500000,0.355146124500000,0.355781448500000,0.357687420600
000,0.362134688700000,0.362770012700000,0.363405336700000,0.364040660700000,0.365946632800000,0.
367852604800000,0.369758576900000,0.370393900900000,0.371029224900000,0.371664548900000,0.3748
41169000000,0.375476493000000,0.376111817000000,0.378017789100000,0.378653113100000,0.380559085
100000,0.381194409100000,0.384371029200000,0.385006353200000,0.385641677300000,0.38627700130000
0,0.386912325300000,0.387547649300000,0.388182973300000,0.388818297300000,0.389453621300000,0.3
90088945400000,0.394536213500000,0.395171537500000,0.395806861500000,0.396442185500000,0.409783
989800000,0.410419313900000,0.411054637900000,0.415501906000000,0.425031766200000,0.42566709020
0000,0.426302414200000,0.426937738200000,0.430114358300000,0.432020330400000,0.435196950400000,0.
435832274500000,0.436467598500000,0.438373570500000,0.439008894500000,0.440914866600000,0.441
550190600000,0.443456162600000,0.444091486700000,0.447268106700000,0.447903430700000,0.44853875
4800000,0.456797967000000,0.461245235100000,0.461880559100000,0.462515883100000,0.4631512071000
00,0.463786531100000,0.477128335500000,0.477763659500000,0.478398983500000,0.479034307500000,0.
479669631500000,0.481575603600000,0.482210927600000,0.487928843700000,0.488564167700000,0.48919
9491700000,0.489834815800000,0.495552731900000,0.496188055900000,0.498094028000000,0.5000000000
00000,0.503176620100000,0.503811944100000,0.504447268100000,0.507623888200000,0.509529860200000
,0.516518424400000,0.520965692500000,0.521601016500000,0.524777636600000,0.525412960600000,0.52
8589580700000,0.529224904700000,0.533672172800000,0.535578144900000,0.536213468900000,0.5368487
92900000,0.537484116900000,0.538119440900000,0.538754764900000,0.539390088900000,0.540025413000
000,0.540660737000000,0.542566709000000,0.543202033000000,0.547649301100000,0.548284625200000,0.
548919949200000,0.549555273200000,0.550190597200000,0.550825921200000,0.551461245200000,0.5520
96569300000,0.552731893300000,0.554637865300000,0.555273189300000,0.555908513300000,0.560355781

400000,0.560991105500000,0.561626429500000,0.562261753500000,0.564167725500000,0.56734434560000
0,0.569250317700000,0.573697585800000,0.574332909800000,0.576238881800000,0.576874205800000,0.5
77509529900000,0.579415501900000,0.580050825900000,0.581956798000000,0.582592122000000,0.583227
446000000,0.588945362100000,0.589580686100000,0.594027954300000,0.594663278300000,0.59529860230
0000,0.595933926300000,0.600381194400000,0.601016518400000,0.601651842400000,0.602287166500000,
0.602922490500000,0.603557814500000,0.604193138500000,0.604828462500000,0.605463786500000,0.606
099110500000,0.606734434600000,0.607369758600000,0.608005082600000,0.609911054600000,0.61181702
6700000,0.612452350700000,0.613087674700000,0.613722998700000,0.614358322700000,0.6149936468000
00,0.615628970800000,0.618805590900000,0.619440914900000,0.620076238900000,0.620711562900000,0.
621346886900000,0.621982210900000,0.622617534900000,0.623252859000000,0.625158831000000,0.62579
415500000,0.626429479000000,0.627064803000000,0.627700127100000,0.630876747100000,0.6315120712
00000,0.632147395200000,0.632782719200000,0.634688691200000,0.637865311300000,0.638500635300000
,0.639135959300000,0.639771283400000,0.640406607400000,0.642312579400000,0.642947903400000,0.64
3583227400000,0.645489199500000,0.646124523500000,0.648030495600000,0.648665819600000,0.6493011
43600000,0.652477763700000,0.654383735700000,0.657560355800000,0.658195679800000,0.658831003800
000,0.659466327800000,0.663913595900000,0.664548919900000,0.665184244000000,0.665819568000000,0.
.666454892000000,0.669631512100000,0.674078780200000,0.674714104200000,0.675349428200000,0.6772
55400300000,0.681702668400000,0.682337992400000,0.684243964400000,0.684879288400000,0.685514612
500000,0.686149936500000,0.686785260500000,0.687420584500000,0.688055908500000,0.68869123250000
0,0.689326556500000,0.689961880600000,0.690597204600000,0.691232528600000,0.691867852600000,0.6
92503176600000,0.693138500600000,0.695044472700000,0.695679796700000,0.696315120700000,0.696950
444700000,0.697585768700000,0.698221092800000,0.698856416800000,0.707115629000000,0.70775095300
0000,0.708386277000000,0.710292249000000,0.710927573100000,0.711562897100000,0.712198221100000,
0.716645489200000,0.718551461200000,0.719186785300000,0.719822109300000,0.72045743300000,0.721
092757300000,0.721728081300000,0.722363405300000,0.722998729400000,0.726175349400000,0.72681067
3400000,0.729987293500000,0.730622617500000,0.733799237600000,0.734434561600000,0.7350698856000
00,0.739517153700000,0.741423125800000,0.742058449800000,0.742693773800000,0.743329097800000,0.
743964421900000,0.745870393900000,0.746505717900000,0.747141041900000,0.757941550200000,0.75857
6874200000,0.759212198200000,0.761118170300000,0.761753494300000,0.762388818300000,0.7630241423
00000,0.763659466300000,0.765565438400000,0.766200762400000,0.766836086400000,0.767471410400000
,0.768106734400000,0.770012706500000,0.770648030500000,0.771283354500000,0.774459974600000,0.77
5095298600000,0.775730622600000,0.776365946600000,0.777001270600000,0.777636594700000,0.7782719
18700000,0.778907242700000,0.779542566700000,0.780177890700000,0.784625158800000,0.785260482800
000,0.785895806900000,0.786531130900000,0.787166454900000,0.787801778900000,0.788437102900000,0.
789072426900000,0.789707751000000,0.790343075000000,0.790978399000000,0.791613723000000,0.7922
49047000000,0.792884371000000,0.793519695000000,0.794155019100000,0.794790343100000,0.797966963
200000,0.798602287200000,0.799237611200000,0.799872935200000,0.800508259200000,0.80114358320000
0,0.803049555300000,0.803684879300000,0.804320203300000,0.804955527300000,0.805590851300000,0.8
06226175300000,0.806861499400000,0.810038119400000,0.810673443500000,0.811308767500000,0.811944
091500000,0.812579415500000,0.813214739500000,0.813850063500000,0.818297331600000,0.81893265570
0000,0.819567979700000,0.820203303700000,0.820838627700000,0.821473951700000,0.822109275700000,
0.822744599700000,0.823379923800000,0.824015247800000,0.824650571800000,0.825285895800000,0.825
921219800000,0.826556543800000,0.827191867900000,0.827827191900000,0.828462515900000,0.83036848
7900000,0.833545108000000,0.834180432000000,0.834815756000000,0.835451080100000,0.8360864041000
00,0.836721728100000,0.838627700100000,0.840533672200000,0.841168996200000,0.841804320200000,0.
842439644200000,0.845616264300000,0.846251588300000,0.846886912300000,0.84752236300000,0.84815
7560400000,0.848792884400000,0.849428208400000,0.850063532400000,0.850698856400000,0.8513341804
00000,0.851969504400000,0.853875476500000,0.854510800500000,0.855146124500000,0.855781448500000
0,0.856416772600000,0.857052096600000,0.857687420600000,0.858322744600000,0.860228716600000,0.86
2134688700000,0.862770012700000,0.863405336700000,0.864040660700000,0.865946632800000,0.8665819
56800000,0.867217280800000,0.867852604800000,0.868487928800000,0.869123252900000,0.871029224900
000,0.872935197000000,0.873570521000000,0.874205845000000,0.874841169000000,0.875476493000000,0.
.876111817000000,0.876747141000000,0.877382465100000,0.878017789100000,0.878653113100000,0.8792
88437100000,0.879923761100000,0.880559085100000,0.881194409100000,0.883100381200000,0.883735705
200000,0.884371029200000,0.885006353200000,0.885641677300000,0.886277001300000,0.88691232530000
0,0.887547649300000,0.888182973300000,0.888818297300000,0.889453621300000,0.890088945400000,0.8
90724269400000,0.891359593400000,0.891994917400000,0.893900889500000,0.894536213500000,0.895171
537500000,0.895806861500000,0.896442185500000,0.897077509500000,0.897712833500000,0.89834815760
0000,0.898983481600000,0.902160101700000,0.902795425700000,0.903430749700000,0.904066073700000,
0.904701397700000,0.905336721700000,0.905972045700000,0.909148665800000,0.911054637900000,0.912
960609900000,0.913595933900000,0.914231257900000,0.914866582000000,0.91550190600000,0.91613723
0000000,0.916772554000000,0.917407878000000,0.918043202000000,0.918678526000000,0.9193138501000
00,0.919949174100000,0.921855146100000,0.925031766200000,0.926937738200000,0.927573062300000,0.
928208386300000,0.928843710300000,0.932020330400000,0.932655654400000,0.933290978400000,0.93392
6302400000,0.934561626400000,0.935196950400000,0.935832274500000,0.936467598500000,0.9371029225
00000,0.937738246500000,0.938373570500000,0.940279542600000,0.942185514600000,0.944091486700000
,0.944726810700000,0.945362134700000,0.945997458700000,0.947903430700000,0.949809402800000,0.95
0444726800000,0.951080050800000,0.951715374800000,0.953621346900000,0.954256670900000,0.9574332
91000000,0.958068615000000,0.958703939000000,0.959339263000000,0.959974587000000,0.960609911100

000,0.961245235100000,0.961880559100000,0.963786531100000,0.965692503200000,0.966327827200000,0.970775095300000,0.971410419300000,0.9720454743300000,0.972681067300000,0.974587039400000,0.97522363400000,0.978398983500000,0.979034307500000,0.979669631500000,0.980304955500000,0.980940279500000,0.981575603600000,0.982210927600000,0.982846251600000,0.983481575600000,0.984116899600000,0.986022871700000,0.986658195700000,0.988564167700000,0.989199491700000,0.989834815800000,0.990470139800000,0.991105463800000,0.991740787800000,0.992376111800000,0.993011435800000,0.993646759800000,0.994282083900000,0.994917407900000,0.995552731900000,0.996188055900000,0.996823379900000,0.997458703900000,0.998094028000000,0.998729352000000,1]];

x=[[1.267980700000000,1.272788260000000,1.413846380000000,1.491714846000000,1.500423059000000,1.525455572000000,1.579175145000000,1.594645335000000,1.604215945000000,1.631599817000000,1.631755240000000,1.639441416000000,1.699288134000000,1.711924671000000,1.725070010000000,1.726361142000000,1.730930648000000,1.742870675000000,1.745756902000000,1.745865747000000,1.746260798000000,1.748744355000000,1.759721850000000,1.774271049000000,1.775824262000000,1.784315502000000,1.785221346000000,1.787460164000000,1.789748792000000,1.790975659000000,1.811824682000000,1.812791478000000,1.818866322000000,1.830788466000000,1.836811019000000,1.858474299000000,1.858504373000000,1.863182811000000,1.863760472000000,1.867615470000000,1.878781350000000,1.879308242000000,1.886716411000000,1.888083540000000,1.894720623000000,1.895132337000000,1.897030760000000,1.900019446000000,1.900365657000000,1.902312003000000,1.902788998000000,1.905230323000000,1.917826647000000,1.922350110000000,1.923621751000000,1.931779244000000,1.934091857000000,1.935067625000000,1.938321310000000,1.939865914000000,1.939907612000000,1.942134358000000,1.946433664000000,1.950922730000000,1.952109047000000,1.952646232000000,1.954455738500000,1.961109776000000,1.962289287000000,1.962317489000000,1.965174424000000,1.984174316000000,1.990426465000000,1.991982938000000,1.997974301000000,2.003119205000000,2.012146872000000,2.016675575000000,2.021965945000000,2.028541496000000,2.039006578000000,2.039237500000000,2.041803021000000,2.041877456000000,2.046290472000000,2.046827257000000,2.052428808000000,2.058700044000000,2.061455579000000,2.064336476000000,2.064779919000000,2.065632925000000,2.066102570000000,2.069738003000000,2.081196198000000,2.087019635000000,2.088112773000000,2.092905776000000,2.095518781000000,2.096704880000000,2.097750525000000,2.099209359000000,2.102690524000000,2.104633460000000,2.108906263000000,2.112745524000000,2.116947026000000,2.118315132000000,2.120084597000000,2.128166456000000,2.129487548000000,2.129762013000000,2.133856992000000,2.134274313000000,2.136805990000000,2.137131608000000,2.142317838000000,2.144101645000000,2.144397433000000,2.149180364000000,2.153287305000000,2.167273715000000,2.167766083000000,2.172459924000000,2.173911708000000,2.176070246000000,2.176258279000000,2.178638969000000,2.179945041000000,2.180569841000000,2.182893005000000,2.187051352000000,2.188200446000000,2.188234870000000,2.191079956000000,2.194744289000000,2.197993709000000,2.200081244000000,2.200148483000000,2.201264531000000,2.203942082000000,2.210114383000000,2.212505630000000,2.218074276000000,2.219930700000000,2.221614665000000,2.221723789000000,2.228010125000000,2.229145910000000,2.230499348000000,2.234662093000000,2.237388663000000,2.239375032000000,2.240709574000000,2.243188391000000,2.246022317000000,2.249219296000000,2.250607766000000,2.252834922000000,2.261238257000000,2.263559575000000,2.268289335000000,2.281106425000000,2.281549389000000,2.284639856000000,2.285117312000000,2.285206625000000,2.290521210000000,2.292487031000000,2.297951744000000,2.298859682000000,2.300624832000000,2.303278663000000,2.304384034000000,2.309151920000000,2.312472254000000,2.319228926000000,2.326450656000000,2.328880873000000,2.328895183000000,2.329138820000000,2.331121346000000,2.334911912000000,2.335999860000000,2.342649761000000,2.344956659000000,2.352546002000000,2.352935867000000,2.353970714000000,2.354285766000000,2.354503882000000,2.356649166000000,2.360314002000000,2.365126023000000,2.368478539000000,2.373860019000000,2.382483369000000,2.385005642000000,2.388971941000000,2.389240489000000,2.396545969000000,2.399249887000000,2.405301219000000,2.410026539000000,2.410667530000000,2.411300536000000,2.412651025000000,2.421301949000000,2.422059217000000,2.423997717000000,2.430093015000000,2.432708657000000,2.433106069000000,2.435762286000000,2.438604929000000,2.443367140000000,2.453438590000000,2.456550628000000,2.457248914000000,2.458915339000000,2.460555109000000,2.464013174000000,2.464524710000000,2.466110757000000,2.467214847000000,2.468046070000000,2.469126812000000,2.470210487000000,2.471550805000000,2.471719795000000,2.472305578000000,2.476414947000000,2.476850640000000,2.482819231000000,2.487349444000000,2.488287001000000,2.489862446000000,2.490907420000000,2.491961678000000,2.494752474000000,2.504874649000000,2.509514522000000,2.516282541000000,2.516916896000000,2.522931567000000,2.529273706000000,2.529749526000000,2.538396888000000,2.541397650000000,2.544421102000000,2.545215554000000,2.551810035000000,2.557075948000000,2.561276240000000,2.564896429000000,2.567385218000000,2.567995509000000,2.572616188000000,2.572838050000000,2.573383035000000,2.574052155000000,2.581757505000000,2.582185937000000,2.591776191000000,2.594158916000000,2.595430154000000,2.597171705000000,2.599457491000000,2.599547300000000,2.600721963000000,2.607120138000000,2.607480952000000,2.611201630000000,2.612731284000000,2.618131544000000,2.618932183000000,2.619456413000000,2.627169415000000,2.641144907000000,2.643982638000000,2.646340965000000,2.651444804000000,2.658937007000000,2.663731699000000,2.663746987000000,2.670997519000000,2.678331966000000,2.680352288000000,2.689766118000000,2.691239755000000,2.700326641000000,2.707881319000000,2.710690479000000,2.713646447000000,2.717491802000000,2.722233320000000,2.732510156000000,2.739270438000000,2.740015835000000,2.742860745000000,2.745802014000000,2.748412527000000,2.749542007000000,2.750229149000000,2.751797607000000,2.754100073000000,2.756861774000000,2.757802394000000,2.758427341000000,2.758959052000000,2.765770755000000,2.791219613000000,2.793378428000000,2.796231753000000,2.799781750000000,2.800387548000000,2.801187772000000,2.808272766000000,2.816684922000000,2.819749637000000,2.823553185000000,2.824369997000000,2.825634182000000,2.833555075000000,2.833666677000000,2.833990238000000,2.834406947000000,2.842840189000000,2.848168503000000,2.85446565000000,2.856085213000000,2.858503048000000,2.868460880000000,2.871060910000000,2.873092381000000,2.874436335000000,2.879825160000000,2.885746542000000,2.894346757000000,2.896361454000000,2.908232605

00000,2.92304941100000,2.92861720900000,2.93055282000000,2.93150416300000,2.93453220700000,2.93503308400000,2.94340894800000,2.94822652700000,2.95505390700000,2.96943353600000,2.97068136900000,2.97947942700000,2.98106354900000,2.98156371600000,2.98466107800000,2.98508064300000,2.98657045600000,2.98888227800000,2.99287640900000,2.99508992500000,2.99621874900000,2.99853649900000,3.01721648100000,3.02116303500000,3.02521803000000,3.02845895200000,3.03029066500000,3.03180260300000,3.04031865000000,3.04509461600000,3.04963080100000,3.05006433700000,3.05191763300000,3.05239061900000,3.05780267000000,3.06076988300000,3.06441776600000,3.06750722400000,3.06795996200000,3.06921695800000,3.06952116500000,3.07462730200000,3.07748601800000,3.09733075700000,3.10716871500000,3.11562588300000,3.12468172600000,3.12796564400000,3.13194883000000,3.13725962500000,3.14697919200000,3.16201377500000,3.16486492800000,3.16661383700000,3.16697332000000,3.17001861600000,3.17238682400000,3.17317069300000,3.17645226000000,3.17721100500000,3.17984192400000,3.18919515000000,3.19371459100000,3.19939585900000,3.20021424200000,3.20253642300000,3.21464327300000,3.22294439800000,3.23377260000000,3.23768037200000,3.24926015700000,3.27576539500000,3.28301966800000,3.28363933200000,3.28601882200000,3.28812236600000,3.28993696800000,3.29073411900000,3.29539893300000,3.30642115700000,3.31054564200000,3.31569991600000,3.31869648400000,3.32699080700000,3.33110196100000,3.33265948600000,3.33521761200000,3.33863454700000,3.33928560200000,3.34043276200000,3.34420788400000,3.34709962300000,3.34762843700000,3.34985869200000,3.36257726600000,3.36285588400000,3.36609059800000,3.36847993600000,3.37346833200000,3.37732176700000,3.37991202500000,3.40708675800000,3.41487123900000,3.42060855300000,3.42439821700000,3.42595934700000,3.42730210600000,3.44519030800000,3.45003697500000,3.45659990100000,3.45997547300000,3.47012647800000,3.47066443400000,3.47189490600000,3.47478000800000,3.47606242200000,3.48892272900000,3.49097765700000,3.49316680500000,3.50224841900000,3.50237315600000,3.51075431500000,3.51082832800000,3.51326923400000,3.51891416900000,3.52046184000000,3.52521339100000,3.52738497100000,3.54031721100000,3.54209330000000,3.57015662500000,3.57165404000000,3.57472393500000,3.58020260600000,3.58272025000000,3.59356381800000,3.60729003500000,3.60736788900000,3.61085399200000,3.65505708800000,3.67865413200000,3.68000729900000,3.68399848500000,3.68567099600000,3.68960601700000,3.69905046100000,3.70618288200000,3.70981579600000,3.71606072500000,3.71839569400000,3.72695787500000,3.75744271700000,3.76848837300000,3.77728067400000,3.77894151200000,3.78450950700000,3.78979778900000,3.79658438000000,3.79961562900000,3.80777118400000,3.81947510100000,3.82746378500000,3.84485058700000,3.84657493300000,3.86022484200000,3.86128513600000,3.86857185400000,3.87315521000000,3.87445202100000,3.87957670300000,3.88648517800000,3.88703357400000,3.89467204900000,3.91276097200000,3.91623260500000,3.93273174700000,3.93654447800000,3.93754762900000,3.94356833800000,3.95167142300000,3.96967144400000,3.97491150800000,3.97643099400000,3.98429712900000,3.99117450700000,3.99719558400000,4.00029544500000,4.00078997800000,4.00212342900000,4.01698251700000,4.02002178600000,4.02311635200000,4.02763838000000,4.03165077100000,4.03851146300000,4.03972215200000,4.04204815600000,4.04376343000000,4.07468697800000,4.07652609400000,4.09421464100000,4.09774710900000,4.10609232200000,4.12335276600000,4.15548371100000,4.15778266400000,4.16207769100000,4.16554056500000,4.16792855900000,4.17193905500000,4.17851263000000,4.19226263800000,4.19227539100000,4.19230640100000,4.20015475000000,4.20377315100000,4.20661176400000,4.2175119200000,4.22532084800000,4.23264591600000,4.24494568300000,4.26982342200000,4.29023241100000,4.31679915200000,4.32501055100000,4.32941913000000,4.33059315600000,4.33095103400000,4.36237386200000,4.37499086800000,4.39690319300000,4.40744498200000,4.40898170100000,4.41158309700000,4.45729169000000,4.45894705500000,4.48863705500000,4.51458329100000,4.52546783100000,4.52845991000000,4.54041376200000,4.54844983700000,4.57478454000000,4.57865211700000,4.59026168200000,4.59247665800000,4.59577749700000,4.59800925200000,4.60146440600000,4.63968080300000,4.64515001700000,4.64766527900000,4.67503334100000,4.67824921000000,4.68338453000000,4.69643111500000,4.70764905800000,4.73207681600000,4.75119337400000,4.78821210600000,4.83107914500000,4.83395777100000,4.85458338400000,4.85977062500000,4.90093947400000,4.91135990200000,4.91957466200000,4.92112950600000,4.92881998400000,4.94145551500000,4.94191535400000,4.95303632400000,4.96888872200000,4.98142287700000,4.98379252700000,4.98660642800000,5.00490750800000,5.01423414900000,5.01634396400000,5.02464558800000,5.03414034200000,5.05888464600000,5.07312479200000,5.07912202500000,5.08515132400000,5.10273525000000,5.10477022500000,5.11189752200000,5.12271793400000,5.14774391800000,5.14958297500000,5.15757892700000,5.17444833500000,5.18115784000000,5.19610559300000,5.21210212100000,5.22933723700000,5.25429512700000,5.26856514900000,5.28974253100000,5.29131605000000,5.29913836900000,5.33359527200000,5.33475155300000,5.33970473100000,5.37093916800000,5.39606419600000,5.40701729000000,5.42050257200000,5.43379627400000,5.44116559600000,5.44422663400000,5.48502892600000,5.49292940200000,5.51019456100000,5.52744181800000,5.55491216600000,5.56672739800000,5.57397866500000,5.59144008500000,5.60510694600000,5.60935895700000,5.65203034600000,5.6569309900000,5.69318721700000,5.70343025900000,5.71044336200000,5.71443100500000,5.72051156000000,5.72860958300000,5.73213527800000,5.76405555900000,5.77302209900000,5.77528171200000,5.80211267100000,5.81140606300000,5.81931879500000,5.82113114200000,5.83875409900000,5.84230007300000,5.85275339400000,5.87300527900000,5.92116575700000,5.93975544200000,5.96617076600000,5.98783700900000,6.03979349800000,6.11147539800000,6.17589775700000,6.19231146700000,6.33114361600000,6.33126192100000,6.34192686800000,6.40089326800000,6.41399834100000,6.44625510300000,6.46560481000000,6.48114076900000,6.60154682300000,6.61615050200000,6.63114606400000,6.67155633200000,6.68769678100000,6.73819397800000,6.74326769600000,6.76557745100000,6.98357348000000,7.11024381700000,7.14101812700000,7.14518235800000,7.34079238100000,7.44517159700000,7.47193049900000,7.53039762700000,7.53938297000000,7.56725002600000,7.69025089900000,7.69152607200000,7.71287096300000,7.75034424200000,7.75367264300000,7.77584301200000,7.79153922500000,7.95460182300000,8.08820982600000,8.112

```
321194700000,8.16294550500000,8.17980829000000,8.22581214900000,8.28602697400000,8.359964847000
00,8.83484000200000,8.97670825700000,9.01953073100000,9.28740271600000,9.31889203400000,9.36069
640100000,9.78309279100000,9.84645613600000,9.90466529300000,10.7049498500000,11.1551261200000,
11.1794618600000,11.6316594900000,11.6512765000000,11.7589091000000,12.8036298000000,13.3565604
100000,13.5621489600000,13.5717860400000,13.5876977500000,13.5936881200000,13.6621178300000,13.
6721115000000,13.6890081600000,13.7662102800000,13.7957012500000,13.8297001600000,13.8500102000
000,13.9590584700000,13.9813365000000,14.0112248200000,14.1206767900000,14.1431514600000,14.154
5726100000,14.1657619800000,14.2076155900000,14.2245268100000,14.2375059300000,14.2459426500000
,14.2530731500000,14.3113058600000,14.3732419700000,14.4046238200000,14.4116001900000,14.479793
6800000,14.5701441400000,14.6680969800000,14.8450068600000,14.9306283400000,14.9551318300000,14
.9737008900000,14.9887976400000,15]];
```

end

7.8 Appendix H: Pial growth model for human cortex

The human pial growth is significantly more complicated than that of the mouse. This is due to deep gyrations in the human brain, leading to ridges and valleys that are not present in the relatively smooth, concave surface of the mouse brain. When the growth algorithm generates a sample on a concave surface, such as the mouse cortex, connecting linearly to any segment and projecting the bifurcation point to the surface is an adequate model. Unfortunately, the gyrations in the human cortex causes these connections to “cut” through the pial surface in the human as shown in Figure 7.57.



Figure 7.57. Example of attaching two points on a human pial surface. This connection (between points p_o and p_n) will cut through at least one gyration of the pial surface. Other connections may avoid such cuts while some may have many ridges to cut through.

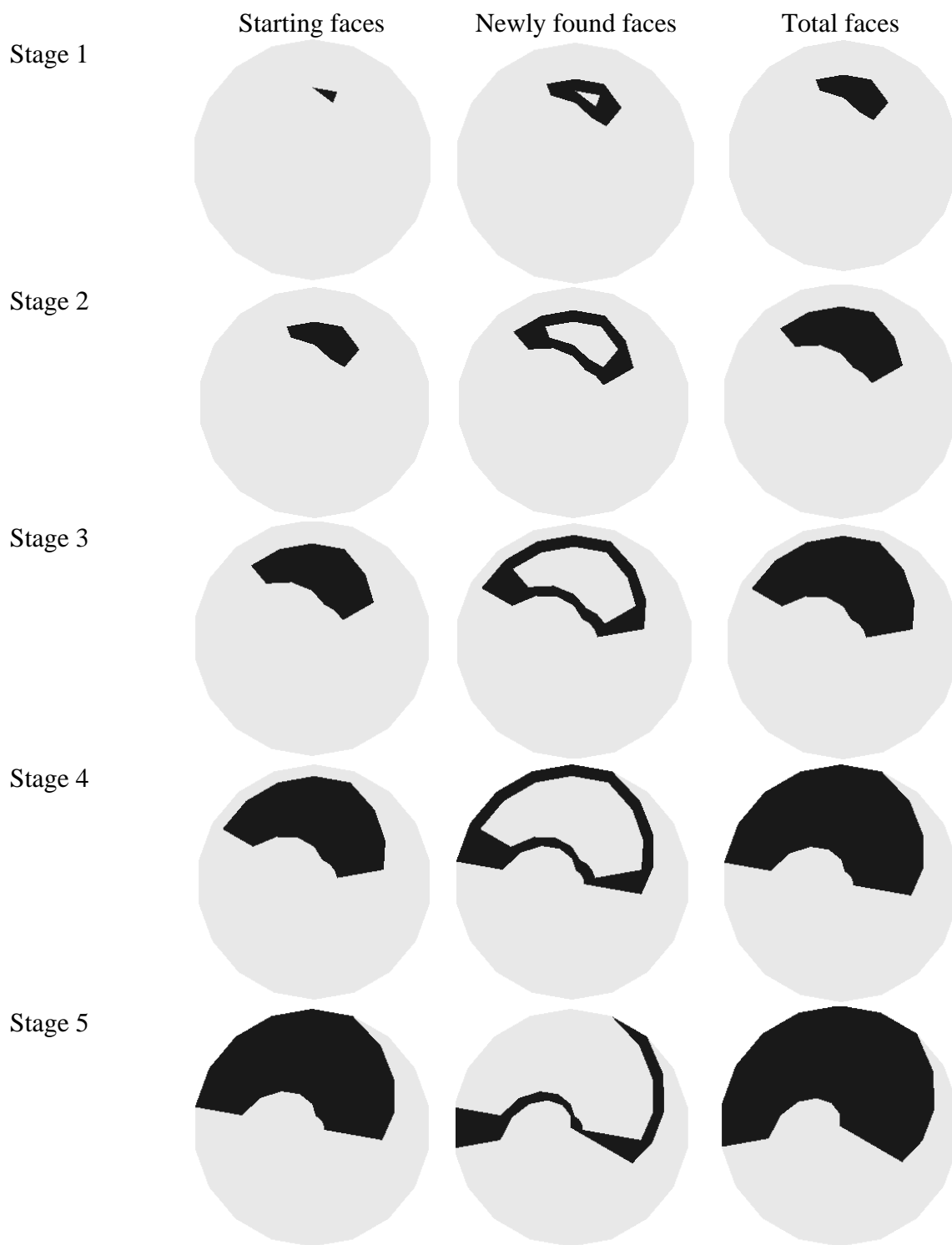
This chapter investigates methods for modifying the previously proposed pial growth methodology in light of the human cortex. The recommended method is to implement a 2-stage growth; stage 1 will growth large vessel “trunks” deep in the gyrations while stage 2 will growth the smaller pial vessels onto the adjacent ridges. The valleys should be identified using a curvature estimation method such as those investigated in this chapter. The growth will require a moving-island sample generator as described later.

7.8.1 Growth from expansion

The simplest way to grow along a surface that has a lot of peaks and valleys is to limit the growth to only the near neighborhood of the current vessels. This ensures the new segment will not be far away from the segment and the probability of cutting through a gyration decreases significantly (e.g. the terminal will always have a segment close by to connect to). This method relies on a structure called a *point matrix* that holds the connected mesh triangles attached to a point in the mesh. This data structure can be generated once and interrogated throughout the growth process (as opposed to interrogating the entire mesh structure for connectivity at each new growth of a segment). The general overview of the algorithm starts with a single face and expands until the number of iterations has been reached. Within each iteration, the user may select any number of expansion steps (=how many levels to expand). Each expansion step includes 3 parts: (i) for each selected face, track all connected faces, (ii) remove all previous selections (including current selection) from list, and (iii) assign remaining faces as new selection. The result of this method can be seen below for a circle mesh structure.

7.8.1.1 Case Studies:

The first case study involves a single-step expansion where each iteration of expansion only identifies 1st connections to the initial face selection (as opposed to branching to 2nd or 3rd order connections that are up to 2 or 3 elements away). The results indicate the growth algorithm is capable of expanding until every element of the surface has been selected at least one time.



Stage 6



Stage 7



Stage 8



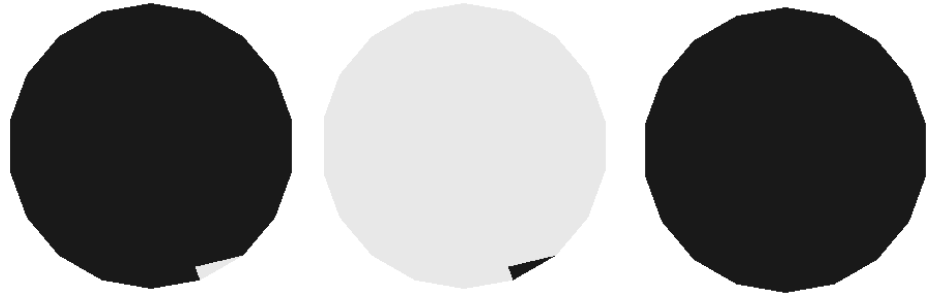
Stage 9



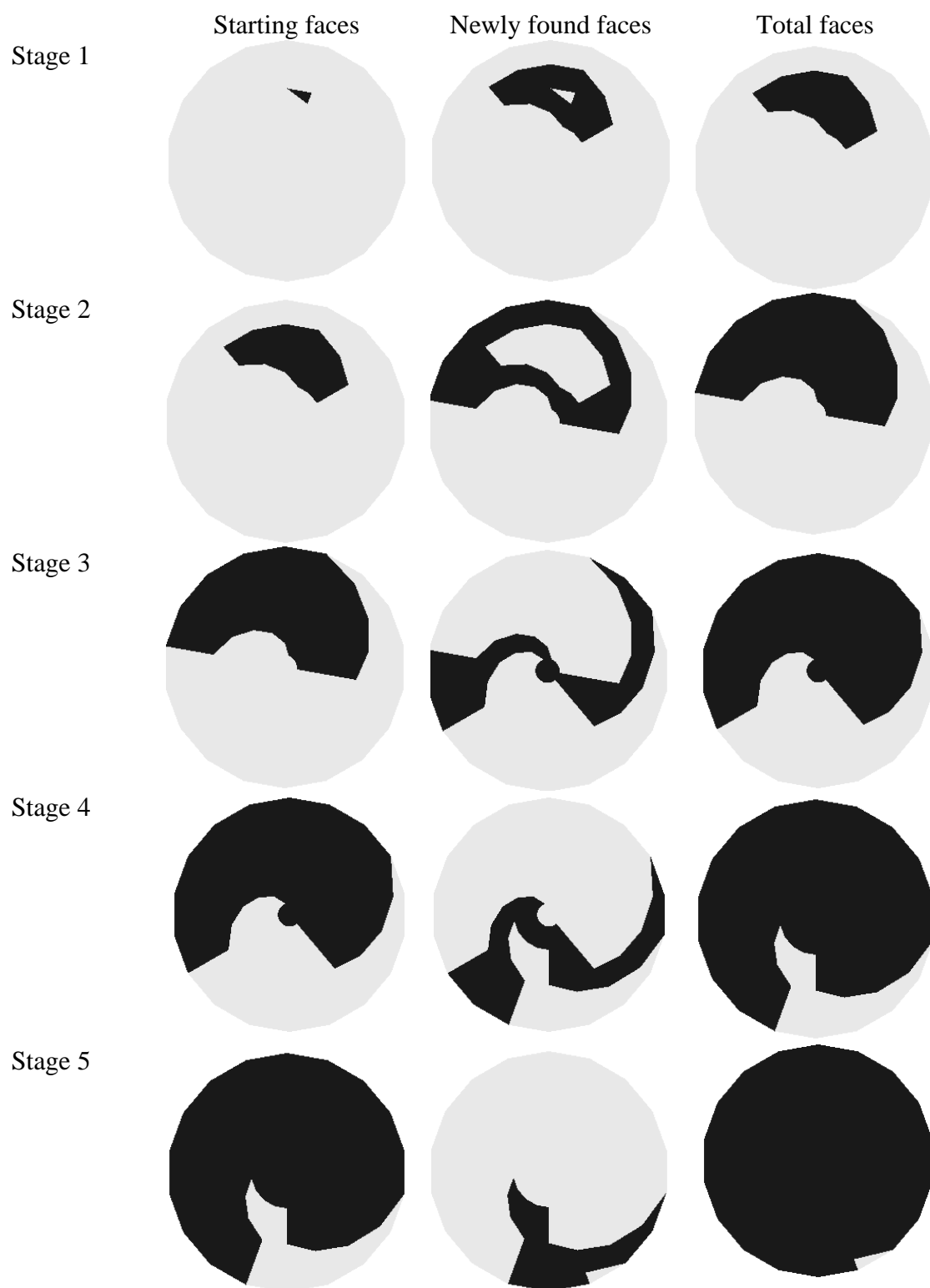
Stage 10



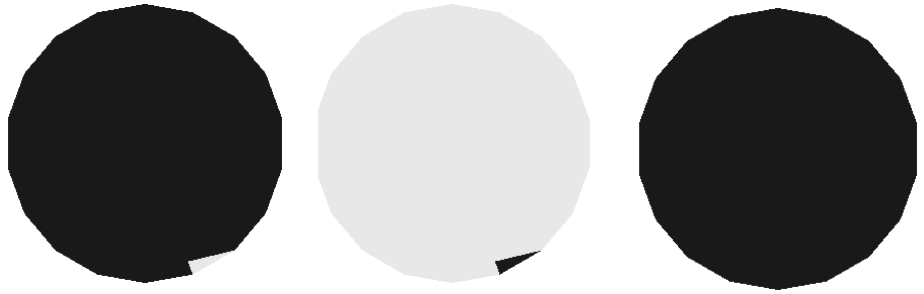
Stage 11



The second case study implements a double-step expansion which allows each iteration to select elements up to 2 orders away from the current selection. This can be useful if the algorithm is expanding through a mesh with small triangles, where the vascular density is much lower than the triangle density. The results again indicate excellent ability to identify mesh elements in an ordered fashion.



Stage 6



7.8.2 Using ridge detection (paraboloid fitting method)

Now each stage of growth can identify a list of mesh elements for sample generation by expansion. The next problem is to identify favorable candidate elements and unfavorable elements. This can be calculated as a function of surface curvature or, more importantly, high curvature in at least one direction and low curvature in one direction. This pattern would indicate a ridge or a valley in the 3D structure.

The curvature of a triangulated (discrete) surface is not straightforward, because a single triangle only has enough information (3 points) to represent a plane (flat sheet) in 3D space. In order to assess the local curvature of the surface structure, neighboring triangles and their position/orientation must be accounted for. The most direct method to compare the curvature of a single triangle is to use linear regression to fit a quadratic surface (elliptical paraboloid) to the near-neighborhood of points (points of current triangle and neighboring triangles). Once the coefficients of the quadratic paraboloid have been identified, through linear least-square fitting, the curvature of the surface is easily identified by the eigenvectors and eigenvalues of the Hessian (partial derivative matrix) of the paraboloid. The eigenvectors represent the maximum and minimum directions of curvature while the eigenvalues reflect the magnitude of each eigenvector.

The methodology for finding the curvature for all faces in a mesh uses the following logic:

1. Choose a single mesh element to use as the base of the coordinate system
2. Calculate base vectors of the element
3. Calculate offset that transposes all coordinates to a local coordinate system whose origin lies at the mesh element center
4. Transpose all near-neighborhood points (all points connected to neighboring faces) to the new coordinate system
5. Fit a paraboloid to these points
6. Calculate eigenvalues of system
7. Report maximum and minimum eigenvalue

Theory. As an example, the coefficients of a paraboloid following the model in Equation (7.45) have been chosen and reported in Equation (7.46).

$$x = ax^2 + bxy + cy^2 + dx + ey + f \quad (7.45)$$

$$x = 1x^2 + 2x + 3y^2 + 4x + 5y + 6 \quad (7.46)$$

The method chooses points in a near vicinity in order to represent the paraboloid. The method will then evaluate the derivative matrix (also known as the Hessian matrix, Equation (7.47)). The maximum and minimum curvature of the paraboloid is defined by the eigenvalues of the matrix D . Moreover, the eigenvectors of this matrix are the directions of the maximum and minimum curvature.

$$D = \begin{bmatrix} \frac{\partial^2}{\partial x^2} z & \frac{\partial}{\partial x} \frac{\partial}{\partial y} z \\ \frac{\partial}{\partial y} \frac{\partial}{\partial x} z & \frac{\partial^2}{\partial y^2} z \end{bmatrix} \quad (7.47)$$

Relationship to system energy. One way to represent a system of equations is with the energy of the system. The energy of a system is the integral of the gradient as shown in Section 7.25.3.

The energy of a system is given by:

$$E = \frac{1}{2} x^T A x + x^T b$$

Where (7.48)

$$A = \begin{bmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{bmatrix}, \quad x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, \quad b = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

When plugging in and expanding, this results in:

$$E = \frac{1}{2} (x_1 \quad x_2) \begin{bmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + (x_1 \quad x_2) \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

$$E = \frac{1}{2} (x_1 c_{1,1} + x_2 c_{2,1} \quad x_1 c_{1,2} + x_2 c_{2,2}) \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + x_1 b_1 + x_2 b_2 \quad (7.49)$$

$$E = \frac{1}{2} (c_{1,1} x_1^2 + c_{2,1} x_1 x_2 + c_{1,2} x_1^2 + c_{2,2} x_1 x_2) + x_1 b_1 + x_2 b_2$$

Which can take a form similar to Equation (7.45) by making substitutions in naming convention:

$$\begin{aligned}
E &= \frac{c_{1,1}}{2}x_1^2 + \frac{c_{2,1}}{2}x_1x_2 + \frac{c_{1,2}}{2}x_1^2 + \frac{c_{2,2}}{2}x_1x_2 + x_1b_1 + x_2b_2 \\
E &= \frac{c_{1,1}}{2}x_1^2 + \left(\frac{c_{2,1}}{2} + \frac{c_{2,2}}{2}\right)x_1x_2 + \frac{c_{1,2}}{2}x_1^2 + b_1x_1 + b_2x_2 \\
E &= z, \quad \frac{c_{1,1}}{2} = a, \quad \left(\frac{c_{2,1}}{2} + \frac{c_{2,2}}{2}\right) = b, \quad \frac{c_{1,2}}{2} = c, \\
&\quad b_1 = d, \quad b_2 = e
\end{aligned} \tag{7.50}$$

And the final coefficient (f) is not seen in the energy surface, because the energy surface is assumed to touch the $z=0$ plane at one point. This value appears in Equation (7.45) because a generic elliptic paraboloid can have any value at its minimum.

Implementation. Equation (7.47) applied to Equation (7.45) simplifies to a matrix of constants:

$$D = \begin{bmatrix} 2a & b \\ b & 2c \end{bmatrix} \tag{7.51}$$

This derivative matrix can undergo eigenvalue decomposition to acquire the eigenvectors and eigenvalues:

$$\begin{aligned}
|D - \lambda I| &= 0 \\
\begin{vmatrix} 2a - \lambda & b \\ b & 2c - \lambda \end{vmatrix} &= 0 \\
(2a - \lambda)(2c - \lambda) - 2b &= 0 \\
\lambda^2 - 2\lambda(a + c) + 4ac - 2b &= 0 \\
c_1\lambda^2 - c_2\lambda + c_3 &= 0
\end{aligned} \tag{7.52}$$

where

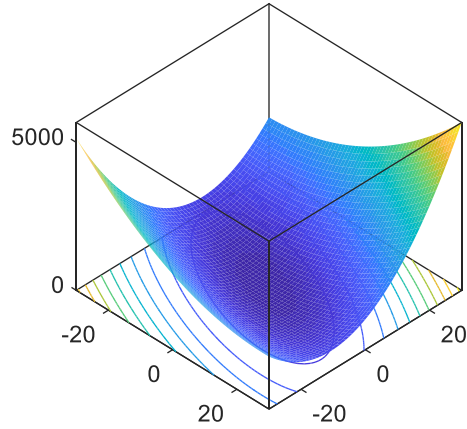
$$c_1 = 1, \quad c_2 = -2a - 2c, \quad c_3 = 4ac - 2b$$

Which has the roots (eigenvalues):

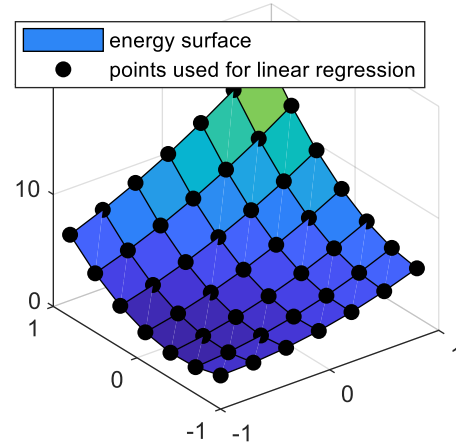
$$\begin{aligned} \lambda &= \frac{-c_2 \pm \sqrt{c_2^2 - 4c_1c_3}}{2c_1} \\ \lambda &= \frac{2(a+c) \pm \sqrt{(2a+2c)^2 - 4(1)(4ac-2b)}}{2} \\ \lambda &= \frac{1}{2} \left[2(a+c) \pm \sqrt{(4a^2 + 8ac + 4c^2) - 16ac + 8b} \right] \\ \lambda &= \frac{1}{2} \left[2(a+c) \pm \sqrt{4a^2 - 8ac + 4c^2 + 8b} \right] \end{aligned} \tag{7.53}$$

Which can be computed directly, giving the magnitude of the two eigenvalues (maximum and minimum curvature). The case study has been visually verified using Matlab and exhaustive enumeration of the sample space as in Figure 7.58. Note, the method used points sampled around the origin ($x = [-1..1]$, $y = [-1..1]$) and off-origin ($x = [-10..-9]$, $y = [20..21]$) producing the same eigenvectors and eigenvalues (data not shown). The sample points for the off-center sampling are shown below:

A)



B)



C)

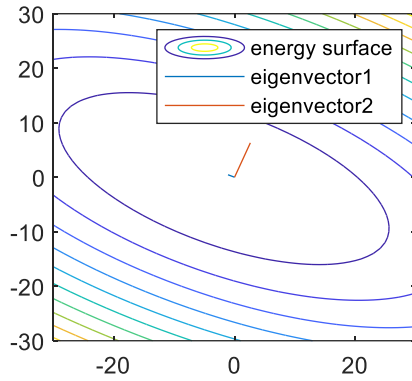
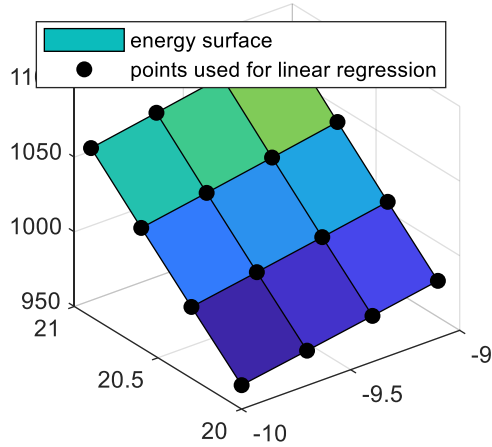


Figure 7.58. Prediction of max and minimum curvature for an analytic surface defined by a series of points using a pre-known function.

A) The energy surface of a function defined by Equation (7.46). B) a selection of points from the analytic surface used for the linear fit. Note, the linear fit reproduces the exact same coefficients as originally defined in Equation (7.46). C) Eigenvectors scaled by respective eigenvalues give the base functions of curvature for the system in Equation (7.46).

A)



B)

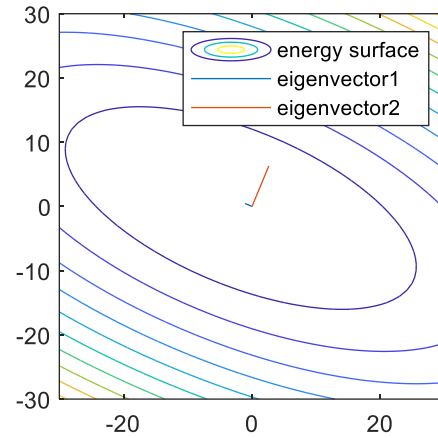


Figure 7.59. Prediction of maximum and minimum curvature for an analytic surface defined by a series of points using a pre-known function using off-center sampling.

A) a selection of points from the analytic surface used for the linear fit do not surround the origin. Note, the linear fit reproduces the exact same coefficients as originally defined in Equation (7.46). B) Eigenvectors scaled by respective eigenvalues give the base functions of curvature for the system in Equation (7.46).

Sampling a triangular surface mesh for investigating these values is also reasonable. A case study sampled points along the analytic paraboloid equation and used these points to form a series of triangles. These triangles reasonably approximates the curvature of the same function as seen in Figure 7.60.

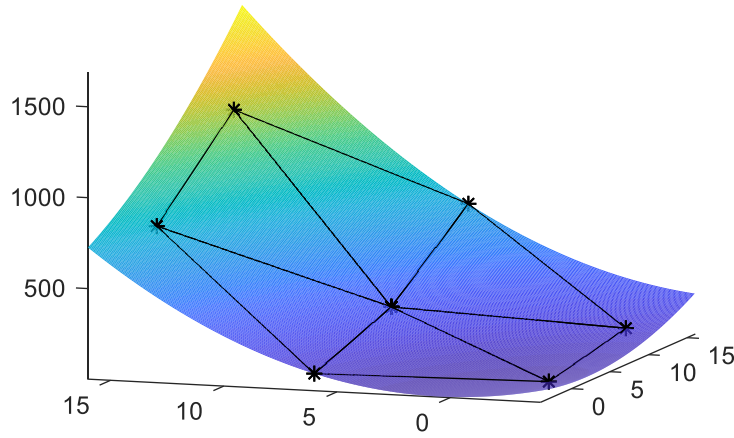


Figure 7.60. The analytic curve given by Equation (7.46) is recreated from the point coordinates of a triangular surface mesh.

Coordinate transformation and z -calculation. The global fitting of a paraboloid to a surface using Cartesian coordinates (x, y, z) can recreate the shape of the surface, but does not capture the local curvature accurately. The definition of the paraboloid and the surface derivatives in Section 7.8.2 finds the surface curvature with respect to (w.r.t.) the $Z=0$ plane. In the case of local curvature, the coordinate system must define the Z -dimension as the perpendicular distance from the current mesh element (=in the direction of the normal of the triangle). If the Z -coordinate (Z -axis) is replaced with the normal vector of the element, the values of curvature will report the desired deviation. This can be accomplished by first computing the unit vector in the normal direction (\vec{e}_n) as shown for a triangle:

$$\begin{aligned}\vec{u} &= \langle \vec{p}_2 - \vec{p}_1 \rangle, \quad \vec{v} = \langle \vec{p}_3 - \vec{p}_1 \rangle \\ \vec{n} &= \vec{u} \times \vec{v}\end{aligned}\tag{7.54}$$

$$\vec{e}_n = \frac{\vec{n}}{|\vec{n}|} = \frac{\vec{u} \times \vec{v}}{|\vec{u} \times \vec{v}|}$$

Where p_1, p_2, p_3 are the point coordinates for point 1, point 2, and point 3 in the triangle, respectively. Here, \times denotes the cross product of two vectors. Two other base vectors and an offset are also necessary to define the coordinate system. The center of the triangle will be used as the new origin, so the transformation offset (\vec{c}) will be:

$$\vec{c} = \frac{\vec{p}_1 + \vec{p}_2 + \vec{p}_3}{3} \quad (7.55)$$

The other two base vectors should be perpendicular to the normal vector (new Z-axis). To accomplish this, it is known that the vectors \vec{u} and \vec{v} , which lie in the triangle (in the plane), by definition are perpendicular to the normal vector. So the first base vector is chosen to be the unit vector in the \vec{u} direction:

$$\vec{e}_1 = \frac{\vec{u}}{|\vec{u}|} \quad (7.56)$$

And the second base vector should be perpendicular to both the normal (\vec{e}_n) and the first base vector (\vec{e}_1):

$$\vec{e}_2 = \frac{\vec{e}_n \times \vec{e}_1}{|\vec{e}_n \times \vec{e}_1|} \quad (7.57)$$

This can be visualized first on the analytic surface defined in Equation (7.46) as in Figure 7.61.

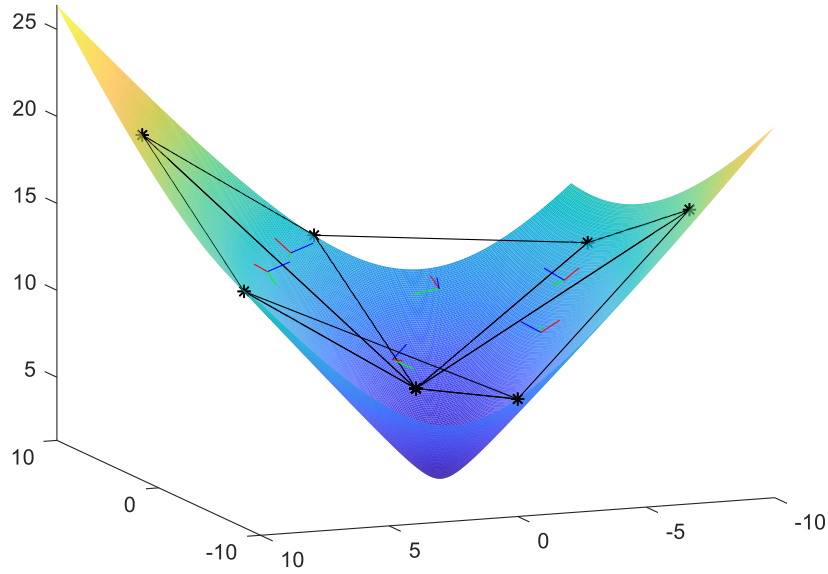


Figure 7.61. The new coordinate systems for each triangle in the mesh as it sits on the plane. For scaling purposes, the z-coordinate shown here is the square root of the Z-coordinate defined by Equation (7.46). The blue lines on each triangle correlate to \vec{e}_n , the red line is \vec{e}_1 , and the green line is \vec{e}_2 . The blue lines represent \vec{e}_n , the green lines represent \vec{e}_2 , and the red lines represent \vec{e}_1 .

Once the base vectors of the new coordinate system are identified, the coordinate transformation can take place. The transposition of any coordinate from Cartesian (x, y, z) to the new coordinates $(\vec{e}_n, \vec{e}_1, \vec{e}_2)$ requires a decision on the new origin location, here taken as the triangle center. The transposition of any coordinate from Cartesian coordinates to the new origin needs the linear transformation vector (\vec{t}):

$$\vec{t} = -\vec{c} \quad (7.58)$$

The implementation of the transformation requires the Cartesian coordinate (vector \overrightarrow{aP}) to be projected onto the new base vectors using a dot (or inner) product and translation vector to the new origin. This results in the same coordinate in the new coordinate system ($\overrightarrow{p'}$):

$$\begin{bmatrix} \overrightarrow{e_1}^T \\ \overrightarrow{e_2}^T \\ \overrightarrow{e_n}^T \end{bmatrix} \overrightarrow{aP}^T + t^T = \overrightarrow{p'}^T \quad (7.59)$$

The coordinate transformation can be shown for one triangle (the blue triangle):

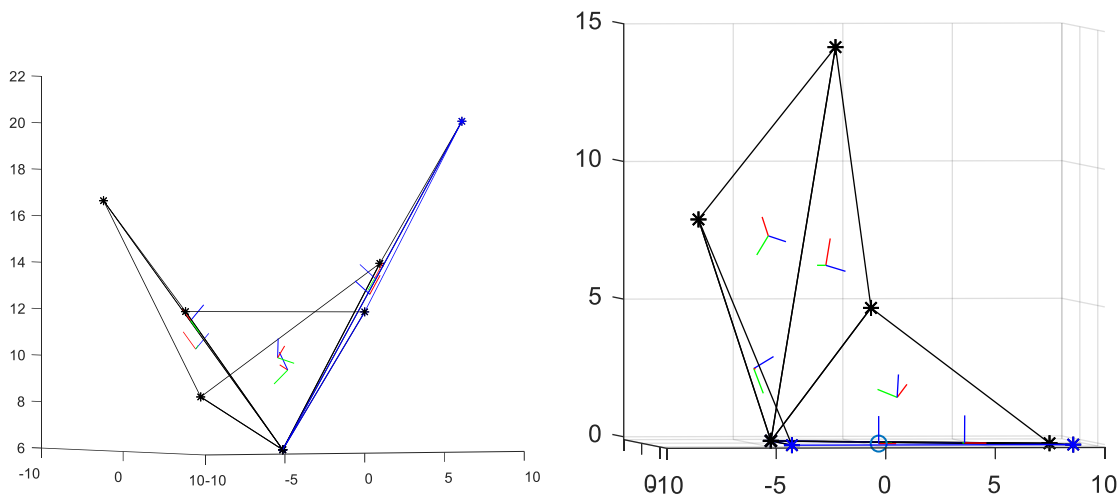


Figure 7.62. Translating the (Left) original structure in Cartesian coordinates to (Right) the coordinate system defined by the blue triangle.

The new Z-coordinate is the distance from the triangle center in the direction of the normal vector. This new set of coordinates can be used for the curvature paraboloid fitting problem. The blue circle indicates the center of the triangle, which is now the origin of the new coordinate system.

7.8.2.1 Case studies

Small section of human pial surface. A portion of a human cortex was cut out of a pial surface reconstruction and evaluated using the curvature estimation described here. The surface was colored by different variations of the curvature. The entire hemisphere was also evaluated and the surface colored by maximum curvature and sum of curvature. These methods assist in identification of the gyrations. The sum of curvature identifies the peaks of the gyrations as well.

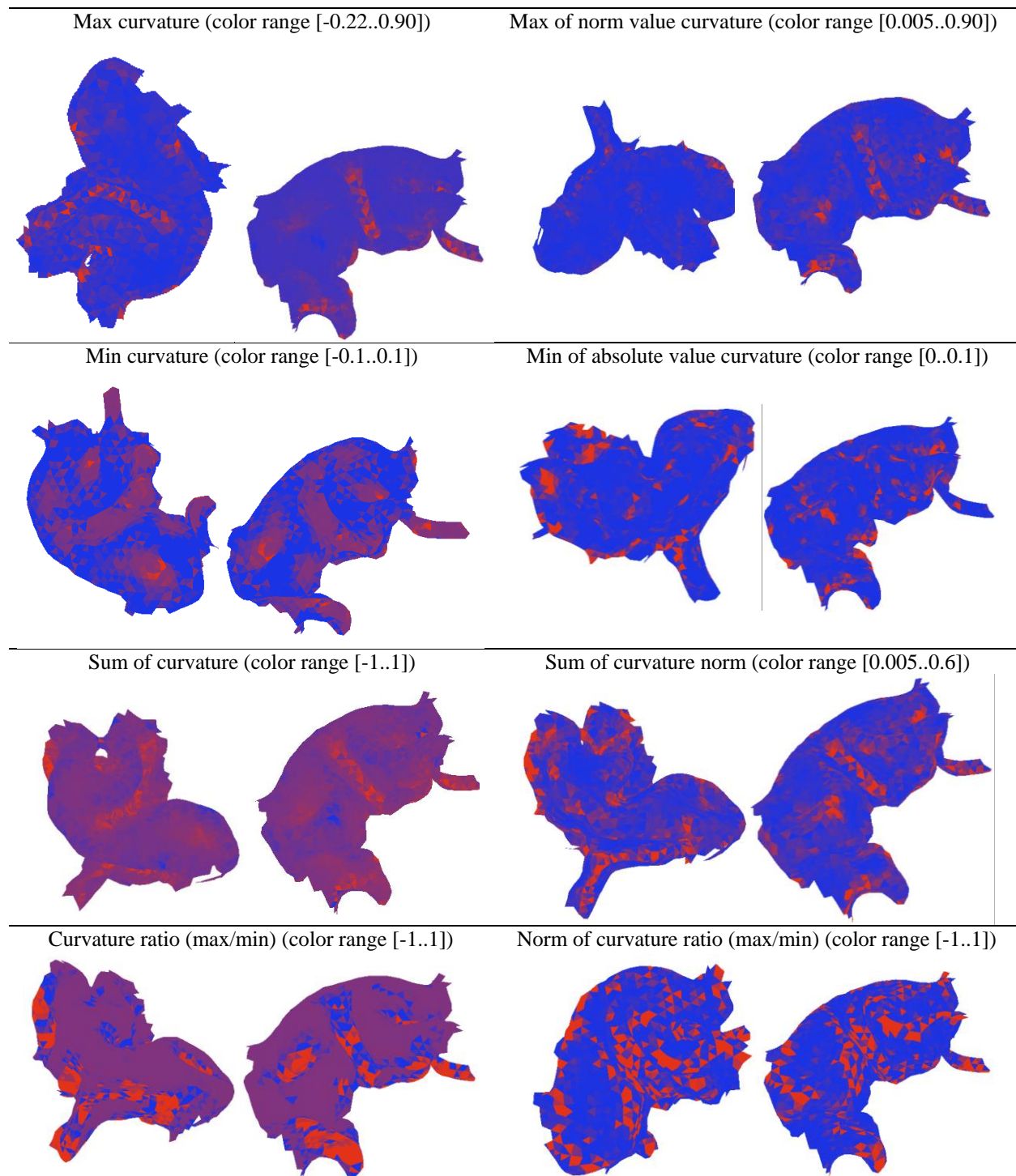
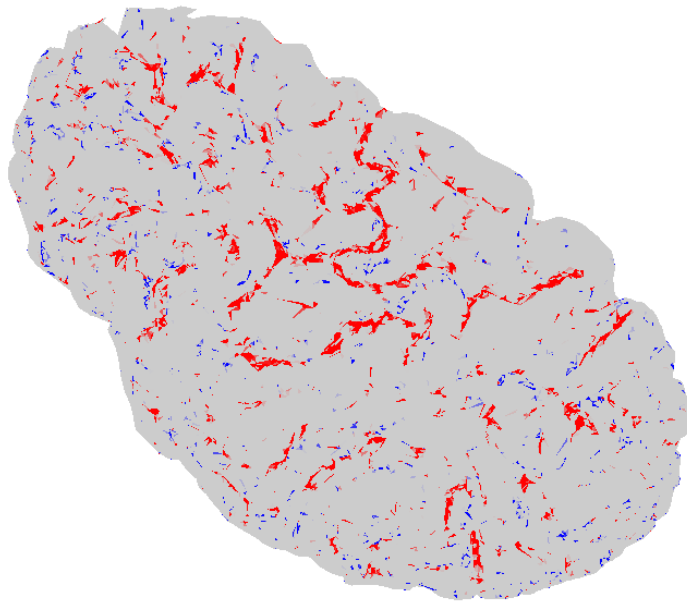


Figure 7.63. Visualization of many types of curvature calculations across a small section of the human cortical surface.

The best identification of the peaks and valleys of the surface are obtained from sum of curvature, min and max of curvature.

A)



B)

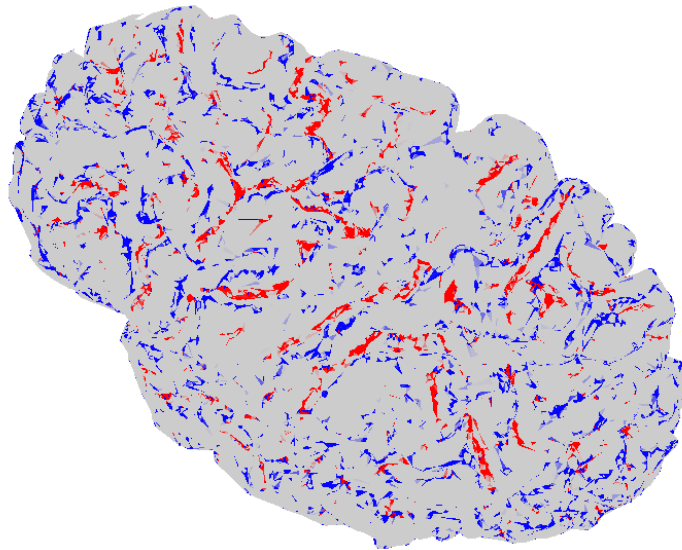


Figure 7.64. Curvature calculations on the hemisphere reveal excellent identification of ridges and valleys.

A) maximum curvature shows gyrations (valleys) and B) sum of curvature and ridges, sum of curvature). Color scale for both cases is [-1..1].

7.8.2.2 Growth along the valleys

Growing throughout the valleys requires a logic for backbone additions. Key items to consider during this phase includes:

- Vessels should be distanced outwards from the pial surface, so as not to collide with the triangular mesh
- Growth should create a tree but not a binary tree (extensions should be allowed)
- Growth should “creep” along the valleys and not cut through gyrations
- Growths should not change direction by more than 60 degrees at a single point

To follow these guidelines, the growth can begin with the current backbone (either an entire pial reconstruction backbone, or simply the MCA M1 segment). Note, if the entire pial surface reconstruction is used as a backbone, a set of starting mesh faces needs to be identified and a list of faces belonging to mesh elements already occupied by vessels must be provided manually to avoid growing new pial vessels where the backbone already exists. If only the M1 segment is used, then the growth can simply find the nearest face as the initial face selection.

Next, the ridge detection algorithm will create a mask for the surface elements (a value of 1 indicates an element from the bottom of the valley, a 0 indicates an insignificant face, and a value of 2 indicates a ridge). A list of faces with a value of 1 can be identified by simply interrogating the mask.

The growth can then expand from the initial face list by expanding multiple levels (as described in Section 7.8.1) to identify faces for new additions. From these new additions, a face selection can be chosen and a new face center (or random point in face) can be identified.

To keep the vessels above the surface, the normal of the triangle should be scaled by a random number between the radius (~50 microns) to a maximum (guessing 100 microns) from the triangle.

To add bifurcations at locations where a single gyration breaks into more than 1 gyration, a “floating island” method can be implemented. This algorithm can begin with a single element in

the selection list and expand within the current list selection, removing indices as they are identified. If the method cannot expand any further and at least one element remains on the list, there is more than 1 island of samples. This is where the algorithm can create 2 expansions and attach a bifurcation instead of a simple extension.

7.9 Appendix I: Reconstructing network geometry

7.9.1 Overview

Reconstructing vascular networks can be performed in many ways. Manual reconstruction is discussed in reference to 3D structural geometry of larger, dense structures in Section 7.10. Unfortunately, due to the denseness of the angiograms (image stacks) acquired, it is time-prohibitive to reconstruct the vasculature manually. Semi- or fully-automated anatomical reconstruction has been intensely investigated with many algorithms proposed [41,42,45,68,69]. Here, execution of 4 of these algorithms for extracting vessel centerlines from medical images will be discussed.

In order to enhance the signal-to-noise ratio of the source image stack, a vesselness filter was applied. More information on this filter can be found in another paper [68]. This algorithm is a standalone application that loads a text file with a parameter list and then loads a group of DICOM images. The result of the filtering procedure is a new set of DICOM images that have the background repressed and the vessels enhanced. This algorithm (and a sample dataset) can be found on the lab website [175]. This method uses the eigenvalues of the Hessian matrix to measure local intensity curvature. Each curvature magnitude corresponds to a unique eigenvector which is aligned to the maximum/minimum curvature. A large eigenvalue indicates a steep drop-off of intensity in one direction while a small eigenvalue indicates that Eigen-direction has low intensity curvature (flat). This will imply a longitudinal structure while the edges drop off in intensity rather quickly.

Once the images have been filtered, they can be processed by 4 methods for detecting the vascular skeleton (centerlines), (i) open source Matlab program and (ii) open source VMTK

software built on the visualization toolkit (VTK) libraries, and (iii) an open source Matlab program for centerline axis thinning, (iv) imageJ centerline extraction. A nice comparison of centerline extraction tools has been published elsewhere [176].

The theory [177] and implementation [178] of the open-source algorithm in Matlab is explained in detail elsewhere. This algorithm takes a 3D binary matrix as an input and uses integer math to identify the centerline and diameter information by marching through adjacent voxels and recording distances. One major drawback to this method is that it implies voxel dimensions of 1x1x1mm (voxel size in x, y, and z dimensions are the same). If one of the dimensions is significantly off, then this dimension should have to march twice as many cubes to account for the same physical distance, something that is not accounted for in this algorithm.

The VMTK algorithm uses a marching sphere and seed points to identify the centerline and diameter information of an image. This method pushes a sphere (with variable radius) through the isosurface (STL triangulated surface structure) and tracks the center of the sphere and radius. In order to ensure the radius is correct, it performs a method of expanding a minimal sphere until it reasonably approximates the surface structure. The triangulated surface can be generated in many ways such as marching tetrahedrons.

The Matlab program entitled centerline axis thinning uses a binary 3D matrix as input to process the centerline and diameter information. The centerline is identified by simply using Boolean operations to constrict the vessels until only a single voxel remains. These single voxels are then connected to create a centerline structure. The diameter is then approximated by an integer-stepping logic to identify the edge of the vessel.

The open-source image reviewing software entitled Fiji ImageJ also comes equipped with a centerline extraction tool. This extraction tool is optimal for identifying centerlines (but not

diameter) of single-image vascular stacks such as images of mouse retina. This tool works in 2D and in 3D alike. This tool can be run from ImageJ→Analyze→Skeleton→Skeleton 2D/3D.

Although the Matlab tools are easy to use, only requiring DICOM loading and a threshold value, they are not accurate in all cases. For instance, noisy or dense data (such as the images from the Sled group [70,179]) leads to excessive artificial arcs between close segments. This leads to a web of segments that does not accurately represent the structure contained within the image. In most cases of sparse networks with reasonable background suppression, the VMTK method is more robust. For the best results, reconstructions should be performed with all methods and compared to the original image stack using in-house tools described elsewhere [180]. To assist with execution, a walkthrough on how to use VMTK to reconstruct images is offered in the next section.

7.9.2 Reconstructing with VMTK

The steps for reconstructing image vasculature with VMTK include; (i) run *VesselFilterRelease.exe*, (ii) install and open VMTK, and (iii) run VTP case converter.

7.9.2.1 Step 1. run vesselness filter

Run *VesselFilterRelease.exe* downloaded and installed from the website or found internally at *S:/00_numberedItem/22_Pipeline/VesselFilter/VesselFilterFixed/for_testing/*. A window will pop up, select the *filterOptions* file in the same folder as *VesselFilterRelease.exe*. Next, another window will pop up. This will prompt the user to select the stack of MRA/MRV DICOM images for filtering. The process will then run for 10-15 minutes (time based on a standard desktop

for an MRA stack). It will take considerably longer for larger datasets (because of the nonlinear relationship between voxel dimension and the total voxel number).

When the process is finished, two images will pop up. One is the maximum intensity projection (MIP) of original data, and the other is the MIP of the filtered data (for verification purposes). If the filter did not work correctly, adjust the filtration parameters in the *filterOptions* file.

A new directory (folder) will be made in the root folder (the same folder as *VesselFilterFixed.exe*) named *FilteredImages*. The new images will be written to this directory in DICOM format, although they will not have any file extension. This helps delineate the processed files from the originals.

7.9.2.2 Step 2. Run VMTK

The next step can proceed after installation of VMTK. Instructions for this can be found online. A directory must be created to put the new images in after processing. The following code can be appended by replacing the red portion with the new directory name. Extra attention must be paid to using forward slashes for directory names as backslashes (default in Windows) will not process in the compiler. An additional code is offered to obtain an STL file also from this pipeline. This code (lower box) must be inserted into the main code (upper box) between the 2nd and 3rd block of code. The directory in blue is where the filtered images from Step 1 are located.

```
vmtkimagereader -f dicom -d "C:/Users/LPPD-IS/Desktop/David  
Folder/VesselFilter/VesselFilterRelease/for_testing/FilteredImages/" --pipe vmtklevelsetsegmentation -ofile  
"C:/Users/LPPD-IS/Desktop/6 Subjects/Filtered6Subjects/KTVMTK/LS2.vti"  
  
vmtkimageviewer -ifile "C:/Users/LPPD-IS/Desktop/6 Subjects/Filtered6Subjects/CHVMTK/LS2.vti" --pipe  
vmtkmarchingcubes -l -0.0 -connectivity 1 -ofile "C:/Users/LPPD-IS/Desktop/6  
Subjects/Filtered6Subjects/KTVMTK/MC2.vtp"
```

```
vmrksurfacecapper -ifile "C:/Users/LPPD-IS/Desktop/6 Subjects/Filtered6Subjects/CHVMTK/MC2.vtp" --pipe  
vmrksurfaceclipper --pipe vmtknetworkextraction -ofile "C:/Users/LPPD-IS/Desktop/6  
Subjects/Filtered6Subjects/KTVMTK/CL2.vtp"  
  
vmtkcenterlineviewer -ifile "C:/Users/LPPD-IS/Desktop/6 Subjects/Filtered6Subjects/KTVMTK/CL2.vtp"
```

```
vmrksurfacereader -ifile "C:/Users/LPPD-IS/Desktop/6 Subjects/Filtered6Subjects/KTVMTK/MC2.vtp" --pipe  
vmrksurfacewriter -ofile "C:/Users/LPPD-IS/Desktop/6 Subjects/Filtered6Subjects/KTVMTK/MC2.stl"
```

The following image will pop up if the code compiled correctly. Note, if instead of this image, a red 2D plane is presented, the internal information in the DICOM header is incorrect and must be cleaned. An internal program can be used in Matlab located in *S:\00_numberedItem\20_Software\Data Converters\DicomToDicom_infoRewrite.m* to rewrite voxel dimensions.

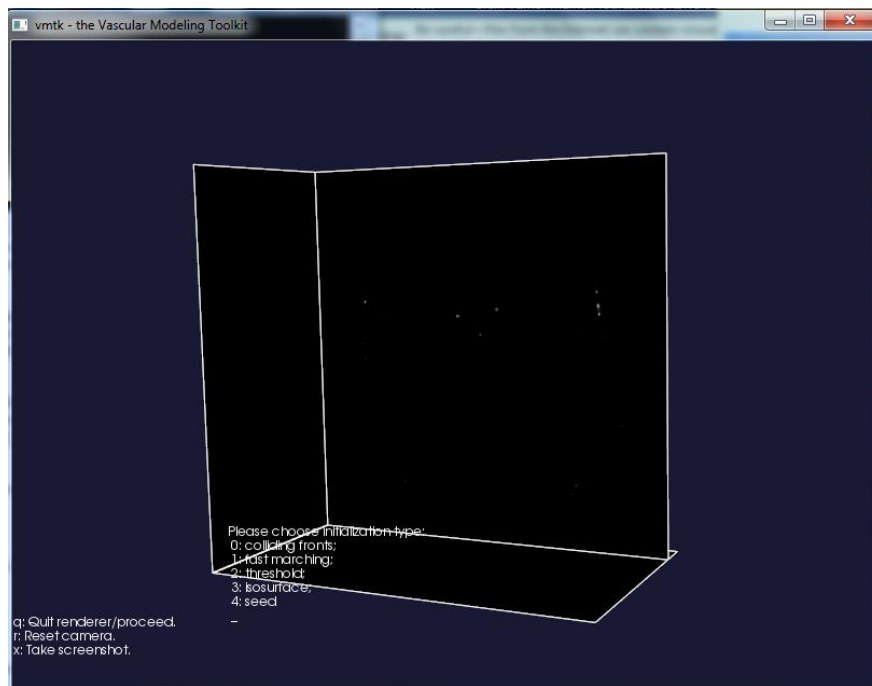


Figure 7.65. Screenshot of the VMTK software after successful compiling of the code and execution of the first lines.

Next, press 1 for *Fast Marching Cubes*. The next step involves setting an upper and lower threshold. This can be obtained during the current step by scrolling through images, clicking on them, and recording values of background noise and desirable voxels. The lower threshold should be appropriately high to eliminate subtle background noise that may not be visible in the viewer. A choice of ~200 is suggested for standard MRA images that have been filtered through Step 1. The next step is to type in the lower threshold value and press *enter*. The process is repeated for the upper threshold. Note, a value of n may be used for the upper threshold in lieu of a value if there is not an upper bound to the intensity of desirable voxel intensities.

The next step, in cerebrovascular imaging data, is to find (towards one of the ends of the stack) three white spots that pertain to the basilar (BA) and carotid arteries (LCA and RCA). The user must hold the *control* (*CTRL*) button while simultaneously clicking on these white spots to place three red spheres. These spheres will be the seed points used to initialize the segmentation procedure.

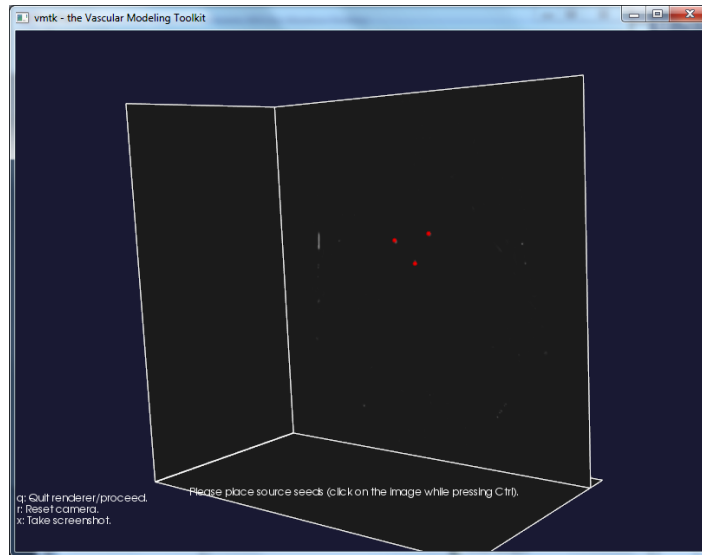


Figure 7.66. Screenshot of the VMTK software after successful identification of the BA, LCA, and RCA and placement of red spheres on these locations.

The next steps are straightforward:

1. Press *q* twice
2. If the new figure is correct, press *y* and then press *enter*
3. If all branches have been captured, press *n*. Otherwise, click *yes* and add additional seed points and repeat the segmentation.
4. Press *enter* twice
5. Press *y* and then *enter*
6. Press *y* and then *enter*
7. Press *n* and then *enter*

At this point, the current screen will close and a new one will pop up. The user must then press *q* again. Note, if vessels have been cut off, the selection of *-connectivity 1* in the provided code (upper box above) must be omitted as it only shows largest connected segments.

Next, the user must type one or more of the numbers on the screen with a space between them (like *0 1 2*) before pressing enter. Note, the selection of numbers does not seem to produce different structures, so any selection will work. In the event no numbers are visualized on the screen, the user must modify the provided code to replace *vmtksurfacecapper* with *vmtksurfaceviewer* and rerun the code.

The user must then press *I* to interact with the new structure in the provided interface. The next step requires placement of a box over the end of the carotid artery and pressing *space* to cut the end of this vessel. It is important that the box cut perpendicular to the vessel for best results. The user then presses *q* again.

7.9.2.3 Step 3. run VTP-to-case/nwk converter

At this point, three files have been created. A proprietary C++ code has been developed by the LPPD to convert the new files into the internal mesh file format used in the lab. To run this C++ code, the user must locate the source code in the directory *VMTK2NWK* located on internal servers at *S:\00_numberedItems\22_Pipeline_CY*. This directory must be copied to the local hard drive. The three files generated previously (CL2.vtp, centerline info, LS2.vti, dicom info, and MV2.vtp, marching cubes info) need to be copied into *VMTK2NWK/executable/* directory. The user can then run load the *centerlines.sln* file from the *VMTK2NWK/bin* directory into Visual Studio and run it. For ease, the user may also run only the executable to produce a new case and network file.

To run with Visual Studio, the user must place the three files into */bin* instead of */executable*. At this point, if on the right hand side in the “solution explorer” the line “centerline” is not bolded, so the user must double-click it to set it as the startup project. The project can then be compiled and run in debug mode.

A command window will pop up and prompt the user for the marching cubes file. The user will respond by typing *MC2.vtp* and hit *enter*. After the prompt for the centerline, the user should type *CL2.vtp* and hit *enter*.

Once the system has completed its operation a new window will pop up for review. If the information is correct, the user can close the window. The new case and network files will be found in the directory *VMTK2NWK/bin/data/*.

The user must then open the case file and include a line that reads *<meshfile>* above the line reading *meshfile=*. The user must also delete the *Data/nwk/* text in the meshfile name.

7.10 Appendix J: Mesh reconstruction

Anatomical reconstructions can be performed with the help of automated software, such as Freesurfer [181], or manually using manual segmentation tools such as ITK-Snap [182]. These structures accurately capture the topology of the surface gyrations and interior voids, such as the ventricles. This lends to a detailed 3-dimensional geometry of the pial surface and white matter which can be exported as a stereo lithography (STL) file. This file is then amenable to meshing through tools such as ICEM (Geneva, CH) which can make a 3D volumetric mesh from a surface mesh using a method similar to Voronoi tessellation which simply fills the space within the 3D region with points and connects the points to the neighbors until all volumes between points is accounted for.

The images for reconstruction can be acquired from either an MRI imaging machine or similar machine capable of obtaining the grey/white matter interface with high signal-to-noise-ratio (SNR). This 3D imaging stack must be in DICOM format for proper reconstruction with previously mentioned tools. Once the DICOM images have been reconstructed, the anatomical regions in the brain can be meshed and labeled using ICEM. An example of this can be seen in Figure 7.67.

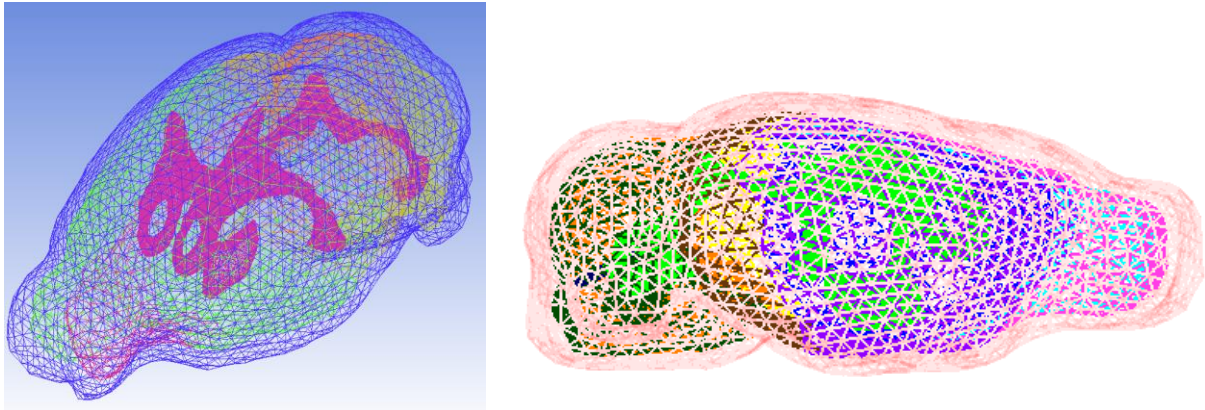


Figure 7.67. Anatomical reconstruction showing a mouse brain in two viewpoints. Reconstruction was performed with ITK-Snap and meshing/labeling was performed by ICEM.

This method of growth is great for obtaining realistic geometry, however it suffers from a large number of volume elements (>10 million). Moreover, the quality of mesh at sharp interfaces, such as the CSF-white matter interface, needs significantly more volume elements. The best way to solve problems of this scale is to use the commercial software ANSYS (Canonsburg, PA) as was used in a tumor diffusion problem [183].

7.11 Appendix K: Network analysis

7.11.1 Modeling with networks

Network analysis is a powerful tool for reducing 3-dimensional vascular systems to a lower dimensional structure (radially symmetric tubes) that can represent the 3D geometry. These one dimensional networks are represented by nodes and arcs with accompanying diameters. This method simplifies the partial derivatives in multiple dimensions in ODEs and PDEs to a single dimension. This facilitates the simulation of large geometric structures such as every vessel in an entire mouse hemisphere. Network analysis often implies simplifying assumptions. One such assumption is that the arcs can be represented by a series of adjacent cylinders as opposed to a structure with a non-circular cross sectional. This section will summarize how to use network analysis to solve blood flow equations for cerebral vascular trees.

Stationary blood flow can be predicted using the Hagen Poiseuille (HP) equations (derived in Section 7.11.4). These equations can be expressed in a compact matrix form using incidence matrices that relate resistance, flows, arc connectivity, and pressure drop. By using these matrices, the problem of solving flow equations is simply converting the graph structure into suitable connectivity matrices (C_1 and C_2) that are directly amenable to conservation balances. This approach is similar to the cut matrices used by Tellegen's theorem [131] which formalizes the relationship between adjacency matrices, cut matrices, and conservation balances.

The proposed network formulation is also applicable to DC electric circuits by merely replacing the HP resistance with suitable electrical resistances. It also applies to heat conduction and diffusion problems as long as the domain can be represented by a network of nodes and arcs

and each arc can be endowed with a characteristic resistance. This formulation can thus be considered a universal representation of any diffusion problem.

7.11.2 Problem Formulation and solving

Using network analysis to solve physical problems requires minimal steps to set up and execute the simulation. The steps include (i) identifying the geometry, (ii) describe the states and flux equations, (iii) apply flux to arcs, (iv) apply conservation to flux expressions, (v) identify boundary conditions and finally (vi) solve linear algebraic system of equations.

Identify geometry. In order to simulate a physical domain, the geometry of the domain must be clearly defined. This can be performed from with idealized geometries, image reconstruction as discussed in Section 7.9, or with synthetic geometries as discussed in Section 2-3. In order to store the network to a file, sparse matrix format is recommended such as those reviewed in an internal report [184].

Identify the physics of the system. The governing states (heat, energy, moles, etc.) and the relevant flux equations must be identified using physical knowledge and empirical relationships that define the system. Using the laws of transport phenomena, the flux of extensive properties can be computed using first principles such as diffusion and convection. An example of a diffusion problem with isotropic diffusivity constant D is given in Equation (7.60), although this value can vary between arcs, as exists in blood flow computations. Here, ϕ is a scalar field representing extensive property values at each node. f is the vector field of flows across the corresponding arcs.

$$C_1\phi = Df \tag{7.60}$$

Apply constitutive equation to each arc. Now that the equations and states have been identified, the domain consisting of points and arcs can be used for equation formulation. Each flux/flow equation must be applied to each arc in the system. The connectivity matrices stored while identifying the geometry should give the node-arc relationship. The connectivity matrix C_1 is generated in this step and should be encoded as a sparse matrix to save memory overhead. It is common practice use the arc index as the row index in the C_1 matrix.

Apply conservation to each node. Extensive properties (heat, moles, mass, etc.) must be conserved in a system. To conserve these properties, a new connectivity matrix (C_2) needs to be defined following Equation (7.61). This matrix is formulated by identifying all arcs connected to a given node, and expressing the flows of these nodes all sum to a value of 0. Note, this matrix will need to be appended at terminal nodes with boundary condition equations. It is common practice that the row index of the C_2 matrix is the same as the node index in the graph. It is worth noting that the C_2 matrix is equivalent to $-C_1^T$ except for the boundary terminal equations.

$$C_2 f = 0 \quad (7.61)$$

Solve the linear algebraic system. In steady-state simulations, Equations (7.60)-(7.61) should constitute a linear (or in some cases nonlinear) algebraic, non-singular, set of equations which can be solved with iterative or direct methods. Briefly, direct methods such as Gaussian elimination (MA48 is recommended for efficiency and robustness [185]) and iterative linear algebraic solvers (recommended to use GMRES with a preconditioner such as Block Jacobi, both available through open-source PETSc [186]) are openly available. These solvers are further discussed in another report [187]. In the case of a dynamic system, the discretized ODEs can be solved using knowledge

from the eigenvalue report [188]. PDEs can be solved using knowledge from the collocation report [189] and the analytic solution to the wave equation report [190]. In the event of a PDE with only time and spatial discretization, the time can be numerically integrated as discussed in Section 7.31.

Example. A simplified network can be constructed as in Figure 7.68. This network will be used to simulate blood flow. HP flow will be applied to each arc to relate our state variable (pressure) to volumetric flow rate as a function of the arc resistance (α_i) as described by Equation (7.62). Mass conservation will be applied at each node as described in equation (7.63). These equations can be written in matrix form as in Equations (7.64)-(7.66) in both component and compact form. The problems can also be formulated in a more general sense as originally proposed by Tellegen [131].

$$\Delta P = \alpha F \quad (7.62)$$

$$F_{in} - F_{out} = 0 \quad (7.63)$$

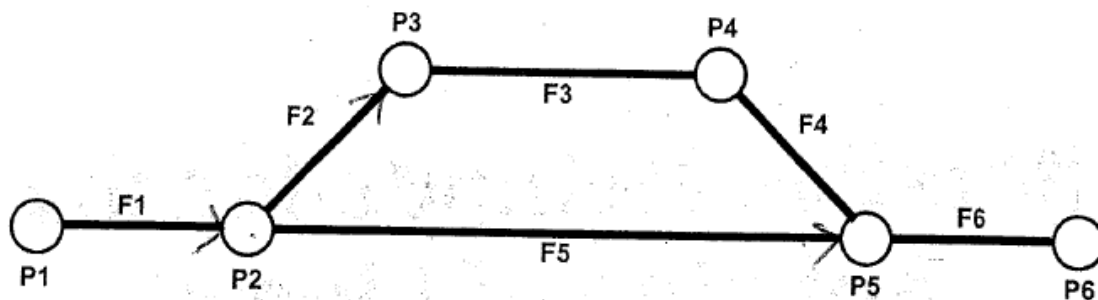


Figure 7.68: A simplified network of nodes (P) and arcs (F).

$$\begin{aligned}
C_1 &= \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & 1 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix} \\
C_2 &= \begin{bmatrix} 1 & -1 & 0 & 0 & -1 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & -1 \end{bmatrix}
\end{aligned} \tag{7.64}$$

$$\alpha = \begin{bmatrix} \alpha_1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \alpha_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \alpha_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & \alpha_4 & 0 & 0 \\ 0 & 0 & 0 & 0 & \alpha_5 & 0 \\ 0 & 0 & 0 & 0 & 0 & \alpha_6 \end{bmatrix}$$

$$C_1 p = \alpha f \tag{7.65}$$

$$C_2 f = 0 \tag{7.66}$$

7.11.3 Tellegen theorem

One alternative method for solving network problems is by using Tellegen's theorem, published in 1952 by Bernard Tellegen, which uses Kirchhoff's laws of electrical circuit theory [131]. The basic assumptions in such a network are the conservation of extensive properties, such as flows (via Kirchhoff's current law), and the state potential at the nodes of the network (Kirchhoff's voltage law). The general form for a given electrical circuit is given by equation (7.67) and it states that the energy stored in a circuit must be equal to the power consumed/dissipated ($P = V \cdot I$) and any source power from the terminals ($P_{e,i}$). More on this derivation can be found in a previously published thesis [191].

$$\underbrace{\sum_{i=1}^{n_c} \frac{dQ_{e,i}}{dt} V_{e,i}}_{\text{Energy Stored (capacitance)}} = - \underbrace{\sum_{i=1}^{n_f} I_{ij} \Delta V_{e,ij}}_{\text{Power Loss/Dissipation}} + \underbrace{\sum_{i=1}^{n_f} P_e}_{\text{boundary sources (terminals)}} \quad (7.67)$$

7.11.4 Linear flow (Hagen Poiseuille)

The Hagen Poiseuille (HP) model for 1-dimensional simulation of blood flow has been used for over a century and is considered commonplace when calculating flow through a tube. While blood flow through a vessel is more complicated than single-phasic liquid flowing through a tube, this model has the advantages of giving a linear relationship between pressure and flow. This is in opposition to biphasic blood flow (red blood cells suspended in plasma) which relates pressure and flow nonlinearly. The HP model has been verified both experimentally and mathematically.

Derivation. The Navier-Stokes (NS) equations involve the balancing of 3 states in order to maintain continuity, conserve energy, and conserve mass. This set of equations been accepted as the gold-standard for predicting fluid flow in a 3D system. The HP equations can be derived from the NS equations beginning with the momentum continuity portion of the Navier Stokes equations (in radial coordinates), assuming incompressible fluid and no-slip boundary condition at the wall:

$$\text{accumulation} + \text{inertia} = \text{sources} = \text{driving force} + \text{shear loss}$$

$$\rho \frac{dv}{dt} + \rho v \nabla v = -\Delta p + \nabla \cdot \tau \quad (7.68)$$

Where v is velocity, ρ is density, p is pressure, t is time and τ is the stress tensor In cylindrical coordinates:

$$\tau = -\mu \left. \frac{dv}{dr} \right|_{r=R} \quad (7.69)$$

Where R is the radius of the vessel, μ is viscosity and r is radius. At low Reynold's number (viscous forces \gg convective forces), as is assumed in the case of blood vessels in the brain, inertia can be neglected. At steady-state, the d/dt terms become 0. Implementing these steps and simplifying, equation (7.68) becomes (in radial coordinates):

$$\begin{aligned} -\Delta p &= \mu \nabla \cdot \nabla v \\ &= \mu \frac{1}{r} \frac{d}{dr} \left(r \frac{dv}{dr} \right) \end{aligned} \quad (7.70)$$

Where:

$$v = \frac{Q}{A} = \frac{Q}{\pi r^2}$$

Where Q is the volumetric flow rate. Considering diffusion in cylindrical coordinates:

$$\nabla \cdot \nabla v = \frac{1}{r} \frac{d}{dr} \left(r \frac{dv}{dr} \right) + \frac{1}{r^2} \frac{d^2 v}{d\theta^2} + \frac{d^2 v}{dz^2} \quad (7.71)$$

Where z is the longitudinal direction. If symmetry is assumed around θ (i.e. $dQ/d\theta=0$), and at the wall, z directional velocity has a magnitude of 0 (due to the imposed no-slip boundary condition) then this simplifies to:

$$\nabla \cdot \nabla v = \frac{1}{r} \frac{d}{dr} \left(r \frac{dv}{dr} \right) \quad (7.72)$$

Equation (7.70) is rewritten as:

$$\begin{aligned} -\Delta p &= \mu \frac{1}{r} \frac{d}{dr} \left(r \frac{dv}{dr} \right) \\ \int_r -r \Delta p \, dr &= \mu r \frac{dv}{dr} \\ -\frac{1}{2} r^2 \Delta p &= \frac{\mu}{\pi r} \frac{dv}{dr} \\ -\frac{1}{2} r^2 \Delta p &= \frac{\mu}{\pi r} \frac{dv}{dr} \\ \int_r -\frac{1}{2} r^3 \Delta p \, dr &= \mu \frac{dv}{dr} \\ -\frac{1}{8} r^4 \Delta p &= \mu \frac{dv}{dr} \\ -\frac{1}{8} r^4 \Delta p &= \frac{\mu}{\pi r^2} dQ \\ -\Delta p &= \frac{8\mu}{2\pi r^4} dQ \end{aligned} \quad (7.73)$$

Which, when applied to a network, the loss term (right hand side) needs to be integrated over the length of the vessel which gives the HP equation:

$$\begin{aligned}
 -\Delta p &= \int_l \frac{8\mu}{2\pi r^4} dQ \, dl \\
 -\Delta p &= \frac{8\mu l}{2\pi r^4} dQ
 \end{aligned}
 \tag{7.74}$$

Where l is the length of the vessel. Note, the viscosity term is an empirical value that can either be a static value, or a diameter-dependent value as described later in the context of biphasic blood flow (see Section 7.12.1).

7.11.4.1 Accounting for turbulent flow in HP

Modeling non-laminar flow or non-cylindrical flow is summarized by Equation (7.75) with corresponding coefficients in

Table 7.14 as originally printed in Bird [192]. R_t is the tube resistance, R_d is the disturbance resistance, and E_f is loss due to friction in an energy balance. R is the hydraulic resistance in the vessel, L is the vessel length, f is the friction along the vessel, v is the velocity of the fluid in the vessel, and e_v is the frictional factor as described by

Table 7.14.

$$E_l = R_t + R_d$$

(7.75)

$$E_l = W - \sum \frac{1}{2} v^2 \frac{L}{R} f - \sum \frac{1}{2} v^2 e_v$$

Table 7.14: Summary of frictional losses due to turbulent flow in a tube

Disturbance	Friction factor (e_v)
Changes in cross sectional area	
Rounded entrance to pipe	0.05
Sudden contraction	$0.45(1-\beta)$
Sudden expansion	$\left(\frac{1}{\beta} - 1\right)^2$
Orifice (sharp-edged)	$2.71(1 - \beta)(1 - \beta^2)$
Fittings and valves	
90° elbows (rounded)	0.4-0.9
90° elbows (square)	1.3-1.9
45° elbows	0.3-0.4
Globe valve (open)	6-10
Gate valve (open)	0.2

7.11.4.2 Problem simplification

The previously formulated state-flux (pressure-flow) problem in Section 7.11.1 gives a representation in both unknown pressures and unknown flows. With the help of substitution, this problem can be reduced by roughly half by eliminating the flows and solving for only pressures. This is only one kind of simplification to reduce problem size and make such large networks more tangible for solving.

7.11.5 Examples of Network Analysis

Case study 1, simplified cerebral circulation model. Figure 7.69 depicts a simplified vasculature model. For each point, a mass balance (Equation (7.66)) must be performed. The Hagen-Poiseuille law (Equation (7.65)) is applied to relate the flow through a tube with the resistance and the pressure drop across it. The values for α are listed in Table 7.15.

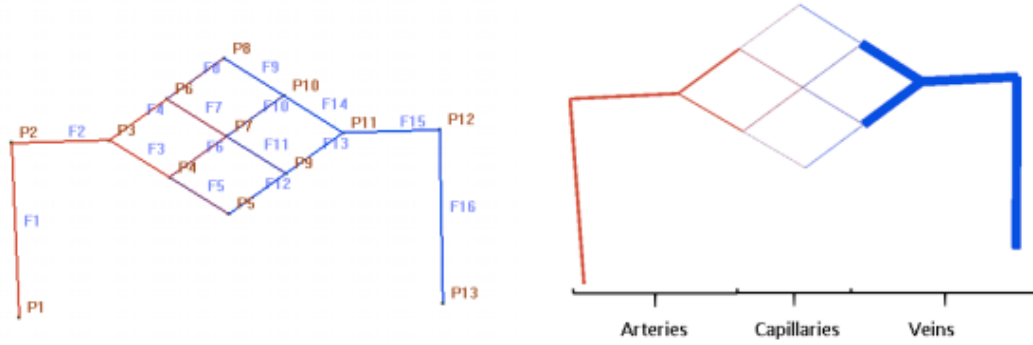


Figure 7.69: Simplified Cerebral Angioarchitecture (flow is from left to right)

Table 7.15. Segment resistance for simulating the network in Figure 7.69

Face index	Resistance [mmHg min/mL]	Face index	resistance [mmHg min/mL]
1	22423	9	2299668
2	12457	10	2299667
3	143729	11	2299668
4	28391	12	2299667
5	17744330	13	69
6	2299666	14	69
7	2299665	15	96
8	17744339	16	173

The equation inventory for each face and node in the network are listed in Table 7.16.

Table 7.16. Inventory of equations used for simulating the network in Figure 7.69

Nodes	Mass conservation	Hagen-Poiseuille Equations
P ₂	$F_1 - F_2 = 0$	$F_1 \alpha_1 = P_1 - P_2$
P ₃	$F_2 - F_3 - F_4 = 0$	$F_2 \alpha_2 = P_2 - P_3$
P ₄	$F_3 - F_5 - F_6 = 0$	$F_3 \alpha_3 = P_3 - P_4$
P ₅	$F_5 - F_{12} = 0$	$F_4 \alpha_4 = P_3 - P_6$
P ₆	$F_4 - F_7 - F_8 = 0$	$F_5 \alpha_5 = P_4 - P_5$
P ₇	$F_6 + F_7 - F_{10} - F_{11} = 0$	$F_6 \alpha_6 = P_4 - P_7$
P ₈	$F_8 - F_9 = 0$	$F_7 \alpha_7 = P_6 - P_7$
P ₉	$F_{11} + F_{12} - F_{13} = 0$	$F_8 \alpha_8 = P_6 - P_8$
P ₁₀	$F_9 + F_{10} - F_{14} = 0$	$F_9 \alpha_9 = P_8 - P_{10}$
P ₁₁	$F_{13} + F_{14} - F_{15} = 0$	$F_{10} \alpha_{10} = P_7 - P_{10}$
P ₁₂	$F_{15} + F_{16} = 0$	$F_{11} \alpha_{11} = P_7 - P_9$
		$F_{12} \alpha_{12} = P_5 - P_9$
		$F_{13} \alpha_{13} = P_9 - P_{11}$
Nodes	Boundary value (mmHg)	
P ₁	100	

 P_{13}

5

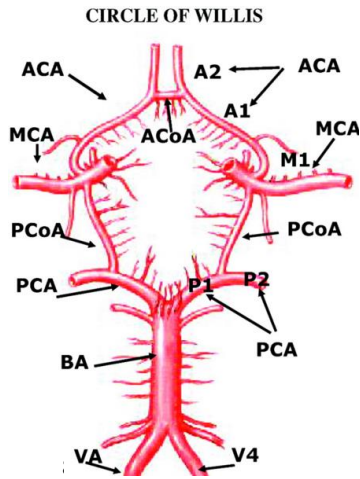
$$F_{14}\alpha_{14} = P_{10} - P_{11}$$

$$F_{15}\alpha_{15} = P_{11} - P_{12}$$

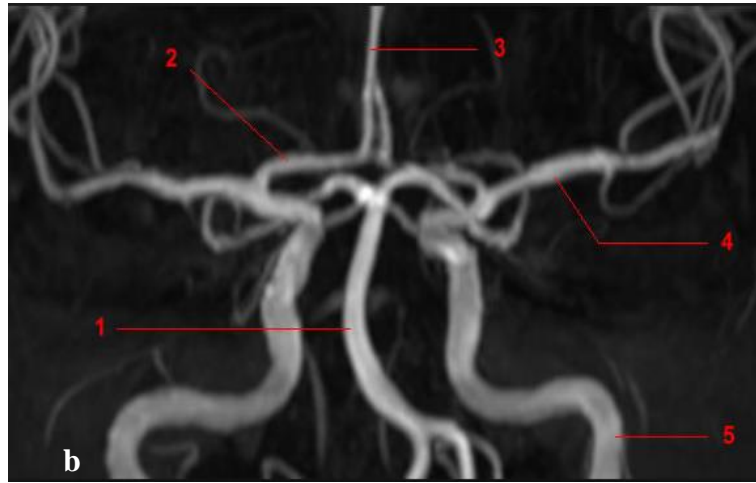
$$F_{16}\alpha_{16} = P_{12} - P_{13}$$

Case study 2, Circle of Willis model. In the first portion of the assignment we are to use mass (flow) and momentum (HP) conservation equations to compute the flow in an enclosed Circle of Willis (CoW) network (Figure 7.70).

A)



B)



C)

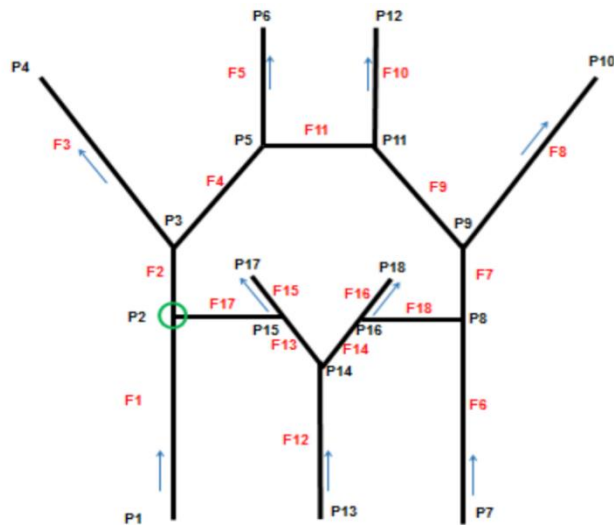


Figure 7.70: Representations of the human Circle of Willis (CoW).

A) Cartoon image of labeled CoW diagram. B) Maximum intensity projection of CoW anatomy in a human. C) Schematic figure of CoW in node and arc format. All the arcs are numbered in red, and all the points (nodes) are labeled in black. The blue arrows represent the direction of the flow assigned for all the network equations.

For each point, a node balance (Equation (7.62)) must be performed, accounting for all the flows going in and out. Then, for every face, the HP equations (Equation (7.63)) is applied to relate the flow and pressure drop.

Another way of solving this system of equations from is to solve the simplified system described in Section 7.11.4.2. From Equation (7.63) the flow may be rewritten as in Equation (7.76). This simplifies to equation (7.77) and Equation (7.62) can be written as equation (7.78), where only the unknown pressure must be calculated and the system is smaller (more easily solved).

$$C_1 p = Rf \Rightarrow R^{-1} C_1 p = f \quad (7.76)$$

$$\underbrace{C_2 R^{-1} C_1}_{C_4} p = 0 \quad (7.77)$$

$$\begin{bmatrix} C_4 \\ C_3 \end{bmatrix} [p] = \begin{bmatrix} 0 \\ \bar{p} \end{bmatrix} \quad (7.78)$$

Writing Network Problem in Matrix Format. First, for smaller systems, all equations may be written by hand, in order to validate computations. These equations are listed in Table 7.17. The first step in writing the equations is defining a direction for each arc in the network. Once a direction is chosen, the HP equation (Equation (7.63)) can be applied to all faces in the system to create Equations (7.79)-(7.104).

Table 7.17: Hagen Poiseuille law applied on each face in the CoW

$P_1 - P_2 = \alpha_1 F_1$	(7.79)	$P_8 - P_9 = \alpha_7 F_7$	(7.80)
$P_2 - P_3 = \alpha_2 F_2$	(7.81)	$P_7 - P_8 = \alpha_6 F_6$	(7.82)
$P_3 - P_4 = \alpha_3 F_3$	(7.83)	$P_{13} - P_{14} = \alpha_{12} F_{12}$	(7.84)
$P_3 - P_5 = \alpha_4 F_4$	(7.85)	$P_{14} - P_{15} = \alpha_{13} F_{13}$	(7.86)
$P_5 - P_6 = \alpha_5 F_5$	(7.87)	$P_{15} - P_{17} = \alpha_{15} F_{15}$	(7.88)
$P_5 - P_{11} = \alpha_{11} F_{11}$	(7.89)	$P_{15} - P_2 = \alpha_{17} F_{17}$	(7.90)
$P_{11} - P_{12} = \alpha_{10} F_{10}$	(7.91)	$P_{14} - P_{16} = \alpha_{14} F_{14}$	(7.92)
$P_9 - P_{11} = \alpha_9 F_9$	(7.93)	$P_{16} - P_{18} = \alpha_{16} F_{16}$	(7.94)
$P_9 - P_{10} = \alpha_8 F_8$	(7.95)	$P_{16} - P_8 = \alpha_{18} F_{18}$	(7.96)

Because there are nine interior points, nine flow balances may be obtained (odd numbered equations from Equations (7.97)-(7.114)) using conservation of mass (Equation (7.62)). The boundary conditions are the even numbered equations from Equations (7.97)-(7.114).

Table 7.18. Flow Balances and Dirichlet Boundary for each point in the CoW

$F_1 + F_{17} - F_2 = 0$	(7.97)	$P_1 = \bar{P}_1 = 100$	(7.98)
$F_2 - F_3 - F_4 = 0$	(7.99)	$P_{13} = \bar{P}_{13} = 100$	(7.100)
$F_4 - F_5 - F_{11} = 0$	(7.101)	$P_7 = \bar{P}_7 = 100$	(7.102)
$F_{11} + F_9 - F_{10} = 0$	(7.103)	$P_4 = \bar{P}_4 = 50$	(7.104)
$F_7 - F_8 - F_9 = 0$	(7.105)	$P_6 = \bar{P}_6 = 50$	(7.106)
$F_6 + F_{18} - F_7 = 0$	(7.107)	$P_{12} = \bar{P}_{12} = 50$	(7.108)
$F_{12} - F_{13} - F_{14} = 0$	(7.109)	$P_{10} = \bar{P}_{10} = 50$	(7.110)
$F_{13} - F_{17} - F_{15} = 0$	(7.111)	$P_{17} = \bar{P}_{17} = 50$	(7.112)
$F_{14} - F_{16} - F_{18} = 0$	(7.113)	$P_{18} = \bar{P}_{18} = 50$	(7.114)

Between equations (7.79)-(7.96) and equations (7.97)-(7.114), there are 36 linear algebraic equations to be solved. In this report, the system will be written in matrix form (part b) and implemented in MATLAB using the resistances in Table 7.19 for all faces and the boundary pressures in Table 7.18. Three separate conditions are used for all the momentum balances (Hagen-Poiseuille), the flow balances, and boundary respectively.

Table 7.19: Resistance vector used in flow computations for the CoW (mmHg min/mL)

Arc	Alpha	Arc	Alpha	Arc	Alpha
1	0.3	7	0.09	13	0.09
2	0.09	8	0.3	14	0.06
3	0.3	9	0.12	15	0.3
4	0.12	10	0.3	16	0.3
5	0.3	11	0.06	17	0.09
6	0.3	12	0.3	18	0.09

The equations from Table 7.17 may be written in the general form as shown in equation (7.115), where the matrix C_1 comprises all the coefficients of the pressures (p), and the square matrix R has the resistance values for each branch.

$$C_1 p = R f \quad (7.115)$$

The matrix C_2 in Equation (7.116) describes the set of equations for the flow (f) balances at each internal node:

$$C_2 f = 0 \quad (7.116)$$

Lastly, in equation (7.117), the boundary conditions are set in a third matrix, C_3 , that is equivalent to the given inlet and outlet pressures (\bar{p}) from Table 7.18.

$$C_3 p = \bar{p} \quad (7.117)$$

All the knowns may then be concatenated into a single large matrix, A , all the unknown variables are organized into an x vector and the product of these is set equal to the known b vector as shown in Equation (7.118).

$$\underbrace{\begin{bmatrix} C_1 & -R \\ 0 & C_2 \\ C_3 & 0 \end{bmatrix}}_A \underbrace{\begin{bmatrix} p \\ f \end{bmatrix}}_x = \underbrace{\begin{bmatrix} 0 \\ 0 \\ \bar{p} \end{bmatrix}}_b \quad (7.118)$$

The residuals were evaluated through insertion. Manual spot checks were performed on several nodes and faces as an additional validation.

Table 7.20: Results of solving linear flow equations for simplified Circle of Willis model

Arc	Flow (mL/min)	Residual Flow	Node	Pressure (mmHg)	Residual Pressure
1	97.905	3.93E-15	1	100.000	-7.11E-15
2	77.479	5.37E-15	2	70.628	0.00E+00
3	45.518	4.06E-15	3	63.655	-8.88E-16
4	31.962	6.96E-15	4	50.000	8.88E-16
5	32.733	-3.47E-16	5	59.820	0.00E+00
6	96.312	-3.52E-16	6	50.000	3.55E-15
7	80.011	4.53E-15	7	100.000	-7.11E-15
8	46.351	-5.88E-15	8	71.106	0.00E+00
9	33.659	-2.98E-15	9	63.905	-3.55E-15
10	32.888	-3.45E-16	10	50.000	0
11	0.772	5.50E-17	11	59.866	-1.42E-14
12	91.370	1.01E-15	12	50.000	-1.42E-14
13	42.208	8.17E-15	13	100.000	-1.42E-14
14	49.162	9.62E-16	14	72.589	-1.42E-14
15	62.634	-7.83E-15	15	68.790	-1.42E-14
16	65.464	-3.54E-15	16	69.639	-1.42E-14
17	20.426	1.22E-14	17	50.000	0.00E+00
18	16.302	-2.89E-15	18	50.000	-7.11E-15

Case study 3, Circle of Willis model with extreme resistances. An occlusion can be simulated by setting the resistance (α) to infinity in a segment ($\alpha_{11} = \infty$). This can be achieved practically by setting the alpha value to a very high number (10e15, for example). Likewise, in other instances, short segments can have very small resistances which cause numerical problems when solving the linear algebraic system. To test this phenomenon, the resistance, α_{11} , can be set to a value of 0. In this case study, three methods will be used to attempt evaluation of these conditions; (i) original (no modification to any vessel resistances), (ii) Model A (Equation. (7.62)-(7.63)) and (iii) Model

B (Equation (7.78)). The results of solving the simplified Circle of Willis network with an occlusion are listed in and Table 9.21.

Table 7.21: Simulation results for an occlusion using different problem formulations

Node	Pressure Original	Pressure Model A	Pressure Model B	Arc	Flow Original	Flow MODEL A	Flow MODEL B
P1	100	100	100	F1	97.91	97.99	97.99
P2	70.63	70.6	70.6	F2	77.48	77.74	77.74
P3	63.66	63.61	63.61	F3	45.52	45.35	45.35
P4	50	50	50	F4	31.96	32.39	32.39
P5	59.82	59.72	59.72	F5	32.73	32.39	32.39
P6	50	50	50	F6	96.31	96.22	96.22
P7	100	100	100	F7	80.01	79.75	79.75
P8	71.11	71.13	71.13	F8	46.35	46.52	46.52
P9	63.91	63.96	63.96	F9	-33.66	-33.23	-33.23
P10	50	50	50	F10	32.89	33.23	33.23
P11	59.87	59.97	59.97	F11	-0.77	0	0
P12	50	50	50	F12	91.37	91.37	91.37
P13	100	100	100	F13	42.21	42.35	42.35
P14	72.59	72.59	72.59	F14	49.16	49.02	49.02
P15	68.79	68.78	68.78	F15	62.63	62.6	62.6
P16	69.64	69.65	69.65	F16	65.46	65.5	65.5
P17	50	50	50	F17	20.43	20.25	20.25
P18	50	50	50	F18	16.3	16.48	16.48

Table 7.22: Simulation results for a very short vessel (resistance = 0) using different problem formulations

Node	Pressure Original	Pressure Fully Coupled	Pressure Simplified	Arc	Flow Original	Flow Fully Coupled	Flow Simplified
P1	97.91	100	NaN		F1	97.91	97.88
P2	77.48	70.63	NaN		F2	77.48	77.42
P3	45.52	63.67	NaN		F3	45.52	45.56
P4	31.96	50	NaN		F4	31.96	31.86
P5	32.73	59.84	NaN		F5	32.73	32.81
P6	96.31	50	NaN		F6	96.31	96.33
P7	80.01	100	NaN		F7	80.01	80.07
P8	46.35	71.1	NaN		F8	46.35	46.31
P9	-33.66	63.89	NaN		F9	-33.66	-33.76
P10	32.89	50	NaN		F10	32.89	32.81
P11	-0.77	59.84	NaN		F11	-0.77	-0.95
P12	91.37	50	NaN		F12	91.37	91.37
P13	42.21	100	NaN		F13	42.21	42.18
P14	49.16	72.59	NaN		F14	49.16	49.19
P15	62.63	68.79	NaN		F15	62.63	62.64
P16	65.46	69.64	NaN		F16	65.46	65.46
P17	20.43	50	NaN		F17	20.43	20.47
P18	16.3	50	NaN		F18	16.3	16.26

The problem formulation that solves for pressures alone is not capable of evaluating an arc with 0 resistance thus the fully coupled problem must be used for this scenario. Both approaches to solving the system work with a very large resistance vessel.

7.12 Appendix L: Alternative models of computing hematocrit and biphasic viscosity

Blood flow in the body is governed primarily by the pressure difference between the blood leaving the heart (arterial blood pressure) and the pressure of blood returning to the heart (venous blood pressure). The blood is, however, not just a homogenous fluid but rather a mixture of blood plasma (bulk fluid) and red blood cells (RBCs, the particulate). This suspension gives rise to non-linear behavior in viscosity, leading to an uneven distribution of blood flow and pressure throughout a microcirculatory network. Moreover, the distribution of blood cells in a blood vessel network is uneven as well, with nonlinear relationships between vessel diameter ratios at bifurcations and the RBC distribution. This report investigates many mathematical models of the viscosity dependence on RBC volume fraction (hematocrit) and the models for determining the RBC split at a vascular junction (plasma skimming).

The findings of this report indicate the Pries In-Vitro model for viscosity and the KPSM model for plasma skimming are the most stable and should be used for solving large microvascular network simulations.

7.12.1 Viscosity:

Fahraeus-Lindqvist Effect. In the microcirculation, diameters $< 300 \mu\text{m}$, the biphasic suspension of blood does not simply adhere to a simple parabolic shape that would be observed in laminar fluid flow, but instead a significant decrease in the apparent viscosity (effective viscosity) caused by the margination of a plasma layer near the vessel wall which lubricates the RBC-filled core. This layer develops as the RBCs migrate to the faster-flowing center of the vessel. As a vessel diameter decreases and the RBCs continue to aggregate in the smaller, central cross sectional area,

the speed of each RBC must increase to obey mass conservation leading to a shift in the velocity profile of the biphasic fluid from a flat “plug flow” to a parabola.

A vessel junction is the meeting between two or more vessels. In the event a vessel continues to only one daughter branch, it is known as a continuation (part of the same vessel). A junction with two daughter vessels is known as a bifurcation, a trifurcation for three daughter branches, or a multifurcation for more than three daughters. For simplicity, this report will mostly discuss the case of a bifurcation unless otherwise noted. When blood flow in small vessels approaches a bifurcation, the flow of the RBCs is naturally biased towards the branch with the higher flow due to convection. This bias is *enhanced*, however, by an effect known as “plasma skimming” where the RBCs also favor a daughter with a larger cross sectional area.

Cell-free marginal layer overview. While the viscosity of the whole blood in large vessels (diameter > 300 μm) is generally defined as an isotropic constant with a value of $\sim 3.0\text{cP}$ [193,194], at high shear rates biphasic fluid has been empirically observed to be less viscous (less friction). Specifically, the viscosity is a function of the particulate density as diameter decreases in the range of 7-300 μm . This observation is known as the *Fahreus-Lindqvist effect* and is generally attributed to the axial accumulation of the RBCs with decreasing diameters, which leads to the formation of a cell free layer (CFL) that can lubricate the movement of the red blood cell core and reduce the effective viscosity. Additionally, the *Fahreus effect* (Figure 7.71) enhances this effect, describing the reduced hematocrit concentrations in smaller vessels.

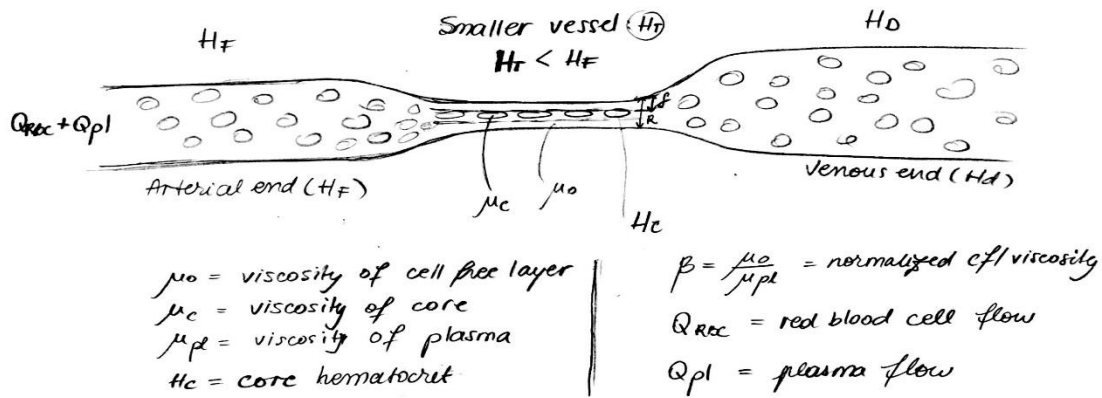


Figure 7.71: Fahreus and Fahreus-Lindqvist effect illustration.

The tube hematocrit (H_T), defined across the smaller vessel is smaller than the feed (H_F), or discharge (H_d) due to the axial accumulation of the red blood cells (less red blood cells can fit in the reduced cross sectional area – Fahreus effect); This agglomeration of RBCs in the center of the is then referred to as the core hematocrit (H_c) and the cross section outside the concentrated middle is named the cell free layer (CFL). This separation of the two phases (core and annulus) leads to the distinction between the core viscosity (μ_c) and the cell free layer viscosity (μ_o —usually approximated as plasma viscosity). The lower viscosity at the edge of the wall provides a lubrication effect for the central RBCs, reducing the apparent viscosity of the biphasic fluid (Fahreus-Lindqvist effect).

Previous work has empirically derived a value for blood viscosity as a function of diameter and RBC density. The parameter frequently used to represent the RBC density is the volume fraction of RBCs to vessel volume, termed *hematocrit* (Hct), as calculated in Equation (7.119). At large diameters, the viscosity tends to increase (more friction) as a function of higher hematocrit levels. A comparison of viscosity to diameter and hematocrit is given below. As observed in Figure 7.72, Fahraeus, Gaetgens and Bayliss [195] were able to measure the flow and pressure drop using an *in vitro* experiment to describe a diameter- and hematocrit- dependent apparent viscosity.

$$Hct = \frac{V_{RBC}}{V_{RBC} + V_{pl}} = \frac{V_{RBC}}{V_{tot}} \quad (7.119)$$

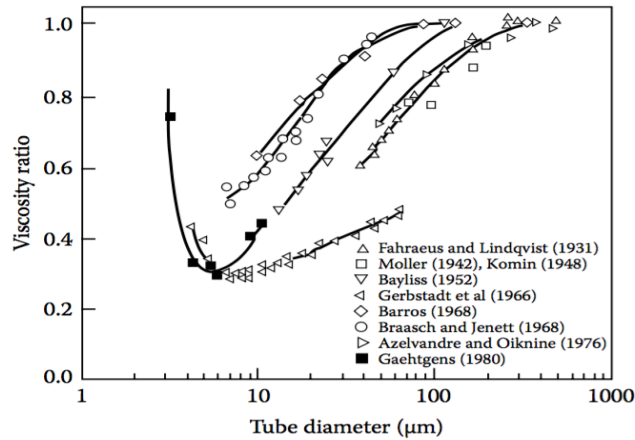


Figure 7.72: Experimental data for changes in apparent viscosity with diameter. The diameter (x-axis) is a logarithmic scale. This summary of phenomenological data [195] describes the Fahraeus –Lindqvist effect, which predicts a lower viscosity at smaller diameters due to the annular RBC free layer that lubricates the concentrated core.

One reason a decrease in diameter leads to a decrease in viscosity in the range of 10-100 μm is due to the agglomeration of RBCs along the longitudinal axis of the blood vessel, which creates a cell free layer (CFL) containing only plasma in small vessels. The existence of a CFL lowers the effective diameter, and thus the volume, available for RBCs to travel. In this range, a *tube hematocrit* can be calculated that expresses the RBC volume fraction of the core volume (as opposed to total vessel volume). This value is always higher than discharge hematocrit (the total vessel hematocrit; RBC volume that leaves the tube over a given time frame), or feed hematocrit (RBC volume that enters the vessel over a given time frame). When the cross sectional area occupied by the RBCs is diminished, mass conservation causes the core phase (plasma and RBCs that are in the center of the vessel) to speed up beyond the plasma in the CFL. Several models predict this change in the velocity profile of the two phases as described in a later section of this report. A summary of these models is offered in Table 7.23.

Table 7.23: Overview Viscosity models

Viscosity Models	Year	Abbreviation	CFL dependent	Highest Power coefficients
<i>Pries – in vitro</i> [160]	1989	Pvt	No	D^{12}
<i>Pries – in vitro modified</i> [196,197]	2013	Pvm	No	D^{12}
<i>Pries – in vivo</i> [198,199]	2005	Pvv	no	D^{12}
<i>Kiani and Hudetz</i> [200]	1991	KH	Yes	D^4
<i>Charm and Kurland</i> [201]	1974	CK	Yes	$e^{(-0.35D)}$

7.12.1.1 Pries In Vitro and In Vitro Modified models

The first viscosity model examined is termed the Pries In-Vitro model. This model was suggested by Pries in 1989 [202] derived from experiments in glass tube capillary surrogates. It computes a *relative viscosity* μ_{rel} (apparent viscosity/suspension viscosity) as a function of vessel diameter and hematocrit. This model predicts viscosity for diameters <1mm.

In a later experiment, Lipowsky et al. [203] measured a larger apparent viscosity than predicted by Pries. To compensate for this, Pries altered his 1989 model and offered an alternative parametric description of blood viscosity termed the “in-Vitro Modified” formula [204,205]. This model predicts the diameter decrease results in a decrease of apparent viscosity until 40 μ m, below which diameter the trend reverses (lower diameter increases viscosity). The discrepancy between Pries’s first and second model has been attributed to the presence of a 1 μ m layer of macromolecules (glycocalyx or endothelial surface layer), which was claimed absent in the *in-vitro* experiments.

Equations (7.120)-(7.125) describe the apparent viscosity in the Pries In Vitro model (Equations (7.120)-(7.122)) and Modified In-Vitro model (Equations (7.120)-(7.125)). The C variable describes the dependence of viscosity on hematocrit. Note, C remains linear as diameter increases from 0 to 8 μ , beyond which it becomes highly nonlinear. The units and description of relevant parameters are listed in Table 7.24. General trends are investigated in Figure 7.73.

$$\mu_{vitro} = \left[1 + (\mu_{45} - 1) \frac{(1 - H_d)^c - 1}{(1 - 0.45)^c - 1} \right] \mu_{plasma} \quad (7.120)$$

$$\mu_{0.45} = 220 e^{-1.3 D} + 3.2 - 2.44e^{-0.06D^{0.645}} \quad (7.121)$$

$$C = (0.8 + e^{-0.075 D}) \left(-1 + \frac{1}{1 + 10^{-11} D^{12}} \right) + \frac{1}{1 + 10^{-11} D^{12}} \quad (7.122)$$

$$\mu_{vitroModified} = \left[1 + (\mu_{0.45}^* - 1) \frac{(1 - H_d)^c - 1}{(1 - 0.45)^c - 1} \left(\frac{D}{D - 1.1} \right)^2 \right] \left(\frac{D}{D - 1.1} \right)^2 \mu_{plasma} \quad (7.123)$$

$$\mu_{0.45}^* = 6 e^{-0.085 D} + 3.2 - 2.44e^{-0.06D^{0.645}} \quad (7.124)$$

$$C = (0.8 + e^{-0.075 D}) \left(-1 + \frac{1}{1 + 10^{-11} D^{12}} \right) + \frac{1}{1 + 10^{-11} D^{12}} \quad (7.125)$$

Table 7.24: Table of variables for Pries's in vitro viscosity models

Symbol	Definition	Units	Value
$\mu_{vitroModified}$	Apparent viscosity Modified	cP	F(dia,Hd)
μ_{vitro}	Apparent viscosity Original	cP	F(dia,Hd)
D	Vessel diameter	μm	Known scalar
H_d	Discharge hematocrit	Vol%	Unknown scalar
μ_{45}	Viscosity at Hd =0.45	cP	F(dia,Hd)
μ_{plasma}	Plasma viscosity	cP	1.0

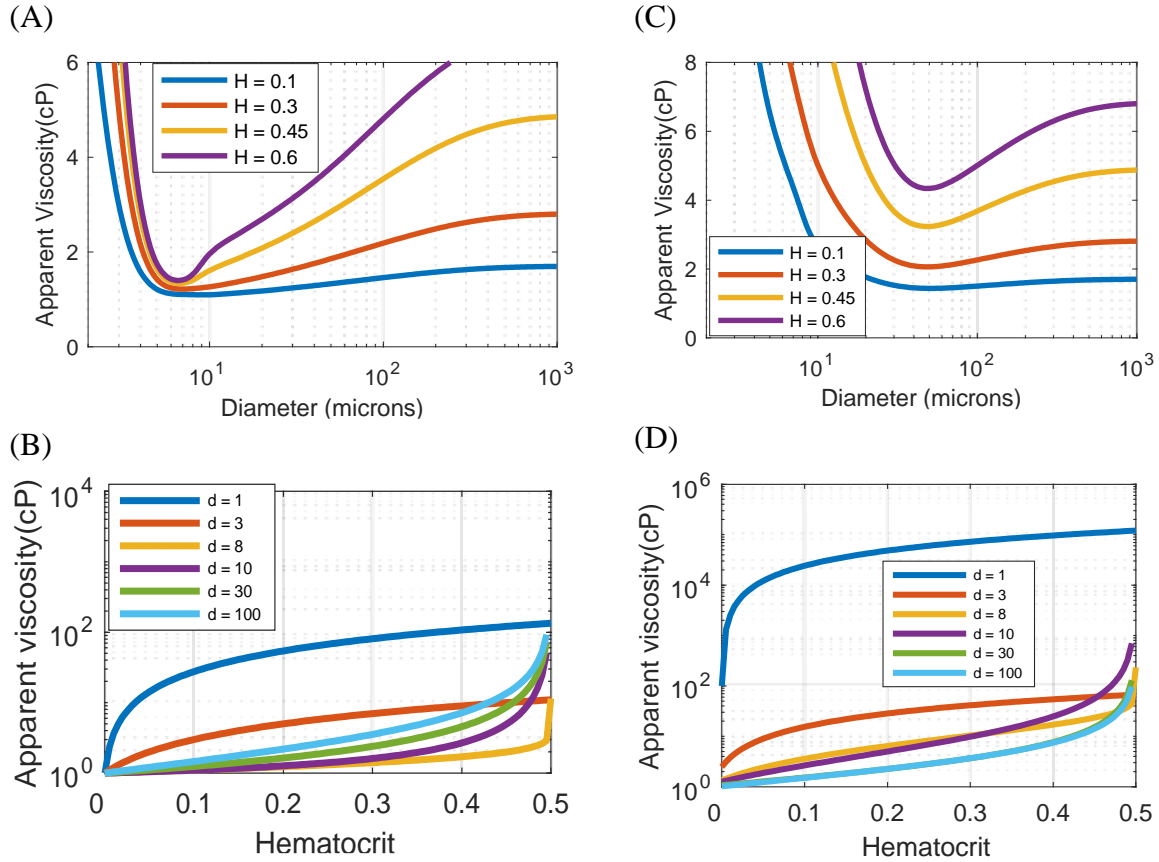


Figure 7.73: Apparent viscosity of for the (A,B) Pries In-Vitro model and (C,D) Pries In-Vitro Modified model.

(A, B) Apparent viscosity trends as vessel diameter varies. (C, D) Apparent viscosity as a function of varying Hematocrit. Both models exhibit a decrease in viscosity as the diameter decrease below 300 μm . The In-Vitro Modified model gives a higher viscosity at low diameters.

7.12.1.2 Pries In Vivo model

A third model by Pries and Secomb [206,207], that was empirically derived using optical microscopy of capillaries in the rat mesentery was used to advance the previous models by accounting for the endothelial surface layer (ESL). An ESL thickness of 0.8-1 μm was reported for diameters of 10-40 μm and declines considerably for diameters below 10 μm . Note, there was no experimental validation of this ESL layer effect. This new model was termed the “In-Vivo” viscosity model.

The viscosity formula expressed in (7.126)-(7.132) reflects the original nomenclature and no simplification. W_{as} is a monotonically increasing function that contains a vertical asymptote at the maximum thickness of W_{max} . W_{peak} grows linearly to a peak at D_{crit} (critical diameter). D_{crit} decays exponentially to 0 as the diameter of a vessel increases. D_{eff} is the effective diameter without the ESL and W_{ph} is the assumed thickness of the layer.

Table 7.25: summarizes the variables from (7.126)-(7.132). Figure 7.74 expresses the viscosity over a range of hematocrit and diameter.

$$W_{as} = \begin{cases} \frac{D - D_{off}}{D + D_{50} - 2D_{off}} W_{max} & D > D_{off} \\ 0 & D \leq D_{off} \end{cases} \quad (7.126)$$

$$W_{peak} = \begin{cases} 0 & D \leq D_{off} \\ E_{amp} \frac{D - D_{off}}{D_{crit} - D_{off}} & D_{crit} > D \geq D_{off} \\ E_{amp} e^{-E_{width}(D - D_{crit})} & D \geq D_{crit} \end{cases} \quad (7.127)$$

$$W_{ph} = W_{as} + W_{peak} E_{peak} \quad (7.128)$$

$$D_{ph} = D - 2W_{ph} \quad (7.129)$$

$$D_{eff} = D - 2W_{eff} \quad (7.130)$$

$$W_{eff} = W_{as} + W_{peak} [1 + H_D E_{HD}] \quad (7.131)$$

$$\mu_{vivo} = \mu_{vitro} \left(\frac{D}{D_{eff}} \right)^4 \quad (7.132)$$

Table 7.25: Variables for Pries's in vivo viscosity model

Symbol	Definition	Value	Units
D_{off}	Threshold diameter below which ESL is 0	2.4	μm
D_{crit}	Critical diameter	10.5	μm
D_{50}	Estimated free parameter	100	μm
W_{max}	Maximal thickness ESL	2.6	μm
E_{amp}	Estimated free parameter	1.1	--
E_{width}	Estimated free parameter	0.03	--
E_{peak}	Estimated free parameter	0.6	--
E_{HD}	Estimated free parameter	1.18	--
D	Vessel diameter	Known Scalar	μm
Hd	Discharge hematocrit	Known Scalar	--
μ_{vitro}	In Vitro Viscosity	Computed Scalar (from Equation (7.120) - (7.125))	--
W_{eff}	Definition	--	--
W_{as}	Definition	--	--
W_{peak}	Definition	--	--
D_{ph}	Definition	--	--

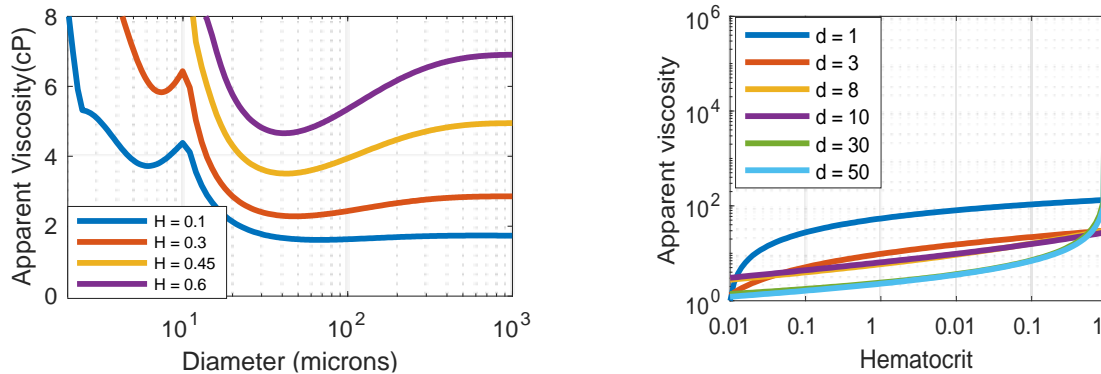


Figure 7.74: Apparent viscosity for Pries's in vivo model.

Here, we see apparent viscosity as a function of hematocrit (left) and of diameter (right): The apparent viscosity exhibits a local maxima (peak) around 10 microns, a global maxima at lower diameters and global minima at larger diameters.

7.12.1.3 Charm and Kurland, marginal zone layer

Charm and Kurland presented an empirically derived model relating the apparent viscosity to the cell free layer thickness (δ), the core hematocrit (H_c) and the core viscosity (μ_c) of the blood [201]. Charm and Kurland's viscosity model was proposed as a method for fitting

experimental data with a model. To fit the model, the cell free layer thickness and the core/tube hematocrit are required to compute the apparent viscosity of the blood. This limits the comparison with other viscosity models, as experimental data is not always available for all of the necessary variables.

One way to solve this dilemma and calculate viscosity using the Charm and Kurland method is to employ previously reported models for tube hematocrit and CFL thickness, Equation (7.133), to fill in the missing experimental data [201]. Equation (7.134) describes the relationship between the cell free layer thickness (λ), the core hematocrit (H_c) and tube hematocrit (H_t) as derived from the marginal zone layer theory [208,209]. Here, α_c is an experimentally derived as a function of the H_c . Using Equation (7.133) - (7.136), the core hematocrit can be calculated through a system of nonlinear equations and the apparent viscosity can be determined through Equation (7.137). Figure 7.75 reflects the trends of this model under different hematocrit values.

$$\frac{H_t}{H_d} = H_d + (1 - H_d)(1 + 1.7 e^{-0.35D} - 0.6e^{-0.01D}) \quad (7.133)$$

$$\lambda^2 = H_t/H_c \quad (7.134)$$

$$\frac{H_c}{H_d} = 1 + \frac{(1 - \lambda^2)^2}{\lambda^2[2(1 - \lambda^2) + \lambda^2(1 - \alpha_c H_c)]} \quad (7.135)$$

$$\alpha_c = 0.07 e^{2.49 H_c + (1107/T)e^{-1.69H_c}} \quad (7.136)$$

$$\mu_{app} = \frac{\mu_p}{1 - \lambda^4(\alpha_c H_c)} \quad (7.137)$$

Table 7.26: Variables for Charm and Kurland's viscosity model

Symbol	Definition	Units	Value
μ_p	Plasma viscosity	cP	1.0-1.3
H_t	Tube Hematocrit	Vol %	Computed Scalar (from Equation (7.133))
H_c	Core hematocrit	Vol %	Computed Scalar (from Equation (7.134))
T	Temperature	K	298 - 310
H_d	Discharge Hematocrit	Vol %	Known Scalar
D	Vessel diameter	μm	Known Scalar
λ	Variable accounting for cell free layer	--	Computed Scalar (from Equation (7.135))
α_c	Experimentally derived parameter	--	Computed Scalar (from Equation (7.136))
μ_p	Apparent Viscosity	cP	Computed Scalar (from Equation (7.137))

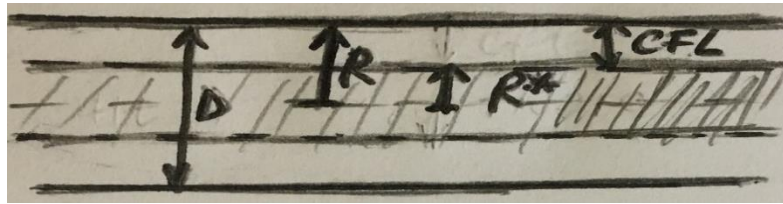


Figure 7.75: Relevant dimensions in a single tube.

Across a single tube, D is the diameter and R is the radius of the vessel. The cell free layer (CFL) occupies the outer cross section of the capillary and the effective radius (R^*) is the radius without the thickness of the CFL. The tube hematocrit may then be defined across the effective radius as the core hematocrit (H_c) and is approximated as 0 for the section between R^* and R . In Equation (7.134)-(7.135), λ is the fraction between R^*/R .

7.12.1.4 Kiani and Hudetz model

Kiani and Hudetz developed a viscosity model by modifying and combining the marginal zone theory of Haynes [201] for large vessels with the axial-train model of Whitmore [210] for smaller vessels [211]. Starting from the Navier-Stokes equation, Haynes derived a model for vessels larger than $50\mu m$, which accounts for flow both in the core and the cell free layer. In this axial-train model, RBCs move in single file at diameters approaching the radius of an RBC.

Hudetz empirically derived a hematocrit dependence for the cell free layer thickness (δ) and μ_c (data from Reinke et al [212]). He reports a constant plasma viscosity of 1.7cP at room temperature.

$$\mu_{app} = \left[\frac{1}{1 - \left(1 - \frac{\mu_p}{\mu_c}\right) \left(1 - \frac{2\delta}{d}\right)^4} \cdot \frac{1}{1 - \left(\frac{D_p}{d}\right)^4} \right] \mu_{plasma} \quad (7.138)$$

$$\mu_c = e^{0.48 + 2.35 H_d} \quad (7.139)$$

$$\delta = 2.03 - 2 H_d \quad (7.140)$$

$$D_m = 2.7 \quad (7.141)$$

Table 7.27: Variables for Kiani and Hudetz's viscosity model

Symbol	Definition	Units	Value
μ_p	Plasma viscosity	cP	1.7
μ_c	Core apparent viscosity for large tubes ($d > 300 \mu m$)	cP	Equation (7.139)
δ	Cell free layer thickness	μm	Equation (7.140)
d	Vessel diameter	μm	Known Scalar
D_m	Maximal deformation of an RBC	μm	2.7
D_p	Parent diameter	μm	Known Scalar

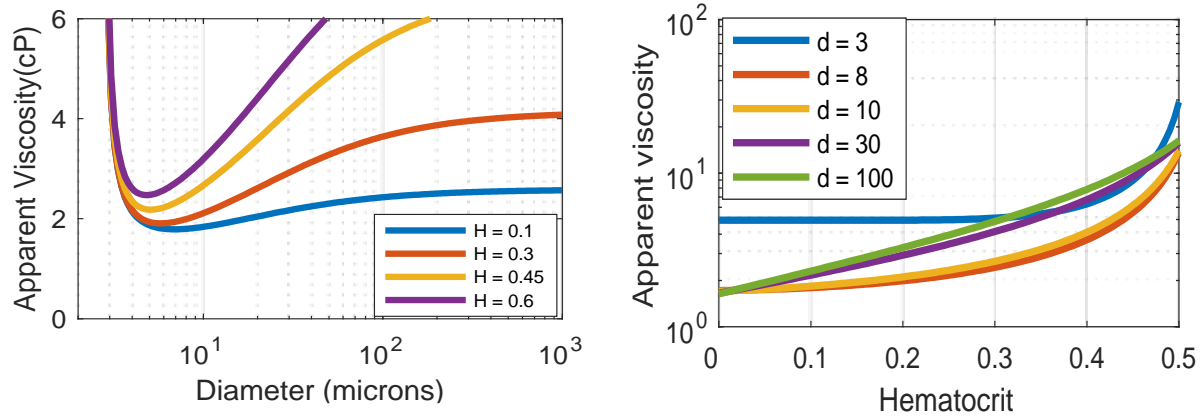


Figure 7.76: Apparent viscosity as predicted by Kiani and Hudetz viscosity model. Left) Diameter-dependence of viscosity at different levels of discharge hematocrit. Right) Hematocrit-dependence of viscosity at varying diameters. The model is less sensitive to hematocrit, but it does not resolve any diameters below $2.7 \mu m$.

7.12.1.5 Viscosity trends

A sparse representation of the diameter spectra at chosen hematocrit levels is presented in Figure 7.77. As hematocrit approaches 1, the models become more unstable and creates a field that is difficult to converge in a fixed-point iterative solver. The Pries In-Vitro viscosity model is the most stable of all Pries models. The Charm and Kurland model is not recommended due to its restrictive behavior (it was only derived for <0.6 hematocrit and diameters greater than $25 \mu m$).

The Pries In-Vivo and In-Vitro Modified models account for the existence of a glycocalyx, hence their predictions of viscosity are much higher at smaller diameters, whereas Kiani and Hudetz and the In-Vitro models give similar results.

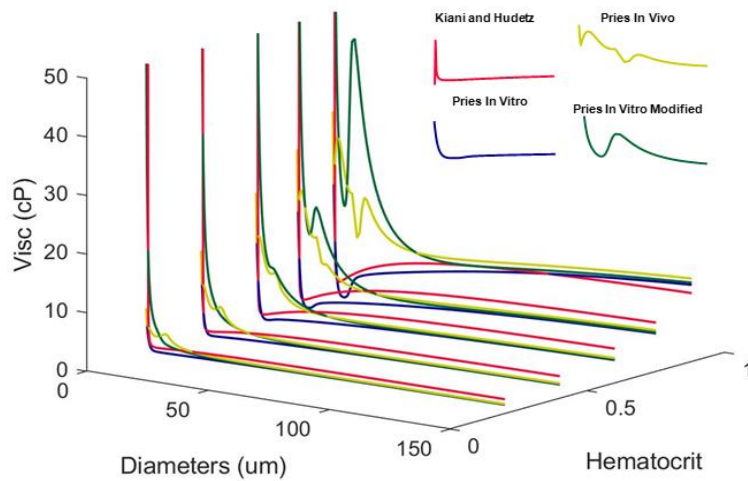


Figure 7.77. Parametric investigation of viscosity over a range of diameter and hematocrit. As the hematocrit level approaches 1, the models all begin to show discontinuities. Pries in Vitro is the most stable of all Pries models.

7.12.2 Plasma skimming

This section describes the non-ideal splitting of RBCs at a bifurcation into daughter branches known as *plasma skimming*. An overview of available models for plasma skimming (known as hematocrit split models) is also provided. While all models are described with individual sets of parameters, all models will be re-derived to yield the red blood cell flux ratio (λ) as a function of the flow split (γ) for direct comparison and straightforward implementation.

As observed by Bayliss [213] RBCs tends to aggregate to the main axis of a vessel, leaving a plasma-rich peripheral section (known as the Cell Free Layer or CFL). Krogh [214] and later Fourman and Mofatt [215] explained that branching of a tube leads to one daughter branch preferentially siphoning from the CFL rather than the RBC-rich vessel center. This leads to the smaller branch receiving a lower density of RBCs compared to the parent vessel (Figure 7.78).

The models below attempt to predict the percent of total RBC flux ($\lambda = f_{total}H_d$) that each daughter branch will receive.

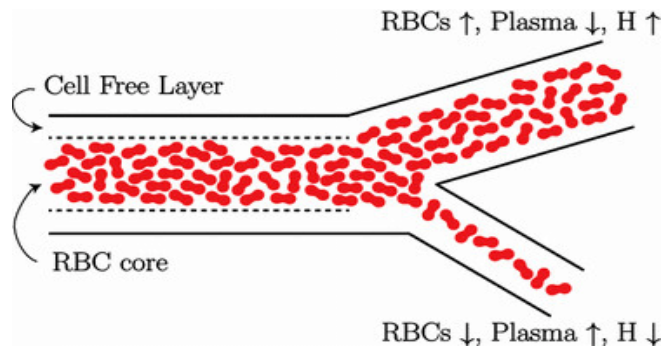


Figure 7.78. Plasma Skimming in small vessels. In vessels with small diameter, the red blood cells aggregate at the center of the capillary, creating a cell free layer. At high shear rates (high flow rate), the fraction of the red blood cells that one branch will receive will be affected by the diameter of the branch and by its flow rate.

Available hematocrit split models. In order to establish a common system of reference in spite of the numerous variables in each split rule, Figure 7.79 summarizes the main structures found in a given bifurcation. F_1 refers to the parent branch face, while F_2 and F_3 are the daughter faces.

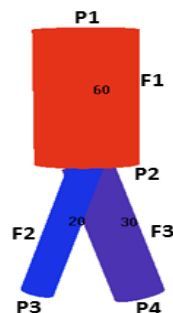


Figure 7.79: Graphic representation of a basic bifurcation.

All the points (P) and faces (F) of the structure are annotated. The black labels on each face represent the diameters in microns. In this case the first face is the parent branch, and face 2 and 3 are the daughters.

The proper identification of the parent and daughter branches is necessary for implementation of a model on a discretized network because all split rules compute the RBC fraction (λ) as a function of inlet and outlet parameters (flow, diameter, etc). For implementation of all models on discretized 1-dimensional networks, fixed point iteration was used to compute the vessel hematocrit as seen in Section 4.6.

7.12.2.1 Pries and Secomb model

The most widely accepted model for plasma skimming on a single bifurcation was empirically derived by Pries and Secomb in 2005 [207] to describe blood flow measured from rat mesentery. This model is not amenable to networks containing multifurcations (a split from a parent branch to 3 or more daughter branches) due to the variable definition within the formula that is defined by the diameters of only 2 daughter vessels (A in Equation (7.142) - (7.143) and in Table 9.28).

The models developed by Pries in his 1989 publication [202] are derived from 2 distinct bifurcations. In one, the parent diameter was 20 μm , the two daughter branches were 16.5 and 17.5 μm , and the inlet hematocrit was 0.49. The coefficients of the model were tuned to this case study. The second bifurcation had a parent vessel of 7.5 μm , outlet branches measuring 6 and 8 μm and an inlet hematocrit of 0.43 and the second set of coefficients were tuned to this second case study.

The Pries model has been described by base equations (Equations (7.142) - (7.143)) with variable parameters (Table 9.28). x_0 is a threshold value (bounds the γ between $[x_0]$ and $[1 - x_0]$), i is the index of the daughter branch in question and j is the index of the other daughter branch in

a given bifurcation. γ is the flow split of a given daughter branch and d_i and d_j correspond to diameters of face i and face j respectively.

$$\text{logit}(\lambda_i) = A + B \text{logit}\left(\frac{\gamma_i - x_0}{1 - 2x_0}\right) \quad (7.142)$$

$$\lambda_i = \begin{cases} \frac{1}{1 + e^{A-B \ln(G_\alpha)}}, & x_0 < \gamma_i < 1 - x_0 \\ 0 & , \quad \gamma_i \leq x_0 \\ 1 & , \quad \gamma_i \geq 1 - x_0 \end{cases} \quad (7.143)$$

Table 7.28: Coefficients used for Pries's plasma skimming model
Coefficient Symbol

Method	A	B	X_0	G
Pries (1989): $d_i \cong d_j$	0.069	1.13	0.032	$\frac{\gamma_i - x_0}{1 - \gamma_i - x_0}$
Pries (1989): $d_i < d_j$	-0.35	1.29	0.052	$\frac{\gamma_i - x_0}{1 - \gamma_i - x_0}$
Pries (2005)	$-13.29 \left[\frac{(D_i^2/D_j^2 - 1)}{(D_i^2/D_j^2 + 1)} \right] \frac{(1 - H_p)}{D_p}$	$1 + 6.98 \frac{(1 - H_p)}{D_p}$	$0.964 \frac{(1 - H_p)}{D_p}$	$\frac{\gamma_i - x_0}{1 - \gamma_i - x_0}$
Lorthois (2010)	$-15.47 \left[\frac{(D_i^2 - D_j^2)}{(D_i^2 + D_j^2)} \right] \frac{(1 - H_p)}{D_p}$	$1 + 8.13 \frac{(1 - H_p)}{D_p}$	$1.12 \frac{(1 - H_p)}{D_p}$	$\frac{\gamma_i - x_0}{1 - \gamma_i}$
Chebbi (2015)	$-\frac{6.96}{D_p} \ln\left(\frac{D_i}{D_\beta}\right)$	$1 + 6.96 \frac{(1 - H_p)}{D_p}$	$\frac{0.4}{D_p}$	$\frac{\gamma_i - x_0}{1 - \gamma_i - x_0}$

7.12.2.2 Linninger – KPSM (2015) and Beta Model (2017):

Linninger and Gould derived a model termed the Kinematic Plasma Skimming Model (KPSM) from volatile biphasic gas mixing theory. This model, Equation (7.144)-(7.145), predicts a hematocrit distribution using an intermediate nodal hematocrit as in Equation (7.144) from flow ratio (Equation (7.144)) and area ratio (Equation (7.146)). Here, A is the area of the branch, m is a volatility parameter, and i denotes the branch in question, j denotes the inlet node of the branch, and p denotes the parent branch.

$$h^* = \frac{q_p h}{\sum q_i \theta_i} \quad (7.144)$$

$$h_i = h_p - \Delta h = \theta_i h_j^* \quad (7.145)$$

$$\theta_i = \left(\frac{a_i}{a_p} \right)^{\frac{1}{m}} \quad (7.146)$$

Due to the nonlinearity of the set of equations in Equation (7.142)-(7.143) and Equation (7.144)-(7.146), a fixed-point iteration method is employed for solving 2 systems of linear equations. This process is expressed in more detail in previous work [27,115]. Here, \bar{R} denotes the resistance calculated from previous iteration and k denotes the terminal node of a i^{th} face.

$$\nabla q(p) = C_2 \cdot R^{-1} \cdot C_1 p = 0; \text{ where } H \text{ and } d \text{ are known} \quad (7.147)$$

$$q \cdot \nabla h_j^* = 0 \quad (7.148)$$

$$\max \left(\frac{1}{\bar{R}} C_1 \cdot p, 0 \right) \cdot h_j^* + \max \left(\frac{-1}{\bar{R}} C_1 \cdot p, 0 \right) \cdot h_k^* = 0$$

where p is known

Linninger et al. published their kinematic plasma skimming model in 2015, which computes the adjusted discharge hematocrit (H^*); the nodal concentration of hematocrit that becomes the source of the convective flux across the daughter face (7.148). This model has an adjustable parameter (m) that increases or decreases the severity of plasma skimming.

This same model can be reduced in size by removing the nodal parameter value and performing mass balance of RBCs on the faces directly. Here λ represents the fractional RBC flux for the bifurcation, and it is calculated for each face of a network (as opposed to H^* in the 2015 method). The area fraction (θ_i) is defined by (7.146) and the fractional red blood cell flux (λ_i) is now given by (7.149). The face hematocrit (H_i) is then given by (7.150).

$$\lambda_i = \frac{\theta_i Q_i}{\sum \theta_i Q_i} \quad (7.149)$$

$$H_i Q_i = \lambda_i (H_p Q_p) \quad (7.150)$$

7.12.2.3 Dellimore

Dellimore, Dunlop and Canham [216] modified Klitzman and Johnson's empirical fitting equation [217]. and determined parameter 'B' through a least-square linear regression fit to all 48 sets of experiments. The model combines the fractional hematocrit and fractional flow through a power rule that has no diameter dependence.

This method requires a nonlinear solver (due to hematocrit coupling with viscosity) which can be converged with a fixed-point iterative method as previously described in Equations (7.147)-(7.148).

Dellimore describes the red blood cell fraction of a daughter branch (λ_α) as the function of the flow split per daughter branch (γ_α) and the experimentally derived constant, B . The flow split is computed using only the fractional flow rate and the face hematocrit computation follows.

$$\lambda_{\alpha} = \frac{\gamma_{\alpha}^B}{\gamma_{\alpha}^B + (1 - \gamma_{\alpha})^B} \quad (7.151)$$

$$B = 1.2$$

$$H_{\alpha} = \frac{\lambda_{\alpha} H_p}{\gamma_{\alpha}} \quad (7.152)$$

7.12.2.4 Fenton

Fenton et al [218,219] proposed a model to calculate the fractional red cell flux in the daughter branch (λ_i) as a function of flow split (γ_i), and an experimentally-derived parameter a as in Equation (7.153)-(7.154). a is dependent on the parent diameter (D_m) and RBC diameter (d_c) which was not given, but was found to vary in literature between 5-8 μ m. Combining the experimental data from the 1985 papers with the in vitro measurements of Tvetenstrand [220] in a large scale model of a T bifurcation and the in vivo data from the rabbit ear from Schmid-Schonbein [221] gives an empirical equation relating fractional red blood cell flux with the fractional flow rate (Equation (7.153)). The relationship was used in a small computer network model of the hamster cremaster muscle [222] consisting of 76 vessel segments connected at 50 nodes [218].

This model can be solved using fixed-point iteration as described previously. Fenton et al. propose to calculate the fractional red cell flux in the daughter branch (λ_i) as a function of flow split (γ_i) and an experimentally-derived parameter a (7.153). a is dependent on the parent diameter (D_m) and RBC diameter (d_c) (7.154). The value of d_c seems to range between 5-8 μ m [223], and here $d_c = 5\mu$ m is chosen.

$$\lambda_i = \left(a + \frac{1-a}{2\gamma_i} \right) \gamma_i \quad (7.153)$$

$$a = \frac{1}{1.4 - \sqrt{d_c/D_m}}, \quad \text{where } a \geq 1 \quad (7.154)$$

7.12.2.5 Plouraboue

Plouraboue proposed a model that does not focus on phase separation effects as *local* mechanisms. Instead, a *global* method computes the distribution of hematocrit from a pre-determined flow field.

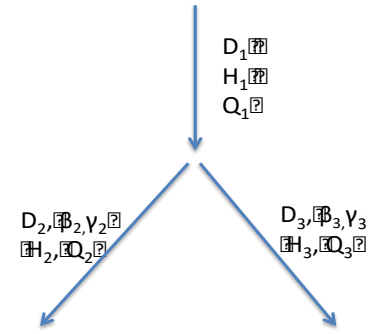
Mass conservation is applied at each node to determine a hematocrit potential (\hat{H}) using a Neumann inlet boundary condition. This ‘elliptical’ implementation constructs a diffusion-like equation for mass balance at each node but using an anisotropic diffusion coefficient across each arc. This anisotropic diffusion coefficient is directly proportional to the bulk flow across the arc by a factor of 1 (it is the same value).

RBC flux conservation is enforced through Equation (7.155) applied to each node in the network. The discharge hematocrit (h_d) for each vessel becomes the potential difference between nodes ($\Delta\hat{h}$).

$$\nabla q_{RBC} = \sum_{i=1}^n (\Delta\hat{h} q) = \sum_{i=1}^n \Delta\hat{h} \frac{\pi d^4}{128 \mu_i l} \Delta p = 0 \quad (7.155)$$

Table 7.29: Variables used in Plouraboué’s plasma skimming model

Symbol	Definition
\hat{H}	Hematocrit potential at face
μ_a	Apparent viscosity at face
L	Length of face
D	Diameter of face
Q_{RBC}	Fractional red blood cell flux of face
Q	Longitudinal flow rate of face
ΔP	Pressure drop at node



7.12.2.6 Balogh and Bagchi (3D immersive boundary method)

With the advancement of massive computing and big data, rigorous 3-dimensional simulations have become capable of simulating small microcirculatory networks and with many deformable spheres within. Balogh and Bagchi were able to perform this feat in 2017 where the model was validated against 3D deformations while sucking an RBC into a micropipette, 3D deformations when pushing an RBC through a small, rigid, square channel, cell separation and lab-on-chip microfluidic devices. The model was initially claimed to exhibit plasma skimming effects that were consistent with other studies, but in a later paper this was further investigated to find the trends only sometimes match those from empirical models [224]. The 3D simulations were extended later that year and the following year to investigate trends of plasma skimming [225,226].

The mathematics are described in more detail in Section 7.12.8 but in here brief, the method uses the immersed boundary method (described elsewhere in detail) for simulating RBCs, white blood cells (WBCs) and platelets. The RBCs are very elastic, WBCs are less elastic, platelets are inelastic, and vessel walls are non-deformable. One of the key advantages of the IBM is that it allows a simulation of fluid-structure interaction (FSI) without body-fitted meshing (which can be computationally prohibitive in a dynamic simulation). For clarity, the FSI is not between the blood

and the wall, as is common in 3D vascular fluidic simulations, because the wall is rigid (immobile and inelastic). The FSI in this model exists between the plasma and the RBC surface, where the RBC is allowed to deform under stress.

The mesh is defined by a dual-mesh technique, a Eulerian mesh (defined as the lattice grid written in Cartesian coordinates) used for the vasculature/plasma mesh and a Lagrangian mesh (the local mesh of each RBC surface) used for each RBC. The independent solving of these two meshes exploits the properties of each domain, significantly reducing computational effort and omitting the need for a body-fitted mesh in every time step. The bulk flow is calculated in the Eulerian grid (Cartesian coordinates) before the deformation and advection of the RBCs is computed on the Lagrangian grids.

The Lagrangian grid points (nodes on the RBC cell surface) are advected (convected) through the Eulerian grid and deformed. The RBC is coupled to the bulk fluid through a forcing function added to the Stokes equations of flow. This forcing function reduces the velocity by the integral of the force across the RBC surface. This means that the velocity will slow down linearly as it deforms the cell surface, which may correspond to a conservation of energy, but is not explicitly derived or discussed in the work. The sink term in the momentum balance uses a Dirac delta function to affect the plasma nodes up to a fixed distance away from the RBC.

The time integrator uses the Adams Bashford method for the conservation equation and a Crank-Nicholson scheme for the convection-diffusion equation. The solving step uses a modified form of the original equations:

$$\rho \frac{u_i^* - u_i^n}{\Delta t} = G_i + \frac{3}{2} \alpha_i^n - \frac{1}{2} \alpha_i^{n-1} + \frac{1}{2} (\beta_i^* + \beta_i^n) \quad (7.156)$$

$$\frac{d^2\phi}{dx_i dx_i} = \frac{\rho}{\Delta t} \frac{du_i^*}{dx_i} \quad (7.157)$$

$$\rho u_i^{n+1} = \rho u_i^* - \Delta t \frac{d\phi}{dx_i} \quad (7.158)$$

Where

$$\begin{aligned} \alpha_i &= F_i + \frac{d}{dx_i} \left(\mu \frac{du_j}{dx_i} \right) \\ \beta_i &= \frac{d}{dx_j} \left(\mu \frac{du_i}{dx_i} \right) \end{aligned} \quad (7.159)$$

This implementation seems to be similar to the SIMPLE algorithm where an intermediate (guess) velocity field (u_i^*) is solved for, then the continuity is solved for, then the velocity field is evaluated for correctness, all in each iteration which is cycled until convergence. The ϕ term in the equations is set to pressure, although the modelers claim this parameter can transposed into a different space using a preconditioner as discussed in Section 7.20.6.

The deformation of the RBC is calculated using conservation of momentum. Note, the Lagrangian and the Eulerian systems are related by a simple linear coordinate transformation (by an offset vector and rotational matrix), although the authors offered a more detailed explanation of how this cooperation works.

The validations are performed on analytic solutions such as a solid, immovable sphere with a flow field around it. Of the many validations, included are the comparisons to experimental deformability of a single RBC through a rigid channel, with a good fit to the data. Another included

showing cell-selective sorting from a microfluidic device. Lab on a chip trends were also given, but these were similar results to the cell separation microfluidic device.

In a later publication, the trends of plasma skimming are in question, as the model appears to give results conducive with the experimental data (Pries, Lorthois, etc.), however the group claims that their model only agrees with previously established models half of the time.

7.12.2.7 Plasma skimming implementation

Solving algorithm. The algorithm used to solve large (>100,000 unknowns), highly nonlinear sets of algebraic equations requires a robust solving algorithm. In other words, the algorithm cannot break in the event a network has a trifurcation, an arc is oriented backwards, or the network has a large branching order. This limitation may omit the use of some of the plasma skimming models reviewed in previous sections, but the solvability of a network with any plasma skimming model must not be limited by the nonlinear convergence technique. The algorithm for splitting and solving the nonlinear set of equations using the KPSM model is offered in previous work [27] and in Section 4.6.

7.12.2.8 Modified m-coefficient model

Across all the experimental observations, a clear range for hematocrit values has been established. The upper limit of measurements, which may not be the maximum physiological hematocrit but rather is a reasonable suggestion. This value of $h \leq 0.7$ [203,204,227,228]. Also importantly, no segments should be entirely void of RBCs ($h \geq 0$). Whereas the KPSM model has only a single adjustable parameter, m , and does not suffer from hematocrit-free vessels, it has the

ability to generate very high hematocrit values. One important assessment is that the splitting model should not act as aggressively (should split more evenly) when the vessel is highly packed with RBCs such that there is no cell-free layer for plasma skimming to act on. The chosen model to represent this hematocrit-dependent plasma skimming effect is given in Equation (7.160) and is visualized across the range of hematocrit in Figure 7.80.

$$h_{max} = \max(h_{daughters})$$

$$m = m + 100 \cdot 10^{20(h_{max}-0.9)} \quad (7.160)$$

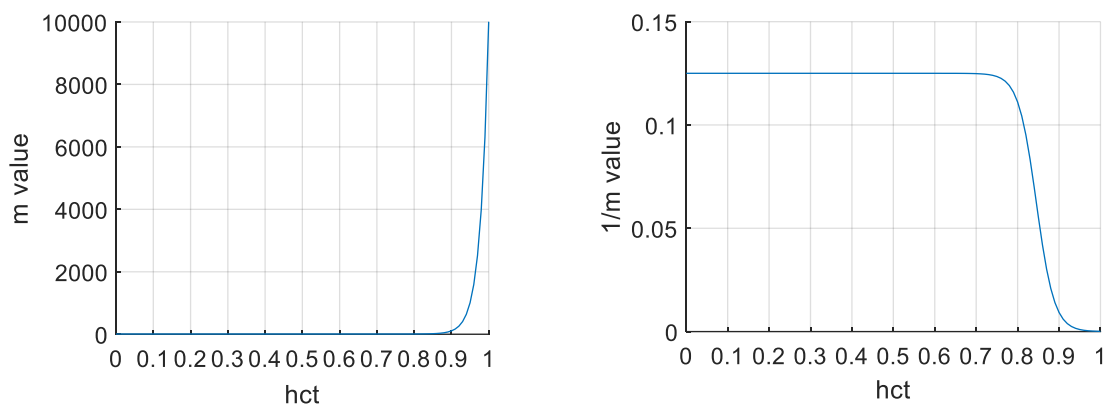


Figure 7.80: Visualization of the m value as a function of hematocrit.

Left) The value of m increases exponentially with the value of hematocrit, with no significant increase until a value of ~ 0.75 .

Code for generating Figure 7.80.

```
function testMValue
close all, h = 0:0.01:1; mBar = 8;
m = mBar + 100*10.^(20*(h-0.9)); figure, plot(h,m);
ylabel('m value'); xlabel('hct'); box off,
set(gca,'xTick',0:0.1:1), grid on

m = 1./m; figure, plot(h,m);
ylabel('1/m value'); xlabel('hct'); box off
set(gca,'xTick',0:0.1:1), grid on
end
```


7.12.3 Application to networks

The networks used for comparison of splitting rules are summarized in the table below. Starting from a limited number of branches (a bifurcation and an artery), plasma skimming can be observed. The structures then become increasingly complex with added bifurcations and eventually a venous closure but without multifurcations (to accommodate Pries's delicate equation). Unions are two or more inlet branches feeding into a point. Multifurcations are more than 2 branches leaving the parent branch. Note, most empirical formulae are based on single, or local bifurcations, so this is the most popular model for testing basic predictions of a model. Some simple statistical comparisons of the split models is offered in Figure 7.81. Network trends reveal hematocrit increasing as the network branches, where the RBCs are concentrated into the larger main vessel as the plasma is siphoned off to the smaller side branches (Figure 7.82). Some network statistics are offered in Table 7.30 where statistics affecting network solvability are labeled with green (low occurrence rate) and red (high occurrence rate).

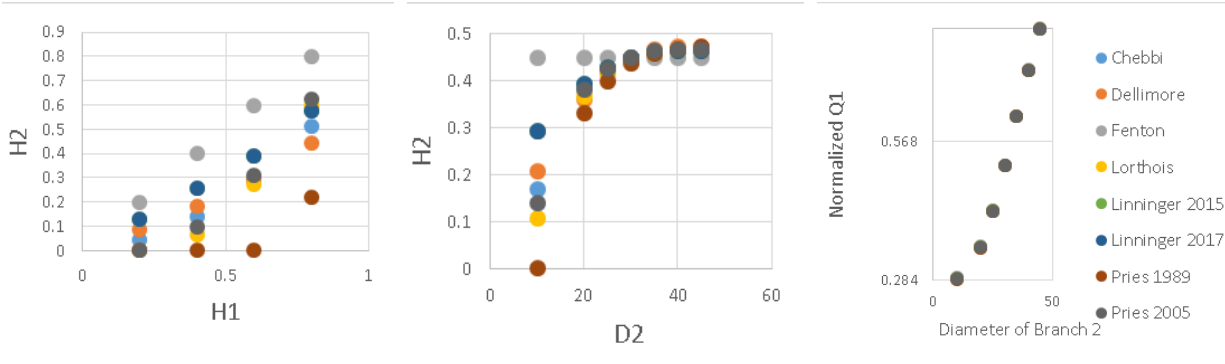


Figure 7.81. The predicted effects of changing diameter and hematocrit as a function of different split rules applied to a single bifurcation.

Left) The dispersion of hematocrit in daughter branch 2 (H2) is somewhat altered by increasing the feed hematocrit (H1). Middle) H2 is more strongly influenced by diameter variations when closer to the other daughter branch (10 μ m) than when approaching the feed diameter (50 μ m). Right) The comparison of all split rules (except Plouraboue) on overall flow rate as a function of diameter ratio. A slight difference can be observed amongst all methods as a function of diameter, however all split rules give similar trends.

Table 7.30: Statistical analysis of networks used for biphasic blood flow simulations

Dataset	nPoints	nFaces	nInlets	nOutlets	MaxDia	MinDia	nBifurc	nUnion	nMultifur
Bifurcation	4	3	1	2	30	15.00	1	0	0
Trifurcation	5	4	1	3	15	9.00	0	0	1
Loop	6	6	1	1	15	10.00	1	1	0
Arteries	42	41	1	21	50	13.60	20	0	0
ArteriesV2	432	431	1	216	50	4.60	215	0	0
ArteriesV3	612	611	1	306	50	3.99	305	0	0
ArtVenFused	1486	1966	1	1	50	3.16	962	482	0
Mesentary	3037	3122	1	31	50	3.16	202	84	0
AU	129282	134229	7	3	50.0	0.33	8542	14518	654
AV	153112	159668	2	3	58.2	1.00	11631	22807	725
CO	28015	39547	6	10	60.0	0.38	19080	11774	1868
DB	127168	138707	1	9	70.0	0.53	20120	19895	1440

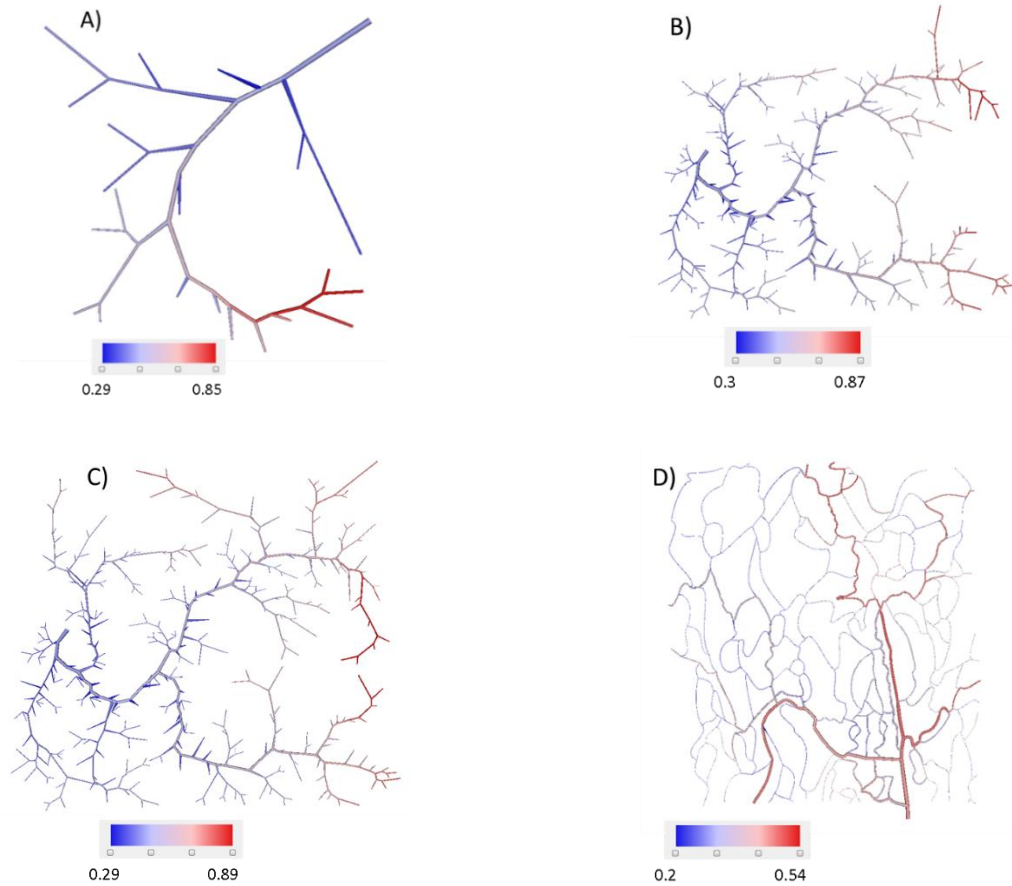


Figure 7.82. The network effect of hematocrit distribution.

A) a small arterial tree, B) a large arterial tree, C) a very large arterial tree, and D) a mesenteric network all reveal a consistent trend of hematocrit concentrating into the larger vessels. The network effect gives a nontrivial distribution of hematocrit and a general increase in hematocrit levels as branching order increases.

7.12.4 Comparison of plasma skimming models to empirical data

There are multiple criteria which the biphasic blood flow models can be compared to the empirical data. Experimental data on RBC distribution at bifurcations focuses on the RBC flux fraction, or the hematocrit in the daughter branch, as a function of the flow split. In Figure 7.83 this relationship was plotted for two bifurcations, in order to compare the trends for all split rules.

The RBC flux prediction by all empirical models (Pries, Fenton, and Dellimore) seem to indicate a higher RBC flux across all flow splits than Linninger and Gould's model. All empirical formulae yield an increase in the hematocrit ratio with the corresponding fractional flow. However, Linninger and Gould's model seems to yield an ideal hematocrit split (like a dye) in the case of a bifurcation with the same diameters. In the first case study, it appears as though the smaller branch increased its hematocrit alongside its corresponding flow.

The main difference between the models is that the empirical formulae predict a much smaller hematocrit delivered at low flows in the smaller diameter, when compared to the Linninger and Gould model.

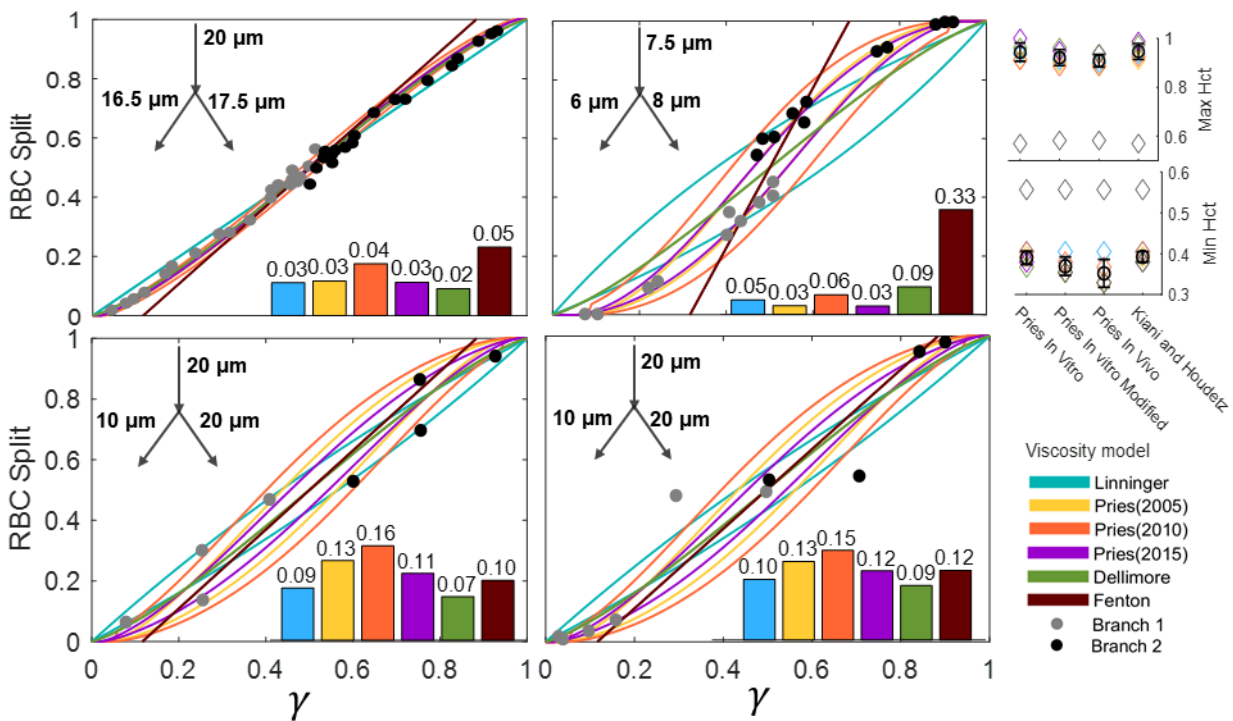


Figure 7.83: Comparing the variation in hematocrit split (top) and RBC flux fraction (bottom) with the flow split.

Left) Comparison of bifurcations with different diameters. Right) The bifurcation contains daughter branches of the same diameters. All errors reported are RMSE errors which were calculated following methods described in Section 7.13.1.

7.12.5 Closer look: variation of viscosity with diameter

The variation of diameter shows a clear trend in all viscosity values to approach infinity as diameter approaches 0. The viscosity has a “lubrication” effect at a specific diameter range where the viscosity is at a local minimum value. The profile at 0.4 hematocrit for a variety of diameters is offered in Figure 7.77. The variation with diameter at high hematocrit (Figure 7.85) reveals an increase in the rate of local minima (saddle points) along the diameter spectra as hematocrit approaches 1.

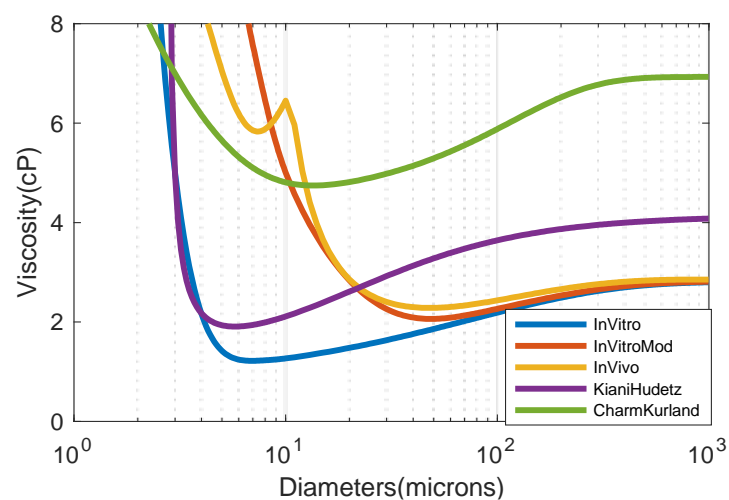


Figure 7.84. General trends of viscosity over diameter for a hematocrit of 0.4. As diameter approaches 0, the viscosity in all models approaches infinity.

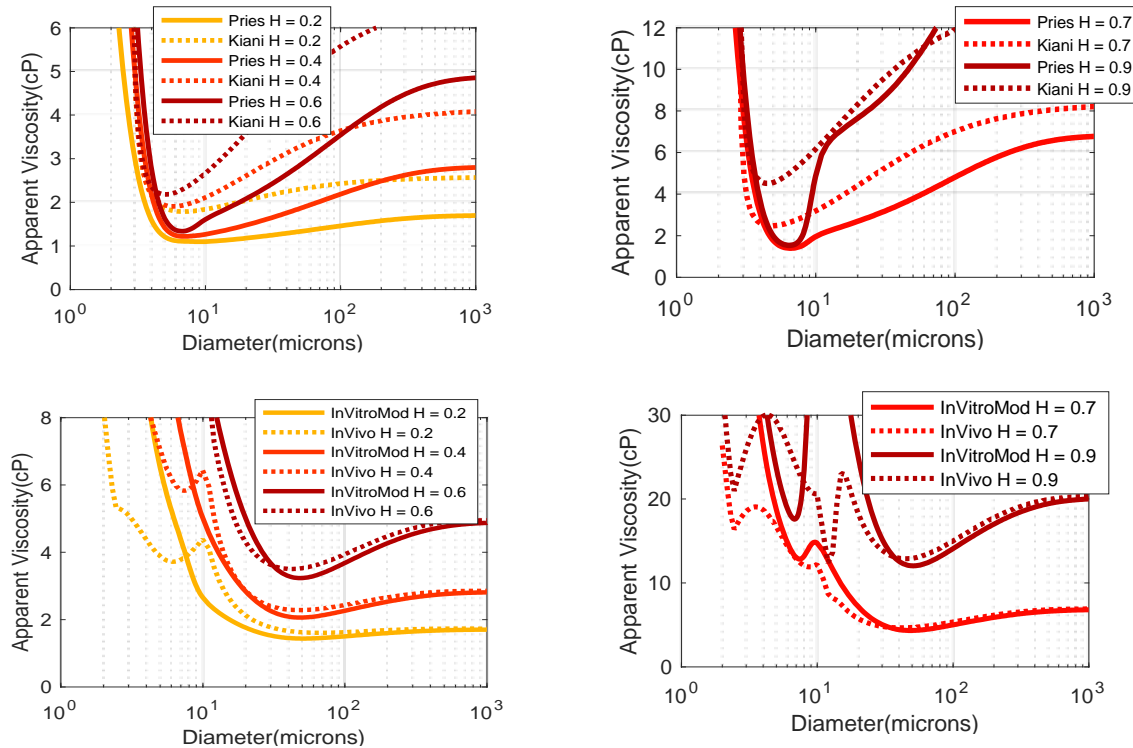


Figure 7.85. Viscosity comparison between Pries in vitro modified and Pries in vivo models. Left column) low-mid hematocrit. Right column) high hematocrit. High hematocrit yields a much higher viscosity across all diameters. At high values of hematocrit, the models exhibit several local maxima.

7.12.6 Closer look: variation of viscosity with hematocrit level

The viscosity formulae are heavily dependent on the hematocrit level. As shown in Figure 7.86 and Figure 7.87, as the hematocrit increases above 0.45, the equations for viscosity present more erratic behavior. This is another driving force behind the dynamic “m” value that has an asymptotic value of ~ 0.9 hematocrit as explained in Section 7.12.2.8. Limiting the hematocrit helps ensure the results lie in a physiologically meaningful range and also stabilizes the solution vector during convergence.

Kiani and Hudetz predictions are seen to stop at the $2.75\mu\text{m}$ threshold. Pries’s in vivo and original in vitro models also seem to yield a rather low viscosity for diameters between $1\text{-}2\mu\text{m}$.

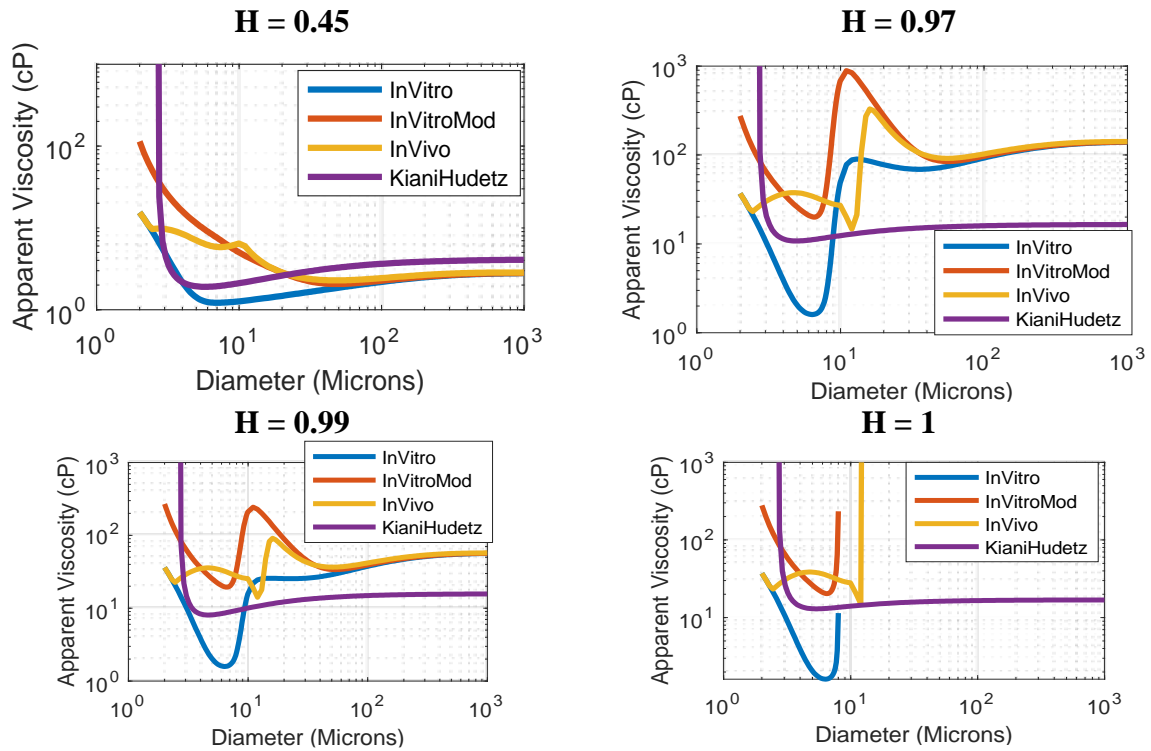


Figure 7.86. Predicted viscosity by all models at high hematocrit values.

The in vivo and modified in vitro formulae predict a rise in viscosity at around $15\mu\text{m}$ due to the glcocalyx. The in vitro formula also yields a maxima around $8\mu\text{m}$. Kiani and Hudetz seems to be fairly constant across all hematocrit levels.

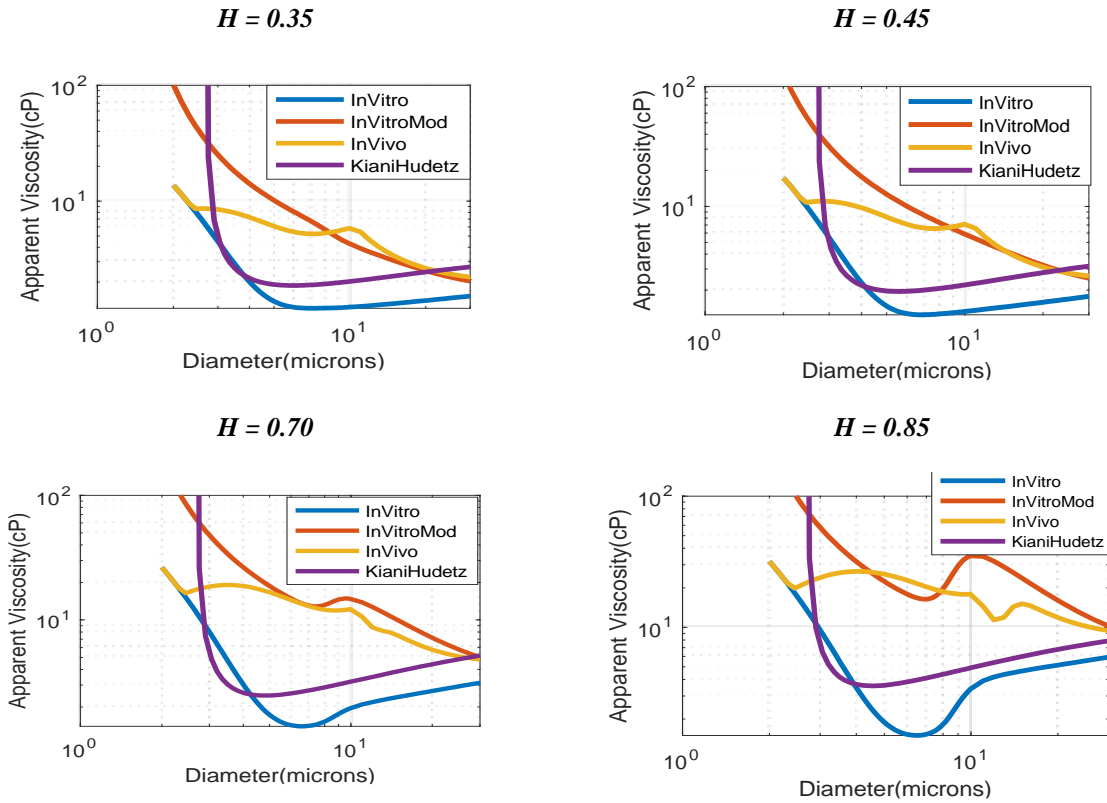


Figure 7.87. Predicted viscosity by all models for diameters $< 20\mu\text{m}$.

Note the trends are nonlinear and, in some cases, almost discontinuous. The Kiani and Hudetz model is the most stable, but only applies for a narrow physiological range of diameter. The Pries In-Vitro model is the next most stable choice for viscosity.

7.12.7 Closer look: Variation in plasma skimming trends with diameter:

The geometry of a bifurcation plays an important role in determining the ratio of RBC splitting into the daughter branches. In Figure 7.88, the resulting flows and hematocrit values are computed using all split rules and one viscosity for a bifurcation whose daughter branch diameter is varied.

The predicted overall flows are very similar in all split rules for all diameters. The differences become more apparent in the fractional hematocrit graphs. Fenton's model predicts an ideal split for both branches, even though the diameter changes. Pries's model yields a hematocrit of 0 at

diameters below $8\mu\text{m}$, and Linninger's model predicts a higher hematocrit in the larger daughter branch.

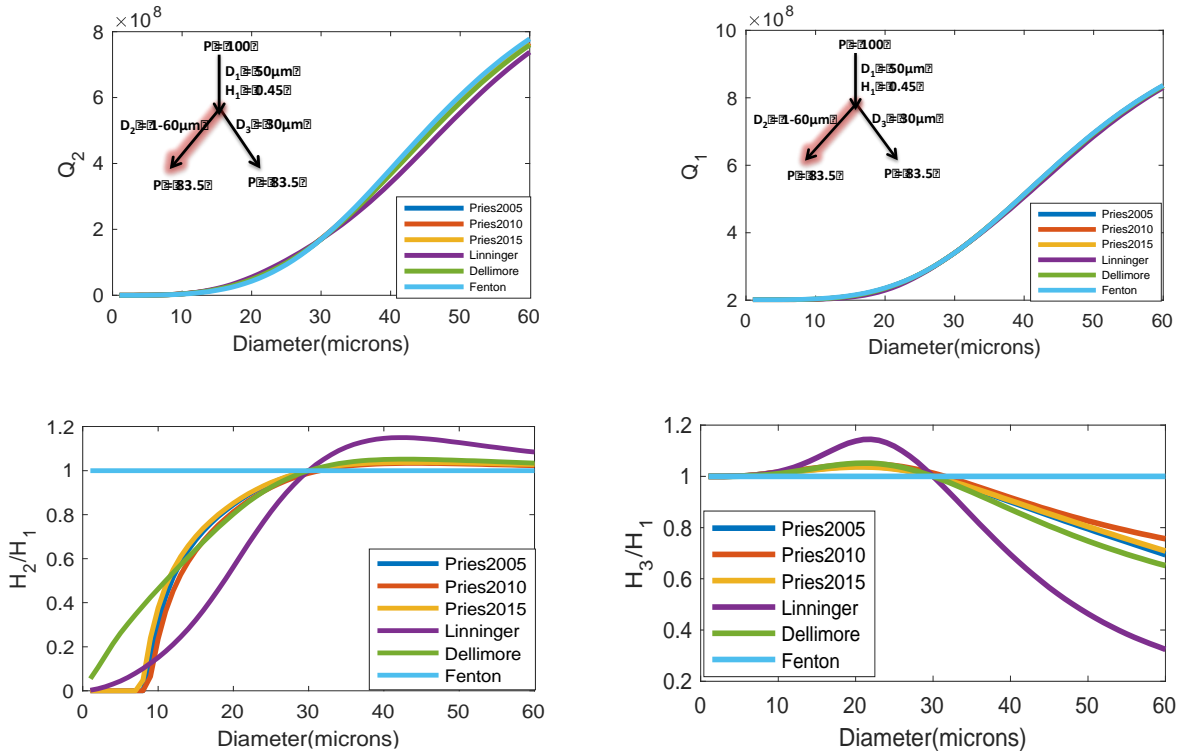


Figure 7.88. Branch hematocrit and flow variation as a function of the diameter for one bifurcation.

Top) the flows were similar for the first and second daughter branch. The bottom figures show the hematocrit ratio as a function of the daughter branch diameters. The viscosity used was Pries In-Vitro

7.12.8 Closer look: Balogh and Bagchi (3D immersive boundary method)

The mathematics of the Balogh and Bagchi method can be described in brief as the application of the immersed boundary method (described elsewhere in detail) for simulating RBCs, white blood cells (WBCs) and platelets. One of the key advantages of the IBM is that it allows a simulation of fluid-structure interaction (FSI) without body-fitted meshing (which can be extremely complicated on a dynamic simulation).

In essence, the bulk flow is calculated in the Eulerian grid (Cartesian coordinates) before the deformation and advection of the RBCs is computed on their own Lagrangian grids. The Lagrangian grid is analogous to an inertial reference frame in physics, i.e. it is computed as if the RBC grid is stationary and all motion/activity/forces are moving around the object). The RBC is allowed to deform due an elasticity law yet its swelling (increase or decrease in surface area) is limited by a very large expansion coefficient. An FEA solver is used to assess the shear and tensive stresses on the surface in order to assess deformation of RBC in each time step. The mathematics are as follows for wall strain energy:

$$W_s = \frac{G_s}{4} [I_1^2 + 2I_1 - 2I_2 + CI_2^2] \quad (7.161)$$

Where G_s is the elastic modulus, C is a controlling parameter that, when set to a large value, limits the dilation of the membrane. I_1 and I_2 are the strain invariants of the Green strain tensor (explained next) and are calculated following:

$$I_1 = \epsilon_1^2 + \epsilon_2^2 - 2 \quad (7.162)$$

$$I_2 = \epsilon_1^2 \epsilon_2^2 - 1 \quad (7.163)$$

And the Green strain tension is defined as:

$$E = F^T \cdot F - I \quad (7.164)$$

Where

$$F = \frac{dx}{dX}$$

Where x is the current configuration and X is the original (baseline) configuration of the membrane. The stress can be evaluated as:

$$\tau_1 = \frac{1}{\epsilon_2} \frac{dW_s}{d\epsilon_1} \quad (7.165)$$

$$\tau_2 = \frac{1}{\epsilon_1} \frac{dW_s}{d\epsilon_2} \quad (7.166)$$

Which is solved with a finite element method. These equations can be interpreted as the strain being linearly proportional to the amount of deformation from baseline, where the two portions are shear strain and dilation strain. These values of strain give rise to quantifiable forces in each element as:

$$f_e = \sum_n \int_{s_n} \frac{dN}{dX} \cdot P dS \quad (7.167)$$

Where N is a vector of shape functions, S is the surface area of n triangles surrounding the node, and P is stress tensor (Piola-Kirchhoff) defined as:

$$P = \epsilon_1 \epsilon_2 \tau \cdot F^{-T} \quad (7.168)$$

And cell resistance to bending:

$$W_b = \frac{E_b}{2} \int_S (2\kappa - c_0)^2 dS \quad (7.169)$$

Where E_b is bending modulus, assumed $2 \cdot 7 \cdot 10^{-19}$ J, κ is the mean curvature, c_0 is the spontaneous curvature, and S is the surface area of the cell membrane. Cell bending force is calculated with:

$$f_b = E_b [(2\kappa + c_0)(2\kappa^2 - 2\kappa_g - c_0\kappa) + 2\Delta_{LB}\kappa] n \quad (7.170)$$

Where κ_g is the Gaussian curvature (no idea what that is), Δ_{LB} is the Laplace-Beltrami operator (similar to Laplace operator but for discrete grids) and n is the normal vector.

The Gaussian curvature is approximated using Gauss theorem on a small surface patch. The final amount of bending, deformation and dilation are calculated as a linear summation of each sub-force. The bulk flow is calculated using a modified Stokes equation with a sink term (F), which can be interpreted as a momentum loss due to deformation, given that the sink term is the integral of the membrane forces over the deformation:

$$\nabla \cdot u = 0 \quad (7.171)$$

$$\rho \frac{du}{dt} = -\nabla p + \nabla \cdot \mu [\nabla u + \nabla u^T] + F \quad (7.172)$$

$$F = \int_S f_m \delta(x - x') dx' \quad (7.173)$$

And with a cosine-derived 3-dimensional Dirac delta function (see reference within for more details on cosine function):

$$\delta(x - x') = \frac{1}{64\Delta^3} \prod_{i=1}^3 1 + \cos\left(\frac{\pi}{2\Delta} [x_i - x'_i]\right) \quad (7.174)$$

Where Δ is the edglength of the Lagrangian mesh. One way to interpret the use of a cosine function could be chosen purely for the shape, being a value of 1 at the node and 0 at a value of 2Δ away.

The RBC nodes (Lagrangian nodes) are advected using the interpolated flow field from the Stokes equations above (u). Because the two mediums (cytoplasm within the RBC and plasma outside the RBC) have different viscosities, this will affect the deformation and flow of the RBCs. The dynamic viscosity field is computed as follows:

$$\mu(x, t) = \mu_p + (\mu_c + \mu_p)I(x, t) \quad (7.175)$$

$$\nabla^2 I = \nabla \cdot G \quad (7.176)$$

Where μ_c is the viscosity of cytoplasm, μ_p is the plasma viscosity, and I is the indicator function (1 inside RBC, 0 otherwise). G is the eulerian variable from cell surface normals (n):

$$G(x, t) = \int_S \delta(x - x') n dS \quad (7.177)$$

The solving of the field for I is proposed in all cases. As G is straightforward to compute, then I can be computed, then viscosity. The finite volume method was decidedly a limiting factor in the time step, so a cell-center base is considered preferential. This was considered by an averaging scheme between the mixed Eulerian-Lagrangian grids to evaluate the derivatives:

$$\nabla \cdot [\mu(\nabla u + \nabla u^T)]_x = \mu \nabla^2 u + 2 \frac{d\mu}{dx} \frac{du}{dx} + \frac{d\mu}{dy} \left(\frac{du}{dy} + \frac{dv}{dx} \right) + \frac{d\mu}{dz} \left(\frac{du}{dz} + \frac{dw}{dx} \right) \quad (7.178)$$

Where it appears that u , v , and w correspond to Eulerian grid velocities in x , y and z direction. What is claimed to be an averaging of the gradient can be interpreted as a numerical evaluation of the derivative. For more information regarding the averaging technique see Equation 16 from original manuscript (considered lengthy and obvious, so omitted from this review).

The time integrator uses the Adams Bashford method for the conservation equation and a Crank-Nicholson scheme for the convection-diffusion equation. The solving step uses a modified form of the equations:

$$\rho \frac{u_i^* - u_i^n}{\Delta t} = G_i + \frac{3}{2} \alpha_i^n - \frac{1}{2} \alpha_i^{n-1} + \frac{1}{2} (\beta_i^* + \beta_i^n) \quad (7.179)$$

$$\frac{d^2 \phi}{dx_i dx_i} = \frac{\rho}{\Delta t} \frac{du_i^*}{dx_i} \quad (7.180)$$

$$\rho u_i^{n+1} = \rho u_i^* - \Delta t \frac{d\phi}{dx_i} \quad (7.181)$$

Where

$$\alpha_i = F_i + \frac{d}{dx_i} \left(\mu \frac{du_j}{dx_i} \right) \quad \beta_i = \frac{d}{dx_j} \left(\mu \frac{du_i}{dx_i} \right) \quad (7.182)$$

This implementation appears similar to the SIMPLE algorithm where an intermediate (guess) velocity field (u_i^*) is computed, then the continuity is computed, then the velocity field is evaluated for correctness, all in each iteration which is cycled until convergence.

The ϕ term in the equations is related to the pressure, but is not pressure. The choice of the function G_i that accounts for the pressure gradient changes how closely related ϕ and pressure are. This can be interpreted as a function similar to a preconditioner, where the coordinate system is transposed so that the system is easier to solve. The authors use $G=dp/dx$, so there is no conditioning and no need to update (de-condition) the parameters post-solving.

The implementation of the ghost nodes (nodes on the rigid bodies of vascular wall) and boundary conditions are claimed to break the tri-diagonal matrix pattern, so an explicit method is used to account for the velocity at the ghost node. Because the ghost node is inside a rigid body and the no-slip ($u=0$) BC is implemented at the boundary interface, the value of u inside the ghost node is nonzero. To calculate this, the velocity is:

$$u_{gn} = 2u_{bi} - \sum_{i=1}^8 \beta_i u_i \quad (7.183)$$

Where β_i is not defined, gn stands for ghost node, and bi is boundary interface. This is the linear interpolation of the 8 nodes at the corners of a hexahedron, which are weighted by the beta factor. Moreover, the ghost node is updated with a similar method to the fully explicit Euler solver:

$$u_{gn}^* = u_{gn}^n + \Delta t \frac{d\phi}{dx_i} \quad (7.184)$$

The deformation of the RBC is calculated using conservation of momentum. It is interesting to note that a detailed, complex explanation was offered to simply account for how the Lagrangian and the Eulerian systems are related by a linear coordinate transformation (by an offset vector and rotational matrix).

The validations are performed on analytic solutions. The first one is a solid, immovable sphere with a flow field around it. It is noticed that the largest error (range 0.001-0.0001) is immediately adjacent to the sphere, something that was acknowledged by the authors and considered an expected side effect of the IBM. This is important, because this can greatly affect the results when deformed near a bifurcation. An important discretization claim in their method is that 20-40 discretizations per sphere appears to result in a reasonable residual. Another important conclusion is that the resolution of the Eulerian mesh and Lagrangian mesh need to be on the same order (same dx , dy and dz) for best results. Importantly, in the bifurcation example offered in case study 4, it was noticed that the RBCs have much more deformation than the videos of RBCs available online. Given that this is directly proportional to the viscosity, and thus the flow and bifurcating pattern, this may cause significant error in the results. Importantly, the “clear” capturing of the

sigmoidal curve by their experiment is challenged by the discontinuity at around 0.7 flow split ratio.

Of the many validations, one included the validation against experimental deformability of a single RBC through a rigid channel, with a good fit to the data. Another included showing cell-selective sorting from a microfluidic device. Lab on a chip trends were also given, but these were similar results to the cell separation microfluidic device.

7.13 Appendix M: Methods for interrogating model predictions

Many models have been developed for explaining data (data fitting, data simulating) and exploring physical domains (exploratory simulations). In the former, it is common practice to identify the best model to explain the results of an experiment. In the latter, data review is imperative to understanding the output of the simulation. This section investigates different methods for evaluating the quality of data fitting methods in section 7.13.1 and best practices for reviewing simulation results in Section 7.13.2.

7.13.1 Measuring goodness of fit

This section will give an overview of methods for calculating the “goodness of fit” between a model and experimental data. Models for linear and nonlinear fitting will be reviewed and an explanation of how excel implementation calculates a goodness of fit. Note, the findings indicate that the coefficient of determination is only useful in a linear model, and when comparing models of different orders to fit the same data, the extra coefficients must be given a penalty to avoid excessive (and meaningless) coefficient usage.

7.13.1.1 The coefficient of determination (R^2) for linear fit:

A best-fit curve can be described by the error between the fit and the data normalized to the data variance as a percentage of fit (i.e. 1 = 100% fit and 0 = 0% fit). This model, by incorporation of the variance, reflects the data inconsistency and does not penalize the goodness of fit for a large variance. The variance is a measure of the variability of the data with itself. For example, if the

average data values are ~1 and the variability is 5, the variability is large compared the data value, whereas if the average data values are ~1000, a variance of ± 5 would not be a large impact.

$$R^2 = \frac{\text{Variance explained by the model}}{\text{Total variance}} \quad (7.185)$$

Generally Accepted Nomenclature

$$R^2 = 1 - \frac{err}{var}$$

$$err = \sum_{i=1}^{nData} (f(x_i) - \tilde{f}(x_i))^2$$

$$var = \sum_{i=1}^{nData} (\tilde{f}(x_i) - \overline{f(x)})^2$$

Preferred Nomenclature

$$R^2 = 1 - \frac{r^2}{var} \quad (7.18)$$

6)

$$r^2 = (Ax - b)^T (Ax - b) = r^T r \quad (7.18)$$

7)

$$var = \sum_{i=1}^{nData} \left(b_i - \frac{\sum_{j=1}^{nData} b_j}{nData} \right)^2 \quad (7.18)$$

8)

Where:

$f(x_i)$ = polynomial evaluation at x_i

$\tilde{f}(x_i)$ = measured data at x_i (at i data point)

$\overline{f(x)}$ = mean of measured data

The steps to solve for the coefficient of determination (involving solving for the model coefficients, plugging the independent variable back into the model and calculating the difference) can be computed in one step (analytic derivation not shown) by squaring the r value calculated as follows:

$$r = \frac{n(\sum xy) - (\sum x)(\sum y)}{\sqrt{[n\sum x^2 - (\sum x)^2][n\sum y^2 - (\sum y)^2]}} \quad (7.189)$$

7.13.1.2 Adjusted coefficient of determination (for use with higher order fits):

There is another accepted form of the R^2 value, known as the “adjusted” R^2 value, which takes into account the number of terms in the polynomial. This is calculated as follows:

$$R_{adj}^2 = 1 - \frac{n-1}{n-p} \cdot \frac{err}{var} \quad (7.190)$$

Where n is the number of observations and p is the number of parameters in the linear regression fit.

There is a body of literature, however, that suggests that for nonlinear fitting, higher order models, and models that span a large domain in the independent variable space cannot be accurately described by the coefficient of determination, because the variation in the dependent variable is not a reflection of the variance of the data amongst itself. i.e. the data naturally has a large range in the dependent variable space which leads to a high R^2 value for good and bad fits alike (due to a high variance amongst the data). An example of this is a measurement among a year-long biological experiment, where the extremely long time scale naturally results in a large difference between the values at the beginning of the experiment and the end of the experiment,

which is not a measure of the volatility of the data itself. For these cases, the following variation is capable of reflecting this information:

$$\begin{aligned}
 ar &= \sum_{i=1}^{nData} (r(x_i) - \overline{r(x)})^2 \\
 &= \sum_{i=1}^{nData} \left(r(x_i) - \frac{\sum_{j=1}^{nData} r_j}{nData} \right)^2
 \end{aligned} \tag{7.191}$$

7.13.1.3 Nonlinear goodness of fit Akaike Information Criterion (AIC):

A history of models to determine goodness of fit between data fitting methods is beyond the scope of this chapter, however for more information, the reader is referred to a history described concisely by Spiess and Neumeyer [229]. Here, the Akaike Information Criterion (AIC) is considered the generally accepted method to measure goodness of fit in higher order models as described by:

$$AIC = 2p - 2\ln(L) \tag{7.192}$$

$$\ln(L) = \frac{1}{2} \left\{ -n \left[\ln \left(\frac{2\pi}{n} \right) + 1 + \ln \left(\sum_{i=1}^n r_i^2 \right) \right] \right\} \tag{7.193}$$

Here, p is the number of parameters and $\ln(L)$ is the maximum log-likelihood of the estimated model. This AIC variable acts as a “goodness of fit” value that has a penalty for having a high residual (r^2) and a high number of parameters (p). The model has a “forgiveness” term that lowers the residual penalty by the number of data points in the set (first logarithmic term in the likelihood

function). In this way, a large sample size with a large error is not a problem. Having a lower AIC value is desirable (better fit), and the more negative the value the better the model fits the data.

7.13.1.4 Linear Regression Examples:

Simple Line. A linear first order model is written in matrix form in Equation (7.194). This can also be represented as a linear system as shown in Equation (7.195) where k and d are unknown parameters representing the slope and y-intercept of the model respectively. The source data is also provided. The results are shown graphically in Figure 7.89, and summarized in Equation (7.196).

$$y(x) = k \cdot x + d \quad (7.194)$$

$$\begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ x_3 & 1 \\ x_4 & 1 \end{bmatrix} \begin{bmatrix} k \\ d \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} \quad (7.195)$$

Table 7.31. Data for linear first order fitting problem

x	1	2	3	4
y	2.1	2.9	6.1	8.3

$$\begin{aligned} k &= 2.18 \\ d &= -0.6 \end{aligned} \quad (7.196)$$

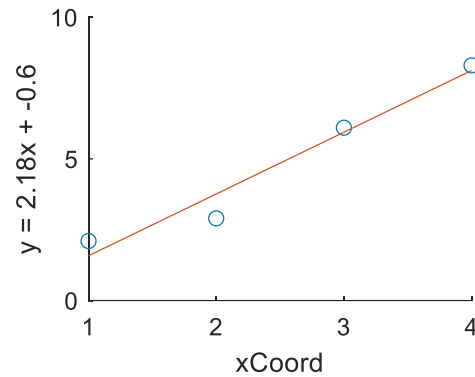


Figure 7.89: Linear first order fitting between data points (blue circles) and line of best fit (orange line) using linear regression.

A Matlab code is offered:

```
A = [1 1;2 1;3 1;4 1];
b = [2.1; 2.9; 6.1; 8.3];
x = inv(A'*A)*A'*b
figure
scatter(A(:,1),b);hold on
plot(x(1).*A(:,1)+x(2));xlabel('xCoord');    ylabel(['y      =      ',num2str(x(1)),'x      +
',num2str(x(2))]);
```

Polynomial curve. In order to find the best fitting quadratic model for the following 8 measurements (Table 7.32) can be accomplished with Equation (7.197) in component form and Equation (7.198) in linear algebraic form. The rectangular matrix A may be solved for using the pseudo-inverse method, $x = (A^T A)^{-1}(A^T b)$. By multiplying both sides of Equation (7.198) by A^T the new matrix of constant coefficients becomes a square matrix $(A^T A)$, which may be inverted, given that it is not still singular.

$$y(x) = aa_1 + aa_2 \cdot x + aa_3 \cdot x^2 \quad (7.197)$$

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \\ 1 & 4 & 16 \\ 1 & 5 & 25 \\ 1 & 6 & 36 \\ 1 & 7 & 49 \end{bmatrix}}_A \underbrace{\begin{bmatrix} aa_1 \\ aa_2 \\ aa_3 \end{bmatrix}}_x = \underbrace{\begin{bmatrix} 0 \\ 4.1 \\ 8.8 \\ 15.7 \\ 25.7 \\ 36.8 \\ 48.1 \\ 63 \end{bmatrix}}_b \quad (7.198)$$

Table 7.32. Data for linear second order fitting problem

x	0	1	2	3	4	5	6	7
y	0	4.1	8.8	15.7	25.5	36.8	48.1	63

$$x = \begin{bmatrix} 0.0417 \\ 2.7631 \\ 0.8893 \end{bmatrix} \quad (7.199)$$

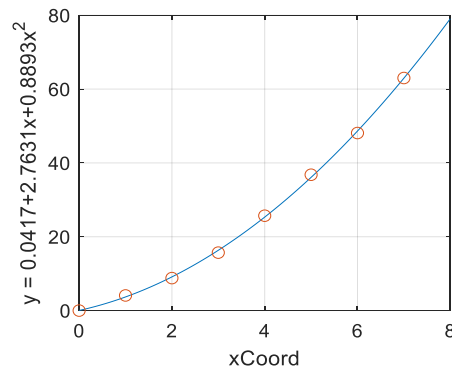


Figure 7.90: Second order linear fitting model (blue line) and source data (orange points).

A Matlab code is offered:

```
close all; clear all;
%Calculating the pseudoinverse
A = [1 0 0; 1 1 1; 1 2 4; 1 3 9; 1 4 16; 1 5 25; 1 6 36; 1 7 49];
b = [0; 4.1; 8.8; 15.7; 25.7; 36.8; 48.1; 63];
x = inv(A'*A)*A'*b
%Plotting the pseudoinverse
r = linspace(0,8,100);
plot(r,polyval([0.8893 2.7631 0.0417],r)); hold on; scatter(A(:,2),b);
xlabel('xCoord'); ylabel('y = 0.0417+2.7631x+0.8893x^2'); grid on;
```


Linear fit of a Plane in a 3D Space. In order to find the best fitting linear plane in three dimensions using measurements from Table 7.33 using the model in Equation (7.200). The solution is given in Equation (7.201). This solution was found using linear regression.

$$z(x, y) = \alpha + \beta x + \gamma y \quad (7.200)$$

Table 7.33. Data for linear fitting problem in 3D

x	-1	2	5	8	10
y	0	3	6	10	12
z	1	4	7	11	15

$$x = \begin{bmatrix} 0.5000 \\ -0.2000 \\ 1.3000 \end{bmatrix} \quad (7.201)$$

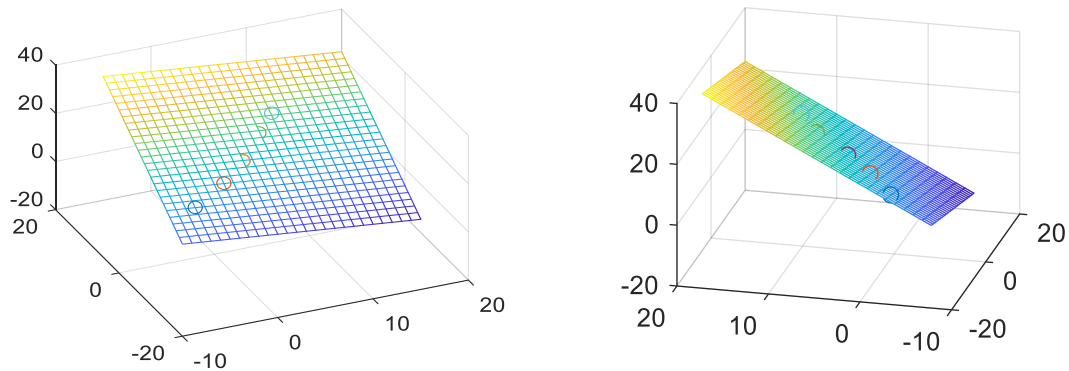


Figure 7.91: Linear fit of a plane through 5 points in 3D.

A Matlab code is offered:

7.13.1.5 Validation by direct minimization of residual error surface

$$\begin{cases} \frac{\partial \phi}{\partial a a 1} = 0 \\ \frac{\partial \phi}{\partial a a 2} = 0 \\ \frac{\partial \phi}{\partial a a 3} = 0 \end{cases} \quad (7.202)$$

```
syms aa1 aa2 aa3
A=[1 0 0;1 1 1;1 2 4;1 3 9;1 4 16;1 5 25;1 6 36;1 7 49];
b=[0;4.1;8.8;15.7;25.7;36.8;48.1;63];
x=[aa1;aa2;aa3];
r=A*x-b;
r1=transpose(r);
phi=r1*r;
e1=diff(phi,aa1);
e2=diff(phi,aa2);
e3=diff(phi,aa3);
display(e1);
display(e2);
display(e3);
%           We get      e1=16*aa1+56*aa2+280*aa3-2022/5
%                      e2=56*aa1+280*aa2+1568*aa3-10852/5
%                      e3=280*aa1+1568*aa2+9352*aa3-63304/5

[aa1,aa2,aa3]=solve(e1,e2,e3)

% Solve aa1,aa2 and aa3 from “e1=0,e2=0 and e3=0”%
%           We get
%           aa1 = 1/24=0.0417      aa2 =2321/840=2.7631      aa3 =249/280=0.8893
%           Therefore the model is
%           Y=0.0417+2.7631x+0.8893x^2
```

7.13.2 Investigating simulation results

7.13.2.1 3D immersive visualizations

Once the 3D state fields and flow vectors have been calculated, the data needs to be analyzed for general trends. The initial investigation begins with 3D visualizations of the fields in an immersive visualization environment designed by our lab [180,230]. This can be seen in Figure 7.92. As can be noted, this is only a qualitative investigation and a quantitative investigation must follow. Many of the common methods of analysis are defined and implemented next.

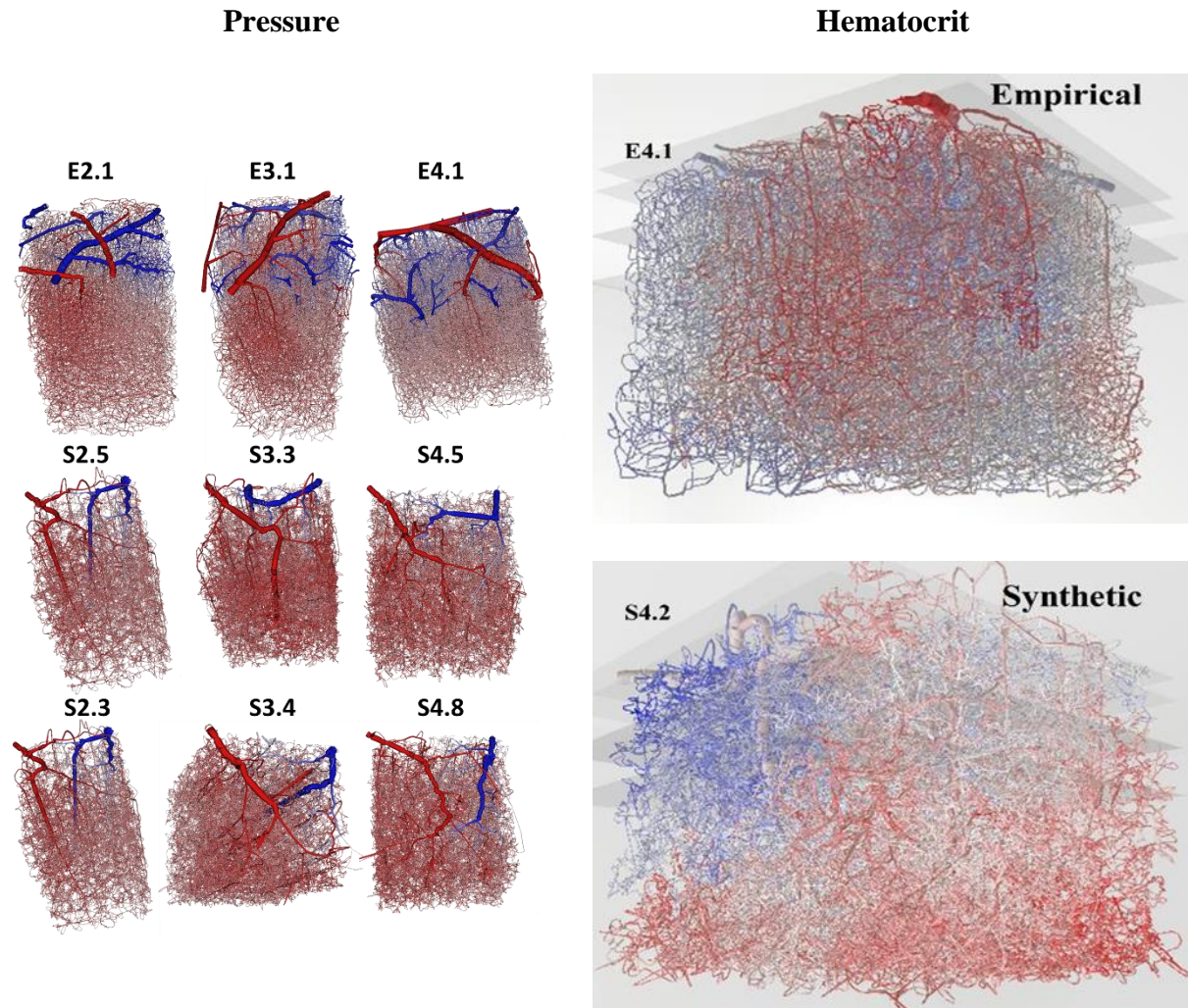


Figure 7.92. Blood pressure and hematocrit visualized in an immersive 3D environment for qualitative analysis of global trends.
 Note, these results indicate a stronger hematocrit (more red) in the lower layers of the cortex than at the surface. These results are expanded more in an accompanying paper [27].

7.13.2.2 Calculating Perfusion

Perfusion to the brain is frequently reported in units of ml/100g/min which translates to how much blood is flowing through a given tissue over a given time. The mass of brain tissue is measured in weight because the soft tissue of brain is more easily weighed with a scale than submerged and volumetric displacement evaluated, given that the tissue has pockets in the

gyrations and thus is less dense than the surrounding water. This perfusion is basically dividing the flow of a network by the volume of the bounding box of the network and scaling by a literature-derived conversion factor between brain volume and brain mass. A common selection for this conversion factor was originally referenced by Gould in 2015 as $6,000\mu\text{m}^3/\mu\text{m}^3/\text{s}$ compares to $1\text{ml}/100\text{g}/\text{min}$ translating to the well-established relationship between 1g of tissue per 1cm^3 (or 1ml) and 60s in 1 minute. The conversion from $\mu\text{m}^3/\mu\text{m}^3$ to ml/ml is dimensionless.

7.13.2.3 Path analysis

Path analysis is an efficient method for tracking the states traversed during the flow history through the cerebral microcirculation [39,63,90]. Such investigations can help identify trends in the network that would otherwise go unnoticed, as the individual segments may not reveal these trends. Moreover, a robust algorithm for path analysis must be developed to handle special cases as discussed next.

Path analysis algorithm. A path analysis algorithm must be able to traverse a tree not only using topological information, but should account for errors in topological annotations (improper face direction). For example, when storing a complete Circle of Willis as a network, each vessel is prescribed two points (a start point and an end point) and an orientation. If the modeler stores the wrong orientation, the flow will have a negative value (as in Figure 7.93). In this way, a structure can only be interrogated after a simulation completes.

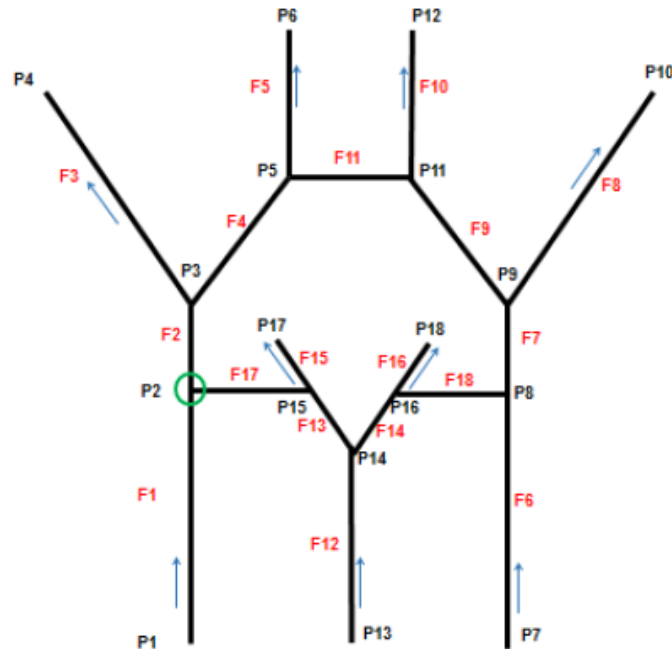


Figure 7.93. Anatomical network of a complete Circle of Willis (lengths not to scale) without diameter information.

Note, for arc labeled “F11”, either P5 or P11 has to be the starting point, a decision that is chosen by the modeler. If the modeler chooses wrong, the flow of that arc is simply the negative of the flow value.

The algorithm starts at a network inlet (points that have only one attached arc and the flow vector pointing into the network) and follows the flow downstream until it reaches an outlet terminal. The general steps are covered in Figure 7.94. Note, the sub-step that finds the outflowing segments of a given point cycles through attached segments with evaluation of flow vector and arc orientation to define which arcs flow out of the node.

The algorithm must traverse the vascular structure and store a dense matrix of segment indices. Each row of the matrix corresponds to a single path and each column index corresponds to the face placement in the path. This matrix can later be used to reference the solution vectors and interrogate path analysis of flow vectors. Likewise, the point indices can be stored along the same

path and the state vectors can be interrogated in the same manner. This method of storing indices rather than values avoids the need to recurse the network multiple times. This procedure is implemented in a button inside “*pathAnalysis.x64_Project.exe*” under the title “*FullPathAnalysis*”.

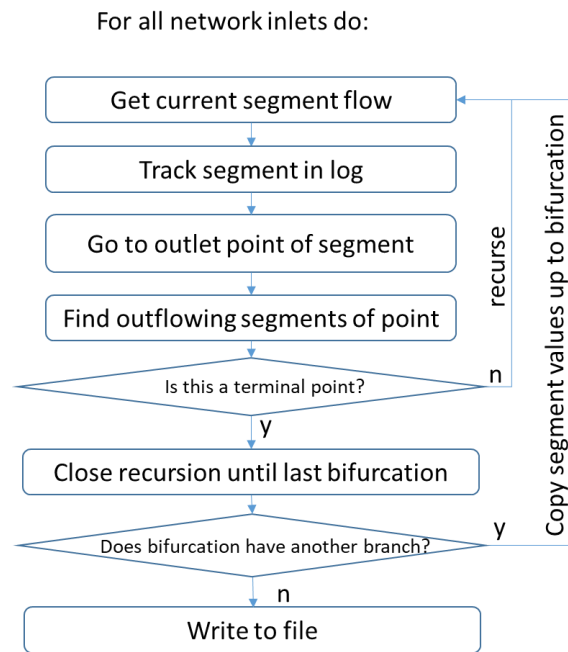


Figure 7.94. A workflow diagram for the path analysis.

Special case, multiple inlets and loops. Networks reconstructed from the mouse cortex include severed pial vessels feeding and draining the capillary bed. This leaves many inlets and outlets that impact the path analysis by increasing the number of paths. The highly interconnected trees of the microcirculatory bed reconstructions [41] also include many loops in the arterial and venous sides of the capillary bed. These loops compound the investigation of the path analysis, as all downstream paths from one side of the loop will be repeated for the choice of the other side of the loop. Moreover, the connections between the multiple inlets and each other inlet leads to further duplication. An example of an arterial loop is seen in Figure 7.96.

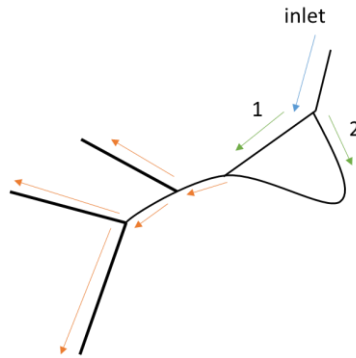


Figure 7.95. Example of loop in arterial tree.

The path analysis will repeat all orange paths for each choice of the green arrows in the loop, expressing the compounding problem when loops occur in the arterial tree.

In these cases, it is important to implement a skipping algorithm that allows the recursion to evenly sample paths dispersed throughout the network. This can be accomplished by tracking bifurcation depth. After an end terminal is found, a series of bifurcations identified by a multiple of the bifurcation depth (~ 0.1 - 0.2) are skipped before restarting the recursion process. This is enforced by triggering a Boolean variable when a terminal is found while tracking the bifurcation depth. After exiting the recursion due to finding a network terminal, the upstream bifurcations are skipped until the number of skips is the same or larger than the desired amount ($0.1 * \text{bifurcationDepth}$ for instance). This procedure is implemented with a button inside *pathAnalysis.x64_Project.exe* under the title *Path Analysis skipping*. The programmatic implementation of these procedures can be found in *ghPathAnalysisSource.V2.pas*.

7.13.3 Logarithmic and power scaling

In order to obtain consistently good visualizations, sometimes a linear coloration is not sufficient. In the case of blood flow in the microcirculation, for instance, the flow on an individual face can range as much as 4-6 orders of magnitude between the largest vessels (ml/min) to the smallest vessels (nL/min). To successfully account for this, a logarithmic scale (with a base of 10) is frequently implemented as the most logical way to focus the visualization across the entire spectrum instead of focusing on only the largest values. In the case of nonlinear trends, this works very nicely.

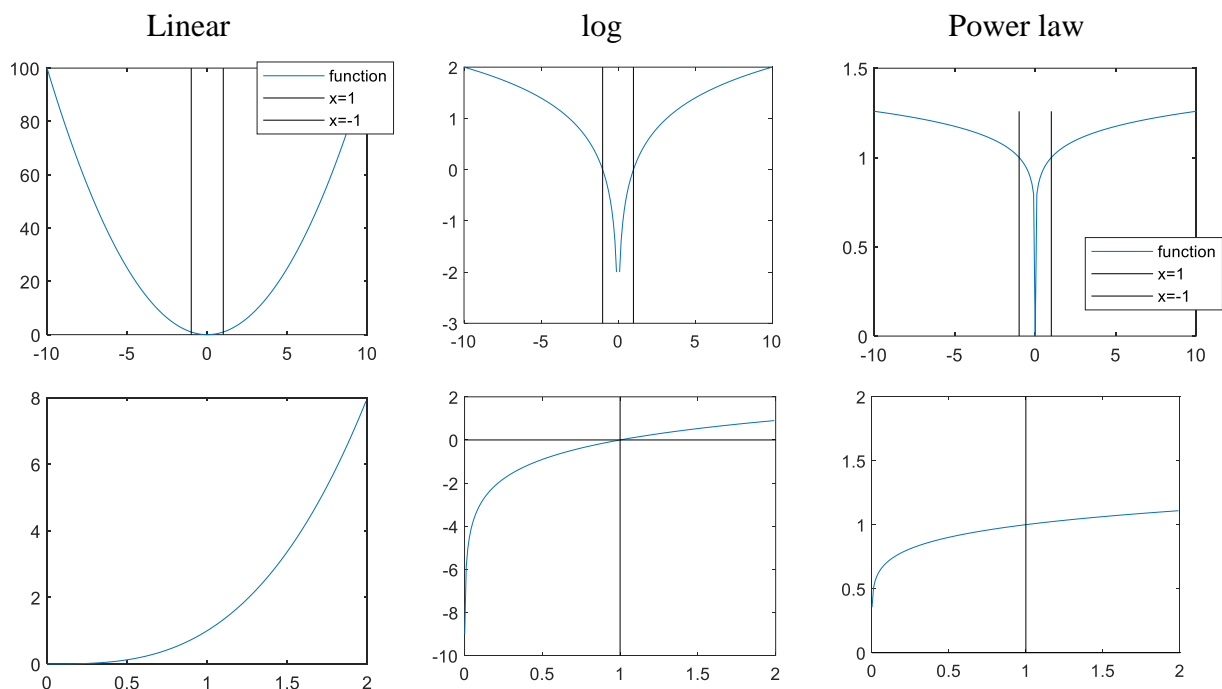


Figure 7.96. The comparison of linear, log, and power law between the values of -1 and 1 shows a power law can overcome the asymptote in the log plot at $y < 1$.

This method has an inherent drawback, however, as it approaches infinity when the value approaches 0, specifically it begins to diverge below a value of 1. This is exemplified by the

diagram below. To overcome this dilemma, a new coloration has been implemented with a *power scaling* which, instead of taking the logarithmic value of all elements of the vector, takes the power of all elements of the vector to a very small power (1/100 for instance). This new method is *positive definite* for all cases (non-divergent) as shown below.

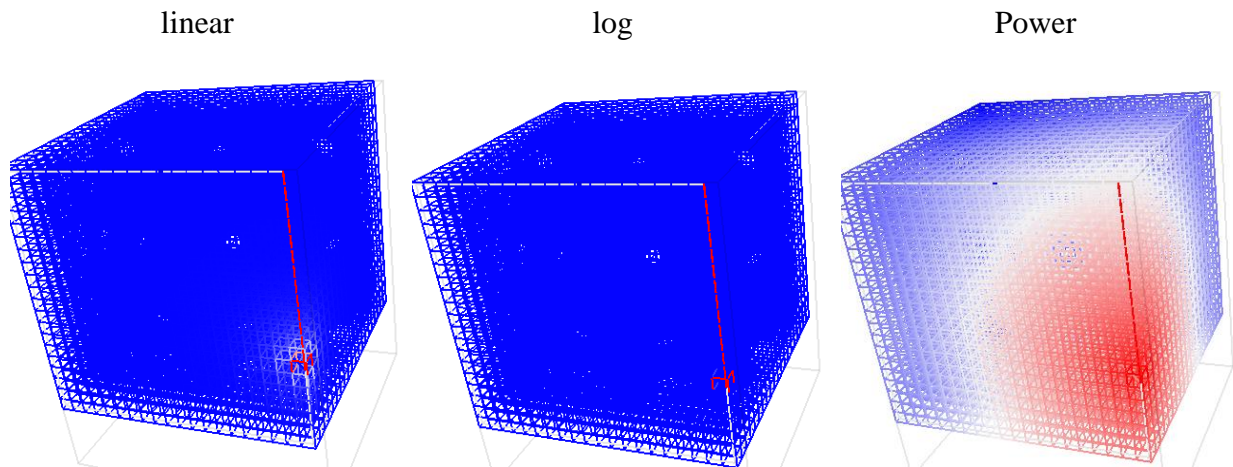


Figure 7.97. The coloration of the previously proposed log plots shows the power law has a distinct advantage over the log scaling. The skewedness of the values in the log spectrum as the values approach 0 (less than 1) is removed in the power scaling.

7.13.3.1 Surface flow divergence visualization

The blood flow in the pial vessels of the brain transfer flow laterally to other pial vessels and dive deep into the brain via penetrating arterioles. This section summarizes an approach to identifying the ratio of lateral to penetrating flow along the pial surface of the brain. In future work, this investigation could be used to identify changes in collateral blood supply at the level of major arteries. The implementation of this visualization is in *pathAnalysis.x64_Project.exe* under a button entitled *calculateSurfaceDivergence*.

The approach consists of 3 steps: (i) assign surface face index to all points in network, (ii) compute divergence of flow in pial vessels that cross that face, and (iii) assign a ratio of outletFlowOfFace to LossOnFace to that face. These three steps are explained in more detail in Figure 7.99 - Figure 7.101. The result gives a value for lateral and diving flow for each surface mesh triangle. These two values can be compared through a ratio, through magnitude, through magnitude difference, and through magnitude product. Figure 7.98 and Figure 7.102 investigate many ways to visualize such vectors.

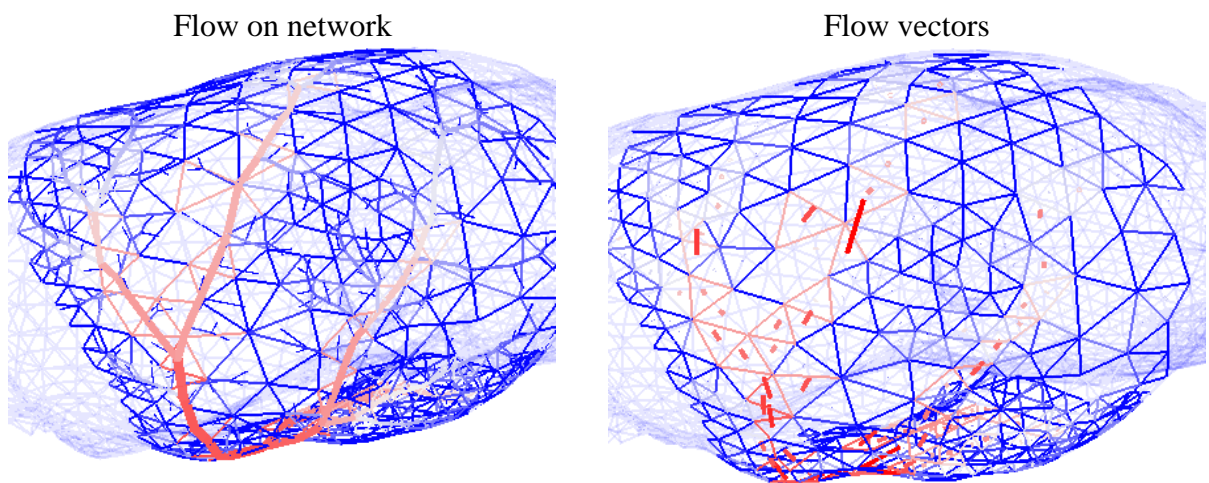


Figure 7.98. Visualization of surface flow vectors.

Left) traditional visualization with the network overlaid on the mesh surface for reference. Right) new visualization with vectors indicating direction and magnitude of general flow on the mesh surface.

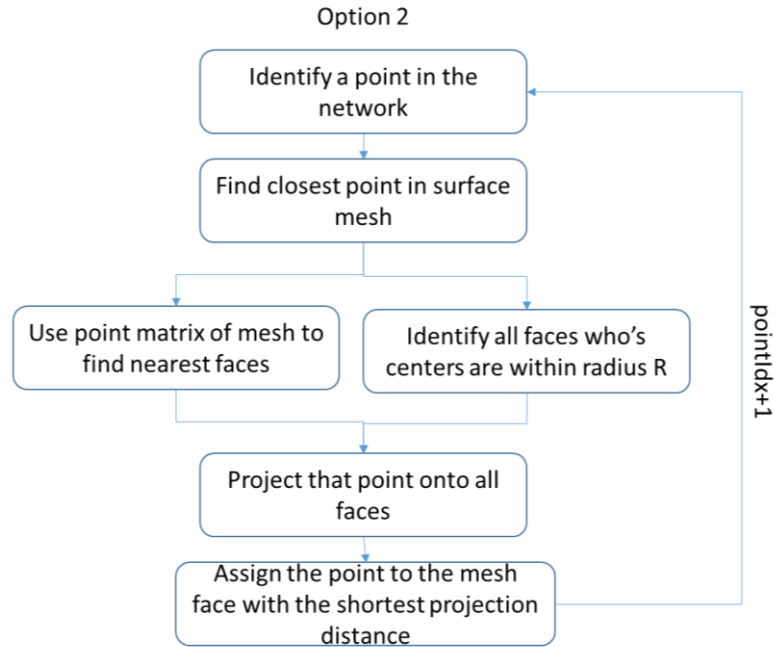


Figure 7.99. Step 1 of the surface divergence calculating algorithm.

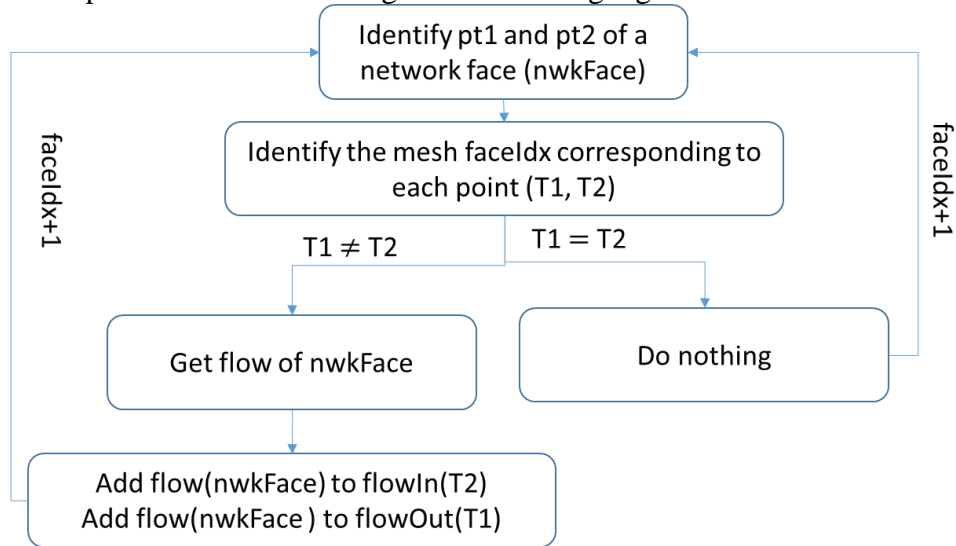


Figure 7.100. Step 2 of the surface divergence calculating algorithm.

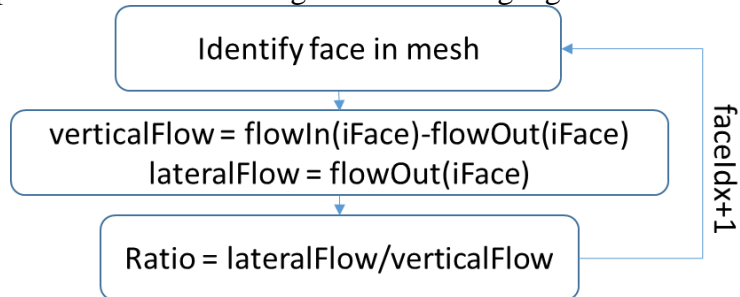


Figure 7.101. Step 3 of the surface divergence calculating algorithm.

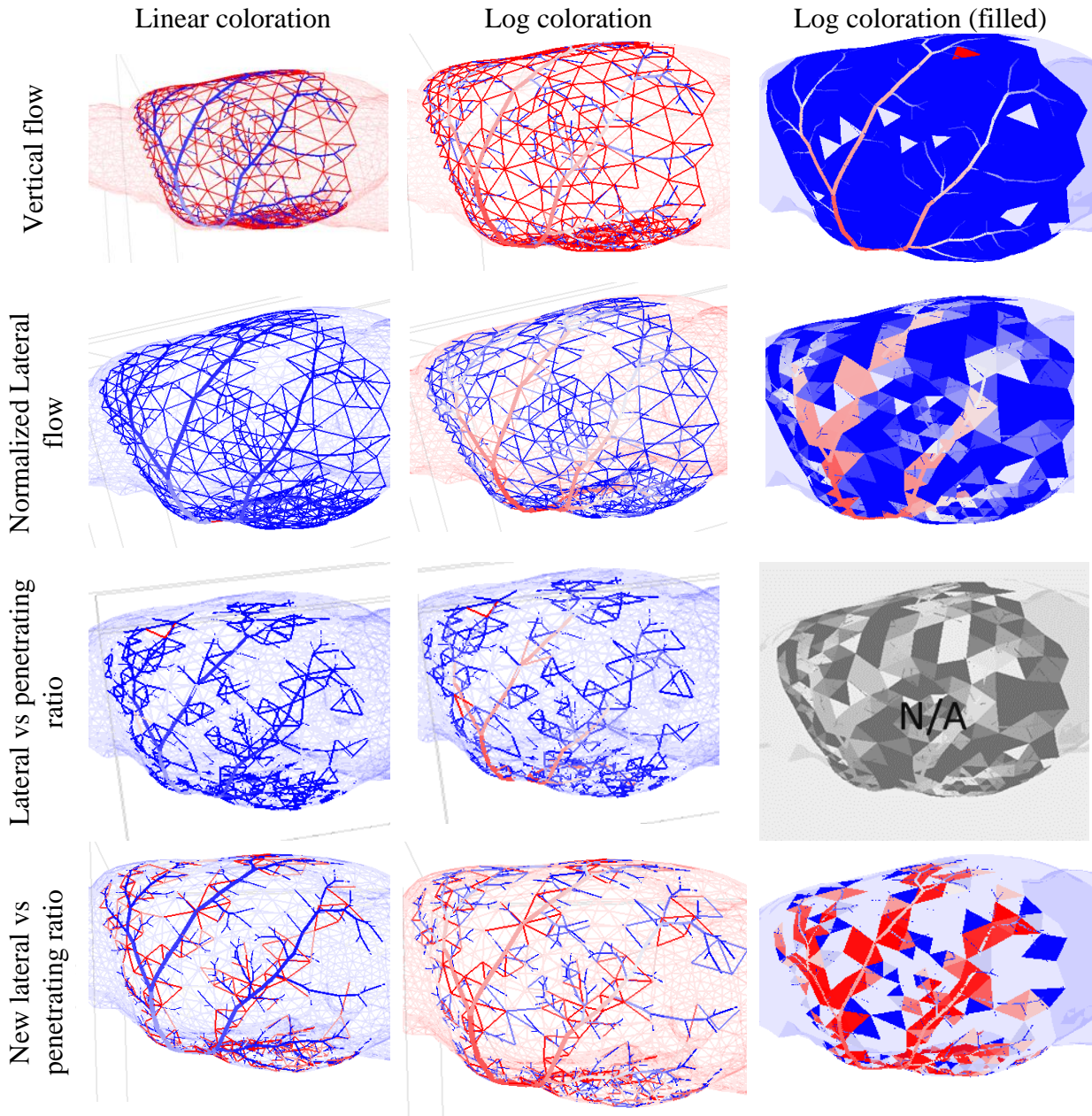
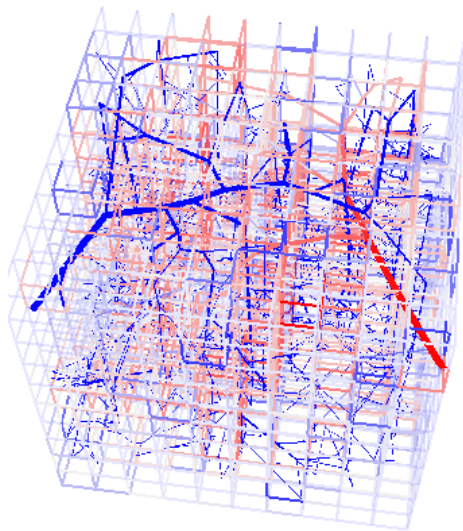


Figure 7.102. Coloration comparison between linear and log scaling for pial surface divergence. As can be seen, the linear coloration is heavy with blue values and only the highest indices of volumes (right edge) are red whereas the log scale has majority red values and only the first indices (lower left corner) are blue. Note the flow is the same in every terminal in the network, so any changes in vertical flow are representations of the number of terminals per surface triangle (more evident in the filled than the non-filled display).

7.13.3.2 Volume flow divergence visualization

The volume divergence is the same as the surface divergence, where the flow into the voxel and out of the voxel are used to see how much mass transfer occurs into the tissue in that volume. This can also be a good method for visualizing the network solving stability, as it is a visualization of the error at each junction in a network simulation encased in an artificial mesh.

Tissue: Flow divergence (0 - 4.3×10^{-5} $\mu\text{L/s}$)
Vasc: Flow (-1×10^6 – 1×10^6 $\mu\text{L/s}$)
Log coloration



Tissue: oxygen mass transfer (0-340 mMol/s)
Vasc: oxygen (0.29-1000 mMol)
Log coloration

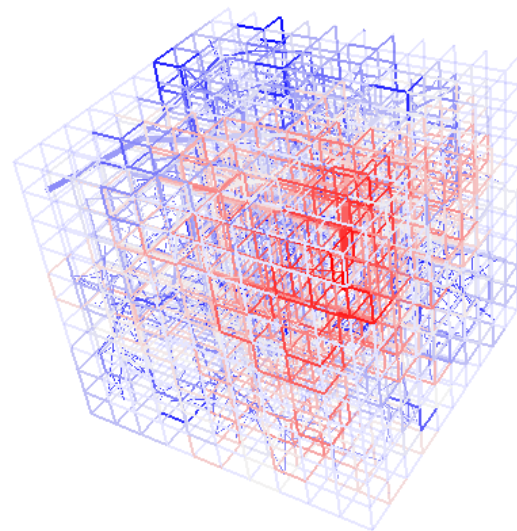


Figure 7.103. Visualization of the volume divergence implemented over a network and a 10x10x10 cartesian mesh.

Left) mass conservation of flow in the network (there is no mass transfer) is only visualized using numerical error as the mass transfer is uniformly small throughout the mesh. This is a nice way to visualize the vascular network solvability. Right) The mass transfer from vasculature to tissue exhibits a higher flux at the inlet than the outlet.

7.13.3.3 Displaying lognormal network trends

When plotting data to show trends (for example, plotting flow vs length) is a common practice in the experimental field. In such cases the frequent assumption is that the collection of data

represents a sparse sampling of the total data set. Due to the lack of data, it is important for the user to assess what type of model best fits the data (Gaussian, uniform, etc.). The most common type of assumed distribution is a Gaussian distribution, which assumes the range of values, and thus the probability density function, has a peak centered at a mean value and the trailing arms are described by the standard deviation. When a much larger group of data is available, as is common in computational practices where an entire domain is modeled simultaneously, the distribution does not need to be assumed as it may be fully determined. In the case of a large number of values and a known distribution, the case of reporting a simple mean and standard deviation may not be the most informative method for expressing the knowledge within data.

Calculating a mean and standard deviation/variance from a dataset is a commonly used method for interpreting a large quantity of data in an easy to read, intuitive and compact visualization as investigated in a case study below (Section 7.13.3.5).

In a dataset where the probability density function skews heavily in one direction (a long tail such as seen in a lognormal distribution) or on a bounded problem (flow cannot be negative, for instance), the mean and standard deviation can be heavily skewed by infrequent and significant outliers (such as pial vessels when analyzing the microcirculatory network). For these types of data, the standard deviation subtracted from the mean can give negative numbers, which is physiologically meaningless. In these cases, it is more practical to use a boxplot for the following reasons; (i) a boxplot focuses on the bulk of the data as opposed to the entire dataset, (ii) a boxplot gives more statistically calculated values about the data (instead of only the mean and standard deviation), and (iii) a boxplot does not ever give values outside the simulation range (such as negative flow or hematocrit >1).

A boxplot reports the median (with a single horizontal line), the interquartile range (IQR, 25%-75% of the array) and the values inside of 3 scaled median absolute deviations (MAD) as described in Section 7.13.3.6. This kind of an analysis removes skewing due to infrequent, outlier data points. In exchange, this type of analysis focuses on the region in which the large majority (>80%) of data occurs. An example of the distribution of flow in a microcirculatory dataset is given in Section 7.13.3.4.

7.13.3.4 Case Study 1: Flow in a Microcirculatory KF dataset

In the case of microcirculatory networks, pial vessels and large arterioles and large venules carry flow many orders of magnitude larger than the capillary vessels. Because the capillary bed accounts for >80% of vessels in the microcirculation, one approach is to account for the infrequent pial and penetrating vessels is to calculate the median, interquartile range and median absolute deviation values, which show the details of distribution in the region where the capillaries exist. An example of flow in a microvascular network is offered:

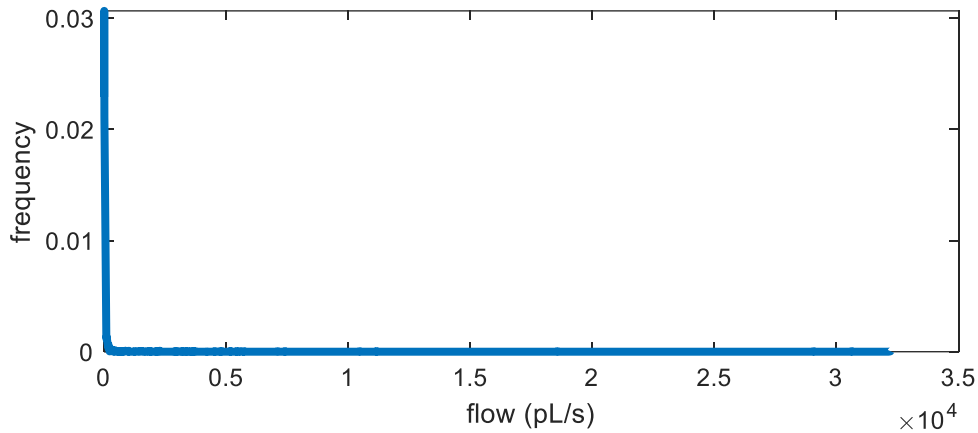


Figure 7.104: Probability density function of flow in Layer 1 after pial vessels have been removed.

Almost all of the data lies at the left of the curve, however the trailing tail of the distribution is very long and can skew the mean and standard deviation computations.

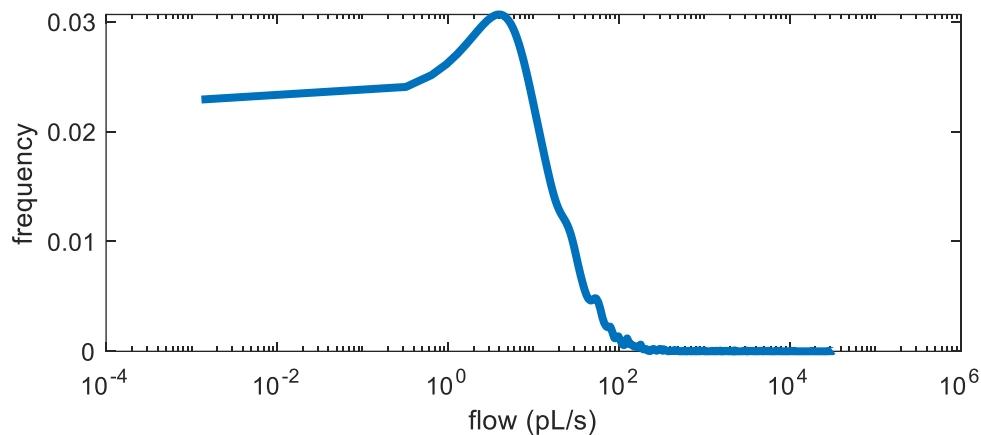


Figure 7.105: Probability density function of flow in Layer 1 after pial vessels have been removed with a logarithmic x-axis.

Almost all of the data lies under 100pL/s, however there are still values greater than 10,000pL/s in the network, which can skew the mean and standard deviation computations.

The mean and standard deviation of this dataset are have also been annotated on the same figure in Figure 7.106. Labeling the median, IQR, and 3·MAD values (values from a boxplot diagram) are a better representation of the data contained within the PDF in Figure 7.107.

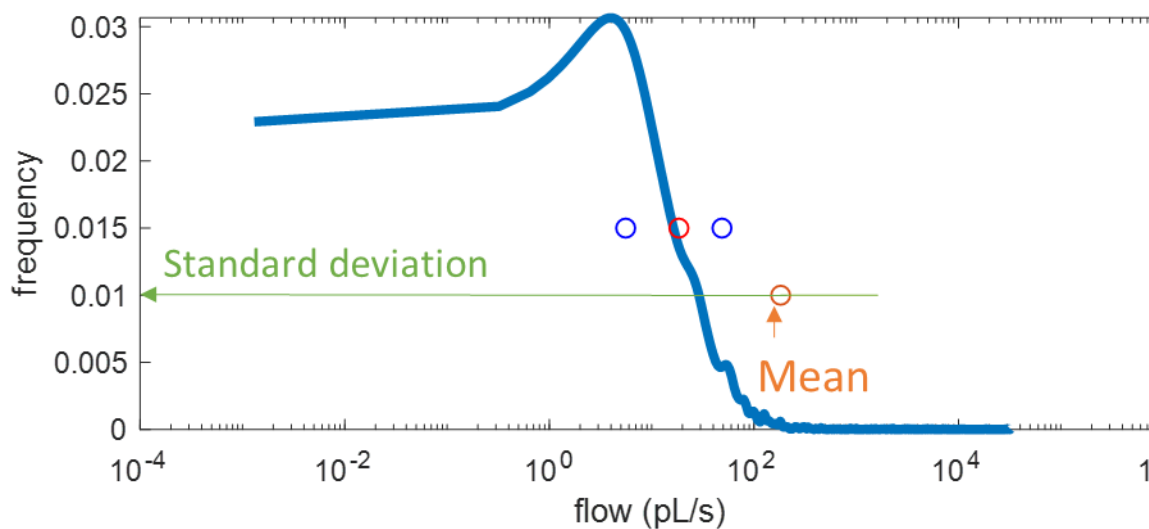


Figure 7.106: Annotated PDF of flow in Layer 1 of the microcirculatory bed. The values for mean (orange circle) and standard deviation (green lines) have been identified. The mean value lies above the majority of data and the standard deviation is very large.

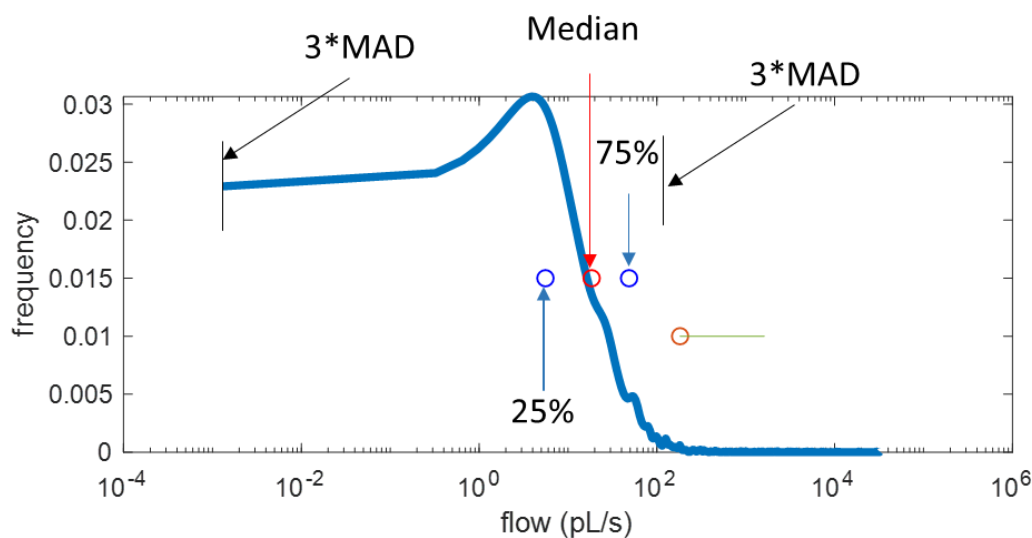


Figure 7.107: Annotated PDF of flow in Layer 1 of the microcirculatory bed. The values for median (red circle), 25-75 percentile (blue circles), and 3·MAD (black vertical lines) have been identified. These statistics represent a boxplot of data. Note, these statistics display more properties of the region of the PDF where the majority of the data lie.

In conclusion, in cases where the probability density function (PDF) has infrequent data points that significantly impact the standard deviation and the mean, a boxplot is the preferred visualization paradigm. This is because it expresses more detailed information about the highest density region of the PDF. In other words, if a PDF is highly concentrated in one region (>80% in that region) and the mean lies outside that region because of significantly deviating outlying values, a boxplot is preferred over a mean/variance plot.

7.13.3.5 Case study 2: calculation of mean and standard deviation

The mean value of a data vector is simply the average of the entire dataset:

$$mean = \mu = \frac{1}{N} \sum_{i=1}^N data_i \quad (7.203)$$

And the calculation of the standard deviation is:

$$std = S = \sigma = \sqrt{\frac{1}{N-1} \sum_{i=1}^N |data_i - \mu|^2} \quad (7.204)$$

Where the standard deviation can be thought of as the average divergence from the mean value. This can be used to describe a PDF sample distribution in many kinds of distributions such as a Gaussian, Binomial, Poisson, etc. A graphical interpretation of these values with a Gaussian distribution is offered:

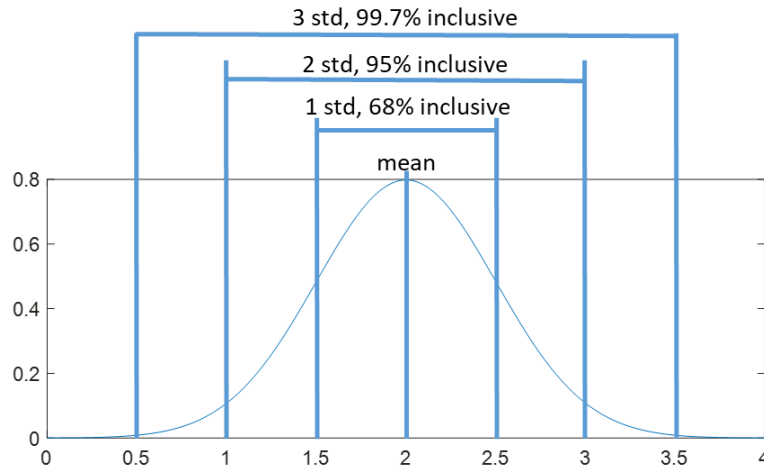


Figure 7.108: The graphical interpretation of standard deviation on a PDF with a normal distribution.

The normal distribution has a mean of 2 and standard deviation of 0.5.

7.13.3.6 Case study3: calculation of median, interquartile range and median absolute deviations

A boxplot reports the median (with a single horizontal line, usually colored red), the interquartile range (IQR, 25%-75% of the array, usually denoted by a blue box) and the values inside of 3 scaled median absolute deviations (MAD) as described in the case study below (frequently identified by black lines). These can be interpreted with the following diagram:

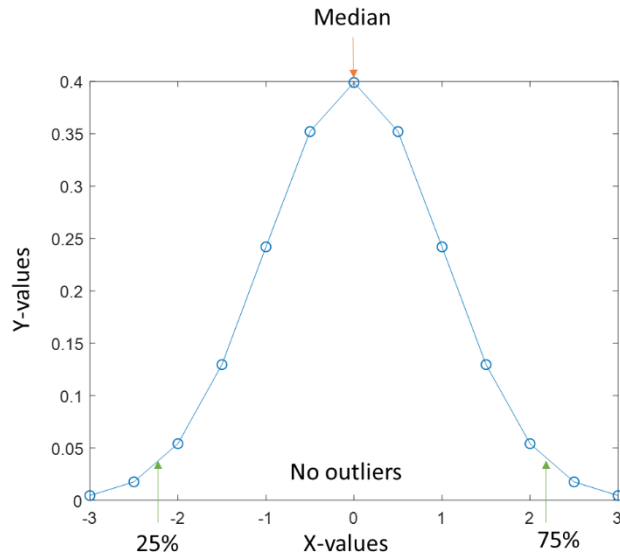


Figure 7.109: Identification of the median, 25% and 75% location on a Gaussian distribution. These numbers are used for the calculations below. Note, there are not outliers in this case study.

Where the calculation of the median, IQR and MAD are given in an example below. Step 1, sort the array of values from low to high:

(y-values)		(y-values)	
Index	An Array:	Index	A Sorted Array:
1	0.004432	1	0.004432
2	0.017528	2	0.004432
3	0.053991	3	0.017528
4	0.129518	4	0.017528
5	0.241971	5	0.053991
6	0.352065	6	0.053991
7	0.398942	7	0.129518
8	0.352065	8	0.129518
9	0.241971	9	0.241971
10	0.129518	10	0.241971
11	0.053991	11	0.352065
12	0.017528	12	0.352065
13	0.004432	13	0.398942

Sort Low
to High

Figure 7.110: An example of an array before and after sorting.

Step 2, identify the median, 25th percentile and 75th percentile. The median is the number in the middle of the array, the 25th percentile is indexed halfway between the middle and the 1st index

of the array and the 75th percentile is indexed halfway between the median and the highest index of the array:

(y-values)		A Sorted Array:	
Index			
		1	0.004432
		2	0.004432
25%	→	3	0.017528
		4	0.017528
		5	0.053991
		6	0.053991
Median	→	7	0.129518
		8	0.129518
		9	0.241971
75%	→	10	0.241971
		11	0.352065
		12	0.352065
		13	0.398942

Figure 7.111: A sorted array with annotations for the median and interquartile range (25% - 75%).

Step 3, calculate the norm deviation of each element from the median

(y-values)			(y-values)	
Index	A Sorted Array:		Index	MAD Array
1	0.004432	- Median (0.129518) =	1	0.125086
2	0.004432		2	0.125086
3	0.017528		3	0.111989
4	0.017528		4	0.111989
5	0.053991		5	0.075527
6	0.053991		6	0.075527
7	0.129518		7	0
8	0.129518		8	0
9	0.241971		9	0.112453
10	0.241971		10	0.112453
11	0.352065		11	0.222548
12	0.352065		12	0.222548
13	0.398942		13	0.269425

Figure 7.112: An example of a sorted array before and after subtracting the median value from the every element of the vector.

The resulting vector is the MAD array.

Step 4, sort the median array from low to high and find the median of this array. This new value is your MAD value.

(y-values) Index	MAD Array		(y-values) Index	Sorted MAD array
1	0.125086		1	0
2	0.125086		2	0
3	0.111989		3	0.075527
4	0.111989		4	0.075527
5	0.075527		5	0.111989
6	0.075527		6	0.111989
7	0	Sort Low to High →	7	0.112453 ← Median
8	0		8	0.112453
9	0.112453		9	0.125086
10	0.112453		10	0.125086
11	0.222548		11	0.222548
12	0.222548		12	0.222548
13	0.269425		13	0.269425

Figure 7.113: An example of finding the MAD value of a MAD array (see Figure 7.112 for more details).

The process sorts the MAD array and finds the median of this new array. The value representing this median is the MAD value.

Step 5, identify outliers as all values with a MAD value in the MAD value array larger than $3 \cdot (\text{MAD value})$ (in our case, $3 \cdot 0.112453 = 0.3374$):

(y-values) Index	MAD Array
1	0.125086
2	0.125086
3	0.111989
4	0.111989
5	0.075527
6	0.075527
7	0
8	0
9	0.112453
10	0.112453
11	0.222548
12	0.222548
13	0.269425

Figure 7.114: An example of a MAD array compared to the outlier value, $3 \cdot (\text{MAD value})$.

Any values with magnitude larger than the outlier value are deemed *outliers* and are indicated with stars on a boxplot diagram.

In this example, there are no outliers as no MAD array values are larger than 0.3374, so there would be no outliers.

7.14 Appendix N: Discretization schemes

7.14.1 Translating between analytic and discretized form in 1D

The analytic representation of the diffusion-reaction problem in steady state is presented in Equation (7.205) is in analytic form. On finite grids (or meshes), the equations must be discretized and applied to the elements of the grid.

$$-\frac{\partial}{\partial x} \left(-D \frac{\partial c}{\partial x} \right) = -R_0 \quad (7.205)$$

To understand how to apply the transport equation from analytics to a discretized system, a decision must be made to either enforce a finite element method (FEM) or finite volume method (FVM). The FEM applies Equation (7.205) directly for each element, making it easier to implement, but the FEM suffers from discontinuities at non-identical interfaces (interfaces between two tissues for instance) at low mesh density. This occurs because the balance equations approaching the interface from one side do not match the balance equations as they approach from the other side. Physically, this can be interpreted from the balance equation. Enforcing Equation (7.205) enforces the acceleration of the state across any given volume is 0 or equal to the reaction rate (in the steady-state case). This criteria does not enforce equivalent flux between adjacent elements (i.e. it does not use the same flux vector and balance these vectors). To overcome these discontinuities, a large number of grid points is required (high density mesh).

The FVM works on the integral form of the base equation. Instead of solving for the acceleration at each node, the divergence is calculated as the sum of fluxes through the volume. This enforces the flux out of one element is arrives at the neighboring element which guarantees

mass is conserved and reduces the occurrence of discontinuities at lower mesh densities. Because concentration must be conserved in these predictions, FVM is the better choice for discretization methods. To implement an FVM method, Equation (7.205) must be integrated as follows:

$$0 = \int_L^R -\frac{\partial}{\partial x} \left(-D \frac{\partial c}{\partial x} \right) dx - \int_L^R R_0 dx \quad (7.206)$$

Where R is right and L is left branch. When evaluated numerically:

$$0 = \int_L^R -\frac{\partial}{\partial x} \left(D \frac{c_i - c_{i+1}}{x} \right) dx - \int_L^R R_0 dx \quad (7.207)$$

Note that the second part of the fundamental theorem of calculus (also known as the Newton-Leibniz axiom) gives:

If

$$f(x)' = F(x)$$

Then:

$$\int_L^R F(x) dx = f(R) - f(L) \quad (7.208)$$

Giving:

$$\int_L^R f'(x) dx = f(R) - f(L)$$

Which is gives the integral form of the balance equation as:

$$D \frac{c_{i-1} - 2c_i + c_{i+1}}{dx} = R_0 dx \quad (7.209)$$

7.14.1.1 Graphical example

An example mesh has been created in 1D and is shown in Figure 7.115. Applying Equation (7.208) to Equation (7.207) gives Equation (7.210).



Figure 7.115. Graphical representation of the discretized 1D domain on which the diffusion-reaction problem will be formulated. Note, Δx is uniform.

$$0 = - \left(D \frac{c_i - c_{i+1}}{x} \Big|_R - D \frac{c_i - c_{i+1}}{x} \Big|_L \right) - R_0 x|_R - R_0 x|_L$$

$$0 = D \frac{c_i - c_{i+1}}{\Delta x} \Big|_L - D \frac{c_i - c_{i+1}}{\Delta x} \Big|_R - R_0 \Delta x \quad (7.210)$$

Where $\Delta x = x$ is the length of distance between adjacent nodes. Evaluated at node 4 gives:

$$\begin{aligned}
D \frac{c_i - c_{i+1}}{x} \Big|_L &= D \frac{c_3 - c_4}{\Delta x} & D \frac{c_i - c_{i+1}}{\Delta x} \Big|_R &= D \frac{c_4 - c_5}{x} \\
0 &= D \frac{c_3 - c_4}{\Delta x} - D \frac{c_4 - c_5}{\Delta x} - R_0 \Delta x \\
0 &= D \frac{c_3 - 2c_4 + c_5}{\Delta x} - R_0 \Delta x
\end{aligned} \tag{7.211}$$

Note, Equation (7.211) is equivalent to:

$$0 = D \frac{c_3 - 2c_4 + c_5}{\Delta x} - R_0 \tag{7.212}$$

For a 1st order reaction can be written as:

$$\begin{aligned}
0 &= D \frac{c_3 - 2c_4 + c_5}{\Delta x} - k_1 c_4 \Delta x \\
0 &= \frac{D}{\Delta x} c_3 - \left(\frac{2D}{\Delta x} + k_1 \Delta x \right) c_4 + \frac{D}{\Delta x} c_5
\end{aligned} \tag{7.213}$$

Note, the diffusivity here has units of m²/s and the reaction is of units 1/s, which results in an overall flux rates of mol/m²/s. This can be interpreted as an area-averaged flux. What this means, is that the fluxes assume that the other 2 dimensions are homogenous, well-mixed and uniform throughout the domain of x_{min} to x_{max}. In order to account for the 3 dimensions of the material property D, and rate of reaction k₁, it is important to multiply the flux equations by a user-defined

choice of cross sectional area. Because the example uses a uniform diameter, this value has been divided out of the entire equation.

7.14.2 Two dimensions

A more transparent problem formulation to is the 2D formulation. Because 2 dimensions is not physiologically relevant nor is it a significant problem reduction, it is not frequently used. As such, the derivation will be brief. In the case of 2D, the integration over the volume will take place in 2 dimensions instead of 1, making Equation (7.206) expand to:

$$0 = \int_B^T \int_L^R -\frac{\partial}{\partial x} \left(-D \frac{\partial c}{\partial x} \right) - \frac{\partial}{\partial y} \left(D \frac{\partial c}{\partial y} \right) dx dy - \int_B^T \int_L^R R_0 dx dy \quad (7.214)$$

Where B is bottom and T is top. Evaluating the integral gives:

$$\begin{aligned} \int_B^T \int_L^R R_0 dx dy &= \int_B^T \int_L^R -\frac{\partial}{\partial x} \left(-D \frac{\partial c}{\partial x} \right) - \frac{\partial}{\partial y} \left(D \frac{\partial c}{\partial y} \right) dx dy \\ \int_B^T R_0 \Delta x dy &= \int_B^T D \frac{c_i - c_{i+1}}{\Delta x} \Big|_L - D \frac{c_i - c_{i+1}}{\Delta x} \Big|_R - \frac{\partial}{\partial y} \left(D \frac{\partial c}{\partial y} \right) \Delta x dy \\ R_0 \Delta x \Delta y &= \left(D \frac{c_i - c_{i+1}}{\Delta x} \Big|_L - D \frac{c_i - c_{i+1}}{\Delta x} \Big|_R \right) \Delta y + \left(D \frac{c_i - c_{i+1}}{\Delta y} \Big|_B - D \frac{c_i - c_{i+1}}{\Delta y} \Big|_T \right) \Delta x \end{aligned} \quad (7.215)$$

7.14.2.1 Graphical example of numerical discretization

An example mesh has been created in 2D and visualized in Figure 7.116.

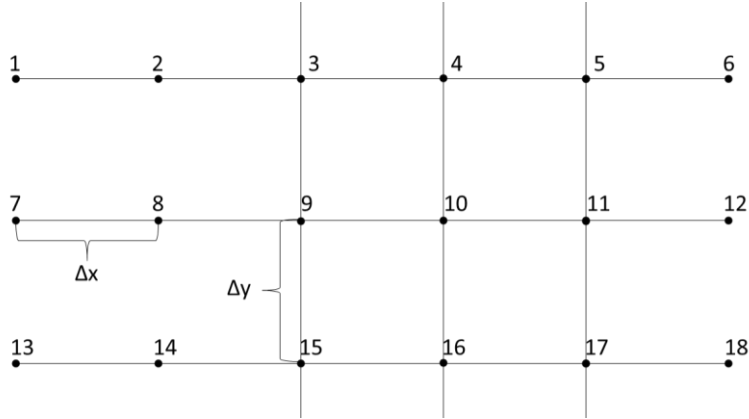


Figure 7.116. 2-dimensional grid system on which the transport equations will be discretized.

Evaluation at node 10 gives:

$$R_0 \Delta x \Delta y = D \Delta y \frac{c_9 - 2c_{10} + c_{11}}{\Delta x} + D \Delta x \frac{c_{16} - 2c_{10} + c_4}{\Delta y} \quad (7.216)$$

Where the grid is still evenly spaced and the values of flux are in moles/m/s. Note, the third dimension is considered homogenous and uniform throughout the domain, so it has been divided out of the equation.

7.14.3 Three dimensions

Likewise in 3D, Equation (7.206) can be expanded to:

$$\int_S^D \int_B^T \int_L^R R_0 \, dx \, dy \, dz = \int_S^D \int_B^T \int_L^R -\frac{\partial}{\partial x} \left(-D \frac{\partial c}{\partial x} \right) - \frac{\partial}{\partial y} \left(D \frac{\partial c}{\partial y} \right) - \frac{\partial}{\partial z} \left(D \frac{\partial c}{\partial z} \right) \, dx \, dy \, dz \quad (7.217)$$

Where S is shallow and D is deep into the page. Evaluating the integral gives:

$$\begin{aligned}
\int_S^D \int_B^T \int_L^R R_0 \, dx \, dy \, dz &= \int_S^D \int_B^T \int_L^R -\frac{\partial}{\partial x} \left(-D \frac{\partial c}{\partial x} \right) - \frac{\partial}{\partial y} \left(D \frac{\partial c}{\partial y} \right) - \frac{\partial}{\partial z} \left(D \frac{\partial c}{\partial z} \right) \, dx \, dy \, dz \\
&= \int_B^T \int_L^R D \frac{c_i - c_{i+1}}{\Delta x} \Big|_L - D \frac{c_i - c_{i+1}}{\Delta x} \Big|_R - \frac{\partial}{\partial y} \left(D \frac{\partial c}{\partial y} \right) \Delta x - \frac{\partial}{\partial z} \left(D \frac{\partial c}{\partial z} \right) \Delta x \, dy \, dz \\
&= \int_B^T \left(D \frac{c_i - c_{i+1}}{\Delta x} \Big|_L - D \frac{c_i - c_{i+1}}{\Delta x} \Big|_R \right) \Delta y + \left(D \frac{c_i - c_{i+1}}{\Delta y} \Big|_B - D \frac{c_i - c_{i+1}}{\Delta y} \Big|_T \right) \Delta x \\
&\quad - \frac{\partial}{\partial z} \left(D \frac{\partial c}{\partial z} \right) \Delta x \Delta y \, dz \\
R_0 \Delta x \Delta y \Delta z &= \left(D \frac{c_i - c_{i+1}}{\Delta x} \Big|_L - D \frac{c_i - c_{i+1}}{\Delta x} \Big|_R \right) \Delta y \Delta z \\
&\quad + \left(D \frac{c_i - c_{i+1}}{\Delta y} \Big|_B - D \frac{c_i - c_{i+1}}{\Delta y} \Big|_T \right) \Delta x \Delta z \\
&\quad + \left(D \frac{c_i - c_{i+1}}{\Delta x} \Big|_S - D \frac{c_i - c_{i+1}}{\Delta x} \Big|_D \right) \Delta x \Delta y
\end{aligned} \tag{7.218}$$

This final form is the diffusivity in each dimension multiplied by the cross sectional area perpendicular to the diffusive dimension. Moreover, the reaction is multiplied by the mesh element volume. Note, this formulation assumes isotropic, uniform diffusivity.

7.14.3.1 Graphical example

An example mesh has been created in 3D and visualized in Figure 7.117.

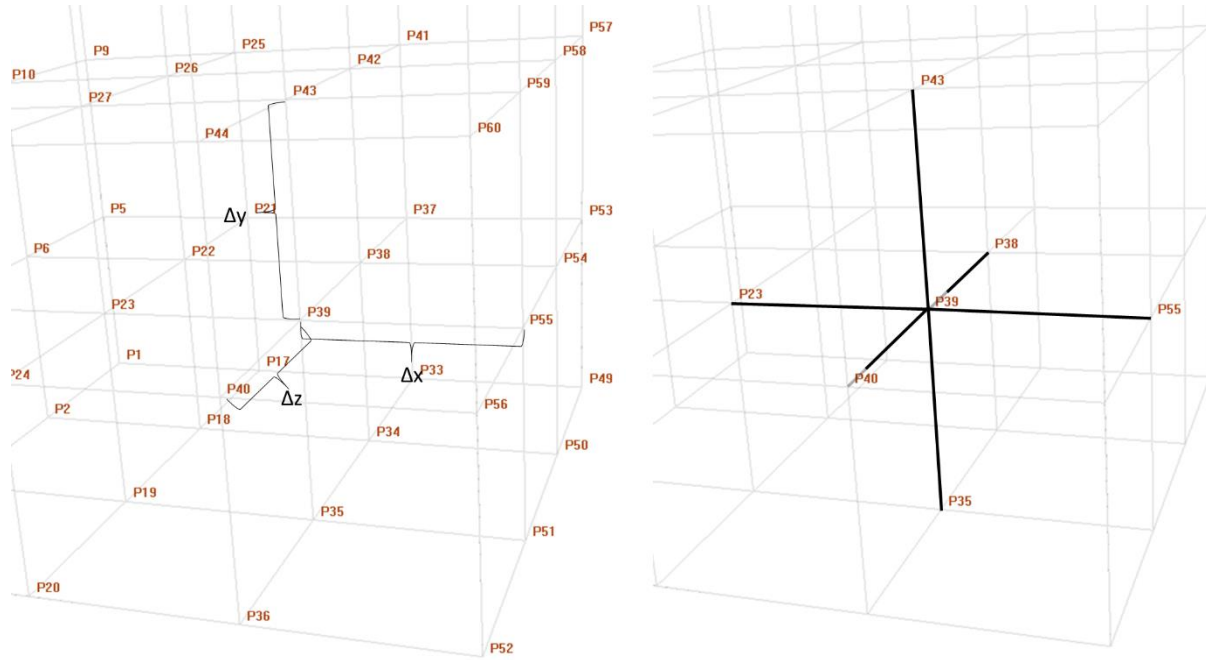


Figure 7.117. 2-dimensional grid system on which the transport equations will be discretized.

Evaluated at node 39 (assuming nodes are centers of hexahedral volumes) gives:

$$\begin{aligned}
 R_0 \Delta x \Delta y \Delta z = & \frac{\Delta y \Delta z}{\Delta x} (c_{23} - 2c_{39} + c_{55}) \\
 & + \frac{\Delta x \Delta z}{\Delta y} (c_{35} - 2c_{39} + c_{38}) \\
 & + \frac{\Delta x \Delta y}{\Delta z} (c_{40} - 2c_{39} + c_{38})
 \end{aligned} \tag{7.219}$$

Where the connected nodes to node 39 are 23 (L), 55 (R), 38 (D), 40 (S), 43 (T), and 35 (B).

Where the grid is still evenly spaced and the values of flux are in moles/s.

nVolumes	10	25	50	100	200	300	500	1000	2000	5000	10000
Solving time (s)	3.7e-3	6.6e-3	12.3e-3	45.1e-3	191e-3	491e-3	1.52	7.43	40.0	381	2452

7.14.4 Finite mesh discretization

The discretization of a mesh should not impact the solution vector. Discretization refers to the method for delineating the bounding volumes of a mesh. Two methods for discretizing a mesh are considered here: (i) the half-volume technique and (ii) the half distance spacing technique. Both methods are visualized in Figure 7.118.

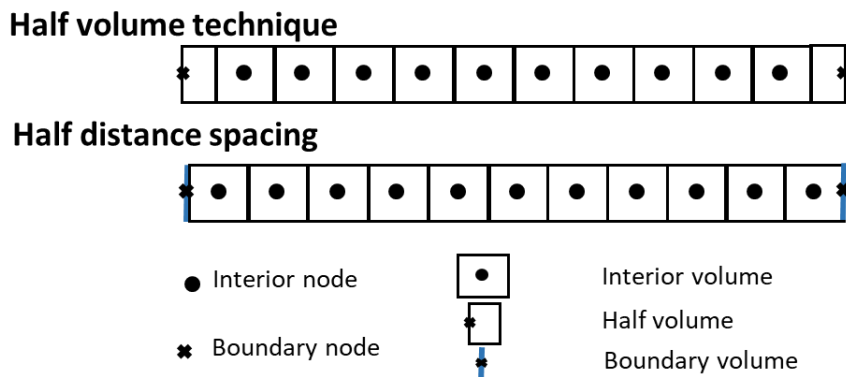


Figure 7.118. Schematic of domain on which the current 1D implementation is being solved.

The former method is convenient for deriving the volume centers, as they are simply the ratio of the domain length to the number of volumes in the domain. This method suffers from inconsistent boundary assignment, where the mandatory replacement of a boundary equation at the edge volumes changes the domain volume. An example of this is the case of a diffusion problem with a reaction. When the mesh is discretized at two different mesh densities, the size of the boundary volumes changes, thus changing the domain volume available for reactions. This is an inherent mesh instability of the method.

The second method creates a boundary volume at the exterior of the domain, ensuring the interior volume extent remains unchanged at any discretization density. These boundary volumes only have a $\frac{1}{2}$ distance from the neighboring volume, which can make implementation less readable. The boundary volumes, however, have no effect on the reaction domain, so this method does not have inherent mesh instabilities.

In all cases it is recommended to use model (ii), the half distance spacing method. This method offers stable convergence and ensures full domain extent for simulations.

7.15 Appendix O: Boundary conditions

In all simulations, an important modeling decision outside of the geometry (3D structure), relevant transport phenomena (diffusion, rxn, etc.), and properties being conserved is the BCs (BCs). These conditions are chosen by the modeler to enforce what is believed to happen at the modeling boundary. Unlike batch reactors, however, traditional BCs (such as Dirichlet temperature) cannot be assumed at an interface in biological systems. This adds modeling assumptions to the solution space that have a significant impact on the solution; meaning the solution significantly changes based on the BC choice, even if all choices are equally reasonable. One way to circumvent this problem is to remove the BCs as far from the solution space as possible, which can be accomplished by modeling an entire organ as opposed to only a portion. Regardless, at the edge of the organ a BC must still be set, however the BC should have a smaller impact (less boundary effects) and the choice should be more obvious.

In a paper by Lorthois, a very laborious derivation and proof was required in an attempt to claim the BCs had no effect on the outcome of her simulation. Work by our own group includes simulations of large spatial extents [21,22,231–233] and we have employed vascular generation [21,234] in an effort to distance the simulation results from the effects at the artificial boundaries, both for vasculature and tissue. This section will review some of the different BC choices available and some insight into when to apply them to a biological simulation.

It is important to note that the enforcement of a BC involves replacing a conservation equation with said BC. This can have significant impacts on the solution space, for instance, if the BC is set on a volume inside the solution space that encompasses a reaction. In this case, as the mesh is refined, the volume of space devoid of reactions will shrink, thus fundamentally changing the

outcome of the simulation. This is one of the reason that most BCs exist at flat faces on the edge of a 3D simulation mesh (see Section 7.14.4 on discretization schema).

7.15.1 Dirichlet and Neumann

Two of the simplest BCs are Dirichlet (fixed value) and Neumann (fixed flow). A Dirichlet BC is implemented by setting the value for an unknown parameter in the solution space to a specific value. This can be applied to an interior or exterior volume, but as described above, an exterior volume is preferred when possible. A Dirichlet BC is written in Equation (7.220).

A fixed flow/flux (known as Neumann) is simply setting the value of the flow to a known value as in Equation (7.221). This can also be applied to an interior or exterior volume, but preferably to an exterior volume in the case of reactions as described above.

$$P_i = \text{const} \quad (7.220)$$

$$f_i = \text{const} \quad (7.221)$$

7.15.2 Proportional flow

A proportional flow BC is an effective way of enforcing one flow between two volumes in a mesh is directly proportional to the flow between another pair of volumes as in Equation (7.222). An application of this type of BC is where a vascular tree modeler believes the velocity, cross sectional area, or vessel surface area of a terminal segment is indicative of its fractional flow of the mother branch when no other data is available. This application is investigated in more detail in Section 7.15.6 where measured data from mother branches of vascular trees are used to identify

BCs and flows throughout the network. Unfortunately, investigation revealed these BCs create a singular matrix in some cases in the forward simulation as mass conservation and Hagen Poiseuille flow already account for the fractional division of flow at all branches in the network. When assuming the flow out of a terminal segment is proportional to the cross sectional area of that terminal, the fractional flow from the mother branch is proportional to the fractional area of that segment from all terminal segments. This can be expressed mathematically in equation (7.223).

$$f_i = \frac{1}{\beta} f_j \quad (7.222)$$

$$f_i = f_{tot} \frac{A_i}{\sum_{j=1}^N A_j} \quad (7.223)$$

Note, if all BCs are enforced using fractional flow the system becomes fully determined and singular because the fractional flow BC implies the sum of all outlet flows is the inlet flow, which is redundant of the conservation equations.

7.15.3 Periodic

Another BC type covered in this document is a periodic BC. This type of BC is the equivalent of putting a duplicate of the mesh object next to the object, and connecting the two adjacent sides as neighbors. This is implemented by using an artificial face between opposing terminal directions (south connects to north, east to west, and top to bottom) through a *characteristic face*. This characteristic face is defined as the “average” face (cross sectional area and length). This face is then used for conservation equations as if it were a normal interior volume.

7.15.4 Constant Gain

To obtain a time-dependent result without having to make assumptions assume at the inlets, constant gain BCs could be used, however they are outside the scope of this work. Constant gain BCs are effective in dynamic BCs, where the attenuation (or gain) of pressure between the inlets and the outlets for the nominal (steady-state values) can be used as constant throughout the frequencies.

7.15.5 Implementation

This section will investigate the implementation of BCs necessary for linear flow, biphasic blood flow, and oxygen simulations in the brain.

Convection. The hematocrit (the volume fraction of RBCs in blood) has the physical interpretation as a convected medium, meaning that it is moving primarily by the force of the bulk flow (plasma). This bulk flow follows a diffusion mechanism (pressure-driven flow) which requires two BCs (explained next).

Note, convection only requires a single explicit BC. This is because in differential form of the conservation equation, convection is a single order differential equation. In short, the integration would require only one BC to specify all integration constants. In order to solve for n_{Points} unknown states, there must be n_{Points} number of equations (otherwise you will have an overdetermined or underdetermined system and have no solution). This means a choice must be made at the outlet in order to solve the balance envelope around that point. The most logical choice is a fully developed flow profile at the outlet (meaning the flow into the outlet node is the same as exiting the outlet node, or $\vec{\nabla} \cdot \mathbf{f} = 0$). The inlet node must be chosen manually with a Dirichlet BC.

Diffusion. For pressure-driven flow, two BCs are necessary (2nd order differential equation in conservation form). The BCs can be any combination of pressure and flow, as long as every independent set of vessels (vessel group that is not connected to another vessel group) has at least one pressure BC applied to it. This means that if there are two vascular structures being simulated simultaneously (like two trees that do not join in the middle), they will each require at least one pressure BC to ensure the solving matrix is non-singular. This can be physically interpreted as the absolute pressure level which satisfies the necessary pressure drop throughout the network. Without this pressure BC calibration, there are an infinite number of absolute pressures of the inlet and outlet that cause the same pressure drop (i.e., a 15 mmHg inlet and 10 mmHg pressure drop can also give the same flow as a 25 mmHg inlet and 20 mmHg outlet pressure drop, although the exact pressure differs).

Choosing a flux BC is advantageous when such information is readily available, such as measurements of blood flow in the microcirculatory surface vessels. The values of flow vary wildly among the literature, specifically depending on environmental factors such as the level of sedation of the animal [235–237]. A more reliably measured value is the systolic and diastolic blood pressure in the arteries of animals and the venous pressure. Because these values are more widely accepted, they are used in the current work as Dirichlet BCs.

7.15.5.1 Choosing bulk flow BCs

Regardless of biphasic or single-phasic blood flow, BCs must be chosen for the blood flow simulation. In the case of biphasic blood flow across the microcirculation, the BC for pressure at the inlet was derived from mean arterial pressure during systole in the mouse, ~120 mmHg [238]. The diastolic pressure in mouse was found to be ~100 mmHg by the same source, so the range of

inlet can be reasonably chosen between 100 and 120 mmHg. 120 mmHg was chosen for our large, dense, tortuous microcirculatory networks.

7.15.5.2 Choosing hematocrit BCs

Hematocrit is treated like a solute in the 2015 implementation of the KPSM [21], adheres to a single-order differential equation (convection) and requires only 1 BC, as discussed above. The convected medium needs to only know the inlet concentration (or hematocrit level). A reasonable inlet hematocrit level is taken as the systemic value (ranging in the literature between 0.35 and 0.45). With previous models of hematocrit splitting, the tendency in big-network simulations was to use the lowest reasonable value for systemic hematocrit to avoid the hematocrit values leaving the physiological region ($Hct > 1$) [21,22,39,40]. Some groups even required manipulation of the solution vector to keep inside physiological range [40,44]. In experience, the value of systemic hematocrit does not greatly affect the overall flow rate (even though the viscosity is hematocrit-dependent) which is not an actual BC but rather another conservation equation. In the present work, the Dirichlet value of 0.35 is chosen [21,22]. The outlet terminals use a reflective BC (flow into the node is the same as exiting the node through the terminal).

7.15.5.3 Choosing oxygen BCs

In an oxygen simulation, any number of boundary conditions can be considered reasonable for the tissue. Through investigation, however, it was deemed that the two conditions that give the most reasonable profiles are Dirichlet or periodic BCs (see Section 7.28.5) although a fixed flux could be substituted in lieu of this. The next most reasonable is the insulated BC. At the vascular

inlet, a Dirichlet value of systemic arterial oxygen tension is imposed as it gives the most reasonable choice for a steady-state simulation.

7.15.6 BCs from empirical measurements and optimization

Subject-specific anatomical reconstructions of vascular structures from medical images [68,231] allow virtual experimentation (simulations) of structural changes in the human cerebrovasculature at an unprecedented level. Unfortunately, due to limited imaging resolution, these structures are incomplete and lack the microvasculature closure that would normally connect the pial arterial trees to the large venous structures. Due to this missing structure, the pial vessels are endowed with a large number of open-ended segments, known as terminals. Fortunately, some patients who undergo neurovascular imaging also undergo blood flow measurement at the same time. The reconstructions accompanying data imaged using a patented NOVA system [239] can benefit from the empirical information to assist in boundary condition assignment. An approach to use this information to delineate boundary conditions would be to find the minimum least-square error between a blood flow simulation that adheres to conservation balances to the measurements of a few large vessels. Such an approach will be introduced in this section.

Note, all dimensions of matrices and vectors will be reported using the notation that Np = number of points, Nip = number of internal points, Nep = number of terminal points, and Nf = number of faces.

The forward simulation statement. Linear flow can be predicted using the first principle models (Hagen Poiseuille and mass conservation) as seen in equations (7.224)-(7.226).

$$Af = Z_1 P \quad (7.224)$$

$$Z_2 f = 0 \quad (7.225)$$

$$Z_3 p = \bar{p} \quad (7.226)$$

The minimization problem statement. A least-square optimization problem can be constructed to minimize the difference between measured flows and simulated flows while forcing the simulated flows to adhere to the linear flow simulation. This is expressed in Equation (7.227).

$$\begin{aligned} z(p, f) &= \min_{f, p} \|\hat{f} - f\| \\ \text{s. t.} \\ Af &= Z_1 P \\ Z_2 f &= 0 \\ Z_3 p &= \bar{p} \end{aligned} \quad (7.227)$$

Where $z \in \mathbb{R}^{N_f \times N_f}$, $f \in \mathbb{R}^{N_f \times 1}$, $Z_2 \in \mathbb{R}^{N_f \times N_f}$, $Z_1 \in \mathbb{R}^{N_f \times N_p}$, $A \in \mathbb{R}^{N_f \times N_f}$, $Z_3 \in \mathbb{R}^{N_{ep} \times N_p}$.

7.15.6.1 Mathematical formulation of the optimization equation

To find the optimal point of this convex and positive-definite system, the solution lies at a point where the derivative has a value of 0 (gradient in all dimensions is 0). Because this least-square problem is of 2nd order, it has only one position where this occurs, so any solution found is by definition the global solution. So the first step is to take the partial derivative of L with respect to all variables (p, λ , and μ) and find where these differentials are all 0. Validations of matrix derivatives can be seen in Section 7.24.

Lagrangian. The Lagrangian method is used to solve the constrained optimization problem described in equation (7.227). To solve the constrained optimization problem, the Lagrangian function is defined in equation (7.228). The Lagrangian incorporates the artificial variables, λ and μ , to account for the set of node mass balances, $h(p, f)$, and the set of (incomplete) BCs, $g(p)$. The value of these variables are the “shadow price” and are indicative of the importance of the constraint on the solution vector. In other words, if the value of one of these variables is high, it means that relaxing that specific constraint will greatly affect the solution vector. This can also be explained as a way to encode the sensitivity of the objective function to the constraint level. The optimality condition for the Lagrangian in Equation (7.229) leads to a linear set of equations that allow the convenient determination of the unknown vectors p, f, λ, μ as will be explored in the next sections.

$$L(p, f, \lambda, \mu) = z(p, f) + \lambda^T h(p, f) + \mu^T g(p) \quad (7.228)$$

$$\vec{\nabla} L(p, f, \lambda, \mu) = 0 \quad (7.229)$$

Objective function. The objective function, Equation (7.230), sums the difference between given flows (e.g. NOVA measured results, \hat{f}) and the predicted flows, f . It is convenient to define the square weight matrix, $W \in \mathbb{R}^{N_f \times N_f}$, of binary entries to account for available measurements as in equation (7.231). W is a diagonal matrix which holds a value of 1 in the event a corresponding vessel index has been measured and a value of 0 otherwise. Without the weighting matrix, the optimization would assume that there is a measurement value of 0 flow for all faces where there is no measurement. This weighting matrix can also be modified beyond the values of 1 and 0 to

encode the confidence in each measurement, allowing certain measurements to have more weight than others in the optimization but this is not explored here.

$$z(P, f) = [W(\hat{f} - f)]^T [W(\hat{f} - f)] \quad (7.230)$$

$$W = \begin{cases} 1 & \text{where } \hat{f} \text{ is measured} \\ 0 & \text{otherwise} \end{cases} \quad (7.231)$$

To reduce problem size, flows can be eliminated from the unknown vector, f , and replaced with equations of pressure with the help of Equation (7.227) in the form $f = A^{-1}Z_1p$. The new objective function is quadratic in unknown pressures. The solution can then be obtained through the optimality condition as in Equation (7.229).

By substituting Equation (7.227) into Equation (7.230), the constraint equations become equation (7.232). The vector \bar{p} is the vector corresponding to the BCs. $p \in \Re^{Np \times 1}$, $f \in \Re^{Nf \times 1}$, $Z_3 \in \Re^{Nep \times Np}$

$$\begin{aligned} z(p, f) &= \min_{f, p} \|\hat{f} - f\| \\ \text{s. t.} \\ Z_2 A^{-1} Z_1 p &= 0 \\ Z_3 p &= \bar{p} \end{aligned} \quad (7.232)$$

The objective function can now be formulated and reduced as in Equation (7.233). $S \in \Re^{Np \times Np}$, $t \in \Re^{Np \times 1}$, $Z_4 = Z_2 A^{-1} Z_1$.

$$\begin{aligned}
z(p) &= [(W\hat{f} - WZ_4p)]^T [(W\hat{f} - WZ_4p)] \\
&= [(\hat{f}^T W^T - W^T (Z_4p)^T)] [(W\hat{f} - WZ_4p)] \\
&= p^T Z_4^T W^T W Z_4 p - p^T Z_4^T W^T W \hat{f} - \hat{f}^T W^T W Z_4 p + \hat{f}^T W^T W \hat{f} \quad (7.233) \\
&= p^T (WZ_4)^T W Z_4 p - 2(W\hat{f})^T W Z_4 p + (W\hat{f})^T W \hat{f} \\
z(p) &= p^T S p - 2 t^T p + c
\end{aligned}$$

Obtaining the derivative of the Lagrangian. The Lagrangian function can now be written with respect to only 3 variables; pressure (p) and the shadow prices (λ, μ) as in Equation (7.234). As discussed above, the optimal solution of this problem exists where all partial derivatives are a value of 0. These partial derivatives are expressed below.

$$L(p, f, \lambda, \mu) = z(p, f) + \lambda^T h(p, f) + \mu^T g(p) \quad (7.234)$$

Partial derivatives of objective function ($z(p)$). The gradient w.r.t. pressure ($\vec{\nabla}_z(p)$) can be obtained after introducing a symmetric, square matrix, $S = (WZ_4)^T (WZ_4)$ and vector $t = (W\hat{f})^T W Z_4$ for simplification resulting in Equation (7.235).

$$\begin{aligned}
\vec{\nabla}_p z &= 2Sp - 2t \\
\vec{\nabla}_\lambda z &= 0
\end{aligned} \quad (7.235)$$

$$\vec{\nabla}_\mu z = 0$$

Partial derivatives of node mass balances. The set of node mass balances is a set of linear equations with dependent variable pressure $[h(p)]$ as in Equation (7.236). The desired partial derivatives then become Equation (7.237). The introduction of $Z_5 = Z_2 A^{-1} Z_1$ is added here for simplification. $\lambda \in \Re^{N_{ip} \times 1}$, $Z_5 \in \Re^{N_{ip} \times N_p}$

$$h(p) = Z_2 A^{-1} Z_1 p = Z_5 p \quad (7.236)$$

$$\vec{\nabla}_p [\lambda^T h(p)] = Z_5^T \lambda$$

$$\vec{\nabla}_\lambda [\lambda^T h(p)] = Z_5 p \quad (7.237)$$

$$\vec{\nabla}_\mu [\lambda^T h(p)] = 0$$

Partial derivatives of BCs. In practice, the complete BC list is not specified in the estimation problem, instead only a partial list is used. For example, all inlet and outlet pressures cannot be given or else the constraints would be fully determined and there would be no degrees of freedom for optimization. Instead, conditions for the many open branches of the network must be provided (specifically $n_{Terminals} - n_{Measurements}$ is the minimum number of BCs necessary).

In the simplest form, a single inlet BC set with a Dirichlet pressure, the boundary condition matrix Z_3 would be a row vector with a 1 at the index of the inlet node and \bar{p} would be a vector of length 1 and a value that corresponds to the Dirichlet pressure. More information can be found in Section 7.11.1. $Z_3 \in \Re^{N_{ep} \times N_p}$

An example of a single Dirichlet BC with 4 nodes with the first node pressure is set to 10 mmHg can be seen in Equation (7.238). An example of a partial Neumann (flow) BC of the same system with flow 1 (between node 1 and 2) set to 10 ml/min can be seen in Equation (7.239). In this case, α_1 is the resistivity of arc 1 between node 1 and node 2. $\mu \in \Re^{Nep \times 1}$

$$\begin{aligned} g(p) &= Z_3 p - \bar{p} \\ g(p) &= [1 \ 0 \ 0 \ 0]p - 10 \end{aligned} \tag{7.238}$$

$$\begin{aligned} g(p) &= Z_3 p - \bar{p} \\ g(p) &= [-1/\alpha_1 \ 1/\alpha_1 \ 0 \ 0]p - 10 \end{aligned} \tag{7.239}$$

The required partial derivatives then become equation (7.240).

$$\begin{aligned} \bar{\nabla}_p[\mu^T g(p)] Z_3^T \mu \\ \bar{\nabla}_\lambda[\mu^T g(p)] &= 0 \\ \bar{\nabla}_\mu[\mu^T g(p)] &= Z_3 p - \bar{p} \end{aligned} \tag{7.240}$$

Suggested BCs. In an effort to maximize information obtained from measurements during the optimization solving, the assignment of n-6 terminal pressures in order to fulfill the equation/unknown requirement balance (assuming 6 measurements of the network). This is where the partial flow BC described in Section 7.15.2 is implemented.

Final solution. By assembling the partial derivatives in Equations (7.235), (7.237), and (7.240), the result is a linear set of algebraic equations (Equations (7.241)-(7.242)).

$$\vec{\nabla}_p L = -2t + 2Sp + Z_4^T \lambda + Z_5^T \mu = 0$$

$$\vec{\nabla}_\lambda L = Z_5 p = 0 \quad (7.241)$$

$$\vec{\nabla}_\mu L = Z_3 p - \bar{p} = 0$$

$$\begin{bmatrix} 2t \\ 0 \\ \bar{p} \end{bmatrix} = \begin{bmatrix} 2S & Z_5^T & Z_3^T \\ Z_5 & 0 & 0 \\ Z_3 & 0 & 0 \end{bmatrix} \begin{bmatrix} p \\ \lambda \\ \mu \end{bmatrix} \quad (7.242)$$

7.15.6.2 Using the Fourier Transform to produce time-dependent BCs

Blood flow is naturally a dynamic system, oscillating with the heartbeat. In order to capture the transient nature, a time-dependency is necessary to reconstruct the entire signal. Measurements from a NOVA report [239] can be extracted from multiple vessels in the human brain. These match patient-specific anatomical reconstructions as previously demonstrated [68,231]. With these flows, the optimization can be performed at each time point. It can be shown that these problems are separable and adjacent time points do not influence each other, allowing the possibility for discontinuities in the time-dependent simulation.

To avoid these discontinuities, the discrete Fourier series can be computed of the original measurements and the optimization can be performed in the Fourier domain at each frequency. These problems can also be separated and solved individually for each frequency as described in Section 7.15.13. Once computed, the Fourier coefficients can be reconstructed back into the time domain as continuous, analytic functions.

The discrete Fourier series. The discrete Fourier transform is a coordinate transformation between the time domain and the frequency domain in effort to decompose an oscillatory signal

into the constitutive waveforms that, when added together, create the final measured waveform. This can be accomplished with simple algebraic or linear algebraic techniques is a method readily available for filtering the entire signal for noise and for up-sampling the original data. An in-depth investigation of different indexing schemes, transform implementations, comparison of deconstructing and reconstructing in real and complex arithmetic, and case studies are offered in Section 7.15.7. The mathematics of the discrete Fourier series (DFT) and the inverse discrete Fourier series (IDFT) are offered in Equations (7.243)-(7.244). These equations are written in the indexing scheme of Quarteroni [240] due to straightforward expressions of DFT and IDFT, even though it is only capable of deconstructing data with an even number of samples. For a comparison to Matlab and Trefethen [241] indexing can be seen in Section 7.17.4 and in an internal lab report [242].

$$\tilde{f}_k = \frac{1}{N} \sum_{j=0}^{N-1} f(x_j) W_N^{\left(k-\frac{N}{2}\right)j} \quad (7.243)$$

$$f(x_j) = \sum_{k=0}^{N-1} \tilde{f}_k W_N^{-\left(k-\frac{N}{2}\right)j} \quad (7.244)$$

Where:

$$h = \frac{2\pi}{N}, \quad x_j = jh, \quad W_N = e^{-i\frac{2\pi}{N}}$$

Problem formulation in the Fourier space. If this advanced technique for signal processing could be used in conjunction with the current optimization problem, it would allow the measured blood flow signals to produce a continuous function of blood flow throughout the entire network.

In order to execute this, the same optimization problem described above can also be formulated in terms of a discrete Fourier series.

Equation (7.245) expresses an example of the generalized discrete Fourier series of a signal (Equations (7.243)-(7.244)) in terms of a relevant parameter to this problem. This example replaces the measured signal, $f(x_j)$ with measured flows ($\hat{f}(t_j)$), and the Fourier coefficient for that flow, \tilde{f}_k , with a new variable $\tilde{\tilde{f}}_k$.

$$\tilde{\tilde{f}}_k = \frac{1}{N} \sum_{j=0}^{N-1} \hat{f}(t_j) V(t_j) \quad (7.245)$$

$$f(t_j) = \sum_{k=0}^{N-1} \tilde{\tilde{f}}_k V(k) \quad (7.246)$$

Where:

$$V = e^{i\frac{2\pi}{N}(k-\frac{N}{2})t}$$

Using this example, the original data (one value for every time point) is converted to a set of measurements in each frequency. Likewise, all other variables can be reformulated reformulating in the Fourier domain where $\hat{f}(t), f(t), \bar{p}(t)$, and $p(t)$ become $\tilde{\tilde{f}}_k, \tilde{f}_k, \tilde{\tilde{p}}_k$, and \tilde{p}_k . In other words, instead of solving the optimization point across many time points, it will optimize across many frequencies. This is then plugged back into the original optimization problem, where the sum of error is minimized as expressed in Equation (7.247).

$$z(p, f, k) = \min_{f, p} \left\| \sum_{k=0}^{N-1} \widetilde{\widehat{f}}_k V_k - \sum_{k=0}^{N-1} \widetilde{f}_k V_k \right\|$$

s.t.

$$A \sum_{k=0}^{N-1} \widetilde{f}_k V_k - Z_1 \sum_{k=0}^{N-1} \widetilde{p}_k V_k = 0$$

$$Z_2 \sum_{k=0}^{N-1} \widetilde{f}_k V_k = 0$$

$$Z_3 \sum_{k=0}^{N-1} \widetilde{p}_k V_k - \sum_{k=0}^{N-1} \widetilde{\widehat{p}}_k V_k = 0$$

(7.247)

Where: $v_k = e^{i\frac{2\pi}{N}(k-\frac{N}{2})t}$

$\widehat{f}(t) = \text{flow measurements}$ $f(t) = \text{flow}$ $p(t) = \text{pressure}$ $\overline{p}(t)$
 $= \text{pressure BC}$

And in the Fourier space (for k^{th} frequency):

$\widetilde{\widehat{f}}_k = \text{flow measurement coefficient}$ $\widetilde{f}_k = \text{flow coefficient}$

$\widetilde{\widehat{p}}_k = \text{pressure BC coefficient}$ $\widetilde{p}_k = \text{pressure coefficient}$

for k frequency in the fourier space

This formulation constitutes a separable problem where each frequency can be solved with an independent optimization problem as verified in Section 7.15.13. The problem can be further

simplified by removing the unknown flows from the problem space in lieu of pressure relationships as in Equation (7.248). Here, the finite Fourier series (FFS) is calculated using a finite (N) number of samples (referred to as the j^{th} sample). This formulation matches the original formulation, Equation (7.232), with the difference being the unknown vector of flows is replaced with a vector of Fourier coefficients (a_k) for each frequency and likewise the unknown vector of pressures is replaced with an unknown vector of coefficients (b_k).

$$\begin{aligned}
 z(p) = \min_{f,p} & \left\| \sum_{k=0}^{N-1} \widetilde{f}_k V_k - A^{-1} Z_1 \sum_{k=0}^{N-1} \widetilde{p}_k V_k \right\| \\
 \text{s.t.} & \\
 & A^{-1} Z_1 \sum_{k=0}^{N-1} \widetilde{p}_k V_k = 0 \\
 & Z_3 \sum_{k=0}^{N-1} \widetilde{p}_k V_k - \sum_{k=0}^{N-1} \widetilde{\widetilde{p}}_k V_k = 0
 \end{aligned} \tag{7.248}$$

The formulation for each frequency independently follows in equation (7.249).

$$\begin{aligned}
 z(p, k) = \min_{f,p} & \left\| \widetilde{f}_k - A^{-1} Z_1 \widetilde{p}_k \right\| \\
 \text{s.t.} & \\
 & A^{-1} Z_1 \widetilde{p}_k = 0 \\
 & Z_3 \widetilde{p}_k - \widetilde{\widetilde{p}}_k = 0
 \end{aligned} \tag{7.249}$$

Deriving for the optimal solution where all partial derivatives are a value of 0 is expressed in Equation (7.250).

$$\begin{bmatrix} 2t \\ 0 \\ \bar{p} \end{bmatrix} = \begin{bmatrix} 2S & Z_4^T & Z_3^T \\ Z_4 & 0 & 0 \\ Z_3 & 0 & 0 \end{bmatrix} \begin{bmatrix} \hat{p}_k \\ \lambda \\ \mu \end{bmatrix}$$

$$\text{Where: } Z_4 = Z_2 A^{-1} Z_1 \quad t_k = \left[W_1 \tilde{f}_k \right]^T W_1 A^{-1} Z_1 \quad (7.250)$$

$$W_1 = \begin{cases} 0 & \tilde{f}_k = 0 \\ 1 & \tilde{f}_k \neq 0 \end{cases}; \quad S = [W_1 A^{-1} Z_1]^T [W_1 A^{-1} Z_1]$$

This results in a set of Fourier coefficients, one for each pressure correlating to the k^{th} coefficient in the series. These coefficients represent the least error from the measured values and the simulated values and can now be used in the inverse Fourier transform (see Section 7.17 for more information on the discrete Fourier series) to interpolate the pressure waveform for all points for any time point.

Sampling the original data. In order to reconstruct time-lapse measurements of the human arterial tree, NOVA measurements were provided for the 9 largest segments in the arterial tree (RICA, LICA, BA, RMCA, LMCA, LPCA, RPCA, RACA, LACA) for some subjects. The NOVA report generates a graph of the time-lapsed blood flow values. In order to approximate these pictures, a hand-sampling method was employed (web plot digitizer). This results in non-evenly distributed sampling of the data due to the inability of the user to precisely select evenly distributed samples.

Two methods for sampling the data evenly were investigated, as even sampling is required for the discrete Fourier transform. The first method uses linear interpolation between adjacent data points. The second method relies on the knowledge of the oscillatory and continuous nature of the original signal and uses Matlab's Fourier fitting algorithm to resample the data after fitting a group of sines and cosines to the data using linear regression. It is recommended to evenly sample the data using Matlab's Fourier fit interpolation with 4 coefficients or to use linear interpolation, the use of both has rendered similar results (data not shown), and the Fourier fit was preferred for comparison with previous work [29]. Only one cycle of the heartbeat is used, which is identified as the time between two adjacent maximum peaks. The partial flow BCs will be used as described in Section 7.15.2. As an assumption, the inlets will be described by a known function of time, namely equation (7.251). This function can be sampled at time points commensurate with the measured data. Case studies are given in the next section.

$$P_1 = 100 + 20 \sin(\pi t) \ ; \ f_6 = 50 + 10 \sin(\pi t) \quad (7.251)$$

7.15.7 Case studies

This section delineates a few select case studies to exemplify the robustness of the algorithm.

A single tube. A Simple tube with 4 segments has been used for hand calculations and validations. This case study uses a single assigned pressure BC.

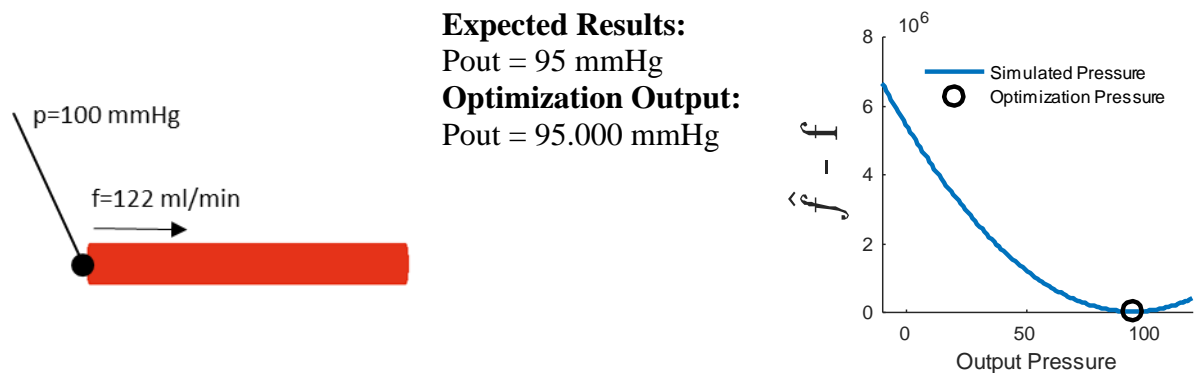


Figure 7.119. Case study 1. Left) Visualization of input BC and two measurements that do not hold to mass balance. Middle) The expected results match the optimized results, where the minimum error is achieved as the linear averaging between the two measurements. Right) The optimization space surrounding the terminal boundary condition shows a clear minimum which was correctly identified by the optimization algorithm, validating the algorithm.

In the second example, two flow measurements are used that do not agree. The optimized solution uses the value exactly halfway between the measurements.

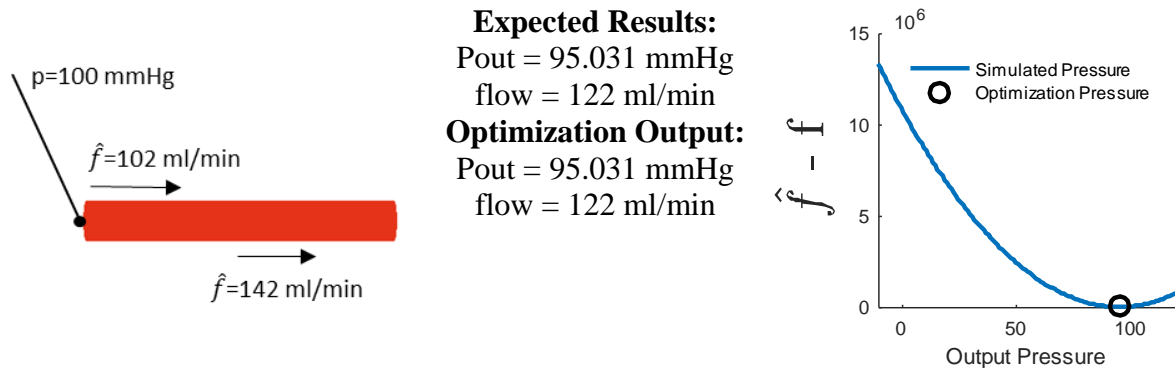


Figure 7.120. Case study 2.

Left) Visualization of input BC and two measurements that do not hold to mass balance. Middle) The expected results match the optimized results, where the minimum error is achieved as the linear averaging between the two measurements. Right) The optimization space surrounding the terminal boundary condition shows a clear minimum which was correctly identified by the optimization algorithm, validating the algorithm.

Simple tree. This example uses 3 BCs (1 inlet pressure and 2 outlet flow BCs given as proportional to the cross sectional area ratio) and has simulated noise in the measured flows (one flow is $\sim 10 \text{ ml/min}$ higher than the forward simulation with the same BCs). It is important to omit at least one BC so that the optimization algorithm has at least one degree of freedom.

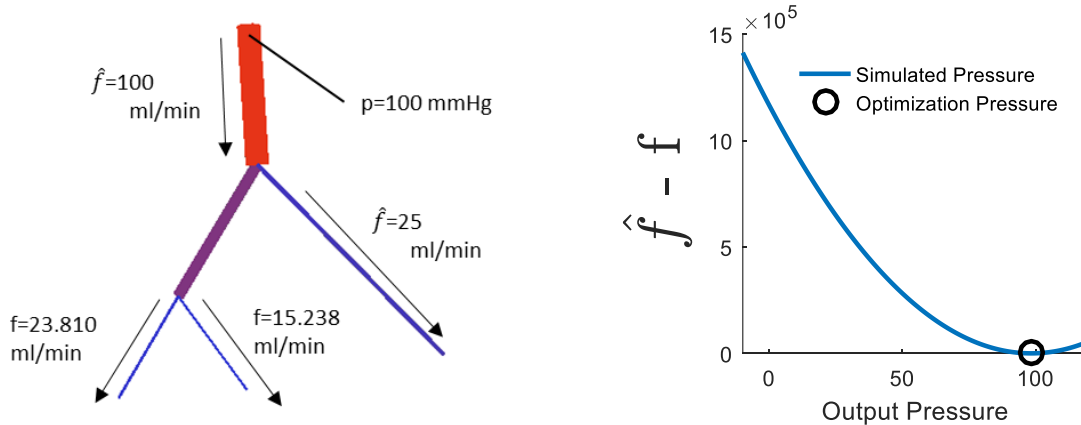


Figure 7.121. Visualization of case study 3.

Left) The network, BCs, and measurements used for this case study. Right) Error between measurement and simulated flow while varying the undefined BC. Note, the optimization pressure lies at the minimum of the solution space.

The predictions indicate the inlet flow will increase by 4.8810 mmHg above the measurement and decreased in the other measured branch (4.8810 mmHg below the measurement). This symmetric altering of the two branches to arrive at the optimal location halfway between the two measurements gives insight on how measurements are treated during optimization. Specifically, measurements are treated with equal weight and the optimal solution is midway between the values.

Due to the nature of optimization, all BCs can be enforced with partial flow regardless of matrix singularity yet this is not indicative of stable solvability but rather a property of optimization, which can optimize overdetermined systems by finding the minimum and optimize underdetermined systems by finding a line of solutions and returning the point on the vector with the shortest length. This method is inadvisable as the system is more stable when boundary conditions are removed as discussed above.

7.15.7.1 Intermediate network

For this case study, measured flows have been generated to be commensurate with simulated flows. For this case study, fractional outlet flows will be determined as a function of the major branch (numbered 1-6 in Figure 7.122). For validation purposes, the fractional flow will be determined from the actual fractional flow in the forward simulation, not from fractional area.

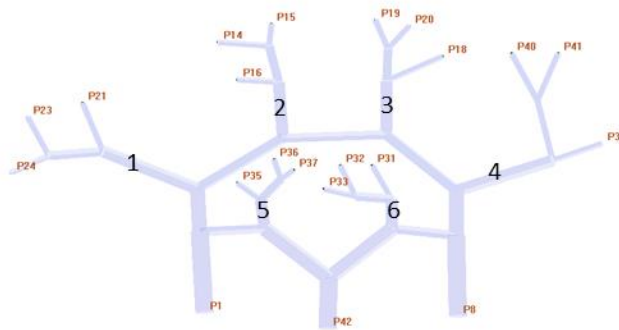


Figure 7.122. The network used for the present case study. The main branches (numbered 1-6) are used as root branches for each sub-tree in the network and are used in the partial-flow BC assignment. The terminals of the network are identified.

Table 7.34: Optimized values for terminal pressures and overall mass conservation		
inlet Pressures (mmHg)	outletPressures (mmHg)	DeviationFromMeasurements =
(1,1) 100.0000	(1,1) 79.9979	1.0e-03 *
(2,1) 100.0000	(2,1) 80.0001	
(3,1) 100.0000	(3,1) 80.0011	-0.1331
	(4,1) 80.0013	0.1371
	(5,1) 80.0002	0.1292
	(6,1) 79.9989	0.0000
	(7,1) 79.9998	0.1027
	(8,1) 79.9989	0.2977
	(9,1) 80.0015	-0.2134
	(10,1) 79.9985	-0.0000
	(11,1) 80.0004	-0.2002
	(12,1) 80.0015	-0.0363
	(13,1) 80.0007	0.2365
	(14,1) 79.9991	-0.2525
	(15,1) 80.0000	
	(16,1) 80.0012	
	(17,1) 88.2905	
	(18,1) 88.2911	

$$\text{inletFlows-outletFlows} = 4.4338\text{e-}12$$

All values for input (measured flows, Dirichlet pressures and Neumann flows) were accurate to the order of 10^{-3} . The error of the optimization method (error of 10^{-4}) is considered acceptable in this case.

7.15.7.2 Steady-state simple brain with circle of Willis

A simplified cerebrovascular network was constructed by hand and is depicted in Figure 7.123. This network utilizes 3 inlet BCs and nine real-world NOVA steady-state measurements to investigate the resulting output BC that best matches the data to the simulation. The Nova measurements are as follows: RMCA = 166, RICA = 168, BA = 181, RPCA = 70, LPCA= 89, LICA=312, LMCA=172, LACA=62, and RACA = 61.

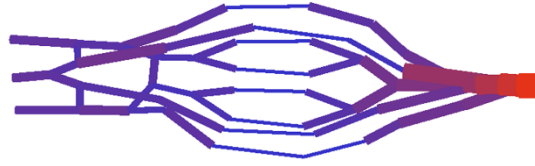


Figure 7.123. Simplified cerebrovascular network used for optimizing real-world data to simulation constraints.

Expected results are that the simulation will not match the nova results perfectly, but will trade off between the values (some results will be larger than measurements and others will be smaller). The predicted flows will be some complex averaging of the measured values that also adhere to mass balance.

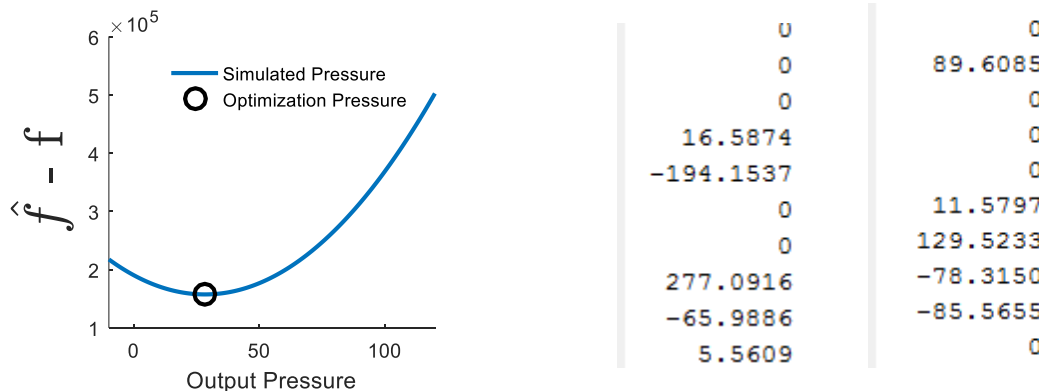


Figure 7.124. Optimization gives best BC to match measured flows.

Left) parametric study of outlet BC exemplifying that the optimized solution identified the pressure with the least overall error. Right) Difference between measured and predicted flows shows that some values were larger and others were smaller than the measurements, as expected.

7.15.7.3 Steady-state patient

This section will describe the methodology of the optimizing Nova results on a human cerebrovascular reconstruction. The following assumptions will be made; (i) the nova results will be sampled in time for one cardiac cycle, (ii) the time-averaged flow values will be used as the

measurements for the steady state simulation, and (iii) the sampled data from NOVA flows for the RICA, LICA, and BA will be normalized to having a nominal value of 100 mmHg and amplitude of ± 20 mmHg to be used as known pressure waveforms for inlet BCs. An investigation of the high frequency noise due to the Gibbs phenomenon of the Fourier series approximation will be investigated in Section 7.15.10. The steady-state optimization results can be reviewed in Figure 7.125. The results show a reasonable tradeoff between measured flows to fit the data to a simulation enforcing transport properties and mass conservation.

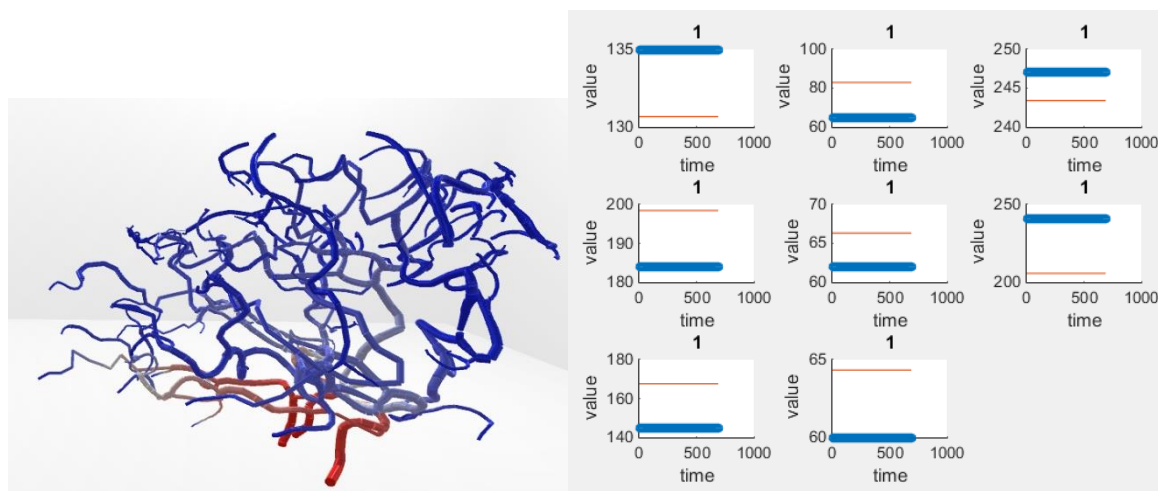


Figure 7.125. A steady state simulation using the nominal values from NOVA.

Left) The human cerebrovascular network. Right) The optimization trades off from the measurements (blue circles) and the simulation results (orange line) to find a good level of fit. Note, these offsets (between the blue circles and the orange lines) will be used to calibrate the dynamic measurements prior to optimizing in the Fourier domain.

7.15.8 Dynamic inversion case studies

7.15.8.1 Simple tube

The first case study considers a single tube with centerline and labeling drawn below. The given pressure BC waveforms can also be seen below. The measured flow is obtained directly

from a forward simulation with both BCs yet in the optimization step, the outlet BC is removed. These time points are commensurate with an analytical function ($\cos(2\pi/4 t)$ and $2\cos(2\pi/4 t)$). The result is a perfect recreation of the original outflow signal.

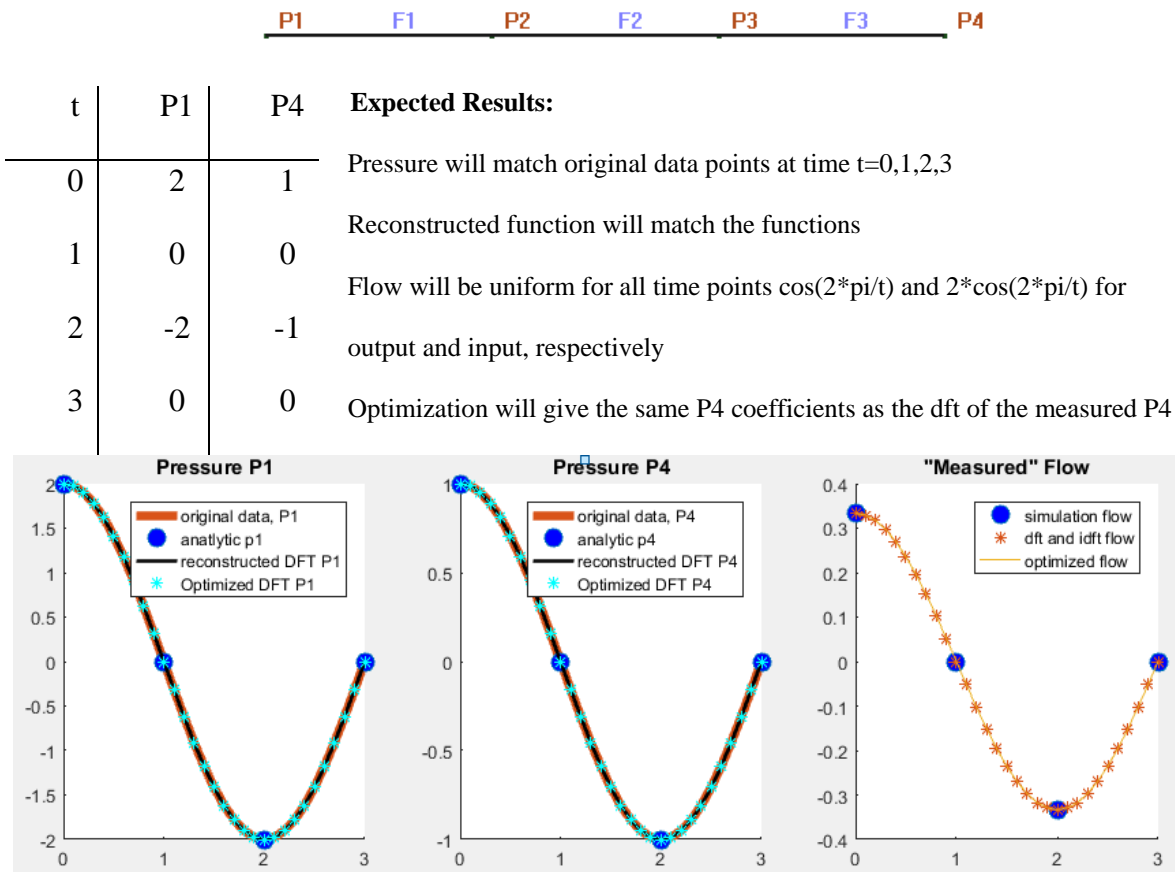


Figure 7.126. The dynamic inlet and outlet pressures and flow measurements. DFT/IDFT from simulation, DFT/IDFT from optimization (after deleting one BC) and the analytic solution from which the BCs were derived. The system is a simple tube with Dirichlet BCs. Here the inlet and outlet are in phase. Top) The 4-node structure being used. Middle) The BCs at inlet and outlet point at 4 time points and expected results. Bottom) The optimization was capable of reconstructing the correct Fourier coefficients to make a signal that matches the BCs and measurements at the 4 time points.

The next case study uses a measured flow to match boundary condition pressures that do not fit any analytic function. These time points are commensurate with an analytical function

($\cos(2\pi/4 t)$ and $\sin(2\pi/4 t)$). The forward simulation was performed to calculate the measured flows for face 1. The result is a perfect recreation of the outflow signal.

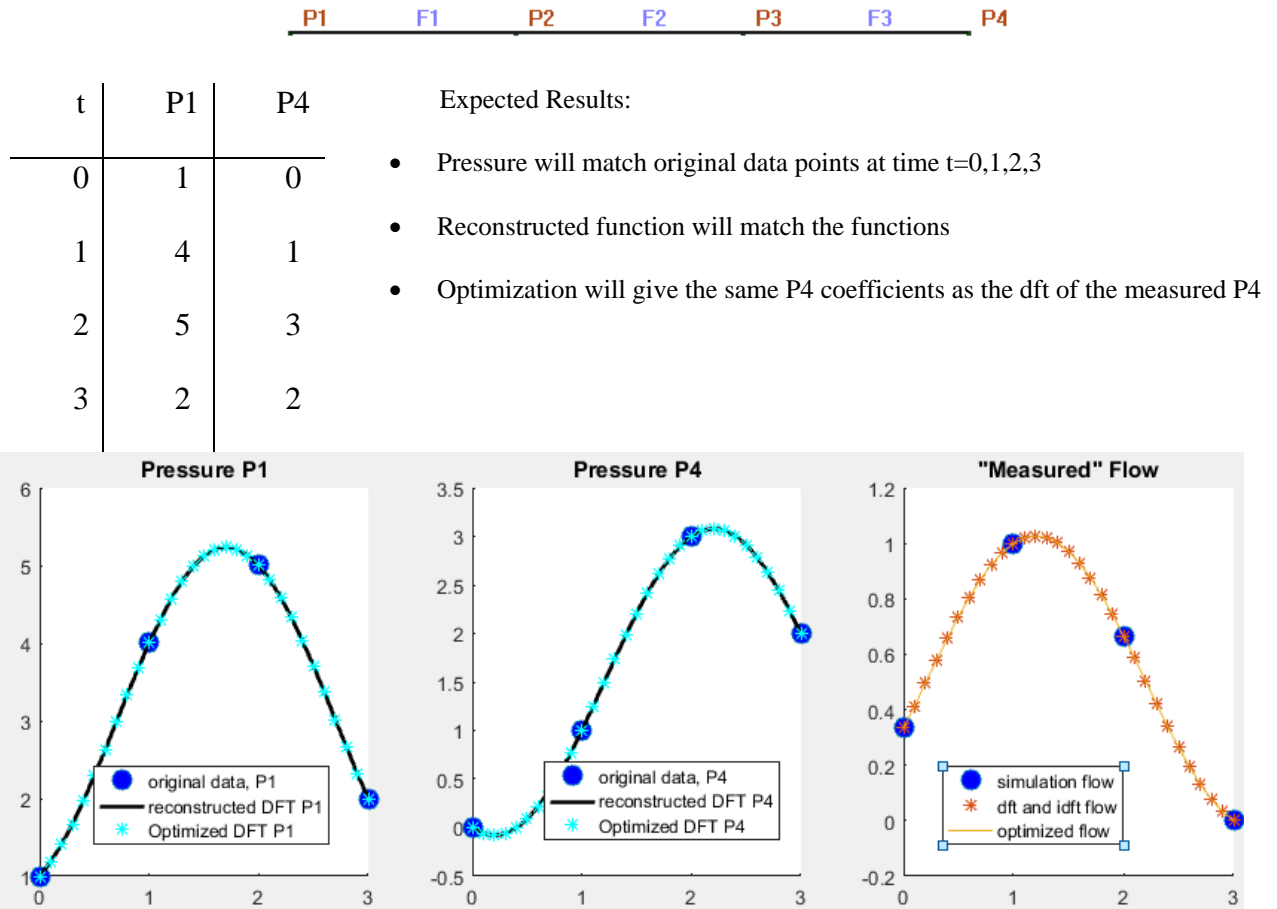
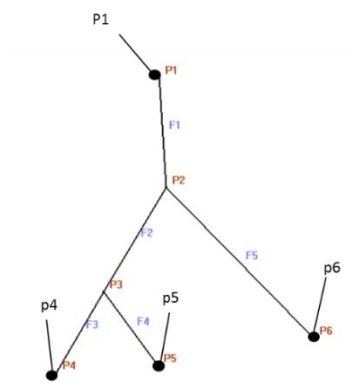


Figure 7.127. The dynamic inlet and outlet pressures and flow measurements. DFT/IDFT from simulation, DFT/IDFT from optimization (after deleting one BC) and the analytic solution from which the BCs were derived. The system is a simple tube with Dirichlet BCs and the sampling points were chosen at random. Top) The 4-node structure being used. Middle) The BCs at inlet and outlet point at 4 time points and expected results. Bottom) The optimization was capable of reconstructing the correct Fourier coefficients to make a signal that matches the BCs and measurements at the 4 time points.

Simple tree case 1. This case study follows the design of Section 7.15.8.1 but with a larger network. All pressure wave forms and expected results are given in each case. Case 1: 4 Pressures given with analytic solution known:

$$P_1 = 2 \cos\left(\frac{2\pi t}{4}\right) \quad P_4 = \cos\left(\frac{2\pi t}{4}\right) \quad P_5 = 1.1 \cos\left(\frac{2\pi t}{4}\right) \quad P_6 = 1.2 \cos\left(\frac{2\pi t}{4}\right) \quad (7.252)$$



t	P1	P4	P5	P6
0	2	1	1	1
1	0	0	0	0
2	-2	-1	-1	-1
3	0	0	0	0

Expected Results:

Pressure will match original data points at time $t=0,1,2,3$

Reconstructed function will match the functions

Optimization will give the same P6 coefficients as the dft of the measured P6

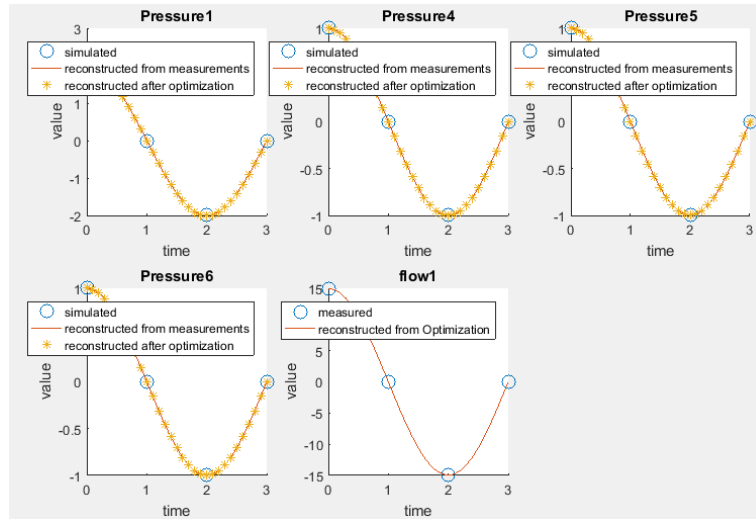
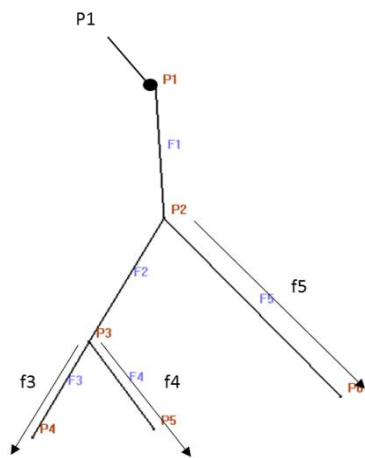


Figure 7.128. The dynamic inlet and outlet pressures and flow from measurements.

DFT/IDFT from simulation, DFT/IDFT from optimization (after deleting one BC) and the analytic solution from which the BCs were derived. The blue circle is the measured values, the red line is the reconstruction (DFT and IDFT) performed on the forward simulation prior to optimization. The gold stars are the reconstructions performed post-optimization in the fourier domain. The system is a simple network with Dirichlet BCs.

Simple tree case 2. 1 Pressure and 3 flows given with analytic solution known:

$$P_1 = 2 \cos\left(\frac{2\pi t}{4}\right) \quad P_4 = \cos\left(\frac{2\pi t}{4}\right) \quad P_5 = \sin\left(\frac{2\pi t}{4}\right) \quad P_6 = 2 \sin\left(\frac{2\pi t}{4}\right) \quad (7.253)$$



t	P1	F3	f4	F5
0	2	6.1	1.9	15.5
1	0	0.4	-1.1	-15.6
2	-2	-6.1	-1.9	-15.5
3	0	-0.4	1.1	15.6

Expected Results:

Pressure will match original data points at time t=0,1,2,3

Pressure reconstruction should match Dirichlet simulation (Case 2)

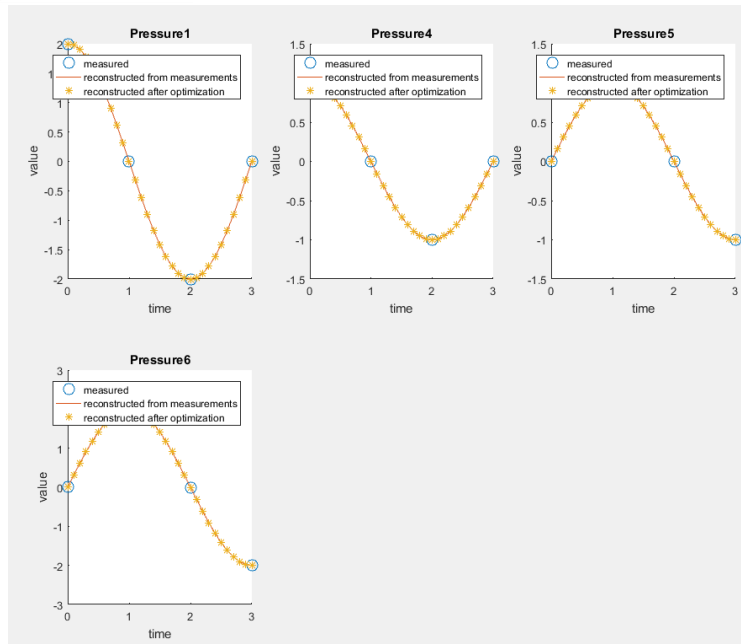
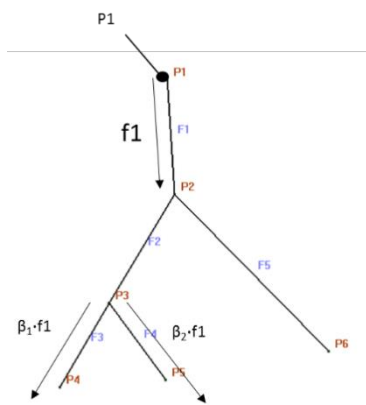


Figure 7.129. The dynamic inlet and outlet pressures and flow from measurements. DFT/IDFT from simulation, DFT/IDFT from optimization (after deleting one BC) and the analytic solution from which the BCs were derived. The blue circles are the measured values, the red line is the reconstruction (DFT and IDFT) performed on the forward simulation prior to optimization. The gold stars are the reconstructions performed post-optimization in the fourier domain. The system is a simple network with Neumann BCs.

Simple tree case 3. 1 Dirichlet (inlet) and 1 interior flow (inlet) and 2 fractional BCs (outlets):

$$P_1 = 2 \cos\left(\frac{2\pi t}{4}\right) \quad P_4 = \cos\left(\frac{2\pi t}{4}\right) \quad P_5 = \sin\left(\frac{2\pi t}{4}\right) \quad P_6 = 2 \sin\left(\frac{2\pi t}{4}\right) \quad (7.254)$$



t	P1	F1	f4	F5
0	2	2.0	1.9	15.5
1	0	6.1	-1.1	-15.6
2	-2	1.9	-1.9	-15.5
3	0	15.5	1.1	15.6

Expected Results:

Pressure will match original data points at time t=0,1,2,3
Pressure reconstruction should match Dirichlet simulation (Case 2)

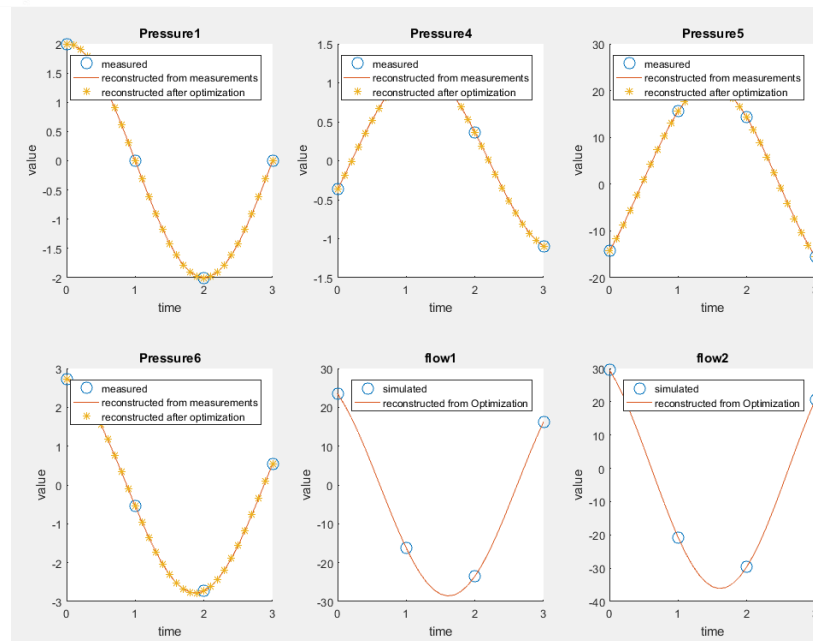


Figure 7.130. The dynamic inlet and outlet pressures and flow from measurements.

DFT/IDFT from simulation, DFT/IDFT from optimization (after deleting one BC) and the analytic solution from which the BCs were derived. The blue circles are the measured values, the red line is the reconstruction (DFT and IDFT) performed on the forward simulation prior to optimization. The gold stars are the reconstructions performed post-optimization in the fourier domain. The system is a simple network with 1 Dirichlet, 1 Neumann (Inlet) and 2 fractional (outlets) BCs.

7.15.8.2 Time-Dependent Patient

This is a case study uses all measurements and registers all measurements and BCs so that they are in phase prior to optimization. The goal is to optimize a human subject for time-dependent BCs that give the best possible fit to the measured data. The measurements from NOVA suffer from asynchronous pulsations (because they were each measured independently) which is exacerbated much beyond the natural pressure wave lag seen in the cerebral blood stream. Unfortunately, a rigid simulation with no deformations cannot account for such discrepancies.

The NOVA report includes both a nominal value and a time-lapse plot of blood flow for each measured vessel. Neither of these data adhere to conservation laws. This discrepancy is generated from a combination of experimental error (previously described) and the connection of these larger arteries to smaller vessels which are simply absent from the reconstructed vascular structure. To account for this, the measurements must be calibrated for mass conservation prior to optimizing the dynamic system. This step does not guarantee optimal alignment with the measured data, it only aids in comparison of the data after optimization. In order to account for experimental error in the time-dependent measurements, the nominal values calculated in the previous step can be used to rectify the experimental error in the time-dependent NOVA measurements. This error rectification can be executed as in equation (7.255).

$$\widehat{f}_i^n(t) = \widehat{f}_i(t) - \Delta \widehat{f}_i^{ss} \quad (7.255)$$

Where $\widehat{f}_i^n(t)$ is the new measurement at time t , $\widehat{f}_i(t)$ is the crude measurement at time t , and $\Delta\widehat{f}_i^{ss}$ is the difference between the steady-state measurement and the optimized steady-state flow. For reference, the results for the steady-state optimization are reported in Table 7.35.

Table 7.35: Measured Values (\widehat{f}) from NOVA report

Vessel	Outlets						Inlets		Sum	
	RMCA	RPCA	LPCA	LACA	LMCA	BA	RICA	LICA	Inlet	Outlet
Nova (ml/min)	166.00	70.00	89.00	62.00	172.00	181.00	168.00	312.00	661.00	559.00
Optimized	167.00	77.39	96.73	87.82	197.82	173.74	167.00	286.02	626.75	626.75

The optimization can now be performed using a Fourier series as mentioned above. To validate the results, each time step was simulate using the boundary pressures obtained in using the Fourier optimization method. The alignment of the rectified NOVA flows, the optimized flows and the simulated flows using the results of the optimization can be seen in Figure 7.131. The corresponding difference between the Fourier optimization and the simulation was $6.7056 \cdot 10^{-8} \pm 1.031 \cdot 10^{-8}$ and $8.602 \cdot 10^{-11} \pm 3.137 \cdot 10^{-11}$ for the flow and pressure respectively.

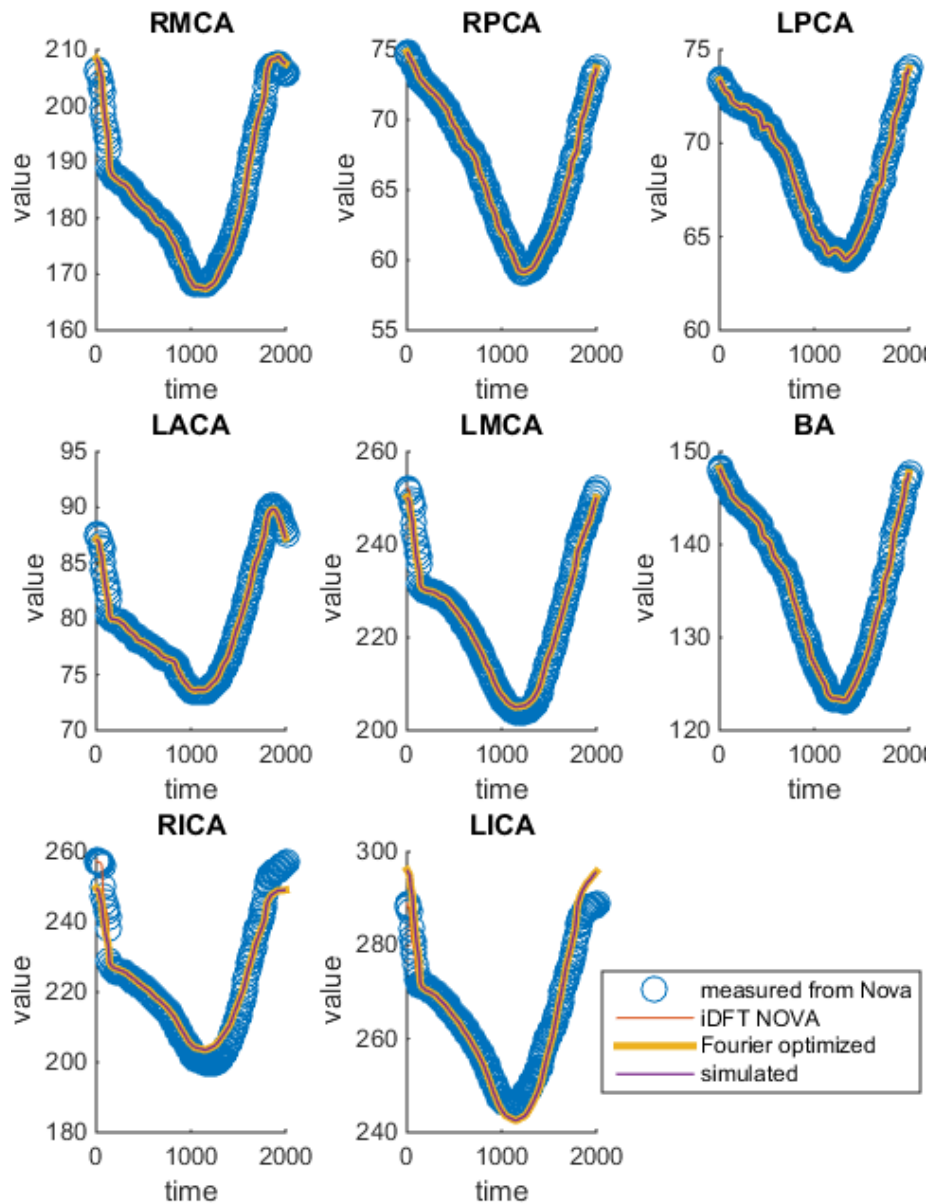


Figure 7.131. The alignment of the NOVA results and predicted flow patterns using Fourier Series optimization.

The NOVA report was sampled manually, then resampled using linear interpolation between measured values (to use uniform spacing). The NOVA report was adjusted to adhere to mass balances using the steady-state optimized results to dictate the deviation.

7.15.8.3 Optimization from measurements that are out of phase

For completeness, it is necessary to acknowledge how optimization accounts for out-of-phase input data. For the first step, the addition of a sine wave to one flow will give the phase shift necessary for the investigation. The predicted outcome is a phase-shifted flow, where the degree of phase shift is an optimized average similar to that achieved between the amplitude.

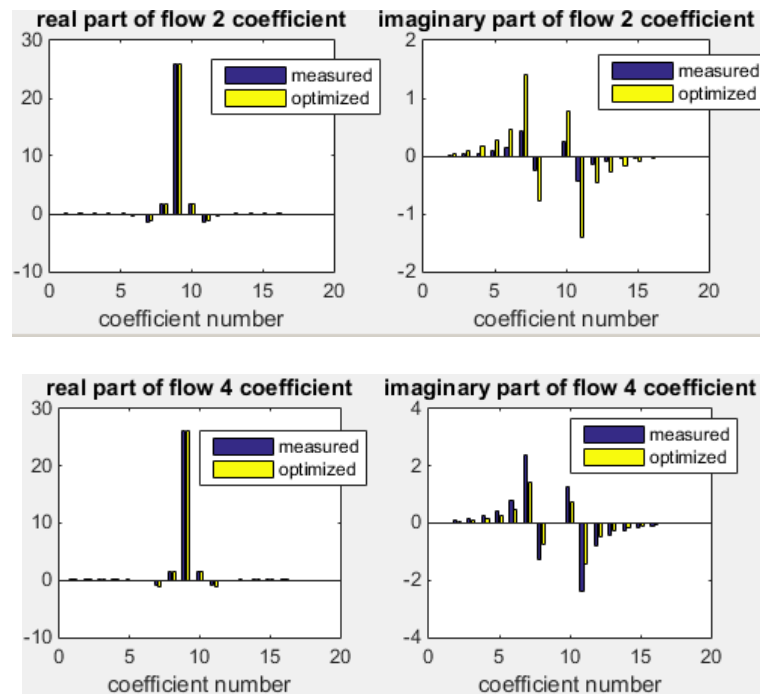


Figure 7.132. The resulting flow of two segments when the inlet and outlet BCs are not in phase with one another and the measurements have known error.

10 ml/min of error will be added to the measurements of flow 4 and 2 ml/min will be added to flows in segment 2. The optimization trades off in the Fourier domain in multiple frequencies to accommodate the phase shift in the measurements.

These are averaged in each frequency and the resulting wave form is an average in time for the two waveforms. To elucidate how these averages are assembled, some sample frequencies will be further investigated and the coefficients will be interrogated for trends (Figure 7.133). The final

reconstructed waveform can be seen below. The optimization averages in phase the same way it averages in amplitude. Independent visualizations of the averaging in both the real domain and the imaginary domain were evaluated to show that the averaging works in both domains independently (data not shown).

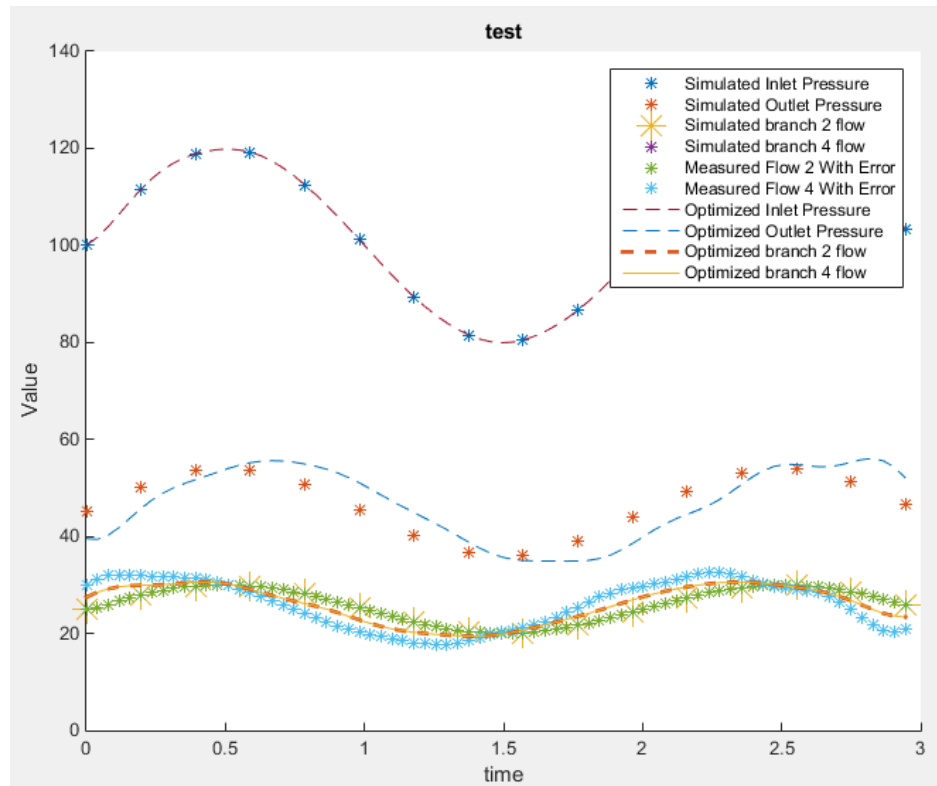


Figure 7.133. Optimization on a simple bifurcation with known error in amplitude and phase of measurements. This results in a complete averaging of each waveform in each frequency. The averaging occurs across both amplitude and phase.

7.15.9 Findings

When the system is fully determined, the optimization will not fail but rather gives the same result as the forward simulation. For best results, it is important to identify sources of error that may skew the results, such as a lack of mass conservation in the nominal flow values prior to optimization.

From this formulation, it is evident that optimizing in the Fourier domain is the same effort as solving in the time domain for a series of steady-state cases. The Fourier optimization does have the benefit of reconstructing the solution as a continuous function in time.

In the event flow measurements are available, the proposed optimization method is optimal for identifying BCs that give the closest matching simulation. Hand calculations have been performed as can be seen in Section 7.15.12 that show the symbolic solution to a system that has 1 measurement.

7.15.10 Fourier filtering and Gibbs phenomena

It is important to identify the level of high frequency noise generated during the Fourier series conversion of the source data. To investigate this, Figure 7.134 shows a reconstruction of the BA using a range of frequencies. It is difficult to see where the noise appears at this resolution, but it is apparent that the waveform is smoothly reconstructed with 21-25 terms before noise begins.

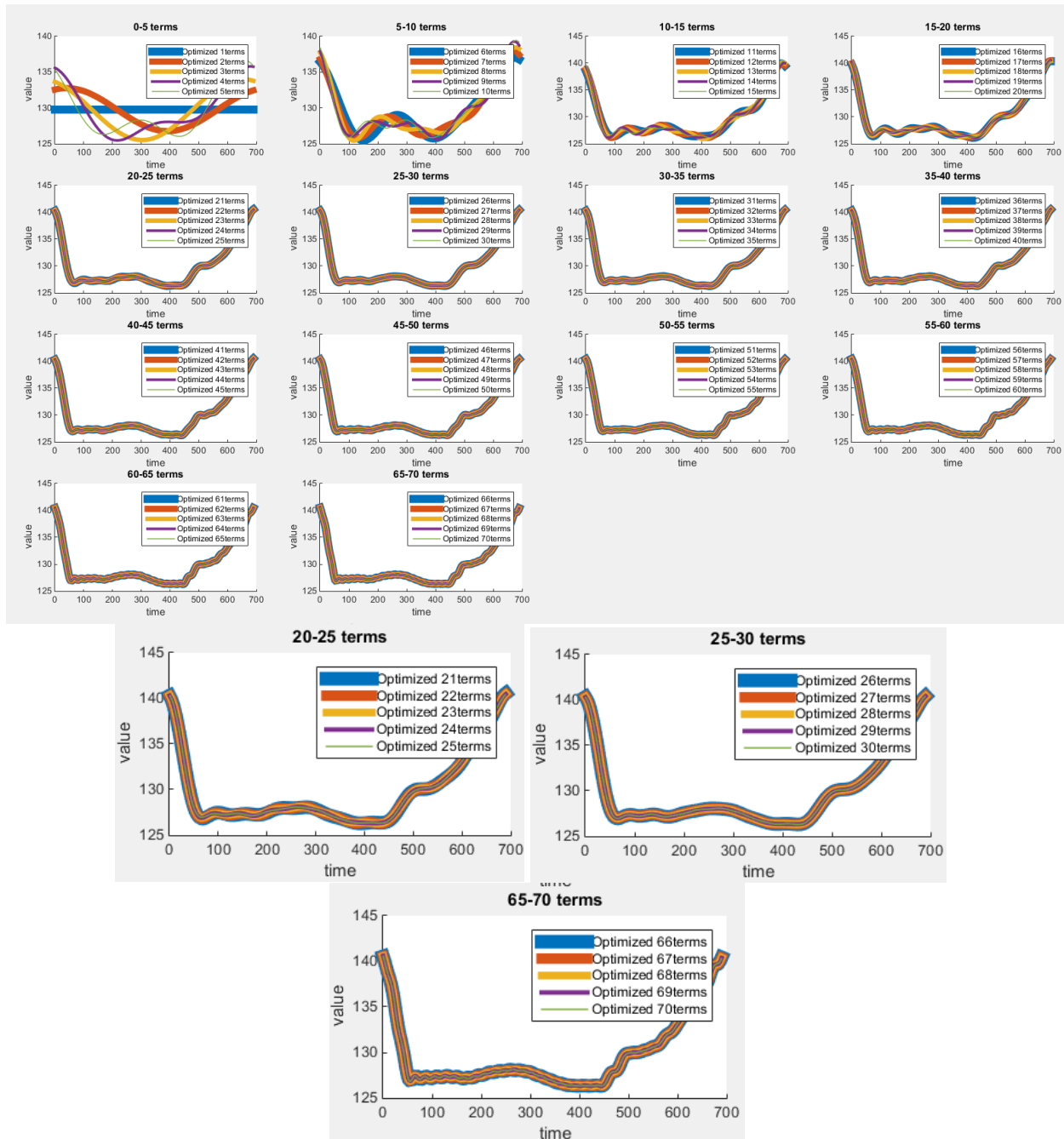


Figure 7.134. The reconstructed BA flows using a multitude of frequencies. It is difficult to see where the noise appears at this resolution, but we can see that the waveform is reconstructed well with 21-25 terms before the noise begins.

7.15.11 Considering phase shifts

When using perfectly aligned signals (pressure BCs and flow measurements), the signal can be easily reconstructed using the optimization paradigm. This can be seen demonstrated in Figure 7.135.

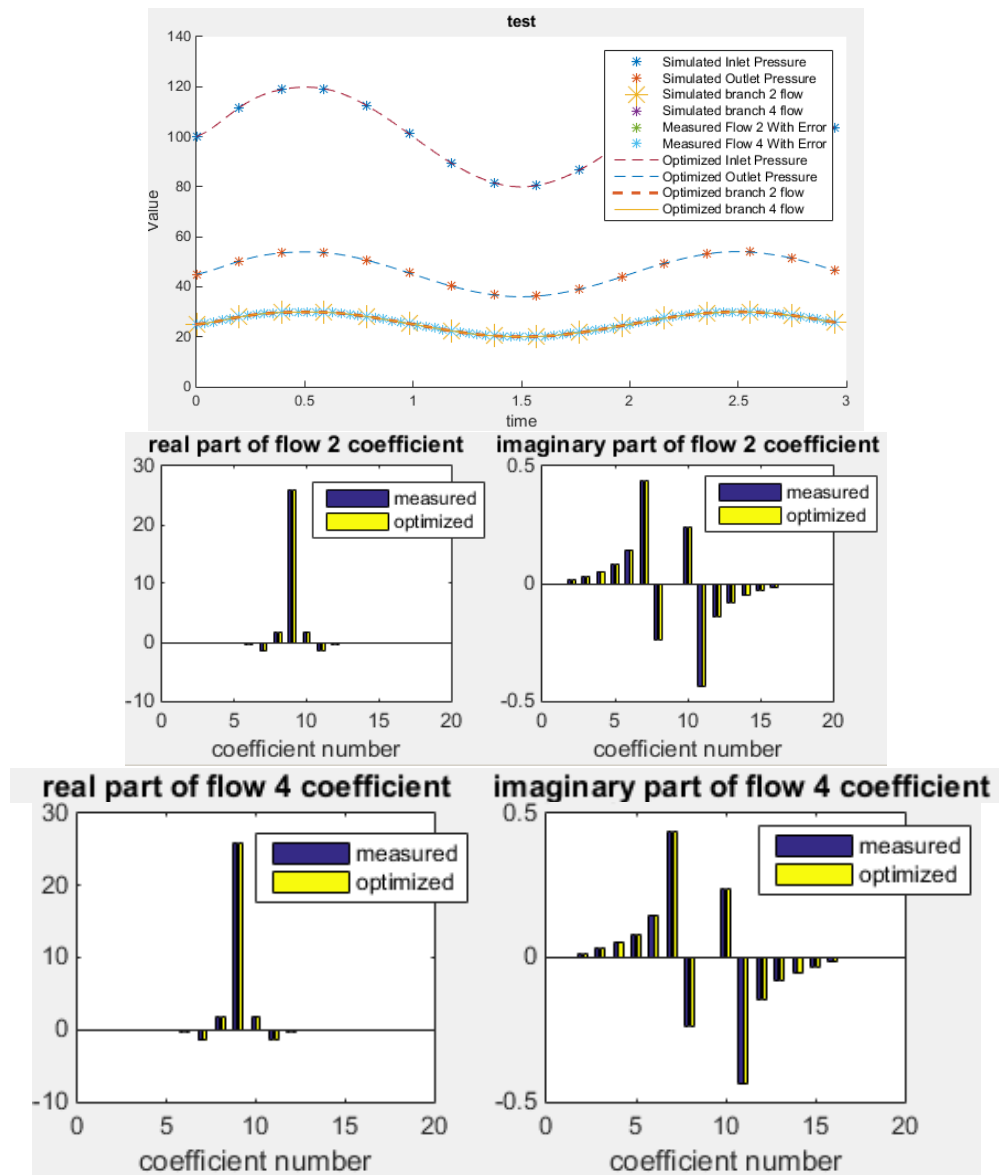


Figure 7.135. The resulting pressure and flow curves in time after being optimized with no error.

The result of using out of phase information, however, is less obvious. The next step is to see what happens when a phase shift is introduced between in inlet and the outlet pressure wave forms.

The first case study uses pressure BCs that are not in-phase in an effort to recreate a perfectly out-of-phase outlet boundary condition. These time points are commensurate with an analytical function ($\cos(2\pi/4 t)$ and $\sin(2\pi/4 t)$). The forward simulation was performed to calculate the measured flows for face 1. The result is a perfect recreation of the outflow signal.

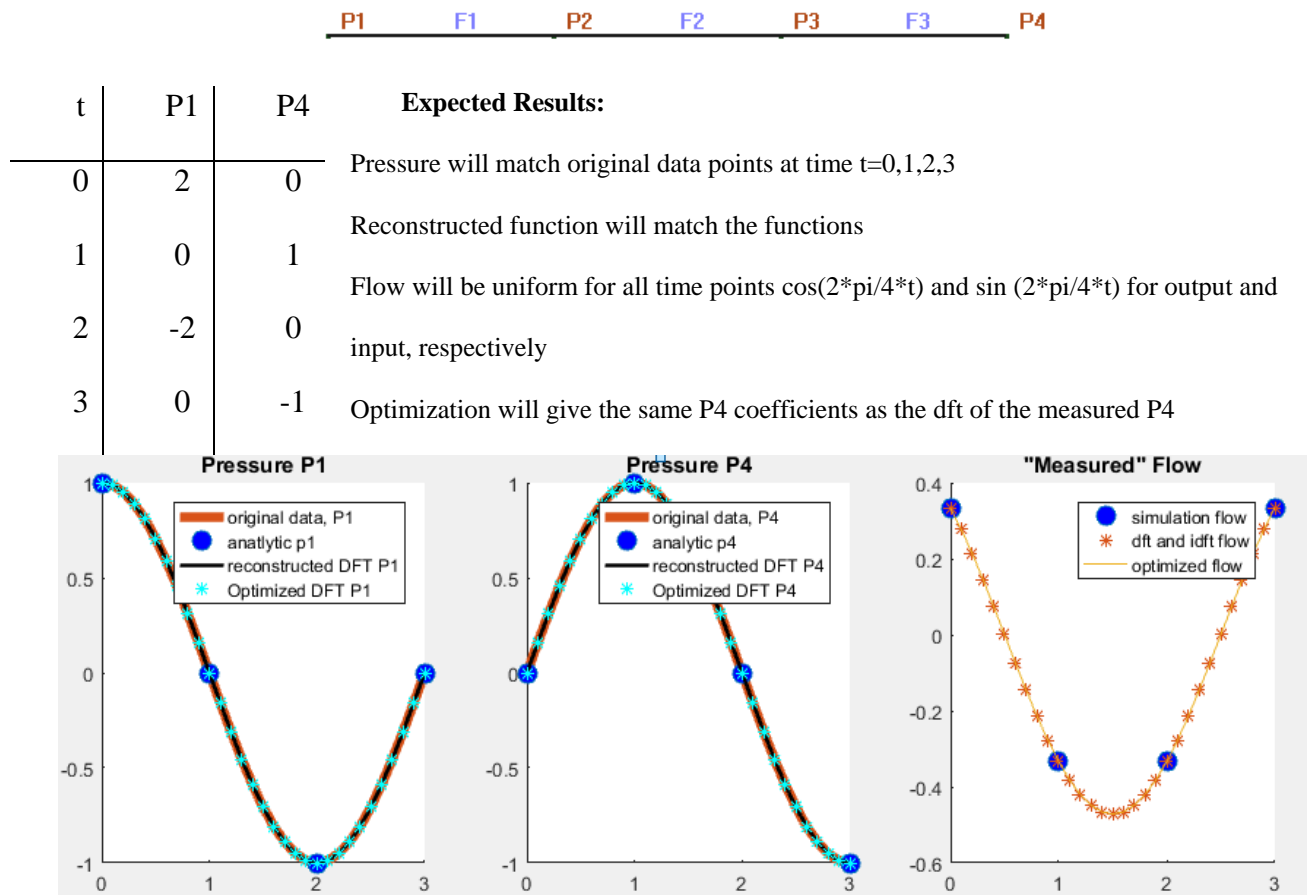


Figure 7.136. The dynamic inlet and outlet pressures and flow measurements. DFT/IDFT from simulation, DFT/IDFT from optimization (after deleting one BC) and the analytic solution from which the BCs were derived. The system is a simple tube with Dirichlet BCs. Here the inlet and outlet are out of phase with each other.

The same approach can be applied to the network using a flow wave form following Equation (7.256). The result is some sort of averaging in phase that minimizes the error between the simulation and the measured values. The measured flows are commensurate with the forward simulation.

$$f_6 = 50 + 10 \sin(\pi t) + 5 \cos(\pi t) \quad (7.256)$$

Another example of simulated phase error introduced into the boundaries can be seen in Figure 7.138. Again, the solution is an averaging in the phase spectra as well as the magnitude spectra.

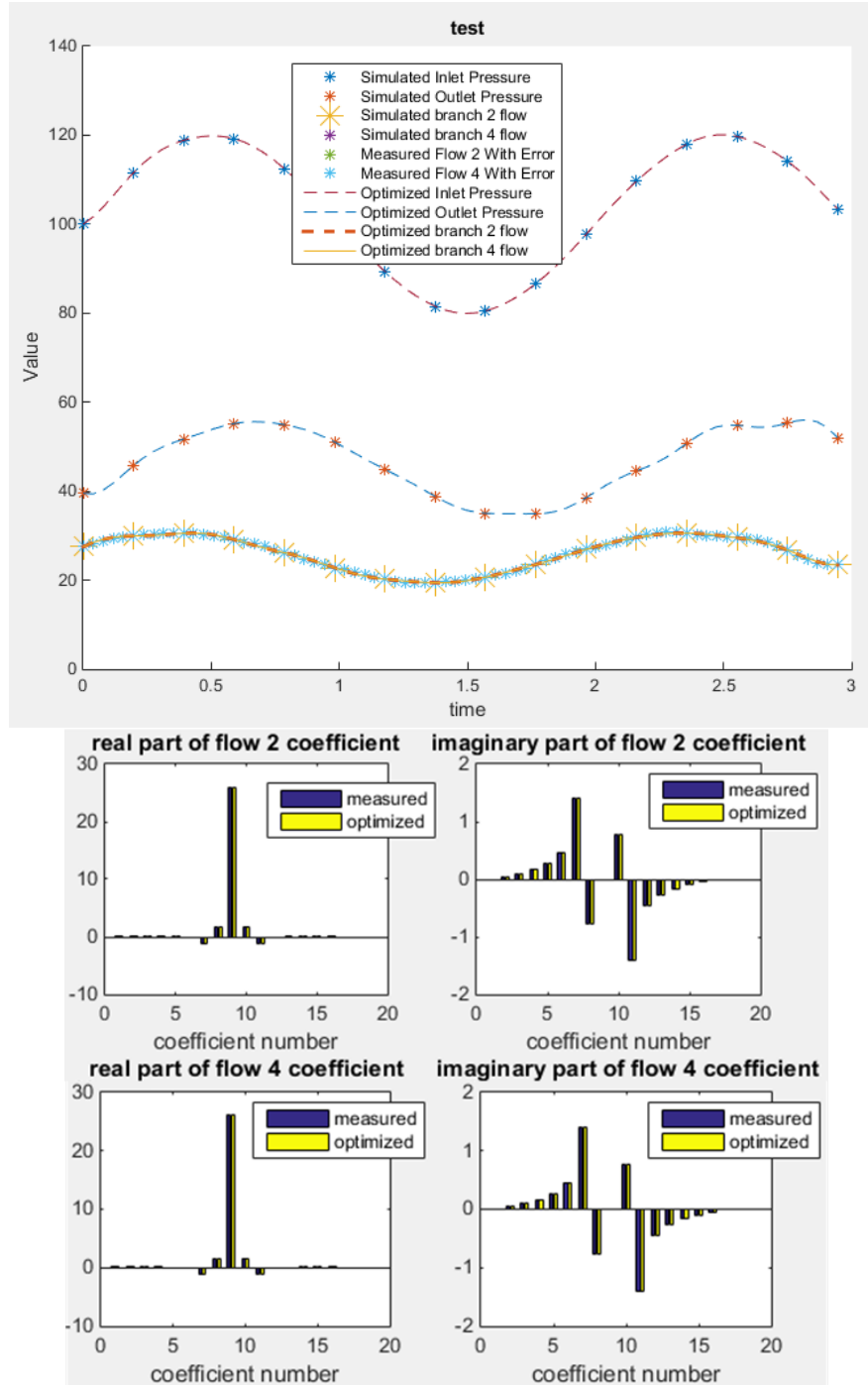


Figure 7.137. The resulting pressure and flow when the inlet and outlet BCs are not in phase with one another and the measurements are commensurate with the forward simulation. The optimization algorithm is capable of reconstructing the missing BC perfectly.

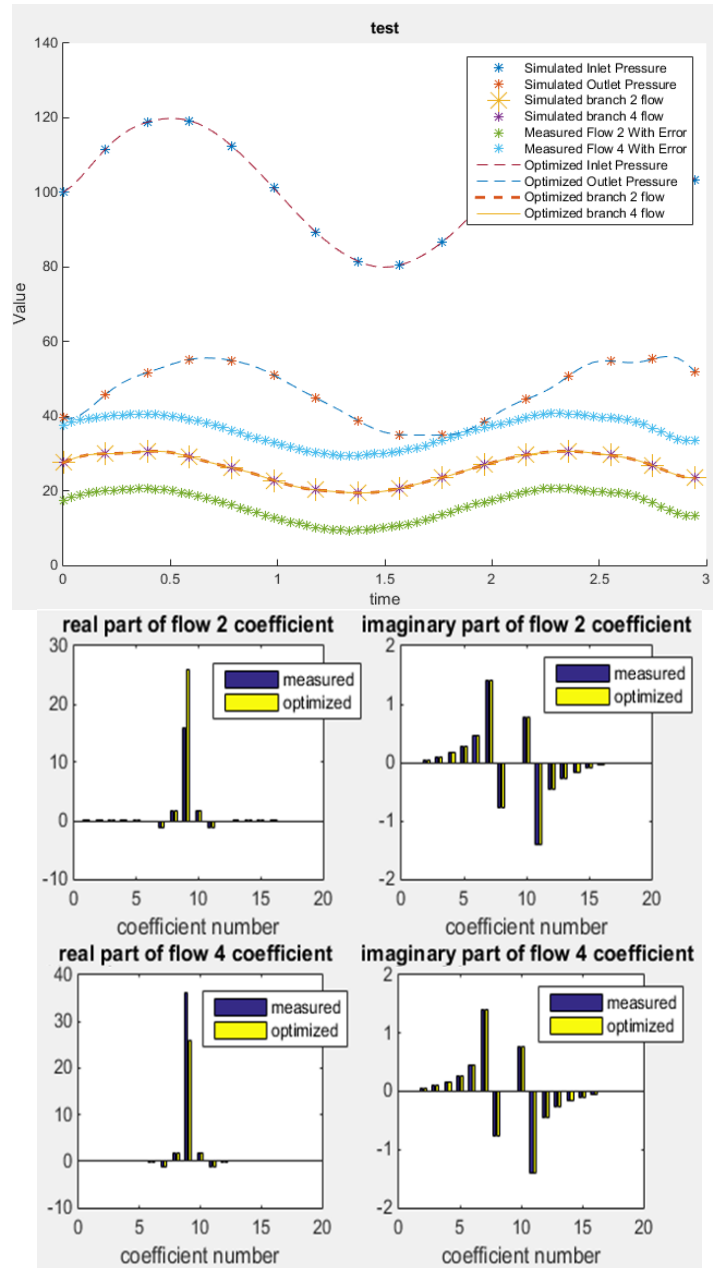
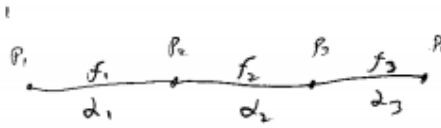


Figure 7.138. The resulting pressure and flow when the inlet and outlet BCs are not in phase with one another and the measurements have known error.

In this case, 10 ml/min was added to flow4 and 10 ml/min was subtracted from from all flow measurements in flow2. The optimization algorithm is capable of reconstructing the missing BC perfectly. The tradeoff occurs in the coefficient corresponding only to the nominal value.

7.15.12 Hand calculated symbolic validation:

(C)



$$\begin{aligned} \bar{f}_1 &= \bar{f}_1 \\ p_1 &= \bar{p}_1 \end{aligned} \quad \left\{ \begin{aligned} \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix} p &= \begin{bmatrix} d_1 & 0 & 0 \\ 0 & d_2 & 0 \\ 0 & 0 & d_3 \end{bmatrix} f \\ \begin{bmatrix} \frac{1}{2} & -\frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & -\frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} & -\frac{1}{2} \end{bmatrix} p &= f \end{aligned} \right.$$

$$X(f) = [w(f\bar{f})]^T [w(f\bar{f})]$$

$$x(f) = [wf - w\bar{f}]^T [wf - w\bar{f}]$$

$$x(f) = f^T w^T w f - 2 \bar{f}^T w^T w f + \bar{f}^T w^T w \bar{f}$$

$$X(p) = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \end{bmatrix} \begin{bmatrix} d_1 & 0 & 0 \\ -\frac{1}{2}d_1 & \frac{1}{2}d_1 & 0 \\ 0 & -\frac{1}{2}d_2 & \frac{1}{2}d_2 \\ 0 & 0 & -\frac{1}{2}d_3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{2} & -\frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & -\frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} & -\frac{1}{2} \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix}$$

$$\rightarrow \begin{bmatrix} \bar{f}_1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{2} & -\frac{1}{2} & 0 & 0 \\ 0 & \frac{1}{2} & -\frac{1}{2} & 0 \\ 0 & 0 & \frac{1}{2} & -\frac{1}{2} \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix}$$

$$+ \begin{bmatrix} \bar{f}_1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \bar{f}_1 \\ 0 \\ 0 \end{bmatrix}$$

$$X(p) = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \end{bmatrix} \begin{bmatrix} \frac{1}{2}d_1 & 0 & 0 \\ -\frac{1}{2}d_1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{2} & -\frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix} - 2 \begin{bmatrix} \bar{f}_1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{2}d_1 & -\frac{1}{2}d_1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix} + \begin{bmatrix} \bar{f}_1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \bar{f}_1 \\ 0 \\ 0 \end{bmatrix}$$

$$X(p) = \begin{bmatrix} \frac{1}{2}d_1 - \frac{p_2}{2} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_1 \\ \frac{1}{2}d_1 - \frac{p_2}{2} \\ 0 \\ 0 \end{bmatrix} - 2 \begin{bmatrix} \bar{f}_1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{1}{2}d_1 - \frac{p_2}{2} \\ 0 \\ 0 \end{bmatrix} + \bar{f}_1^2$$

$$X(p) = \left(\frac{p_1^2}{2d_1} - 2 \frac{p_1 p_2}{d_1} + \frac{p_2^2}{2d_1} \right) - 2 \frac{\bar{f}_1 p_1}{d_1} + \frac{2 \bar{f}_1 p_2}{d_1} + \bar{f}_1^2$$

$$h(p) = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} \frac{1}{2} \alpha_1 & -\frac{1}{2} \alpha_1 & 0 & 0 \\ 0 & \frac{1}{2} \alpha_2 & -\frac{1}{2} \alpha_2 & 0 \\ 0 & 0 & \frac{1}{2} \alpha_3 & -\frac{1}{2} \alpha_3 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix}$$

$$h(p) = \begin{bmatrix} \frac{1}{2} \alpha_1 & -\frac{1}{2} \alpha_1 - \frac{1}{2} \alpha_2 & \frac{1}{2} \alpha_2 & 0 \\ 0 & \frac{1}{2} \alpha_2 & -\frac{1}{2} \alpha_2 - \frac{1}{2} \alpha_3 & \frac{1}{2} \alpha_3 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix} = \begin{pmatrix} \frac{1}{2} \alpha_1 - \frac{1}{2} \alpha_2 - \frac{1}{2} \alpha_3 + \frac{1}{2} \alpha_3 \\ \frac{1}{2} \alpha_2 - \frac{1}{2} \alpha_2 - \frac{1}{2} \alpha_3 + \frac{1}{2} \alpha_3 \end{pmatrix} \quad \begin{matrix} \text{BC} \\ (4+6) \end{matrix}$$

$$\mathcal{L}_{\lambda p} = (X(p) + \lambda^T h(p))$$

$$\begin{aligned} \mathcal{L}_{\lambda p} &= \frac{p_1^2}{\alpha_1^2} - 2 \frac{p_1 p_2}{\alpha_1^2} + \frac{p_2^2}{\alpha_1^2} - \frac{2 \bar{f}_1 p_1}{\alpha_1} + \frac{2 \bar{f}_1 p_2}{\alpha_1} + \bar{f}_1^2 \\ &\quad + \lambda_1 \left(\frac{1}{2} \alpha_1 - \frac{1}{2} \alpha_2 - \frac{1}{2} \alpha_3 + \frac{1}{2} \alpha_3 \right) \\ &\quad + \lambda_2 \left(\frac{1}{2} \alpha_2 - \frac{1}{2} \alpha_2 - \frac{1}{2} \alpha_3 + \frac{1}{2} \alpha_3 \right) \\ &\quad + \lambda_3 (p_1 - \bar{p}_1) \end{aligned}$$

$$\nabla_{p, \lambda} \mathcal{L}_{\lambda p} = \begin{pmatrix} \frac{2 p_1}{\alpha_1^2} - \frac{2 p_2}{\alpha_1^2} - 2 \frac{\bar{f}_1}{\alpha_1} + \lambda_1/2 + \lambda_3 \\ -2 \frac{p_1}{\alpha_1} + \frac{2 p_2}{\alpha_1^2} + 2 \frac{\bar{f}_1}{\alpha_1} + \lambda_1(-1/2 - 1/2) + \lambda_2/2 \\ \lambda_1/2 - \lambda_2/2 - \lambda_3/2 \\ \lambda_2/2 \end{pmatrix}$$

(3)

$$\nabla_{\lambda} \mathcal{L}_{\lambda P} = \begin{pmatrix} p_1/2_1 - p_2/2_1 - p_3/2_2 + p_3/2_2 \\ p_2/2_2 - p_3/2_2 - p_3/2_3 + p_4/2_3 \\ p_1 - \bar{p}_1 \\ \text{rhs} \end{pmatrix}$$

$$\nabla_{\lambda P} \mathcal{L}_{\lambda P} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 & \lambda_1 & \lambda_2 & \lambda_3 \\ 2/2_1 & -2/2_1 & 0 & 0 & 1/2_1 & 0 & 1 \\ -2/2_1 & 2/2_1 & 0 & 0 & -1/2_1 - 1/2_2 & 1/2_2 & 0 \\ 0 & 0 & 0 & 0 & 1/2_2 & -1/2_2 - 1/2_3 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/2_3 & 0 \\ 1/2_1 & -1/2_1 - 1/2_2 - 1/2_2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/2_2 & -1/2_2 - 1/2_3 & 1/2_3 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{bmatrix} = \begin{bmatrix} 2f_1/2_1 \\ -2f_1/2_1 \\ 0 \\ 0 \\ 0 \\ 0 \\ \bar{p}_1 \end{bmatrix}$$

The solution using $\bar{p}_1 = 95$ mmHg and $\bar{f}_1 = 122.756$ ml/min. Alpha is computed offline and given here for ease of simulation for comparison. Matlab code input is offered:

```
p1Bar = 95; %mmHg
alpha = [1.3577062793e-05, 1.3577062793e-05, 1.3577062793e-05]*1000;
f1Bar = 122.756054979338;
P1Bar = 95;
```


Manual
bVector:
18082858.8408132
-18082858.8408132
0
0
0
0
95

results
95
93.3333333333246
91.6666666666492
89.9999999999737
0
0
0

flow
122.756054979337
122.756054979337
122.756054979337

Algorithmic/Matrix implementation
bVector:
18082859149.3685
-18082859149.3685
0
0
0
0
95

Results:
95
93.33333333617636
91.6666667235272
90.0000000852908
0
0
0

flow
122.756054979338
122.756054979338
122.756054979337

AMatrix =

1.08E+10	-1.1E+10	0	0	73653.63	0	1
-1.1E+10	1.08E+10	0	0	-147307	73653.63	0
0	0	0	0	73653.63	-147307	0
0	0	0	0	0	73653.63	0
73653.63	-147307	73653.63	0	0	0	0
0	73653.63	-147307	73653.63	0	0	0
1	0	0	0	0	0	0

Algorithmic/Matrix implementation

AMatrix:

1.08E+10	-1.1E+10	0	0	-73653.6	0	1
-1.1E+10	1.08E+10	0	0	147307.3	-73653.6	0
0	0	0	0	-73653.6	147307.3	0
0	0	0	0	0	-73653.6	0
-73653.6	147307.3	-73653.6	0	0	0	0
0	-73653.6	147307.3	-73653.6	0	0	0
1	0	0	0	0	0	0

Matrix formulation:

AMatrix =

$2/(\alpha(1)^2)$	$-2/(\alpha(1)^2)$	0	0	$1/\alpha(1)$	0	1
				$-1/\alpha(1)-$		
$-2/(\alpha(1)^2)$	$2/(\alpha(1)^2)$	0	0	$1/\alpha(2)$	$1/\alpha(2)$	0
					$-1/\alpha(2)-$	
0	0	0	0	$1/\alpha(2)$	$1/\alpha(3)$	0
0	0	0	0	0	$1/\alpha(3)$	0
	$-1/\alpha(1)-$					
$1/\alpha(1)$	$1/\alpha(2)$	$1/\alpha(2)$	0	0	0	0
		$-1/\alpha(2)-$				
0	$1/\alpha(2)$	$1/\alpha(3)$	$1/\alpha(3)$	0	0	0
1	0	0	0	0	0	0

```
rhs = [ 2*f1Bar/alpha(1);  
-2*f1Bar/alpha(1);  
0;  
0;  
0;  
0;  
P1Bar];  
  
SolutionVector = AMatrix\rhs;  
flow = [1/alpha(1) -1/alpha(1) 0 0  
0 1/alpha(2) -1/alpha(2) 0  
0 0 1/alpha(3) -1/alpha(3)]*SolutionVector(1:4);
```

7.15.12.1 Manual Implementation

```
p1Bar = 95; %mmHg
alpha = [1.3577062793e-05, 1.3577062793e-05, 1.3577062793e-05]*1000;
f1Bar = 122.756054979338;
P1Bar = 95;

AMatrix = [2/(alpha(1)^2) -2/(alpha(1)^2) 0 0 1/alpha(1) 0 1
           -2/(alpha(1)^2) 2/(alpha(1)^2) 0 0 -1/alpha(1)- 1/alpha(2) 1/alpha(2) 0
           0 0 0 0 1/alpha(2) -1/alpha(2)-1/alpha(3) 0
           0 0 0 0 0 1/alpha(3) 0
           1/alpha(1) -1/alpha(1)-1/alpha(2) 1/alpha(2) 0 0 0 0
           0 1/alpha(2) -1/alpha(2)-1/alpha(3) 1/alpha(3) 0 0 0
           1 0 0 0 0 0 0
           ];
rhs = [ 2*f1Bar/alpha(1); -2*f1Bar/alpha(1); 0; 0; 0; 0; 0; P1Bar];

SolutionVector = AMatrix\rhs;

flow = [1/alpha(1) -1/alpha(1) 0 0
        0 1/alpha(2) -1/alpha(2) 0
        0 0 1/alpha(3) -1/alpha(3)]*SolutionVector(1:4);
```

7.15.13 Case study for frequency independence in the Fourier domain representation of the optimization problem

A case study has been generated to explore the frequency independence (problem separation within each frequency) for the Fourier problem formulation.

t	P1	P4
0	2	1
1	0	0
2	-2	-1
3	0	0

P1 F1 P2 F2 P3 F3 P4

$$A = \begin{bmatrix} -\alpha_1 & & & \\ & -\alpha_2 & & \\ & & -\alpha_3 & \\ & & & -\alpha_4 \end{bmatrix}; \quad Z_1 = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix}; \quad z_2 = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \end{bmatrix}$$

$$z(p, k) \min_{f, p} \left\| \left(\hat{f}_1 V_1 + \hat{f}_2 V_2 + \hat{f}_3 V_3 + \hat{f}_4 V_4 \right) - A^{-1} Z_1 (\hat{p}_1 V_1 + \hat{p}_2 V_2 + \hat{p}_3 V_3 + \hat{p}_4 V_4) \right\|$$

s.t.

$$A^{-1} Z_1 (\hat{p}_1 V_1 + \hat{p}_2 V_2 + \hat{p}_3 V_3 + \hat{p}_4 V_4) = 0$$

$$Z_3 (\hat{p}_1 V_1 + \hat{p}_2 V_2 + \hat{p}_3 V_3 + \hat{p}_4 V_4) - (\hat{p}_1 V_1 + \hat{p}_2 V_2 + \hat{p}_3 V_3 + \hat{p}_4 V_4) = 0$$

Where each \hat{f}_k and \hat{p}_k represents a vector of unknowns for k^{th} frequency

With the problem now formulated using the sum of unknown coefficients multiplied by known weight functions (V_k), we can rewrite the summation in matrix form.

$$z(p, k) \min_{f, p} \left\| \begin{bmatrix} \hat{f}_1 \\ \hat{f}_2 \\ \hat{f}_3 \\ \hat{f}_4 \end{bmatrix} - A^{-1} Z_1 \begin{bmatrix} \hat{p}_1 \\ \hat{p}_2 \\ \hat{p}_3 \\ \hat{p}_4 \end{bmatrix} \right\|$$

s.t.

$$A^{-1}Z_1 \begin{bmatrix} V_1 & & & \\ & V_2 & & \\ & & V_2 & \\ & & & V_4 \end{bmatrix} \begin{bmatrix} \hat{p}_1 \\ \hat{p}_2 \\ \hat{p}_3 \\ \hat{p}_4 \end{bmatrix} = 0$$

$$Z_3 \begin{bmatrix} V_1 & & & \\ & V_2 & & \\ & & V_2 & \\ & & & V_4 \end{bmatrix} \begin{bmatrix} \hat{p}_1 \\ \hat{p}_2 \\ \hat{p}_3 \\ \hat{p}_4 \end{bmatrix} - \begin{bmatrix} V_1 & & & \\ & V_2 & & \\ & & V_2 & \\ & & & V_4 \end{bmatrix} \begin{bmatrix} \tilde{p}_1 \\ \tilde{p}_2 \\ \tilde{p}_3 \\ \tilde{p}_4 \end{bmatrix} = 0$$

Here, each coefficient corresponds to an isolated vector V. i.e. each vector of unknowns does not affect other unknown in any of the equations, making this a separatable problem.

7.16 Appendix P: Validation of matrix identities

- I. $\bar{\nabla}(Ax) = A$
 - II. $\bar{\nabla}(x^T Ax) = (A^T + A)x$
 - III. $\bar{\nabla}(a^T x) = a$
 - IV. $\bar{\nabla}(x^T Sx) = 2Sx$ when S is symmetric ($S^T=S$)
- i. A linear algebraic set of equations can be defined in Equation (7.379). This linear algebraic set of equations written in component form in Equation (7.258). The derivative is given in Equation (7.259). The result of the derivative returns the original A matrix as in Equation (7.259). This holds the relationship in I.

$$Ax = \begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 0 \\ 0 & 1 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad (7.257)$$

$$Ax = \begin{bmatrix} x_1 + 3x_2 + 5x_3 \\ 2x_1 + 4x_2 \\ x_2 + 3x_3 \end{bmatrix} \quad (7.258)$$

$$\bar{\nabla} \begin{bmatrix} x_1 + 3x_2 + 5x_3 \\ 2x_1 + 4x_2 \\ x_2 + 3x_3 \end{bmatrix} = \begin{bmatrix} \frac{df_1}{dx_1} & \frac{df_1}{dx_2} & \frac{df_1}{dx_3} \\ \frac{df_2}{dx_1} & \frac{df_2}{dx_2} & \frac{df_2}{dx_3} \\ \frac{df_3}{dx_1} & \frac{df_3}{dx_2} & \frac{df_3}{dx_3} \end{bmatrix} = \begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 0 \\ 0 & 1 & 3 \end{bmatrix} = A \quad (7.259)$$

Verified

- ii. The nonlinear set of equations in Equation (7.260) can be written in component form as in Equation (7.261). The derivative is given in Equation (7.262). This is equivalent to the evaluation of $(A^T + A)x$ as seen in equation (7.263). This verifies the relationship in II.

$$x^T Ax = [x_1 \quad x_2 \quad x_3] \begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 0 \\ 0 & 1 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad (7.260)$$

$$[x_1 \quad x_2 \quad x_3] \begin{bmatrix} x_1 + 3x_2 + 5x_3 \\ 2x_1 + 4x_2 \\ x_2 + 3x_3 \end{bmatrix} \quad (7.261)$$

$$x_1^2 + 3x_1x_2 + 5x_1x_3 + 2x_1x_2 + 4x_2^2 + x_2x_3 + 3x_3^2$$

$$\vec{\nabla} \cdot (x_1^2 + 3x_1x_2 + 5x_1x_3 + 2x_1x_2 + 4x_2^2 + x_2x_3 + 3x_3^2) =$$

$$[2x_1 + 3x_2 + 5x_3 + 2x_2 \quad 3x_1 + 2x_1 + 8x_2 + x_3 \quad 5x_1 + x_2 + 6x_3] \quad (7.262)$$

$$= \begin{bmatrix} 2 & 5 & 5 \\ 5 & 8 & 1 \\ 5 & 1 & 6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

$$(A^T + A)x = \left(\begin{bmatrix} 1 & 3 & 5 \\ 2 & 4 & 0 \\ 0 & 1 & 3 \end{bmatrix} + \begin{bmatrix} 1 & 2 & 0 \\ 3 & 4 & 1 \\ 5 & 0 & 3 \end{bmatrix} \right) \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 2 & 5 & 5 \\ 5 & 8 & 1 \\ 5 & 1 & 6 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad (7.263)$$

- iii. Two vectors (coefficient vector a and unknown vector x) are given in Equation (7.264). The differential of the component form (Equation (7.265)) is equivalent to the coefficient vector as given in Equation (7.265). This verifies the relationship in III.

$$a^T x = [2 \quad 1 \quad 4] \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = 2x_1 + x_2 + 4x_3 \quad (7.264)$$

$$\vec{\nabla}(2x_1 + x_2 + 4x_3) = \begin{bmatrix} df/dx_1 \\ df/dx_2 \\ df/dx_3 \end{bmatrix} \quad (7.265)$$

$$\vec{\nabla}(2x_1 + x_2 + 4x_3) = \begin{bmatrix} 2 \\ 1 \\ 4 \end{bmatrix}$$

iv. A set of linear nonlinear equations defined by a symmetric coefficient matrix is given in Equation (7.266). This is a special case of II where the derivative is given as $(A^T + A)x$. In the event of a symmetric matrix, this simplifies to $2Ax$ because $A^T = A$. This verifies the relationship in IV.

$$\vec{\nabla}(x^T Ax) = (A^T + A)x$$

$$A^T = A \quad (7.266)$$

$$(A^T + A)x = (A + A)x = 2Ax \quad (7.267)$$

7.17 Appendix Q: Discrete Fourier series

Biological signals are frequently periodic as they follow a periodic gate. Therefore, it is reasonable to model these signals using oscillatory functions such as sine and cosine. These functions belong to the family of transcendental functions (exponential function, sine, cosine). It can be shown that transcendental functions are convergent. This relationship is commonly known as the Euler identity as in Equations (7.268)-(7.270).

$$e^{-ix} = \cos(x) - i\sin(x) \quad (7.268)$$

$$e^{-ix} = \sum_{j=0}^{\infty} \frac{(-ix)^j}{j!} = \frac{(-ix)^0}{0!} + \frac{(-ix)^1}{1!} + \frac{(-ix)^2}{2!} + \frac{(-ix)^3}{3!} \dots$$

$$e^{-ikx} = \frac{1}{0!} + \frac{-ix}{1!} + \frac{-(x)^2}{2!} + \frac{i(x)^3}{3!} + \frac{(x)^4}{4!} \dots \quad (7.269)$$

$$\cos(kx) = -\left(\frac{1}{0!} + \frac{-(x)^2}{2!} + \frac{(x)^4}{4!} \dots\right); \quad i \cdot \sin(kx) = i \cdot \left(\frac{-x}{1!} + \frac{(x)^3}{3!} \dots\right) \quad (7.270)$$

Moreover, sine and cosine can be defined by the sum of exponential functions as in Equation (7.271).

$$\sin(x) = \frac{e^{ix} - e^{-ix}}{2i} \quad \cos(x) = \frac{e^{ix} + e^{-ix}}{2} \quad (7.271)$$

Note that exponential function was developed as a convergent infinite series that is a solution to differential equations, particularly 1st order homogenous differential equations (ODEs).

If considering a system with complex eigenvalues (upon eigenvalue decomposition of the differential system), the solution will give exponentials with complex exponents, which are merely sines and cosines. This is consistent with the understanding that all differential systems with complex eigenvalues are oscillatory in nature (even without oscillatory forcing functions). The derivation of the generalized Fourier series and finite Fourier series is given elsewhere [242].

7.17.1 Discrete Fourier Transform (DFT) - complex arithmetic

7.17.1.1 Truncation of Order N

To compute the Fourier series for discrete signals with N data points, the infinite Fourier series is truncated to a maximum of N values as in Equation (7.272). This is because N data points are used to exactly match N degrees of freedom in the Fourier coefficients, making the system fully determined. It is programmatically convenient to shift the indices from the range $k = -N/2 \dots +N/2$ to the range $\kappa = 0 \dots N-1$ with the shift factor $\kappa = k + N/2$ as explained elsewhere [243]. This is further investigated in *Section 7.17.4*.

$$f_N^*(x) = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} \tilde{f}_k e^{-i k x} = \sum_{\kappa=0}^{N-1} \tilde{f}_\kappa e^{-i \left(\kappa - \frac{N}{2}\right) x}$$

$$\tilde{f}_\kappa = \frac{1}{2\pi} \int_0^{2\pi} f(x) e^{-i \left(\kappa - \frac{N}{2}\right) x} dx, \quad \kappa = 0 \dots N-1, \quad \varphi_\kappa = e^{-i \left(\kappa - \frac{N}{2}\right) x} \quad (7.272)$$

$$\text{Lower Bound: } k = -N/2 \quad \kappa = 0$$

$$\text{Upper Bound: } k = N/2 - 1 \quad \kappa = N - 1$$

For discrete functions, $f(x)$, the Fourier coefficients can be evaluated numerically as in Equation (7.273) which can be interpreted as using Riemann sums. More notes on how this Riemann sum is an adequate approximation can be found in the handwritten notes in [242]. More details about the Fourier interpolate can also be found in [242]. In brief, given N data points, the original data is evaluated at the frequency of the principle root, W_N , as described by Equation (7.273). Every coefficient in the Fourier domain corresponds to this root raised to an integer power between $-N/2$ and $N/2-1$ (corresponding to multiples of the base frequency).

$$W_N = e^{-i\frac{2\pi}{N}} \quad (7.273)$$

7.17.1.2 DFT in Matrix, Component and Index Format

The DFT computes the vector of complex Fourier coefficients, \tilde{f}_k , as the product of vector of functional values, $f(x_j)$ with the Fourier Transformation Matrix, T as in Equation (7.274). The Fourier reconstruction can be performed using Equation (7.275).

$$\tilde{f}_k = \frac{1}{N} T f(x_j), \quad T_{kj} = W_N^{\left(k-\frac{N}{2}\right)j}, \quad \forall k = 0 \dots N-1, \quad j = 0 \dots N-1 \quad (7.274)$$

$$f(x_j) = \sum_{k=0}^{N-1} \tilde{f}_k e^{i\left(k-\frac{N}{2}\right)x_j} \quad h = \frac{2\pi}{N}, \quad x_j = jh \quad (7.275)$$

Note that the computation of \tilde{f}_k from f only involves an evaluation step but no matrix inversion. In short, this is because the periodic exponential terms in the problem definition, which are complex conjugate to one another, along with the symmetry of the matrix lends itself easily to

inversion and thus the inverted matrix can be generated explicitly as the complex conjugate transpose of the originally formulated matrix C . More on this can be found elsewhere [242]. This has the added benefit of not requiring the solving of an entire matrix in order to find a single coefficient, and the lengthy process of inverting a matrix is avoided which is advantageous for very large systems. Formulations in index form and in component form are given in Equations (7.276)-(7.277).

Index form

$$\begin{aligned}\tilde{f}_k &= \frac{1}{N} \sum_{j=0}^{N-1} f(x_j) W_N^{\left(k-\frac{N}{2}\right)j}, \quad h = \frac{2\pi}{N}, \quad x_j = jh, \quad W_N = e^{-i\frac{2\pi}{N}} \\ \tilde{f} &= \frac{1}{N} T_{kj} \cdot f, \quad T_{kj} = W_N^{\left(k-\frac{N}{2}\right)j}, \quad \forall k = 0 \dots N-1, \quad j = 0 \dots N-1\end{aligned}\quad (7.276)$$

Component Form

$$\begin{bmatrix} \tilde{f}_0 \\ \tilde{f}_1 \\ \vdots \\ \tilde{f}_{k-1} \\ \tilde{f}_k \end{bmatrix} = \frac{1}{N} \begin{bmatrix} W_N^{\left(0-\frac{N}{2}\right)0} & W_N^{\left(0-\frac{N}{2}\right)1} & \dots & W_N^{\left(0-\frac{N}{2}\right)(N-2)} & W_N^{\left(0-\frac{N}{2}\right)(N-1)} \\ W_N^{\left(1-\frac{N}{2}\right)0} & & \ddots & & W_N^{\left(1-\frac{N}{2}\right)(N-1)} \\ \vdots & & & \ddots & \vdots \\ W_N^{\left(k-1-\frac{N}{2}\right)0} & & & \ddots & W_N^{\left(k-1-\frac{N}{2}\right)(N-1)} \\ W_N^{\left(k-\frac{N}{2}\right)0} & W_N^{\left(k-\frac{N}{2}\right)1} & \dots & W_N^{\left(k-\frac{N}{2}\right)(N-2)} & W_N^{\left(k-\frac{N}{2}\right)(N-1)} \end{bmatrix} \begin{bmatrix} f(x_0) \\ f(x_1) \\ \vdots \\ f(x_{N-2}) \\ f(x_{N-1}) \end{bmatrix} \quad (7.277)$$

7.17.2 Inverse DFT (IDFT) in Matrix, Component and Index Form

The inverse discrete Fourier transform converts the complex Fourier coefficients, \tilde{f}_k , back to an analytic signal, $f(x_j)$ in the time domain. Because of the special functions (complex exponents)

and format of the transformation matrix T , its inverse C can be obtained without the need for matrix inversion. The rows in the C matrix are conjugate complex to the columns in the T matrix (not shown here). Accordingly, C is the conjugate complex of the transpose of T as in Equations (7.278)-(7.279).

$$C = T^{-1} = \overline{T^T} \quad (7.278)$$

$$C_{jk} = W_N^{-\left(k-\frac{N}{2}\right)j}, \quad T_{kj} = W_N^{\left(k-\frac{N}{2}\right)j} \quad (7.279)$$

Three different forms of the inverse DFT are given in Equations (7.280)-(7.282)

IDFT in matrix form

$$f(x_j) = C \tilde{f}_k \quad (7.280)$$

IDFT in index form

$$f(x_j) = \sum_{k=0}^{N-1} \tilde{f}_k W_N^{-\left(k-\frac{N}{2}\right)j} \quad (7.281)$$

$$f = C_{jk} \cdot \tilde{f} ; C_{jk} = W_N^{-\left(k-\frac{N}{2}\right)j} ; \forall k = 0 \dots N-1, j = 0 \dots N-1$$

IDFT in component form

$$\begin{bmatrix} f(x_0) \\ \vdots \\ f(x_j) \end{bmatrix} = \begin{bmatrix} W_N^{-\left(0-\frac{N}{2}\right)0} & \dots & W_N^{-\left(k-\frac{N}{2}\right)0} \\ \vdots & \ddots & \vdots \\ W_N^{-\left(0-\frac{N}{2}\right)(N-1)} & \dots & W_N^{-\left(k-\frac{N}{2}\right)(N-1)} \end{bmatrix} \begin{bmatrix} \tilde{f}_0 \\ \vdots \\ \tilde{f}_{N-1} \end{bmatrix} \quad (7.282)$$

7.17.2.1 Reconstruction

Once the Fourier coefficients have been computed, it is useful to create the so-called “interpolate” of the function, also referred to as a reconstruction of the original function. This can be evaluated numerically as in Equation (7.281). This interpolation formula gives the nodal values, $f(x_j)$, for the arguments x_j . In other words, the interpolation returns the same functional values at the original grid points (see proof in Appendix B in [242]).

To interpolate between grid points, special care is needed for the case with even number of samples, N , as Quarteroni requests [243]. Details on why this is necessary can be found in Trefethen papers [244,245] and more on this in Section 7.17.4. In a nutshell, the coefficient for \tilde{f}_N (for the highest frequency) is the conjugate complex for \tilde{f}_0 (the second coefficient associated with the highest frequency) but this coefficient is omitted in Equation (7.281).

To correct this omission, Equation (7.283) is used instead and both \tilde{f}_0 and \tilde{f}_N are multiplied with 0.5. This is expressed in Equation (7.283). Note that for the reconstruction, any values of j between 0 and N can be used, or in terms of x ; values between 0 and 2π .

$$f(x_j) = \sum_{k=0}^{N-1} \tilde{f}_k e^{i(k-\frac{N}{2})x_j} \quad x_j = jh \quad h = \frac{2\pi}{N}$$

$$f(\hat{x}) = \sum_{k=0}^N \hat{\tilde{f}}_k e^{i(k-\frac{N}{2})\hat{x}} \quad \hat{x} = jh = j\frac{2\pi}{N}; \quad \hat{x} \in [0..N] \quad (7.283)$$

$$\text{with } \hat{\tilde{f}}_k = \frac{c}{N} \sum_{j=0}^N f(x_j) W_N^{(k-\frac{N}{2})j} \quad c = \begin{cases} \frac{1}{2} & j = 0, N \\ 1 & j \neq 0, N \end{cases}$$

There are also three options to create a valid function for interpolate. These are mentioned briefly for completeness, implementations are given in Section 7.17.7. (i) Take only the real part of the interpolated IDFT signal (because only the dangling coefficient generates any imaginary numbers). (ii) Only calculate the IDFT signal for $k=0$ to $k=N-2$ and calculate the final term as $\cos([N-1]2\pi t/N)$ – thus ignoring the imaginary part of the exponential. (iii) Manually augment the coefficients with a complex conjugate to this dangling coefficient and multiply both the dangling coefficient and its new complex conjugate by a factor of $1/2$. This new coefficient will also have an IDFT shape function to match.

7.17.2.2 Illustration of the Principal Roots

To evaluate the Fourier coefficients using Equation (7.273), the principle roots and powers of these roots need to be evaluated. It is helpful to illustrate these operations graphically. The principle roots are defined in Equation (7.273) where N is the number of data points. The visualization of principal roots $N=1-4$ is offered in Figure 7.139.

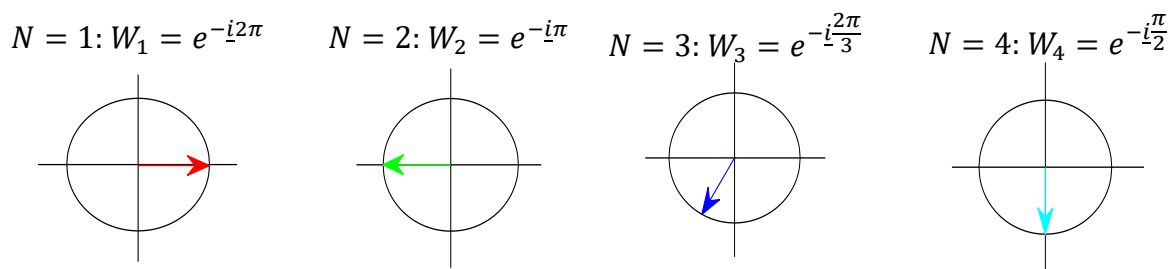


Figure 7.139. Visualization of the first 4 principle roots in the discrete Fourier series.

We can see that each root is a vector inside the unit circle rotated by $-\frac{2\pi}{N}$ radians.

Powers of the Principal Roots. This section shows that powers of the principal roots, W_N , are vectors in the unit circle. Moreover, the entries needed in the matrix T of the DFT are just powers

of these roots, $(W_N^{(k-2)})^j$, which will be shown to be mere rotations of the principal root.

Accordingly, these vectors, which are the complex entries of the T and C matrices, are relatively easy to compute as shown next. We use, W_4 , as illustrated above as in Figure 7.139, far right.

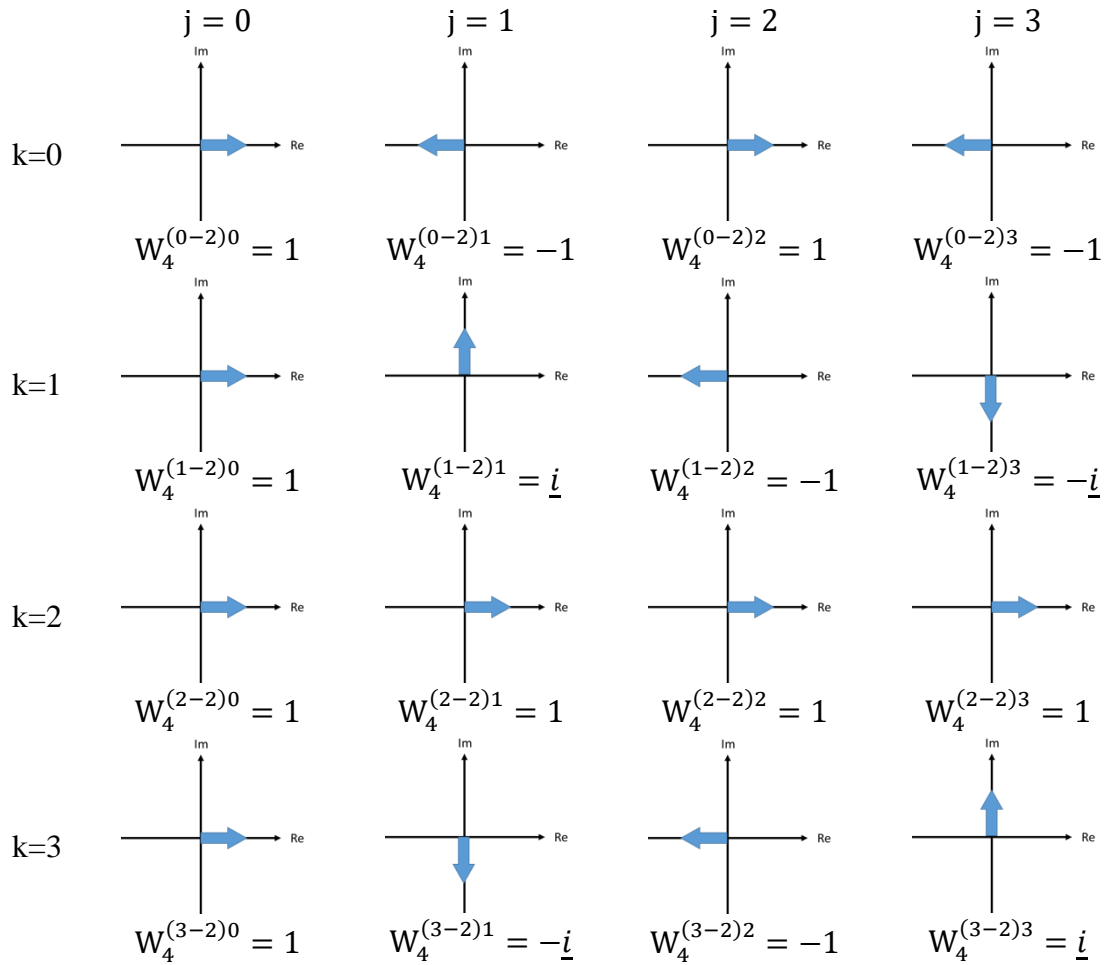


Figure 7.140. Visualization of the powers of the fourth principle root needed in DFT for data sets with $N=4$ points.

Note that each image depicts a vector equivalent to an entry in the T matrix.

Fourier transformation matrices, T and C, are constant coefficient matrices that can be computed ahead of time. Table 7.36 summarizes T and C for small datasets ($N= 2-6$) using the

Quarteroni indexing scheme[243]. Note, this scheme only works for even numbers of sample points.

Table 7.36. DFT Fourier transformation matrix T and IDFT transformation matrix C for 2, 4, and 6 data points.

	$\tilde{f}_k = \frac{1}{N} T f(x_j)$	$f(x_j) = C \tilde{f}_k$
	T	C
N=2	$\begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$	$\begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$
N=4	$\begin{bmatrix} 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \\ 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \end{bmatrix}$	$\begin{bmatrix} 1 & 1 & 1 & 1 \\ -1 & -i & 1 & i \\ 1 & -1 & 1 & -1 \\ -1 & i & 1 & -i \end{bmatrix}$
N=6	$T = \begin{bmatrix} 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -\frac{1}{2} + 0.866i & -\frac{1}{2} - 0.866i & 1 & -\frac{1}{2} + 0.866i & -\frac{1}{2} - 0.866i \\ 1 & \frac{1}{2} + 0.866i & -\frac{1}{2} + 0.866i & -1 & -\frac{1}{2} - 0.866i & \frac{1}{2} - 0.866i \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \frac{1}{2} - 0.866i & -\frac{1}{2} - 0.866i & -1 & -\frac{1}{2} + 0.866i & \frac{1}{2} + 0.866i \\ 1 & -\frac{1}{2} - 0.866i & -\frac{1}{2} + 0.866i & 1 & -\frac{1}{2} - 0.866i & -\frac{1}{2} + 0.866i \end{bmatrix}$	$C = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ -1 & -\frac{1}{2} - 0.866i & \frac{1}{2} - 0.866i & 1 & \frac{1}{2} + 0.866i & -\frac{1}{2} + 0.866i \\ 1 & -\frac{1}{2} + 0.866i & -\frac{1}{2} - 0.866i & -1 & -\frac{1}{2} + 0.866i & -\frac{1}{2} - 0.866i \\ -1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -\frac{1}{2} - 0.866i & -\frac{1}{2} + 0.866i & -1 & -\frac{1}{2} - 0.866i & -\frac{1}{2} + 0.866i \\ -1 & -\frac{1}{2} + 0.866i & \frac{1}{2} + 0.866i & 1 & \frac{1}{2} - 0.866i & -\frac{1}{2} - 0.866i \end{bmatrix}$

Graphical construction of the Fourier coefficients. The computation of the discrete Fourier series can be graphically interpreted as the rotation of the principle root (each row in the DFT matrix). These roots are all then scaled by the values of the original function, $f(x_j)$. The sum of the scaled vectors give the Fourier coefficients as shown in Figure 7.141.

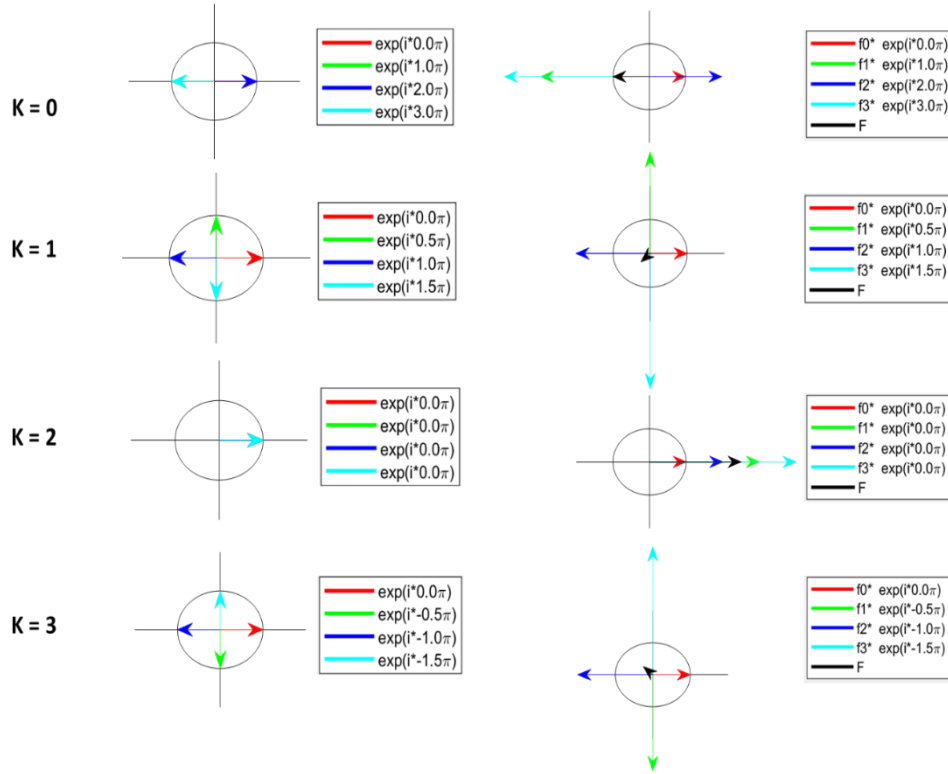


Figure 7.141. Graphical representation of the principle root being rotated (raised to different powers) and scaled by $f(x_j)$. Each row represents a different frequency. Left) The vector orientation of all vectors used for computing the coefficient. Right) the scaled form of all base vectors (scaled by the functional values) and the summation of all vectors.

7.17.3 Applications

A case study with 2 data points ($N=2$). Periodical function $f(x) = [1, 2]$ is given in Figure 7.142.

The DFT is computed shown in Equations (7.284) and (7.285). The Discrete Fourier Transform (DFT) can be computed easily:

$$\tilde{f}_k = \frac{1}{N} \sum_{j=0}^{N-1} f(x_j) W_N^{(k-\frac{N}{2})j} = \frac{1}{2} \sum_{j=0}^1 f(x_j) W_2^{(k-\frac{2}{2})j} \quad (7.284)$$

$$\begin{aligned}
\begin{bmatrix} \widehat{f}_0 \\ \widehat{f}_1 \end{bmatrix} &= \frac{1}{2} \begin{bmatrix} e^{-\frac{i2\pi}{2}(0-\frac{2}{2})0} & e^{-\frac{i2\pi}{2}(0-\frac{2}{2})1} \\ e^{-\frac{i2\pi}{2}(1-\frac{2}{2})0} & e^{-\frac{i2\pi}{2}(1-\frac{2}{2})1} \end{bmatrix} \begin{bmatrix} f(x_0) \\ f(x_1) \end{bmatrix} \\
&= \frac{1}{2} \begin{bmatrix} 1 & e^{-i\pi} \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} -1 \\ 3 \end{bmatrix}
\end{aligned} \tag{7.285}$$

Inverse Discrete Fourier Transform (IDFT): The IDFT can be computed as shown in equation (7.286).

$$\begin{aligned}
f &= C_{jk} \cdot \tilde{f} \ ; \ C_{jk} = \begin{bmatrix} e^{-(-\frac{i2\pi}{2})(0-\frac{2}{2})0} & e^{-(-\frac{i2\pi}{2})(1-\frac{2}{2})0} \\ e^{-(-\frac{i2\pi}{2})(0-\frac{2}{2})1} & e^{-(-\frac{i2\pi}{2})(1-\frac{2}{2})1} \end{bmatrix} = \overline{T^T} = T^{-1} \\
\begin{bmatrix} f(x_0) \\ f(x_1) \end{bmatrix} &= \begin{bmatrix} e^{-(-\frac{i2\pi}{2})(0-\frac{2}{2})0} & e^{-(-\frac{i2\pi}{2})(1-\frac{2}{2})0} \\ e^{-(-\frac{i2\pi}{2})(0-\frac{2}{2})1} & e^{-(-\frac{i2\pi}{2})(1-\frac{2}{2})1} \end{bmatrix} \begin{bmatrix} \widehat{f}_0 \\ \widehat{f}_1 \end{bmatrix} \\
&= \begin{bmatrix} 1 & 1 \\ e^{i\pi} & 1 \end{bmatrix} \frac{1}{2} \begin{bmatrix} -1 \\ 3 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} -1 \\ 3 \end{bmatrix} = f
\end{aligned} \tag{7.286}$$

Reconstruction: The reconstruction process calculates the shape functions between the sampling intervals using Equation (7.287). The result can be seen in Figure 7.142.

$$f(x) = \sum_{k=0}^2 \widehat{f}_k e^{i(k-\frac{N}{2})x} = \begin{bmatrix} e^{i(-1)x} & e^{i0} & e^{i(1)x} \end{bmatrix} \begin{bmatrix} \widehat{f}_0 \\ \widehat{f}_1 \\ \widehat{f}_2 \end{bmatrix} \tag{7.287}$$

$$\text{with } \widehat{\widehat{f}}_k = \begin{bmatrix} \frac{1}{2} \widehat{f}_0 & \widehat{f}_1 & \frac{1}{2} \widehat{\widehat{f}}_0 \end{bmatrix} = \begin{bmatrix} -\frac{1}{4} & \frac{3}{2} & -\frac{1}{4} \end{bmatrix} \text{ and } \widehat{x} = \frac{2\pi}{N} = \pi x$$

$$f(\widehat{x}) = \frac{3}{2} - \frac{1}{2} \cos(x) = \frac{3}{2} - \frac{1}{2} \cos(\pi \widehat{x}) \quad \widehat{x} \in [0..2]$$

The highest frequency Fourier coefficients ($k=0, 2$) had to be scaled as indicate in Equation (7.287).

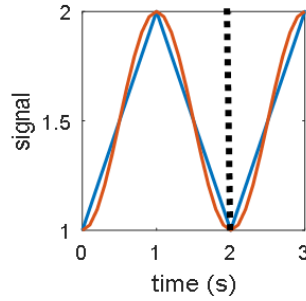


Figure 7.142. (Blue) Original signal and (orange) reconstruction using discrete Fourier series and inverse discrete Fourier series.

7.17.3.1 Reconstruction and noise removal for a heartbeat signal

During the sampling of a periodic signal in experimentation, noise is often generated by environmental factors. Such a noise-endowed waveform is shown in Figure 7.143. The initial signal contains 84 data points. The full Fourier approximation using 84 coefficients represents the noisy signal exactly.

To eliminate noise, a filter can be developed in the Fourier domain to reduce the magnitude of the coefficients corresponding to the noise. The first step is to compute the Fourier coefficients. Note that the DFT requires as many coefficients as the original data, no data reduction, curve fitting or smoothing. Based on physiological values (such as knowledge that the heartbeat is 60-120 beats

per minute, BPM), a filter can be made that suppresses the non-heartbeat portion of the signal(<60 and >120 BPM) by setting the coefficients (\tilde{f}_k) associated with those frequencies to 0.

After dropping the high frequency coefficients, the filtered signal can be up-sampled and plotted as in Figure 7.143 by the orange curve. Table 7.37 shows the Fourier coefficients and their magnitude. A period of 1 second, not N seconds, was used. To account for this, a shift of the x variable from $0..2\pi$ to $0..1$ (or more generally speaking, $0..T$ where T is the sampling period) was implemented. The implementation is shown in Section 7.17.7. The modified DFT and IDFT to account for the new period is offered in Equation (7.288).

$$\tilde{f}_k = \frac{1}{N} \sum_{j=0}^{N-1} f(x_j) e^{-i \frac{2\pi}{T} (k - \frac{N}{2}) j}$$

$$f(x) = \frac{1}{N} \sum_{k=0}^{N-1} \tilde{f}_k e^{-i \frac{2\pi}{T} (k - \frac{N}{2}) x}$$
(7.288)

Table 7.37. Fourier polynomial and its coefficients to reconstruct signal without noise using Quateroni indexing (N=84).

φ_i	\hat{f}_0	\hat{f}_1	\hat{f}_2	\hat{f}_3	\hat{f}_4	\hat{f}_5	\hat{f}_6
ω	$3(2\pi/84)x$	$2(2\pi/84)x$	$1(2\pi/84)x$	0	$-1(2\pi/84)x$	$-2(2\pi/84)x$	$-3(2\pi/84)x$
C_i	$-.0319+.018i$	$-.0089+.0482i$	$-.027+.1043i$.7199	$.027-.1043i$	$-.0089-.0482i$	$-.0319-.018i$

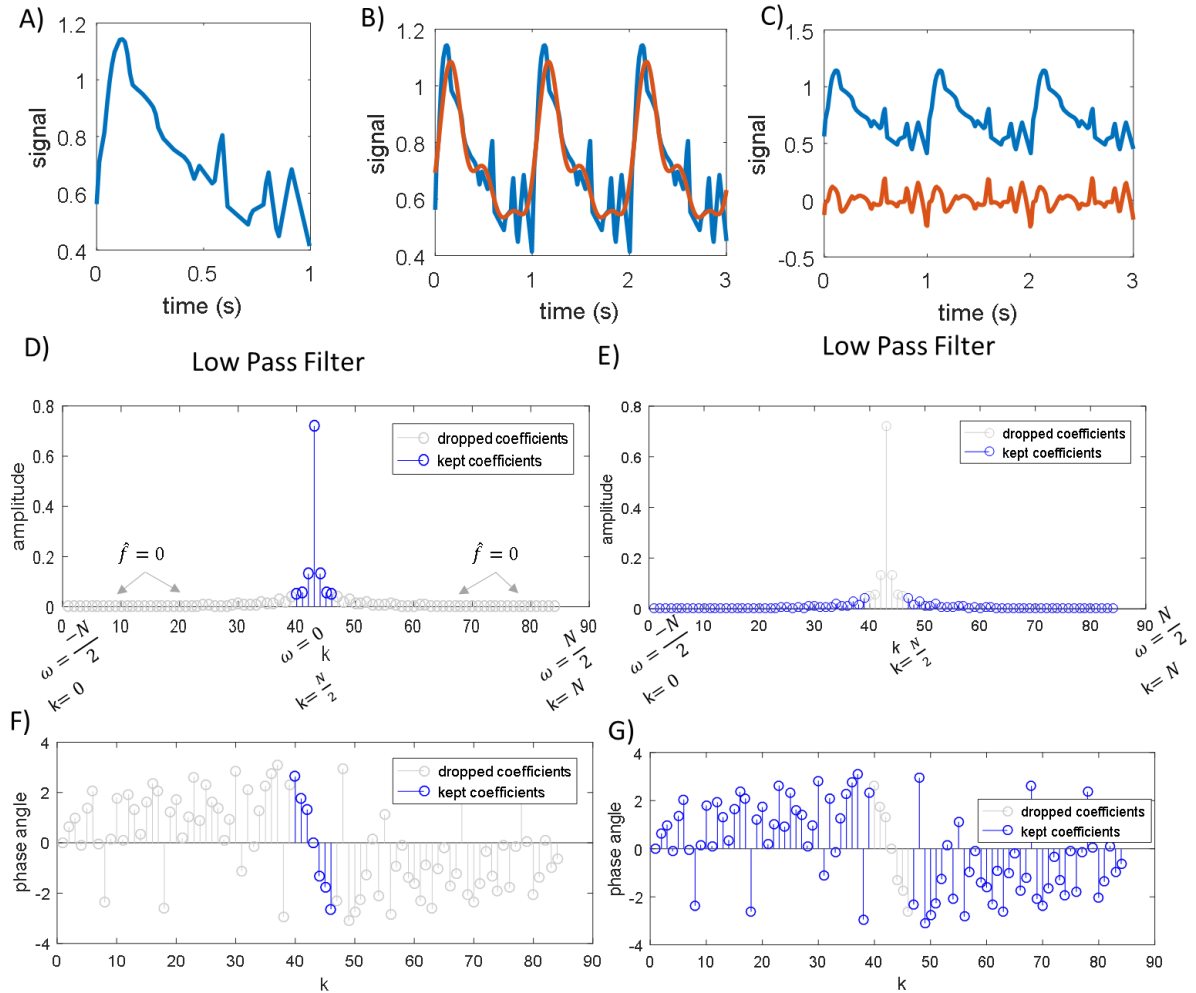


Figure 7.143. A) Noisy periodic signal. B) Blue curve is constructed by using all coefficients in IDFT with a low-pass filter applied.

Orange curve is constructed by keeping the low frequency coefficients (blue in C) and filtering the high frequencies (grey in C). C) Reconstructed signal (blue) and a reconstruction after applying a high-pass filter (orange) by setting the coefficients of the lower frequencies to 0. Amplitude of the Fourier coefficients in the Frequency spectrum for the low-pass (D) and high-pass (E) filters. Phase spectrum of the signal after applying a low-pass (F) and high-pass (G) filter. The filtering allows us to remove the noise in the signal as shown by the green curve.

The reconstructed heartbeat signal then becomes:

$$f(x) = (-.0319 + .018i)e^{i\frac{3\pi}{42}x} + (-.0089 + .0482i)e^{i\frac{2\pi}{42}x} + (-.027 + .1043i)e^{i\frac{\pi}{42}x} + (.7199)1$$

Imag

$$+ (-.027 - .1043i)e^{-i\frac{\pi}{42}x} + (-.0089 - .0482i)e^{-i\frac{2\pi}{42}x} + (-.0319 - .018i)e^{-i\frac{3\pi}{42}x}$$

$$.027 \cos\left(-\frac{\pi}{42}x\right) - .1043 \sin\left(-\frac{\pi}{42}x\right) - .0089 \cos\left(-\frac{2\pi}{42}x\right) - .0482 \sin\left(-\frac{2\pi}{42}x\right)$$

$$- .0319 \cos\left(-\frac{3\pi}{42}x\right) - .018 \sin\left(-\frac{3\pi}{42}x\right)$$

Real

Which simplifies to:

$$f(x) = -.0638 \cos\left(\frac{3\pi}{42}x\right) + .036 \sin\left(\frac{3\pi}{42}x\right) - .0178 \cos\left(\frac{2\pi}{42}x\right) + .0964 \sin\left(\frac{2\pi}{42}x\right)$$

$$- .054 \cos\left(\frac{\pi}{42}x\right) + .2086 \sin\left(\frac{\pi}{42}x\right) + .7199$$

Functional values written in Matlab script:

```
Time = 0:0.002:0.166;
FunctionalValues
[0.561953033988024,0.710124125433452,0.763805541362655,0.813625600359705,0.907392562011622,0.991953033988024,1.0552273702
7121,1.09836601333906,1.12001793074319,1.14060790124466,1.14295598384053,1.13402088059569,1.09097958266059,1.022012031038
17,0.982436809799233,0.973501706554395,0.965091677055870,0.957433859946726,0.948926485315457,0.938714095934926,0.92817722
2778584,0.916687547262360,0.903628550212212,0.879020880595694,0.831032680005723,0.811879287675339,0.792902886495398,0.783
329140182714,0.773755393870030,0.764181647557345,0.754607901244661,0.747784892395103,0.741404361421652,0.735023830448201
,0.728643299474749,0.717764243427552,0.704997281775634,0.677840939592744,0.651073977940826,0.673988432218112,0.6966236337
91367,0.684282434184681,0.671941234577994,0.659600034971308,0.647258835364622,0.634917635757935,0.643345364371505,0.70721
5570861180,0.771085777350856,0.804914685905428,0.674637399769735,0.553680910094218,0.545703771451151,0.537726632808083,0.
529749494165015,0.521772355521947,0.513795216878879,0.505818078235811,0.497840939592744,0.489863800949676,0.523014980890
678,0.539457458766785,0.544572503014572,0.549687547262360,0.554802591510148,0.559917635757935,0.634979582660590,0.6736565
73811033,0.607882237527847,0.542107901244661,0.476333564961475,0.448650379120767,0.498214980890679,0.547779582660589,0.59
7344184430501,0.646908786200412,0.684653413254775,0.645975789993081,0.607298166731387,0.568620543469693,0.52994292020799
8,0.491265296946304,0.452587673684610,0.413910050422916];
```

7.17.4 Special topic: different indexing scheme of the DFT with implementation

When using different indexing schemes, the mathematical form of the discrete Fourier Series differs. Specifically, the index of 0 as in Quarteroni is given in Equation (7.289), while MATLAB index begins at an index of 1 as in Equation (7.290). Moreover, the Matlab model for powering

the principle root differs from that of Quarteroni, causing a reordering of the coefficients. The Fourier coefficients are only comparable after an indexing shift and reordering. Simply stated, a reordering is necessary for the coefficient of $k=1$ in the Matlab implementation to match the coefficient for $k=1$ in the Quarteroni indexing scheme.

This is also relevant for matching Fourier coefficients with their respective shape functions, as is used when up-sampling the data. The following section discusses the details of the Quarteroni and MATLAB indexing and their implications on such reconstructions. Trefethen [244,245] has yet another indexing scheme that is similar to Matlab but shifted by an index of 1 as seen in Figure 13.1.

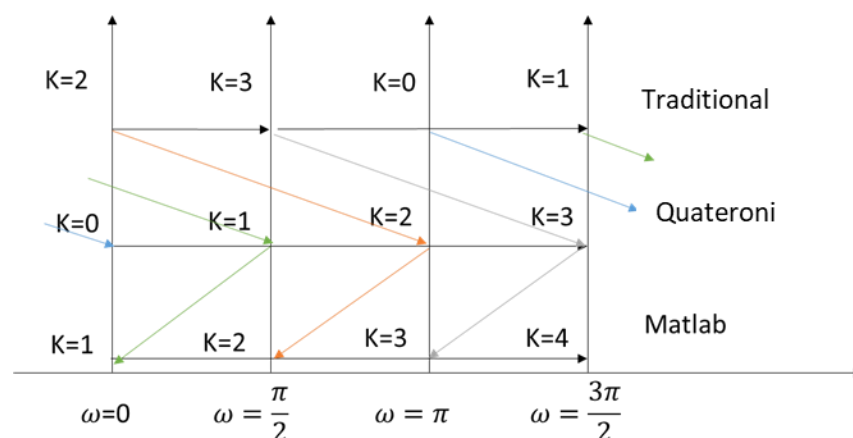


Figure 7.144. Comparison of indexing schemes between 3 different DFT indexing paradigms. (Top) Traditional indexing runs from $-N/2$ to $N/2 - 1$ whereas Quarteroni [243] indexes in a programmatically convenient way ($0..N-1$). Matlab/Trefethen use yet a third method for indexing over the range $1..N$. (Bottom) The three methods calculate the same complex vectors as even though the indexing is very different.

Another comparison of the indexing schemes are offered below with code given in Section 7.17.9. This comparison exemplifies the rearrangement of the same coefficients using the two different methods.

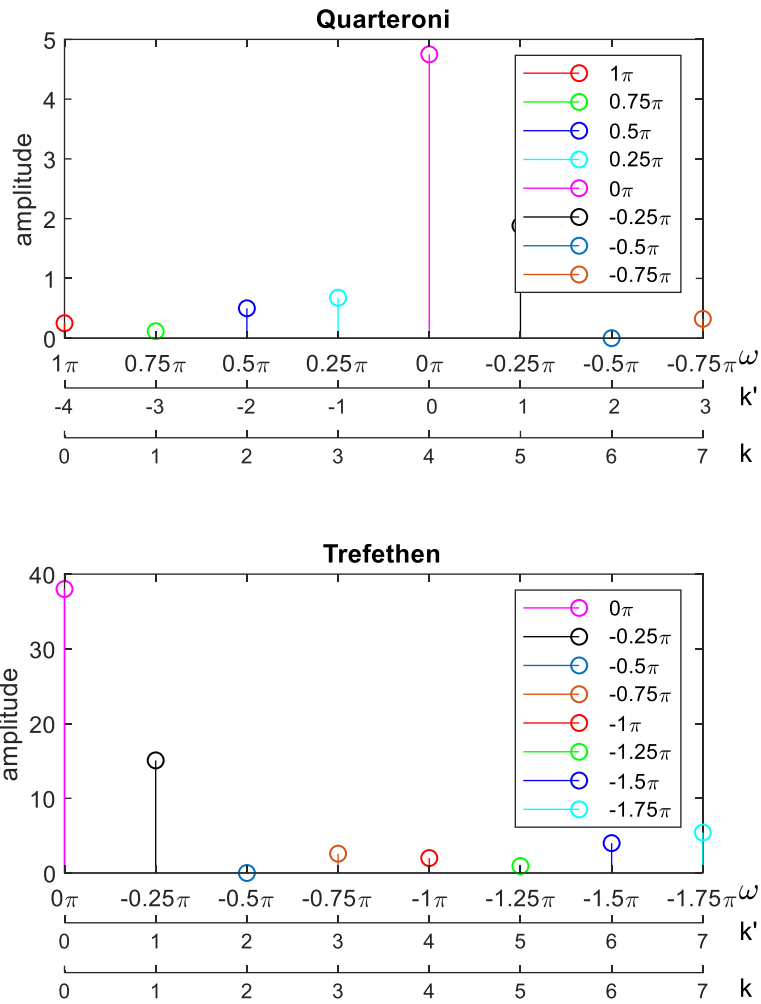


Figure 7.145. Comparison of indexing schemes between two different DFT indexing paradigms.

(Top) Quarteroni [243] indexes in a programmatically convenient way (0..N-1) while (Bottom) Trefethen indexing runs from $-N/2$ to $N/2 - 1$.

7.17.4.1 Quarteroni implementation

$$\tilde{f}_k = \frac{1}{N} \sum_{j=0}^{N-1} f(x_j) W_N^{\left(k-\frac{N}{2}\right)j}; \quad f(x_j) = \sum_{k=0}^{N-1} \tilde{f}_k W_N^{-\left(k-\frac{N}{2}\right)j};$$

(7.289)

$$h = \frac{2\pi}{N}, \quad x_j = jh, \quad W_N = e^{-i\frac{2\pi}{N}}$$

Quarteroni provided implementations of the DFT and IDFT given below (extracted from [246]):

```

Program 88 (Quarteroni) – dft
function fx = dft(N,f)
h = 2*pi/N; x = [0:h:2*pi*(1-1/N)]; fx = eval(f); wn = exp(-i*h);
for k = 0:N-1
s = 0;
for j = 0:N-1
s = s+fx(j+1)*wn^((k-N/2)*j);
end
fx(k+1) = s/N;
end
Program 89 (Quarteroni) – idft
function fv = idft(N,fc)
h = 2*pi/N; wn = exp(-i*h);
for k = 0:N-1
s = 0;
for j = 0:N-1
s = s+fc(j+1)*wn^((j-N/2)*-k);
end
fv(k+1) = s;
end

```

7.17.4.2 Matlab/Trefethen implementation

Matlab indexing looks the same as Quarteroni on the face, yet it differs in a key, namely that the power of the weight function begins at 0 and approaches N-1. Whereas the coefficient k and j

in the Quarteroni scheme follows this pattern, the power of the principle root actually goes from $-N/2$ to $N/2$; an important distinction visited later.

Matlab:

$$X(k) = \sum_{j=1}^N x(j) W_N^{(j-1)(k-1)}, \quad x(j) = \frac{1}{N} \sum_{k=1}^N X(k) W_N^{-(j-1)(k-1)}, \quad (7.290)$$

$$W_N = e^{(-2\pi i)/N}$$

Trefethen:

$$X(k) = \sum_{j=0}^{N-1} x(j) W_N^{j k}, \quad x(j) = \frac{1}{N} \sum_{k=1}^N X(k) W_N^{-j k}, \quad W_N = e^{(-2\pi i)/N} \quad (7.291)$$

7.17.4.3 Conventional implementation

The conventional implementation gives the same result as the Quarteroni method, yet the $N/2$ shift occurs in the coefficient itself, as opposed to being imposed in the power as it is in Quarteroni.

$$X(k) = \sum_{j=-\frac{N}{2}}^{\frac{N}{2}-1} x(j) W_N^{j k}, \quad x(j) = \frac{1}{N} \sum_{k=1}^N X(k) W_N^{-j k}, \quad W_N = e^{-i k x} \quad (7.292)$$

MATLAB added an $N/2$ shift and scaled the coefficients by N . In order to account for this scaling, Matlab employs a $1/N$ scaling when performing the ifft as shown below. This results in a

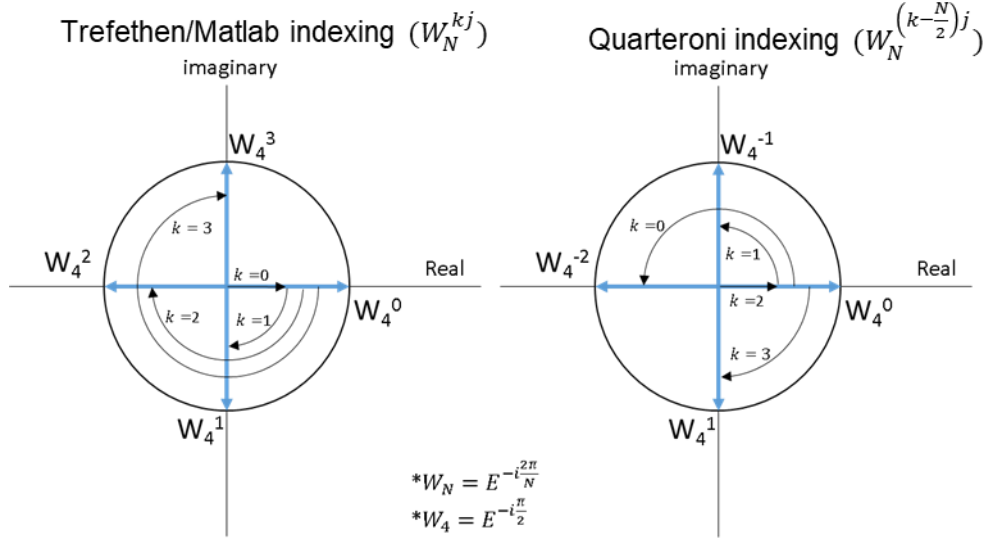
scaling and shifting difference compared to dft and idft. The corresponding shape functions are therefore also shifted by $N/2$ in the frequency domain. For even N , this results in a shift $N/2$ in the coefficients and functions. However, for odd N , this results in completely different shape functions and coefficients.

Table 7.38. Comparison of indexing schemes. Quarteroni ($k-N/2$), Trefethen (k), and Matlab ($k-1$), $N=4$

	\hat{f}_k	k	W^*	\hat{f}_k	k	W^*	\hat{f}_k	k	W^*	\hat{f}_k	k	W^*
Quarteroni	2.5	2	0	$0.25+0.25\underline{i}$	3	1	$\underline{1}$	0	-2	$0.25-0.25\underline{i}$	1	-1
Trefethen	10	0	0	$1+\underline{i}$	1	1	$\underline{4}$	2	2	$1-\underline{i}$	3	3
MATLAB	10	1	0	$1+\underline{i}$	2	1	$\underline{4}$	3	2	$1-\underline{i}$	4	3
Frequency	$\cos 0 + \underline{i} \sin 0$			$\cos \pi/2 + \underline{i} \sin \pi/2$			$\cos \pi + \underline{i} \sin \pi$			$\cos \frac{3\pi}{2} + \underline{i} \sin \frac{3\pi}{2}$		

W^* - power of the root (W^P)

In order to interpolate using the Trefethen/Matlab indexing scheme, the coefficients and base functions need to be un-scrambled as visualize in Figure 7.146. Figure 7.146 displays the difference between interpolating using the IDFT formula directly using the weight functions calculated with the Trefethen method compared to the Quarteroni method. It can be clearly seen that the Matlab indexing does not hold the pairing of the complex conjugates between principle roots.



Interpolating at $t=0.1$

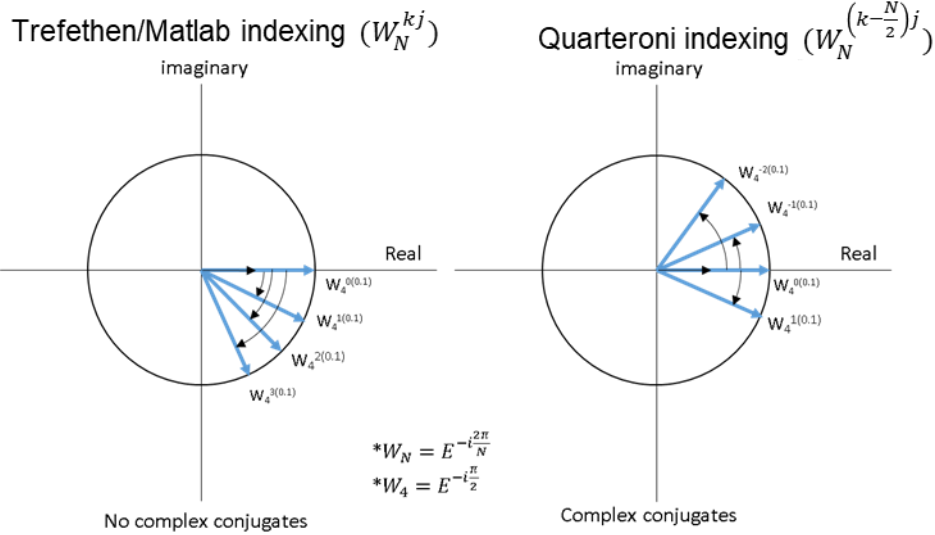


Figure 7.146. Graphical representation of interpolation vectors calculated with indexing schemes of Trefethen (Left) and Quarteroni (Right).

The Trefethen method does not properly reconstruct the formula, as the shape functions are no longer complex conjugate while interpolating. To fix this, the proper shape functions must be assigned to the proper coefficients as explained next.

For the Trefethen method using an odd number of coefficients, the shape function associated with each coefficient must be calculated independent of the IDFT formula. Given the knowledge that the values have a complex conjugate associated with them, the computation of the pair of

complex conjugates can be accomplished using a specified weight function (W) and the Trefethen shape functions (\hat{f}):

$$f(t) = W_N^0 + \sum_{k=1}^{\frac{N-1}{2}} [\hat{f}_k W_N^{-k t} + \hat{f}_{N-k} W_N^{k t}] \quad (7.293)$$

Where: $W_N = e^{-i \frac{2\pi}{N}}$ and $x = 0..N$

In order to account for the “dangling” coefficient when using an even number of coefficients, the final term can be calculated independently. This gives a new formulation of the series in one of two equivalent ways:

$$\begin{aligned} f(t) &= W_N^0 + \sum_{k=1}^{\frac{N}{2}-1} [\hat{f}_k W_N^{-k t} + \hat{f}_{N-k} W_N^{k t}] + \left[\hat{f}_{\frac{N}{2}} W_N^{-\frac{N}{2} t} + \hat{f}_{\frac{N}{2}} W_N^{\frac{N}{2} t} \right] \\ &= W_N^0 + \sum_{k=1}^{\frac{N}{2}} [\hat{f}_k W_N^{-k t} + \hat{f}_{N-k} W_N^{k t}] \end{aligned} \quad (7.294)$$

Or

$$\begin{aligned} f(t) &= W_N^0 + \sum_{k=1}^{\frac{N}{2}-1} [\hat{f}_k W_N^{-k t} + \hat{f}_{N-k} W_N^{k t}] + \left[\hat{f}_{\frac{N}{2}} \cos\left(\frac{2\pi}{N} \frac{N}{2} t\right) \right] \\ &= W_N^0 + \sum_{k=1}^{\frac{N}{2}-1} [\hat{f}_k W_N^{-k t} + \hat{f}_{N-k} W_N^{k t}] + \left[\hat{f}_{\frac{N}{2}} \cos(\pi t) \right] \end{aligned} \quad (7.295)$$

Where: $W_N = e^{-i\frac{2\pi}{N}}$ and $x = 0..N$

Principle root matrices for common small datasets (N= 2-6) using the *Trefethen* numbering scheme (this scheme can accommodate odd numbers of data points) is offered. Note, the T and C matrices are symmetric.

Table 7.39. DFT Fourier transformation matrix T and IDFT transformation matrix C for 2, 4, and 6 data points using Trefethen indexing.

	$\tilde{f}_k = \frac{1}{N} T f(x_j)$	$f(x_j) = C \tilde{f}_k$
	T	C
N=2	$\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$	$\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
N=3	$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -\frac{1}{2} - 0.866i & -\frac{1}{2} + 0.866i \\ 1 & -\frac{1}{2} + 0.866i & -\frac{1}{2} - 0.866i \end{bmatrix}$	$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -\frac{1}{2} + 0.866i & -\frac{1}{2} - 0.866i \\ 1 & -\frac{1}{2} - 0.866i & -\frac{1}{2} + 0.866i \end{bmatrix}$
N=4	$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{bmatrix}$	$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{bmatrix}$
N=5	$T = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 0.309 - 0.9511i & -0.809 - 0.5878i & -0.809 + 0.5878i & 0.309 + 0.9511i \\ 1 & -0.809 - 0.5878i & 0.309 + 0.9511i & 0.309 - 0.9511i & -0.809 + 0.5878i \\ 1 & -0.809 + 0.5878i & 0.309 - 0.9511i & 0.309 + 0.9511i & -0.809 - 0.5878i \\ 1 & 0.309 + 0.9511i & -0.809 + 0.5878i & -0.809 - 0.5878i & 0.309 - 0.9511i \end{bmatrix}$	$C = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 0.309 + 0.9511i & -0.809 + 0.5878i & -0.809 - 0.5878i & 0.309 - 0.9511i \\ 1 & -0.809 + 0.5878i & 0.309 - 0.9511i & 0.309 + 0.9511i & -0.809 - 0.5878i \\ 1 & -0.809 - 0.5878i & 0.309 + 0.9511i & 0.309 - 0.9511i & -0.809 + 0.5878i \\ 1 & 0.309 - 0.9511i & -0.809 - 0.5878i & -0.809 + 0.5878i & 0.309 + 0.9511i \end{bmatrix}$
N=6	$T = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \frac{1}{2} - 0.866i & -\frac{1}{2} - 0.866i & -1 & -\frac{1}{2} + 0.866i & \frac{1}{2} + 0.866i \\ 1 & -\frac{1}{2} - 0.866i & -\frac{1}{2} + 0.866i & 1 & -\frac{1}{2} - 0.866i & -\frac{1}{2} + 0.866i \\ 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -\frac{1}{2} + 0.866i & -\frac{1}{2} - 0.866i & 1 & -\frac{1}{2} + 0.866i & -\frac{1}{2} - 0.866i \\ 1 & \frac{1}{2} + 0.866i & -\frac{1}{2} + 0.866i & -1 & -\frac{1}{2} - 0.866i & \frac{1}{2} - 0.866i \end{bmatrix}$	$C = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \frac{1}{2} + 0.866i & -\frac{1}{2} + 0.866i & -1 & -\frac{1}{2} - 0.866i & \frac{1}{2} - 0.866i \\ 1 & -\frac{1}{2} + 0.866i & -\frac{1}{2} - 0.866i & 1 & -\frac{1}{2} + 0.866i & -\frac{1}{2} - 0.866i \\ 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -\frac{1}{2} - 0.866i & -\frac{1}{2} + 0.866i & 1 & -\frac{1}{2} - 0.866i & -\frac{1}{2} + 0.866i \\ 1 & \frac{1}{2} - 0.866i & -\frac{1}{2} - 0.866i & -1 & -\frac{1}{2} + 0.866i & \frac{1}{2} + 0.866i \end{bmatrix}$

7.17.5 Discrete Fourier Transform (DFT) – real arithmetic

Given the definition in Equation (7.272), the Fourier series can be re-written using sines and cosines as in Equation (7.296). The +k values of the series can be combined with the -k values of the series as seen in Equation (7.297). This new form of the equation can be simplified using the

definitions in Equation (7.298) to simplify Equation (7.296) to Equation (7.299). Note, here an odd number of data points is assumed (N is odd).

$$\tilde{f}_k = \frac{1}{N} \sum_{j=-n}^n f(x_j) e^{ik \frac{2\pi}{N} j} \quad (7.296)$$

$$= \frac{1}{N} \sum_{j=-n}^n f(x_j) \left[\cos\left(k \frac{2\pi}{N} j\right) + i \sin\left(k \frac{2\pi}{N} j\right) \right]$$

$$\tilde{f}_k = \frac{1}{N} \sum_{j=-n}^n f(x_j) e^{ik \frac{2\pi}{N} j} \quad (7.297)$$

$$= \frac{1}{N} \sum_{j=0}^n f(x_j) \left[e^{-ik \frac{2\pi}{N} j} + e^{ik \frac{2\pi}{N} j} \right]$$

$$\cos\left(k \frac{2\pi}{N} j\right) = \cos\left(-k \frac{2\pi}{N} j\right) \quad \sin\left(k \frac{2\pi}{N} j\right) = -\sin\left(-k \frac{2\pi}{N} j\right) \quad (7.298)$$

$$\tilde{f}_k = \frac{1}{N} \sum_{j=0}^{\frac{N-1}{2}} f(x_j) \left[\cos\left(k \frac{2\pi}{N} j\right) + i \sin\left(k \frac{2\pi}{N} j\right) + \cos\left(-k \frac{2\pi}{N} j\right) - \left\{ -i \sin\left(-k \frac{2\pi}{N} j\right) \right\} \right] \quad (7.299)$$

$$\tilde{f}_k = \frac{1}{N} \sum_{j=0}^{\frac{N-1}{2}} f\left(\frac{2\pi}{N} j\right) 2 \left[\cos\left(k \frac{2\pi}{N} j\right) + i \sin\left(k \frac{2\pi}{N} j\right) \right]$$

The coefficient 2 can be brought to the outside of the summation and it can be broken into the real part and the imaginary part to individually derive the coefficients as in Equation (7.300). Note, $\sin(0)$ is 0 so b_0 is always 0 and $\cos(0)$ is always 1, so a_0 is always be the average value of the data. The linear average can be computed independently and b_0 can be entirely omitted, so the summation can begin at 1 instead of 0 as reflected in Equation (7.300). This is a common expression of the Fourier series in the real space that exists in some signal analysis books.

$$\tilde{f}_k = a_k + i b_k$$

$$\tilde{f}_k = a_0 + \sum_{j=1}^n \left[a_k \cos \left(k \frac{2\pi}{N} j \right) + b_k \sin \left(k \frac{2\pi}{N} j \right) \right]$$

$$\tilde{f}_k = a_0 + \frac{2}{N} \sum_{j=1}^n f(x_j) \left[\cos \left(k \frac{2\pi}{N} j \right) \right] + \frac{2}{N} \sum_{j=1}^n f(x_j) \left[i \sin \left(k \frac{2\pi}{N} j \right) \right] \quad (7.300)$$

$$a_k = a_0 + \frac{2}{N} \sum_{k=1}^n f(x_j) \cos \left(k \frac{2\pi}{N} j \right) \quad b_k = \frac{2}{N} \sum_{k=1}^n f(x_j) \sin \left(k \frac{2\pi}{N} j \right)$$

$$n = \begin{cases} \frac{N-1}{2} & N \text{ is odd} \\ \frac{N}{2} - 1 & N \text{ is even} \end{cases}$$

The IDFT can now be written as in Equation (7.301). It is important to notice that for an even number of data points, the last coefficient in the series does not have a complex conjugate. In order to account for this, Equation (7.302) is available. Equation (7.302) is equivalent to Equation (7.303) and accounts for the final unmatched coefficient.

N is odd:

$$f(x) = \sum_{k=0}^{\frac{N-1}{2}} \tilde{f}_k \left[a_k \cos \left(k \frac{2\pi x}{N} \right) + b_k \sin \left(k \frac{2\pi x}{N} \right) \right] \quad (7.301)$$

N is even:

$$f(x) = \sum_{k=0}^{\frac{N}{2}-1} \tilde{f}_k \left[a_k \cos\left(k \frac{2\pi x}{N}\right) + b_k \sin\left(k \frac{2\pi x}{N}\right) \right] + \frac{1}{2} a_{\frac{N}{2}} \cos\left(\frac{N}{2} \frac{2\pi x}{N}\right) \quad (7.302)$$

$$f(x) = \sum_{k=0}^{\frac{N}{2}-1} \tilde{f}_k \left[a_k \cos\left(k \frac{2\pi x}{N}\right) + b_k \sin\left(k \frac{2\pi x}{N}\right) \right] + \frac{1}{2} a_{\frac{N}{2}} \cos(\pi x)$$

$$f(x) = \sum_{k=0}^{\frac{N}{2}-1} \tilde{f}_k \left[a_k \cos\left(k \frac{2\pi x}{N}\right) + b_k \sin\left(k \frac{2\pi x}{N}\right) \right] + \frac{1}{2} a_{\frac{N}{2}} \cos\left(\frac{N}{2} \frac{2\pi x}{N}\right) \quad (7.303)$$

$$f(x) = \sum_{k=0}^{\frac{N}{2}-1} \tilde{f}_k \left[a_k \cos\left(k \frac{2\pi x}{N}\right) + b_k \sin\left(k \frac{2\pi x}{N}\right) \right] + \frac{1}{2} a_{\frac{N}{2}} \cos(\pi x)$$

Matrix representation of the real space Fourier series is expressed below (note, for convenience the coefficient of $2/N$ in the DFT, Equation (7.300), and the $1/2$ coefficient in the last term in Equations (7.302) and (7.303) are already factored into the T matrix).

Table 7.40. Coefficient C and T matrices for the DFT and IDFT using real algebra

	$\tilde{f}_k = T f(x_j)$	$f(x_j) = C \tilde{f}_k$
N=2	$T = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$	$C = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$
N=4	$T = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 0 & -1 & 0 \\ 1 & -1 & 1 & -1 \\ 0 & 1 & 0 & -1 \end{bmatrix}$	$C = \begin{bmatrix} 1 & 0 & -1 & 1 \\ 1 & 0 & -1 & 0 \\ 1 & -1 & 1 & 0 \\ 1 & 0 & -1 & -1 \end{bmatrix}$
N=6	$T = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 & -2 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & 1 & -1 \\ 0 & 1.7321 & 1.7321 & 0 & -1.7321 & -1.7321 \\ 0 & 1.7321 & -1.7321 & 0 & 1.7321 & -1.7321 \end{bmatrix}$	$C = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & -1 & -1 & 1.7321 & 1.7321 \\ 1 & -1 & -1 & 1 & 1.7321 & -1.7321 \\ 1 & -2 & 2 & -1 & 0 & 0 \\ 1 & -1 & -1 & 1 & -1.7321 & 1.7321 \\ 1 & 1 & -1 & -1 & -1.7321 & -1.7321 \end{bmatrix}$

7.17.6 Matlab implementation of DFT and iDFT

This section has a code written in Matlab “object-orientation” with useful tools for deriving the DFT and iDFT procedures.

```

% a program that calculates the DFT and IDFT variables given an input of data
% Written by G Hartung 4/25/2019
% ***** To reconstruct function for any timpoint, use iDFTVect^(newTimePoint)*fk *****
% need to add methods utilizing matlab fft
classdef ooDFT_lean
    methods (Static)
        function [T] = getQDFTMatrix(time) %quateroni DFT
            N = length(time);      T = zeros(N,N);
            for iRow = 0:N-1
                for iCol = 0:N-1
                    T(iRow+1,iCol+1) = exp(-1i*2*pi/N*(iRow-N/2)*iCol); %N should be replaced by T for NDFT in the future
                end
            end
        end

        function [fk] = QDFT(time,f) %quateroni DFT to get coefficients
            N = length(time);      delT = time(2)-time(1);      T = zeros(N,N);
            for iRow = 0:obj.N-1
                for iCol = 0:obj.N-1
                    T(iRow+1,iCol+1) = exp(-1i*2*pi/N*(iRow-N/2)*iCol); %N should be replaced by T for NDFT in the future
                end
            end
            fk = 1/N*T*f;
        end

        function [fk] = MatlabFFT(f) %quateroni DFT to get coefficients
            fk = fft(f);
        end

        function [fk] = unscrambleMatlabFFT(f) %quateroni DFT to get coefficients
            fk = fft(f);
            nHalf = floor(length(fk)/2);
            fk = [fk(nHalf+1:length(fk)), fk(1:nHalf)];
        end

% %      this code is hard to read and structured improperly, however
% Matlab does not evaluate exponents properly, so it must be written this
% way to work
        function [f] = evalAt(newTimeVect,time,fk,delT) %used for interpolation
            N = length(fk);      iDFTVect = zeros(1,N); %might offer alpha here in the future
            for iCol = 0:N-1
                iDFTVect(iCol+1) = exp(1i*2*pi/N*(iCol-N/2));
            end

            nTimepoints = length(newTimeVect); f = zeros(nTimepoints,size(fk,2));
            for iTime = 1:nTimepoints
                f(iTime,:) = iDFTVect.^((newTimeVect(iTime)-time(1))/delT)*fk;
            end
        end

        function [f] = computeIDFT(N,fHat)
            iDFTMx = computeIDFTMx(N);
            nTimepoints = length(newTimeVect);
            f = iDFTMx*fHat;
        end

        function [fHatMx] = computeDFTMatrix(N) %j stands for time vector
            for iRow = 0:N-1
                for iCol = 0:N-1
                    fHatMx(iRow+1,iCol+1) = exp(-1i*2*pi/N*(iRow-N/2)*iCol); %N should be replaced by T for NDFT in the future
                end
            end
        end

        function [iDFTMx] = computeIDFTMx(N)
            dftMx = computeDFTMatrix(N); iDFTMx = dftMx';
        end
    end
end

```

7.17.7 Matlab implementation of case study in section 31.3.1 with different methods for solving for even coefficients

```
% % a method written by GHartung 6.6.2018 for computing DFT and iDFT using new method
function newFourierMethod
close all, figure(1), clf, hold on,
do2DataPoints
do4DataPoints
end

function W = getPrincipleRoot(N)
W = exp(-i*2*pi/N);
subplot(5,1,N-1) %for plotting
end

function fHatAugmented = getFHatAugmented(fHat,N)
nHalf = N/2;
% fHatAugmented = [fHat(1:nHalf); 0.5*fHat(nHalf+1); 0.5*conj(fHat(nHalf+1)); fHat(nHalf+1:N)];
fHatAugmented = [fHat(1:nHalf); 0.5*fHat(nHalf+1); 0.5*fHat(nHalf+1); fHat(nHalf+2:N)]; %fHat(nHalf+1) has no imaginary part
end

function iDFTInterpVector = getIDFTVector(W,N,t,isEven)
if isEven %when N is even - could calculate this from N
% this adds a term and inserts the complex conjugate in it
iDFTInterpVector = zeros(1,N+1); iDFTInterpVector(1) = W^(-0*t);
for j = 1:(N/2)
iDFTInterpVector(j+1) = W^(-j*t); iDFTInterpVector(N-j+2) = W^(j*t);
end
else %when N is odd - could calculate this from N
iDFTInterpVector = zeros(1,N); iDFTInterpVector(1) = W^(-0*t);
for j = 1:(N/2)
iDFTInterpVector(j+1) = W^(-j*t); iDFTInterpVector(N-j+1) = W^(j*t);
end
end
end

function do2DataPoints
t = [0;1]; f = [1;2]; N=2;
W = getPrincipleRoot(N);

%DFT
DFTMx = [W^0 W^0;
W^0 W^1];
fHat = DFTMx*f; matlabFHat = fft(f);
residual = norm(fHat-matlabFHat)
stem(abs([fHat,matlabFHat])), legend('manual DFT','matlab fft')

%simple reconstruction
iDFTMx=DFTMx';%matlab uses complex conjugate transpose
% iDFTMx = [W^0 W^0;
% W^0 W^-1];
fReconst = 1/N*iDFTMx*fHat; reconstructionResidual = norm(fReconst-f)

%interpolating
tInterp = min(t):0.01:3*max(t); fHatAugmented = [fHat(1:1); 0.5*fHat(2); 0.5*conj(fHat(2))];
for i = 1:length(tInterp)

% %blindly using formula-- does not work for even numbers
% iDFTInterpVector = [W^(-0*tInterp(i)) W^(-1*tInterp(i))];
% fInterp(i) = 1/N*iDFTInterpVector*fHat;
```

```

% %using knowledge of extra complex root that is complex conjugate
% iDFTInterpVector = [W^(-0*tInterp(i)) W^(-1*tInterp(i)) W^(1*tInterp(i))];
% ftInterp(i) = 1/N*iDFTInterpVector*fHatAugmented;

%using knowledge that last term MUST be cosine only
iDFTInterpVector = [W^(-0*tInterp(i)) W^(-1*tInterp(i))];
%% %% next 2 lines are equivalent
% iDFTInterpVector(2) = real(iDFTInterpVector(2)); % manually set coefficient to 0, causing entire imaginary term to be 0
iDFTInterpVector(2) = cos(2*pi/N*1*tInterp(i)); % manually multiplying coefficient with cosine
%% %%
ftInterp(i) = 1/N*iDFTInterpVector*fHat;
end
figure(2), subplot(5,1,N-1), scatter(t,f); hold on,
% figure, plot(t,f), hold on
plot(tInterp,real(ftInterp)), plot(tInterp,real(ftInterp)+imag(ftInterp)),
legend('original data','real part of reconstruction','real+imag part of reconstruction');
figure(1);
end

function do4DataPoints
t = [0;1;2;3]; f = [1; 3; 2; 4]; N=4;
% f = [1;2;1.5;0.5];
W = getPrincipleRoot(N);
DFTMx = [W^0 W^0 W^0 W^0;
         W^0 W^1 W^2 W^3;
         W^0 W^2 W^4 W^6;
         W^0 W^3 W^6 W^9];
fHat = DFTMx*f; matlabFHat = fft(f);
residual = norm(fHat-matlabFHat)
stem(abs([fHat,matlabFHat]))

%simple reconstruction
iDFTMx=DFTMx'; %matlab uses complex conjugate transpose
% iDFTMx = [W^0 W^0 W^0 W^0;
%          W^0 W^-1 W^-2 W^-3;
%          W^0 W^-2 W^-4 W^-6;
%          W^0 W^-3 W^-6 W^-9];
ftReconst = 1/N*iDFTMx*fHat; reconstructionResidual = norm(ftReconst-f)

%interpolating
tInterp = min(t):0.01:max(t);
fHatAugmented = getFHatAugmented(fHat,N);
% fHatAugmented = [fHat(1:2); 0.5*fHat(3); 0.5*conj(fHat(3)); fHat(4)];
for i = 1:length(tInterp)

% %blindly using formula -- does not work for even numbers
% iDFTInterpVector = [W^(-0*tInterp(i)) W^(-1*tInterp(i))];
% ftInterp(i) = 1/N*iDFTInterpVector*fHat;

% %using knowledge of extra complex root that is complex conjugate
iDFTInterpVector = getIDFTVector(W,N,tInterp(i),true);
ftInterp(i) = 1/N*iDFTInterpVector*fHatAugmented;
end
figure(2), subplot(5,1,N-1), scatter(t,f); hold on,
plot(tInterp,real(ftInterp)), plot(tInterp,real(ftInterp)+imag(ftInterp)),
figure(1);
end

```

7.17.8 Matlab implementation of case study in section 31.3.2

```
% GH 3/19/2018 -- to test 6 frequencies
function fourierTest
close all
[tData,fData] = sampleAndPlotData;
newT = 0:0.001:1;
% % do DFT
dft = ooDFT.CreateAndCompute(tData,fData);
dft2 = dft.filterDownTo(dft,7);
fNew2 = dft2.computeIDFTTest(newT,dft2);
fData = [fData fData fData fData fData fData];
tMax = max(tData); tData=[tData tData+tMax tData+tMax*2 tData+tMax*3 tData+tMax*4 tData+tMax*5];
plot(tData,fData,'linewidth',2); hold on
plot(newT,fNew2,'linewidth',2);

legend('original data','filtered reconstruction')
end

function [tSamples,fSamples] = sampleAndPlotData
fSamples
[0.561953033988024,0.710124125433452,0.763805541362655,0.813625600359705,0.907392562011622,0.991953033988024,1.0552273702
7121,1.09836601333906,1.12001793074319,1.14060790124466,1.14295598384053,1.13402088059569,1.09097958266059,1.022012031038
17,0.982436809799233,0.973501706554395,0.965091677055870,0.957433859946726,0.948926485315457,0.938714095934926,0.92817722
2778584,0.916687547262360,0.903628550212212,0.879020880595694,0.831032680005723,0.811879287675339,0.792902886495398,0.783
329140182714,0.773755393870030,0.764181647557345,0.754607901244661,0.747784892395103,0.741404361421652,0.735023830448201
,0.728643299474749,0.717764243427552,0.704997281775634,0.677840939592744,0.651073977940826,0.673988432218112,0.6966236337
91367,0.684282434184681,0.671941234577994,0.659600034971308,0.647258835364622,0.634917635757935,0.643345364371505,0.70721
5570861180,0.771085777350856,0.804914685905428,0.674637399769735,0.553680910094218,0.545703771451151,0.537726632808083,0.
529749494165015,0.521772355521947,0.513795216878879,0.505818078235811,0.497840939592744,0.489863800949676,0.523014980890
678,0.539457458766785,0.544572503014572,0.549687547262360,0.554802591510148,0.559917635757935,0.634979582660590,0.6736565
73811033,0.607882237527847,0.542107901244661,0.476333564961475,0.448650379120767,0.498214980890679,0.547779582660589,0.59
7344184430501,0.646908786200412,0.684653413254775,0.645975789993081,0.607298166731387,0.568620543469693,0.52994292020799
8,0.491265296946304,0.452587673684610,0.413910050422916];
tSamples
[0,0.002000000000000000,0.004000000000000000,0.006000000000000000,0.008000000000000000,0.010000000000000000,0.012000000000000000,
0.014000000000000000,0.016000000000000000,0.018000000000000000,0.020000000000000000,0.022000000000000000,0.024000000000000000,0.0260
00000000000000,0.028000000000000000,0.030000000000000000,0.032000000000000000,0.034000000000000000,0.036000000000000000,0.0380000000
000000,0.040000000000000000,0.042000000000000000,0.044000000000000000,0.046000000000000000,0.048000000000000000,0.050000000000000000
,0.052000000000000000,0.054000000000000000,0.056000000000000000,0.058000000000000000,0.060000000000000000,0.062000000000000000,0.0640
000000000000,0.066000000000000000,0.068000000000000000,0.070000000000000000,0.072000000000000000,0.074000000000000000,0.0760000000
000000,0.078000000000000000,0.080000000000000000,0.082000000000000000,0.084000000000000000,0.086000000000000000,0.088000000000000000
,0.090000000000000000,0.092000000000000000,0.094000000000000000,0.096000000000000000,0.098000000000000000,0.100000000000000000,0.10200
0000000000,0.104000000000000000,0.106000000000000000,0.108000000000000000,0.110000000000000000,0.112000000000000000,0.114000000000000000,
0.116000000000000000,0.118000000000000000,0.120000000000000000,0.122000000000000000,0.124000000000000000,0.126000000000000000,0.128000000000
000,0.130000000000000000,0.132000000000000000,0.134000000000000000,0.136000000000000000,0.138000000000000000,0.140000000000000000,0.1420000
000000000,0.144000000000000000,0.146000000000000000,0.148000000000000000,0.150000000000000000,0.152000000000000000,0.154000000000000000,0.15
6000000000000000,0.158000000000000000,0.160000000000000000,0.162000000000000000,0.164000000000000000,0.166000000000000000];
end
```


7.17.9 Matlab implementation of Figure 7.145

```
% GH 3/19/2018 -- Fourier filtering case study from report
function fourierFiltering
close all
[tData,fData] = sampleAndPlotData;

% runFilterin5gAndPlotting(tData,fData);
plotMatlabVsQuarteroni(tData,fData);
end

function plotMatlabVsQuarteroni(tData,fData)
pos = [500 500 200 200];
clr = [1 0 0; 0 1 0; 0 0 1; 0 1 1; 1 0 1; 0 0 0;
       0 0.4470 0.7410; 0.85 0.325 0.098; 0.929 0.694 0.125; 0.494 0.184 0.556];

dft = ooDFT.CreateAndCompute(tData,fData);
for iCol = 0:dft.N-1
    w(iCol+1) = -2*pi/dft.N*(iCol-dft.N/2); %N should be replaced by T for NDFT in the future
end
w = w./pi;
k = [0:(dft.N-1)] - dft.N/2;
makePlotWithTwoXAxes(dft.fk,w,k,clr); title('Quarteroni'),

for iCol = 0:dft.N-1
    w(iCol+1) = -2*pi/dft.N*(iCol); %N should be replaced by T for NDFT in the future
end
w = w./pi; fk = fft(fData); k = 0:dft.N-1;
makePlotWithTwoXAxes(fk,w,k,clr); title('Trefethen'),

end
function makePlotWithTwoXAxes(fk,w,k,clr)
pos = [500 500 400 250]; fk = (abs(real(fk)+imag(fk)));
figure,
% axis for k
b=axes('Position',[.1 0.1 .8 1e-12]);
set(b,'Units','normalized'); set(b,'Color','none');
% axis for frequency
a=axes('Position',[.1 .2 .8 1e-12]); set(a,'Units','normalized'); set(a,'Color','none');
% axis for shifted k
c=axes('Position',[.1 .3 .8 .6]); set(c,'Units','normalized');

for i = 1:length(fk)
    switch w(i)
        case 1, colorPlot = clr(1,:);
        case 3/4, colorPlot = clr(2,:);
        case 2/4, colorPlot = clr(3,:);
        case 1/4, colorPlot = clr(4,:);
        case 0 , colorPlot = clr(5,:);
        case -1/4, colorPlot = clr(6,:);
        case -1/2, colorPlot = clr(7,:);
        case -3/4, colorPlot = clr(8,:);

        case -4/4, colorPlot = clr(1,:);
        case -5/4, colorPlot = clr(2,:);
        case -6/4, colorPlot = clr(3,:);
        case -7/4, colorPlot = clr(4,:);
    end
    stem(i,fk(i),'color',colorPlot), hold on
end

for i = 1:length(fk), xlbl{i} = [num2str(w(i)) '\pi']; end
kOrig = 0:length(fk)-1;
set(c,'xlim',[1,length(fk)],'xTickLabels',xlbl); % xlabel('\omega');
set(b,'xlim',[1,length(fk)],'xTickLabels',num2str(kOrig)); % xlabel('k');
set(a,'xlim',[1,length(fk)],'xTickLabels',num2str(k)); % xlabel('k');
```

```

text(length(fk)+0.4,-0.04*max(fk) ,'\omega','FontSize',12)
text(length(fk)+0.4,-0.2*max(fk) ,'\k','FontSize',10)
text(length(fk)+0.4,-0.4*max(fk) ,'\k','FontSize',10)
ylabel('amplitude'), set(gcf,'position',pos), hold on
legend(xlbl{:})
end

function runFilteringAndPlotting(tData,fData)
newT = 0:0.01:3; pos = [500 500 200 200];
% % do DFT
dft = ooDFT.CreateAndCompute(tData,fData);
figure, stem(abs(real(dft.fk))+abs(imag(dft.fk))), xlabel('k'); ylabel('amplitude'), set(gcf,'position',pos)
% legend('Quateroni')

% % low pass filter
% fNew = dft.computeIDFTTest(newT,dft);
dft2 = dft.filterDownTo(dft,7);
fNew2 = dft2.computeIDFTTest(newT,dft2);
% plot(newT,fNew,'linewidth',2); hold on
filteredFk = dft2.fk;
fkHalf = floor(size((dft.fk),1)/2); newSample= fkHalf-ceil(7/2)+2:fkHalf+floor(7/2)+1;

figure, stem(abs(real(dft.fk))+abs(imag(dft.fk)),'color',[0.8 0.8 0.8]),hold on, stem(newSample,abs(real(dft2.fk))+abs(imag(dft2.fk)),'b')
xlabel('k'); ylabel('amplitude'), legend('dropped coefficients','kept coefficients')
figure,plot([tData],[fData],'linewidth',2); xlabel('time (s)'), ylabel('signal'), set(gcf,'position',pos),

figure, stem(angle(dft.fk),'color',[0.8 0.8 0.8]), hold on, stem(newSample,angle(dft2.fk),'b')
xlabel('k'); ylabel('phase angle'), legend('dropped coefficients','kept coefficients')
figure,plot([tData],[fData],'linewidth',2); xlabel('time (s)'), ylabel('signal'), set(gcf,'position',pos),

fData = [fData fData fData];
% dft.fk
tMax = max(tData)+tData(2)-tData(1); %need to increment the data forward
tData=[tData tData+tMax tData+tMax*2];
figure,plot(tData,real(fData),'linewidth',2); hold on, xlabel('time (s)'), ylabel('signal')
plot(newT,fNew2,'linewidth',2); set(gcf,'position',pos), xlim([0 3])

% % % high pass filter
dft2 = dft.highPassFilter(dft,7);
fNew2 = dft2.computeIDFTTest(newT,dft2);
% plot(newT,fNew,'linewidth',2); hold on
filteredFk = dft2.fk;
fkHalf = floor(size((dft.fk),1)/2); removedIdx= fkHalf-ceil(7/2)+2:fkHalf+floor(7/2)+1;
newSample = 1:length(dft.fk); newSample(removedIdx) = [];
figure, stem(abs(real(dft.fk))+abs(imag(dft.fk)),'color',[0.8 0.8 0.8]),hold on,
stem(newSample,abs(real(dft2.fk))+abs(imag(dft2.fk)),'b')
xlabel('k'); ylabel('amplitude'), legend('dropped coefficients','kept coefficients')
figure,plot([tData],[fData],'linewidth',2); xlabel('time (s)'), ylabel('signal'), set(gcf,'position',pos),

figure, stem(angle(dft.fk),'color',[0.8 0.8 0.8]), hold on, stem(newSample,angle(dft2.fk),'b')
xlabel('k'); ylabel('phase angle'), legend('dropped coefficients','kept coefficients')
figure,plot([tData],[fData],'linewidth',2); xlabel('time (s)'), ylabel('signal'), set(gcf,'position',pos),

fData = [fData fData fData];
% dft.fk
tMax = max(tData)+tData(2)-tData(1); %need to increment the data forward
tData=[tData tData+tMax tData+tMax*2];
figure,plot(tData,real(fData),'linewidth',2); hold on, xlabel('time (s)'), ylabel('signal')
plot(newT,fNew2,'linewidth',2); set(gcf,'position',pos), xlim([0 3])
end

function [tSamples,fSamples] = sampleAndPlotData
fSamples = [1 4 6 8 7 5 4 3];
tSamples = 0:0.1:0.7;
end

```

7.17.10 Implementation notes

The rows of the IDFT can be interpreted as a repeat of a single vector raised to different powers. Due to this, a single row vector of the IDFT matrix, can be created as in Equation (7.304) and each element of the vector can be raised to the power of “t” before summing the elements. For this evaluation technique, t represents the new time point. Another note, is that all evaluations of the exponential function and powers must be implemented in Matlab at once, because raising an exponential function to a power (to generate the principle root, W) and then raising to another power is evaluated improperly in Matlab. The creators of Matlab have acknowledged this error and advised to use sine and cosine logic (real implementation) for best results.

$$f(t) = V^t \quad \text{where } V = \sum_{k=0}^{N-1} \tilde{f}_k W_N^{-\left(k-\frac{N}{2}\right)} \quad (7.304)$$

This section lists a code for calculating the matrix T for DFT (fBarMx), the matrix T^{-1} for the IDFT (fBarPrimeMx) and the time-independent matrix used for IDFT reconstructions (fBarWithoutTime). The Fourier coefficients (fBar) can be seen after the loop. Calculating the IDFT. “x” is the independent variable (could be time), chosenLine is the line of the IDFT time-dependent matrix calculated above. timeVect is the time vector of the new time sampling.

```

% a program that calculates the DFT and IDFT variables given an input of data
% Written by G Hartung 3/22/2017
% ***** To reconstruct function for any timpoint, use iDFTVect^(newTimePoint)*fk *****
classdef ooDFT
    properties
        fk,iDFTVect,f,time,N, delT
    end
    methods (Static)
        function obj = CreateAndCompute(time,f)
            obj = ooDFT; obj.f = f; obj.time = time;
            if(size(f,1)==length(time))
                f=f;
            elseif(size(f,2)==length(time))
                f=f';
            else
                disp('time vector and data vector are not the same length; operation terminated')
                return
            end
            obj.delT = time(2)-time(1);
            obj.N = length(time);
            DFTMx = computeDFTMatrix(obj);
            obj.fk = 1/obj.N*DFTMx*f;
            obj.iDFTVect = computeIDFTVector(obj);
        end
        function [f] = computeIDFT(newTimeVect,obj)
            iDFTVect= obj.iDFTVect; fk = obj.fk; delT=obj.delT;
            nTimepoints = length(newTimeVect);
            if size(fk,1)~=length(iDFTVect)
                fk = conj(fk)'; %flip orientation of Mx
            end
            f = zeros(nTimepoints,size(fk,2));
            for iTime = 1:nTimepoints
                f(iTime,:) = iDFTVect.(newTimeVect(iTime)/delT)*fk;
            end
        end
    end
    methods
        function [fHatMx] = computeDFTMatrix(obj) %j stands for time vector
            for iRow = 0:obj.N-1
                for iCol = 0:obj.N-1
                    fHatMx(iRow+1,iCol+1) = exp(-1i*2*pi/obj.N*(iRow-obj.N/2)*iCol); %N should be replaced by T for NDFT in the future
                end
            end
        end
        function [iDFTfHatVect] = computeIDFTVector(obj) %all rows the same in IDFT Matrix
            iDFTfHatVect = zeros(1,obj.N); %might offer alpha here in the future
            for iCol = 0:obj.N-1
                iDFTfHatVect(iCol+1) = exp(1i*2*pi/obj.N*(iCol-obj.N/2));
            end
        end
    end
end
end

```

7.17.11 Non-Uniform Sampling Rates

Most of the time when data is acquired from a signal, it is uniformly sampled using some type of recording machine. There are times, however, when sampling must be done by hand and is prone to non-uniform sampling rates due to human error. This could include experimental

protocols where a person is sampling a cell culture at different times, measuring data from a chart, etc. In these cases, the DFT does not apply. In order to solve this problem 2 options are available. The first is to interpolate between the data and sample evenly. This is a fairly simple solution to implement by hand or in a Matlab procedure. The second option is to use the non-uniform discrete Fourier transform (NDFT), which is outside the scope of this work.

7.17.12 Code for generating and sampling wavelengths of different frequencies

```
% GH 3/15/2018 -- a function to generate uneven frequencies
% To save space, using a previously validated ooDFT class
function fourierTest
close all
[tData,fData] = sampleAndPlotData;

% do DFT
dft = ooDFT.CreateAndCompute(tData,fData);
% new time vector
newT = linspace(0,max(tData),100);
% do IDFT
reconstructedF = dft.computeIDFT(newT,dft);
figure, scatter(tData,fData); hold on; plot(newT,reconstructedF); title('IDFT full signal')

% drop some fK
dft10 = dft.filterDownTo(dft,10);
reconstructedF10 = dft10.computeIDFT(newT,dft10);
figure, scatter(tData,fData); hold on; plot(newT,reconstructedF10); title('10 coefficients')

dft5 = dft.filterDownTo(dft,5);
reconstructedF5 = dft5.computeIDFT(newT,dft5);
figure, scatter(tData,fData); hold on; plot(newT,reconstructedF5); title('5 coefficients')

% do linear averaging
dataLength = floor(length(tData)/3); %ignore the last 2 entries
dft1 = ooDFT.CreateAndCompute(tData(1:dataLength),fData(1:dataLength));
dft2 = ooDFT.CreateAndCompute(tData(dataLength+1:2*dataLength),fData(dataLength+1:2*dataLength));
dft3 = ooDFT.CreateAndCompute(tData(2*dataLength+1:3*dataLength),fData(2*dataLength+1:3*dataLength));
dftAverage = dft1;
for i = 1:dataLength
    dftAverage.fk(i) = (dft1.fk(i)+dft2.fk(i)+dft3.fk(i))/3;
end
reconstructedAverage = dftAverage.computeIDFT(newT,dftAverage);
figure, scatter(tData,fData); hold on; plot(newT,reconstructedAverage); title('averaging')
end

function [tSamples,fSamples] = sampleAndPlotData
omega = [10 20 30];
omegaBar = 10;
a = [2 3 4]; b = [5 6 7]; c = 50;
period = 2*pi./(omega+omegaBar);
timeOffset = [0, period(1), period(1)+period(2)];
tPlot = []; fPlot = []; tSamples = []; fSamples = [];
for i = 1:length(omega)
    %full sampling for plotting
    t = linspace(0,period(i),25);
    f = evaluateFunction(a,b,c,omegaBar+omega(i),t);
    tPlot = [tPlot t+timeOffset(i)]; fPlot = [fPlot f];
end
t = linspace(0,sum(period),20);
for i = 1:length(t)
    %downsampling for data
    if t(i) < period(1), f = evaluateFunction(a,b,c,omegaBar+omega(1),t(i));
    elseif t(i) < period(2), f = evaluateFunction(a,b,c,omegaBar+omega(2),t(i));
    else f = evaluateFunction(a,b,c,omegaBar+omega(3),t(i)); end;
    fSamples = [fSamples f];
end
tSamples = [t];
plot(tPlot,fPlot); xlabel('time'); ylabel('pressure');
hold on, scatter(tSamples,fSamples)
end

function F = evaluateFunction(a,b,c,omega,t)
f1 = a(1)*sin(omega*t) + b(1)*cos(omega*t);
```

```

f2 = a(2)*sin(omega*t) + b(2)*cos(omega*t);
f3 = a(3)*sin(omega*t) + b(3)*cos(omega*t);
F = f1+f2+f3+c;
End

% a program that calculates the DFT and IDFT variables given an input of data
% Written by G Hartung 3/22/2017
% ***** To reconstruct function for any timpoint, use iDFTVect^(newTimePoint)*fk*****
classdef ooDFT
    properties
        fk,iDFTVect,f,time,N, delT
    end
    methods (Static)
        function obj = CreateAndCompute(time,f)
            obj = ooDFT; obj.f = f; obj.time = time;
            if(size(f,1)==length(time))
                f=f;
            elseif(size(f,2)==length(time))
                f=f';
            else
                disp('time vector and data vector are not the same length; operation terminated')
                return
            end
            obj.delT = time(2)-time(1);
            obj.N = length(time);
            DFTMx = computeDFTMatrix(obj);
            obj.fk = 1/obj.N*DFTMx*f;
            obj.iDFTVect = computeIDFTVector(obj);
        end
        function [f] = computeIDFT(newTimeVect,obj)
            iDFTVect= obj.iDFTVect; fk = obj.fk; delT=obj.delT;
            nTimepoints = length(newTimeVect);
            if size(fk,1)~=length(iDFTVect)
                fk = conj(fk); % flip orientation of Mx
            end
            f = zeros(nTimepoints,size(fk,2));
            for iTime = 1:nTimepoints
                f(iTime,:) = iDFTVect.^(newTimeVect(iTime)/delT)*fk;
            end
        end
        function obj = filterDownTo(obj,NewFkLength)
            fkHalf=floor(size((obj.fk),1)/2);
            newSample= fkHalf-ceil(NewFkLength/2)+1:fkHalf+floor(NewFkLength/2);
            obj.fk = obj.fk(newSample,:);
            obj.iDFTVect = obj.iDFTVect(newSample);
        end
        function f = getSingleFrequency(obj,fkNumber,newTimeVect)
            if(length(fkNumber) > 1)
                fkSelected = obj.fk(fkNumber(1),fkNumber(2));
                iDFTVect = obj.iDFTVect(fkNumber(1));
            else
                fkSelected = obj.fk(fkNumber); iDFTVect = obj.iDFTVect(fkNumber);
            end

            nTimepoints = length(newTimeVect); f = zeros(length(newTimeVect),1);
            for iTime = 1:nTimepoints
                f(iTime) = iDFTVect.^(newTimeVect(iTime)/obj.delT)*fkSelected;
            end
        end
    end

    methods
        function [fHatMx] = computeDFTMatrix(obj) % j stands for time vector
            for iRow = 0:obj.N-1
                for iCol = 0:obj.N-1
                    fHatMx(iRow+1,iCol+1) = exp(-1i*2*pi/obj.N*(iRow-obj.N/2)*iCol); % N should be replaced by T for NDFT in the future
                end
            end
        end
    end
end

```

```

    end
end
function [iDFTfHatVect] = computeIDFTVector(obj) %all rows the same in IDFT Matrix
    iDFTfHatVect = zeros(1,obj.N); %might offer alpha here in the future
    for iCol = 0:obj.N-1
        iDFTfHatVect(iCol+1) = exp(1i*2*pi/obj.N*(iCol-obj.N/2));
    end
end
end
end
end

```


7.17.13

evaluation of the DFT matrices using real algebra

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ b_1 \end{bmatrix} = \begin{bmatrix} \cos(0) & \cos(\frac{2\pi}{4} \times 0) & \cos(\frac{2\pi}{4} \times 0) & \cos(\frac{2\pi}{4} \times 0) \\ \cos(0) & \cos(\frac{2\pi}{4} \times 1) & \cos(\frac{2\pi}{4} \times 2) & \cos(\frac{2\pi}{4} \times 3) \\ \cos(0) & \cos(\frac{2\pi}{4} \times 1) & \cos(\frac{2\pi}{4} \times 2) & \cos(\frac{2\pi}{4} \times 3) \\ \sin(0) & \sin(\frac{2\pi}{4} \times 1) & \sin(\frac{2\pi}{4} \times 2) & \sin(\frac{2\pi}{4} \times 3) \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \end{bmatrix}$$

 $N=4$

$$\begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \end{bmatrix} = \begin{bmatrix} \cos(0) & 2\cos(\frac{2\pi}{4} \times 0) & \cos(\frac{2\pi}{4} \times 2) & 2\sin(\frac{2\pi}{4} \times 0) \\ \cos(0) & 2\cos(\frac{2\pi}{4} \times 1) & \cos(\frac{2\pi}{4} \times 2) & 2\sin(\frac{2\pi}{4} \times 1) \\ \cos(0) & 2\cos(\frac{2\pi}{4} \times 2) & \cos(\frac{2\pi}{4} \times 2) & 2\sin(\frac{2\pi}{4} \times 2) \\ \cos(0) & 2\cos(\frac{2\pi}{4} \times 3) & \cos(\frac{2\pi}{4} \times 2) & 2\sin(\frac{2\pi}{4} \times 3) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ b_1 \end{bmatrix}$$

$$\begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = \begin{bmatrix} \cos(0) & \cos(0) \\ \cos(0) & \cos(\pi) \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \end{bmatrix}$$

 $N=2$

$$\begin{bmatrix} f_0 \\ f_1 \end{bmatrix} = \begin{bmatrix} \cos(0) & \cos(0) \\ \cos(0) & \cos(\pi) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \end{bmatrix}$$

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ b_1 \\ b_2 \end{bmatrix} = \begin{bmatrix} \cos(0) & \cos(0) & \cos(0) & \cos(0) & \cos(0) & \cos(0) \\ \cos(0) & \cos(\frac{\pi}{6} \times 1) & \cos(\frac{\pi}{6} \times 2) & \cos(\frac{\pi}{6} \times 3) & \cos(\frac{\pi}{6} \times 4) & \cos(\frac{\pi}{6} \times 5) \\ \cos(0) & \cos(\frac{\pi}{6} \times 2) & \cos(\frac{\pi}{6} \times 2) & \cos(\frac{\pi}{6} \times 3) & \cos(\frac{\pi}{6} \times 4) & \cos(\frac{\pi}{6} \times 5) \\ \cos(0) & \cos(\frac{\pi}{6} \times 3) & \cos(\frac{\pi}{6} \times 3) & \cos(\frac{\pi}{6} \times 3) & \cos(\frac{\pi}{6} \times 4) & \cos(\frac{\pi}{6} \times 5) \\ \sin(0) & \sin(\frac{\pi}{6} \times 1) & \sin(\frac{\pi}{6} \times 2) & \sin(\frac{\pi}{6} \times 3) & \sin(\frac{\pi}{6} \times 4) & \sin(\frac{\pi}{6} \times 5) \\ \sin(0) & \sin(\frac{\pi}{6} \times 2) & \sin(\frac{\pi}{6} \times 2) & \sin(\frac{\pi}{6} \times 3) & \sin(\frac{\pi}{6} \times 4) & \sin(\frac{\pi}{6} \times 5) \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{bmatrix}$$

 $N=6$

$$\begin{bmatrix} f_0 \\ f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{bmatrix} = \begin{bmatrix} \cos(0) & 2\cos(0) & 2\cos(0) & \cos(0) & 2\sin(0) & 2\sin(0) \\ \cos(0) & 2\cos(\frac{\pi}{6} \times 1) & 2\cos(\frac{\pi}{6} \times 2) & \cos(\frac{\pi}{6} \times 3) & 2\sin(\frac{\pi}{6} \times 1) & 2\sin(\frac{\pi}{6} \times 2) \\ \cos(0) & 2\cos(\frac{\pi}{6} \times 2) & 2\cos(\frac{\pi}{6} \times 2) & \cos(\frac{\pi}{6} \times 3) & 2\sin(\frac{\pi}{6} \times 2) & 2\sin(\frac{\pi}{6} \times 2) \\ \cos(0) & 2\cos(\frac{\pi}{6} \times 3) & 2\cos(\frac{\pi}{6} \times 3) & \cos(\frac{\pi}{6} \times 3) & 2\sin(\frac{\pi}{6} \times 3) & 2\sin(\frac{\pi}{6} \times 3) \\ \cos(0) & 2\cos(\frac{\pi}{6} \times 4) & 2\cos(\frac{\pi}{6} \times 2) & \cos(\frac{\pi}{6} \times 3) & 2\sin(\frac{\pi}{6} \times 4) & 2\sin(\frac{\pi}{6} \times 2) \\ \cos(0) & 2\cos(\frac{\pi}{6} \times 5) & 2\cos(\frac{\pi}{6} \times 2) & \cos(\frac{\pi}{6} \times 3) & 2\sin(\frac{\pi}{6} \times 5) & 2\sin(\frac{\pi}{6} \times 2) \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ b_1 \\ b_2 \end{bmatrix}$$

7.18 Appendix R: Copyright permissions for previously published material

Computers in biology and medicine Personal Use policy

(<https://www.elsevier.com/about/policies/copyright>)

Authors can use their articles, in full or in part, for a wide range of scholarly, non-commercial purposes as outlined below:

Use by an author in the author's classroom teaching (including distribution of copies, paper or electronic)

Distribution of copies (including through e-mail) to known research colleagues for their personal use (but not for Commercial Use)

Inclusion in a thesis or dissertation (provided that this is not to be published commercially)

Use in a subsequent compilation of the author's works

Extending the Article to book-length form

Preparation of other derivative works (but not for Commercial Use)

Otherwise using or re-using portions or excerpts in other works

These rights apply for all Elsevier authors who publish their article as either a subscription article or an open access article. In all cases we require that all Elsevier authors always include a full acknowledgement and, if appropriate, a link to the final published version hosted on Science Direct.

Plos computational biology License agreement (<https://www.plos.org/license>)

PLOS applies the [Creative Commons Attribution](#) (CC BY) license to works we publish. Under this license, authors retain ownership of the copyright for their content, but they

allow anyone to download, reuse, reprint, modify, distribute and/or copy the content as long as the original authors and source are cited.

Appropriate attribution can be provided by simply citing the original article (e.g., Huntingtin Interacting Proteins Are Genetic Modifiers of Neurodegeneration. Kaltenbach LS et al. *PLOS Genetics*. 2007. 3(5) doi:10.1371/journal.pgen.0030082).

Neuro-Oncology Author Rights policy

(<https://theoncologist.alphamedpress.org/site/misc/InfoForContributors.xhtml>)

As an author, you are granted rights for a large number of author uses, including use by your employer (institution or company). These rights are granted and permitted without the need to obtain specific permission from the copyright holder, AlphaMed Press, provided a full credit line is prominently placed [i.e., author name(s), journal name, copyright year, volume number, inclusive pages, and copyright holder]. These author rights are granted and apply only to articles for which you are named as the author or co-author. The author rights include:

- the right to make copies of the article for your own personal use, including for your own classroom teaching use;

- the right to make copies and distribute copies (including via e-mail) of the article to research colleagues, for the personal use by such colleagues (but not commercially or systematically, e.g., via an e-mail list or listserv);

- the right to present the article at a meeting or conference and to distribute copies of such paper or article to the delegates attending the meeting;

- for the author's employer, if the article is a "work for hire," made within the scope of the author's employment, the right to use all or part of the information in (any version of) the article for other intracompany use (e.g., training);

patent and trademark rights and rights to any process or procedure described in the article;

the right to include the article in full or in part in a thesis or dissertation (provided that this is not to be published commercially);

the right to use the article or any part thereof in a printed compilation of works of the author, such as collected writings or lecture notes (subsequent to publication of the article in the journal);

the right to prepare other derivative works, to extend the article into book-length form, or to otherwise reuse portions or excerpts in other works, with full acknowledgment of its original publication in the journal; and

the right to self-archive the work by posting the work as the final peer-reviewed author's manuscript (but not published layout) on his/her own website and his/her institution's website and repository no earlier than six months after print publication in *The Oncologist* provided that a link is made to the AlphaMed Press version.

7.19 Appendix S: Validating oxygen model generation

Mixed-mesh oxygen simulations require the combination of diffusion, reaction and convection models across multiple mesh types (in this case hexahedral meshes and 2-point networks). This document details the validation of the equation generation platform in the OX libraries implemented in Delphi 10.2 Tokyo.

The results of this report indicate that linear flow, tissue diffusion, vascular convection, vessel-tissue mass transfer, vascular reactions and tissue reactions are all properly implemented and can be used interchangeably.

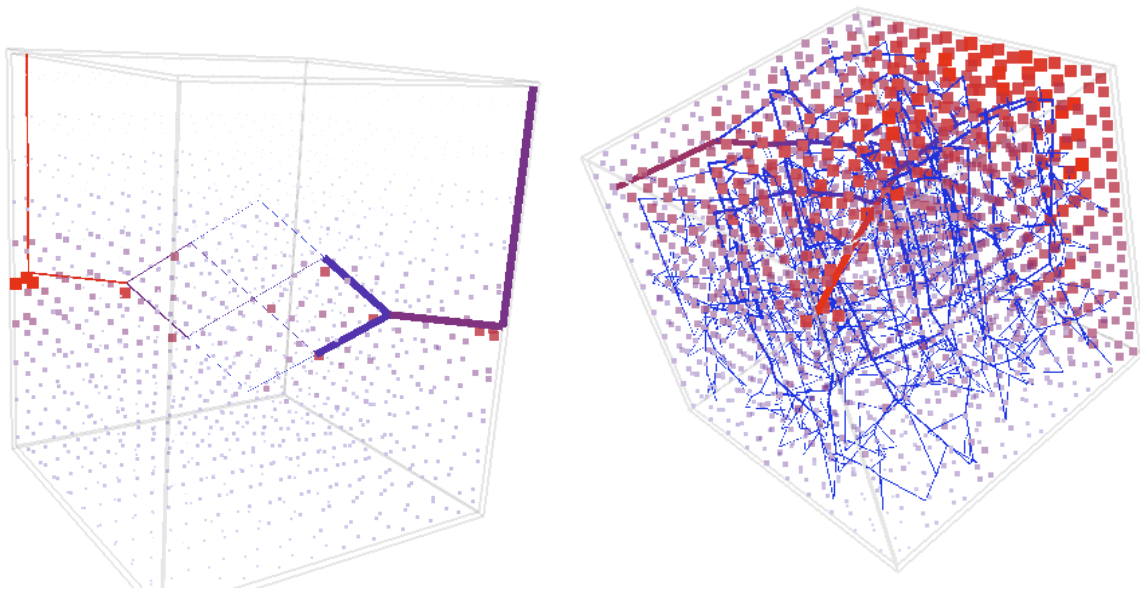


Figure 7.147. A graphical representation of two mixed mesh systems (with reaction, diffusion, mass transfer, and convection).

The distributions exemplify the nontrivial distribution achieved with the findings in this section.

7.19.1 Introduction

The recent rewrite to the equation generation libraries and matrix storage formats provided improved readability and efficient model generation. These libraries were endowed with sample codes for linear vascular flow and a model for a coupled system of water transfer due to diffusion in network (Poiseuille flow), diffusion in tissue (Darcy's law) and a point-centered mass transfer between the network and the mesh (see Section 7.28.7.3 for more information on the point-centered mass transfer methodology). These base libraries, as well as the sample applications, needed to be validated and vetted. Moreover, some mathematical models still required development for new boundary conditions (Neumann for instance) and for new matrices (such as reactions or time integration).

In order to move forward with simulating oxygen, many simulations were implemented and validated to show the independent mechanisms of mass transport are upheld with general functions such as *makeDiffusionMatrix* or *makeMassTransferMatrix*. The software made for this demonstration is entitled *ghDualDarcyAppValidation.exe*. The case studies are hardcoded using dropdown menus and different buttons as seen in the figure below.

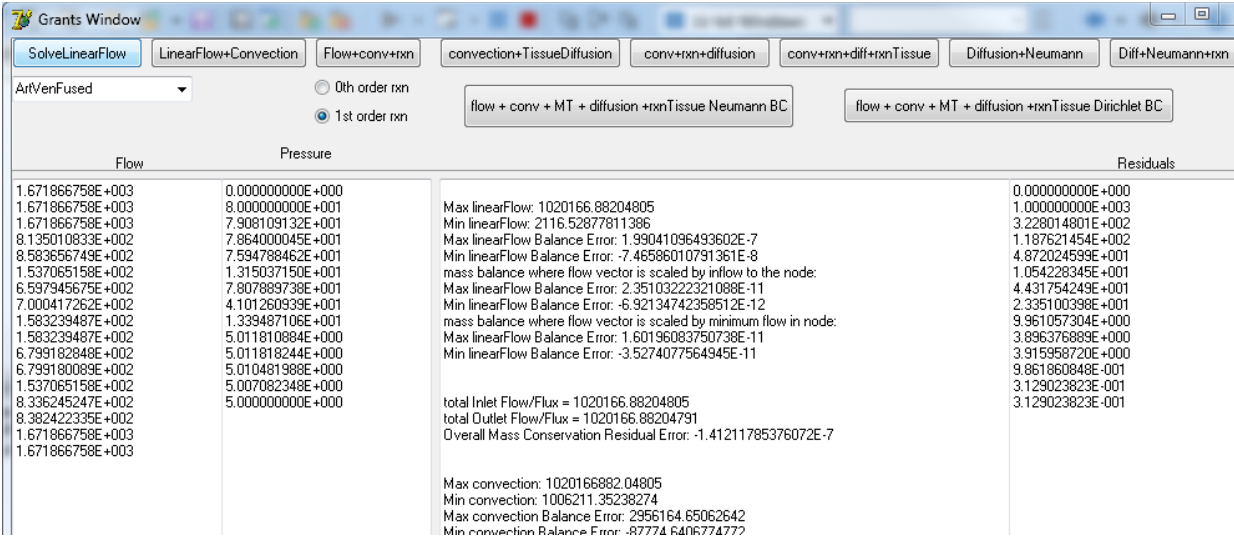


Figure 7.148. The interface used to run all case studies.
This executable will be available upon request with hardcoded case study files.

7.19.2 Linear flow

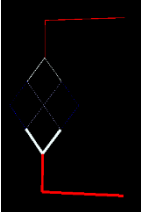
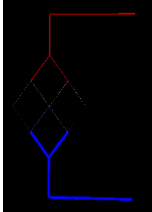
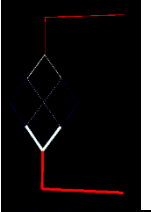
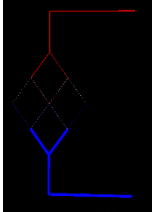
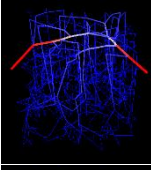
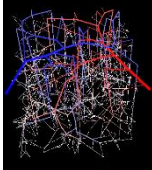
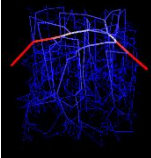
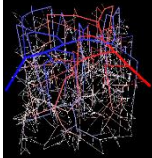
The first step in implementing the oxygen model is to validate linear blood flow. This was accomplished by comparing the OX libraries to the OO libraries using a direct sparse linear algebraic solver (MA48). The OO libraries have been previously validated (data not shown).

OO libraries (V88)	OXLibraries (V88)
File: ooGrantLinearVasculatureSource.V4.pas	File: oxGrantLinearVasculatureSource.V4.pas
Object type: ooLinVascApplication	Object type: oxLinVascApplication
Executable: Project_ooPMEVasculatureSimpleGrant	Executable: ghDualDarcyAppValidation

The mathematics follow linear flow (Hagen Poiseuille, HP, equation):

$$0 = \vec{\nabla} \cdot \left(\frac{\pi d^4}{128 \mu l} \Delta p \right), \quad p = \bar{p} \quad (7.305)$$

Where p is pressure, Δp is change in pressure, and α is the HP tube resistance. The residuals are reasonably small, the overall mass balances close, the spot checks are reasonable, and the flow vector matches that obtained from the OO libraries (previously validated). This model is validated.

Name	#	Flow (ml/min)	Average pressure (mmHg)	Flow values (ml/min)	Pressure range (mmHg)	Max norm of residual (mmHg)	Max spot check error (ml/min)	Overall balance error (ml/min)
Test 310 ooLibraries	M_ LF_ 1			153.71 – 1671.87	5-80	1.48E-9	3.0E-10	1.9E-11
Test 310 ooLibraries	M_ LF_ 2			153.71 – 1671.87	5-80	4.9E-10	3.0E-10	1.9E-11
ArtVen Fused ooLibraries	M_ LF_ 3			2116.5 – 1020167	5-80	9.8E-8	3.4E-7	1.4E-7
ArtVen Fused ooLibraries	M_ LF_ 4			2116.5 – 1020167	5-80	1.3e-7	3.4E-7	1.4E-7

7.19.3 Linear flow + convection

The next step is to correctly model convection using the blood flow as the bulk medium through which the dilute substance will be advected. The results were compared between OX to the OO libraries.

OO libraries (V86)	OXLibraries (V88)
File: ooDyeConvectionSource.pas	File: oxGrantLinearVasculatureSource.V4.pas
Object type: ooDyeConvection	Object type: oxLinVascApplication
Executable: Project_ooPMEVasculature	Executable: ghDualDarcyAppValidation

Vascular flow is modeled using HP flow but with conservation balance using only pressure formulation (see Section 7.11.4 for more information on modeling flow this way):

$$0 = \vec{\nabla} \cdot \left(\frac{\pi d^4}{128 \mu l} \Delta p \right), \quad p = \bar{p} \quad (7.306)$$

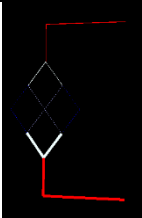
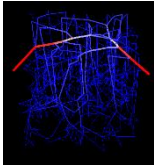
Where p is pressure, Δp is change in pressure, and α is the HP tube resistance. In vasculature, convection:

$$0 = -\vec{\nabla} \cdot (f c_A), \quad f = \frac{\pi d^4}{128 \mu l} \Delta p \quad (7.307)$$

Boundary conditions, Dirichlet at inlet:

$$c_A = \bar{c}_A \quad (7.308)$$

Findings: The residuals are reasonable, mass balances close, and the solution vector matches that from the OO libraries, validating the solution.

Name	#	Convection flow (mol/min)	Convection flow (mol/min)	Flow values (ml/min)	Pressure range (mmHg)	Concentration range (mol)	Max norm of residual (mol)	Max spot check error (mol/min)	Overall balance error (mol/min)
Test 310 ooLibraries	M_ LF_ 5	NA	NA	153.71 – 1671.87	5-80	1000 - 1000	NA	NA	NA
Test 310 ooLibraries	M_ LF_ 6		1.54E5 – 1.67E6	153.71 – 1671.87	5-80	1000 - 1000	1.5E-9	3.0E-10	1.9E-11
ArtVen Fused ooLibraries	M_ LF_ 7	NA	NA	2116.5 – 1020167	5-80	1000 - 1000	NA	NA	NA
ArtVen Fused ooLibraries	M_ LF_ 8		2.12E6 – 1.02E9	2116.5 – 1.02E6	5-80	1000 - 1000	1.3e-7	3.4E-7	1.4E-7

7.19.4 Linear flow + convection + rxn

The next level of complexity involves adding a reaction term (0th and 1st order) to remove the dilute substance as it is convected through the network. A value of $R_0 = -0.01 * nwk.NPoints$ and 1st order $k_1 = 0.01$ were chosen for reaction rates for the 0th order and 1st order reactions respectively. These values were hand-chosen to give reasonable profiles in the chosen networks.

Findings: The residuals are reasonable and all the reacting species are accounted for, thus closing the mass balance. This solution is considered validated. Vascular flow:

$$0 = \vec{\nabla} \cdot \left(\frac{\pi d^4}{128 \mu l} \Delta p \right), \quad p = \bar{p} \quad (7.309)$$

Where p is pressure, Δp is change in pressure, and α is the HP tube resistance. Reaction-convection system in vasculature follows:

$$0 = -\vec{\nabla} \cdot (f c_A) - R_0 V_v, \quad f = \frac{\pi d^4}{128 \mu l} \Delta p$$

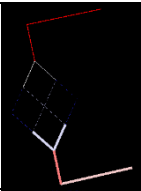
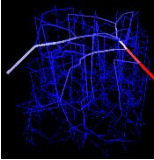
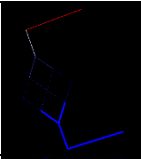
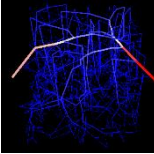
Or

$$0 = -\vec{\nabla} \cdot (f c_A) - k_1 c_A V_v, \quad f = \frac{\pi d^4}{128 \mu l} \Delta p \quad (7.310)$$

Where V_v is the vascular element volume, and c_A is the concentration of species A. Note, f is the volumetric flow rate of blood, equivalent to the velocity multiplied by the cross sectional area.

Boundary conditions (Dirichlet at inlet):

$$c_A = \overline{c_A} \quad (7.311)$$

Name	#		Conv flow (mol/ min)	Flow values (ml/ min)	Pressur e range (mmHg)	Concen tration range (mol)	Max norm of residual (mol)	Total reaction (mol/ min)	Max norm of flow balance – reaction (mol/min)
Test 310 0 th order oXLibraries	M_ LF_ 9		1.48E5 – 1.67E6	153.71- 1671.87	5-80	455.1- 1000	1.5E-9	-9.11E5	1.75E-10
ArtVen Fused 0 th order oXLibraries	M_ LF_ 10		3.60E5 – 1.02E9	2116.5 – 1.02E6	5-80	140.5 - 1000	1.29E-7	-6.63E8	1.08E-7
Test 310 1 st order oXLibraries	M_ LF_ 11		96.9 – 1.67E6	153.71- 1671.87	5-80	0.058 - 1000	1.5E-9	-1.67E6	1.46E-11
ArtVen Fused 1 st order oXLibraries	M_ LF_ 12		1.24E6 – 1.02E9	2116.5 – 1.02E6	5-80	571.9 - 1000	1.3E-7	-3.5E8	1.47e-7

7.19.5 Linear flow + convection + tissue diffusion (no reaction)

The next stage to add is the tissue diffusion (uncoupled) to the current flow-convection system to show that the concentration can diffuse through the tissue with appropriate mass balance and the vasculature is unaffected by this addition. The boundary conditions for the diffusive mesh was Dirichlet with one boundary edge set to a high value (value=100) and all others boundaries set to a low value (value=0). The chosen mesh was *ghCartesianMesh* with nVolX x nVolY x nVolZ to be 10x10x10. More information on Cartesian meshing can be found in Section 7.21. To keep the model simple, the vascular reaction has been removed from the system. D is 10 $\mu\text{m}^2/\text{s}$. The number of volumes per dimension is 10 (total of 1000 volume elements).

Note, to avoid overcomplicating the implementation, a dense mesh (all core data structures) were generated and used as if reading a mesh from file. Also note the vasculature occupies the

same physical space as the mesh, however there is no mass flux of blood or the diffuse substance between the meshes.

The residuals of the system confirm that this system solves appropriately and the mass balances close. The concentration range is also reasonable. This indicates this model is validated. Vascular flow is modeled by HP flow:

$$0 = \vec{\nabla} \cdot \left(\frac{\pi d^4}{128 \mu l} \Delta p \right), \quad p = \bar{p} \quad (7.312)$$

Where p is pressure, Δp is change in pressure, and α is the HP tube resistance. In tissue concentration:

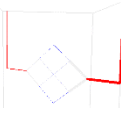
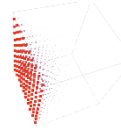
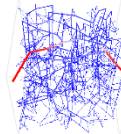
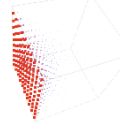
$$0 = \vec{\nabla} \cdot (D \nabla c_A) \quad (7.313)$$

Where D is the diffusivity (scalar), and c_A is the concentration of species A. In vasculature:

$$0 = -\vec{\nabla} \cdot (f c_A), \quad f = \frac{\pi d^4}{128 \mu l} \Delta p \quad (7.314)$$

Where f is the bulk fluid flow in vasculature. Boundary conditions:

$$c_A = \bar{c}_A \quad (7.315)$$

Name	#	Convection flow (mol/min)	Flow values (ml/min)	Pressure range (mmHg)	Concent ration range (mol)	Max norm of residual (mol)	Max spot check error (mol/min)	Overall balance error (mol/min)
Test 310 nwk oXLibraries	M_ LF_ 13		153.71- 1671.87	5-80	1000- 1000	1.5E-9	3.0E-10	1.9E-11
Test 310 Mesh oXLibraries	M_ LF_ 14		5.0E-15 – 3.9E5	N/A	0-100	1.5E-9	4.2E-10	1.6E-8
ArtVen Fused nwk oXLibraries	M_ LF_ 15		2116.5 – 1.02E6	5-80	1000- 1000	9.4E-8	1.2E-7	0
ArtVen Fused Mesh oXLibraries	M_ LF_ 16		1.1E-15 – 3.8E5	N/A	0-100	9.4E-8	5.8E-10	2.6E-8

7.19.6 Linear flow + convection + tissue diffusion + vascular reaction

Now the reaction in the vasculature will be re-added to the simulation from the previous step. The chosen mesh and mesh boundary conditions match the previous section as well. The reaction rates follow Section 7.19.4. Note, there is still no mass transfer between the two meshes. D is $10 \mu\text{m}^2/\text{s}$. The number of volumes per dimension is 10 (total of 1000 volume elements).

These results are in agreement with section 3.3, showing that the uncoupled system solves consistently. The residuals indicate that the system is solved and the mass balances are closing, meaning the reaction acts as it should. Vascular flow:

$$0 = \vec{\nabla} \cdot \left(\frac{\pi d^4}{128 \mu l} \Delta p \right), \quad p = \bar{p} \quad (7.316)$$

Where p is pressure, Δp is change in pressure, and α is the HP tube resistance. Concentration in tissue:

$$0 = \vec{\nabla} \cdot (D \nabla c_A) \quad (7.317)$$

Where D is the diffusivity (scalar), and c_A is the concentration of species A. The BCs for the tissue include one exterior wall is high concentration and the other 5 are low concentration. The concentration in vasculature follows:

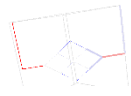
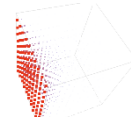
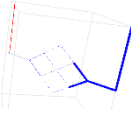
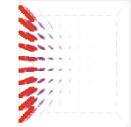
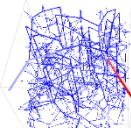
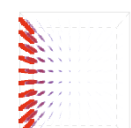
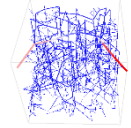
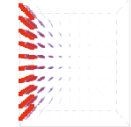
$$0 = -\vec{\nabla} \cdot (f c_A) - R_0 V_v, \quad f = \frac{\pi d^4}{128 \mu l} \Delta p$$

Or (7.318)

$$0 = -\vec{\nabla} \cdot (f c_A) - k_1 c_A V_v, \quad f = \frac{\pi d^4}{128 \mu l} \Delta p$$

Where f is the bulk fluid flow in vasculature. Boundary conditions:

$$c_A = \overline{c_A} \quad (7.319)$$

Name	#	Convective flow (mol/min)	Convective flow (mol/ min)	Flow values (ml/ min)	Concen- tration range (mol)	Max norm of residual (mol)	Total reaction (mol/ min)	Max norm of flow balance – rxn (mol/min)
Test 310 0 th order oXLibraries	M_ LF_ 17		1.48E5 – 1.67E6	153.71- 1671.87	455.1- 1000		-9.1E5	1.98E-10
						1.5E-9		
Test 310 Mesh 0 th order oXLibraries	M_ LF_ 18		1.1E-15 – 3.8E5	N/A	0-100		N/A	N/A
Test 310 1 st order oXLibraries	M_ LF_ 19		96.9 – 1.67E6	153.71- 1671.87	0.058 - 1000		-1.67E6	1.16E-10
						1.5E-9		
Test 310 Mesh 1 st order oXLibraries	M_ LF_ 20		1.1E-15 – 3.8E5	N/A	0-100		N/A	N/A
ArtVen Fused 0 th order oXLibraries	M_ LF_ 21		3.60E5 – 1.02E9	2116.5 – 1.02E6	140.5 - 1000		-6.6E8	1.08E-7
						9.4E-8		
ArtVen Fused Mesh 0 th order oXLibraries	M_ LF_ 22		1.1E-15 – 3.8E5	N/A	0-100		N/A	N/A
ArtVen Fused 1 st order oXLibraries	M_ LF_ 23		1.24E6 – 1.02E9	2116.5 – 1.02E6	571.9 - 1000		-3.5E8	1.47e-7
						9.8E-8		
ArtVen Fused Mesh 1 st order oXLibraries	M_ LF_ 24		1.1E-15 – 3.8E5	N/A	0-100		N/A	N/A

7.19.7 Linear flow + convection + tissue diffusion + vascular rxn(0th) + tissue rxn(0th)

This section introduces a reaction into the tissue. The mesh follows the previous 2 sections and the network follows the previous section. The tissue reactions follow Section 7.19.4. In mesh, a

0th order reaction was implemented with a value of $R_0=0.01/\text{mesh.iNVolumes}$. D is $10 \mu\text{m}^2/\text{s}$. The number of volumes per dimension is 10 (total of 1000 volume elements). The residuals indicate that the mass balances are closing and that the reaction acts as it should. Vascular flow:

$$0 = \vec{\nabla} \cdot \left(\frac{\pi d^4}{128 \mu l} \Delta p \right), \quad p = \bar{p} \quad (7.320)$$

Where p is pressure, Δp is change in pressure, and α is the HP tube resistance. Concentration in tissue:

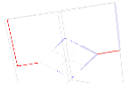
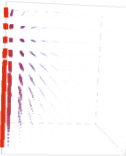
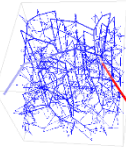
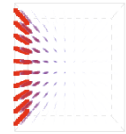
$$0 = \vec{\nabla} \cdot (D \nabla c_A) - r_0^t V_t \quad (7.321)$$

Where R_0^t is the tissue reaction rate, V_t is the volume of the tissue element, D is the tissue diffusivity constant (scalar), and c_A is the concentration of species A. Concentration in vasculature:

$$0 = -\vec{\nabla} \cdot (f c_A) - R_0^v V_v, \quad f = \frac{\pi d^4}{128 \mu l} \Delta p \quad (7.322)$$

Where R_0^v is the vasculature reaction rate and V_v is the volume of the vascular element. Boundary conditions:

$$c_A = \bar{c}_A \quad (7.323)$$

Name	#	Convection flow (mol/min)	Solute flux (mol/ min)	Flow values (ml/ min)	Concen- tration range (mol)	Max norm of residual (mol)	Total reaction (mol/ min)	Max norm of flow balance – rxn (mol/min)
Test 310 0 th order oXLibraries	M_ LF_ 25		1.48E5 – 1.67E6	153.71- 1671.87	455.1- 1000		-9.1E5	1.98E-10
Test 310 Mesh 0 th order oXLibraries	M_ LF_ 26		7.61E5 – 1.67E6	N/A	0-100	1.5E-9	-1.7E5	6.5E-10
ArtVen Fused 0 th order oXLibraries	M_ LF_ 27		3.60E5 – 1.02E9	2116.5 – 1.02E6	140.5 - 1000		-6.6E8	1.08E-7
ArtVen Fused Mesh 0 th order oXLibraries	M_ LF_ 28		0.25 – 3.79E5	N/A	-1.0E-3- 100	9.4E-8	-2.08E5	5.95E-10

7.19.8 Flow + convection + tissue diffusion + tissue rxn(0th) + MT (Dirichlet BC)

Now a mass transfer element will be added to the system. To simplify the system, the vascular reaction will be removed. The mesh size, boundary conditions, and reactions will follow the previous section. The vasculature will also follow the previous section. The mass transfer (MT) coefficient (U) is 0.01. The MT model is elaborated in Section 7.28.7.3. In short, the MT flux is created between each vascular node and the corresponding nearest mesh element. The number of volumes per dimension is 10 (total of 1000 volume elements).

The effect on the network is similar to having a vascular reaction. Given that there is no vascular reaction, this can only be attributed to the nonzero mass transfer flux. The overall mass

transfer balances when considering the mass transfer to the tissue from the vasculature (loss from the vasculature inlet to the outlet). Vascular flow:

$$0 = \vec{\nabla} \cdot \left(\frac{\pi d^4}{128 \mu l} \Delta p \right), \quad p = \bar{p} \quad (7.324)$$

Where p is pressure, Δp is change in pressure, and α is the HP tube resistance. Concentration in tissue follows:

$$0 = \vec{\nabla} \cdot (D \nabla c_A) A_t + \frac{U A_v}{t} \Delta c_A - R_0 V_t \quad (7.325)$$

Where R_0^t is the tissue reaction rate, V_t is the volume of the tissue element, D is the tissue diffusivity constant (scalar), U is the mass transfer coefficient, t is the vascular wall thickness (taken to be 1 here), A_v is the vascular element surface area, and c_A is the concentration of species A. D is set to $10 \mu\text{m}^2/\text{s}$. Concentration in vasculature:

$$0 = -\vec{\nabla} \cdot (f c_A) - \frac{U A_v}{t} \Delta c_A \quad (7.326)$$

Where

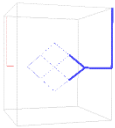
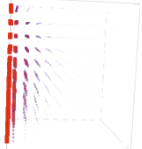
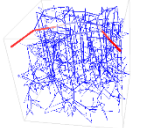
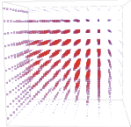
$$f = \frac{\pi d^4}{128 \mu l} \Delta p$$

And

$$\Delta c_A = c_v - c_t \quad (7.327)$$

Boundary conditions (Dirichlet):

$$c_A = \overline{c_A} \quad (7.328)$$

Name	#	Convection flow (mol/min)	Solute flux (mol/ min)	Flow values (ml/ min)	Concen- tration range (mol)	Max norm of residual (mol)	Total MT- reaction (mol/ min)	MT vasc - MT mesh (mol/min)
Test 310oXLib raries	M_ LF_ 29		5.90E4 – 1.67E6	153.71- 1671.87	189.8- 1000		1.42E6	
Test 310 Mesh 0 th order oXLibraries	M_ LF_ 30		8.16 – 1.67E6	N/A	0-100	1.5E-9	-1.25E5	1.98E-10
ArtVen Fused oXLibraries	M_ LF_ 31		1.70E6 – 1.02E9	2116.5 – 1.02E6	804.48 - 1000		-1.01E8	
ArtVen Fused Mesh 0 th order oXLibraries	M_ LF_ 32		34.66 – 6.34E5	N/A	0-218.96	1.29E-7	-1.01E8	9.9E-8

7.19.9 Tissue diffusion + Neumann BC

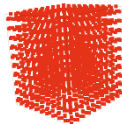
This case study validates the Neumann BC with a diffusion code. Here, at least 1 BC value must be Dirichlet, otherwise the matrix is singular and it fails to solve. In this example, I leave the first 5 BC volumes to Dirichlet 100. D is 10 $\mu\text{m}^2/\text{s}$. The number of volumes per dimension is 10 (total of 1000 volume elements).

Findings: This is not an exciting case study, but the residuals reveal the equations were solved and the solution to the system is that all values are the same as the Dirichlet value, so this case study is validated. Concentration in tissue (Diffusion):

$$0 = \vec{\nabla} \cdot (D \nabla c_A) \quad (7.329)$$

Where D is the diffusivity (scalar) and c_A is the concentration of species A. Insulated boundary conditions were implemented other than the first 5 elements of the mesh (to avoid singularity):

$$\nabla c_A = 0, \quad c_A^{1-5} = 100 \quad (7.330)$$

Name	#	Concen- tration (mol)	Flux values (ml/ min)	Concen- tration range (mol)	Max norm of residual (mol)
TestMesh oXLibraries	M_LF_33		8.5E-10 – 2.8E-14	100-100	3.8E-9

7.19.10 Tissue diffusion + Neumann BC + rxn(0th)

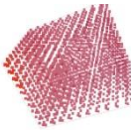
This case study validates the Neumann BC with a diffusion code. The tissue model is otherwise matching the previous section. There is no vasculature in this case study. The reaction is a 0th order with a value of $R_0 = 0.01$. D is $10 \mu\text{m}^2/\text{s}$. The number of volumes per dimension is 10 (total of 1000 volume elements).

This case study has been solved (as can be seen by the small residual error) and mass is conserved. This case study is validated. Concentration in tissue:

$$0 = \vec{\nabla} \cdot (D \nabla c_A) - R_0 V_t \tag{7.331}$$

Where D is the diffusivity (scalar), V_t is the volume of the tissue element, and c_A is the concentration of species A. Insulated boundary conditions were implemented other than the first 5 elements of the mesh (to avoid singularity):

$$\nabla c_A = 0, \quad c_A^{1-5} = 100 \tag{7.332}$$

Name	#	Concen- tration (mol)	Flux values (ml/ min)	Concen- tration range (mol)	Max norm of residual (mol)	Total reaction	Max norm of conservation
TestMesh oXLibraries	M_LF_34		1.14E-13 – 5.63E4	72.94- 100	1.5E-9	1.81E5	5.3E-10

7.19.11 Flow + convection + tissue diffusion + tissue rxn(0th) + MT (Neumann BC)

Now the Neumann BC is applied to the case study from Section 7.19.8. The reaction rate in the tissue is $k = 0.05/mesh.INVolumes$ and the diffusivity, D , is $10 \mu\text{m}^2/\text{s}$. The mass transfer coefficient (U) is set to $0.1 \mu\text{m}^2/\text{s}$. The number of volumes per dimension is 10 (total of 1000 volume elements).

The equations were solved and the solution vector is meaningful. A lot of mass transfer occurs in the regions near high concentration in the vasculature and the species reacts away as it moves away from the vasculature in the tissue. This case study is validated. Vascular flow (HP):

$$0 = \vec{\nabla} \cdot \left(\frac{\pi d^4}{128\mu l} \Delta p \right), \quad p = \bar{p} \quad (7.333)$$

Where p is pressure, Δp is change in pressure, d is the vessel diameter, μ is the viscosity and l is the vessel length. Concentration in tissue:

$$0 = \vec{\nabla} \cdot (D \nabla c_A) + \frac{UA_v}{t} \Delta c_A - R_0 V_t \quad (7.334)$$

Where R_0 is the tissue 0th order reaction rate constant, V_t is the volume of the tissue element, D is the tissue diffusivity constant (scalar), U is the mass transfer coefficient, t is the vascular wall thickness (taken to be 1 here), A_v is the vascular element surface area, and c_A is the concentration of species A. Concentration in vasculature:

$$0 = -\vec{\nabla} \cdot (f c_A) - \frac{UA_v}{t} \Delta c_A, \quad f = \frac{\pi d^4}{128\mu l} \Delta p \quad (7.335)$$

Where

$$\Delta c_A = c_v - c_t \quad (7.336)$$

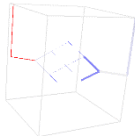
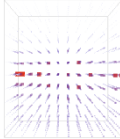
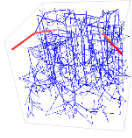
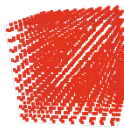
Boundary conditions:

In tissue:

$$-\nabla c_A = 0 \quad (7.337)$$

In vasculature:

$$c_A = \overline{c_A}$$

Name	#	Convection flow (mol/min)	Solute flux (mol/ min)	Concent- ration range (mol)	Max norm of residual (mol)	Total MT- reaction (mol/ min)	MT vasc - MT mesh (mol/min)
Test 310oXLib raries	M_LF_35		1.5E8– 1.7E9	993.5– 1000			
Test 310 Mesh 0 th order oXLibraries	M_LF_36		2.3E-13 – 4.7E5	576.2– 913.2	1.2E-6	1.1E7	9.7E-8
ArtVen Fused oXLibraries	M_LF_37		2.11E9 – 1.02E12	999.9 - 1000			
ArtVen Fused Mesh 0 th order oXLibraries	M_LF_38		0 – 2.16E5	976.6 – 995.0	1.1E-4	1.3E7	2.2E-4

7.19.12 Flow + convection + tissue diffusion + tissue rxn(1st) + MT (Neumann BC)

This case follows the previous section with the exception it replaces the 0th order reaction model with a 1st order model with a rate of $k_I=0.05/mesh.INVolumes$. The diffusivity is $D=10 \mu\text{m}^2/\text{s}$, and the mass transfer coefficient (U) is set to $0.1 \mu\text{m}^2/\text{s}$. The number of volumes per dimension is 10 (total of 1000 volume elements).

The equations were solved and the solution vector is meaningful. A lot of mass transfer occurs in the regions near high concentration in the vasculature and the species reacts away as it moves away from the vasculature in the tissue. This case study is validated. Vascular flow:

$$0 = \vec{\nabla} \cdot \left(\frac{\pi d^4}{128 \mu l} \Delta p \right), \quad p = \bar{p} \quad (7.338)$$

Where p is pressure, Δp is change in pressure, d is the vessel diameter, μ is the viscosity and l is the vessel length. Concentration in tissue:

$$0 = \vec{\nabla} \cdot (D \nabla c_A) + \frac{U A_v}{t} \Delta c_A - k_1 c_A V_t \quad (7.339)$$

Where R_0^t is the tissue reaction rate, V_t is the volume of the tissue element, D is the tissue diffusivity constant (scalar), U is the mass transfer coefficient, t is the vascular wall thickness (taken to be 1 here), A_v is the vascular element surface area, and c_A is the concentration of species A. Concentration in vasculature:

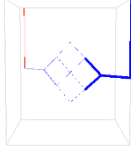
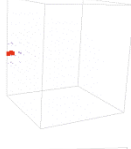
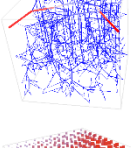
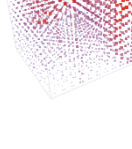
$$0 = -\vec{\nabla} \cdot (f c_A) - \frac{U A_v}{t} \Delta c_A, \quad f = \frac{\pi d^4}{128 \mu l} \Delta p \quad (7.340)$$

Where

$$\Delta c_A = c_{vasc} - c_{tiss} \quad (7.341)$$

Boundary conditions:

$$-\nabla c_A = 0, \quad c_A = \overline{c_A} \quad (7.342)$$

Name	#	Convection flow (mol/min)	Solute flux (mol/ min)	Concen- tration range (mol)	Max norm of residual (mol)	Total MT (mol/ min)	MT vasc - MT mesh (mol/min)
Test 310oXLib raries	M_LF_39		1.5E8 – 1.7E9	974.1- 1000	1.2E-6	4.3E7	1.5E-7
Test 310 Mesh 0 th order oXLibraries	M_LF_40		1.0E-19 – 1.4E6	3.0E-5- 492.1			
ArtVen Fused oXLibraries	M_LF_41		2.1E9 – 1.02E12	998.8 - 1000	1.1E-4	1.0E9	1.9E-4
ArtVen Fused Mesh 0 th order oXLibraries	M_LF_42		1.5e-12 – 8.3E5	6.3-364.0			

7.19.13 Conclusion

The newest implementation of the OX libraries is accurate and reasonable to solve all sub-problems necessary for the mixed-mesh simulation of oxygen or Dual Darcy.

7.20 Appendix T: PETSc implementation and validation

7.20.1 Summary

This document summarizes case studies used for validating the 64-bit implementation of the Portable Extensible Toolkit for Scientific computing (PETSc) [168,169,247] in Delphi. Successful implementation will be considered validated using three criteria; (i) the evaluation of the equation residuals must be tolerable (less than acceptable tolerance), (ii) overall mass balance and mass balance spot checks must be within tolerance, and (iii) the values of the solution vector must be within tolerance between the PETSc solving methods and other solvers like MA48 (a direct method for solving linear algebraic systems).

GMRES has a tendency to bounce around when it approaches its lowest residual value, causing it to iterate until it hits the iteration limit if the maximum acceptable residual (tolerance) is larger than the minimum achievable by the solver.

BCGS does not bounce around and will consider itself converged when reaching a saddle point, but frequently diverges when no preconditioners are used. BCGS results in a larger residual infinity norm when converged. BCGS is only designed for symmetric matrices, so it may work for diffusion problems, but fails the non-symmetric cases (after preconditioning). BCGS is also much faster at solving simple diffusion problems than GMRES.

For linear vasculature problems, the best success was found with BCGS using PETSc preconditioners to converge to an initial solution. This solution can then be “polished” by using GMRES either with or without PETSc preconditioners. This 2-step method was able solve the largest linear vascular problems with the same residual as the direct solver. This method was able to solve systems larger than the direct solvers.

For the Dual Darcy problem, a reasonable tolerance was difficult to achieve without preconditioners, and that GMRES with PETSc preconditioners gave the best results. In simple Cartesian diffusion problems, BCGS is the faster option. In the case of other problems, GMRES with the PETSc block Jacobi preconditioner is the best option.

Conclusions:

- Matrices have a working precision. This is ~ 14 orders of magnitude difference between the magnitude range of the coefficients in the A matrix and the magnitude of the residual (i.e. if you have 12 orders of magnitude difference between the largest and smallest coefficients in the A matrix, the best residual achievable by double precision numbers is $1e-2$). This value is a rough estimate.
- The more accurate (and preferred) method of read/write is by binary file I/O
- The preferred PETSc solver is GMRES
- PETSc solution vectors can be “polished” (to obtain lower residuals) with Gauss Seidel
- A reasonable tolerance must be chosen before the GMRES solver begins, otherwise the solver will hover until it runs out of iterations (currently hardcoded to 500,000). It is satisfactory to run the problem first without knowing the residual and watch the residual history to identify the range of minimum residual.
- For preconditioning the matrix,
 - $< 10,000$ unknowns can be solved with no preconditioning,
 - $10,000 - 100,000$ can be solved with manual preconditioning (implementation described in the report),

- 100,000 – 500,000 can be solved with a single PETSc preconditioner (block Jacobi is the preferred choice)
- > 500,000, use a dual-iterative-solver approach
- For initialization vector
 - PETSc preconditioning significantly reduces the benefit from initialization, however there are cases where solving without an initialization vector may be impossible
 - BCGS shows no gain from good initialization vector
 - GMRES shows significant reduction in computational time and improvement in solution residual vector when using initialization

7.20.2 Introduction

The current PETSc solver libraries follow the examples for KSP linear algebraic solvers presented by Argonne National Laboratories on their website. Specific configuration properties to the current implementation involves using a C++ wrapper, a C++ compiler, and 64-bit integers. The C++ wrapper calls C++ file I/O stream functions so that the binary file transfer is not limited to a maximum size of 2GB (as is true with C binary file I/O stream functions).

These examples require the compiling of a PETSc application into a standalone executable in Windows using a Linux shell (such as Cygwin) that is compatible with the message passing interface (MPI) in Windows Visual Studios. In order to send information to and from the libraries, a file writing/waiting/reading procedure is used. The preferred method for transferring the data is by binary files (*mx.aar*, *mx.aav*, *mx.aac*, *mx.bb*, *mx.xxGuess*, and *mx.cnt*).

Future advancements could include the PETSc libraries as a dynamically linked library (currently deemed impossible due to limitations with shell implementation) or COM object (currently under investigation, but the passing of the GUID/UUID between languages, compilers, and operating systems creates difficulty interpreting the values at the global reference state).

All linear vasculature simulations were conducted using *ghSolverValidation.exe* and all diffusion problems were solved using *petscInitializationApp.exe*. Dual mesh mass-transfer case studies were executed using *ghDualDarcyAppValidation.exe*.

7.20.3 Working precision of a matrix

When solving a linear algebraic set of equations numerically, the precision of the matrix solution is highly dependent on factors within the matrix, one of which being the coefficient magnitude range. In fact, this can be identified before attempting to solve the matrix by identifying the matrix coefficient ratio. An example of this problem is evident when using ANSYS; in the event that this ratio is larger than $1e8$, ANSYS gives a warning message, acknowledging this is a difficult matrix to solve.

Iterative solvers rely even more heavily on the precision of the coefficients in the A matrix. This means that the range of values in the A matrix affects the maximum tolerance of the update vector, and thus the residual vector. A case study is proposed below consisting of a 2x2 A matrix and coefficients represented by minifloats (a reduced bitrate floating precision number frequently used in graphics where the evaluations are fast and the results only cosmetic). The minifloats will have 8 bits of data, giving a range of $0-2^8$, an order of magnitude range from 0 to 10^2 translating to 2 digits of precision accuracy.

$$\frac{2}{1000}x_1 + \frac{1}{1000}x_2 = \frac{2}{1000} \quad (7.343)$$

$$x_1 + x_2 = 1$$

These two equations have a unique solution of $x = [1, 0]^T$. The same problem written in decimal form shows the precision:

$$\begin{bmatrix} 0.002 & 0.001 \\ 1.00 & 1.00 \end{bmatrix} x = \begin{bmatrix} 0.002 \\ 1.00 \end{bmatrix} \quad (7.344)$$

When adding floating point numbers of different ranged precision, the computer will generate random numbers where the precision stops. This generates new error and changes the problem:

$$\begin{bmatrix} 0.002 & 0.001 \\ 1.009 & 1.004 \end{bmatrix} x = \begin{bmatrix} 0.002 \\ 1.002 \end{bmatrix} \quad (7.345)$$

Which gives a unique solution of $x = [1.007, -0.014]^T$. In 2-dimensions, singularity is difficult to achieve due to numerical roundoff error (the error would not cause 2 nonparallel lines to become parallel). In higher dimensions (like 500,000 equations), such numerical “gibberish” accumulated across many equations will result in matrix singularity past a certain precision. In other words, the same values that should solve the original system (Equation (7.343)) will not also solve the new set of equations (Equation (7.345)).

When considering that iterative methods add, subtract, and multiply numbers of different orders of magnitude, the solution vector (and update vector) are impacted by the range of

coefficients in the matrix. Furthermore, the solution vector has only double precision (~ 16 digits of precision depending on language and implementation), meaning that if the solution vector satisfies the equation with the largest coefficients and simultaneously satisfies the smallest coefficient equations, the same update vector satisfies all orders of magnitude present in the matrix. When a working precision is ~ 16 digits but, as is true with blood flow simulation in microcirculation, the values of coefficients have range $\sim 10^{10}$ to $\sim 10^{-2}$ (a 10^{12} range) and the update can only satisfy equations to within 10^{16} range then the minimum achievable residual by an iterative method is only 10^{-4} . This can be called the working precision of a matrix as 10^{-4} , or that the matrix is singular to the precision of 10^{-4} .

This value assumes perfect precision on coefficients, which is not the case in equation generation. In the equation generation many floating point operations (flops) are applied to each coefficient prior to entering the matrix; introducing new roundoff error. In large cases (as with simulations of microcirculation), the combination of coefficient magnitude and flop error, the matrix is singular to $\sim 10^{-2}$ and thus a solution with a lower residual is physically impossible to achieve without switching to quadruple precision. In such cases, all results should be evaluated on overall mass balances and scaled local mass balances, not on the residual alone.

7.20.4 Using BCGS and GMRES

7.20.4.1 Solver Validation

The solver libraries of GMRES and BCGS will be validated using 15 case studies of linear flow with results summarized in the table below. Residual tolerance is set to $1e-6$ in all cases. These cases have no preconditioner. The problem is defined in the equation below.

Application Name: *ghSolverValidation.exe*

Button name: *compare solvers*


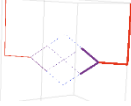
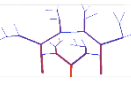
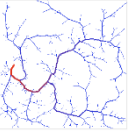
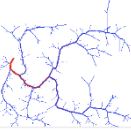
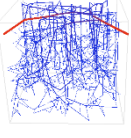
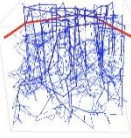
Naming convention is *\$problemType_Petsc_\$section.\$caseStudyIndex*

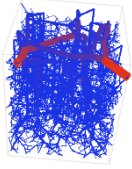
Directory: *nwk.Grant/testingPetscSolver/linearVasculature/*

Example: A linear vascular problem, chapter 3, problem 1 would be *LV_Petsc_3.1*

$$0 = \vec{\nabla} \cdot \left(-\frac{1}{\alpha} \Delta p \right), \quad f = -\frac{1}{\alpha} \Delta p \quad (7.346)$$

A) BCGS, B) GMRES, C) ma48

Name	Structure with Coloration in Flow (ml/min)		Flow values (ml/min)	Max norm of residual (mmHg)	Number of iterations	Max spot check error (ml/min)	Overall balance error (ml/min)	Max norm between solution vectors (mmHg)
<u>LV Petsc 3.1</u> <i>GrantsTestTree:</i> 6 unknowns		A	10-117	6.6e-11	2	6.5e-11	7.6e-11	8.7e-13
		B	10-117	5.5e-11	2	5.4e-11	5.2e-11	2.1e-13
		C	10-117	[4.4e-13]	NA	[1.0e-12]	[1.3e-12]	[NA]
<u>LV Petsc 3.2</u> <i>Test310</i> 13 unknowns		A	153-1,672	5.5e-7	28	5.7e-7	2.7e-7	5.0e-9
		B	153-1,672	2.9e-8	11	2.9e-8	2.1e-8	8.7e-11
		C	153-1,672	[1.5e-9]	NA	[3.0e-10]	[1.9e-11]	NA
<u>LV Petsc 3.3</u> <i>CoW extended:</i> 42 unknowns		A	0.2-32.6	5.3e-8	33	6.0e-8	1.1e-8	2.2e-8
		B	0.2-32.6	3.7e-11	21	1.0e-10	1.0e-10	2.1e-12
		C	0.2-32.6	[4.8e-12]	NA	[6.7e-12]	[1.4e-11]	NA
<u>LV Petsc 3.4</u> <i>Arteries V2:</i> 432 unknowns		A	2.8e4 – 6.0e6	Diverged	Diverged	Diverged	Diverged	Diverged
		B	2.8e4 – 6.0e6	6.6e-7	1196	9.6e-7	1.7e-6	1.0e-10
		C	2.8e4 – 6.0e6	[2.6e-7]	NA	[3.3e-7]	[6.6e-7]	NA
<u>LV Petsc 3.5</u> <i>Arteries V3:</i> 612 unknowns		A	2.8e4 – 6.0e6	Diverged	Diverged	Diverged	Diverged	Diverged
		B	2.8e4 – 6.0e6	5.8e-6	2180	6.0e-6	2.7e-5	1.6e-10
		C	2.8e4 – 6.0e6	[2.6e-7]	NA	[4.1e-7]	[2.1e-7]	NA
<u>LV Petsc 3.6</u> <i>Sparse Microcirc (mm scale)**</i> 1486 unknowns		A	2.1e-6 – 1e-3	1.9e-10	21547	4.2e-10	2.2e-10	8.3e-5
		B	2.1e-6 – 1e-3	4.6e-6	510	4.3e-6	2.5e-3	41.7
		C	2.1e-6 – 1e-3	[1.9e-16]	NA	[6.5e-17]	[1.3e-16]	NA
<u>LV Petsc 3.7</u> <i>Sparse Microcirc (um scale)</i> 1486 unknowns		A	153.7 – 1.0e+6	Diverged	Diverged	Diverged	Diverged	Diverged
		B	153.7 – 1.0e+6	3.9e-6	53632	3.9e-6	9.8e-4	2.0e-8
		C	153.7 – 1.0e+6	[9.4e-6]	NA	[2.0e-7]	[1.4e-7]	NA

<u>LV Petsc 3.8</u> <i>Medium</i> <i>Microcirc</i> 10,066 unknowns		A	232.3-7.8e5	Diverged	Diverged	Diverged	Diverged	Diverged
		B	232.3-7.8e5	1.0e-6	234735	1.0e-6	7.4e-3	3.0e-7
		C	232.3-7.8e5	[1.3e-7]	NA	[4.8e-7]	[4.8e-7]	NA

** - set tolerance to 1e-15 due to scaling

BCGS only worked for really simple case studies, diverges in all larger models. GMRES works in all case studies in this section, but is unable to solve systems larger than 11,000 unknowns. MA48 succeeded in all case studies of this section.


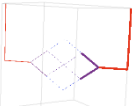

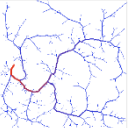
7.20.4.2 Validating Binary reading/writing

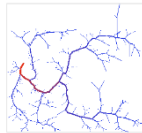
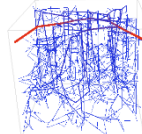
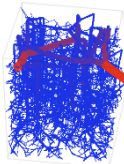
In order to save time and hard drive space, the file passing has also been implemented using binary reading and writing procedures in Delphi and c. This allows efficient and accurate file transfer between the two programs. The results are summarized below. Tolerance is set to 1e-6 in all cases. No preconditioner was used.

Results: The binary reading solves nearly equivalently to the ASCII version. The binary version is faster, and shows to be more accurate.

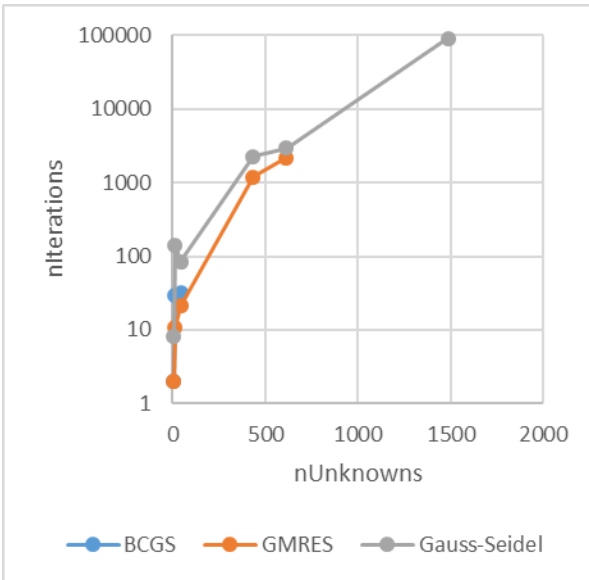
$$0 = \vec{\nabla} \cdot \left(\frac{1}{\alpha} \Delta p \right) \quad f = \frac{1}{\alpha} \Delta p \quad (7.347)$$

A) BCGS ASCII; B) GMRES ASCII; C) BCGS Binary; D) GMRES Binary; E) Gauss Seidel; F)ma48

Name	Structure with Coloration in Flow (ml/min)	Flow (ml/min)	Max norm of residual (mmHg)	Number of iterations	Max spot check error (ml/min)	Overall balance error (ml/min)	Max norm between solution vectors (mmHg)	Time to solve (s)	
<u>LV Petsc 4.1</u> <i>GrantsTestTree</i> : 6 unknowns		A	10-117	6.63 e-11	2	6.58 e-11	-7.60 e-11	8.67 e-13	0
		B	10-117	5.52 e-11	2	5.46 e-11	-5.23 e-11	2.13 e-13	0
		C	10-117	1.03 e-11	2	1.03 e-11	-1.85 e-11	6.82 e-13	0
		D	10-117	3.88 e-12	2	3.30 e-12	-6.51 e-12	2.27 e-13	0
		E	10-117	1.66 e-6	8	1.66 e-6	1.66 e-6	6.54 e-9	5.8e-6
		F	10-117	0	NA	1.0e-12	1.3e-12	NA	1.2e-4
<u>LV Petsc 4.2</u> <i>Test310</i> 13 unknowns		A	154-1,672	5.48 e-7	28	5.70 e-7	2.68 e-7	4.95 e-9	0.004
		B	154-1,672	2.88 e-8	11	2.89 e-8	-2.09 e-8	8.71 e-11	0.002
		C	154-1,672	1.13 e-8	30	1.50 e-8	4.22 e-8	7.04 e-10	0.003
		D	154-1,672	6.46 e-9	11	6.73 e-9	9.02 e-9	1.87 e-11	0.003
		E	154-1,672	0.52	141	0.52	1.32	9.11e-6	5.6e-5
		F	154-1,672	1.5e-9	NA	3.0e-10	1.9e-11	NA	6.7e-5
<u>LV Petsc 4.3</u> <i>CoW extended</i> : 42 unknowns		A	0.2-32.6	5.28 e-8	33	6.01 e-8	-1.08 e-8	2.24 e-8	0.005
		B	0.2-32.6	3.73 e-11	21	1.04 e-10	1.02 e-10	2.10 e-12	0.005
		C	0.2-32.6	8.43 e-7	32	9.47 e-7	-4.86 e-7	4.09 e-7	0.004
		D	0.2-32.6	3.02 e-11	21	3.76 e-11	2.25 e-11	2.66 e-12	0.003
		E	0.2-32.6	3.55e-5	84	3.55e-5	1.4e-4	3.66e-6	6.8e-5
		F	0.2-32.6	4.8e-12	NA	6.7e-12	1.4e-11	NA	7.9e-5
<u>LV Petsc 4.4</u> <i>Arteries V2</i> : 432 unknowns		A	Div	Diverged	Diverged	Diverged	Diverged	Diverged	Div
		B	2.8e4 – 6.0e6	6.72 e-6	1255	8.68 e-6	1.82 e-5	9.78 e-11	0.095
		C	Div	Diverged	Diverged	Diverged	Diverged	Diverged	Div
		D	2.8e4 – 6.0e6	9.65 e-7	1196	9.65 e-7	-1.72 e-6	1.01 e-10	0.09
		E	2.8e4 – 6.0e6	13.99	2281	14.0	-186.2	1.7e-4	0.03
		F	2.8e4 – 6.0e6	2.6e-7	NA	3.3e-7	6.6e-7	NA	2.5e-4

Name	Structure with Coloration in Flow (ml/min)	Flow (ml/min)	Max norm of residual (mmHg)	Number of iterations	Max spot check error (ml/min)	Overall balance error (ml/min)	Max norm between solution vectors (mmHg)	Time to solve (s)	
<u>LV Petsc 4.5 Arteries V3:</u> 612 unknowns		A	Div	Diverged	Diverged	Diverged	Diverged	Div	
		B	1.7e4 – 5.3e6	5.82 e-6	2180	6.01 e-6	-2.70 e-5	1.63 e-10	0.233
		C	Div	Diverged	Diverged	Diverged	Diverged	Div	
		D	1.7e4 – 5.3e6	9.94 e-7	2181	9.94 e-7	-9.88 e-6	1.63 e-10	0.198
		E	1.7e4 – 5.3e6	12.7	2958	12.7	-205.3	2.2e-4	0.037
		F	1.7e4 – 5.3e6	2.8e-7	NA	4.1e-7	2.0e-7	NA	2.8e-4
<u>LV Petsc 4.7 Sparse Microcirc (um scale)</u> 1486 unknowns		A	Div	Diverged	Div	Diverged	Diverged	Div	
		B	2116 – 1.0e+6	3.86 e-6	53632	3.91 e-6	-9.75 e-4	2.037 e-8	8.71
		C	Div	Diverged	Div	Diverged	Diverged	Diverged	Div
		D	2116 – 1.0e+6	1.12 e-6	58069	1.04 e-6	-9.87 e-4	2.06 e-8	9.9
		E	2116 – 1.0e+6	20.0	90723	20.0	-35.9	1.1e-2	3.5
		F	2116 – 1.0e+6	1.3e-7	NA	3.4e-7	1.4e-7	NA	0.002
<u>LV_Petsc_4.8 Sparse KF1</u> 10,066 unknowns		A	232.3 – 7.8e5	Diverged	Div	Diverged	Diverged	Div	
		B	232.3 – 7.8e5	9.99 e-7	234735	5.32 e-6	-7.42 e-3	3.01 e-7	237.9
		C	Div	Diverged	Div	Diverged	Diverged	Diverged	Div
		D	Div	Diverged	Div	Diverged	Diverged	Diverged	Div
		E	232.3 – 7.8e5	0.60	11383	0.60	-21.9	9.9e-4	2.87
		F	232.3 – 7.8e5	2.4e-8	NA	4.8e-7	4.8e-7	NA	0.023

** - set tolerance to 1e-12 due to mm scaling



nUnknowns	BCGS	GMRES	Gauss-Seidel
6	2	2	8
13	30	11	141
42	32	21	84
432	NA	1196	2281
612	NA	2181	2958
1486	NA	NA	90723

Figure 7.149. The comparison of different iterative solving algorithms with simple diagonal preconditioner.

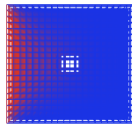
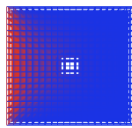
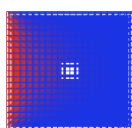
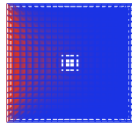
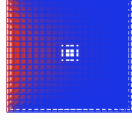
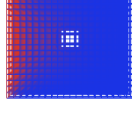
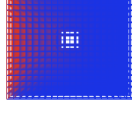
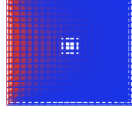
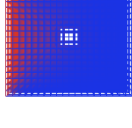
Delphi binary writing to file is limited in bits to BigEndian which is truncated at 64-bits (the exact precision of a double precision floating number). This is in contrast to Delphi writing to string in ASCII format which writes more digits to file than the double precision number holds. The differences in the results between ASCII and binary solving was attributed to random numbers being written to the ASCII file past the precision of the actual number. The final case study was able to converge with the ASCII file but not binary, but the number of iterations was very high. In this case, it appears the randomness in the numerical application assists the solver, however this should not be considered a good method. This error production is further expressed in the next section.

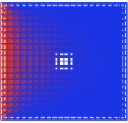
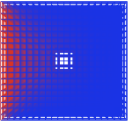
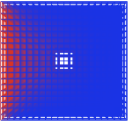
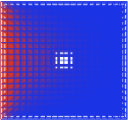
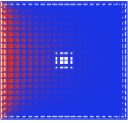
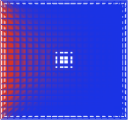
7.20.4.3 Validating Using Initialization Vector

This section validates the solving routines by passing initialization vector; a good guess should lower the iterations and the solution should return 0 iterations. The preconditioner was Block Jacobi. With four different initial guesses were used: the exact solution (from a previous solution with ma48), an initialization with all 1's, an initialization with all 100's and a modification of the solution vector by randomly generated noise between 0 and +/- 10% of the maximum of the solution vector. The file transfer is binary. The solution vector immediately converged the methods. The other initialization techniques did not greatly improve performance.

Findings: Irrespective of the initial guess, the final solutions were almost identical as expected. The initialization vectors did not greatly outperform the b vector in this case. The binary file transfer does not introduce error. Equation (7.348) was used with a Cartesian mesh. The program used for this section is the *PetscInitializationApp.exe*.

$$0 = \vec{\nabla} \cdot (D \nabla c) \quad \bar{c} = c \quad (7.348)$$

	Visualization	Values of flow across faces (ml/min)	Solver	Time to solve (s)	Num ber of itera- tions	Max spot check error (ml/min)	Overall balance error (ml/min)
<u>D Petsc 4.1</u> 20x20x20 Without initialization		0-685.4	BCGS	0.041	27	8.8e-11	-3.1e-9
<u>D Petsc 4.2</u> 20x20x20 With initialization from solution		0-685.4	BCGS	0.005	0	8.7e-11	-3.1e-9
<u>D Petsc 4.3</u> 20x20x20 With initialization all 1		0-685.4	BCGS	0.037	27	1.3 e-11	-1.8e-9
<u>D Petsc 4.4</u> 20x20x20 With initialization all 100		0-685.4	BCGS	0.035	26	3.9e-11	8.6e-9
<u>D Petsc 4.5</u> 20x20x20 With initialization +/- 10%		0-685.4	BCGS	0.036	26	3.9e-11	8.6e-9
<u>D Petsc 4.6</u> 20x20x20 Without initialization		0-685.4	GMRES	0.057	45	9.5e-11	5.0e-8
<u>D Petsc 4.7</u> 20x20x20 With initialization from solution		0-685.4	GMRES	0.005	0	9.5e-11	5.0e-8
<u>D Petsc 4.8</u> 20x20x20 With initialization all 1		0-685.4	GMRES	0.057	46	9.3e-11	4.0e-8
<u>D Petsc 4.9</u> 20x20x20 With initialization all 100		0-685.4	GMRES	0.059	47	9.2e-11	7.5e-8

D_Petsc_4.1								
0								
20x20x20		0-685.4	GMRES	0.058	47	9.2e-11	7.5e-8	
With								
initialization								
+/- 10%								
D_Petsc_4.1								
1		0-685.4	Gauss-					
20x20x20			Seidel	0.17	549	1.5e-5	-0.03	
Without								
initialization								
D_Petsc_4.1								
2		0-685.4	Gauss-	0.0003	1	1.5e-5	-0.03	
20x20x20			Seidel					
With								
initialization								
from								
solution								
D_Petsc_4.1								
3		0-685.4	Gauss-	0.18	546	1.5e-5	-0.03	
20x20x20			Seidel					
With								
initialization								
all 1								
D_Petsc_4.1								
4		0-685.4	Gauss-	0.19	621	1.85e-5	0.03	
20x20x20			Seidel					
With								
initialization								
all 100								
D_Petsc_4.1								
5		0-685.4	Gauss-	0.19	621	1.5e-5	0.03	
20x20x20			Seidel					
With								
initialization								
+/- 10%								

7.20.4.4 Solving Networks using ma48 Solution to Initialize and Converge Diverging Solvers


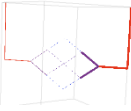

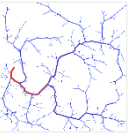
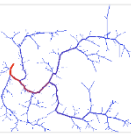
If the initialization vectors are operating properly and the solver is stable, then an initialization with the ma48 solution should arrive at the same solution. This should also work for cases where the solution was previously unachievable (diverged or DNC due to saddle-like point). The results are summarized below.

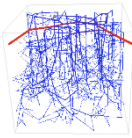
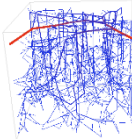
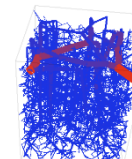
Findings: The initialization using the solution vector causes significant enhancement to the iterative solvers. The ASCII file transfer introduces numerical error which was not observed during binary file I/O.

Network Directory: *nwk.Grant/testingPetscSolver/linearFlow*

$$0 = \vec{\nabla} \cdot \left(\frac{1}{\alpha} \Delta p \right) \quad f = \frac{1}{\alpha} \Delta p \quad (7.349)$$

A) BCGS ASCII; B) GMRES ASCII; C) BCGS Binary; D) GMRES Binary; E) Gauss Seidel

Name	Structure with coloration in flow (ml/min)	Flow (ml/min)	Max norm of residual (mmHg)	Number of iterations	Max spot check error (ml/min)	Overall balance error (ml/min)	Max norm between solution vectors (mmHg)
LV_Petsc_4.9 <i>GrantsTestTree</i> : 6 unknowns		A	10-117	3.6e-13	0	4.0e-13	8.4e-12
		B	10-117	3.6e-13	0	4.0e-13	8.4e-12
		C	10-117	0	0	1.0e-12	1.3e-12
		D	10-117	0	0	1.0e-12	1.3e-12
		E	10-117	0	1	4.8e-12	4.8e-12
LV_Petsc_4.10 <i>Test310</i> 13 unknowns		A	154-1,672	6.2e-10	0	8.11e-10	3.7e-10
		B	154-1,672	6.2e-10	0	8.11e-10	3.7e-10
		C	154-1,672	1.9e-9	0	7.3e-10	1.9e-10
		D	154-1,672	1.9e-9	0	7.3e-10	1.9e-10
		E	154-1,672	5.8e-10	1	3.0e-10	-2.0e-10
LV_Petsc_4.11 <i>CoW extended</i> : 42 unknowns		A	0.2-32.6	1.3e-11	0	1.3e-11	9.6e-12
		B	0.2-32.6	1.3e-11	0	1.3e-11	9.6e-12
		C	0.2-32.6	4.8e-12	0	6.7e-12	1.4e-11
		D	0.2-32.6	4.8e-12	0	6.7e-12	1.4e-11
		E	0.2-32.6	2.6e-12	1	6.5e-12	6.8e-12
LV_Petsc_4.12 <i>Arteries V2</i> : 432 unknowns		A	2.8e4 – 6.0e6	6.3e-6	15	9.4e-6	1.3e-5
		B	2.8e4 – 6.0e6	6.5e-6	13	8.7e-6	8.7e-6
		C	2.8e4 – 6.0e6	2.6e-7	0	3.3e-7	6.6e-7
		D	2.8e4 – 6.0e6	2.6e-7	0	3.3e-7	6.6e-7
		E	2.8e4 – 6.0e6	3.8e-7	1	3.6e-7	7.9e-7
LV_Petsc_4.13 <i>Arteries V3</i> : 612 unknowns		A	2.8e4 – 6.0e6	5.1e-6	681	5.3e-6	1.6e-5
		B	2.8e4 – 6.0e6	4.6e-6	10	4.8e-6	8.5e-8
		C	2.8e4 – 6.0e6	2.8e-7	0	4.1e-7	2.0e-7
		D	2.8e4 – 6.0e6	2.8e-7	0	4.1e-7	2.0e-7
		E	2.8e4 – 6.0e6	3.4e-7	1	6.8e-7	2.0e-7

LV_Petsc_ 4.14 Sparse Microcirc** (mm scale) 1486 unknowns		A	$2.1e-6 - 1e-3$	$1.1e-15$	0	$1.1e-15$	$1.0e-16$	$5.0e-14$
		B	$2.1e-6 - 1e-3$	$1.5e-15$	1	$1.1e-15$	$1.0e-16$	$5.0e-14$
		C	$2.1e-6 - 1e-3$	$1.9e-16$	0	$6.5e-17$	$1.3e-16$	0
		D	$2.1e-6 - 1e-3$	$1.9e-16$	0	$6.5e-17$	$1.3e-16$	0
		E	$2.1e-6 - 1e-3$	$1.4e-16$	1	$1.8e-16$	$1.3e-16$	0
LV_Petsc_ 4.15 Sparse Microcirc (um scale) 1486 unknowns		A	$2116 - 1.0e+6$	$2.8e-6$	25	$2.7e-6$	$4.0e-7$	$2.0e-9$
		B	$2116 - 1.0e+6$	$2.9e-6$	18	$3.4e-6$	$1.6e-7$	$6.5e-11$
		C	$2116 - 1.0e+6$	$1.3e-7$	0	$3.4e-7$	$1.4e-7$	0
		D	$2116 - 1.0e+6$	$1.3e-7$	0	$3.4e-7$	$1.4e-7$	0
		E	$2116 - 1.0e+6$	$1.3e-7$	1	$1.3e-7$	$1.2e-7$	0
LV_Petsc 4.16 Sparse KFI 10,066 unknowns		A	$232.3 - 7.8e5$	$2.1e-7$	1	$4.6e-6$	$4.7e-6$	$4.7e-13$
		B	$232.3 - 7.8e5$	$4.1e-7$	1	$5.2e-6$	$5.3e-6$	$4.8e-13$
		C	$232.3 - 7.8e5$	$2.2e-8$	0	$4.8e-7$	$4.8e-7$	0
		D	$232.3 - 7.8e5$	$2.2e-8$	0	$4.8e-7$	$4.8e-7$	0
		E	$232.3 - 7.8e5$	$2.2e-8$	1	$4.8e-7$	$4.8e-7$	0

The binary solvers always solved when initialized with the ma48 initialization. The ASCII reading/writing implementation is not as reliable, as the roundoff error caused the ASCII solvers to require a few iterations to re-solve the previously solved solution (to a different solution). The

binary solvers appear to truncate all numbers at the working precision of 64-bits of information (double floating precision, 8 Bytes per value).

Given that the numbers are not modified but only copied to file and copied back, the continued solving by the ASCII file transfer can be attributed to the file writing format having higher/lower precision than 64-bits. This then causes the system to converge to a different (albeit similar) solution vector. The binary I/O is the preferred method because it converges to the same solution as the ma48 direct method. This method also saves hard drive space and time. The current implementation also writes the multiple files (discussed in Section 7.20.2) in parallel.

7.20.5 Simple diagonal matrix conditioning

While working with very large structures ($>250,000$ unknowns), iterative solvers are sensitive to the matrix condition number as explained elsewhere [248]. Preconditioning a system is a method for improving the condition number of a matrix prior to solving it; making a poorly conditioned matrix more solvable. In the event that the matrix had a reasonable condition number prior to applying the preconditioner, the preconditioner will still improve the condition number and lower the number of iterations to converge the high-frequency noise in the solver.

In experience, solving linear vasculature flow systems with no preconditioners is prone to failure at $>10,000$ unknowns. Using a simple diagonal preconditioner like the one described below can extend the solvability to solve up to $100,000$ unknowns. To solve larger systems than this, the PETSc preconditioners can assist with both speed and stability of convergence. PETSc preconditioners allow solving the largest stable systems passed. The solver can be improved for networks with $>500,000$ unknowns and poor condition number by using a method discussed in

Section 7.20.8. Systems with >1,000,000 unknowns and reasonable condition number have been solved with PETSc GMRES and Block Jacobi preconditioner.

Findings: Matrix preconditioning enhances solvability of a system. Manual preconditioning increases the maximum size of linear vascular flow equations that can be adequately solved using GMRES from 10,000 (without preconditioning) to >100,000 (with manual preconditioning).

7.20.5.1 Matrix Conditioning

One way to condition a matrix reviewed here will use the ratio between the values of the unknown vector (in the case of pressure, range is 120-5) and the values of the matrix coefficients (in our case values 1e4-1e15). When these values vary by more than an order of magnitude, it makes a set of equations difficult to solve, so to investigate the effect conditioning has on the matrix, the coefficients will be manually scaled to the order of magnitude of the solution vector (in this case 10^1 - 10^2). This can be accomplished using Equation (7.350). An example of this scaling applied to a linear algebraic set of equations is shown in Equation (7.351) with corresponding solution vector. This conditioning is applied to every equation that has the diagonal coefficient magnitude significantly larger than the solution vector.

$$\alpha^* = \frac{\alpha}{\max(\alpha)} 100 \quad (7.350)$$

Original equations:

$$\begin{bmatrix} 1 & 0 & 0 \\ -100 & 200 & -100 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1.00001 \\ 1.99999 \\ 2.90000 \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \\ 3 \end{bmatrix} = \begin{bmatrix} 1e-5 \\ 9.997 \\ -0.1 \end{bmatrix} \quad (7.351)$$

Conditioned equations:

$$\begin{bmatrix} 1 & 0 & 0 \\ -0.00015 & 0.0003 & -0.00015 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 3 \end{bmatrix}; \quad \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

An example of the residual error for a given guess of p is evaluated in the original and conditioned problem as shown in equation (7.352).

Original equations:

$$\begin{bmatrix} 1 & 0 & 0 \\ -100 & 200 & -100 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1.00001 \\ 1.99999 \\ 2.90000 \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \\ 3 \end{bmatrix} = \begin{bmatrix} 1e-5 \\ 9.997 \\ -0.1 \end{bmatrix}$$

(7.352)

Conditioned equations:

$$\begin{bmatrix} 1 & 0 & 0 \\ -0.00015 & 0.0003 & -0.00015 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1.00001 \\ 1.99999 \\ 2.90000 \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \\ 3 \end{bmatrix} = \begin{bmatrix} 1e-5 \\ 1.5e-5 \\ -0.1 \end{bmatrix}$$


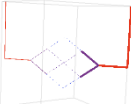
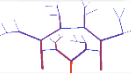
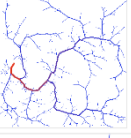
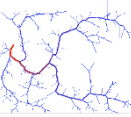
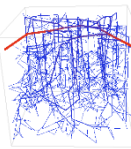
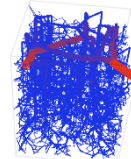
The conditioned equations have a much lower residual. Due to this, the residual tolerance will need to be adjusted for the solution vector, but the equation set will be closer to orthogonal (less oblique) with conditioning than without. This will improve update step direction and improve the solving time of the matrix.

Case studies. The residual was set to 1e-12 for *LV_Petsc_6.4 - LV_Petsc_6.7* and 1e-6 for the remainder. All residuals are computed on the unconditioned matrix for direct comparison.

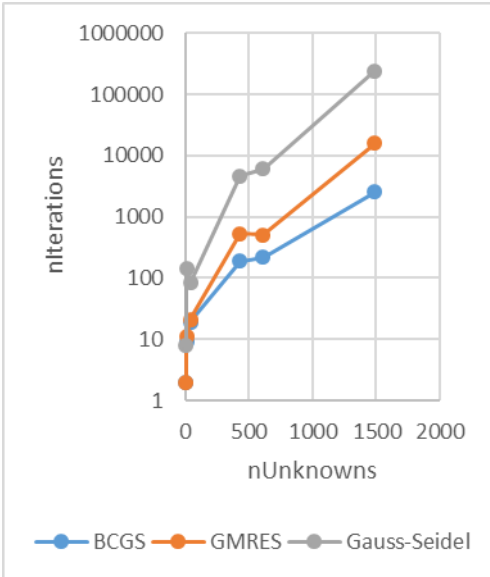
Network Directory: *nwk.Grant/testingPetscSolver/linearFlow*

$$0 = \bar{\nabla} \cdot \left(\frac{1}{\alpha} \Delta p \right) \quad f = \frac{1}{\alpha} \Delta p \quad (7.353)$$

A) BCGS Binary; B) GMRES Binary; C) Gauss-Seidel

Name	Structure with coloration in flow (ml/min)		Inflow (ml/min)	Max norm of residual (mmHg)	Number of iterations	Max spot check error (ml/min)	Overall balance error (ml/min)	Max norm between solution vectors (mmHg)
<u>LV Petsc 5.1</u> <i>GrantsTestTree:</i> 6 unknowns		A	116.887	2.6e-12	2	2.1e-12	2.4e-12	4.3e-14
		B	116.887	3.0e-12	2	2.5e-12	2.3e-12	1.4e-14
		C	116.887	1.6e-6	8	1.7e-6	1.7e-6	6.5e-9
<u>LV Petsc 5.2</u> <i>Test310</i> 13 unknowns		A	1671.867	7.1e-3	9	7.1e-3	2.3e-2	1.5e-7
		B	1671.867	4.8e-8	11	4.8e-8	5.2e-8	4.3e-13
		C	1671.867	0.52	141	0.52	-1.32	9.1e-6
<u>LV Petsc 5.3</u> <i>CoW extended:</i> 42 unknowns		A	32.6373	3.7e-7	19	5.6e-7	1.6e-7	5.5e-8
		B	32.6373	1.1e-11	21	1.5e-11	1.9e-11	1.1e-13
		C	32.6373	3.5e-5	84	3.5e-5	1.4e-4	3.7e-6
<u>LV Petsc 6.4</u> <i>Arteries V2:</i> 432 unknowns**		A	5.97e6	2.3e-6	188	2.3e-6	5.9e-7	1.6e-12
		B	5.97e6	5.7e-7	527	7.3e-7	-1.3e-5	1.4e-11
		C	5.97e6	1.43e-5	4616	1.41e-5	1.8e-4	1.7e-10
<u>LV Petsc 5.5</u> <i>Arteries V3:</i> 612 unknowns **		A	5.25e6	3.4e-6	217	3.2e-6	5.8e-7	3.6e-13
		B	5.25e6	5.2e-7	504	3.8e-7	7.41e-6	9.6e-12
		C	5.25e6	1.3e-5	6042	1.3e-5	2.1e-4	2.2e-10
<u>LV Petsc 5.6</u> <i>Sparse Microcirc (um scale)</i> 1486 unknowns **		A	1.020e6	2.3e-6	2556	2.4e-6	3.0e-7	3.0e-10
		B	1.020e6	0.5	15747	0.5	-1.42	4.2e-3
		C	1.020e6	2.0E-5	241489	2.0e-5	3.6e-5	1.1e-8
<u>LV Petsc 5.7</u> <i>Sparse KFI</i> 10,066 unknowns **		A	7.762e5	3.8e-7	734	1.4e-7	1.3e-6	2.1e-12
		B	7.762e5	3.5e-8	1602	9.2e-8	1.1e-6	5.0e-11
		C	7.762e5	5.9e-7	25109	6.0e-7	2.2e-5	9.8e-10

**residual manually changed to 1e-12 instead of 1e-6



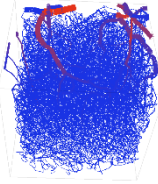
nUnknowns	BCGS	GMRES	Gauss-Seidel
6	2	2	8
13	9	11	141
42	19	21	84
432	188	527	4616
612	217	504	6042
1486	2566	15474	241489

Figure 7.150. The comparison of different iterative solving algorithms with simple diagonal preconditioner.

These case studies conclude that the conditioning is effective, however the tolerance must also be scaled accordingly. No automated method for scaling the residual tolerance with preconditioning has yet been identified. The numerical resolution of a double precision floating point is $\sim 1e-12$, so when $1e-15$ is chosen as the residual tolerance, the residual never drops to the tolerance (the matrix was singular in that level of precision).

To test the robustness of the preconditioners, larger case studies than previously solved will be examined. The results are reported below. The tolerance was set to $1e-6$. The manual preconditioning does enhance the result solvability.

A) BCGS ASCII; B) GMRES ASCII; C) BCGS Binary; D) GMRES Binary; E) MA48;
F) Gauss-Seidel

Name	Structure with coloration in flow (ml/min)		Inflow (fL/s)	Max norm of residual (mmHg)	Number of iterations	Max spot check error (fL/s)	Overall balance error (fL/s)	Max norm between solution vectors (mmHg)
LV_PetSc_6.13 KF1: 129,282 unknowns		A	7180	0.023	2789	0.023	-1.7	0.0029
		B	7180	0.065	26300	0.065	-2.6	0.010
		C	7180	0.015	3101	0.015	-0.6	0.0025
		D	7180	0.011	16213	0.011	-2.48	0.0097
		E	7180	2.5e-7	NA	1.7e-7	2.1e-7	NA
		F	7180	0.162	479996	0.162	-18.0	0.071

7.20.6 PETSc block Jacobi preconditioner

PETSc comes standard with preconditioners that can help systems solve faster and more robustly. These solvers allow us to converge systems that are too large to converge without preconditioners (larger systems than in Section 7.20.5). A review of the Block Jacobi preconditioner is offered elsewhere [249]. In short, the Jacobi preconditioner conditions the matrix using a positive-definite nonsingular matrix L to assist in solving the linear algebraic system:

$$Ax = b \quad (7.354)$$

$$M^{-1}Ax = M^{-1}b \quad (7.355)$$

$$M^{-1} = LL^T \quad (7.356)$$

$$L^T Ax = L^T b \quad (7.357)$$

$$\hat{A}\hat{x} = \hat{b} \quad (7.358)$$

$$\hat{A} = L^T AL, \quad \hat{x} = L^{-1}x, \quad \hat{b} = L^T b \quad (7.359)$$

Where the L matrix is defined as the decomposition into:

$$A = L + D + L^T \quad (7.360)$$

$$M = D \quad (7.361)$$

Or more generally:

$$A = L + D + U \quad (7.362)$$

$$M = D \quad (7.363)$$


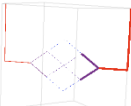
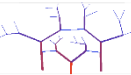
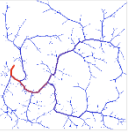
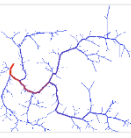
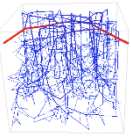
Where the matrix D is a diagonal matrix, L is the lower triangular factor and U is the upper triangular factor. The preconditioner is computed and applied in every iteration to the update vector. The block variant breaks the matrix by mutually disjoint sub-blocks of the matrix or based on the physical domain. The discussion of blocking algorithms can be found elsewhere [250].

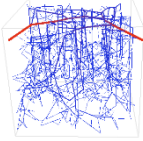
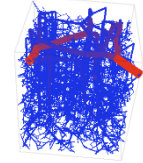
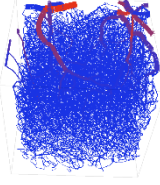
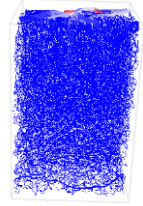
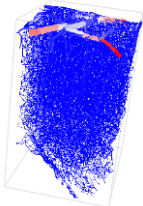
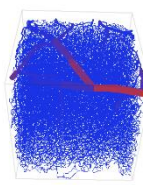
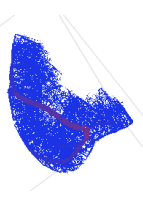
To investigate the effects of this preconditioner, a few case studies have been conducted below. All solutions used tolerance of 1e-12 for LV_Petsc_3.41 – 3.46 and LV_Petsc_3.46, 3e-10 for LC_Petsc_3.51, and 1e-6 for the remainder. Here A is BCGS ASCII, B is GMRES ASCII, C is BCGS Binary, and D is GMRES Binary.

Findings: PETSc preconditioning using Block Jacobi allows solving >1,000,000 unknowns in linear vasculature. Note, GMRES (binary) using Block Jacobi has solved other linear vasculature flow problems for over 3 million unknowns and BCGS has solved diffusion problems >200 million unknowns (data not shown). Note, all results are subject to natural orthogonality of equation set.

$$0 = \vec{\nabla} \cdot \left(\frac{1}{\alpha} \Delta p \right) \quad f = \frac{1}{\alpha} \Delta p \quad (7.364)$$

BCGS ASCII; B) GMRES ASCII; C) BCGS Binary; D) GMRES Binary;

Name	Structure with coloration in flow (ml/min)		Inflow (mL/s)	Max norm of residual (mmHg)	Number of iterations	Max spot check error (ml/min)	Overall balance error (ml/min)	Max norm between solution vectors (mmHg)	Time to solve (s)
<u>LV Petsc 6.1</u> <i>GrantsTestTree:</i> 6 unknowns		A	116.887	5.3e-11	1	5.3e-11	5.3e-11	2.1e-13	0.001
		B	116.887	4.9e-11	1	4.9e-11	4.9e-11	2.1e-13	0.001
		C	116.887	0	1	1.0e-12	1.3e-12	0	0.001
		D	116.887	1.3e-13	1	1.3e-13	1.2e-12	1.4e-14	0.001
<u>LV Petsc 6.2</u> <i>Test310</i> 13 unknowns		A	1671.86 7	2.0e-8	9	2.0e-8	1.7e-8	5.0e-13	0.053
		B	1671.86 7	1.5e-8	6	1.4e-8	1.8e-8	5.4e-13	0.093
		C	1671.86 7	2.7e-9	9	2.6e-9	1.1e-9	4.3e-14	0.001
		D	1671.86 7	4.6e-10	37	5.8e-10	6.9e-10	4.3e-14	0.002
<u>LV Petsc 6.3</u> <i>CoW extended:</i> 42 unknowns		A	32.6373	5.9e-11	9	7.6e-11	7.7e-11	1.4e-12	0.007
		B	32.6373	5.0e-11	13	7.2e-11	8.0e-11	1.3e-12	0.004
		C	32.6373	4.8e-12	13	8.2e-12	7.9e-12	9.9e-14	0.002
		D	32.6373	4.5e-12	43	8.5e-12	8.7e-12	7.1e-14	0.002
<u>LV_Petsc_6.4</u> <i>Arteries V2:</i> 432 unknowns		A	5.97e6	7.4e-7	38	7.4e-7	2.4e-6	6.0e-11	0.008
		B	5.97e6	3.3e-7	89	8.6e-7	2.5e-6	2.5e-11	0.011
		C	5.97e6	2.5e-7	89	9.1e-7	1.6e-6	2.3e-11	0.011
		D	5.97e6	2.5e-7	89	9.1e-7	1.6e-6	2.3e-11	0.010
<u>LV_Petsc_6.5</u> <i>Arteries V3:</i> 612 unknowns		A	5.25e6	5.7e-6	72	6.1e-6	3.3e-5	1.8e-11	0.009
		B	5.25e6	5.0e-6	117	5.2e-6	3.4e-5	1.8e-11	0.017
		C	5.25e6	1.2e-6	71	1.1e-6	2.7e-7	6.5e-13	0.009
		D	5.25e6	5.7e-7	117	6.8e-7	1.3e-7	1.7e-12	0.013
<u>LV_Petsc_6.6</u> <i>Sparse Microcirc (mm scale)</i> 1486 unknowns		A	1.020e-3	9.3e-13	83	9.6e-13	4.9e-12	1.2e-6	0.021
		B	1.020e-3	9.0e-13	202	9.0e-13	1.7e-11	1.3e-6	0.058
		C	1.020e-3	9.0e-13	202	9.0e-13	1.7e-11	1.2e-6	0.048
		D	1.020e-3	9.0e-13	202	9.0e-13	1.7e-11	1.3e-6	0.059

LV_Petsc_ 6.7 <i>Sparse Microcirc (um scale)</i> 1486 unknowns		A	1.020e6	3.3e-6	100	3.2e-6	2.8e-6	4.0e-10	0.028
		B	1.020e6	3.3e-6	263	3.3e-6	5.9e-6	1.3e-9	0.061
		C	1.020e6	9.6e-7	103	1.0e-6	3.3e-7	1.1e-10	0.02
		D	1.020e6	8.3e-7	264	8.3e-7	2.1e-6	1.1e-9	0.054
LV_Petsc_ 6.8 <i>Medium Microcirc</i> 10,066 unknowns		A	7.762e5	5.4e-7	155	6.4e-7	4.9e-6	3.2e-10	0.188
		B	7.762e5	9.7e-7	392	9.7e-7	1.9e-5	1.1e-9	0.491
		C	7.762e5	1.0e-6	391	1.0e-6	2.0e-5	1.0e-9	0.552
		D	7.762e5	1.0e-6	391	9.9e-7	2.0e-5	1.0e-9	0.489
LV_Petsc_ 6.25 KF1: 129,282 unknown		A	4.9e-4-7181	2.3e-6	1841**	8.0e-6	3.1e-5	1.0e-9	39.5
		B	4.9e-4-7181	1.5e-6	3091*	6.2e-6	2.4e-5	2.7e-8	60.8
		C	4.9e-4-7181	2.7e-6	1563**	2.6e-6	2.0e-8	2.2e-10	23.73
		D	4.9e-4-7181	7.7e-7	3402*	9.6e-7	2.5e-5	7.7e-8	48.02
		E	4.9e-4-7181	0.16	479996	0.16	18.01	NA	1518.5
		F	4.9e-4-7181	2.5e-7	NA	1.7e-7	2.1e-7	NA	1.19
LV_Petsc_ 6.26* KF2: 159,668 unknown		A	-7589.7-7589.7	2.01e-6	2067	6.8e-6	2.2e-5	NA	38.7
		B	-7589.7-7589.7	2.6e-6	24391	5.1e-6	6.0e-6	NA	516.4
		C	-7589.7-7589.7	3.1e-6	1915	3.0e-6	3.2e-7	NA	45.79
		D	-7589.7-7589.7	9.5e-7	18001	8.4e-7	2.9e-5	NA	389.4
		E	-7589.7-7589.7	NA	NA	NA	NA	NA	NA
		F	-7589.7-7589.7	1.92e-7	NA	2.2e-8	2.15e-8	NA	1.67
LV_Petsc_ 6.27 KF3 39,547 unknown		A	-24325-25862.5	1.5e-6	557	1.5e-5	5.4e-5	5.9e-10	2.5
		B	-24325-25862.5	1.2e-5	1375	1.2e-5	5.7e-5	1.3e-7	6.6
		C	-24325-25862.5	3.2e-7	586	3.1e-6	1.4e-6	4.2e-11	2.4
		D	-24325-25862.5	9.7e-7	1381	9.7e-7	3.5e-6	1.2e-7	6.0
		E	-24325-25862.5	0.019	177084	0.01	0.61	NA	160.5
		F	-24325-25862.5	3.5e-7	NA	5.1e-7	6.7e-7	NA	8.9
LV_Petsc_ 6.26 KF4: 127,169 unknown		A	1.2e-3 - 52992	2.4e-5	790**	2.4e-5	2.4e-4	5.7e-9	13.6
		B	1.2e-3 - 52992	1.2e-5	4048*	1.3e-5	2.3e-4	5.2e-9	63.8
		C	1.2e-3 - 52992	2.7e-4	848**	4.2e-4	2.2e-4	1.4e-8	13.67
		D	1.2e-3 - 52992	1.36e-6	4404*	1.2e-6	1.9e-5	5.1e-10	63.94
		E	1.2e-3 - 52992	41.8	353151	41.8	1218	NA	1322.5
		F	1.2e-3 - 52992	1.18e-6	NA	8.9e-7	5.7e-6	NA	9.3
LV_Petsc_ 6.27 LMCA Arteries (um Scale 1e-6 tolerance) 84,216 unknown		A	1.8e-4 - 7.9	5.2e-7	127*	9.4e-7	1.6e-6	4.2e-4	7.74
		B	1.8e-4 - 7.9	9.9e-7	293*	9.9e-7	1.2e-5	1.0e-4	21.1
		C	1.8e-4 - 7.9	5.8e-7	120*	9.5e-7	2.9e-7	1.9e-4	1.24
		D	1.8e-4 - 7.9	9.9e-7	293*	9.9e-7	1.2e-5	1.0e-4	3.37
		E	1.8e-4 - 7.9	0.0032	707250	0.003	0.007	0.09	1257
		F	1.8e-4 - 7.9	6.6e-11	NA	1.1e-10	1.8e-10	NA	0.05

Solver: GMRES Binary:

Name	Flow range (fL/s)	Max norm of residual (mmHg)	Number of iterations	Max spot check error (fL/s)	Overall balance error (fL/s)	Time to solve (s)
LV Petsc 6.31 Full LMCA:1,323,888 unknowns	-2.8e8 – 2.8e8	0.2	2171	0.018	0.68	506
LV Petsc 6.32 Average Density Full LMCA: 606,765 unknowns	-3.1e8 - 3.1e8	0.17	719	0.17	0.43	76.1
LV Petsc 6.32 Highest Density hemisphere: 1,437,121 unknowns	-1.3e9 - 1.3e9	0.19	986	0.026	3.8	248.83

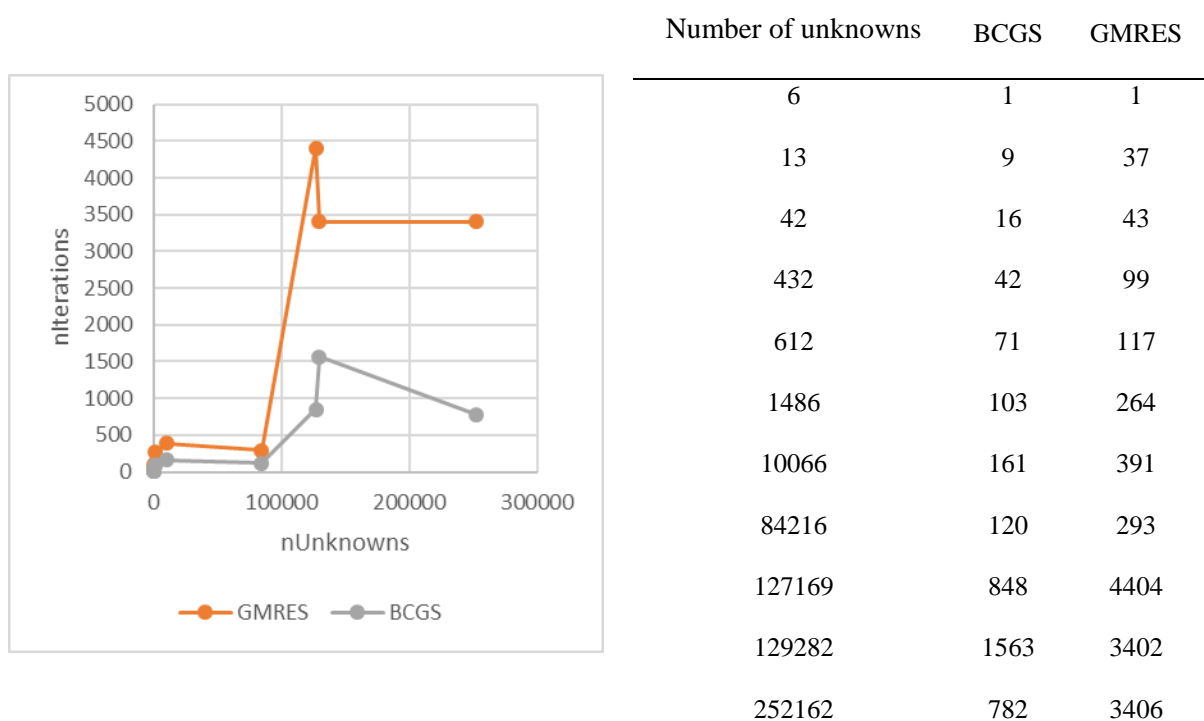
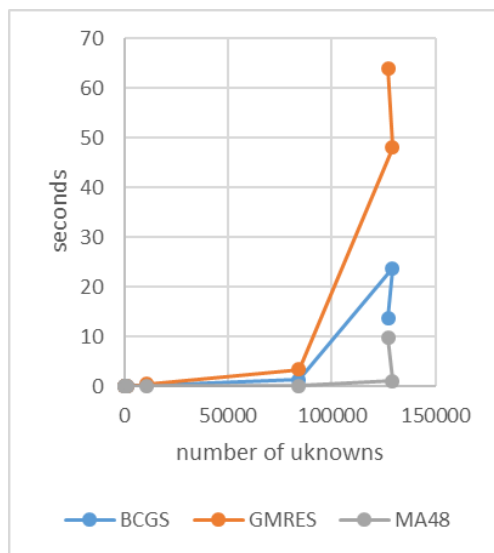


Figure 7.151. The comparison of different iterative solving algorithms with PETsc Block Jacobi preconditioner.

7.20.7 Time to solve networks

It is also important to assess the CPU time to solve linear algebraic systems. This evaluation is more than merely the number of iterations, as each iteration can vary in time to evaluate. This section investigates the runtime of the solver I many conditions. The findings indicate MA48 is

the shortest (known to hold true up to ~200,000 elements, where this method breaks down). The second fastest method is BCGS, the third is GMRES and the slowest is Gauss-Seidel (not shown).



Number of unknowns	BCGS	GMRES	MA48
6	0.001	0.001	1.27E-04
13	0.001	0.002	5.62E-05
42	0.002	0.002	6.16E-05
432	0.005	0.01	2.24E-04
612	0.009	0.013	3.51E-04
1486	0.02	0.054	0.0022
10666	0.168	0.444	0.024
84216	1.24	3.37	0.047
129282	23.73	48.02	1.05
127169	13.67	63.94	9.76

Figure 7.152. Comparison of the runtimes for different size systems with different solvers. Note, BCGS and GMRES use binary file transfer and use Block Jacobi preconditioning.



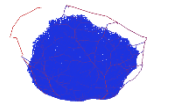
7.20.8 Using a two-step solving method for lowering residuals

For more difficult problems, the convergence required PETSc preconditioners and the infinity norm of the residual vector was larger than the direct method (ma48). In order to lower the residual, the solution vector can be obtained from a 2-step solving technique. In this method, an initialization vector is obtained using BCGS with PETSc preconditioning. This vector is used to initialize GMRES without any conditioning to solve a final solution. This avoids errors generated by the

blocking algorithm in the Block Jacobi preconditioner and still allows the GMRES to converge (it would diverge without an initialization or preconditioner). Tolerance is set to 0.2. The equations of conservation are given below.

Findings: The 2-step method allowed expansion of the equation set to over 1,000,000 unknowns adequately. GMRES that was initialized with BCGS + Block Jacobi preconditioner was able to accurately and efficiently solve the system. GMRES with an initialization using GMRES+Block Jacobi preconditioning did not benefit from the second polishing step.

$$0 = \vec{\nabla} \cdot \left(\frac{1}{\alpha} \Delta p \right), \quad f = \frac{1}{\alpha} \Delta p \quad (7.365)$$

Linear Vasculature: A) BCGS with PC; B) GMRES, initialized without PC; C) ma48;								
	Structure with coloration in flow (ml/min)		Flow range (fL/s)	Time to solve (s)	Max norm of residual (mmHg)	Number of iter- ations	Max spot check error (fL/s)	Overall balance error (fL/s)
LV_Petsc_6.33		A	0.19- 5.0e9	30	0.83	782	1.01	0.44
Sparse LMCA: 252,162 unknowns		B	0.19- 5.0e9	0.11	0.10	4	0.21	0.66
		C	0.19- 5.0e9	NA	0.24	NA	0.35	0.23
LV_Petsc_6.34		A	0.02- 4.8e9	77.5	1.96	911	2.9	0.25
Medium LMCA: 500,166 unknowns		B	0.02- 4.8e9	2.43	0.26	35	0.21	0.47
		C	0.02- 4.8e9	56.0	0.2	NA	0.3	0.22
LV_Petsc_6.35		A	0 – 1.0e10	323	4.1	972	4.2	0.97
Full LMCA: 1,158,040 unknowns		B	0-1.0e10	0.76	0.40	5	0.86	0.33
		C	NA	NA	NA	NA	NA	NA

With this combination, the solvers have achieved a sub-analytic solution for residuals, yet the overall balance seems to be slightly worsened.

7.20.9 Solving Meshes

7.20.9.1 Diffusion Problem with Coarse-Grid Interpolation

This section details initialization vectors and the increased performance due to interpolation in the diffusion problem. The solver used is listed, with binary reading/writing and the preconditioner was a Block Jacobi. Tolerance is set to 1e-6 in all cases. The equations of conservation are given below.

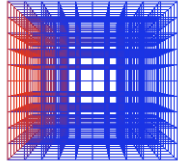
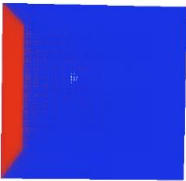


Findings: The GMRES solver benefits from the interpolation while the BCGS does not. Both methods solve adequately for simple diffusion problems.

Executable filename: *petscInitializationApp.exe*

Button name: *test Initialization Time Benefit Medium Mesh*

$$0 = \vec{\nabla} \cdot (D \nabla c) \quad (7.366)$$

A) BCGS , B) GMRES and C) Gauss Seidel.

Grid	Visualization		Values of Pressure At the Volume (mmHg)	Time to solve (s)	Itera- tions to solve	Max Norm of residual (mol)	Max spot check error (mol/min)	Overall balance error (mol/ min)
<u>D_Petsc_9</u> <u>.1</u> 10x10x10		A	0-100 0-100	0.004	15	5.3e-11	5.4e-11	4.7e-10
		B	0-100 0-100	0.007	23	4.1e-11	4.1e-11	8.7e-10
		C	0-100	0.005	151	2.9e-5	2.9e-5	0.008
<u>D_Petsc_9</u> <u>.2</u> 50x50x50 initialized		A	0-100 0-100	0.903	51	7.7e-11	7.7e-11	2.2e-7
		B	0-100 0-100	1.7	101	5.8e-11	5.8e-11	8.6e-8
		C	0-100 0-100	5.0	1229	6.3e-6	6.3e-6	0.197
<u>D_Petsc_9</u> <u>.3</u> 100x100x 100 initialized		A	0-100 0-100	11.09	82	6.7e-11	6.7e-11	5.7e-6
		B	0-100 0-100	22.8	174	9.1e-11	9.1e-11	2.1e-6
		C	0-100 0-100	48.3	1437	3.1e-6	3.1e-6	0.28
<u>D_Petsc_9</u> <u>.4</u> 100x100x 100		A	0-100 0-100	15.7	119	8.1e-11	8.1e-11	1.5e-7
		B	0-100 0-100	37.24	287	8.0e-11	8.0e-11	4.2e-6
		C	0-100 0-100	344.9	10520	3.1e-6	3.1e-6	0.81

7.20.10 Large meshes

Large meshes are memory-prohibitive for direct solvers and time-prohibitive for commercial solvers (Matlab, for instance). To test the robustness of the PETSc solvers at these large meshes (>10 million unknowns), many case studies of a simple diffusion simulation were implemented using the Cartesian mesh implementation discussed in Section 7.21. The findings indicate the solver is not limited by size and the combination of BCGS + Block Jacobi can converge blocks up to 217 million elements (largest size the computer can handle in memory).

Name	Number of elements	Discretizations per side	Time to solve (hrs)	Memory requirement (GB)	Number of iterations
<u>D Petsc 10.1</u> 450	92,340,000	450	8.0	65.7	2469
<u>D Petsc 10.2</u> 500	126,500,000	500	13.6	90.1	2949
<u>D Petsc 10.3</u> 600	218,160,000	600	33.3	>160.0	4003

7.20.11 Solving Dual-mesh problems

7.20.11.1 Mass Transfer Problem with Coarse-Grid Interpolation

This section details initialization vectors and the increased performance due to interpolation in the diffusion problem. The solver used was GMRES and the preconditioner was a composite. On these small systems, it appears that the read/write time is longer than the solving time, so the difference in solving time is negligible. Tolerance is set to $1e-6$ and the method for file transfer is binary. PETSc preconditioner was used (Block Jacobi). The executable used for this case study was *petscInitializationApp.exe*. Further case studies exemplified how a coarse-grid interpolation can solve previously unsolvable dual-mesh problems. See Section 7.21.8 for more information on the interpolation method.

Findings: BCGS diverged in all cases. GMRES converged in all cases and initialization with interpolation lowered the overall solving time. In the event of even larger cases, such as a 3x3x1 Kleinfeld dataset converging on a 50x50x50 mesh, the solution diverges with all solvers unless an iterative refining process is implemented. The equations of conservation are given below.

In vasculature: (linear flow):

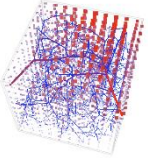
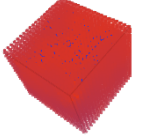

$$0 = \vec{\nabla} \cdot \left(-\frac{1}{\alpha} \Delta p \right), \quad f = -\frac{1}{\alpha} \Delta p \quad (7.367)$$

In tissue: (concentration)

$$0 = -\vec{\nabla} \cdot (D \nabla c_A) - UA_{vasc} \Delta c_A - k_1 c_A V_{tiss} \quad (7.368)$$

In vasculature: (concentration)

$$0 = \vec{\nabla} \cdot (f c_A) + UA_{vasc} \Delta c_A \quad (7.369)$$

	Visualization	Time to solve (s)	Iterations to solve	Max residual error (mol)	Total mass transfer (mol/s)	Overall balance error (ml/s)
<u>MT_Petsc_5.1</u> 10x10x10		0.06	100	9.9e-8	3.7e8	8.3e-7
<u>MT_Petsc_5.2</u> 20x20x20 initialized		0.345	182	2.0e-7	1.2e7	3.7e-7
<u>MT_Petsc_5.3</u> 50x50x50 initialized		11.4	483	2.9e-7	1.0e7	1.2e-5
<u>MT_Petsc_5.4</u> 100x100x100 initialized	NA	238.8	1056	1.2e-7	1.3e6	1.1e-5
<u>MT_Petsc_5.5</u> 100x100x100	NA	255.5	1145	3.3e-7	1.3e6	3.6e-5



Without preconditioners all systems diverged. With first solving using a preconditioner then solving using the coarse solution as an initialization vector, the results indicate solvability in both cases and a minor benefit from the interpolation. Many larger cases (KF1 and 300x300x300 mesh for instance) require multiple coarse-grained interpolations before the full matrix can be solved (data not shown). Tolerance was set to $1e-7$.

7.20.11.2 Splines

No multiscale operations (coarse grid interpolation) were performed on splined networks because the structural conversion of 2-pt face matrices to splines already reduces flow and pressure calculation to the smallest possible size. In other words, splined networks already have minimum size and no further multiscaling techniques are needed.




7.20.11.3 Using connection vs boundary conditions

A simple case study was conducted to investigate whether using boundary conditions (and thus reducing the number of unknowns) improved the solution. This can be accomplished by solving an arterial tree and venous tree (disconnected, with many terminals) and solving the connected network (two boundary conditions but more equations) and comparing the results. The findings indicate the introduction of boundary conditions in lieu of equations does not greatly affect the solvability of the matrix. Results reflect BCGS with PETSc preconditioning. The mathematics are the same as in Section 7.20.6.

	Visualizati on	Flow Values [<i>min flow</i>] (fL/s)	Max Norm of residual PETSc [MA48] (mmHg)	Max spot check error PETSc (fL/s)	Overall balance error PETSc (fL/s)	Max difference between solution vectors (mmHg)
Sparse Microcirc 15085 unknown s		-1.9e+8 - 4.7e+8 [5.3e-2]	4.2e-1 [2.3e-2]	4.2e-1	5.1e-1	4.7e-8
Sparse Microcirc (connected) 22086 unknown s		-3.2e+8 - 3.2e+8 [1.5]	8.1e-2 [4.2e-3]	8.2e-2	6.4e-2	5.6e-8

7.20.11.4 Investigation of units on solver

To investigate the effects of coefficient scaling in the matrix (to prevent bad conditioning simply from data storage methods), two versions of the same network were solved using micron (10^{-6} meters) units and millimeter (10^{-3} meters) units. Results indicate units do not make a significant difference in the solving, yet the residual must be set accordingly to accommodate this shift in units. Results reflect BCGS with PETSc preconditioning.

		Visual- ization	Flow Values [<i>min flow</i>] (fL/s)	Max Norm of residual PETSc [MA48] (mmHg)	Max spot check error PETSc (fL/s)	Overall balance error PETSc (fL/s)	Max difference between solution vectors (mmHg)
Highly Sparse Microcirc (mm) 7146 unknowns	M 18		-3.2e-1 – 3.2e-1 [1.8e-7]	2.4e-10 [1.4e-11]	2.3e-10	2.3e-10	3.7e-8
Highly Sparse Microcirc (um) 7146 unknowns	M 19		-3.2e+8 – 3.2e+8 [1.8e+2]	4.1e-1 [1.4e-2]	4.1e-1	4.2e-1	3.6e-8
Highly Sparse Microcirc (um+ splined) 4048 Unknowns*	M 20		-6.0e+7 – 6.0e+7 [2.4]	4.7e-2 [3.0e-3]	4.7e-2	6.9e-2	4.8e-8

**

*** uses Mahsa's viscosity for splines, not the viscosity from the 2-pt meshes

7.20.12 Testing Different Solvers

This report highlights the use of BCGS and GMRES in PETSc. Other work was investigated in solving the linear vascular network with >1,000,000 unknowns using many other solvers and preconditioners independently and as pairs. The results that did not improve the solution or solving time were not reflected in this report.

7.21 Appendix U: Cartesian Mesh

This document summarizes how to implement and benefit from a Cartesian mesh in place of the frequently used unstructured (tetrahedral) mesh. Benefits from using a Cartesian mesh compared to an unstructured mesh include:

- Direct calculation of closest volume to a 3D point coordinate, as opposed to searching
- Fixed equation bandwidth of 7 elements per equation in the balance matrix
- Direct calculation of transport equations without necessitating memory-intensive mesh data structures
- Excellent skewedness factor for all mesh elements (uniformly set to 1 or 0 for all faces)

7.21.1 Creating a Cartesian mesh

A Cartesian mesh consists of connectivity information and point coordinates that describes the mesh cells. When making a Cartesian mesh, these data structures can be calculated by simply offering a number of divisions in each dimension ($nVolX$, $nVolY$, and $nVolZ$ for x , y , and z dimensions respectively) and the bounds of the whole mesh (upper and lower limit in each dimension). Note, these divisions describe the interior volumes of the mesh. Additional boundary edges must be generated after the mesh is set.

Point coordinates. The point coordinates can be calculated directly with the knowledge that they divide each dimension between the minimum and maximum coordinate by $nVol+1$ divisions. To calculate all points, it is recommended to have a triple nested for loop and calculate each element of the point coordinates following the example:

$$pt_x(i) = x_{min} + i \frac{(x_{max} - x_{min})}{nVolX}, \quad i = 0..nVolX + 1 \quad (7.370)$$

Points for cells. The points for each cell are directly accessible, as they follow the same pattern in each cell. This means that each cell (denoted by the i , j , and k index in each dimension) has a directly accessible set of points:

$$ptA(i, j, k) = \begin{bmatrix} pt_1 \\ pt_2 \\ pt_3 \\ pt_4 \\ pt_5 \\ pt_6 \end{bmatrix} = \begin{bmatrix} pt(i-1, j-1, k) \\ pt(i-1, j-1, k) + 1 \\ pt(i-1, j, k) \\ pt(i-1, j, k) + 1 \\ pt(i, j, k) \\ pt(i, j, k) + 1 \\ pt(i, j-1, k) \\ pt(i, j-1, k) \end{bmatrix} \quad (7.371)$$

Where i , j , and k are the volume indices in each dimension, pt_i is the point index for the i^{th} point in the cell, and the $pt()$ operator converts the i , j , and k point indices in each dimension into a global point index following the example in Equation (7.370).

Faces. The connectivity between each cell is denoted as a *face* which can be calculated by knowing the points. One important note is that all cells should only compute the first 3 faces (of 6 in the hexahedron), to avoid duplication of faces. The unmatched faces on the last row of cells will be calculated as a last step. The face information calculation simply requests the point indices for each cell, selects a face index in the cell (between face 1 and face 6) and retrieves the 4 points from ptA that correspond to the face in question.

Cell connectivity. The cell connectivity associated with a given face can also be directly calculated, as the current cell adjoins itself with a cell with 2 of 3 indices matching. In other words, the adjoining cell has the same i , j , and k index, but one value is shifted by 1 ($i\pm 1$, $j\pm 1$, or $k\pm 1$).

7.21.2 Creating a Cartesian mesh in memory

Because the simulation meshes can be extremely large (hundreds of millions of mesh cells), simply holding the data structures in memory can overwhelm modern workstations. A Cartesian mesh is capable of generating all relevant equations for finite volume simulations without ever instantiating mesh data structures. This is done by storing the relevant information (domain bounds, discretization elements, number of total volumes, etc.) which amount to <1MB of information and refraining from generating other data structures. Note, this data overhead (<1MB) is not affected by the size of the mesh (nVolumes). As the equations were calculated above to fill the data structures, they can be calculated on-the-fly to generate equations. This facilitates rapid equation generation.

7.21.3 Making flux balances without mesh data structures

This function uses the local variable to assign a matrix with isotropic, homogenous diffusivity throughout the Cartesian mesh, but could be expanded to anisotropy in the future if need be.

The interior volumes are simply the value of the local cell and the divergence in all 3 dimensions ($i \pm 1$, $j \pm 1$, and $k \pm 1$). Using the *getIJKfromGlobalCellIdx* procedure, the evaluation of the divergence for any cell is straightforward. This is further enhanced by the constant distance

between cells in a single direction, i.e. $x(i-1) - x(i)$ is constant for all i . An example for finite volume discretization in the x direction is given:

$$\frac{d^2 c_i}{dx^2} = \frac{D_{i-1} + D_{i+1} - 2D_i}{dx} \quad (7.21.372)$$

Because the exterior volumes lie on the boundary faces of interior cells, they are only ½ as far from the cell center as a full volume-volume interface (see Section 7.14 on discretization schemes). Due to this, when calculating the flux to a boundary cell, the distance is halved.

7.21.4 Connecting a 2-point network with a mesh

One important benefit of a Cartesian mesh, is the direct access to correlate any point coordinate with a surrounding mesh cell. This can be done by executing the following equation:

$$L_x = x_{max} - x_{min}$$

$$i = floor\left([pt_x - x_{min}] * \frac{nVol_x}{L_x}\right) \quad (7.21.373)$$

This example for the x-dimension can be repeated likewise for y and z dimensions to retrieve the i, j , and k coordinates of the cell surrounding the point in question. Using this methodology, a vascular network structure can be connected to a Cartesian mesh without searching each volume in the mesh.

7.21.5 Splitting the network to match length scale of mesh (vascular re-segmentation)

In order to ensure that the network and the mesh are on the same scale, it may be necessary to cut the vasculature into smaller segments. This is an even more important task in relationship to the oxygen simulation, where reactions strongly affect the distribution throughout the network. The algorithm was designed with the basis of a Cartesian mesh implemented at runtime (not from file) and does not work for a mesh read from file. The algorithm flow chart is given below. Examples of network centerlines before and after segmentation are also offered in Figure 7.154.

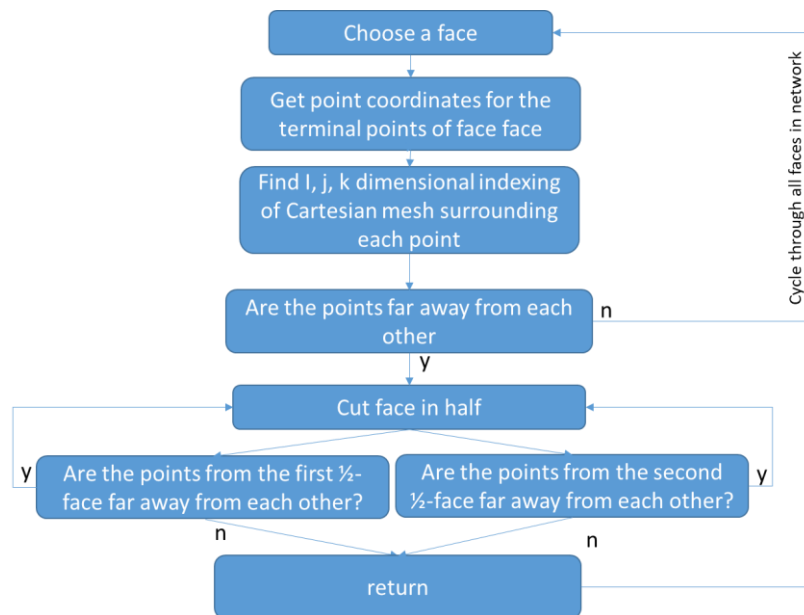


Figure 7.153. Design of the recursive algorithm that splits the network into smaller segments until the criteria of being on the same scale as the mesh is confirmed.

This is only one of two methods to accomplish this goal, the other being to choose the lengths of the segments such that they are close to the order of volume as the tissue elements.

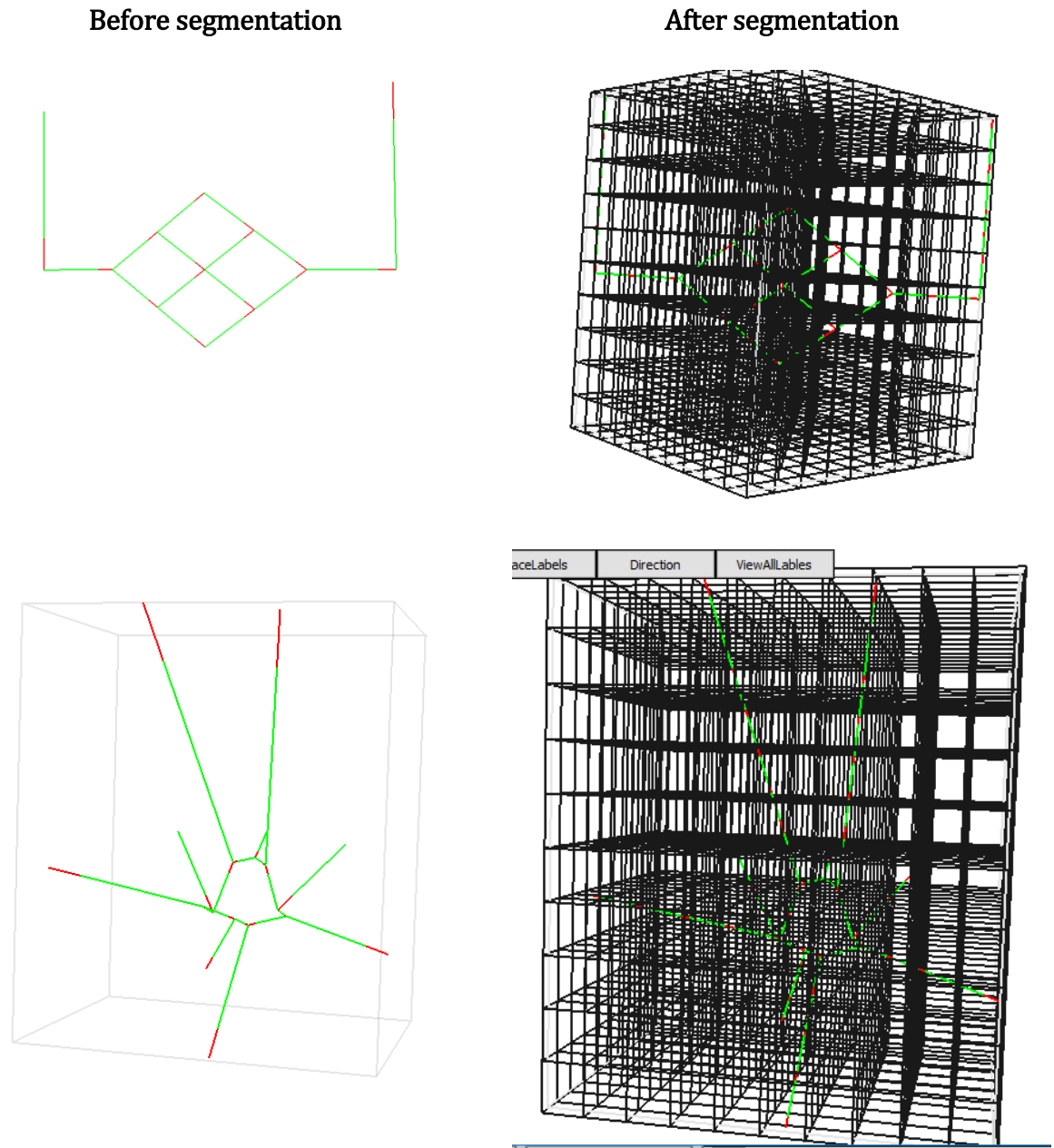


Figure 7.154. The re-segmentation algorithm clearly increases the vascular segment density without altering the segment structure. This allows matching of the mesh resolution to the network resolution automatically.

7.21.6 Benefits from using mesh without data structures

The use of a Cartesian mesh without the cumbersome mesh-related data structures saves a significant amount of time (to generate the matrices) and memory as described by Figure 7.155.

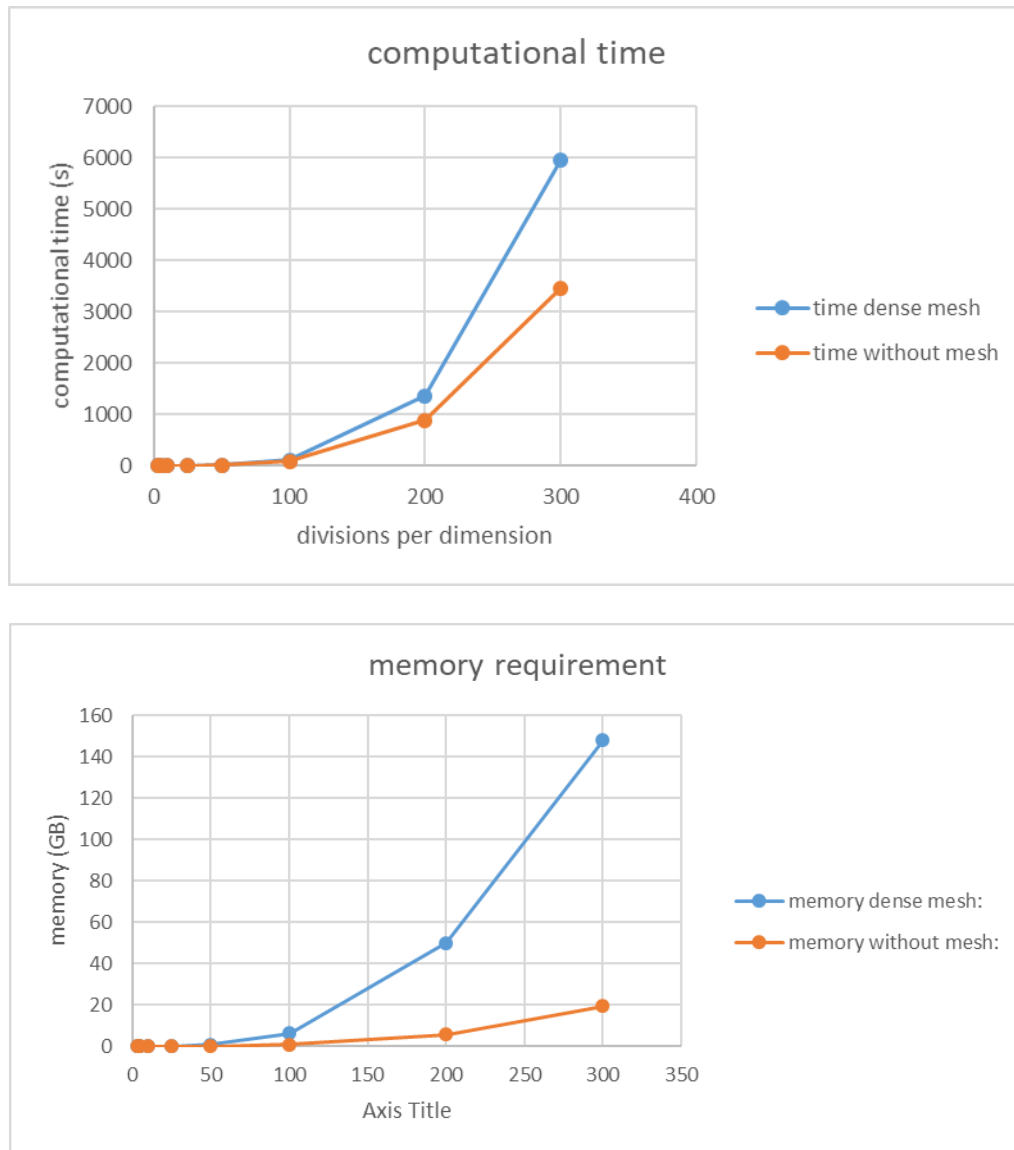


Figure 7.155. Comparison of computational effort to solve a simple diffusion problem with all mesh data structures (dense mesh) and with only minimal data structures (without mesh). The memory requirement is drastically lower and computational time significantly lower for the mesh using a hashing algorithm instead of storing the data structures.

7.21.7 Validation of mesh from memory compared to dense mesh

The validation of dense and lean mesh solving is pre-programmed as a function in the project *ghSSSinglePhasicOxygenValidation.dproj* using the button entitled *validate diffusion without mesh*. Another installment is given in *DualDarcyIterative.dproj* in a button entitled *testMeshFromMemoryDiffusion*. This button will verify that the flows, residuals, mass transfer, and solution vectors are equivalent in the two types of meshes.

7.21.8 Interpolating a coarse-mesh guess for initializing solvers for fine-grid results

A Cartesian mesh is capable of directly computing the index of a volume surrounding a point coordinate (without the need for searching the mesh and comparing distances). This allows an efficient implementation of investigating the coordination of a coarse mesh with a refined counterpart. Due to this enhancement, a method for interpolating a coarse-grained simulation result to obtain a reasonable initial guess for a fine-grained mesh has been developed.

The goal is to create a better initial guess for help converging within the iterative linear algebraic solvers. Note, exterior cell values are simply copied from the coarser mesh, however the interior mesh elements undergo an interpolation.

For an eight-node trilinear hexahedron element, the linear interpolation shape function, in local co-ordinate system (ξ, η, ζ) , can be obtained from:

$$\begin{bmatrix} N_1 \\ N_2 \\ N_3 \\ N_4 \\ N_5 \\ N_6 \\ N_7 \\ N_8 \end{bmatrix} = \frac{1}{8} \begin{bmatrix} (1-\xi)(1-\eta)(1-\zeta) \\ (1+\xi)(1-\eta)(1-\zeta) \\ (1+\xi)(1+\eta)(1-\zeta) \\ (1-\xi)(1+\eta)(1-\zeta) \\ (1-\xi)(1-\eta)(1+\zeta) \\ (1+\xi)(1-\eta)(1+\zeta) \\ (1+\xi)(1+\eta)(1+\zeta) \\ (1-\xi)(1+\eta)(1+\zeta) \end{bmatrix} \quad (7.374)$$

Where $\xi, \eta, \zeta \in [-1, 1]$. The comparison of point indices and the local coordinate system is pictured below:

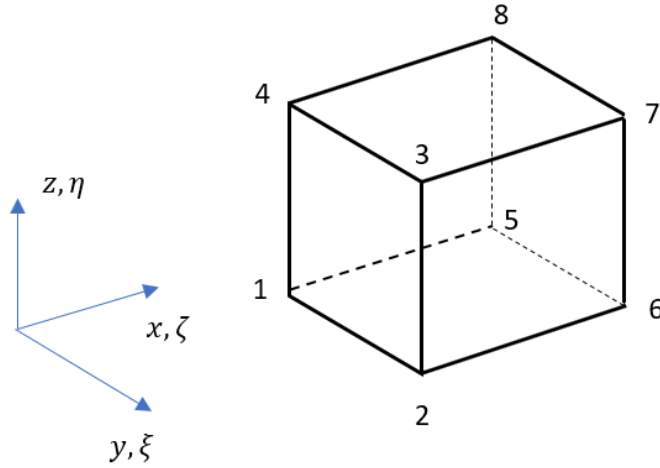


Figure 7.156. Global and local co-ordinate system for the hexahedron element.

Given that the functional values at the neighboring cell centers, φ_i , are known, $\varphi(\xi_0, \eta_0, \zeta_0)$ for a set of coordinates (ξ_0, η_0, ζ_0) lying inside the hexahedron nodes can be interpolated using:

$$\varphi(\xi_0, \eta_0, \zeta_0) = \sum_{i=1}^8 N_i \varphi_i \quad (7.375)$$

For a set of global co-ordinates, the values can be converted to local coordinates using:

$$\xi_0 = \frac{2y_0 - y_1 - y_7}{y_7 - y_1}, \quad \eta_0 = \frac{2z_0 - z_1 - z_7}{z_7 - z_1}, \quad \zeta_0 = \frac{2x_0 - x_1 - x_7}{x_7 - x_1} \quad (7.376)$$

These new values are an interpolated form of the coarse vector which can be used to initialize an iterative linear algebraic solver for more stable and efficient convergence.

7.22 Appendix V: Illustration of diffusion

Diffusion from Fick's law is defined as the divergence of the flux:

$$f = -D \frac{\partial \phi}{\partial x} \quad (7.377)$$

$$\frac{\partial \phi}{\partial t} = -\frac{\partial}{\partial x} \left(-D \frac{\partial \phi}{\partial x} \right) \quad (7.378)$$

The flux is proportional to the negative gradient of the state scaled by a material property, D . The divergence is the acceleration of the state. At steady state, the acceleration (or curvature, also known as accumulation) is 0. In a dynamic simulation, this value is nonzero.

Take a simple example of a dynamic diffusion (without reaction or convection) with a non-steady profile initialization as in Figure 7.157. The final and initial profiles do not match, indicating some accumulation must take place at the nodes. At node 3, the flux vector (flow from high to low state) is positive, yet the gradient (dy/dx) is negative. This is the physical explanation behind the negative sign in the flux calculation (Equation (7.377)). The divergence leads to accumulation. From the flux vector it can be seen that node 3 will lose moles and node 4 will gain mass. The flux at node 3 is now positive, yet the node will lose mass, expressing that divergence is the negative of the gradient of the flux.

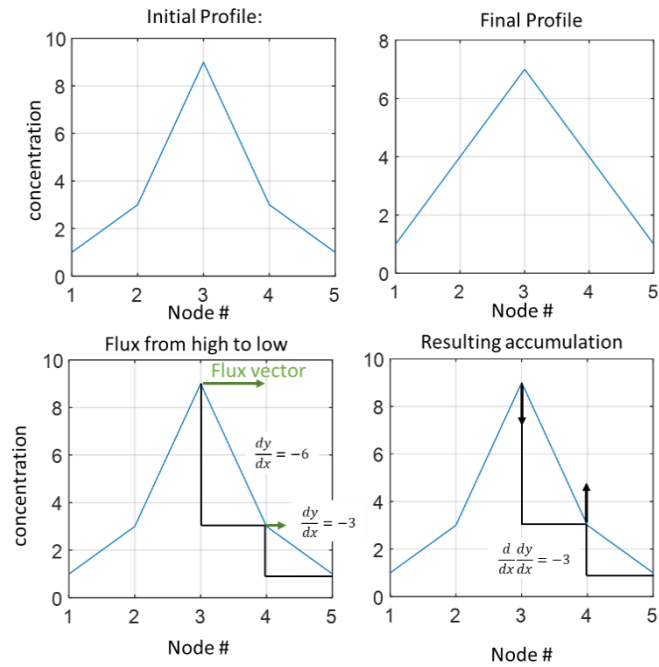


Figure 7.157. Evaluation of transport flux and divergence on unsteady concentration profile in 1-dimension.

The flux vector at node 3 (evaluated in the direction of node 4) is positive, yet the gradient is negative. The accumulation at node 3 in response to this flux is negative, although the flux is positive.

7.23 Appendix W: Starling law and revised Starling law

7.23.1 Summary

The water exchange across a semi-permeable membrane is driven by pressure and osmotic differences as modeled by Starling's law [251]. In clinical practice, the osmotic gradients across the membrane are influenced by the coupling between transmembrane water flux and osmolite permeability.

It has been speculated that the osmolite mass transfer (a diffusive-like process) is coupled with the convective transfer of osmolite across the membrane due to bulk water transfer (a convective process). The bulk fluid and solute exchange across a membrane are two coupled diffusive-convective processes that can be evaluated analytically (using simplifying assumptions) to create a highly nonlinear function of solute flux known as the revised Starling's law [252]. This method is used to explain why fluid flux across a membrane (such as the blood brain barrier) cannot be reversed using hyperosmolarity treatment.

This section lays the mathematical foundation of the simple Starling Law [251] and the highly nonlinear coupling of the Revised Starling Law [252]. The section demonstrates that even large osmolarity gradients cannot reverse the sign of the water flux across the membrane driven by pressure differences when using the revised Starling law model. A detailed case study on the inability of osmolarity to reverse the flow is also offered. The results are summarized in a key figure (Figure 7.166).

This section concludes that regions of filtration and reabsorption in the brain are mainly determined by the direction of the hydrostatic pressure differences and that osmolarity gradients that may occur can only weaken or enhance the magnitude of the flux but cannot reverse the sign.

Accordingly, the relationship between tissue pressure and vascular pressure is the key for filtration and resorption. These preliminary conclusions, however, have not been affirmed by experimental data and require future studies. The future direction of this project is to expand the current implementation of the revised Starling law in a 3 dimensional tissue embedded within a 3 dimensional vascular network.

7.23.2 Introduction

The Starling principle claims water transport between the capillary bed and the interstitial fluid (ISF) is driven by hydrostatic pressure (pressure generated by physical force) and osmotic pressure (pressure generated by a concentration gradient). Figure 7.158 expresses these effects in a blood vessel.

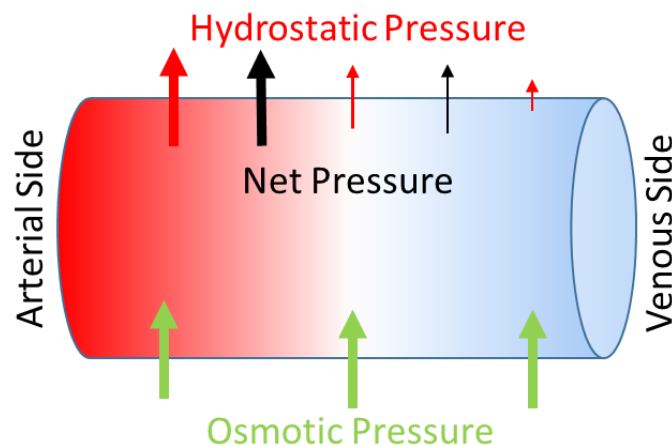


Figure 7.158. Opposing pressures that exist in a system with a semi-permeable membrane and a pressure gradient.

(Red) The hydrostatic pressure opposes (green) the osmotic pressure and (black) the net pressure is the result of summing these two forces together. Note, the pressure gradient gets smaller when approaching the venous side, because the osmotic pressure remains unchanged and the hydrostatic pressure gradient reduces.

7.23.3 Methods

Starling Law. The classic Starling law expresses the flux of water (J_w) as a linear combination of the hydrostatic pressure gradient (ΔP) and osmotic pressure gradient ($\Delta \Pi$) across a semi-permeable membrane. The flux is defined by material properties such as membrane porosity, L_P , and molecular reflectivity, σ , as in Equation (7.379). Equation (7.379) frequently uses the summed osmotic pressure gradient of large macromolecules (ignoring micro-molecules and avoiding balancing the osmotic pressure of each type of molecule independently). Note, the osmotic pressure is calculated using the ideal gas law ($PV=nRT$). This model maintains the independence between the water flux and the solute flux (i.e. the solute flux is independent of any water flux through the membrane).

$$\frac{J_w}{A} = L_P(\Delta P - \sigma[\Pi_c - \Pi_i]) \quad (7.379)$$

Revised Starling Law. Another interpretation is that osmotic gradient is influenced the water flux. This constitutes a convective flow of solute by the bulk flow of water across the membrane. This theory couples the mass transfer of solute with the bulk water flow through two highly nonlinear equations (expressed through an exponential function of the Peclet number). This highly nonlinear equation is claimed to explain why fluid flux across the BBB cannot be freely manipulated by imposing steep osmolarity gradients (i.e. why water reversal is not possible by

osmotic therapy). This model allows only unidirectional flow of water based on the convective direction of flow. If the convective flow of water changes direction, the equation must be modified.

The RSL uses the concentration of the fluid inside the membrane pore in lieu of the ISF concentration as depicted in Figure 7.159. This concentration is approximated using the ratio of the solute flux to volumetric water flux (Equation (7.380)). This assumption for the concentration in the pore (Π^*) is substituted into Equation (7.379) to account for the osmotic driving force ($\Pi_c - \Pi^*$). The RSL is given in Equation (7.382). Derivations of these equations are offered in Sections 7.23.7-7.23.8. Suggested human parameters for this calculation can be seen in in Table 7.41. The corresponding values in rat kidney can be seen in Table 7.42.

$$c_i = \frac{J_s}{J_w} \quad (7.380)$$

$$\frac{J_s}{A} = -D\nabla c_s|_{x=0} + u c_s|_{x=0} = \frac{J_w(1 - \sigma) - c_i + c_c e^{Pe}}{A(e^{Pe} - 1)} \quad (7.381)$$

$$\begin{aligned} \frac{J_w}{A} &= L_P(\Delta P - \sigma^2 \left[\Pi_c \left\{ \frac{(1 - e^{-Pe})}{(1 - \sigma \cdot e^{-Pe})} \right\} \right]) \\ Pe &= \frac{J_w(1 - \sigma)/A}{D/L} = \frac{J_w(1 - \sigma)L}{AD} \end{aligned} \quad (7.382)$$

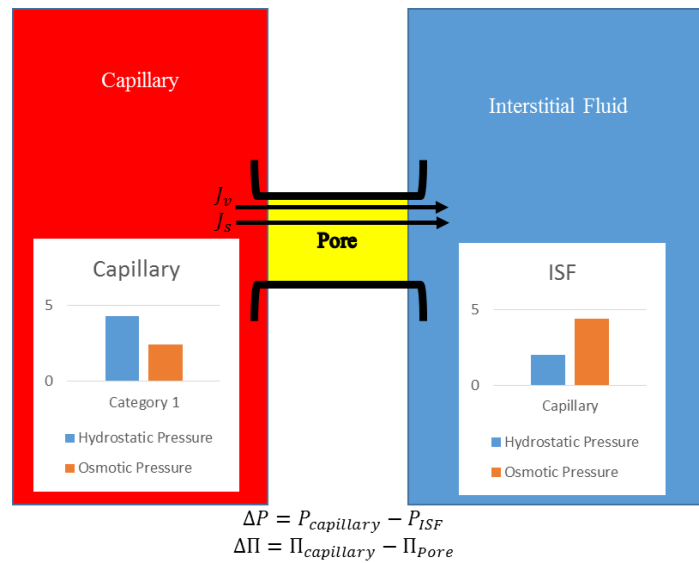


Figure 7.159. Cartoon of the a section of the BBB and the relevant states.
Note the osmotic pressure is calculated through the difference in concentration from the capillary to the pore, not from the capillary to the ISF.

Table 7.41: Suggested parameters in human for the calculations of the RSL

Parameter	Value(s)
P_c (artery)	35-45+ mmHg
P_i (vein)	15-12- mmHg
Π_c	25-28 mmHg
Π_i (steady state)	10 mmHg
Π_i (initial)	3 mmHg
σ	0-1 ($\sigma_{albumin} = 0.99$)

Table 7.42: Parameters for calculation based on rat kidney. Here, Π_p stands for Π_c .

Cortical peritubular capillaries ($\sigma_{\text{albumin}} = 0.99$)					
Fluid balance	Π_p	Π_i	P_c	P_i	$\sigma\Delta\Pi - \Delta P$
Hydropenia (mean \pm SEM)	24.8 ± 1.39	3.9 ± 0.6	13.2 ± 1.3	3.3 ± 0.6	10.0
Volume expanded (mean \pm SEM)	16.6 ± 1.34	1.3 ± 0.18	19.5 ± 2.71	11.5 ± 0.78	7.1
Ascending vasa recta of the renal medulla ($\sigma_{\text{albumin}} = 0.7$)					
Hydropenia	Π_p	Π_i	P_c	P_i	$\sigma\Delta\Pi - \Delta P$
Papilla tip (mean \pm SEM)	26.0 ± 2.3	3.7 ± 0.26	7.8 ± 0.4	6.0 ± 0.3	13.8
Papilla base (mean \pm SEM)	16.7 ± 1.3	3.7 ± 0.26	6.7 ± 0.48	6.0 ± 0.3	8.4

The column headed $\sigma\Delta\Pi - \Delta P$ lists values for the net Starling pressure favouring fluid absorption into the vessels. Data for cortical peritubular capillaries are based on four studies summarized by Ulfendahl and Wolgast.³⁷ Data for ascending vasa recta of the renal medulla are from several laboratories reviewed by MacPhee and Michel⁴³ and Michel⁴⁵

7.23.4 Results

Figure 7.160 and Table 7.43 demonstrate Equation (7.379) and Equation (7.382), showing how the revised Starling law does not reverse water flux by varying osmotic pressure.

Table 7.43: Parameters used for my recreation of the curves from the manuscript.

Parameter	Value
P_c	10-45 mmHg (dependent variable)
P_i	12 mmHg
Π_c	20 mmHg
Π_i	1.3 mmHg
AD/L	0.2
σ	0.75
L_p	1

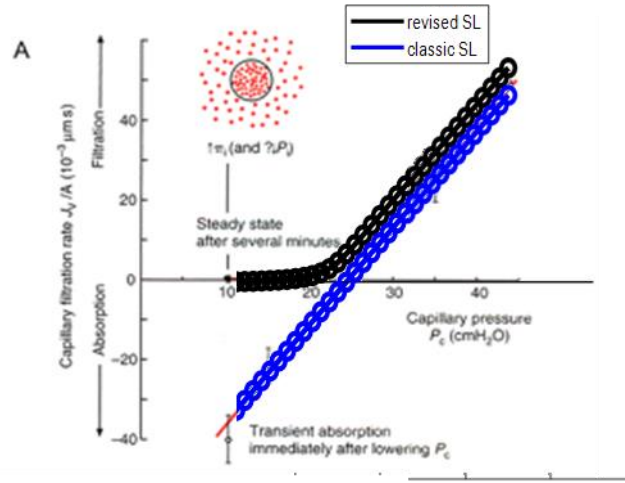


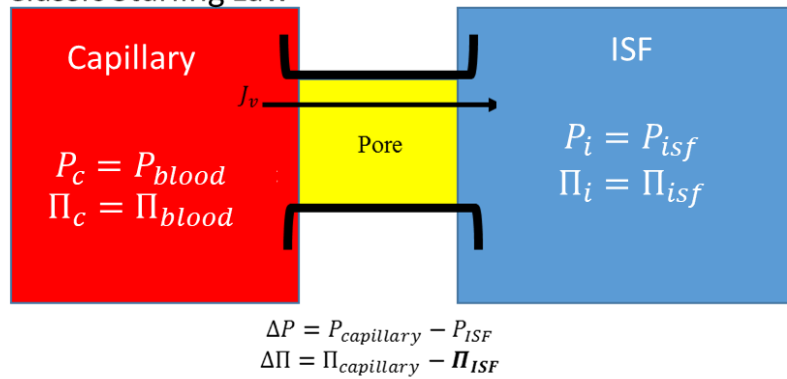
Figure 7.160. Recreated curve from [252].

The black line is the revised Starling Law (equation (7.382)) and the blue line is the original Starling Law (equation 1). We can see here that the osmotic gradient is never capable of reversing the flow of water as long as the convective flow is uni-directional.

Osmotic pressure for the pore section was calculated using Equation (7.383). This expresses the dynamic concentration as a function of water flux through the pore (hidden within the Peclet number). Values for the capillary pressure were tested and the resulting water flux (and subsequent osmotic pressure from the pore) were calculated. This can be seen graphically in Figure 7.162. The result of simulating the two systems of Figure 7.161 gives Figure 7.162. The code to create Figure 7.161 can be seen in Section 7.23.6.

$$c_i = \frac{(1 - \sigma)c_c}{(1 - \sigma e^{-Pe})} \quad (7.383)$$

Classic Starling Law



Revised Starling Law

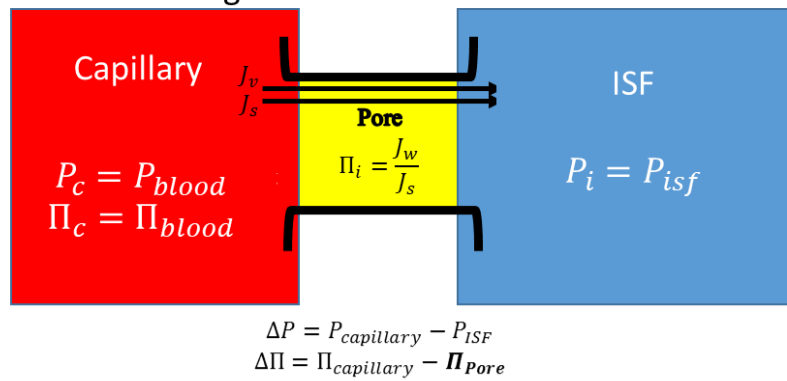


Figure 7.161. Pictographic representation of the two systems being evaluated.

Top) Revised Starling law where the osmotic pressure gradient is a function of the flux through the pore. Bottom) Classic Starling law where the osmotic pressure gradient is based on concentration in ISF.

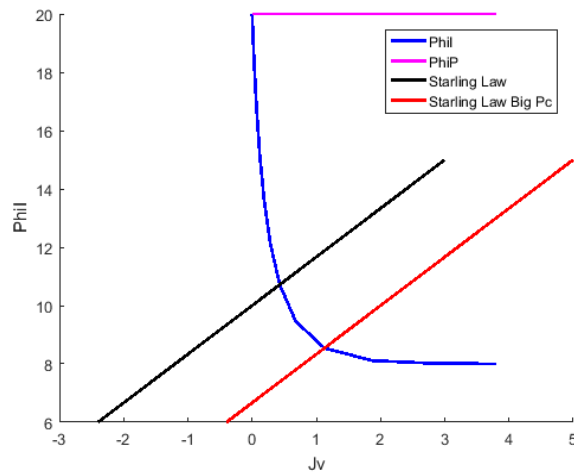


Figure 7.162. Visualziation of trends in different models in relationship to water flux and interstitial pressure.

The intersection of the ISF osmotic pressure of the (Blue) revised starling law with the (Red) original starling law for normal capillary pressure and (Black) reduced capillary pressure.

The Peclet number. The effect of convection on concentration profile can be investigated using the 1 dimensional schematic in Figure 7.163. The equations for flux can be seen in Equation (7.384), steady-state mass conservation is enforced through Equation (7.385), and the Peclet number is calculated through equation (7.386). Proposed property values are given in Table 7.44. The results are summarized in Figure 7.164.

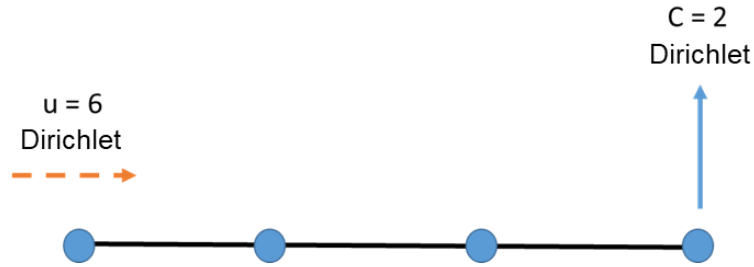


Figure 7.163. The schematic representation of the simulated system. A one-dimensional four node system with concentration as a state and flow as a convective source term along with diffusion. Each end has a Dirichlet boundary condition, left is flow set at 2 ml/min and right is concentration set at 2 mmol.

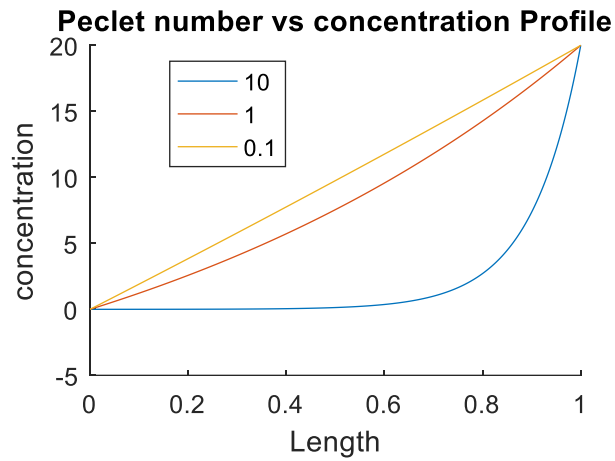


Figure 7.164. The effect of Peclet number on the concentration profile. Note that as the Peclet number rises, the distribution changes drastically.

$$J = Dc' + uc \quad (7.384)$$

$$\frac{dc}{dt} = 0 = \vec{\nabla} J \quad (7.385)$$

$$Pe = \frac{u}{D/L} \quad (7.386)$$

Table 7.44: Parameters used for my recreation of the curves from the manuscript.

Parameter	Value	Units
D (diffusivity)	0.1, 1, 10	mm ² /s
u (flow BC)	1	ml/min
L (segment length)	1	mm

7.23.4.1 Considerations of the basolateral concentrations:

When water flux tends to zero, the concentration becomes singular and when the concentration is negative, a second solution to the problem exists. Because a negative concentration is nonphysical, this situation can be remedied by the following adjustment.

$$c_i = \frac{J_s}{J_w + J_s} \quad (7.387)$$

The concentration must be positive definite, so another constraint can be imposed:

$$c_i = \text{abs}\left(\frac{J_s}{J_w}\right) \quad \text{or} \quad c_i = \sqrt{\left(\frac{J_s}{J_w}\right)^2} \quad (7.388)$$

As described above, the solution space for the revised starling law has two solutions (Figure 7.165) Once the concentration is forced to be positive-definite following Equation (7.388), the non-physiological second solution disappears.

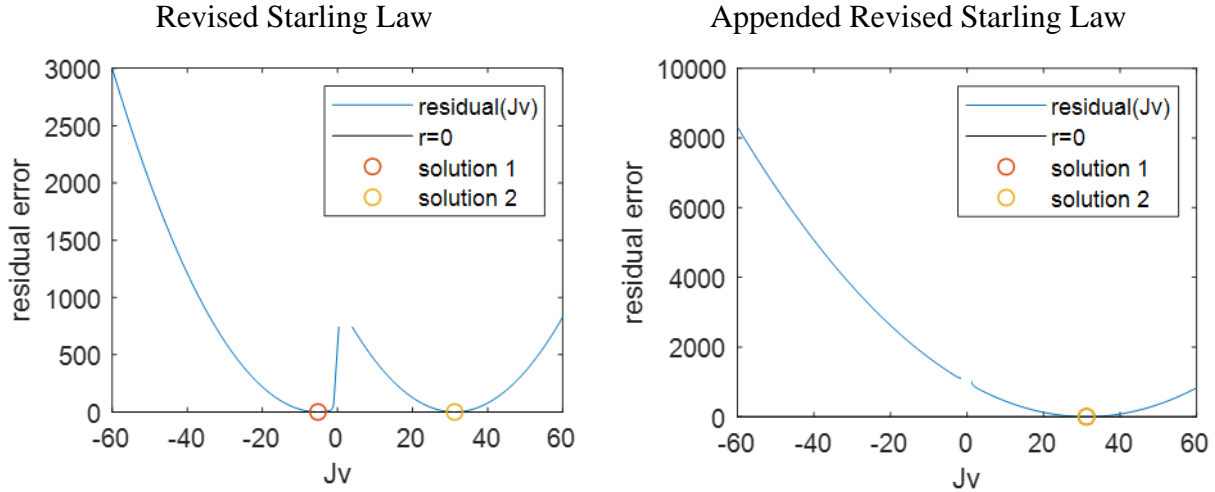


Figure 7.165. The effect of enforcing positive definite concentration for c_i . The revised starling law has a non-physiological solution when $c_i < 0$ which is eradicated in the appended form. The discontinuities in the residual exist around the singular point where $J_v = 0$.

7.23.4.2 Parametric study of water and solute fluxes for different hydrostatic and osmotic pressure gradients

To compare the effects of convection on the water transport and solute transport of the BBB membrane, 5 scenarios were compared in which the flow of water and the flow of solute are computed. These scenarios (Table 7.45) have been computed using Equation (7.379) and Equation (7.382) as seen in Figure 7.166. The solver chosen was Matlab's `fsolve` function which employs a form of the Newton method.

Table 7.45: Values used for the computation of Scenario 1-5 in Figure 7.166

Case	C_c	c_i	P_c	P_i	c_i
1	0	0	40	12	0
2	2	13	40	12	0.4
3	13	13	40	12	2.6
4	25	13	40	12	5
5	13	60	40	12	2.6

Table 7.46: Resulting values computed using equation (7.379) and (7.382) using values in Table 7.45.

Case	c_1 Potential	c_2 Potential	c_2^* Potential	Potential1	Potential2	Potential2*	Δ Potential	Δ Potential *	J _v	J _s
1	0.0	0.0.	00	40.0	12.0.	12.0	28.0	28.0	28.0	0.0
2	01.6	10.4	0.32	41.6	22.4	12.3	19.2	29.3	26.7	10.7
3	10.4	10.4	2.08	50.4	22.4	14.0	28.0	36.3	19.7	51.2
4	20.0	10.4	04.0	60.0	22.4	16.0	37.6	44.0	12.0	60.0
5	10.4	48.0	2.08	50.4	600	14.0	-09.6	36.3	16.7	51.2

The solute and water flux were comparable between the two models for the majority of cases which differ only in magnitude. At extreme concentration differences, however, the water transport differs in both magnitude and direction between the two models (Figure 7.166). The solute transport differed in many scenarios between the two models.

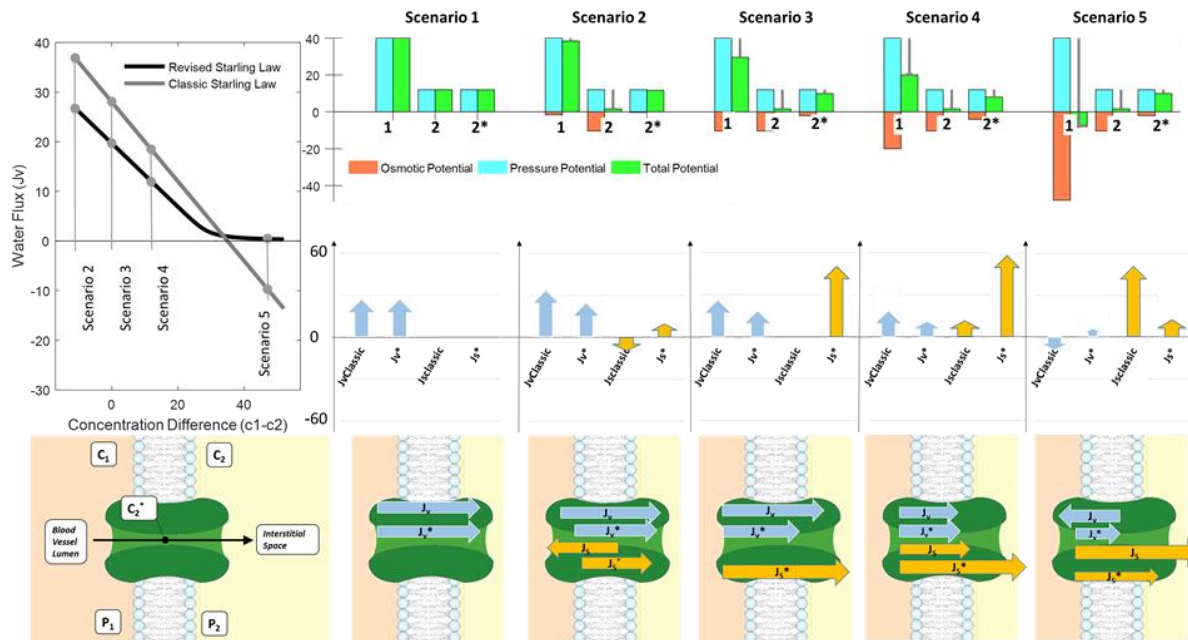


Figure 7.166. Classic and revised Starling's Law across a membrane with hydrostatic (P_1 - P_2) and osmotic (C_1 - C_2) pressure driving forces.

Here, compartment 1 refers to the capillary side of the membrane and compartment 2 corresponds to ISF. The classical hypothesis admits bidirectional water exchange. The revised Starling's law accounts for solute and water exchange coupling. It shows a progressive non-linear reduction of transmembrane flux, but does not allow for flow reversal. Top row shows conceptually hydrostatic, osmotic and total effective pressure potentials for five different scenarios (panels entitled Scenario 1-5). Middle row shows magnitude and direction of transmembrane water (blue) and solute fluxes (yellow). Scenario 5 shows countercurrent water reabsorption for the classical Starling's hypothesis, in which high osmotic gradients reverse the effect pressure potential; this reverse effect does not occur with the revised Starling's law. Bottom row redraws solute (J_s) and water flux (J_w , denoted here with J_v) direction with respect to the membrane for both classical and revised Starling's law (fluxes marked with *).

7.23.5 Discussion:

This report has been a review of the literature surrounding the classical and revised Starling law. The revised Starling law dictates that the orientation of molar flux across the blood brain barrier (BBB) is driven entirely by the hydrostatic pressure gradient based on a convection model for solute transport. This contrasts the classic Starling law, where the molar flux is driven by a linear combination of hydrostatic pressure and concentration gradients.

A simultaneous solution of the coupled nonlinear systems of equations representing flow and solute transport across the BBB was theoretically introduced and a Matlab code for implementation was provided. Conclusive figures representing numerous scenarios have also been published by our group [253].

This report does not address a 3-dimensional network exhibiting BBB transport but such a model was previously published using the classic Starling Law [253]. A discussion of the suitable finite volume discretization methods for the proposed nonlinear equations is outside the scope of this manuscript. Methods such as those described by Patankar [254] using a power law or an implementation of the exponential function using the Peclet number should be further investigated. Future work will reveal whether the revised Starling law can exhibit both filtration and reabsorption in a 3-dimensional model.

7.23.6 Sample Matlab codes

Box 7.23.1: The code used to produce Figure 7.160

```
function AQP4_Paper
syms JV, close all
warning('off','all')
Pi = 12; PhiP = 20; Phii = 1.3; A = 2; r = 0.75; Pd = 2e-1; Lp=1; D=Pd; L=1;
ci=Phii/0.4; cp=PhiP/0.4;%ci=pai/RT
for Pc = 10:45
S= solve(Pc-Pi==JV/A/Lp+r^2*PhiP*...
(1-exp(-JV*(1-r)/(Pd*A)))/(1-r*exp(-JV*(1-r)/(Pd*A))), JV);
jw = S;
New_Method(Pc-9) =jw;

pe = jw*(1-r)*L/D/A;
js = jw*(1-r)*(ci-cp*exp(pe))/(1-exp(pe));
cStar(Pc-9) = js/jw;
cStarLin(Pc-9) = js/(js+jw);
jsA(Pc-9) = js;
end

for Pc = 10:45
S= vpasolve(JV/A==Lp*((Pc-Pi)-r*(PhiP-Phii)), JV);
Old_Method(Pc-9) = S;
end
figure(2)
hold on
xlabel('Pc, capillary pressure')
ylabel('Jv, water flux')
plot(10:45,New_Method,'k','LineWidth',2)
plot(10:45,Old_Method,'b','LineWidth',2)
plot(1:55,zeros(1,55),'k');
scatter(10:45,New_Method,'k','LineWidth',2)
scatter(10:45,Old_Method,'b','LineWidth',2)
legend('revised SL','classic SL')
end
```

Box 7.23.2: The code used to generate Figure 7.162.

```
function AQP4_Paper
clf
syms JV
warning('off','all')
% Pi = 3;  Pi2 = 6;
Pc = 10; PcBig = 12; PhiP = 20; Phii = 3; A = 1; r = 0.6; Pd = 2e-1; Lp=1;
for Pi = -1:10
    loop = Pi+2;
    Pe = JV*(1-r)/(Pd*A);
    %    PhiP = Phii*(1-r*exp(-Pe))/(1-r);
    Phii=PhiP*(1-r)/(1-r*exp(-Pe));
    S= solve(Pc-Pi==JV/A/Lp+r*(PhiP-Phii),JV);
    Jv(loop) =S;

    % S2= solve(PcBig-Pi==JV/A/Lp+r*(PhiP-Phii),JV);
    % JvBig(loop) =S2;

    Pe = S*(1-r)/(Pd*A);
    Phii=PhiP*(1-r)/(1-r*exp(-Pe));
    PHII(loop) = Phii;
    PC(loop) = Pc;
end

for Phii = 6:15
    Pi = 4;
    loop = Phii-5;
    S3= vpasolve(JV/A==Lp*((Pc-Pi)-r*(PhiP-Phii)),JV);
    JvOld(loop) = S3;
    S4= vpasolve(JV/A==Lp*((PcBig-Pi)-r*(PhiP-Phii)),JV);
    JvOldBig(loop) = S4;
end
hold on
plot(Jv,PHII,'b','LineWidth',2)
plot(Jv,ones(1,length(PHII))*PhiP,'m','LineWidth',2)
%    plot(Jv,'o')
plot(JvOld,6:15,'k','LineWidth',2)
plot(JvOldBig,6:15,'r','LineWidth',2)
legend('Phii','PhiP','Starling Law','Starling Law Big Pc')
xlabel('Jv')
ylabel('Phii')
```

Box 7.23.4: code for Figure 7.166

```
function AQP4_Paper
syms JV
close all
warning('off','all')

syms JvSym
Area = 1; Pd = 2e-1; D = 1; r = 0.85; Lp = 1; L = 1; c2 = 13; %c right
c1Location = [1 12 24 59]; P1 = 40; P2 = 12; c1Hold = 2:80; %used for a range
%% set up A matrix for 1D with given grid size
JvHold = zeros(1,length(c1Hold));
for Loop1 = 1:length(c1Hold)
    c1 = c1Hold(Loop1);
    %% calculate revised ci
    Pe = JvSym*(1-r)/(Pd*Area);
    ci=c1*(1-r)/(1-r*exp(-Pe));
    S= solve((P1-P2)==JvSym/Area/Lp+r*(c1-ci),JvSym);
    JvHold(Loop1) = S;
    u = S*(1-r)/Area;
    Js(Loop1) = u*c1-u*(c1-c2)/(1-exp(u/Pd*L));
    Pe = S*(1-r)/(Pd*Area);
    ciHold(Loop1) = c1*(1-r)/(1-r*exp(-Pe));

    S4= vpasolve(JvSym/Area==Lp*((P1-P2)-r*(c1-c2)),JvSym);
    JvOld(Loop1) = double(S4);
    JsOld(Loop1) = D*(c1-c2);
end
save('AQP4_Fig_1_data','c1Hold','JvHold','JvOld','c1Location')
figure
plottingFunc
figure
floatingBars
end

function plottingFunc
load AQP4_Fig_1_data
c2 = 13;
%% plotting
clf
plot(c1Hold-c2,JvHold,'k','LineWidth',3)
hold on
plot(c1Hold-c2,JvOld,'g','LineWidth',3) %needs evaluation

for iState = 1:4
    state = [JvHold(c1Location(iState)), JvOld(c1Location(iState))];
    loc = [c1Hold(c1Location(iState))-c2, c1Hold(c1Location(iState))-c2];
    plot(loc,state,'.','MarkerSize',25)
end
xlabel('Concentration Difference')
ylabel('Water Flux (Jv)')
legend('Revised Starling Law','Classic Starling Law','Scenario 2','Scenario 3','Scenario
4','Scenario 5')
legend boxoff
plot(min(c1Hold-c2):max(c1Hold-c2),zeros(1,length(c1Hold+1)), 'k') %plot x axis
plot(zeros(1,length(-30:40)), -30:50, 'k') %plot y axis
ylim([-30,50])
xlim([min(c1Hold-c2),max(c1Hold-c2)])
end

function floatingBars
P2 = [12;12;12;12;12];
P1 = [40;40;40;40;40];
c1Potential = [0;1.60000000000000;10.4000000000000;20;48];
c2Potential1 = [0;0.320000000000000;2.08000000000000;4;2.08000000000000];
c2Potential = [0;10.4000000000000;10.4000000000000;10.4000000000000;10.4000000000000];
c1 = [0;2;13;25;60];
for iScenario = 1:length(c1)
```

```

figure(2)
hold on
plotRectangle(P1(iScenario),P2(iScenario),c1Potential(iScenario),...
    c2Potential(iScenario),c2Potential1(iScenario),iScenario)

end
plot(0:20,zeros(1,21),'k')
% calculate where the xlabel go
xTickLocation = [1.4:3.4,5.4:7.4,9.4:11.4,13.4:15.4,17.4:20.4];
set(gca,'xTick',xTickLocation,'xTickLabel','') %load Labels from data
set(gca,'XAxisLocation','top','TickDir','both')
legend('Osmotic' Potential,'Pressure' Potential,'Total
Potential','Location','southwest','Orientation','horizontal')
legend boxoff
end

function plotRectangle(P1,P2,c1,c2Classic,c2Revised,iScenario)
drawArrow = @(x,y,varargin) quiver( x(1),y(1),x(2)-x(1),y(2)-y(1),0, varargin{:} );
%% define colors
blue = [102,255,255]/255; %Pressure potential
purple = [51,255,51]/255; %Total potential
orange = [255,127,80]/255; %concentration potential
grey = [0.5 0.5 0.5]; %arrows
plot(nan,nan,'color',orange,'LineWidth',10)
plot(nan,nan,'color',blue,'LineWidth',10)
plot(nan,nan,'color',purple,'LineWidth',10)
%% calculate differences at states
totalPotential1= (P1-c1);
totalPotential2 = P2-c2Classic;
totalPotential2Revised = P2-c2Revised;
%% calculate where to plot
plot1 = (iScenario-1)*4+1; %for node 1
plot2 = (iScenario-1)*4+2; %node 2 classic
plot3 = (iScenario-1)*4+3; %node 2 revised
%% Node 1
rectangle('Position',[plot1,-c1,0.4,c1],'FaceColor',orange); %c1
rectangle('Position',[plot1,0,0.4,P1],'FaceColor',blue); %P1
if(totalPotential1) > 0
    rectangle('Position',[plot1+0.4,0,0.4,totalPotential1],'FaceColor',purple)
%deltaPotential1
else
    rectangle('Position',[plot1+0.4,totalPotential1,0.4,-
totalPotential1],'FaceColor',purple) %deltaPotential1
end
drawArrow([plot1+0.6,plot1+0.6],[P1,P1-c1],'color',grey,'LineWidth',2,'MaxHeadSize',0.05);
%% node 2 Classic
rectangle('Position',[plot2,-c2Classic,0.4,c2Classic],'FaceColor',orange) %c2
rectangle('Position',[plot2,0,0.4,P2],'FaceColor',blue) %P2
if(totalPotential2) > 0
    rectangle('Position',[plot2+0.4,0,0.4,totalPotential2],'FaceColor',purple)
%deltaPotential2
else
    rectangle('Position',[plot2+0.4,totalPotential2,0.4,-
totalPotential2],'FaceColor',purple) %deltaPotential1
end
drawArrow([plot2+0.6,plot2+0.6],[P2,P2-
c2Classic],'color',grey,'LineWidth',2,'MaxHeadSize',0.05);
%%node 2 Revised
rectangle('Position',[plot3,-c2Revised,0.4,c2Revised],'FaceColor',orange) %c1
rectangle('Position',[plot3,0,0.4,P2],'FaceColor',blue) %P1
if(totalPotential2Revised) > 0
    rectangle('Position',[plot3+0.4,0,0.4,totalPotential2Revised],'FaceColor',purple)
%deltaPotential1
else
    rectangle('Position',[plot3+0.4,totalPotential2Revised,0.4,-
totalPotential2Revised],'FaceColor',purple) %deltaPotential1
end
drawArrow([plot3+0.6,plot3+0.6],[P2,P2-
c2Revised],'color',grey,'LineWidth',2,'MaxHeadSize',0.05);
end

```

Box 1.4: Code to solve revised SL with Newton method

```
% % GH 9/18/2018 -- this is a procedure that estimates the nonlinear
% revised starling law and solves using Newton
function SLrevised
% close all
tolerance = 1e-6; rememberHistory=true;
jv = [-60:0.01:-1 1:0.01:60];
plotOverJv(jv)

% % solving 1 equation system
jv = -100:0.01:-1;
plotOverJv(jv)
x0 = -5;
[xSoln,fSoln,xHist] = newton(@myFunc,[],x0,tolerance,rememberHistory);
figure(1), scatter(xSoln,fSoln)
plotHistory(xHist,@myProblem2)

jv = 30:0.01:32;
plotOverJv(jv)
x0 = 31;
[xSoln,fSoln,xHist] = newton(@myFunc,[],x0,tolerance,rememberHistory);
figure(1), scatter(xSoln,fSoln)
legend('residual(Jv)','r=0','solution 1','solution 2')
% % plotHistory(xHist,@myProblem2)

% solving 2 equation system
figure
jv = -100:0.1:100; pe = -100:1:100;
x0 = [1;5];
[xSoln,fSoln,xHist] = newton(@myFunc2,[],x0,tolerance,rememberHistory);
plotOverPeJv(pe,jv)
hold on, scatter3(xSoln(1),xSoln(2),fSoln'*fSoln)
plotHistory(xHist,@myFunc2,[-100,100],[-100,100])

% figure(2), scatter3(xSoln(2),fSoln'*fSoln,fSoln'*fSoln,'d')
legend('residual(Jv)','r=0','solution 1D','solution 2D')
end

function plotOverJv(Jv)
figure
residual = myFunc(Jv).*myFunc(Jv);
plot(Jv,residual); xlabel('Jv'); ylabel('residual error');
hold on, plot(Jv,zeros(1,length(Jv)),'k')
legend('revised Starling law','y=0')
end

function plotOverPeJv(Pe,Jv)
for i = 1:length(Pe)
    for j = 1:length(Jv)
        residual(i,j) = myFunc2([Pe(i);Jv(j)])'*myFunc2([Pe(i);Jv(j)]);
    end
end
surf(Pe,Jv,residual','edgecolor','none');
xlabel('Pe'); ylabel('Jv'); zlabel('residual error');
% hold on, plot(Jv,zeros(1,length(Jv)),'k')
% legend('revised Starling law','y=0')
end

% a function to solve Jv in terms of Jv
function residual = myFunc(jv)
pArt = 60; pISF = 15; piArt = 28; piISF = 3; Lp = 1.3; sigma=0.75; Pd = 0.2;
A=0.5;
deltaP = pArt-pISF; deltaPI = piArt-piISF;
```

```

pe = sqrt((jv.*(1-sigma)/(Pd*A)).^2);
% pe = (jv.*(1-sigma)/(Pd*A));
residual = jv - (Lp*(deltaP-sigma*piArt*(1-exp(-pe)./(1-sigma*exp(-pe)))));
end

% this procedure uses 2 equations to solve for Jv and peclet number
function residual = myFunc2(PeJv)
pArt = 60; pISF = 15; piArt = 28; piISF = 3;
Lp = 1.3; sigma=0.75; Pd = 0.2; A=0.5;
deltaP = pArt-pISF; deltaPI = piArt-piISF;
residual = [0;0]; %set dimensions
% residual(1) = PeJv(1)-PeJv(2).*(1-sigma)/(Pd*A);
residual(1) = PeJv(1)-sqrt((PeJv(2).*(1-sigma)/(Pd*A))^2);
residual(2) = PeJv(2) - (Lp*(deltaP-sigma*piArt*(1-exp(-PeJv(1))./(1-sigma*exp(-
PeJv(1))))));
end
% % add and validate 2 versions of this with overload

% fun is a pointer to the function and dfun is a pointer to the jacobian
% % next version -- write iterates and error for each step, possibly with switch, xHistory
and then plot when done
% function [F,x] = newton(fun,dfun,x0,tol)
function [xSolution,FresidualAtSolution,xHistory] = newton(fun,dfun,x0,tol,rememberHistory)
%rememberHistory is boolean
x = x0; err = fun(x)*fun(x); maxIter = 100; iter=1; xHistory(:,iter)=x0;
if isempty(dfun), useNumericalDiff = true;
else useNumericalDiff = false; end
while err > tol
F = fun(x);
if useNumericalDiff, J = getNumericalDerivative(fun,x);
else J = dfun(x); end
if abs(det(J)) < 1e-15, disp('singular Jacobian'); break; end %should
take a gradient step here
delX = - J\F;
% alfa = getStepsizeForDecreasingResiduals(delX,x,fun); %stepsize control
alfa = getStepsizeArmijo(delX,x,fun,dfun);
xNew = x + alfa*delX;
if rememberHistory, xHistory(:,iter) = x; end % remember history %
err = sqrt(fun(xNew)*fun(xNew)); iter = iter+1;
if iter > maxIter,disp('max iteration exceeded'); break; end
if norm(x-xNew) < 1e-6, disp(['no more update, saddle point found at '
num2str(xNew')]); break; end
x = xNew;
end
if rememberHistory, xHistory(:,iter) = x; end % remember history %
xSolution = x; FresidualAtSolution = fun(x);
end

function alfa = getStepsizeArmijo(delX,xOld,fun,dfun)
% % armijo linesearch using quadratic function fitting
alfa = 1; updating = true; delta = 0.1;
phiOld = fun(xOld)*fun(xOld);
phiNew = fun(xOld + alfa*delX)*fun(xOld + alfa*delX);
while updating
alfaNew = phiOld*alfa^2/((2*alfa - 1)*phiOld + phiNew);
phiNew = fun(xOld + alfa*delX)*fun(xOld + alfa*delX);
if alfaNew > 1, alfa = 1; break; %breaking criteria when it wants to go larger
elseif phiNew-phiOld <= -2*delta*alfaNew*phiOld, break;
else alfa = alfaNew; end
end
end

function alfa = getStepsizeForDecreasingResiduals(delX,xOld,fun)
% % primitive linesearch by halving the stepsize, may fail because no
% iteration stop criterion in the while loop
alfa = 1; errOld = fun(xOld)*fun(xOld);
errNew = fun(xOld+delX)*fun(xOld+delX);
while errNew > errOld
alfa = 0.5* alfa;

```



```

        xNew = xOld + alfa*delX;
        errNew = fun(xNew)*fun(xNew);
    end
    if errNew > errOld, display('solution diverged'); alfa = -1; end;
end

% make n-dimensional
function J = getNumericalDerivative(fun,x)
N = length(x); J = zeros(N,N);
dX = 1e-6*ones(N,1);
for i = 1:N
    xNew = x;    xNew(i) = xNew(i) + dX(i);
    dF = fun(xNew)-fun(x);
    J(:,i) = dF./dX;
end
end

function plotHistory(xHist,fun,xRange,yRange)
figure, plotNonLinearSurface(fun,xRange,yRange)
for i = 1:size(xHist,2)-1
    plot([xHist(1,i) xHist(1,i+1)], [xHist(2,i) xHist(2,i+1)], 'k', 'linewidth', 2)
    error(i) = fun(xHist(:,i))*fun(xHist(:,i));
end
error(i+1) = fun(xHist(:,i+1))*fun(xHist(:,i+1));    figure,
plot(error), xlabel('iteration number'); ylabel('residual error')

end

function plotNonLinearSurface(fun,xRange,yRange)
xPlot = xRange(1):0.1:xRange(2);
x2Plot = yRange(1):0.1:yRange(2);
for i = 1:length(xPlot)
    for j = 1:length(x2Plot)
        err = fun([xPlot(i);x2Plot(j)]);
        errorSurface(i,j) = err'*err;
    end
end
contour(xPlot,x2Plot,sqrt(errorSurface'),100,'b'), hold on
end

```

7.23.7 Derivation of the Patlack Equation from conservation

In this derivation, we show how to obtain a steady-state concentration profile for the convection-diffusion problem in Equation (7.389). Computing the concentration profile, $c(x)$, allows determining the transmembrane water and solute flux (j_w and j_s).

Table 7.47. This section details a derivation of the revised Starling Law.

Table 7.48: Variables used for Section 7.23.3

Variable name	Physical Meaning	Units
J_v	Volumetric flux of water	ml/min
J_s	Molar flux of solute	mol/s
L_P	filtration coefficient	mm/s/mmHg
A	Area	mm ²
Π	Osmotic pressure (in Plasma and ISF)	mmHg
P	Hydrostatic pressure (in Capillary and ISF)	mmHg
σ	reflection coefficient	--
D	diffusion coefficient	mm ² /s
P_d	diffusional permeability	mm ² /s
R	Ideal gas constant	J/mol/K
T	absolute temperature	K
u	flow velocity	mm/s
$L(\text{or } x)$	width of membrane	mm
Pe	Peclet number	--

The species conservation is given in equation (7.389) with convective (q_c) and diffusive (q_d) flux:

$$V \frac{dc}{dt} - \vec{\nabla} \cdot q_c = \vec{\nabla} \cdot q_d \quad (7.389)$$

$$V \frac{dc}{dt} + Vuc' = VDc''$$

Which is at steady-state simplifies to:

$$uc' = Dc'' \quad (7.390)$$

It is customary to account for the deflection of the molecules in a semipermeable membrane accounting for the reduction in convection flux because some of the solute cannot pass through the membrane (reduced convection). This reduction is modelled according to the deflection coefficient, σ , as in Equation (7.391). This velocity, u , can be seen as a reduced effective convective driving force.

$$u = \frac{J_w(1 - \sigma)}{A} \quad (7.391)$$

With deflected convection, the molecular transmembrane fluxes can be rewritten:

$$\frac{J_w(1 - \sigma)}{A} c' = Dc'' \quad (7.392)$$

Assuming the solution to this 2nd order ODE follows Equation (7.393), the boundary conditions can be used to uniquely defined the function. It can be shown that this equation is the analytic solution to a linear 2nd order ODE with one root of 0 (Section 7.23.9).

$$c = \alpha e^{\frac{J_w(1-\sigma)}{AD}x} + \beta \quad (7.393)$$

Taking the 1st and 2nd derivatives of the solution gives:

$$c' = \frac{\alpha J_w(1 - \sigma)}{AD} e^{\frac{J_w(1-\sigma)}{AD}x} \quad (7.394)$$

$$c'' = \frac{\alpha [J_w(1 - \sigma)]^2}{A^2 D^2} e^{\frac{J_w(1-\sigma)}{D}x} \quad (7.395)$$

Back substitution into the conservation equation gives:

$$0 = u \frac{\alpha J_w(1 - \sigma)}{AD} e^{\frac{J_w(1-\sigma)}{AD}x} - D \frac{\alpha J_w(1 - \sigma)^2}{A^2 D^2} e^{\frac{J_w(1-\sigma)}{AD}x} \quad (7.396)$$

This can be used to validate the solution:

$$0 = \frac{\alpha [J_w(1 - \sigma)]^2}{A^2 D} e^{\frac{J_w(1-\sigma)}{AD}x} - \frac{\alpha J_w(1 - \sigma)^2}{A^2 D^2} e^{\frac{J_w(1-\sigma)}{AD}x} \quad (7.397)$$

$$0 = \left(\frac{\alpha [J_w(1 - \sigma)]^2}{A^2 D} e^{\frac{J_w(1-\sigma)}{AD}x} \right) - \left(\frac{\alpha J_w(1 - \sigma)^2}{A^2 D} e^{\frac{J_w(1-\sigma)}{AD}x} \right) \quad (7.398)$$

$$0 = \left(\frac{\alpha [J_w(1 - \sigma)]^2}{A^2 D} e^{\frac{J_w(1-\sigma)}{AD}x} \right) (1 - 1) \quad (7.399)$$

$$0 = \left(\frac{\alpha [J_w(1 - \sigma)]^2}{A^2 D} e^{\frac{J_w(1-\sigma)}{AD}x} \right) (0) \quad (7.400)$$

Boundary conditions can be used to calculate the constant coefficients of the specific solution:

$$c(0) = c_1; \quad c(L) = c_2 \quad (7.401)$$

$$\alpha e^{\frac{J_w(1-\sigma)}{D} \cdot 0} + \beta = c_1 \quad (7.402)$$

$$\alpha e^{\frac{J_w(1-\sigma)}{D} \cdot L} + \beta = c_2 \quad (7.403)$$

$$Pe = \frac{u}{D/L} = \frac{J_w(1-\sigma)}{AD} L \quad (7.404)$$

$$\alpha(1 - e^{Pe}) = c_1 - c_2 \quad (7.405)$$

$$\alpha = \frac{c_1 - c_2}{(1 - e^{Pe})} \quad (7.406)$$

$$\beta = c_1 - \alpha \quad (7.407)$$

$$\beta = \frac{c_1(1 - e^{Pe}) - (c_1 - c_2)}{1 - e^{Pe}} \quad (7.408)$$

$$\beta = \frac{-c_1 e^{Pe} + c_2}{1 - e^{Pe}} \quad (7.409)$$

$$c(x) = \frac{c_1 - c_2}{1 - e^{Pe}} e^{\frac{J_w(1-\sigma)}{AD} x} + \frac{c_2 - c_1 e^{Pe}}{1 - e^{Pe}} \quad (7.410)$$

Substituting the boundary conditions back in verifies the solution:

$$c(0) = \frac{c_1 - c_2}{1 - e^{Pe}} + \frac{c_2 - c_1 e^{Pe}}{1 - e^{Pe}} = c_1 \quad (7.411)$$

$$c(L) = \frac{c_1 e^{Pe} - c_2 e^{Pe}}{1 - e^{Pe}} + \frac{c_2 - c_1 e^{Pe}}{1 - e^{Pe}} = c_2 \quad (7.412)$$

Solving for the solute flux, j_s arrives at the Patlack equation, (7.418):

$$\frac{J_s}{A} = \frac{J_w(1 - \sigma)c(x = 0)}{A} - Dc'(x = 0) \quad (7.413)$$

$$\frac{J_s}{A} = \frac{J_w(1 - \sigma)}{A} c_1 - \frac{J_w(1 - \sigma)}{A} \frac{c_1 - c_2}{1 - e^{Pe}} e^{\frac{J_w(1 - \sigma)}{D} x_0} \quad (7.414)$$

$$\frac{J_s}{A} = \frac{J_w(1 - \sigma)}{A} \left[c_1 - \frac{c_1 - c_2}{1 - e^{Pe}} \right] \quad (7.415)$$

$$\frac{J_s}{A} = \frac{J_w(1 - \sigma)}{A} \left[\frac{c_1 - c_1 e^{Pe}}{1 - e^{Pe}} - \frac{c_1 - c_2}{1 - e^{Pe}} \right] \quad (7.416)$$

$$J_s = J_w(1 - \sigma) \left[\frac{c_2 - c_1 e^{Pe}}{1 - e^{Pe}} \right] \quad (7.417)$$

$$J_s = J_w(1 - \sigma) \frac{-c_L + c_0 e^{Pe}}{e^{Pe} - 1} \quad (7.418)$$

7.23.8 Deriving revised Starling law from Patlack equation

This section calculates the revised Starling law from the Patlack equation. Assuming the concentration inside the pore is a ratio of solute flux to water flux gives:

$$c_L = \frac{J_s}{J_w} \quad (7.419)$$

Both sides in the Patlack equation can be divided by the water flux and gives:

$$c_L = \frac{J_s}{J_w} = (1 - \sigma) \frac{-c_L + c_0 e^{Pe}}{e^{Pe} - 1} \quad (7.420)$$

Isolating C_L :

$$c_L = -\frac{c_L(1 - \sigma)}{e^{Pe} - 1} + \frac{(1 - \sigma)c_0 e^{Pe}}{e^{Pe} - 1} \quad (7.421)$$

$$c_L(e^{Pe} - 1) + c_L(1 - \sigma) = (1 - \sigma)c_0 e^{Pe} \quad (7.422)$$

$$\frac{c_L(e^{Pe} - \sigma)}{e^{Pe}} = (1 - \sigma)c_0 \quad (7.423)$$

$$c_L \left(-\frac{(\sigma)}{e^{Pe}} + 1 \right) = (1 - \sigma)c_0 \quad (7.424)$$

$$c_L(1 - \sigma e^{-Pe}) = (1 - \sigma)c_0 \quad (7.425)$$

$$c_L = \frac{(1 - \sigma)c_0}{1 - \sigma e^{-Pe}} \quad (7.426)$$

c_i can be converted to Π_i :

$$\frac{c_L}{c_0} = \frac{\Pi_L}{\Pi_0} = \frac{(1 - \sigma)}{1 - \sigma e^{-Pe}} \quad (7.427)$$

$$\Pi_L = \frac{\Pi_0(1 - \sigma)}{1 - \sigma e^{-Pe}}$$

This pressure can be substituted into the classical Starling law:

$$\frac{J_w}{A} = L_p \left\{ \Delta P - \sigma \left[\Pi_0 - \frac{\Pi_0(1 - \sigma)}{1 - \sigma e^{-Pe}} \right] \right\} \quad (7.428)$$

Which can be simplified:

$$\frac{J_w}{A} = L_p \left\{ \Delta P - \sigma \Pi_0 \left[1 - \frac{(1 - \sigma)}{1 - \sigma e^{-Pe}} \right] \right\} \quad (7.429)$$

$$\frac{J_w}{A} = L_p \left\{ \Delta P - \sigma \Pi_0 \frac{1 - \sigma e^{-Pe} - (1 - \sigma)}{1 - \sigma e^{-Pe}} \right\} \quad (7.430)$$

$$\frac{J_w}{A} = L_p \left\{ \Delta P - \sigma \Pi_0 \frac{1 - 1 + \sigma(1 - e^{-Pe})}{1 - \sigma e^{-Pe}} \right\} \quad (7.431)$$

$$\frac{J_w}{A} = L_p \left\{ \Delta P - \sigma^2 \Pi_0 \frac{1 - e^{-Pe}}{1 - \sigma e^{-Pe}} \right\} \quad (7.432)$$

This arrives at the revised Starling law equation as seen in Equation (7.382).

7.23.9 Analytic solution to linear ODE of 2nd order

A homogenous ordinary differential equation (ODE) can be constructed with a steady-state diffusion problem or a dynamic reaction problem such as the one in Equation (7.433). The analytic solution of this differential equation is an algebraic function whose derivative is merely itself multiplied by an exponent. Only a few equations order will uphold this condition, the majority of which are an infinite series whose derivative is also an infinite series (no loss of power when taking the derivative). A common infinite power series is the exponential function (e) such as in Equation (7.434):

$$\frac{d\phi}{dt} = \alpha\phi \quad (7.433)$$

$$\phi(t) = c_0 e^{\alpha t} \quad (7.434)$$

Where

$$e^{\alpha t} = \sum_{i=0}^{\infty} \frac{(\alpha t)^i}{i!} \quad (7.435)$$

This infinite series holds the condition that the derivative is a constant (α) multiplied by the original function:

$$\frac{de^{\alpha t}}{dt} = \alpha e^{\alpha t} \quad (7.436)$$

The particular solution can be evaluated (α can be computed) using the boundary conditions. In the case of cylindrical diffusion, or diffusion with convection and reaction term, the equation becomes more complicated:

$$D \frac{d^2\phi}{dx^2} - f \frac{d\phi}{dx} - k_1 c = 0 \quad (7.437)$$

If a power series (exponential function used here for simplicity) is chosen, the evaluation can be rewritten as:

$$D \frac{d^2}{dx^2} e^{\alpha x} - f \frac{d}{dx} e^{\alpha x} - k_1 e^{\alpha x} = 0 \quad (7.438)$$

$$\begin{aligned} D\alpha^2 e^{\alpha x} - f\alpha e^{\alpha x} - k_1 e^{\alpha x} &= 0 \\ e^{\alpha x} (D\alpha^2 - f\alpha - k_1) &= 0 \end{aligned} \quad (7.439)$$

Because it is known that $e^{\alpha x}$ is a function of the independent variable x , and thus is cannot be is not a constant value of 0 (in fact, its value is never 0), it can be concluded that the coefficients must be 0. This simplifies the system which now has 2 roots:

$$D\alpha^2 - f\alpha - k_1 = 0 \quad (7.440)$$

$$r = -f \pm \frac{\sqrt{f^2 - 4D}}{2D} \quad (7.441)$$

The roots are the exponential coefficients:

$$\phi(x) = ae^{r_1} + be^{r_2} \quad (7.442)$$

Where r_1 and r_2 are the first and second roots of equation (7.441), respectively. When considering a single diffusion-convection problem as in the Revised Starling Law, the equation has one root of value 0:

$$D \frac{d^2}{dx^2} e^{\alpha x} - f \frac{d}{dx} e^{\alpha x} = 0 \quad (7.443)$$

$$\begin{aligned}
D\alpha^2 e^{\alpha x} - f\alpha e^{\alpha x} &= 0 \\
e^{\alpha x} (D\alpha^2 - f\alpha) &= 0 \\
\alpha (D\alpha - f) &= 0 \\
\alpha = 0, \quad f/D &
\end{aligned}
\tag{7.444}$$

Which renders the solution to be of the form (before plugging in the boundary conditions to find the particular solution):

$$\begin{aligned}
\phi(x) &= \alpha e^{f/Dx} - \beta e^{0x} \\
\phi(x) &= \alpha e^{f/Dx} - \beta
\end{aligned}
\tag{7.445}$$

This equation is consistent with the derivation offered in Section 7.23.7.

7.23.10 Example problem

Introduction. The brain is supplied with nutrients and cleaned of waste material continuously using the blood as the delivery/garbage trucks that carry the nutrients into the brain and carry the waste material away to the kidneys and spleen. One of the most essential nutrients the brain utilizes is water, which is metabolized by cells to produce energy and is in constant demand. The replenishing of water between the blood and the brain tissue occurs through a membrane known as the blood-brain-barrier (BBB) that is composed of a vessel membrane, pericyte end-feet, and other supporting cells. The study of how materials move across the BBB is a highly active field of research including the study of flux through the semi-permeable membrane of the BBB. Water

transport across this membrane has been modeled as a function of hydrostatic pressure gradients (Δp) and osmotic pressure gradients (Δc). Figure 7.167 expresses these pressures on a schematic vessel in the brain.

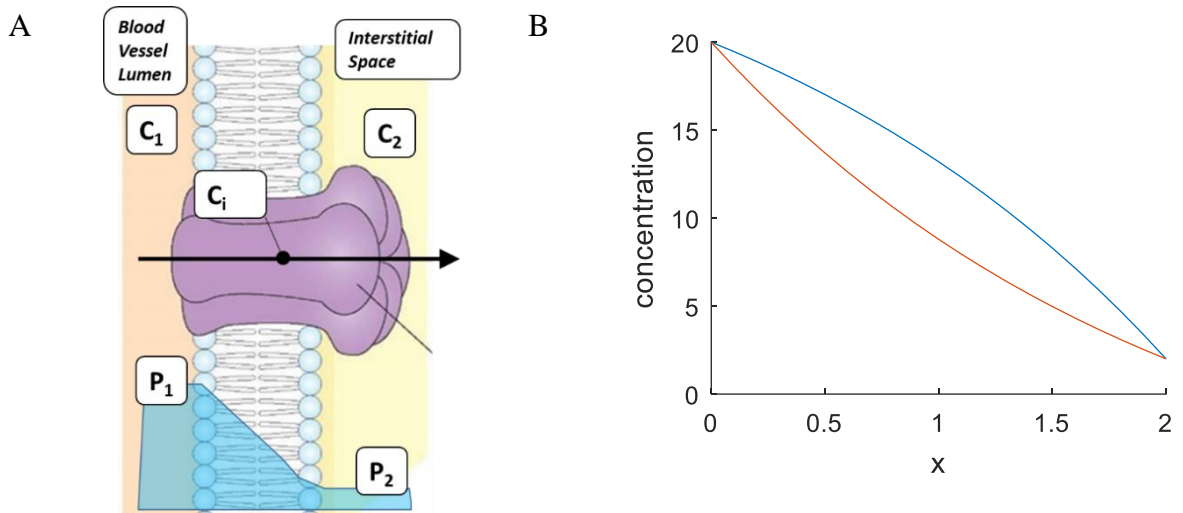


Figure 7.167. (A) A cartoon representation of the pressure gradients that occur across the BBB through a single pore.

The hydrostatic pressure (Δp) is opposed to the osmotic pressure (Δc). Here, the combination of these two pressures results in a total pressure that drives the water transport into or out of the interstitial space. (B) The concentration profile within a membrane with Dirichlet boundary conditions on each side, c_1 and c_2 , is offered for a solute transport due to diffusion and convection. The blue curve has convection from left to right, the red curve has convection from right to left. Note, pure diffusion would give a straight line (not shown).

Part 1, Solving the diffusion-convection equation for concentration. The solute can move via passive (diffusive) and active (convective) means in the brain. The equation that describes how this motion happens dynamically in one-dimension is expressed in Equation (7.446). This partial differential equation can be solved at steady-state (Section 7.23.10.1) to yield the analytic function for solute flux (Equation (7.447)-(7.448)). Here, A is the cross sectional area of the pore, L_p is the trans-membrane permeability, σ is the reflection coefficient of the membrane, R is the ideal gas

constant, v is the velocity of the water flow, A_{\perp} is the cross-sectional area of the pore, T is temperature in Kelvin, and c is the concentration of a solute. J_s and J_w are volumetric solute flux and volumetric water flux, respectively.

$$V \frac{dc}{dt} = \frac{d}{dx} J_s = \frac{d}{dx} (J_s^c + J_s^d)$$

$$V \frac{dc}{dt} = \nabla \cdot (vc - D \nabla c) V \quad (7.446)$$

$$V \frac{dc}{dt} = \frac{d}{dx} \left(vc - D \frac{dc}{dx} \right) V$$

$$c(x) = \frac{c_1 - c_2}{1 - e^{Pe}} \cdot e^{\frac{j_w(1-\sigma)}{DA_{\perp}} x} + \frac{c_2 - c_1 e^{Pe}}{1 - e^{Pe}} \quad (7.447)$$

$$Pe = \frac{uL}{A_{\perp} D} = \frac{J_w(1 - \sigma)}{A_{\perp} D} L \quad (7.448)$$

1. Show by differentiation and insertion that that Equation (7.447)-(7.448) is satisfies the steady version of Equation (7.446).
2. Plot the concentration profile, $c(x)$ in $\mu\text{mol}/\mu\text{m}^3$, as a function of length, x in μm , between $x=0$ and $x=L$. Use values listed in Table 7.48 for all other parameters. Also solve with $J_w = -0.1 \mu\text{m}^3 \cdot \text{s}^{-1}$ and plot the result.
3. Compute the solute flux as given in Equation 3 at $x=0$, $x=0.5 \cdot L$, and $x=L$ and show that the flux is constant at all locations, as expected from a steady-state solution. (Note that you will need this flux later in the Starling's Law portion of this homework).

Note that in Equation (7.446)-(7.448), vc is not the same as the full flux through the channel, $\frac{J_w c}{A_\perp}$, because the full flux is hindered by the semi-permeable membrane. To account for this hindrance of the full flux, vc is reduced by a factor of $(1 - \sigma)$ as reflected in equation (7.449) where σ is a reflection coefficient.

$$\frac{J_s(x=0)}{A_\perp} = (1 - \sigma)vc|_{x=0} - \frac{dc}{dx}\bigg|_{x=0} = \frac{J_w}{A_\perp} (1 - \sigma) \left[\frac{c_2 - c_1 e^{Pe}}{1 - e^{Pe}} \right] \quad (7.449)$$

$$P_e = \gamma J_w$$

$$\gamma = 3.33 \cdot 10^{-5} \mu\text{m} \cdot \text{s}^{-1}$$

Table 7.49: Parameters used in this case study, empirically derived

Parameter	Value	Units	Meaning
p_1	60	mmHg	Arterial pressure
p_2	4	mmHg	ISF pressure
c_1	20	$\mu\text{mol} \cdot \mu\text{m}^{-3}$	Arterial solute concentration
c_2	2	$\mu\text{mol} \cdot \mu\text{m}^{-3}$	ISF solute concentration
D	$1.8\text{e-}2$	$\mu\text{m}^2 \cdot \text{s}^{-1}$	Solute diffusivity*
σ	0.5	--	Membrane reflection coefficient
L	298	K	Temperature
L_P	$1.67 \cdot 10^{-3}$	$\mu\text{m} \cdot \text{mmHg}^{-1} \cdot \text{s}^{-1}$	Membrane permeability
T	3	μm	Endothelial layer thickness
A_\perp	5	μm^2	Endothelial layer surface area
J_w	0.1	$\mu\text{m}^3 \cdot \text{s}^{-1}$	Guess for water flux

*note the diffusivity and mass transfer coefficient in brain tissue of solvents (like oxygen) is on the order of $1\text{e}3$

Part 2. Starling's law. Starling's law concerns the transport of water across a membrane due to the hydrostatic and osmotic pressure difference. In effect, water transmembrane flux and the solute transmembrane flux are coupled. When the water flux carries with it solute across the membrane, the problem includes diffusion with convection. Through substitution, the solute flux can be eliminated, as reflected in the attached derivation, and the water flux can be written as a function

of water flux, i.e. $j_w(j_w)$. Accordingly, this produces two coupled nonlinear equations that determine the net water flux across the membrane (Equation (7.450)-(7.451)).

$$\frac{J_w}{A} = L_p \left\{ \Delta P - \sigma^2 \Pi_0 \frac{1 - e^{-P_e}}{1 - \sigma e^{-P_e}} \right\} \quad (7.450)$$

$$P_e = \frac{J_w(1 - \sigma)}{AD} L \quad (7.451)$$

Plotting the residual error surface by exhaustive enumeration. The water flux through the BBB constitutes two nonlinear equations (g_1 and g_2) for two unknowns (j_w and P_e) as seen in Equation (7.452)-(7.453). Equation (7.452)-(7.453) is the same as Equation (7.450)-(7.451) written in residual form where g_1 represents Equation (7.450) written in residual form and g_2 represents Equation (7.451) written in residual form. Use p_1 , p_2 , c_1 , c_2 and physical properties from Table 7.48.

$$g_1(j_w, P_e) = 0 \quad (7.452)$$

$$g_2(j_w, P_e) = 0 \quad (7.453)$$

4. Substitute Equation (7.453) into Equation (7.452) to create a single nonlinear equation.

Identify the solution for j_w by exhaustive enumeration of the residual line for j_w ranging from -100 to 100 ml/min. Find the solution using the provided code for the Newton method in Matlab (see Section 7.23.10.3 for the method and code). Prove that the Newton solution is

the solution to the original equation. Plot the residual line ($g_1'(j_w)$) and the solution to identify if your solution is the only solution in this space.

5. Find the solution for P_e and j_w by exhaustively enumerating the residual in the space of the unknown P_e and j_w and plotting the residual error contour and residual error surface. Use the values listed in Table 7.48. Evaluate P_e over the range -100 to 100.
6. Solve equation (7.452)-(7.453) simultaneously with Matlab's fsolve function and verify that you get the same solution found by exhaustive enumeration. Solve the two equations using the supplied Newton code and validate the solution compared to Matlab's fsolve procedure and with plotting. Verify that your solution from equation (7.452) and (7.453) matches the solution from Part 2. Mark this solution on your plot. Make sure to supply fsolve with suitable initial guesses. Observe and report what happens when unreasonable initial guesses are used.
7. Also solve equation (7.452)-(7.453) for different values of the concentration at the abluminal side of the membrane as follows. $C_2 = 10$ to $45 \mu\text{mol}\cdot\mu\text{m}^{-3}$ with increments of $1 \mu\text{mol}\cdot\mu\text{m}^{-3}$.

7.23.10.1 Solution of diffusion/convection through a membrane

The species conservation is given in equation (7.389) with convective (q_c) and diffusive (q_d) flux:

$$V \frac{dc}{dt} - \vec{\nabla} \cdot q_c = \vec{\nabla} \cdot q_d$$

$$V \frac{dc}{dt} + Vuc' = VDc'' \quad (7.454)$$

Which is at steady-state simplifies to:

$$uc' = Dc'' \quad (7.455)$$

It is customary to account for the deflection of the molecules in a semipermeable membrane (the hindered diffusion due to the extra resistance from the membrane). This additional resistance hinders both convection and diffusion following a term known as the “deflection coefficient”, σ , as reflected by to equation (7.391). This velocity, u , can be seen as a reduced effective convective driving force.

$$u = \frac{J_w(1 - \sigma)}{A} \quad (7.456)$$

With deflected convection, the transmembrane fluxes can be modeled by:

$$\frac{J_w(1 - \sigma)}{A} c' = Dc'' \quad (7.457)$$

Assuming the solution to this 2nd order ODE is a linear combination of 2 exponential functions and solving for the powers of the exponential function gives one root of 0 (see Section 7.23.9), the solution takes the form of:

$$c = \alpha e^{\frac{J_w(1-\sigma)}{AD}x} + \beta \quad (7.458)$$

Taking the 1st and 2nd derivatives of the solution gives:

$$c' = \frac{\alpha J_w(1-\sigma)}{AD} e^{\frac{J_w(1-\sigma)}{AD}x} \quad (7.459)$$

$$c'' = \frac{\alpha [J_w(1-\sigma)]^2}{A^2 D^2} e^{\frac{J_w(1-\sigma)}{D}x} \quad (7.460)$$

Insertion into the conservation equation gives:

$$0 = u \frac{\alpha J_w(1-\sigma)}{AD} e^{\frac{J_w(1-\sigma)}{AD}x} - D \frac{\alpha J_w(1-\sigma)^2}{A^2 D^2} e^{\frac{J_w(1-\sigma)}{AD}x} \quad (7.461)$$

This can be used to validate the solution:

$$0 = u \frac{\alpha [J_w(1-\sigma)]^2}{A^2 D} e^{\frac{J_w(1-\sigma)}{AD}x} - D \frac{\alpha J_w(1-\sigma)^2}{A^2 D^2} e^{\frac{J_w(1-\sigma)}{AD}x} \quad (7.462)$$

$$0 = \frac{\alpha [J_w(1-\sigma)]^2}{A^2 D} e^{\frac{J_w(1-\sigma)}{AD}x} - \frac{\alpha [J_w(1-\sigma)]^2}{A^2 D} e^{\frac{J_w(1-\sigma)}{AD}x} \quad (7.463)$$

Using boundary conditions to calculate the constants:

$$c(0) = c_1; \quad c(L) = c_2 \quad (7.464)$$

$$\alpha e^{\frac{J_w(1-\sigma)}{D} \cdot 0} + \beta = c_1 \quad (7.465)$$

$$\alpha e^{\frac{J_w(1-\sigma)}{D} \cdot L} + \beta = c_2 \quad (7.466)$$

$$Pe = \frac{u}{D/L} = \frac{J_w(1-\sigma)}{AD} L \quad (7.467)$$

$$\alpha(1 - e^{Pe}) = c_1 - c_2 \quad (7.468)$$

$$\alpha = \frac{c_1 - c_2}{(1 - e^{Pe})} \quad (7.469)$$

$$\beta = c_1 - \alpha \quad (7.470)$$

$$\beta = \frac{c_1(1 - e^{Pe}) - (c_1 - c_2)}{1 - e^{Pe}} \quad (7.471)$$

$$\beta = \frac{-c_1 e^{Pe} + c_2}{1 - e^{Pe}} \quad (7.472)$$

$$c(x) = \frac{c_1 - c_2}{1 - e^{Pe}} e^{\frac{J_w(1-\sigma)}{AD} x} + \frac{c_2 - c_1 e^{Pe}}{1 - e^{Pe}} \quad (7.473)$$

Validating of the boundary conditions gives the correct values:

$$c(0) = \frac{c_1 - c_2}{1 - e^{Pe}} + \frac{c_2 - c_1 e^{Pe}}{1 - e^{Pe}} = c_1 \quad (7.474)$$

$$c(L) = \frac{c_1 e^{Pe} - c_2 e^{Pe}}{1 - e^{Pe}} + \frac{c_2 - c_1 e^{Pe}}{1 - e^{Pe}} = c_2 \quad (7.475)$$

Solving for the solute flux, j_s , arrives at the Patlack equation, (7.418):

$$\frac{J_s}{A} = \frac{J_w(1 - \sigma)c(x = 0)}{A} - Dc'(x = 0) \quad (7.476)$$

$$\frac{J_s}{A} = \frac{J_w(1 - \sigma)}{A} c_1 - \frac{J_w(1 - \sigma)}{A} \frac{c_1 - c_2}{1 - e^{Pe}} e^{\frac{J_w(1 - \sigma)}{D} x_0} \quad (7.477)$$

$$\frac{J_s}{A} = \frac{J_w(1 - \sigma)}{A} \left[c_1 - \frac{c_1 - c_2}{1 - e^{Pe}} \right] \quad (7.478)$$

$$\frac{J_s}{A} = \frac{J_w(1 - \sigma)}{A} \left[\frac{c_1 - c_1 e^{Pe}}{1 - e^{Pe}} - \frac{c_1 - c_2}{1 - e^{Pe}} \right] \quad (7.479)$$

$$J_s = J_w(1 - \sigma) \left[\frac{c_2 - c_1 e^{Pe}}{1 - e^{Pe}} \right] \quad (7.480)$$

$$J_s = J_w(1 - \sigma) \frac{-c_L + c_0 e^{Pe}}{1 - e^{Pe}} \quad (7.481)$$

A. Solving the diffusion-convection equation for concentration

Equation (7.482) describes how solute moves dynamically via diffusive (active) and convective (passive) means. Equation (7.484) shows the analytic function for the solute flux, c . See Section 7.23.10.1 to see that equation (7.484) satisfies equation (7.482).

$$V \frac{dc}{dt} = \frac{d J_s}{dx A} = \frac{d}{dx} \left(\frac{J_s^c + J_s^d}{A} \right) \quad (7.482)$$

$$V \frac{dc}{dt} + Vuc' = VDc'' \quad (7.483)$$

$$c(x) = \frac{c_1 - c_2}{1 - e^{Pe}} e^{\frac{J_w(1-\sigma)}{AD}x} + \frac{c_2 - c_1 e^{Pe}}{1 - e^{Pe}} \quad (7.484)$$

$$Pe = \frac{u}{D/L} = \frac{J_w(1 - \sigma)}{AD} L \quad (7.485)$$

The full flux, $jw^c/A\perp$, is hindered by the semi-permeable membrane. To account for this hindrance of the full flux, vc is reduced by a factor of $(1-\sigma)$ as reflected in equation (7.486) where σ is a reflection coefficient. The hand derivation shows that concentration profile satisfies the steady diffusion equation.

(7.486)

$$\begin{aligned}
 \frac{dc}{dt} &= \frac{d}{dx} \left(VC - D \frac{dc}{dx} \right) \\
 c(x) &= \frac{C_1 - C_2}{1 - e^{Pe}} \cdot e^{\frac{Vx}{DAL}} + \frac{C_2 - C_1 e^{Pe}}{1 - e^{Pe}} \\
 Pe &= \frac{VL}{D} = \frac{JWL}{DAL} \\
 \frac{dc}{dt} &= \frac{d}{dx} \left(V \cdot \left(\frac{C_1 - C_2}{1 - e^{Pe}} \cdot e^{\frac{Vx}{DAL}} + \frac{C_2 - C_1 e^{Pe}}{1 - e^{Pe}} \right) - D \cdot \frac{d}{dx} \left(\frac{C_1 - C_2}{1 - e^{Pe}} \cdot e^{\frac{Vx}{DAL}} + \frac{C_2 - C_1 e^{Pe}}{1 - e^{Pe}} \right) \right) \\
 \frac{dc}{dx} &= \frac{C_1 - C_2}{1 - e^{Pe}} \cdot \frac{V}{DAL} \cdot e^{\frac{Vx}{DAL}} \\
 \frac{dc}{dt} &= V \cdot \left(\frac{C_1 - C_2}{1 - e^{Pe}} \cdot \frac{V}{DAL} \cdot e^{\frac{Vx}{DAL}} \right) - \frac{d}{dx} \left(D \left(\frac{C_1 - C_2}{1 - e^{Pe}} \cdot \frac{V}{DAL} \cdot e^{\frac{Vx}{DAL}} \right) \right) \\
 &= V \cdot \left(\frac{C_1 - C_2}{1 - e^{Pe}} \cdot \frac{V}{DAL} \cdot e^{\frac{Vx}{DAL}} \right) - D \left(\frac{C_1 - C_2}{1 - e^{Pe}} \cdot \frac{V}{DAL} \right)^2 \cdot e^{\frac{Vx}{DAL}} \\
 &= \frac{V^2}{DAL} \left(\frac{C_1 - C_2}{1 - e^{Pe}} \cdot e^{\frac{Vx}{DAL}} \right) - \frac{V^2}{DAL} \left(\frac{C_1 - C_2}{1 - e^{Pe}} \cdot e^{\frac{Vx}{DAL}} \right) = 0 \quad \text{True} \\
 &\quad \text{eq2 satisfies eqn}
 \end{aligned}$$

B. Solving the diffusion-convection equation for concentration

Figure 7.168 shows the concentration profile, $c(x)$ in $\mu\text{mol}/\mu\text{m}^3$, as a function of length, x in μm .

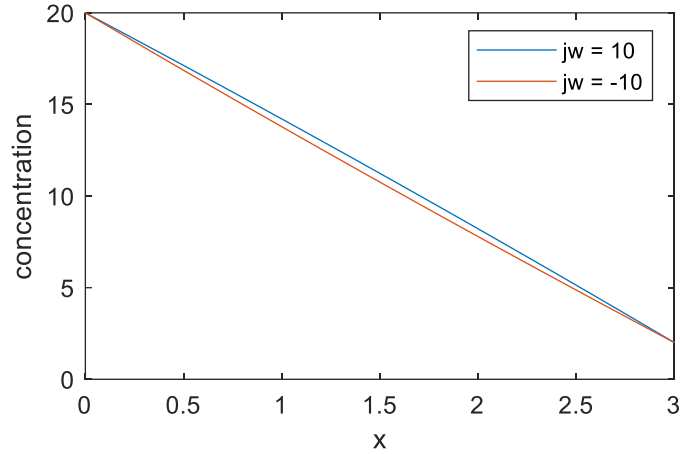


Figure 7.168. Concentration profile, $c(x)$ in $\mu\text{mol}/\mu\text{m}^3$, as a function of length, x in μm , between $x=0$ and $x=L$.

C. Starling's Law

Starling's Law, equation (6), describes the transport of water across a membrane due to the hydrostatic and osmotic pressure difference. Water transmembrane flux and the solute transmembrane flux are coupled. Therefore, the two coupled nonlinear equations determine the net water and solute flux across the membrane.

$$J_w = L_P \left\{ \Delta P - \sigma^2 \Pi_0 \frac{1 - e^{-P_e}}{1 - \sigma e^{-P_e}} \right\} \quad (7.487)$$

$$P_e = \frac{J_w(1 - \sigma)}{D} L \quad (7.488)$$

Figure 7.169 shows the solution for P_e and j_w obtained by exhaustive enumeration. Table 7.50 lists the solution obtained from MATLAB Newton implementation.

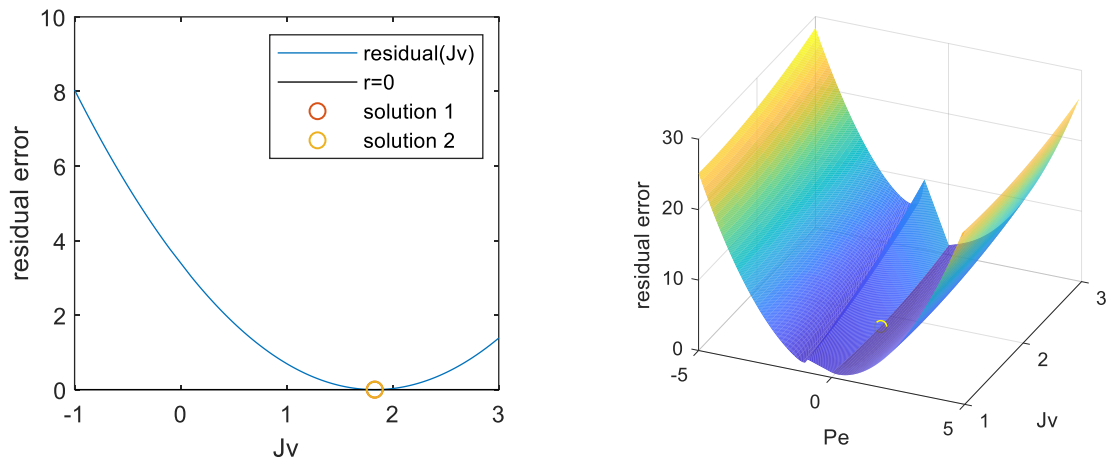


Figure 7.169. Residual error line plot (1D) and surface (2D) computed with exhaustive enumeration.

The solution with Newton has been plotted on both figures.

Table 7.50. Solution for j_w and P_e using Newton method

Unknowns	Solution	Residual
P_e	0.0194	0
j_w	1.8305	1.4e-14

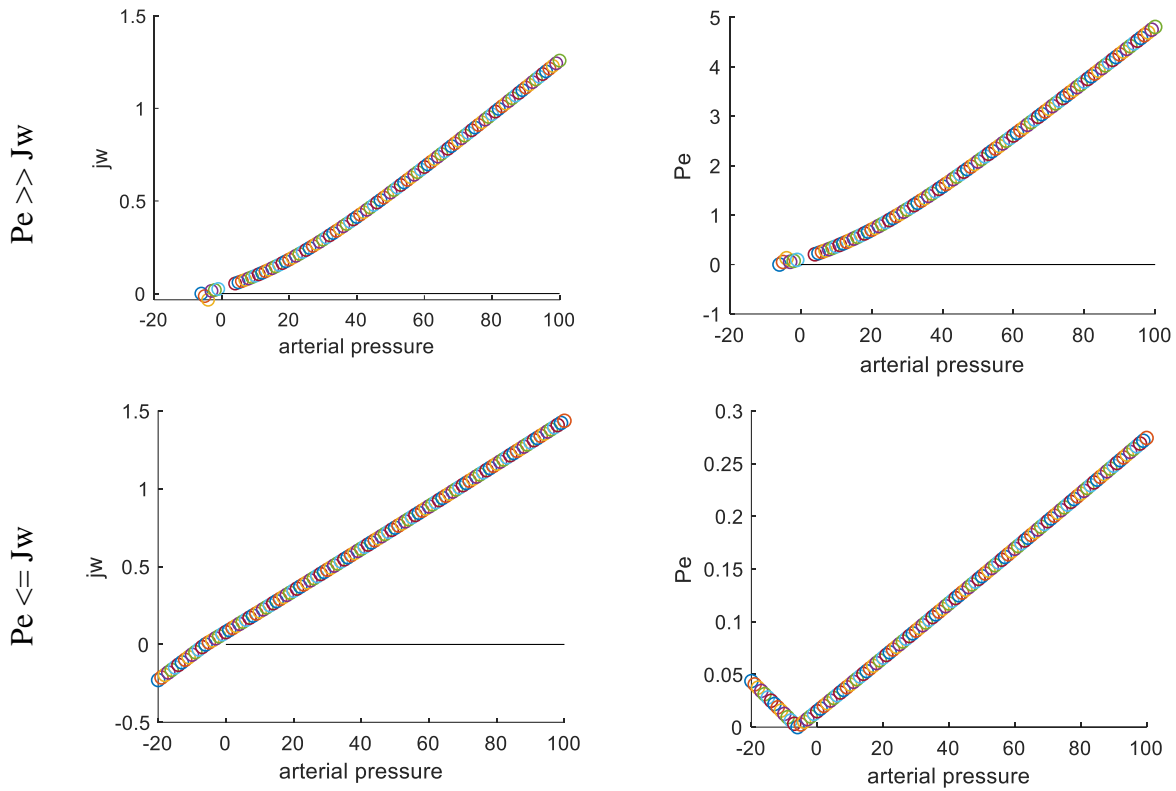


Figure 7.170. Plot of resulting flow when changing parametrically varying the arterial pressure (from $P_c=1-100$ mmHg).

Note, the water flux, J_w , always remains positive never reversing direction.

7.23.10.3 Code Listing

```
% % GH 9/18/2018 -- this is a procedure that estimates the nonlinear
% revised starling law and solves using Newton
function SLrevised
close all
% % area is based on length = 10 um, diameter is 10 um
global pArt pISF piArt piISF Lp sigma Pd A thickness D
dia = 10; LArt = 100; thickness = 3; % microns
D = 1e-3; %D for oxygen is 1.8e3 um^2/s
Lp = 0.19/13; % Lp = 1.67e-3;
% Lp = 1.67e-3; % Lp = 1.67e-3;
A=pi*(dia/2)^2*LArt;
pArt = 120; pISF = 4; piArt = 20; piISF = 2; sigma=0.5; Pd = D/thickness;
tolerance = 1e-6; rememberHistory=true;
plotConcentrationProfile; xlabel('x'); ylabel('concentration'), legend('jw = 10','jw = -
10')
jv = [1:0.01:3];
plotOverJv(jv)

% % solving 1 equation system
x0 = -5;
[xSoln,fSoln,xHist] = newton(@myFunc,[],x0,tolerance,rememberHistory);
xSoln
figure(2), scatter(xSoln,fSoln)

x0 = 31;
[xSoln,fSoln,xHist] = newton(@myFunc,[],x0,tolerance,rememberHistory);
xSoln
figure(2), scatter(xSoln,fSoln)
legend('residual(Jv)','r=0','solution 1','solution 2')

% solving 2 equation system
figure
pe = [-5:0.1:-0.9 0.01:0.1:5];
x0 = [1;1];
[xSoln,fSoln,xHist] = newton(@myFunc2,[],x0,tolerance,rememberHistory);
xSoln, fSoln
plotOverPeJv(pe,jv)
hold on, scatter3(xSoln(1),xSoln(2),fSoln*fSoln,'y')

legend('residual(Jv)','solution','solution 1D','solution 2D')

figure, hold on, figure, hold on, x0 = [1;1];
for pArt = -20:100
[xSoln,fSoln,xHist] = newton(@myFunc2,[],x0,tolerance,rememberHistory);
if fSoln*fSoln < 1e-6
figure(4), scatter(pArt,xSoln(2));
figure(5), scatter(pArt,xSoln(1));
end
end
figure(4), plot(0:100,zeros(101,1),'k'); xlabel('arterial pressure'); ylabel('jw')
figure(5), plot(0:100,zeros(101,1),'k'); xlabel('arterial pressure'); ylabel('Pe')
end

function plotOverJv(Jv)
figure
residual = myFunc(Jv).*myFunc(Jv);
plot(Jv,residual); xlabel('Jv'); ylabel('residual error');
hold on, plot(Jv,zeros(1,length(Jv)),'k')
legend('revised Starling law','y=0')
end

function plotOverPeJv(Pe,Jv)
for i = 1:length(Pe)
for j = 1:length(Jv)
residual(i,j) = myFunc2([Pe(i);Jv(j)])*myFunc2([Pe(i);Jv(j)]);
```

```

end
end
surf(Pe,Jv,residual','edgecolor','none');
xlabel('Pe'); ylabel('Jv'); zlabel('residual error');
end

% a function to solve Jv in terms of Jv
function residual = myFunc(jv)
global pArt piSF piArt piISF Lp sigma thickness
deltaP = pArt-piSF; deltaPI = piArt-piSF;
pe = getPeclet(jv,thickness);
residual = jv - (Lp*(deltaP-sigma*piArt*(1-exp(-pe)/(1-sigma*exp(-pe)))));
end

% this procedure uses 2 equations to solve for Jv and peclet number
function residual = myFunc2(PeJv)
global pArt piSF piArt piISF Lp sigma thickness
deltaP = pArt-piSF; deltaPI = piArt-piSF;
residual = [0;0]; %set dimensions
residual(1) = PeJv(1)-getPeclet(PeJv(2),thickness);
residual(2) = PeJv(2) - (Lp*(deltaP-sigma*piArt*(1-exp(-PeJv(1))/(1-sigma*exp(-PeJv(1))))));
end

function plotConcentrationProfile
global pArt piSF piArt piISF Lp sigma Pd A thickness D
jw=10; x = 0:0.1:thickness;
L = thickness;
pe = getPecletOriginal(jw,L);
c1 = (piArt-piISF)/(1-exp(pe))*exp(getPecletOriginal(jw,x))+(piISF-piArt*exp(pe))/(1-exp(pe));
jw=-jw;
pe = getPecletOriginal(jw,L);
c2 = (piArt-piISF)/(1-exp(pe))*exp(getPecletOriginal(jw,x))+(piISF-piArt*exp(pe))/(1-exp(pe));
figure, plot(x,c1,x,c2)
end

% a modified function that never goes negative, ensures positive concentration
function pe = getPeclet(jw,x)
global sigma A D
pe = sqrt((jw.*(1-sigma)*x/D/A).^2);
end

function pe = getPecletOriginal(jw,x)
global sigma A D
pe = jw.*(1-sigma)*x/D/A;
end

% fun is a pointer to the function and dfun is a pointer to the jacobian
% % next version -- write iterates and error for each step, possibly with switch, xHistory
and then plot when done
function [xSolution,FresidualAtSolution,xHistory] = newton(fun,dfun,x0,tol,rememberHistory)
%rememberHistory is boolean
x = x0; err = fun(x)*fun(x); maxIter = 100; iter=1; xHistory(:,iter)=x0;
if isempty(dfun), useNumericalDiff = true;
else useNumericalDiff = false; end
while err > tol
F = fun(x);
if useNumericalDiff, J = getNumericalDerivative(fun,x);
else J = dfun(x); end
if abs(det(J)) < 1e-15, disp('singular Jacobian'); break; end %should
take a gradient step here
delX = - J\F;
alfa = getStepsizeArmijo(delX,x,fun,dfun);
xNew = x + alfa*delX;
if rememberHistory, xHistory(:,iter) = x; end % remember history %
err = sqrt(fun(xNew)*fun(xNew)); iter = iter+1;
if iter > maxIter,disp('max iteration exceeded'); break; end

```

```

        if norm(x-xNew) < 1e-6, disp(['no more update, saddle point found at ' num2str(xNew)]);
break; end
    x = xNew;
end
if rememberHistory, xHistory(:,iter) = x; end % remember history %
xSolution = x; FResidualAtSolution = fun(x);
end

function alfa = getStepsizeArmijo(delX,xOld,fun,dfun)
% % armijo linesearch using quadratic function fitting
alfa = 1; updating = true; delta = 0.1;
phiOld = fun(xOld)*fun(xOld);
    phiNew = fun(xOld + alfa*delX)*fun(xOld + alfa*delX);
while updating
    alfaNew = phiOld*alfa^2/((2*alfa - 1)*phiOld + phiNew);
    phiNew = fun(xOld + alfa*delX)*fun(xOld + alfa*delX);
    if alfaNew > 1, alfa = 1; break; %breaking criteria when it wants to go larger
    elseif phiNew-phiOld <= -2*delta*alfaNew*phiOld, break;
    else alfa = alfaNew; end
end
end

function alfa = getStepsizeForDecreasingResiduals(delX,xOld,fun)
% % primitive linesearch by halving the stepsize, may fail because no
% iteration stop criterion in the while loop
alfa = 1; errOld = fun(xOld)*fun(xOld);
errNew = fun(xOld+delX)*fun(xOld+delX);
while errNew > errOld
    alfa = 0.5* alfa;
    xNew = xOld + alfa*delX;
    errNew = fun(xNew)*fun(xNew);
end
if errNew > errOld, display('solution diverged'); alfa = -1; end;
end

% make n-dimensional
function J = getNumericalDerivative(fun,x)
N = length(x); J = zeros(N,N);
dX = 1e-6*ones(N,1);
for i = 1:N
    xNew = x; xNew(i) = xNew(i) + dX(i);
    dF = fun(xNew)-fun(x);
    J(:,i) = dF./dX;
end
end

function plotHistory(xHist,fun,xRange,yRange)
figure, plotNonLinearSurface(fun,xRange,yRange)
for i = 1:size(xHist,2)-1
    plot([xHist(1,i) xHist(1,i+1)], [xHist(2,i) xHist(2,i+1)], 'k', 'linewidth', 2)
    error(i) = fun(xHist(:,i))*fun(xHist(:,i));
end
error(i+1) = fun(xHist(:,i+1))*fun(xHist(:,i+1)); figure,
plot(error), xlabel('iteration number'); ylabel('residual error')

end

function plotNonLinearSurface(fun,xRange,yRange)
xPlot = xRange(1):0.1:xRange(2);
x2Plot = yRange(1):0.1:yRange(2);
for i = 1:length(xPlot)
    for j = 1:length(x2Plot)
        err = fun([xPlot(i);x2Plot(j)]);
        errorSurface(i,j) = err'*err;
    end
end
contour(xPlot,x2Plot,sqrt(errorSurface'),100,'b'), hold on
end

```

7.24 Appendix X: Newton method for solving nonlinear equations

The Newton-Raphson method (Newton method) is an iterative method for solving nonlinear sets of equations (root finding). A comprehensive comparison of the newton method for solving nonlinear equations and nonlinear optimization problems is given in a prior report [255]. This section describes the Newton method for solving nonlinear equations and specifically outlines implementation details and techniques for types of line searches for optimal stepsize control (Armijo line search). An overview, example implementation, and code for solving a 2-dimensional and 3-dimensional nonlinear set of equations is given. The mathematics presented are capable of solving higher order nonlinear equations assuming the user can define the functions. The functional partial derivative matrix (Jacobian matrix) can be given analytically or derived from a proposed numerical derivative.

7.24.1 Theory

The following text discussing the theory behind the Newton method is originally derived in previous work [255]. The method begins with an initial guess for the solution vector. This vector, if it is not the solution, can be improved using the Newton step in which each variable in the solution vector has a direction (positive or negative) and magnitude to approach a solution to the system of equations. The difference between the original solution vector and the update vector (after executing a Newton update step) is known as the *update vector*. This update vector, as with all vectors, has a direction (known as the *Newton direction*) and a magnitude. The update vector, $f(x^{k+1})$, can be derived from a multidimensional Taylor's expansion of the functions, $f(x)$, at point, x^k where f and x are vectors:

$$\begin{aligned}
f(x^{k+1}) &= f(x^k) + \nabla f(x^k) \Delta x^k \\
&= f(x^k) + J(x^k) \Delta x^k
\end{aligned} \tag{7.489}$$

The Newton step can be found by replacing the residual, $f(x)$, with its first order Taylor's expansion, \tilde{f} , in multidimensions and letting each equation be equal to zero. Equation (7.490) is expressed in vector form, where f and x are vectors and 0 is the null vector.

$$\begin{aligned}
\tilde{f}(x^{k+1}) &= f(x^k) + J(x^k) \Delta x^k \\
&= 0
\end{aligned} \tag{7.490}$$

The Newton direction, Δx^k , of the nonlinear equation set is given in Equation (7.491) where $J(x)$ at a given point, $x = x^k$, is the Jacobian matrix:

$$\Delta x^k = -J(x^k)^{-1} f(x^k) \tag{7.491}$$

Another method for deriving the Newton method can be obtained from minimizing the residual error surface, $\Phi(x)$, given in Equation (7.492). Accordingly, the Newton step is equal to the direction of the Newton minimizer of the residual error surface.

$$\Phi(x^k) = \frac{1}{2} f(x^k)^T f(x^k) \tag{7.492}$$

$$\tilde{\Phi}(x^k) = \frac{1}{2} \tilde{f}(x^k)^T \tilde{f}(x^k) \quad (7.493)$$

7.24.2 Implementation

An implementation is offered here for Equations (7.490)-(7.493). The source code is added in Section 7.24.5. Figure 7.171 shows a pseudocode and workflow diagram of an implementation of the Newton method.

```

1  x0 = x0;
2  FOR I = 1 DO 100 DO
3    J = computeJacobian(x0);
4    fx = evaluateFunction(x0);
5    dx = -J^-1 * fx;
6    alfa = getAlfa(fx,dx,x0);
7    xn = x0 + alfa*dx;
8    error = evaluateFunction(xn)';
9    evaluateFunction(xn)
10   x0 = xn; //update variables
11   IF error < tolerance, break; ENDIF
12  ENDFOR

```

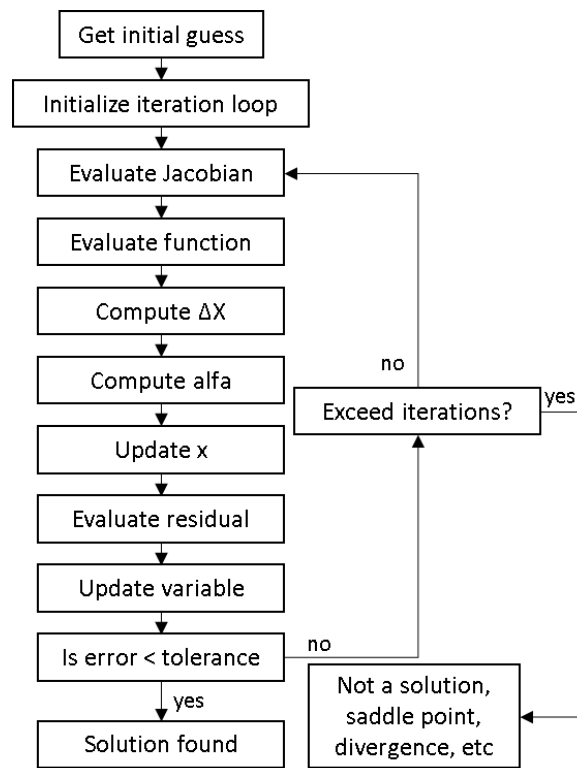


Figure 7.171. Information flow diagram and pseudo-code for the Newton method using stesize control.

A full matlab implementation is provided in section 7.24.5. Note, a max iteration must be set to avoid infinite looping inside a saddle point.

7.24.2.1 Stepsize control

In order to avoid divergence and to increase convergence speed, stepsize control must be employed when using the Newton method. This can be practically implemented by identifying and enforcing a stepsize (α) in Equation (7.494).

$$x^{\text{new}} = x^{\text{old}} + \alpha \Delta x \quad (7.494)$$

The simplest identification of a stepsize is to use a fixed scalar, such as 0.5. A slightly improved method would successively cut the stepsize value in half until the residual of the update is lower than the residual of the previous guess.

Armijo Linesearch. A more sophisticated algorithm to determine the optimal stepsize is the Armijo linesearch. This method simultaneously fits a surrogate 2nd order function to the nonlinear residual error function and finds its minimum. This quadratic line parameterized in α as visualized in Figure 7.172.

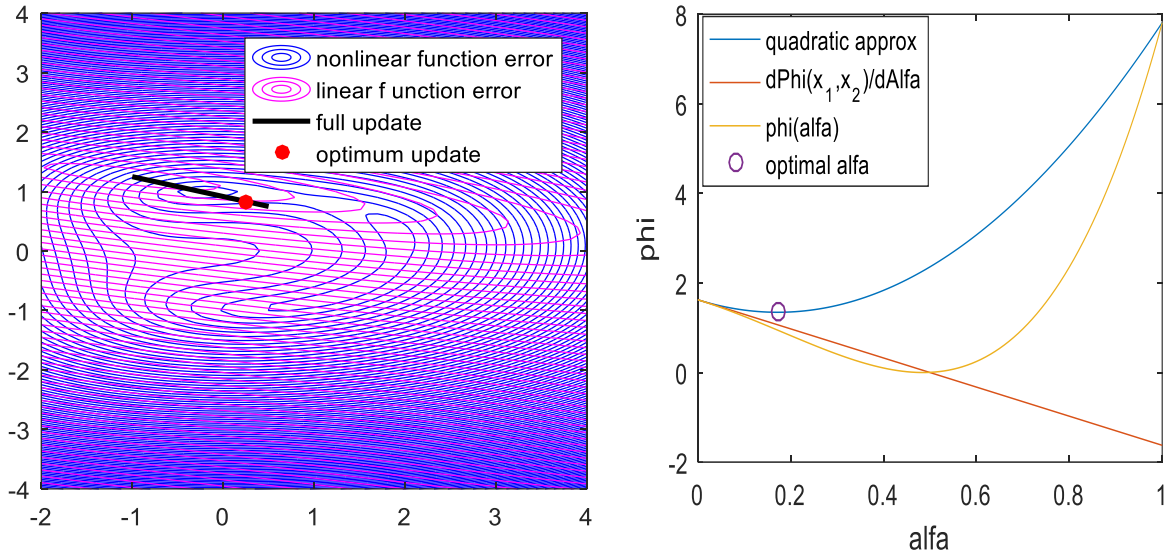


Figure 7.172. The visualization of error of the nonlinear function in the newton direction. The visualization of the surrogate Armijo chord in the newton direction and the optimal alpha as defined by the quadratic approximation are also pictured.

The method of fitting a quadratic curve and finding its minimum gives an analytic equation to computer the optimum stepsize (α^*). The stepsize is consecutively updated (α_k for the k^{th} guess of α) until it meets the condition in Equation (7.496). This condition enforces the update is an improvement beyond a linear update as described by Biegler [256] who suggests setting the minimum update, δ , to 0.1. Here, x^o refers to the old vector of x .

$$\alpha^* = \frac{\alpha_k^2 \Phi(x^o)}{(2\alpha_k - 1)\Phi(x^o) + \Phi(x^o + \alpha_k \Delta x)} \quad (7.495)$$

$$\Phi(x^o + \alpha_k \Delta x) - \Phi(x^o) \leq -2\delta \alpha^* \Phi(x^o) \quad (7.496)$$

7.24.3 Case Study 1 - Two Nonlinear Equations:

For the two nonlinear equations in Equation (7.497) are solved with the Newton method and stepsize control. This example is solved in Section 7.24.5. The convergence history is also shown using the residual error surface (Φ) and the linearized surface ($\tilde{\Phi}$).

$$f(x_1, x_2) = \begin{cases} f_1(x_1, x_2) = x_1^2 - 2x_1 - x_2 + 0.5 \\ f_2(x_1, x_2) = x_1^2 + 4x_2^2 - 4 \end{cases} \quad (7.497)$$

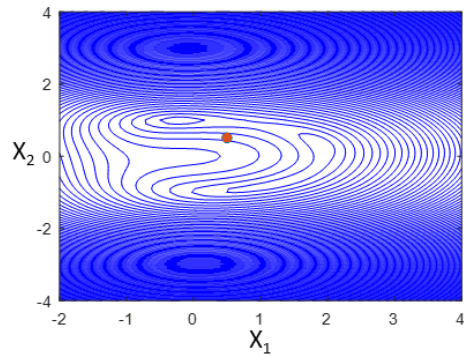


Figure 7.173. Residual error surface for Equation (7.497).

Contour plot of the error function in x_1 (x-axis) and x_2 (y-axis). Initial guess of $[0.5 \ 0.5]^T$ is marked in orange. The result $xSoln$ will be the solution vector, $fSoln$ will be the evaluation of $myProblem(xSoln)$, and $xHist$ will give the history of the value of x at each Newton step.

The convergence history (Figure 7.174 - Figure 7.177) shows the Newton method solving the linearized system iteratively. In some cases, the solution to the linearized system is not an improvement from the initial guess vector in the nonlinear error surface. This is the circumstance when the Armijo linesearch slows the update.

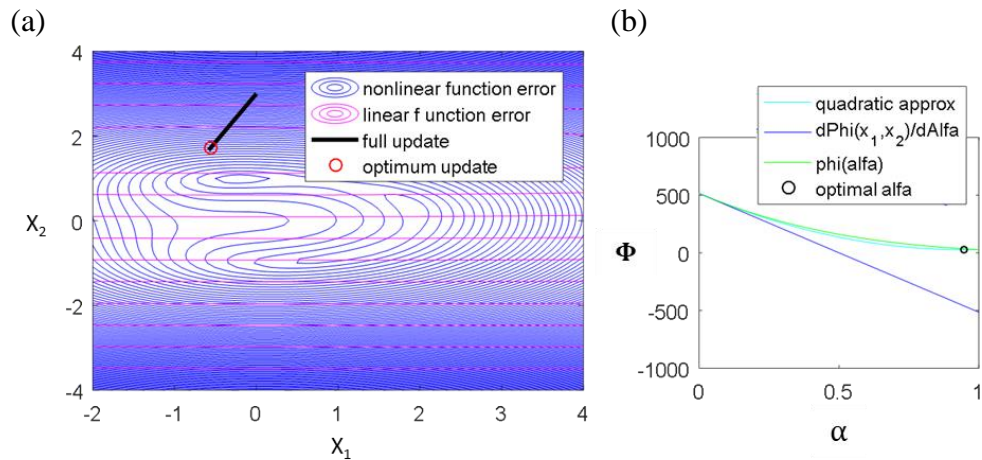


Figure 7.174. First Newton step with Armijo linesearch employed.

In this step, the stepsize needs to be reduced in order to give a reasonable update. (a) Full step reflects a convergence to the solution of the linearized system. (b) The update is reduced to the optimum stepsize using Armijo linesearch. There is no need to update alpha more than once.

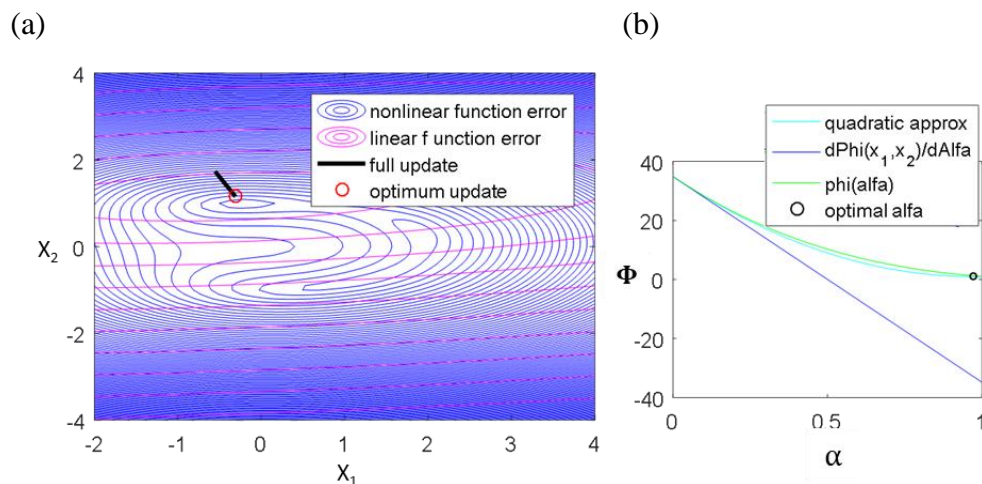


Figure 7.175. The second Newton step with Armijo linesearch employed.

In this step, the stepsize needs to be reduced in order to give a reasonable update. (a) Full step reflects a convergence to the solution of the linearized system. (b) The update is reduced to the optimum stepsize using Armijo linesearch. There is no need to update alpha more than once.

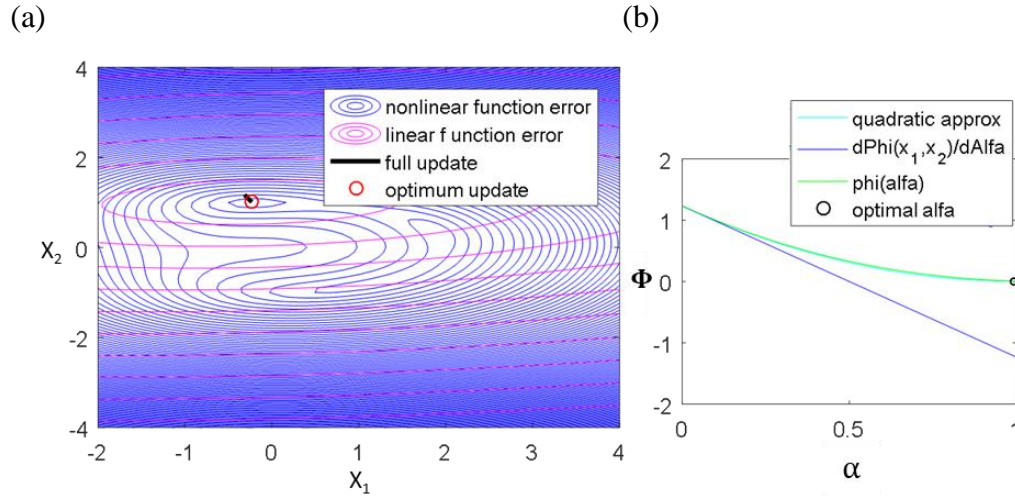


Figure 7.176. Third Newton step with Armijo linesearch employed. In this step, the stepsize needs to be reduced in order to give a reasonable update. (a) Full step reflects a convergence to the solution of the linearized system. (b) The optimal stepsize using Armijo linesearch was found to be 1. There is no need to update alpha more than once.

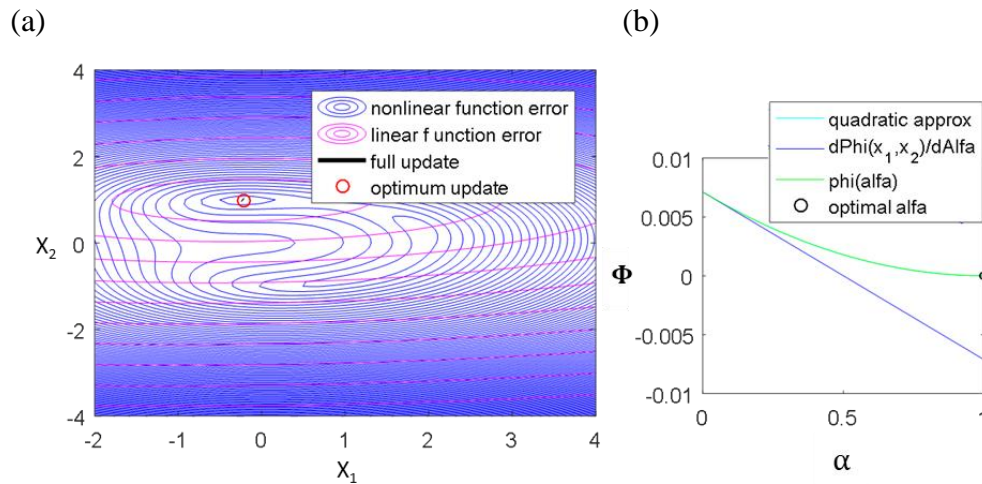


Figure 7.177. Solution to the nonlinear equations is found after a few more small updates (not visible in the figure). (a) Full step reflects a convergence to the solution of the linearized system. (b) The optimal stepsize using Armijo linesearch was found to be 1. There is no need to update alpha more than once. The final convergence may take more steps depending on the convergence criteria (residual tolerance).

7.24.4 Case Study 2, 2 nonlinear equations

The Newton method can be used to solve the following set of equations:

$$f(x_1, x_2) = \begin{cases} f_1(x_1, x_2) = x_1^2 - 2x_1 - x_2 + 0.5 \\ f_2(x_1, x_2) = x_1^2 + 4x_2^2 - 4 \end{cases} \quad (7.498)$$

The convergence history (Figure 7.178 - Figure 7.181) shows the Newton method solving the linearized system iteratively.

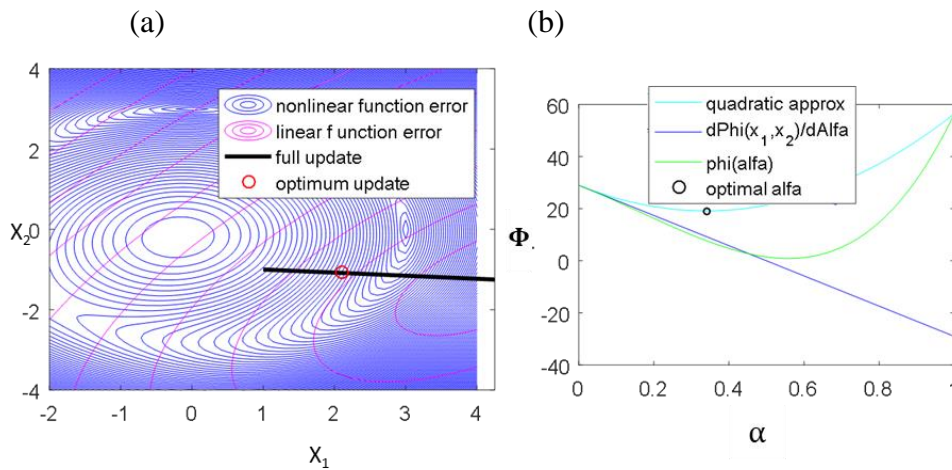


Figure 7.178. First Newton step with Armijo linesearch employed. In this step, the stepsize needs to be reduced in order to give a reasonable update.

(a) Full step reflects a convergence to the solution of the linearized system. (b) The update is reduced to the optimum stepsize using Armijo linesearch. There is no need to update alpha more than once.

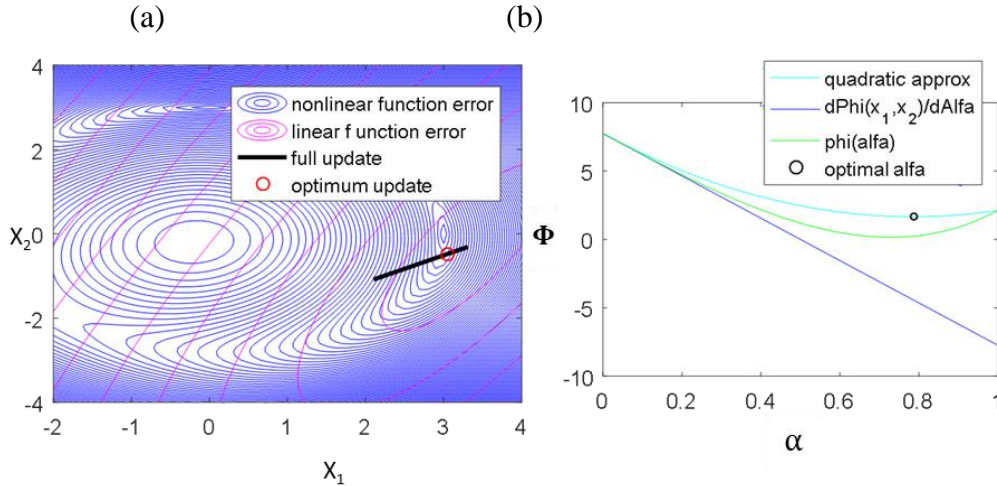


Figure 7.179. Second Newton step with Armijo linesearch employed. In this step, the stepsize needs to be reduced in order to give a reasonable update.

(a) Full step reflects a convergence to the solution of the linearized system. (b) The update is reduced to the optimum stepsize using Armijo linesearch. There is no need to update alpha more than once.

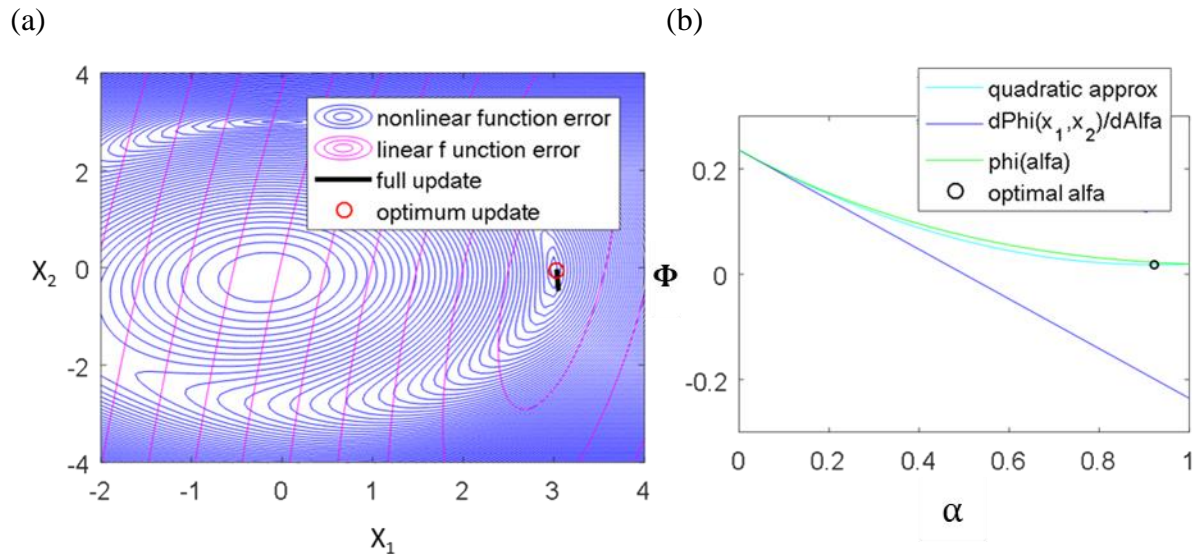


Figure 7.180. Third Newton step with Armijo linesearch employed. In this step, the stepsize needs to be reduced in order to give a reasonable update.

(a) Full step reflects a convergence to the solution of the linearized system. (b) The update is reduced to the optimum stepsize using Armijo linesearch. There is no need to update alpha more than once.

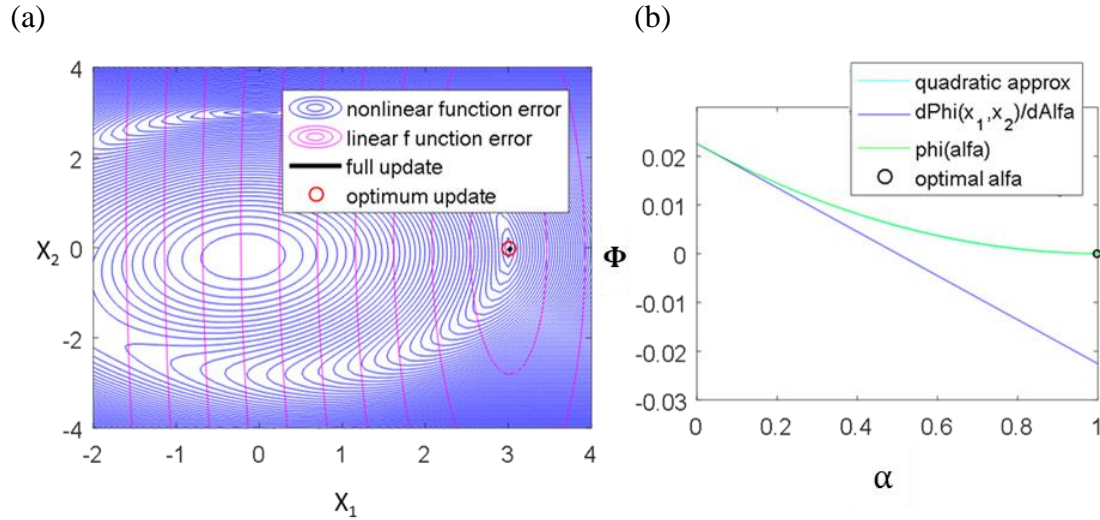


Figure 7.181. Solution to the nonlinear equations is found after a few more small updates (not visible in the figure).

(a) Full step reflects a convergence to the solution of the linearized system. (b) The optimal stepsize using Armijo linesearch was found to be 1.. There is no need to update α more than once. This final convergence may take more steps depending on the convergence criteria.

7.24.5 Matlab N-dimensional Newton code

This section provides a Matlab code that can solve an N-dimensional nonlinear algebraic system of equations using stepsize control. The code gives the option to using numerical derivative (instead of a hardcoded derivative matrix (Jacobian)) by simply passing a null pointer in the place of the Jacobian matrix. The code includes two 2-dimensional nonlinear example problems and one 3-dimensional nonlinear problem with convergent initial conditions for each case.

```

% A function created by Grant Hartung 2/1/2018 to solve nonlinear equations
function LPPDNewton
close all,
rememberHistory = true; tolerance = 1e-6;
% x0 = [0;3]; %converges to solution 2 for problem 1
% x0 = [0.5;0.5]; %converges to solution 1 for problem 1
% x0 = [0.5;0.75]; %converges to solution 1 for problem 1
% x0 = [1;-1]; %stops at saddle point for problem 1
% x0 = [1;1]; %converges to solution 1 for problem 1

% [xSoln,fSoln,xHist] = newton(@myProblem,@myProblemDerivative,x0,tolerance,rememberHistory)
% [xSoln,fSoln,xHist] = newton(@myProblem,[],x0,tolerance,rememberHistory)

% [xSoln,fSoln,xHist] = newton(@myProblem2,@myProblemDerivative2,x0,1e-6,true)

x0 = [1.5;1.5;1.5]; %converges to solution 1 for problem 3, [2;2;2]
[xSoln,fSoln,xHist] = newton(@myProblem3,@myProblemDerivative3,x0,tolerance,rememberHistory)
[xSoln,fSoln,xHist] = newton(@myProblem3,[],x0,tolerance,rememberHistory)

% plotHistory(xHist,@myProblem2)
end

% % add and validate 2 versions of this with overload

% fun is a pointer to the function and dfun is a pointer to the jacobian
% % next version -- write iterates and error for each step, possibly with switch, xHistory and then plot when done
% function [F,x] = newton(fun,dfun,x0,tol)
function [xSolution,FresidualAtSolution,xHistory] = newton(fun,dfun,x0,tol,rememberHistory) %rememberHistory is boolean
x = x0; err = fun(x)*fun(x); maxIter = 100; iter=1; xHistory(:,iter)=x0;
if isempty(dfun), useNumericalDiff = true;
else useNumericalDiff = false; end;
while err > tol
    F = fun(x);
    if useNumericalDiff, J = getNumericalDerivative(fun,x);
    else J = dfun(x); end;
    if abs(det(J)) < 1e-15, disp('singular Jacobian'); break; end %should take a gradient step here
    delX = - J\F;
%    alfa = getStepsizeForDecreasingResiduals(delX,x,fun); %stepsize control
    alfa = getStepsizeArmijo(delX,x,fun,dfun);
    xNew = x + alfa*delX;
    if rememberHistory, xHistory(:,iter) = x; end % remember history %
    err = sqrt(fun(xNew)*fun(xNew)); iter = iter+1;
    if iter > maxIter,disp('max iteration exceeded'); break; end
    if norm(x-xNew) < 1e-6, disp(['no more update, saddle point found at ' num2str(xNew)]); break; end
    x = xNew;
end
xSolution = x; FresidualAtSolution = fun(x);
end

function alfa = getStepsizeArmijo(delX,xOld,fun,dfun)
% % armijo linesearch using quadratic function fitting
alfa = 1; updating = true; delta = 0.1;
phiOld = fun(xOld)*fun(xOld);
phiNew = fun(xOld + alfa*delX)*fun(xOld + alfa*delX);
while updating
    alfaNew = phiOld*alfa^2/((2*alfa - 1)*phiOld + phiNew);
    phiNew = fun(xOld + alfa*delX)*fun(xOld + alfa*delX);
    if alfaNew > 1, alfa = 1; break; %breaking criteria when it wants to go larger
    elseif phiNew-phiOld <= -2*delta*alfaNew*phiOld, break;
    else alfa = alfaNew; end
end
end

function alfa = getStepsizeForDecreasingResiduals(delX,xOld,fun)
% % primitive linesearch by halving the stepsize, may fail because no
% iteration stop criterion in the while loop
alfa = 1; errOld = fun(xOld)*fun(xOld);

```



```

errNew = fun(xOld+delX)*fun(xOld+delX);
while errNew > errOld
    alfa = 0.5* alfa;
    xNew = xOld + alfa*delX;
    errNew = fun(xNew)*fun(xNew);
end
if errNew > errOld, display('solution diverged'); alfa = -1; end;
end

% make n-dimensional
function J = getNumericalDerivative(fun,x)
N = length(x); J = zeros(N,N);
dX = 1e-6*ones(N,1);
for i = 1:N
    xNew = x; xNew(i) = xNew(i) + dX(i);
    dF = fun(xNew)-fun(x);
    J(:,i) = dF./dX;
end
end

%% problem definitions
function FF = myProblem(x)
% two solutions are at: S1=[1.90068;0.31122] and S2 = [-0.2222; 0.9938]
FF = zeros(2,1);
FF(1) = x(1)^2 - 2*x(1) - x(2) + 0.5;
FF(2) = x(1)^2 + 4*x(2)^2 - 4;
end
function JJ = myProblemDerivative(x)
JJ(1,1) = 2*x(1)-2; JJ(1,2) = -1;
JJ(2,1) = 2*x(1); JJ(2,2) = 8*x(2);
end

function F = myProblem2(x)
f(1) = x(1) + x(2) - 3;
f(2) = x(1)^2 + x(2)^2 - 9;
F = [f(1);f(2)];
end
function dx = myProblemDerivative2(x)
df1 = [1 1];
df2 = [2*x(1) 2*x(2)];
dx = [df1;df2];
end

function F = myProblem3(x)
f(1) = x(1) + x(2) + x(3)^2 - 8;
f(2) = x(1)^2 + x(2)^2 - x(3) - 6;
f(3) = x(1)^3 + x(2)^4 - x(3)^2 - 20;
F = [f(1);f(2);f(3)];
end
function dx = myProblemDerivative3(x)
df1 = [1 1 2*x(3)];
df2 = [2*x(1) 2*x(2) -1];
df3 = [3*x(1)^2 4*x(2)^3 -2*x(3)];
dx = [df1;df2;df3];
end

function plotHistory(xHist,fun)
figure, plotNonLinearSurface(fun)
for i = 1:size(xHist,2)-1
    plot([xHist(1,i) xHist(1,i+1)], [xHist(2,i) xHist(2,i+1)], 'k', 'linewidth', 2)
    error(i) = fun(xHist(:,i))*fun(xHist(:,i));
end
error(i+1) = fun(xHist(:,i+1))*fun(xHist(:,i+1)); figure,
plot(error), xlabel('iteration number'); ylabel('residual error')
end

```

7.24.6 Derivation of Armijo linesearch optimal stepsize

In order to fit a second order residual error curve to the nonlinear error curve, a parametric curve in N-dimensional space will be defined by the error, r , parameterized in stepsize α as in Equation (7.499).

$$r = c_1 \alpha^2 + c_2 \alpha + c_3 \quad (7.499)$$

This function has 3 unknowns for which 3 pieces of information are required. The first two pieces are the residual at $\alpha=0$ and $\alpha=1$. The third piece of information is that the gradient at x^{old} is the same. These pieces of information are expressed in Equation (7.501).

$$\Phi = F^T F \quad (7.500)$$

$$r(0) = F(x^{\text{old}})^T F(x^{\text{old}})$$

$$r(1) = F(x^{\text{new}})^T F(x^{\text{new}}) \quad (7.501)$$

$$\frac{dr(0)}{d\alpha} = \frac{d\Phi(x^{\text{old}})}{d\alpha}$$

At this point, Φ is parameterized in all independent variables (x), not in the parametric independent variable, α . In order to transpose these derivatives from the multivariable x space into the parametric α space, the chain rule is used as in Equation (7.502). Here, $dx/d\alpha$ is constant for $\alpha \in 0,1$ so this ratio can be evaluated numerically by choosing a value for α (say 0.1) and calculating the change in x corresponding to that value of α .

The dot product of the transpose of the gradient in the x -dimension and the gradient of the x -dimension with the alpha dimension is analogous to the summation of the transposed gradient vectors pertaining x_1 and x_2 . The resulting change with respect to α accounts for the cumulative changes in all dimensions of x simultaneously.

$$\frac{d\Phi(x^{\text{old}})}{d\alpha} = \frac{d\Phi(x^{\text{old}})^T}{dx} \cdot \frac{dx}{d\alpha}$$

Where

$$\frac{d\Phi(x^{\text{old}})}{dx} = J(x^{\text{old}})^T F(x^{\text{old}}) \quad (7.502)$$

$$\frac{dx}{d\alpha} = \begin{bmatrix} dx_1/d\alpha \\ dx_2/d\alpha \end{bmatrix}$$

$dx/d\alpha$ can be evaluated using Equation (7.503) when choosing any value for α .

$$\begin{aligned} \frac{dx_1}{d\alpha} &= \frac{\alpha \Delta x_1}{\alpha} \\ \frac{dx_2}{d\alpha} &= \frac{\alpha \Delta x_2}{\alpha} \end{aligned} \quad (7.503)$$

Once evaluated, the final solution for the gradient $d\Phi(x^{\text{old}})/(d\alpha)$ can be simplified as in Equation (7.504).

$$\begin{aligned}
\frac{d\Phi(x^{\text{old}})}{d\alpha} &= \left[J(x^{\text{old}})^T F(x^{\text{old}}) \right]^T \cdot \frac{\epsilon \Delta x}{\epsilon} \\
\frac{d\Phi(x^{\text{old}})}{d\alpha} &= F(x^{\text{old}})^T J(x^{\text{old}}) \cdot -J(x^{\text{old}})^{-1} F(x^{\text{old}}) \\
\frac{d\Phi(x^{\text{old}})}{d\alpha} &= -F(x^{\text{old}})^T F(x^{\text{old}}) \\
\frac{d\Phi(x^{\text{old}})}{d\alpha} &= -2\Phi(x^{\text{old}})
\end{aligned} \tag{7.504}$$

This operation (from Equation (7.503)) can also be viewed as the projection of the gradient into the Newton direction. This projection will occur when performing the inner product (dot product) between the gradient and the Newton direction as seen in Equation (7.504). To perform the inner product of these 2 vectors, the first vector can be transposed.

$$\begin{aligned}
\frac{d\Phi(x^{\text{old}})}{d\alpha} &= \frac{d\Phi(x^{\text{old}})}{dx} \cdot \Delta x \\
\frac{d\Phi(x^{\text{old}})}{d\alpha} &= \left[J(x^{\text{old}})^T F(x^{\text{old}}) \right]^T J(x^{\text{old}})^{-1} F(x^{\text{old}}) \\
\frac{d\Phi(x^{\text{old}})}{d\alpha} &= F(x^{\text{old}})^T J(x^{\text{old}}) \cdot -J(x^{\text{old}})^{-1} F(x^{\text{old}}) \\
\frac{d\Phi(x^{\text{old}})}{d\alpha} &= -F(x^{\text{old}})^T F(x^{\text{old}}) \\
\frac{d\Phi(x^{\text{old}})}{d\alpha} &= -2\Phi(x^{\text{old}})
\end{aligned} \tag{7.505}$$

The final step to evaluate the coefficients can be seen for our example in equation (7.506).

$$\begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} F(x^{old})^T F(x^{old}) \\ F(x^{new})^T F(x^{new}) \\ -2\Phi(x^{old}) \end{bmatrix} \quad (7.506)$$

This is a perfectly defined system and has 1 unique solution (because the A matrix is regular). A parametric curve in α is now defined that constitutes a pseudo-residual function. The minimum of this new curve exists where the derivative is 0 which is described by Equation (7.507).

$$\begin{aligned} \frac{dr}{d\alpha} = 0 &= c_1\alpha + c_2 \\ \alpha^{opt} &= -c_2/c_1 \end{aligned} \quad (7.507)$$

7.25 Appendix Y: Properties of linear algebraic set of equations

A brief overview of the matrix properties is given in this section, including an in-depth look at the gradient, residual and the energy of a matrix. Regardless of how the equations are formed, they can be considered a series of n-dimensional lines/planes/hyperplanes in n-dimensional space. The solution of the system of equations is the intersection of all lines simultaneously, which only occurs once in a fully defined system. An underdefined system will have an infinite number of solutions (an intersection along an entire line). An overdefined system has infinite solutions or no solution. For the remainder of this section, the discussion will revolve around a system of 2 equations in 2 dimensions, allowing plotting functions to visualize concepts, but this does not preclude the topics to be readily expanded to n-dimensional space using the same concepts.

Terminology. The terminology of this section assumes the linear algebraic system takes the form of:

$$Ax = b$$

Where

$$A = \begin{bmatrix} c_{1,1} & c_{1,2} \\ c_{2,1} & c_{2,2} \end{bmatrix}, \quad x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, \quad b = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \quad (7.508)$$

7.25.1 Residual

The residual form of a linear system of equations follows:

$$Ax - b = r \quad (7.509)$$

Where r is the residual of the equation. This vector has a magnitude, which is a measure of how far the value of x is from the solution to the system of equations (the intersection of the 2

lines). As a vector, the value of the 1st element is the value of error (the distance) the coordinate of x is from the 1st line, and the second element of r evaluates the distance from the second line.

The transposition of a guess vector from Cartesian coordinate system (x - y coordinates) into the oblique coordinate system made up from the 2 vectors that are the A matrix and b vector (v - w coordinates) assists in visualizing the error. The origin of the oblique coordinate system is the solution to the system of equations. Note, the new coordinates of the guess vector after being transposed into the oblique coordinate system make up a new vector from the new origin to the new point. The length of this vector (or square of the length of this vector) is a fair measure of the “wrong-ness” of the guess vector. This vector is known as the residual vector and is a measure of coordinates (error) in the oblique coordinate system from each axis (each axis is a single equation).

In other words, the operation of evaluating the residual error is actually performing a coordinate transformation of a point in Cartesian coordinates into the oblique coordinate system and evaluating the projection of the error onto each base vector. This is visualized in Figure 7.182.

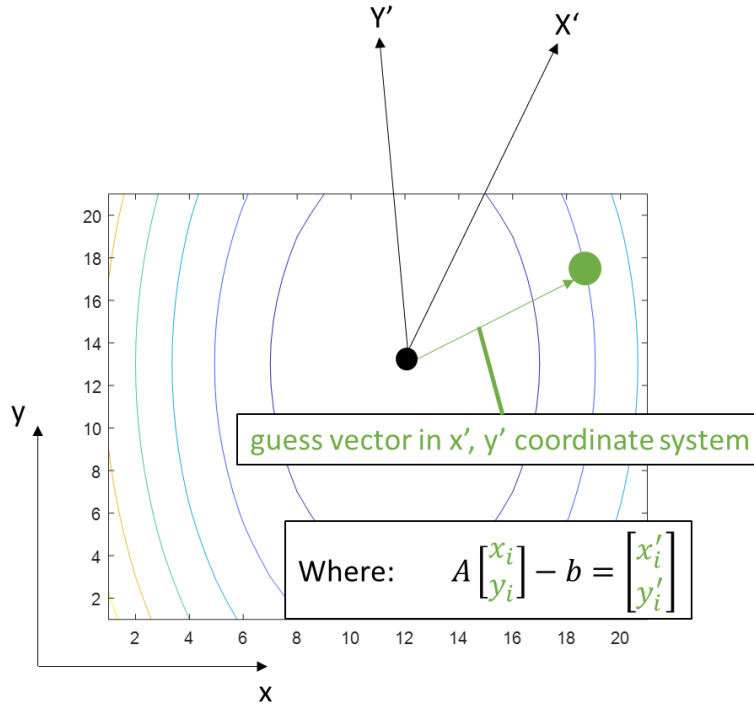


Figure 7.182. Visualization of the residual error on a 2x2 system of equations.

The residual is a measure of error that expresses the wrong-ness of the guess from each base vector of the new (oblique) coordinate system. Note, the calculation of the residual vector is effectively a coordinate transformation of the x vector.

7.25.2 Gradient

The gradient of the linear algebraic system is defined by:

$$g = (Ax - b) \quad (7.510)$$

Which is the same as the residual vector. A better method for imagining the gradient is to think of the residual form of the equations being a coordinate transformation between the X-Y Cartesian

coordinate system and the oblique base vectors (defined by the 2 lines). The \mathbf{b} -vector is the translation of the coordinate system to the origin and the \mathbf{A} matrix is the rotational/scaling matrix for any coordinate.

Under this paradigm, the goal of solving the linear algebraic system is to find the origin of the oblique coordinate system in Cartesian coordinates. In this oblique coordinate system, the residual is a local point coordinate. The vector starting at the origin in the oblique coordinate system (solution of the equations) and the current value of x gives the direction from the solution to the point, the negative of which would be an arrow pointing from the current point to the solution. The reason that evaluating this does not simply solve the system of the equations, is that the new vector (x_i', y_i') is still in the oblique coordinate system. Transforming this vector into the original coordinate system is equivalent to solving the linear algebraic system.

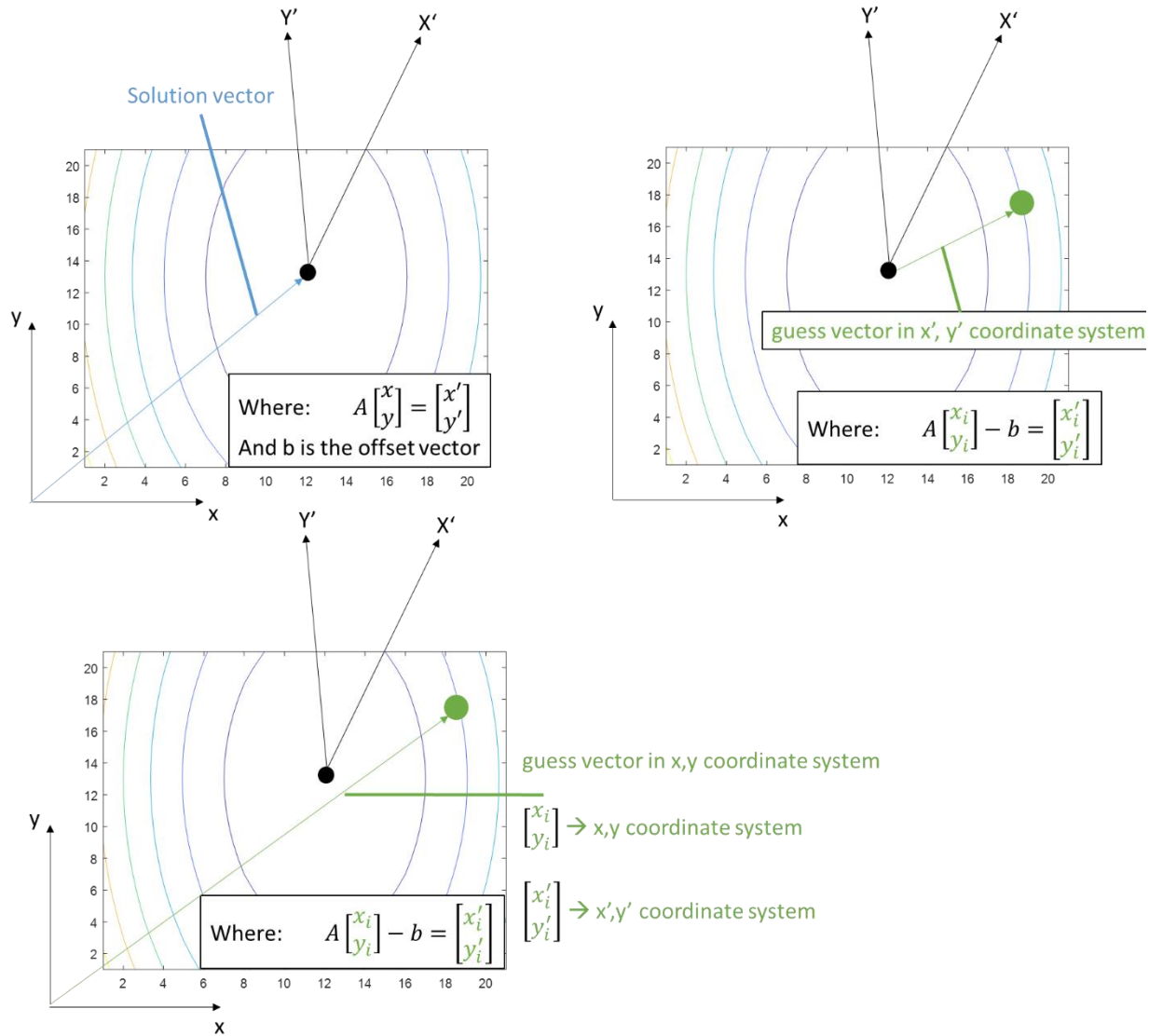


Figure 7.183. Graphic representation of the coordinate transformation expressing the gradient of the function in terms of the matrix A and vectors b and x guess.

Top Left) The solution to the system of equations and the Cartesian coordinate system (x, y) and the oblique coordinate system (x', y'). Top Right) The solution vector in the oblique coordinate system is a transformation of the guess vector from Cartesian space to the oblique space using the A matrix and b vector. Bottom Left) The guess vector in Cartesian coordinates.

7.25.3 Energy

Now that the gradient has been identified, it is imperative to integrate this function to identify the equation for the surface representing this coordinate system. If the gradient is the derivative of

the system of equations, then the energy represents the integral of the gradient, representing the system itself. This can be shown:

$$\begin{aligned} E &= \int g \, dx = \int (Ax - b) \, dx \\ E &= \frac{1}{2} Ax^2 - bx + c \end{aligned} \tag{7.511}$$

Which can be written in proper matrix notation (the integral of a matrix pre-multiplies the coefficients by a transpose of the x-vector) and with the assumption of a 0-Z offset:

$$E = \frac{1}{2} x^T Ax - x^T b \tag{7.512}$$

7.26 Appendix Z: Idealized brain geometry

The simplest method for investigating brain distribution of species is to use an idealized geometry. This may not be the most spatially accurate simulation, but it is easily amenable to parametric studies. For instance, the production of D-2-hydroxyglutarate (D2HG) by isocitrate dehydrogenase 1 (IDH1) or IDH2 tumor cells causes epileptic-like seizures if large enough portions of the brain exhibit a concentration greater than 3 millimolar [257]. The extrapolation of data from cell cultures and CSF measurements allows a modeler to investigate how the concentration profile may change while varying the unknown parameters of production rate and diffusivity. The resulting profiles can be examined for physiological values, such as maximum tumoral concentration, to validate which coefficients are most likely to be accurate. Such a primitive simulation is only a rough estimate of effected region (region with a concentration above a threshold) but the prediction of transport parameters can also be used to inform a simulation of a 3D reconstruction as in [183].

A numerical model of the reaction-diffusion, Equation (7.513) was created using an idealized radially symmetric model. The radius (r) of the spherical system covers the concentric compartments as in Figure 7.184. This steady-states species balance equates the diffusive flux with the production rate of the D2HG protein. Here the concentration in each volume of D2HG protein is C_A , and the diffusion coefficient is D_i . The protein production rate in of the compartment is given by a zeroth order reaction rate, $V_i S_{A,i}$ where V_i is the volume of the element. The production rate ($S_{A,i}$) is equal to $\overline{S_A}$ in the tumor domain (Ω_t). In the other two compartments, brain tissue (Ω_b), and CSF (Ω_{csf}), there is no protein production and hence the reaction rate is 0.

This tumor produces D2HG which spreads by diffusion radially through the brain tissue into the CSF compartment. D2HG is cleared from the CSF by passive transport together with the reabsorbed CSF (unhindered clearance) with the flux of protein into the CSF (f_{CSF}) defined as a function of CSF bulk flow (q_{CSF}) and the concentration of the solute in the CSF ($C_{A,CSF}$) given in Equation (7.515). Symmetry is assumed at the center of the tumor mass.

$$\vec{\nabla} D_i \vec{\nabla} C_{A,i} + V_i S_{A,i} = 0 \quad (7.513)$$

$$\begin{cases} S_{A,i} = \overline{S}_A & x \in \Omega_t \\ S_{A,i} = 0 & x \in \Omega_b, \Omega_{CSF} \end{cases} \quad (7.514)$$

$$f_{CSF} = q_{CSF} C_{A,CSF} \quad (7.515)$$

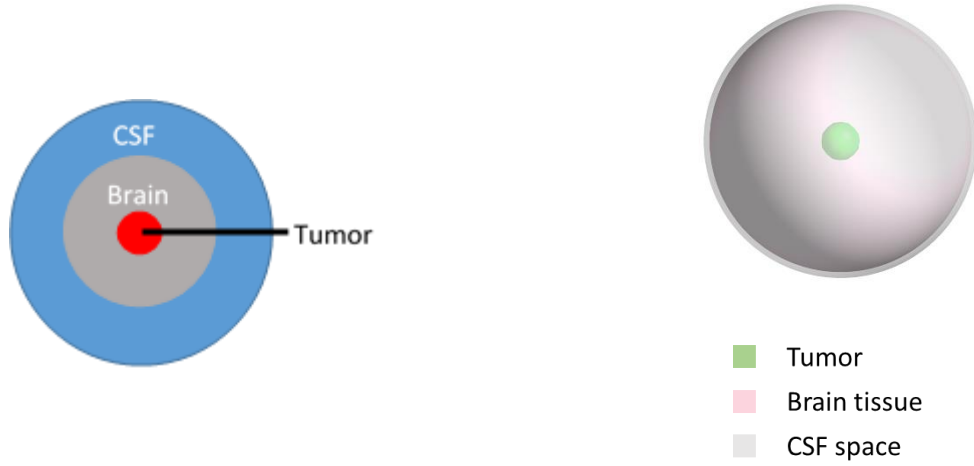


Figure 7.184. Schematic representation of the idealized system. Red delineates the tumor domain, grey is the brain matter and blue is the csf compartment.

The production rate of the tumor mass was predicted using mass conservation between production and clearance as reflected by Equation (7.516). The parameters used can be seen in Table 7.51 alongside the predicted range of production rate of the tumor mass.

$$S_A V_{tumor} - q_{CSF} C_{CSF} = 0 \quad (7.516)$$

Table 7.51: The given values used to predict the generation rate of the protein

Parameter	Value
r_{tumor}	1.5299 cm
C_{CSF}	0.0075 - 0.1 $\times 10^{-6}$ mol/cm ³
q_{CSF}	500 cm ³ /day (5.79e ⁻³ cm ³ /s)
Predicted S_A	2.894 – 12.867 $\times 10^{-12}$ mol/cm ³ /s

The steady-state concentration was formulated in spherical coordinates and solved using Matlab. The spherical model was implemented with both a finite element method and a finite volume method. Both methods were tested for mesh independence and had a difference of $<10e-6$.

Using Equations (7.513) - (7.516), the idealized radially symmetric model shown in Figure 7.184, and the parameters listed in Table 7.51, we simulated a range of D2HG release rates (S_A) and D2HG diffusivity constants within the tumor, brain, and CSF (Figure 7.185Figure). These simulations generated a range of D2HG concentrations in a 2 cm radius around an IDH1^{mut} tumor, with a consistently rapid drop in D2HG away from the tumor center. At all the simulated diffusion constants, D2HG release rates between $2.9-12.9 \times 10^{-12}$ moles/sec generated curves that fit the previously reported steady-state concentrations of D2HG within IDH1^{mut} gliomas,¹ as well as the range of D2HG previously detected in the CSF of IDH1^{mut} glioma patients.⁶ This range of D2HG release rates also fully encompassed the rates that were independently extrapolated from *in vitro* data, $3.2-5.8 \times 10^{-12}$ moles/sec (data in [183]), strengthening the validity of these mathematical models. The implementation of both finite volume method as well as finite element methods were compared. The results indicated the finite volume method was able to converge with significantly lower resolution.

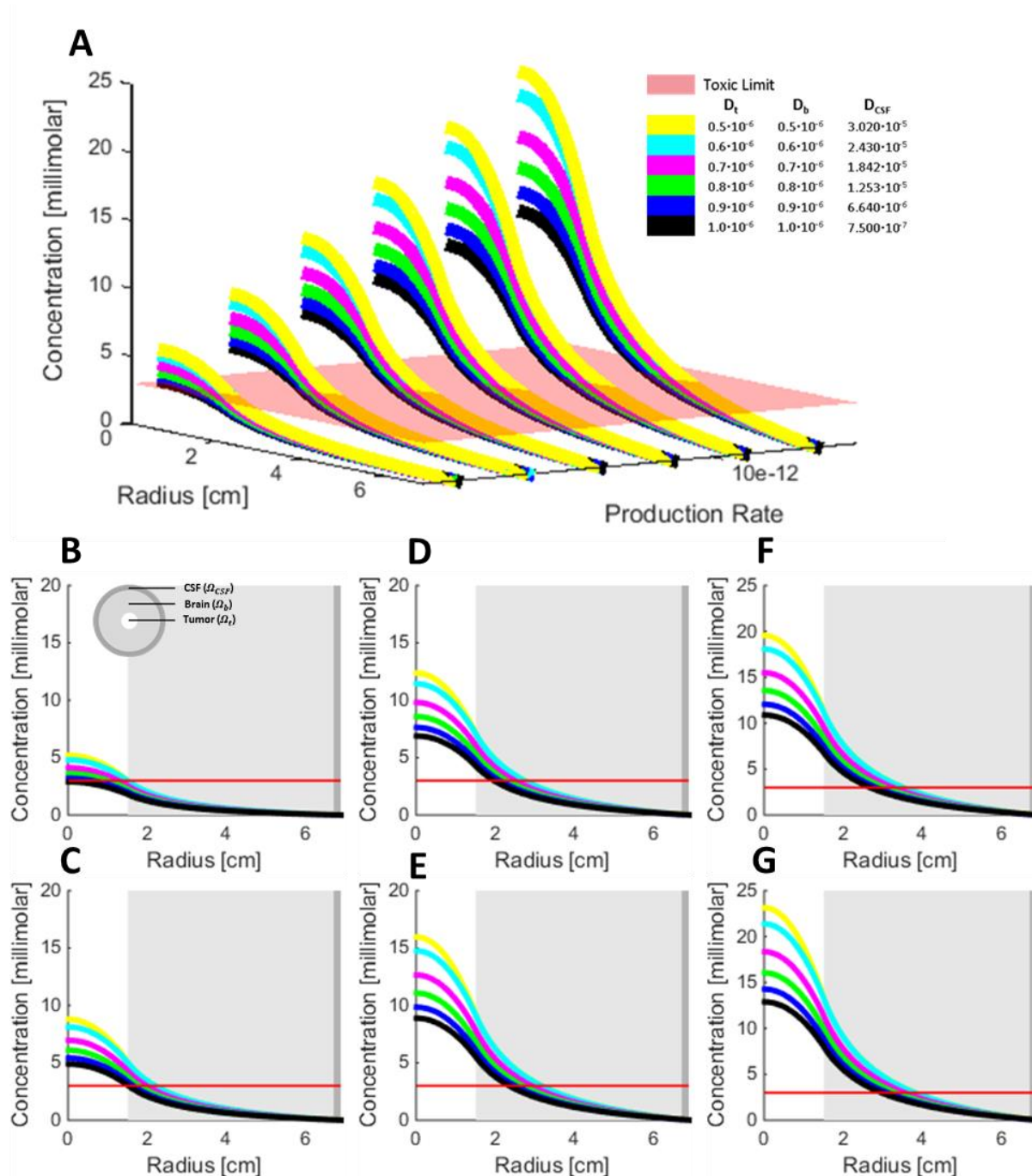


Figure 7.185. Biometric study of differing D2HG release rates and diffusivities.

Differential concentration profiles as a function of tissue diffusivity (D) and protein production rate (S_A) in the tumor. The diffusivities are color-coded as can be reviewed in panel A, where multiple concentration profiles are plotted against each other with differing D2HG diffusivities and production rates. Panels B-G show each D2HG S_A separately, as follows: B) $S_A = 2.89 \cdot 10^{-12}$ moles/ml/sec; C) $S_A = 4.89 \cdot 10^{-12}$; D) $S_A = 6.88 \cdot 10^{-12}$; E) $S_A = 8.88 \cdot 10^{-12}$; F) $S_A = 10.87$

$\times 10^{-12}$; G) $S_A = 12.87 \times 10^{-12}$. The red line is the toxic limit in the tissue, as indicated by the minimum D2HG concentration required to increase neuronal network bursts.¹³ D_t = diffusion constant through tumor; D_b = diffusion constant through brain; D_{CSF} = diffusion constant through CSF.*

This information was later incorporated into a 3D reconstruction simulation, with parameters chosen from the parametric study which matched the intratumoral concentration. The simulation using a 3D reconstruction predicted 7.3% of the brain affected by toxic levels of the protein (Figure 4 in [183]).

7.27 Appendix AA: Validation of spherical geometry for diffusion reaction system

A numerical model of the reaction-diffusion system is represented by Equation (7.517). The discretized form was implemented on an idealized, radially symmetric model as pictured in Figure 7.186. Radial D₂HG concentration profiles, $C_{CSF}(r)$, were simulated as a function of radius r in spherical coordinates with the tumor at its center, $r = 0$. The three compartments include tumor, brain, and CSF space using a 3D spherical coordinate system are shown in Figure 7.227. This tumor produces D₂HG which spreads by diffusion radially through tissue into the CSF compartment. D₂HG is cleared from the CSF by passive transport together with the reabsorbed CSF (unhindered clearance) as given in Equation (7.518). Symmetry is assumed at the center of the tumor mass.

$$\vec{\nabla} D \vec{\nabla} C_A + V_{tumor} S_A = 0 \quad (7.517)$$

$$\begin{cases} S_A = \overline{S_A} & \text{in the tumor} \\ S_A = 0 & \text{in the tissue and CSF} \end{cases}$$

$$\text{Drug outflow} = q_{CSF} C_{A,CSF} \quad (7.518)$$



Figure 7.186. Schematic representation of the idealized system.

Left) 2D picture of the 3D domain where red delineates the tumor domain, grey is the brain matter and blue is the CSF compartment. Right) a 3D visualization of the domain is also offered.

The production rate of the tumor mass was predicted using mass conservation between production and clearance as reflected by Equation (7.519). Relevant parameters of the model are given in Table 7.52 as well as the predicted range of production rate of the tumor mass.

$$S_A V_{tumor} - q_{CSF} c_{CSF} = 0 \quad (7.519)$$

Table 7.52. The given values used to predict the generation rate of the protein

Parameter	Value
r_{tumor}	1.5299 cm
c_{CSF}	0.0075 - 0.1 x10 ⁻⁶ mol/cm ³
q_{CSF}	500 cm ³ /day (5.79e ⁻³ cm ³ /s)
Predicted S_A	2.894 – 12.867 x10 ⁻¹² mol/cm ³ /s

The steady-state concentration predicted using a spherical coordinate system in a proprietary Matlab program. The spherical model was implemented with both a finite element method and a finite volume method. Both methods were investigated for mesh independence and considered achieving this when the update residual was <10e-6.

7.27.1 1D symmetric simulation with a range of production rates and diffusivity rates

7.27.1.1 Consistency test of the production rate

The overall mass balance in Equation (7.520) relates tumor volume (V_{tumor}), production rate (S_A) and CSF concentration (c_{CSF}).

$$S_A V_{tumor} - q_{CSF} c_{CSF} = 0 \quad (7.520)$$

A discrepancy was immediately found between the literature values. Namely, the overall mass balance does not hold given these values as summarized in Equation (7.521).

$$production = removal$$

$$S_A V_{tumor} = q_{CSF} c_{CSF} \quad (7.521)$$

$$1.0412 \times 10^{-8} \neq 5.787 \times 10^{-10}$$

With given Values:

$$V_{tumor} = 4\pi (1.5299)^3 \text{ cm}^3, \quad c_{CSF} = 0.1 \text{e}^{-6} \text{ mol/cm}^3,$$

$$q_{CSF} = 500 \text{ cm}^3/\text{day} = 5.79 \text{e}^{-3} \text{ cm}^3/\text{s}$$

Without knowledge of what parameter is more or less accurate, a selection of permutations spanning the relevant ranges were simulated. Each set permutation was identified by fixing one of the values in the desired range and calculating the corresponding values that adhere to mass balance as in Equation (7.522). The range of production values (S_A) and diffusivity constants are summarized in Table 9.53 and Figure 7.187. The production rates tested can be seen in Table 7.54.

$$production = removal$$

$$S_A = \frac{q_{CSF} c_{CSF}}{V_{tumor}} \quad (7.522)$$

$$2.894 \times 10^{-12} \leq S_A \leq 38.574 \times 10^{-12}$$

Table 7.53. Corresponding diffusivities to the plotted variables

	D_{tumor}	D_{brain}	D_{csf}
	5×10^{-7}	5×10^{-7}	3.02×10^{-5}
	6×10^{-7}	6×10^{-7}	2.43×10^{-5}
	7×10^{-7}	7×10^{-7}	1.842×10^{-5}
	8×10^{-7}	8×10^{-7}	1.253×10^{-5}
	9×10^{-7}	9×10^{-7}	6.64×10^{-6}
	1×10^{-6}	1×10^{-6}	7.5×10^{-7}

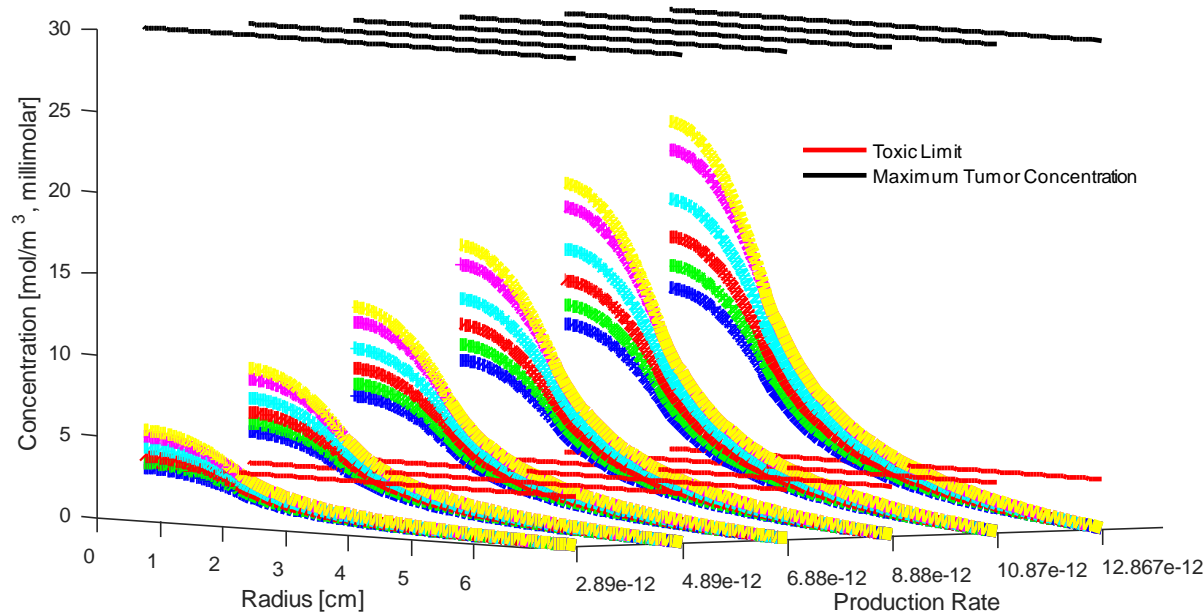


Figure 7.187. Biometric study of differing material diffusivities and tumor production rates. The black line corresponds to the maximum recorded concentration within the tumor and the red line is the toxic limit in the tissue. The diffusivities are color-coded as can be reviewed in Table 9.53.

Table 7.54. Differential production rates (R_A) used in biometric study. (mol/cm³/s)

2.89×10^{-12}	4.89×10^{-12}	6.88×10^{-12}	8.88×10^{-12}	10.87×10^{-12}	12.867×10^{-12}
------------------------	------------------------	------------------------	------------------------	-------------------------	--------------------------

7.27.2 Validation

Spot checks. Mass conservation for many volume (spherical shell) was investigated to ensure the inflow and outflow matched as in Equation (7.523). For a detailed examination of these spot checks for the first simulation results, refer to Section 7.27.5.

$$q_{in} + S_A V - q_{out} = 0 \quad (7.523)$$

Overall Conservation. The overall conservation was also interrogated by calculating the total tumor production and CSF clearance rates and ensuring they satisfy Equation (7.519). Note, this is an indicator of matrix solvability as the boundary conditions were chosen to enforce Equation (7.519).

Analytical Solution. The numerical integration scheme was validated against the analytical solution given in Equation (7.524). The finite volume method (FVM) method passed this test with $N=50$ nodes in the tissue domain, 1 node for the tumor mass and 1 node for the CSF space as shown in Figure 7.188.

$$c(r) = \frac{\bar{q} r_{max}^2}{D} \left(\frac{1}{r} - \frac{1}{r_{min}} \right) + k_{prod} \quad (7.524)$$

$$\bar{q} = \frac{\bar{f}}{SA_{\perp}}$$

$$k_{prod} = k \frac{4}{3} \pi r_t^3$$

Where r_{max}^2 is the radius of the CSF boundary edge, k_{prod} is the total tumor production (rate of production multiplied by the tumor volume), r_{min} is the radius of the first volume element, D is the diffusivity of the species, \bar{f} is the turnover of CSF flow (the convective source in the CSF compartment), SA_{\perp} is the surface area at the CSF-tissue interface, and \bar{q} is the species flux into the CSF at the boundary.

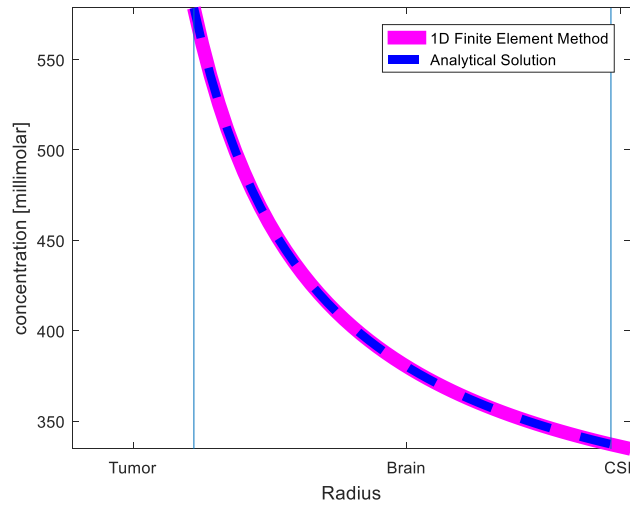


Figure 7.188. The comparison of the finite volume method against the analytic solution shows excellent agreement.

This result demonstrates robustness of implementation for the reaction-diffusion system. Note, the values for diffusivity and production rate in this case study are not the same as for the actual simulation.

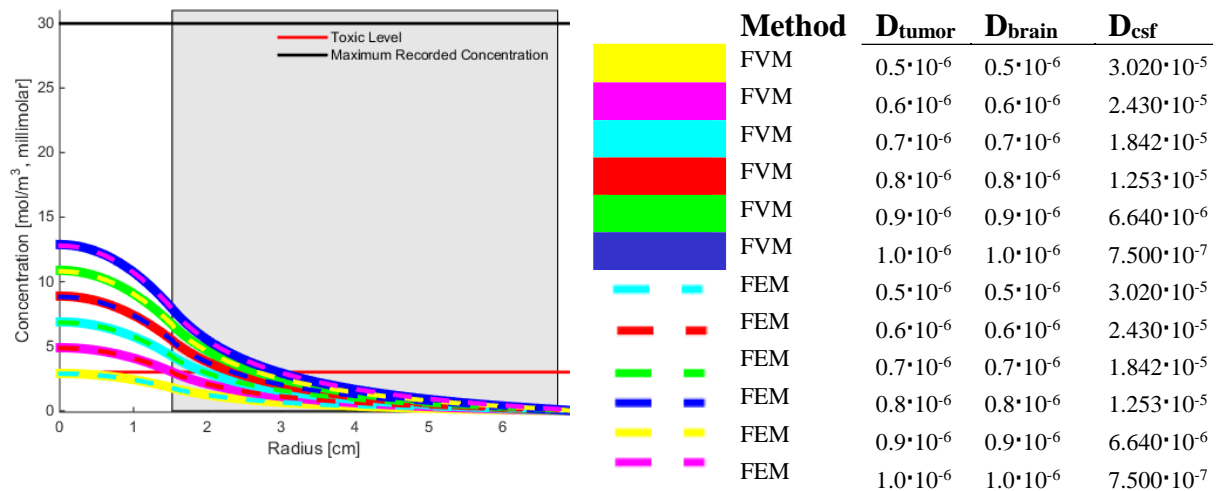
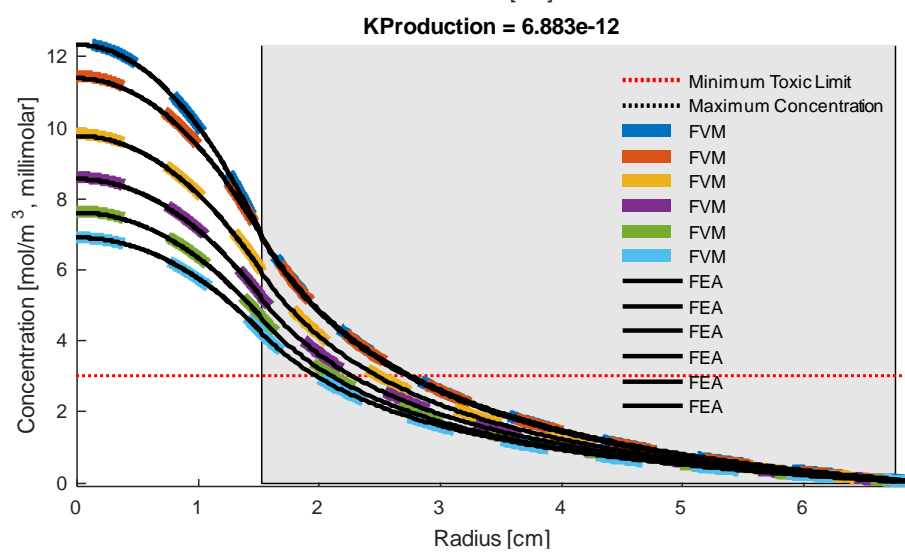
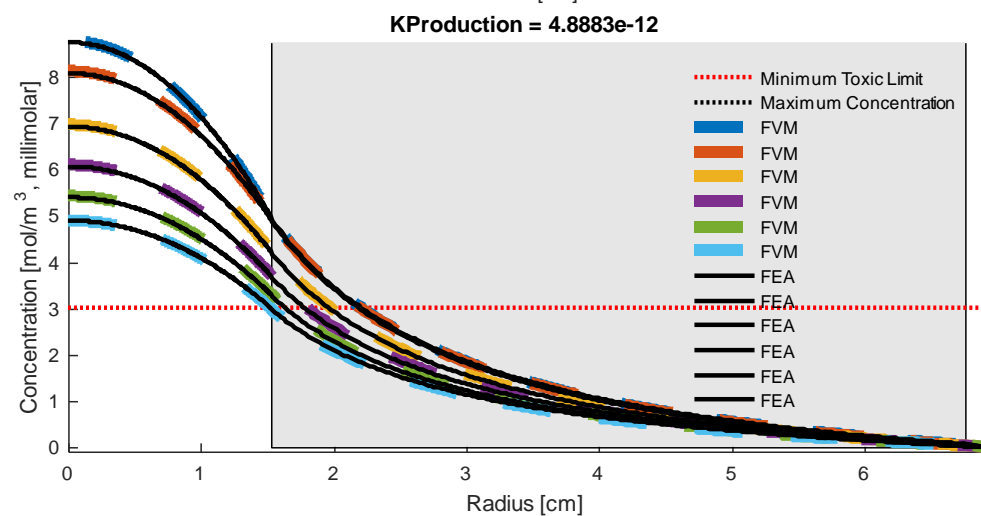
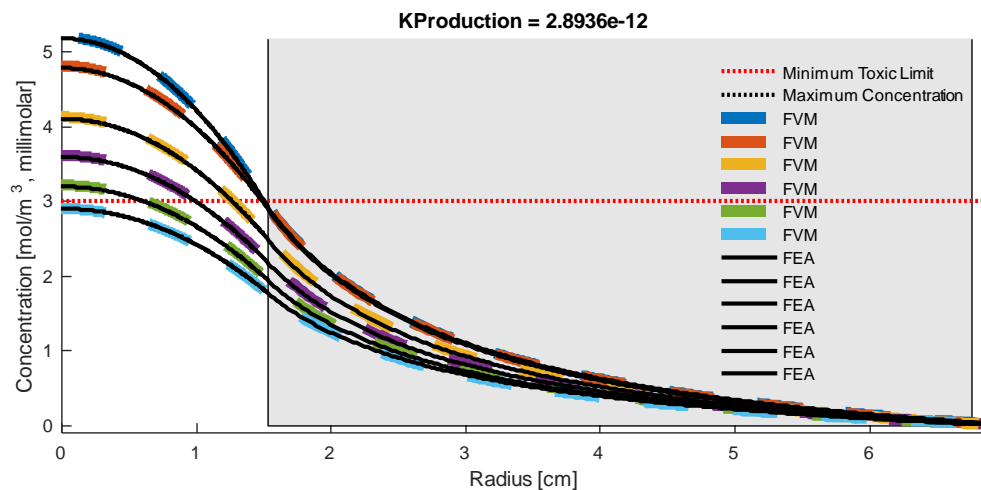


Figure 7.189. Comparison between the FEM and FVM on the spherical geometry system. Note, for comparison the FVM and FEA methods are expressed after reaching mesh independence, although they may not share the same number of values.

7.27.2.1 Finite Volume Method compared to Finite Element Analysis

The numerical integration of the finite volume method has been proposed previously. A finite element analysis method was also implemented for comparison. The comparison of the two methods can be seen in Figure 7.190.



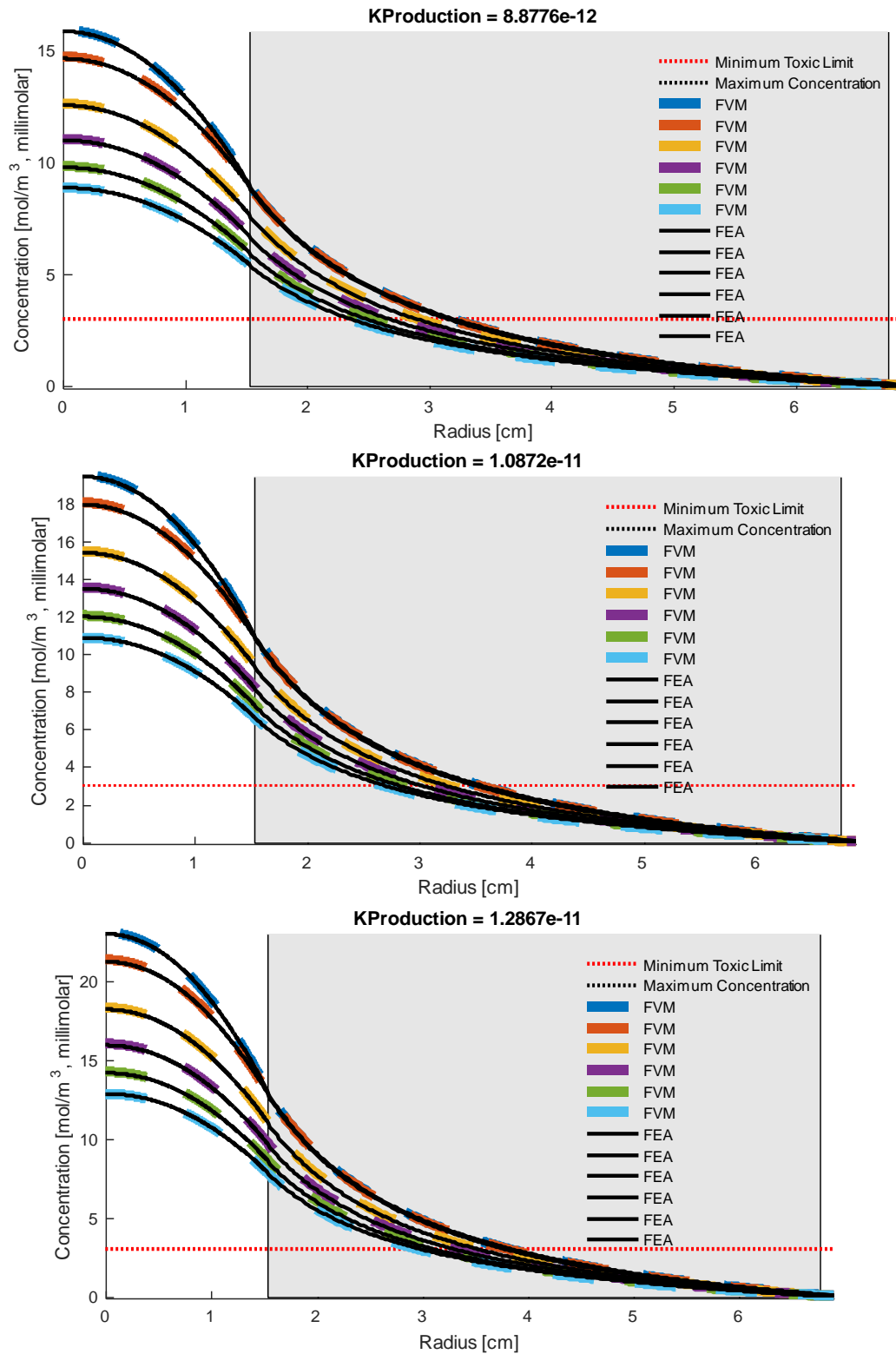


Figure 7.190. The comparison of the finite volume method with the finite element method. From this image it can be seen that the two methods are in good agreement with each other offering another form of validation.

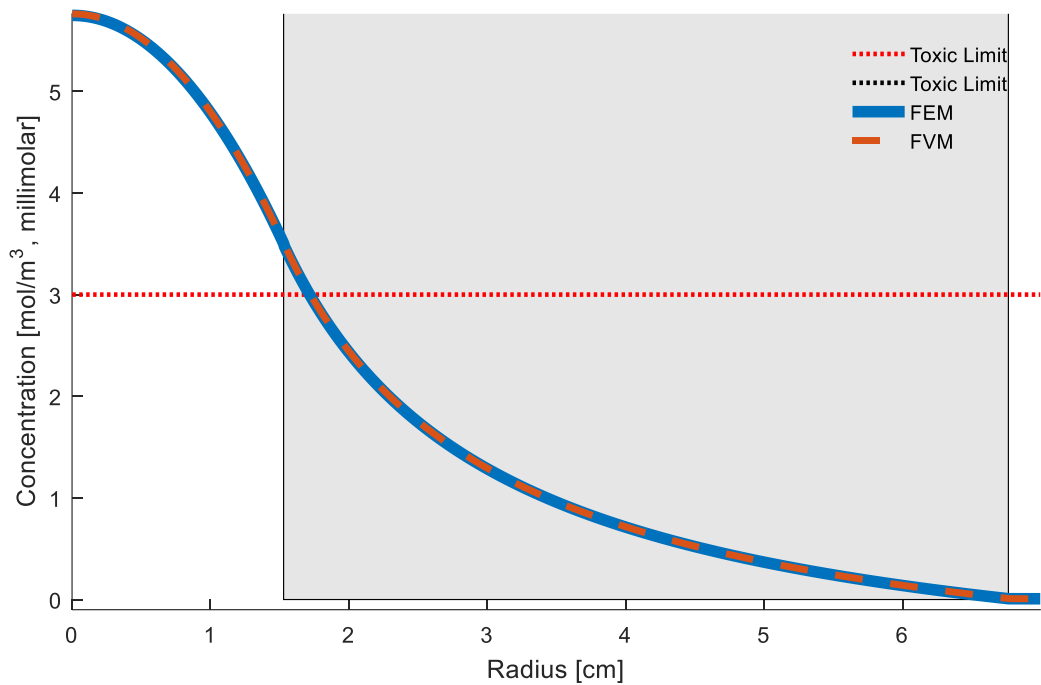


Figure 7.191. The comparison of the finite volume method with the finite element method where each has a sufficient mesh density to provide mesh independence. From this image it can be seen that the two methods are in good agreement with each other offering another form of validation.

7.27.3 Mesh Independence

The mesh error can be seen in Table 7.55 with the respective computation time for simulating all 6 profiles for FEM and FEA methods with the corresponding mesh densities. All computations were performed in Matlab. Concentration and error profiles along the radial direction are shown in Figure 7.192.

Table 7.55. Comparative error and computation time between different discretization methods

Number of Volume Elements (larger mesh – smaller mesh)	Infinity norm of the difference between values (FEA)	Infinity norm of the difference between values (FVM)	Computation time (in seconds)
21	$8.667 \cdot 10^{-6}$	$1.7965 \cdot 10^{-6}$	$9.06 \cdot 10^{-4}$
51	$2.9809 \cdot 10^{-6}$	$2.8108 \cdot 10^{-7}$	$8.33 \cdot 10^{-4}$
101	$1.6023 \cdot 10^{-6}$	$7.7789 \cdot 10^{-8}$	$2.25 \cdot 10^{-3}$
201	$8.3016 \cdot 10^{-7}$	$2.3118 \cdot 10^{-8}$	$2.54 \cdot 10^{-2}$
401	$5.0578 \cdot 10^{-7}$	$8.7651 \cdot 10^{-9}$	$5.59 \cdot 10^0$
1001	$1.7074 \cdot 10^{-7}$	$2.0955 \cdot 10^{-9}$	$4.82 \cdot 10^1$

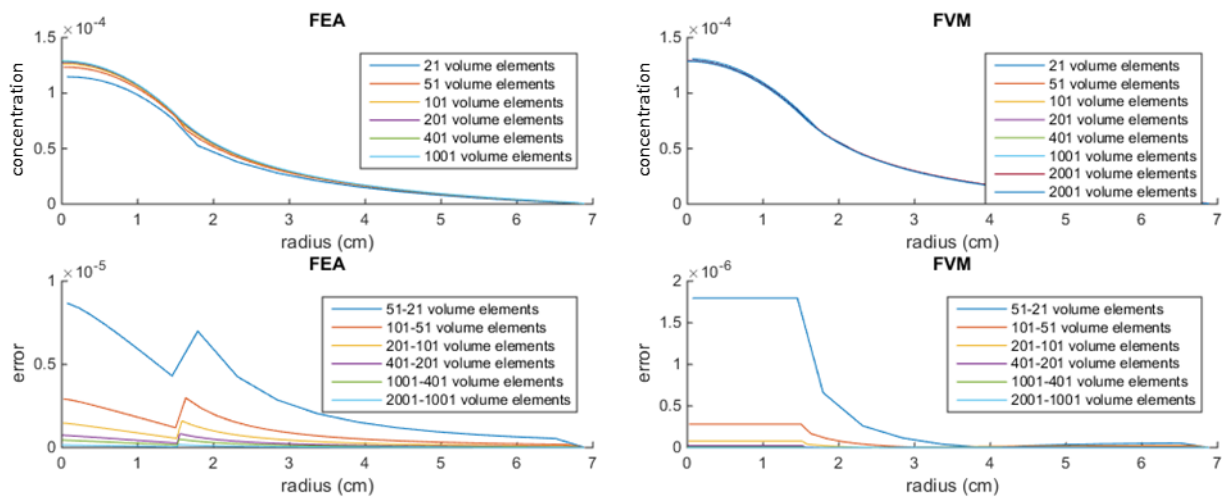


Figure 7.192. Comparative error as a function of mesh density.

Left) FEA and Right) FVM methods show similar profiles. Top) concentration as a function of radius at differing mesh densities show a faster convergence (lower mesh size) in the FVM method compared to the FEA method. Bottom) the FEA method shows an order of magnitude higher error than the FVM method at low discretization numbers. Note, the FEA method is more sensitive to discontinuities than the FVM method.

7.27.4 Material properties

The relevant material properties for the simulation of D₂HG in the spherical brain are offered in Table 9.56. The unit conversions for these values is offered in Table 7.59.

Table 7.56. Simulation parameters necessary for quantifying the concentration profile.

Parameter	Value	Symbol	Units
Diffusion constant (Tumor)	$0.5 \cdot 10^{-6}$ (simulation 2.1 & 2.3) $1 \cdot 10^{-6}$ (simulation 2.2 & 2.4)	D_t	cm^2/s
Diffusion constant (Brain)	$0.5 \cdot 10^{-6}$ (simulation 2.1 & 2.3) $1 \cdot 10^{-6}$ (simulation 2.2 & 2.4)	D_b	cm^2/s
Diffusion constant (CSF)	$30.2 \cdot 10^{-6}$ (simulation 2.1 & 2.3) $50 \cdot 10^{-6}$ (simulation 2.2 & 2.4)	D_{CSF}	cm^2/s
Production Rate	$S_A = 38.6 \cdot 10^{-12}$ (Inferred from in 2.1) (Simulation 2.1 & 2.2) $S_A = 694 \cdot 10^{-12}$ (Simulation 2.3 & 2.4)	S_A	$\text{mol}/\text{cm}^3/\text{s}$
Tumor Outer Radius	1.5299	R_t	cm
Brain Outer Radius	6.7700	R_b	cm
CSF Outer Radius	7.0200	R_{CSF}	cm
D ₂ HG concentration in the tumor	11.6-15.6 (Simulation 2.1) 10-30 (given)	$C_{A,t}$	mol/m^3
D ₂ HG concentration in the tissue (predicted)	1.302-10.6 (Simulation 2.1) (Grey area in figures)	$C_{A,b}$	mol/m^3
D ₂ HG concentration in the CSF	1.299 (Simulation 2.1) 0.1 (given)	$C_{A,\text{CSF}}$	mol/m^3
Neuronal toxicity level of D ₂ HG	3 (given)	$C_{A,\text{toxic}}$	mol/m^3
CSF Turnover Rate	0.5 L/day (Given)	q_{CSF}	L/day or ml/min

Table 7.57. Data conversion from original units to standardized units and simulated units. Red indicates a second value for minimum value.

Parameter	Sym	Given Value and Units		Converted Values and units		Actually Simulated and units	
Diffusion constant (Tumor)	D_t	$1.0 \cdot 10^{-6}$ $5.0 \cdot 10^{-7}$	cm^2/s	$1.0 \cdot 10^{-6}$ $5.0 \cdot 10^{-7}$	cm^2/s	$1.0 \cdot 10^{-6}$ $5.0 \cdot 10^{-7}$	cm^2/s
Diffusion constant (Brain)	D_b	1.0×10^{-6} $5.0 \cdot 10^{-7}$	cm^2/s	1.0×10^{-6} $5.0 \cdot 10^{-7}$	cm^2/s	1.0×10^{-6} $5.0 \cdot 10^{-7}$	cm^2/s
Diffusion constant (CSF)	D_{CSF}	$7.5 \cdot 10^{-6}$ $3.02 \cdot 10^{-5}$	cm^2/s	$7.5 \cdot 10^{-6}$ $3.02 \cdot 10^{-5}$	cm^2/s	$7.5 \cdot 10^{-6}$ $3.02 \cdot 10^{-5}$	cm^2/s
Production Rate (see separate calculation)	S_A	$6.94 \cdot 10^{-4}$	$\text{mol}/\text{m}^3/\text{s}$	$6.94 \cdot 10^{-10}$ (Calcualted) $2.89 \cdot 10^{-12}$ 38.57 $\cdot 10^{-12}$	$\text{mol}/\text{cm}^3/\text{s}$	$6.94 \cdot 10^{-10}$ (Calcualted) $2.89 \cdot 10^{-12}$ 38.57 $\cdot 10^{-12}$	$\text{mol}/\text{cm}^3/\text{s}$
Tumor Outer Radius	R_t	1.5299	cm	1.5299	cm	1.5299	cm
Brain Outer Radius	R_b	6.77	cm	6.77	cm	6.77	cm
CSF Outer Radius	R_{CSF}	7.02	cm	7.02	cm	7.02	cm
D ₂ HG concentration in the tumor	$C_{A,t}$	10-30	mol/m^3	$10 \cdot 10^{-6}$ - $30 \cdot 10^{-6}$	mol/cm^3	$10 \cdot 10^{-6}$ - $30 \cdot 10^{-6}$	mol/cm^3
D ₂ HG concentration in the tissue (guess we will predict it)	$C_{A,b}$		mol/m^3		mol/m^3		mol/m^3
D ₂ HG concentration in the CSF	$C_{A,CSF}$	0.0075-0.1	mol/m^3	$7.5 \cdot 10^{-9}$ - $100 \cdot 10^{-9}$	mol/cm^3	$7.5 \cdot 10^{-9}$ - $100 \cdot 10^{-9}$	mol/cm^3
Neuronal toxicity level of D ₂ HG	$C_{A,toxic}$	3	mol/m^3	$3 \cdot 10^{-6}$	mol/cm^3	$3 \cdot 10^{-6}$	mol/cm^3
CSF Turnover Rate	q_{CSF}	0.5	L/day	$5.787 \cdot 10^{-3}$	cm^3/s	$5.787 \cdot 10^{-3}$	cm^3/s

7.27.5 The validation of flux balance

The spot checks follow from nodal conservation laws and total conservation of the system are investigated below. The system used for interrogation has 10 nodes for the tumor, 10 nodes for the brain and 1 node for the CSF. Spot checks were performed at nodes 3, 6, 10, 15, and node 18.

$5 \cdot 10^{-7} \frac{6.8271 \cdot 10^{-5} - 6.753 \cdot 10^{-5}}{1.53} (1.9118) -$ $5 \cdot 10^{-7} \frac{6.753 \cdot 10^{-5} - 6.623 \cdot 10^{-5}}{1.53} (3.6766) + 3.84 \cdot 10^{-11} (0.285)$ $= -1.777 \cdot 10^{-26}$	At node 3
$5 \cdot 10^{-7} \frac{6.435 \cdot 10^{-5} - 6.188 \cdot 10^{-5}}{1.53} (8.98) -$ $5 \cdot 10^{-7} \frac{6.188 \cdot 10^{-5} - 5.883 \cdot 10^{-5}}{1.53} (15.5) + 3.84 \cdot 10^{-11} (1.365)$ $= 5.69 \cdot 10^{-25}$	At node 6
$5 \cdot 10^{-7} \frac{5.092 \cdot 10^{-5} - 4.607 \cdot 10^{-5}}{1.53} (26.62) -$ $5 \cdot 10^{-7} \frac{4.607 \cdot 10^{-5} - 3.136 \cdot 10^{-5}}{0.542} (41.21) + 3.84 \cdot 10^{-11} (4.06)$ $= -5.43 \cdot 10^{-25}$	At node 10
$5 \cdot 10^{-7} \frac{1.23 \cdot 10^{-5} - 9.127 \cdot 10^{-6}}{0.542} (190.817) -$	At node 15

$5 \cdot 10^{-7} \frac{9.127 \cdot 10^{-6} - 6.656 \cdot 10^{-6}}{0.542} (245.471) = -1.76 \cdot 10^{-24}$	
$5 \cdot 10^{-7} \frac{4.681 \cdot 10^{-6} - 3.067 \cdot 10^{-6}}{0.542} (375.483) -$ $5 \cdot 10^{-7} \frac{3.067 \cdot 10^{-6} - 1.722 \cdot 10^{-6}}{0.542} (450.84) = -1.76 \cdot 10^{-24}$	At node 18
$14.995 * 3.84 \cdot 10^{-11} - 5.79 \cdot 10^{-3} 9.998 \cdot 10^{-8} = -1.034 \cdot 10^{-25}$	Total conservation

7.28 Appendix AB: Oxygen in the brain

This section summarizes and compares many ways to calculate oxygen tension in the brain. In all cases, there is a model for the vascular network (frequently represented by a network of points and arcs with associated diameters) and a tissue mesh. The models are formulated analytically, expressed numerically (in integral form), and implemented in 1, 2 and 3 dimensions. Multiple models of mass transfer were compared and parametric studies were analyzed. Moreover, novel visualization techniques were developed (specifically, raytracing, plane cutouts, small cutouts, and DICOM visualization tools). Applications to models of the aging brain to interpret the changes in oxygen tension due to known vascular disruption mechanisms are proposed. An additional formulation of a functional hyperemia model (dynamic interaction between vasculature and surrounding tissue during neuronal activation) was proposed and implemented.

7.28.1 Introduction

Computing oxygen in the human body is not trivial, as oxygen relies heavily on the multiple binding sites of hemoglobin to assist in carrying the oxygen (O) to the extremities of the body and ensure that O is not depleted near the source (the heart and lungs). The four binding sites of hemoglobin (Hb) work cooperatively to increase the binding affinity inversely to the quantity of bound O. E.g, as O unbinds from Hb, it is harder for O to unbind. This can be also interpreted in chemical reaction rates as a dynamic change in the reaction rate constants as a function of bound substrate. These reaction rate constants are, however, very hard to measure and much speculation still exists regarding their nature.

One common assumption is to consider 4 reversible reactions of Hb to O which produce intermediate forms of Hb as a byproduct. This new Hb molecule has a unique set of reaction rates with O. Another common simplification is to use the Hill equation for saturation, as defined in a later section, which accounts for the nonlinear binding kinetics.

This document outlines how to compute the supply and distribution of oxygen in a discretized vascular network interacting with the surrounding tissue. One of these methods (the point-centered coupling mass transfer model) has been previously published by Gould [21,22,62]. This section will also cover some background regarding the hill equation and alternative computational methods.

A short physiological description of how the oxygen moves through the body is offered: oxygen exchanges with hydrogen in the lungs (oxygen is in the lung and hydrogen is on the hemoglobin to start). Once the hydrogen has entered the lung, it reforms into CO₂ and H₂O immediately. Oxygen replaces the hydrogen on hemoglobin, which carries it throughout the body with high binding affinity so as to not deplete the supply immediately. The nonlinear binding kinetics allow hemoglobin to carry oxygen far distances into the body, instead of expending it all in the first few cm of the blood stream.

Once attached to the red blood cells (RBCs), the oxygen can dissociate from the hemoglobin as a function of free oxygen (in the cytoplasm) and the binding affinity (which itself is a function of bound O). The oxygen, once freed from the hemoglobin, diffuses through the cytoplasm towards the cell wall, which hinders diffusion and transitions across the layer using mass transfer. Once outside the cell, O diffuses through the blood plasma towards the vascular wall where it can again cross through mass transfer. The oxygen can then diffuse through the extravascular space and react away through metabolism with glucose to produce carbon dioxide and water. This water and

carbon dioxide mixes with free molecules in a reversible reaction sequence known as the bicarbonate reaction sequence until it reaches the hemoglobin (following the reverse path as O). At this point, free hydrogen will competitively bind to the hemoglobin and assist in lowering the binding affinity for oxygen. Metabolism also occurs in the cytoplasm and in the blood brain barrier (BBB), however this is not modeled in any of the current implementation. For an in-depth review of current models of oxygen in the cerebral microcirculation, refer to Section 7.37.

7.28.2 Problem formulation

Oxygen is carried through the blood stream into the brain by red blood cells (RBCs). This can be modeled by either (i) a single-phasic bulk flow with convection or (ii) with hemoglobin-bound oxygen, requiring a biphasic blood flow model. The simple model benefits from a linear implementation with stable convergence up to very large mesh sizes while the latter benefits from an established nonlinear relationship between the red blood cells and bifurcation geometry.

The single-phasic blood flow model employs mass conservation and the Hagen Poiseuille relationship between pressure and flow. The oxygen (O) will be convected by the bulk blood flow. Each vessel in the blood stream will be allowed to discharge oxygen into the surrounding tissue through mass transfer (more details on mass transfer models are covered in Section 7.28.7.3). O then diffuses through the tissue and reacts (metabolizes) throughout the domain uniformly.

In the second model, the oxygen first must discharge from the hemoglobin, diffuse through the red blood cell, transfer across the lipid bilayer wall, diffuse through the plasma, and transfer across the endothelial cell layer of the blood vessel all before it enters the brain tissue (extravascular space, specifically the interstitial fluid). In an initial approximation, the diffusion through the cellular cytoplasm, across the lipid bilayer of the cell, and diffusion through the plasma are

assumed instantaneous. In the tissue, oxygen diffuses through the space and is reacted away as in the previous model. An overview of this process and the relevant processes in functional hyperemia is summarized in Figure 7.193.

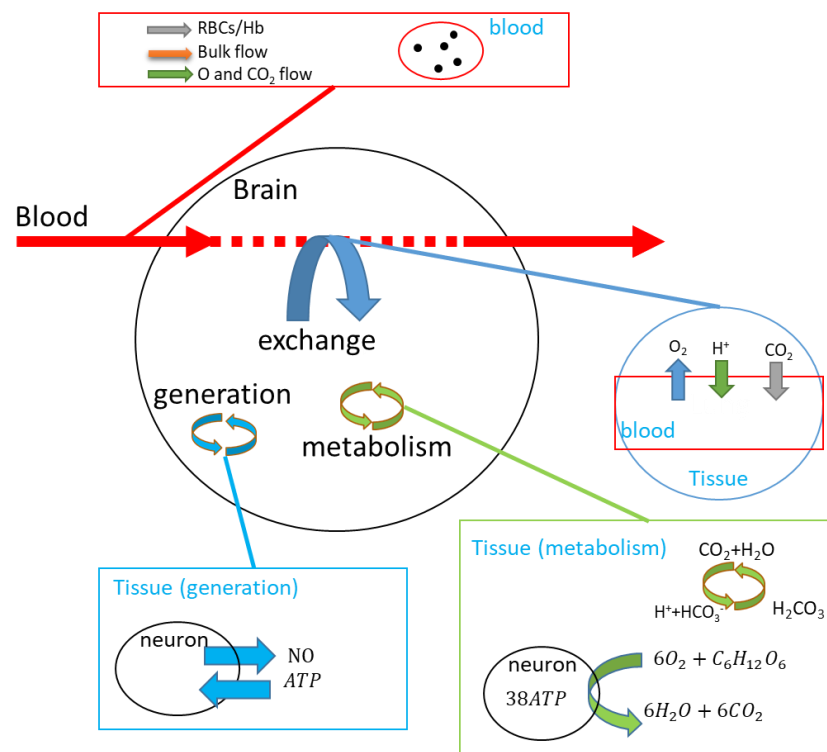


Figure 7.193. Overview of relevant transport phenomena related to the lifespan of oxygen in the brain.

7.28.3 Boundary conditions and material properties

The exterior tissue domain enforces a no flux boundary condition for oxygen. The inlet oxygen content for the vasculature is a fixed value of arterial partial pressure (P_{O_a}). The blood flow model

uses arterial inlet and venous outlet fixed pressures (P_a and P_v , respectively). In the case of biphasic blood flow, inlet hematocrit to the network was fixed at 0.35 (systemic hematocrit).

In the case of functional hyperemia (Section 7.30), the initial state of NO is considered baseline ($0 \text{ mol}/\mu\text{m}^3$) and all computations of NO are considered a deviation from the baseline. The values of the boundary conditions and material properties (diffusivity, consumption rate, etc.) used in this study are listed in Table 7.58.

The hemoglobin inlet is calculated from the number of RBCs ($h \cdot \text{vol}/\text{vol}_{\text{rbc}}$) multiplied by the number of hemoglobin per RBC (need this number) multiplied by 4. This value is used to compute the total number of hemoglobin. Note, the oxygen BC should be calculated following:

$$n_O = 4n_{Hb}^{/RBC} S \frac{hV_{vessel}}{V_{RBC}} \quad (7.525)$$

Where $n_{Hb}^{/RBC}$ is number of hemoglobin per RBC, S is oxygen saturation (calculated from Hill or taken from literature), h is the hematocrit inlet value, V_{vessel} is the volume of the inlet vessel, and V_{RBC} is the volume of a single RBC. The calculation of saturation would involve assuming an inlet partial pressure value of 68.5 mmHg which is converted to a concentration using Equation (7.636). The derivation of the concentration conversion from partial pressure is given in Section 7.34.2. The saturation is then evaluated from the Hill equation to identify how much is bound and unbound in the inlet segment. The Hill equation is discussed in more detail in Section 7.36.

Table 7.58. Parameter and boundary condition choices for this model

Description	Symbol	Value	Units	Ref
Blood pressure inlet	P_a	120	mmHg	[21,22,258]
Blood pressure outlet	P_v	5	mmHg	[21,22,258]
Oxygen partial pressure at network inlet	P_{O_a}	65.4	mmHg	[21,259]
Diffusivity of NO	D_{NO}	3.3e3	$\mu m^2/s$	[260]
Diffusivity of O ₂	D_{O_2}	1.8e3	$\mu m^2/s$	[261]
NO production rate	$K_{NO^{++}}$	$2.1e-17$	mol/s	[262]
		21	attomol/s	
NO consumption rate (reaction to nitrate and nitrite) same for blood and tissue	K_{NO^-}	0.01	s^{-1}	[263]
CMRO2	$K_{O_2^-}$	$41.7e-13$	$nmol/(um^3s)$	[264]
		$41.7e-4$	attomol/(um^3s)	
CMRO2 increase during firing	$K_{O_2^{--}}$	30	%	[265]
Mass transfer across the endothelial layer (blood-brain-barrier) for O ₂	U_{O_2}	$2.4 \cdot 10^{-5}$	cm^2/s	[263]
Solubility of O ₂ in plasma	$H_{O_2,p}$	2.4e3	$\mu m^2/s$	
		$3e-5$	mlO ₂ /ml/torr	[266]
		$1.27e-12$	nmol/ $um^3/mmHg$	[264]
		$1.27e-3$	attomole/ $um^3/mmHg$	
Ratio of O ₂ solubility in brain vs plasma		0.8		[261,266]
Solubility of O ₂ in brain	$H_{O_2,t}$	$2.6e-5$	mlO ₂ /ml/mmHg	[261]
		$1.1007e-12$	Nmol/ $um^3/mmHg$	[261,266]
NO boundary condition domain edge			$\nabla f = 0$	
O ₂ boundary condition domain edge			$\nabla f = 0$	

7.28.4 Numerical Implementation

All systems require a sparse linear algebraic solving library. The libraries of PETSc linear algebraic solvers is employed using the generalized minimal residual method, GMRES, and a block Jacobi preconditioner. For more information on how the nonlinear biphasic blood flow problem can be separated and iteratively converged using linear systems, refer to Section 4.6.

7.28.4.1 Solving a 3D block using successive 2D sweeps for initialization routine

One method of solving large blocks of 3D Cartesian meshes is to iteratively solve 2D boundary value problems, where the boundaries on the upper and lower limits of the plane need to be refined. This method substantially reduces solving time (2D planes vs 3D cubes). The results, however, show that the excessive number of iterations required to converge the system is in excess of the original 3D problem.

Table 7.59. Time lapse for solving a 3D problem with successive 2D sweeps

nVol	10	25	50	100
Time sweep bidirectional (s)	0.17	35.8	740.1	
Time 30 sweeps+converge (s)	0.21	7.2	140.0	394.0
Time 20 sweeps+converge (s)	0.15	6.4	134.3	3717
Time 10 sweeps+converge (s)	0.10	5.16	139.5	
Time dyn sweeps+converge (s)	0.13	7.8	190.15	3687
MA48 ST	0.03	38.7		
Matlab simultaneous:				
To make equations	0.0036	0.053	0.412	3.34
To solve	0.0072	0.27	10.3	633.7
Total Matlab ST	0.0108	0.323	10.7	637
PETSc GMRES	0.48	1.6	10.5	109.73
GMRES (sweeping to initialize)				(50 sweeps) 3385

7.28.5 1D problem

In order to ensure the numerical solution to the full 3D coupled problem of oxygen is reliable, it is important to investigate lower order simulations using different material properties, boundary conditions, and models for mass transfer or reactions. These simplified models give straightforward insight into the effect each modeling choice has on the resulting profile.

7.28.5.1 1D diffusion only

The first place to start with any numerical simulation is to begin with only one aspect of the simulation and ensure it is validated and meaningful before proceeding to add more complications. In this case, a simple diffusion case study is the best place to start.

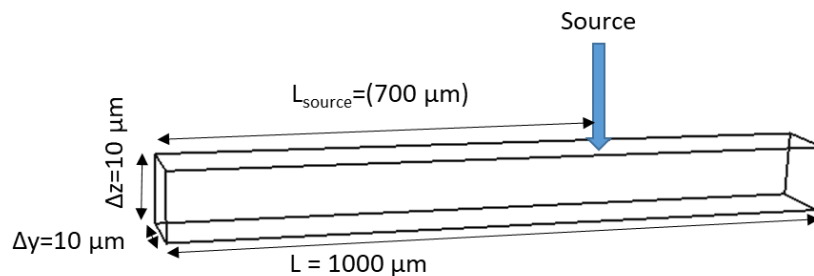


Figure 7.194. Schematic of domain on which the current 1D implementation is being solved.

Conclusions:

- The 1D diffusion only problem with Dirichlet boundary conditions and a fixed flux source term has shown early convergence when the grid cells align perfectly with the source flux (one cell center is exactly at the source point).
- When aligning the mesh with the source term, the lowest residual is the smallest number of elements solving the problem (7 elements), which gave the same peak concentration as calculated analytically.
- When the grid is arbitrarily chosen (10,1000,500, etc.), the convergence takes longer, because the profile changes every time the peak location moves (due to misalignment in the grid spacing).

The best way to validate a numerical simulation is to begin with an analytic solution. The analytic concentration at source point in a 1D diffusion problem can be obtained.

$$q_{left}(700\mu m) + q_{right}(700\mu m) = q_{source}$$

$$D\Delta y\Delta z \left(\frac{c(700\mu m) - c(0\mu m)}{700} + \frac{c(700\mu m) - c(1000\mu m)}{300} \right) = q_{source} \quad (7.526)$$

And given that $c(0)$ and $c(1000)$ is known, this reduces to:

$$\frac{c(700\mu m)}{700} - \frac{c(0\mu m)}{700} + \frac{c(700\mu m)}{300} - \frac{c(1000\mu m)}{300} = \frac{q_{source}}{D\Delta y\Delta z}$$

$$\frac{c(700)}{700} + \frac{c(700)}{300} = \frac{q_{source}}{D\Delta y\Delta z} + \frac{c(1000)}{300} + \frac{c(0)}{700} \quad (7.527)$$

And if $c(1000)=c(0)$, then this further reduces:

$$\frac{c(700\mu m) \cdot 300 + c(700\mu m) \cdot 700}{700 \cdot 300} = \frac{q_{source}}{D\Delta y\Delta z} + \frac{c(0\mu m) \cdot (700\mu m) + c(0\mu m) \cdot 300}{300 \cdot 700} \quad (7.528)$$

$$\frac{c(700\mu m) \cdot 1000}{700 \cdot 300} = \frac{q_{source}}{D\Delta y\Delta z} + \frac{c(0\mu m) \cdot 1000}{300 \cdot 700} \quad (7.529)$$

$$c(700\mu m) \cdot 1000 = \frac{q_{source}}{D\Delta y\Delta z} 700 \cdot 300 + c(0\mu m) \cdot 1000 \quad (7.530)$$

$$c(700\mu m) = \frac{q_{source}}{D\Delta y\Delta z} \frac{700 \cdot 300}{1000} + c(0\mu m) \quad (7.531)$$

If we consider the concentration of 30 mmHg, this correlates to $2.16e-9 \mu\text{mol}$ and a source feed (q_{source}) = $1e-9 \mu\text{mol/s}$, $D = 1.8e3 \mu\text{m}^2/\text{s}$, and $\Delta y\Delta z = 100$, $c(700)$ becomes:

$$c(700\mu m) = \frac{1e-9}{(1.8e3)100} \frac{700 \cdot 300}{1000} + 2.16e^{-9}$$

$$c(700\mu m) = 2.1612e^{-9} \mu\text{mol}/\mu\text{m}^3 \quad (7.532)$$

$$c(700\mu m) = 30.0163 \text{ mmHg}$$

Note, all computations must be computed in line, the truncation used in this example will lead to significant roundoff error.

Discretization schema. There are multiple ways to discretize a domain in even 1-dimension. Two methods to consider are the half-distance spacing and the half-volume technique. The review below will show that the most logical choice is the half-distance spacing as reviewed in Figure 7.195. Conclusions:

- The analytical solution can be obtained for a pure diffusion system with flow source using minimal points necessary to define all domain features (source, boundary, etc.)
- When the source point does not shift the location of the source, all meshes give the correct solution (complete mesh independence)

The half-distance spacing is convenient for equation generation, as all interior equations have the same volume and thus, the same coefficients. The only alterations to the base equation in the half-distance spacing is in volumes sharing an edge with the boundary, where a half-distance is used to calculate the flux between the interior volume and the exterior half-volume. The half-distance spacing has the drawback that the exterior volume has no volume and thus cannot exhibit any reactions, although these equations are replaced with boundary conditions (Dirichlet, Neumann, etc.) that frequently do not require assignment of reactions.

The half-volume technique is computationally convenient, as the volume centers are easily computed as an even distribution between the 0 and L with nVolumes of divisions. This method has the drawback of making equation generation difficult, however, as it requires assignment of a boundary condition over a volume that cannot simultaneously account for the reaction in that volume. This also suffers from the drawback that changing the domain resolution will inherently change the resulting distribution, as the reaction volume will change (the half-volumes that are void of reactions will get larger or smaller).

Source term is ensured to fall on the same location. The best way to calculate a reliable solution is to ensure that all relevant characteristics of the system are captured accurately. For instance, if there is a source term at a specific location, the best integration of that information into the simulation is to ensure there is always a cell centered at this location, regardless of mesh density. In this way, the mesh always accounts for the proper source location (instead of the source location changing as a function of grid spacing). Such a series of simulations was used to validate the simulation in reference to the analytic solution provided above.

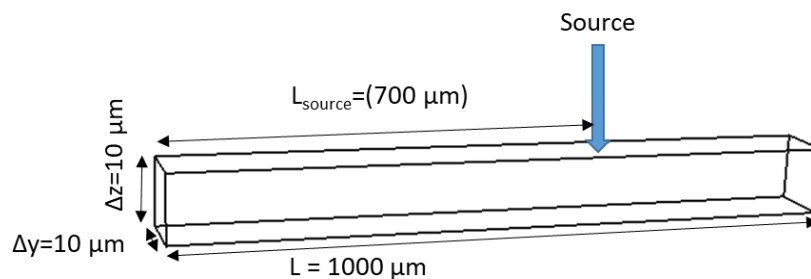


Figure 7.195. Graphical representation of two discretization techniques in 1-dimension. The two methods differ in how the volume center is located and how the boundary volumes are accounted for. The best choice is the half distance spacing technique.

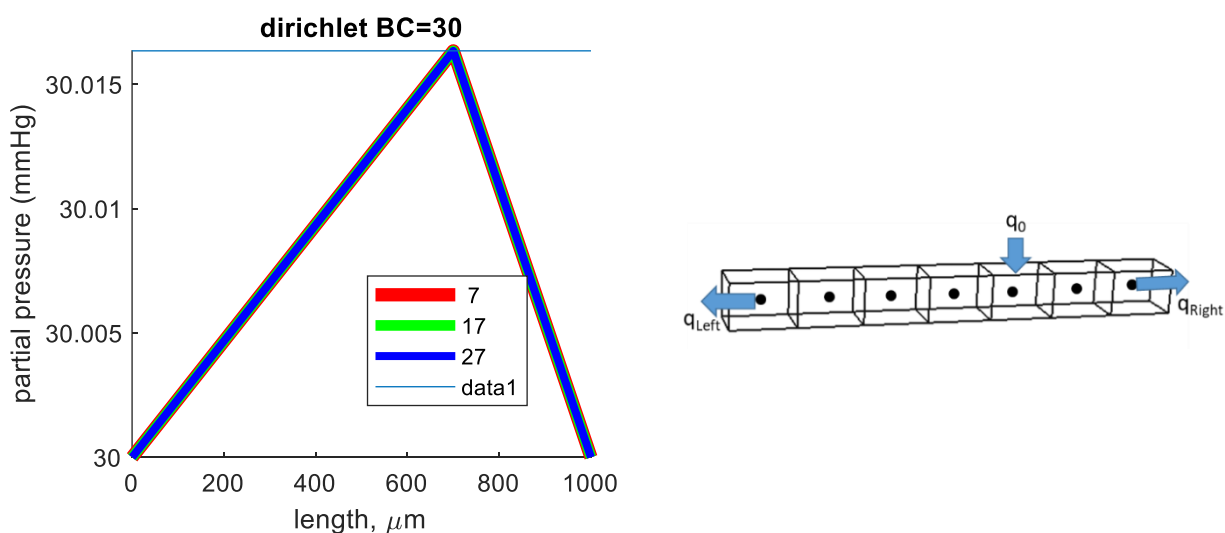


Figure 7.196. Comparison of different mesh sizes with the analytic solution of the peak concentration (computed above).

Left) The distribution across 3 datasets, all of which have a peak near the analytic solution (plotted as data1). Right) the residual as computed between peak pressure analytically calculated to the pressure computed on the distribution. The first dataset (n=7) has an error of 0, so the log scale omits this datapoint.

nVol	qLeft (attomole/s)	qRight (attomole/s)	q0 (attomole/s)	C(700) (mmHg)	Residual (attomole/s)
7	300	700	1000	30.0163	1.819e-10
17	300	700	1000	30.0163	1.819e-10
27	300	700	1000	30.0163	2.274e-9

Source term is not ensured to fall on the same location. One source of numerical error occurs when the grid spacing is not chosen with special consideration of the system. If an arbitrary nodal position is chosen, the solution can vary simply because the features of the simulation do not perfectly align with the cell centers in the mesh, leading to mesh-dependence. The best way to counter this problem is to choose the nodal positions of the simulation to guarantee that all features occur at the center of their respective volumes at all discretization densities.

Conclusion:

- The analytical solution can be obtained for a pure diffusion system with flow source
- When the source point does not shift the location of the source, any mesh gives the correct solution (complete mesh independence)

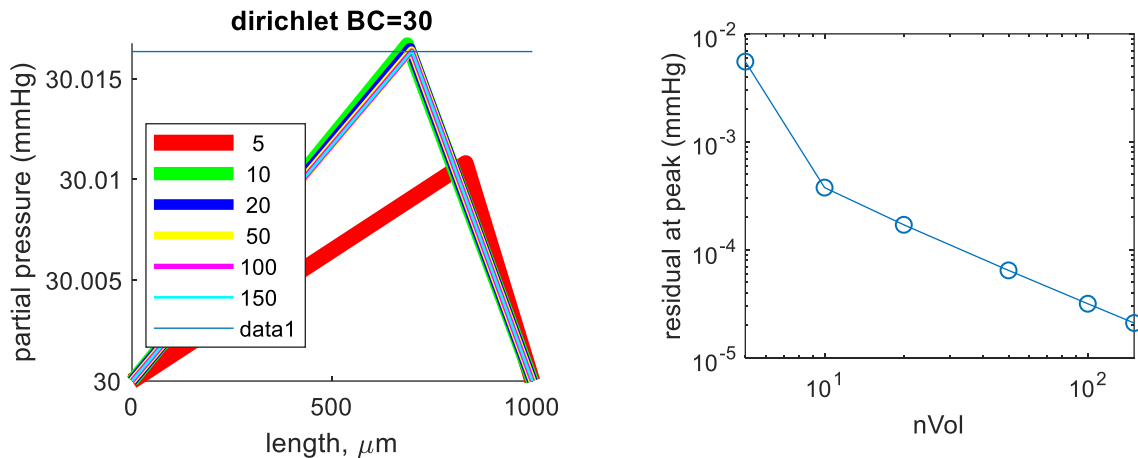


Figure 7.197. Comparison of different mesh sizes with the analytic solution of the peak concentration (computed above).

Left) The distribution changes as a function of grid spacing, primarily because the source location changes causing the gradients and fluxes to change.

nVol	qLeft (attomole/s)	qRight (attomole/s)	Q0 (attomole/s)	C(700) (mmHg)	Residual (attomole/s)
5	166.7	833.3	1000	30.0108	8.0e-10
10	312.5	687.5	1000	30.0167	1.5e-10
20	305.6	594.4	1000	30.0165	7.1e-9
50	302.1	697.9	1000	30.0164	2.2e-8
100	301.0	699.0	1000	30.0164	4.4e-8
150	300.7	699.3	1000	30.0164	3.9e-8

nVol	qLeft (μ mole/s)	qRight (μ mole/s)	Q0 (μ mole/s)	C(700) (mmHg)	Residual (μ mole/s)
5	0.1667e-9	0.8333e-9	1e-9	30.0108	1.3e-21
10	0.3125e-9	0.6875e-9	1e-9	30.0167	2.1e-22
20	0.3056e-9	0.5944e-9	1e-9	30.0165	1.2e-20
50	0.3021e-9	0.6979e-9	1e-9	30.0164	1.3e-20
100	0.3010e-9	0.6990e-9	1e-9	30.0164	1.1e-20
150	0.3007e-9	0.6993e-9	1e-9	30.0164	4.1e-20

Source term not fall on cell centers) with edge detection. This case study follows the previous one, yet the domain is separated by the use of edge-detection. This occurs when the discretization is so dense that there are interior vascular nodes within the system. When this occurs in 1-dimension, the Dirichlet value assigned at the vascular-tissue interface works as a discontinuous diffusion problem in 1D between two Dirichlet points as opposed to the original interior-boundary problem.

Conclusion:

- If the domain in 1D is split with mass transfer occurring only over the vessel surface (and the volumes inside the vascular domain are no longer connected), this changes the domain shape entirely and this becomes 2 independent diffusion problems.
- This new problem is much easier to solve, but is only an artifact of the 1D domain and does not occur in higher dimensions

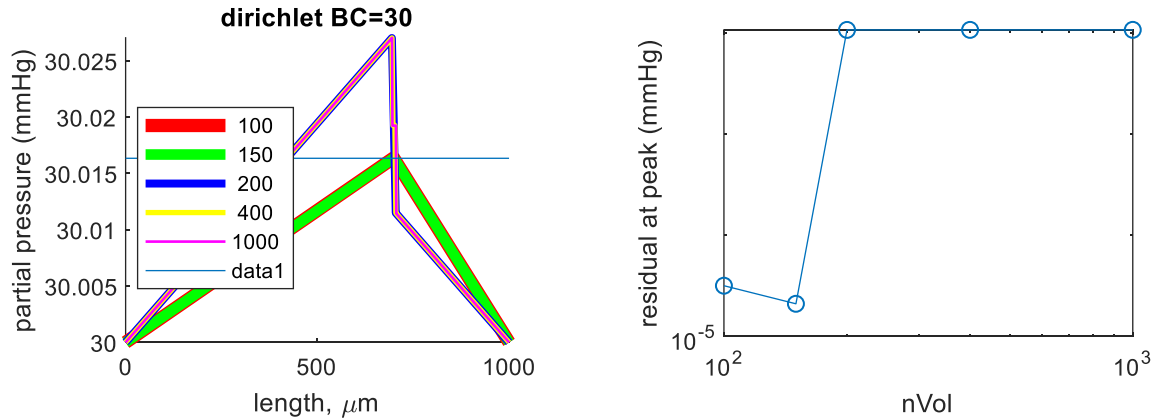


Figure 7.198. Comparison of different mesh sizes.

Left) The distribution changes as a function of grid spacing, primarily because the source location changes causing the gradients and fluxes to change. Right) The residual immediately converges to final result once the edge detection splits the domain. The error at this time is caused by the artificial boundary condition at the vascular center.

nVol	qLeft (μmole/s)	qRight (μmole/s)	Q0 (μmole/s)	C(700) (mmHg)	Residual (μmole/s)
100	0.30e-9	0.699e-9	1e-9	30.0164	1.1e-20
150	0.30e-9	0.699e-9	1e-9	30.0164	4.1e-20
200	5e-10	0.5e-9	1e-9	-	1.5e-20
400	5e-10	0.5e-9	1e-9	-	1.9e-19
1000	5e-10	0.5e-9	1e-9	-	5.0e-19

** note, the results for the value at 700 microns is omitted, because it is a linear averaging

between left and right values – which does not correlate to the source

7.28.5.2 1D problem with reaction and mass transfer

One-dimensional simulations, by definition, have a smaller workspace than the 2D and 3D counterparts. An investigation of boundary condition choices and choices in mass transfer models was conducted in 1D. This simplifies equation generation, manipulation, and drastically reduces domain size (nElements). In all cases, the half distance spacing technique is used for discretizing the domain following discussion in 7.14. The following parameters were used for the predictions:

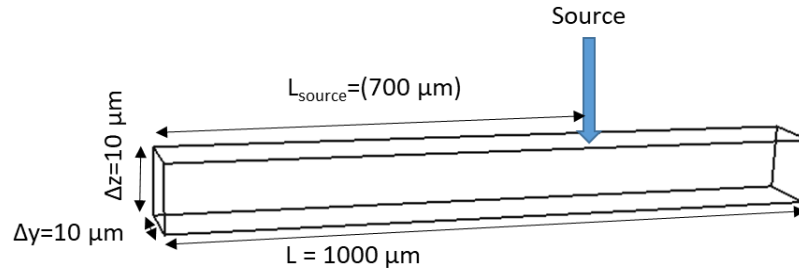


Figure 7.199. Overview of domain on which the current 1D implementation is being solved.

Parameter	Symbol	Value	Units
Diameter	dia	10	μm
Tissue Diffusivity	D	1.8×10^3	$\mu\text{m}^2/\text{s}$
Length of tissue ray	L	1	mm
Cross section	$\Delta y = \Delta z$	10	μm
Specific zeroth order reaction rate (CMRO ₂)	r ₀	0.01	$\mu\text{mol}/\text{mL}/\text{s}$
		0.01×10^{-3}	$\mu\text{mol}/\mu\text{L}/\text{s}$
First order reaction rate	k ₁	4.62×10^{-6}	1/s
Average tissue concentration	\bar{c}	30	mmHg
		2.163	mol/L
Mass Transfer Flux	Q _{MT}	2.163	$\mu\text{mol}/\mu\text{L}$
		1×10^{-9}	$\mu\text{mol}/\text{s}$

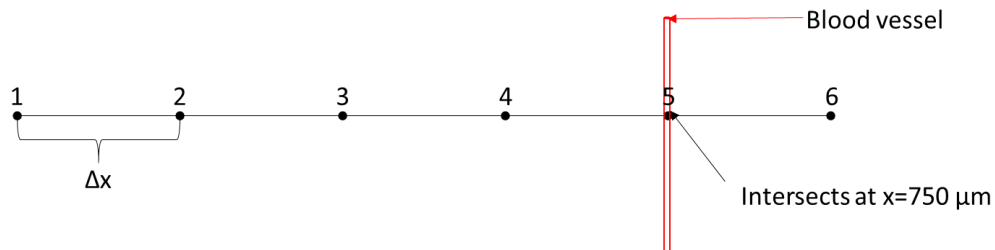


Figure 7.200. Schematic layout of the 1D simplified problem.

Varying boundary conditions. In any simulation, the varying of the boundary conditions should not greatly impact the concentration profile. An investigation of varying boundary conditions was performed and the results indicate boundary condition choices that led to non-physiological results.

Conclusions:

- only the periodic, no flux, and Dirichlet (using physiological values) BCs give meaningful results
- the no flux BC is only sensible in the event the domain is large enough to be considered isolated

The one-dimensional oxygen simulation was implemented with all the mass transfer (flow boundary condition) discharging into a single volume of the mesh. The boundary conditions were varied and the results compared. The boundary condition choices (applied at both ends of the 1D mesh) included: (i) Dirichlet value of 0, (ii) Dirichlet value of 30mmHg, (iii) no flux (symmetry), (iv) periodic. The reaction rate was also varied between a 0th order rate and a 1st order model. The 0th order reaction rate was only implemented on BC choices (i) and (ii). The results of these simulations are given in Figure 7.202.

The most reasonable boundary conditions are periodic and the most stable mass transfer model is the edge dispersion method. Note, when each volume of the mesh is larger than the diameter of the vessel, model 3 for mass transfer is simplified to model 1.

Conceptual:

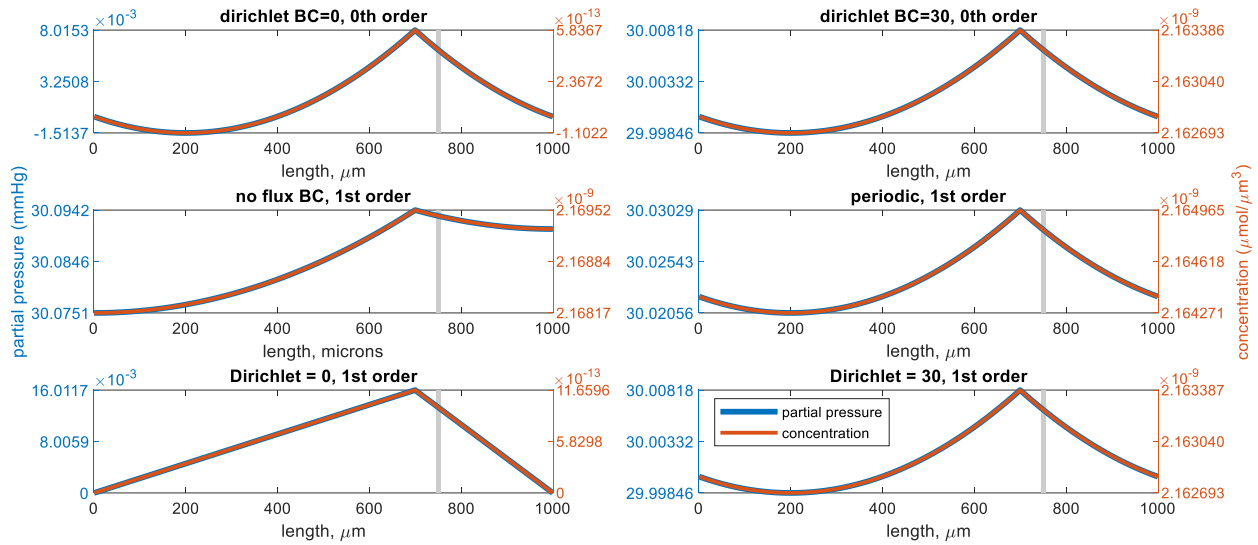


Figure 7.201. Profile shape comparison of different boundary conditions and reaction models for a 1D diffusion/reaction problem.

The vertical axis (partial pressure of oxygen, mmHg) is not standardized between the figures. The profiles show similar trends with the exception of the 1st order reaction model and Dirichlet BCs (bottom left).

Comparative:

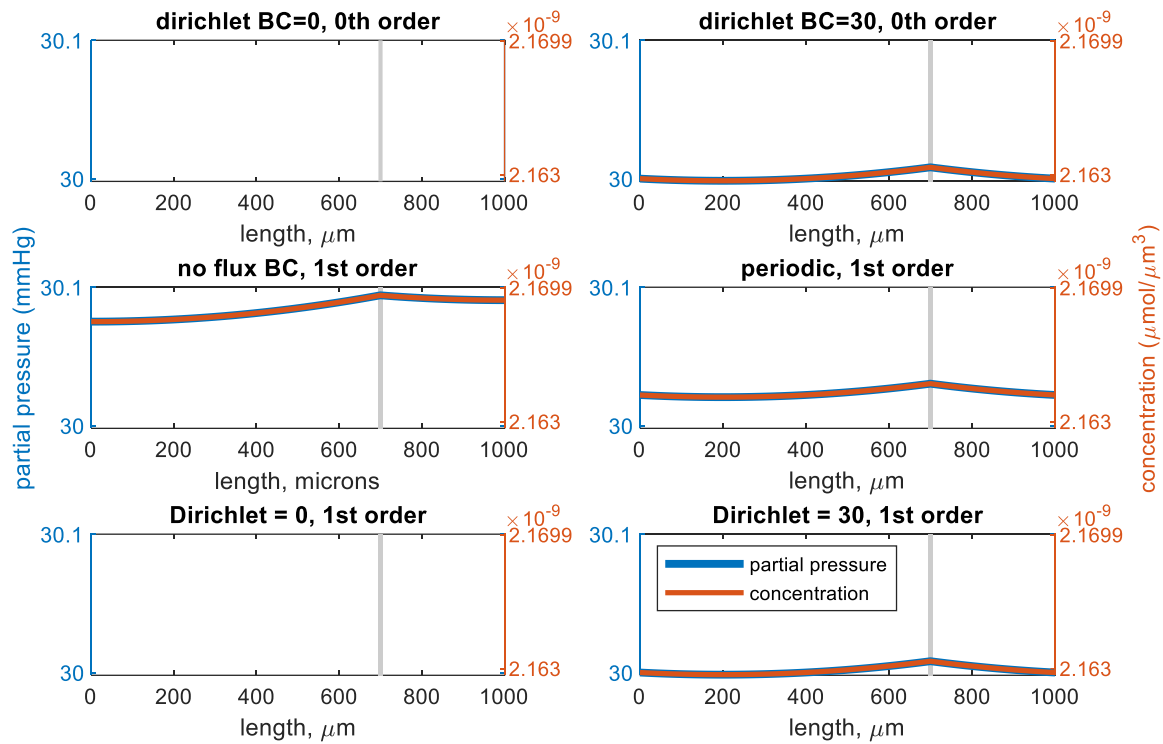


Figure 7.202. Comparison between different reaction rates and boundary conditions in the 1D oxygen simulation.

These results indicate the only reasonable choices are 1st order reaction with no flux, periodic, or Dirichlet (value of 30mmHg) are reasonable.

Varying mass transfer model. The models for mass transfer are described in Figure 7.203. The comparison of mesh density on partial pressure profile for differing boundary conditions is given in Figure 7.204. These results are represented in more detail in Figure 7.205.

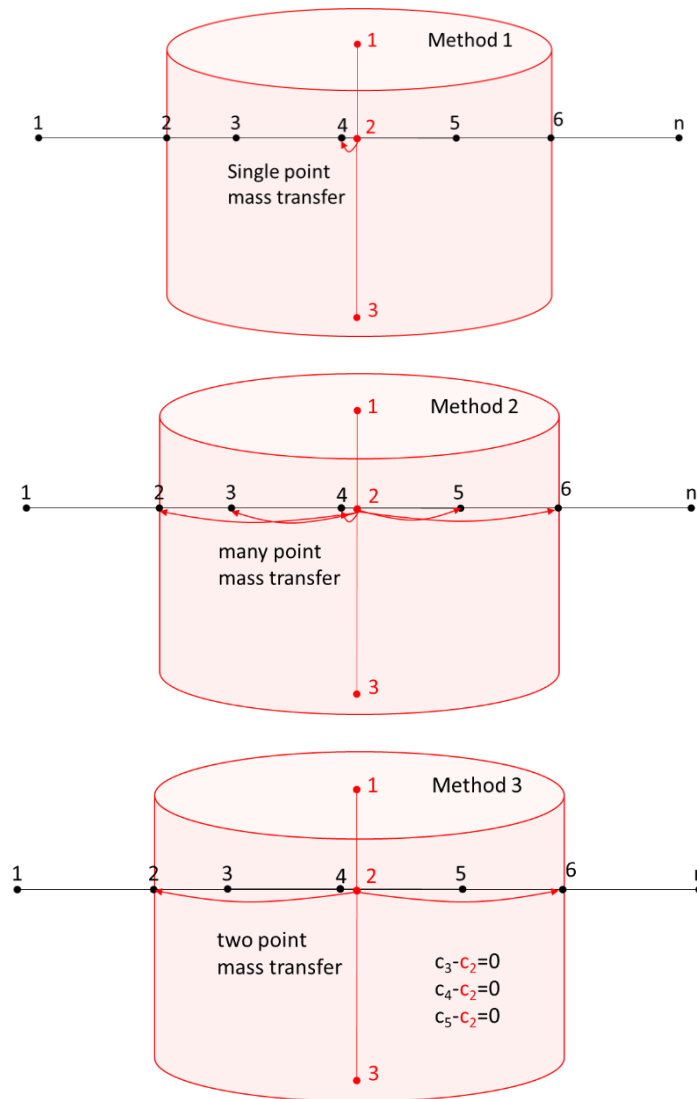


Figure 7.203. Three methods (models) for mass transfer between vasculature and tissue. Top) model 1 is the classic point source interface, Middle) model 2 is a distributed source (non-physiological, but added for completeness), and Bottom) model 3 is transfer only across the vessel edges. In model 3, the mesh elements that lie within the vascular segment will be endowed with boundary condition-like equations that force their concentration to be the same as that of the vasculature. Note, model 3 does separate the domain into 2 disconnected portions, a phenomenon that is not shared in higher dimensions.

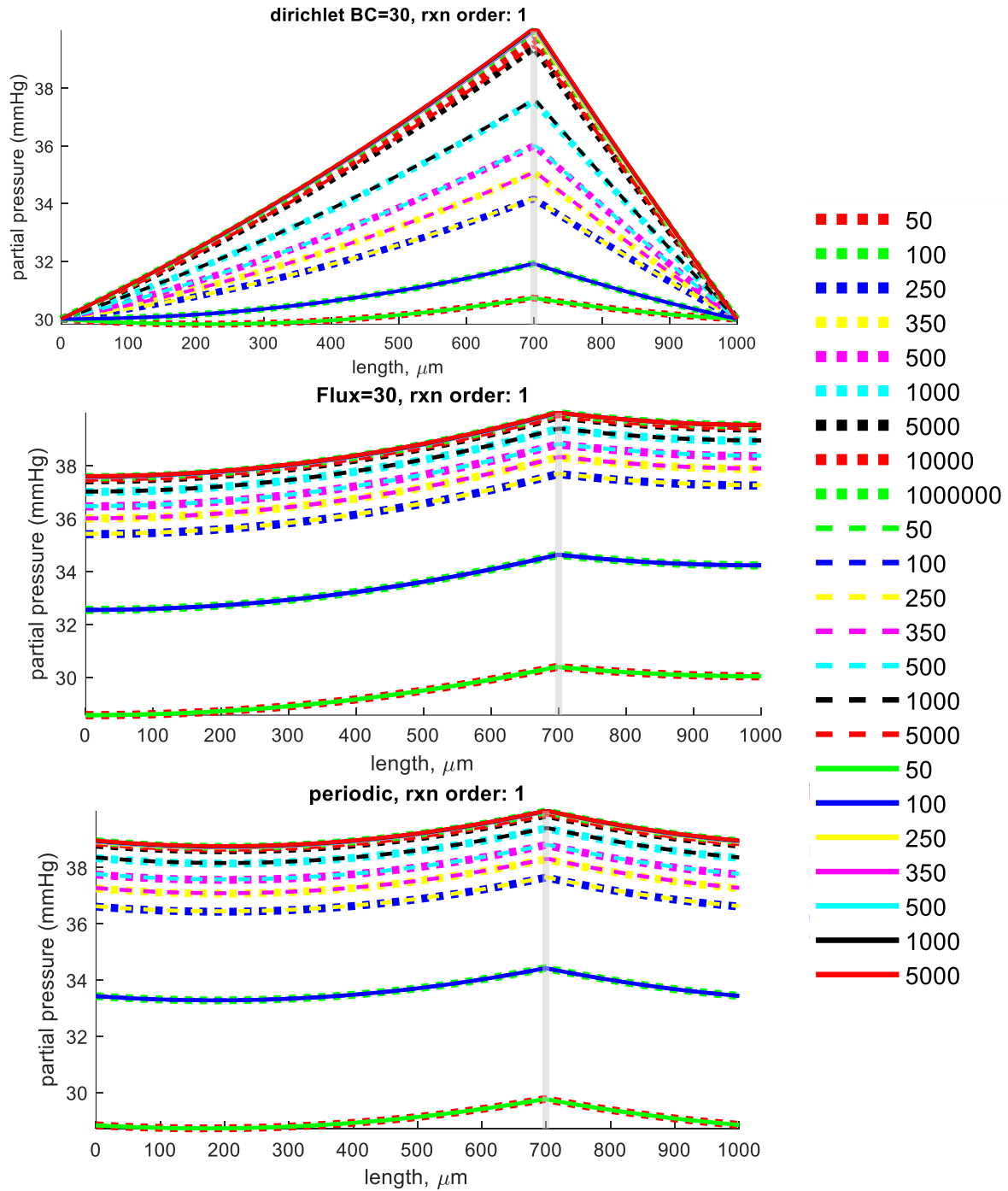


Figure 7.204. Comparison of different mass transfer models on mesh independence convergence between different boundary conditions.

Top) the convergence of all three methods using a Dirichlet value of 30mmHg and a 1st order reaction model. Middle) the convergence of all methods using insulated boundary condition on each of the domain ends and a 1st order reaction model. Bottom) the convergence of all methods using a periodic boundary condition and a 1st order reaction model. The convergence is slowest with the point source, and fastest with the edge distribution technique (model 3).

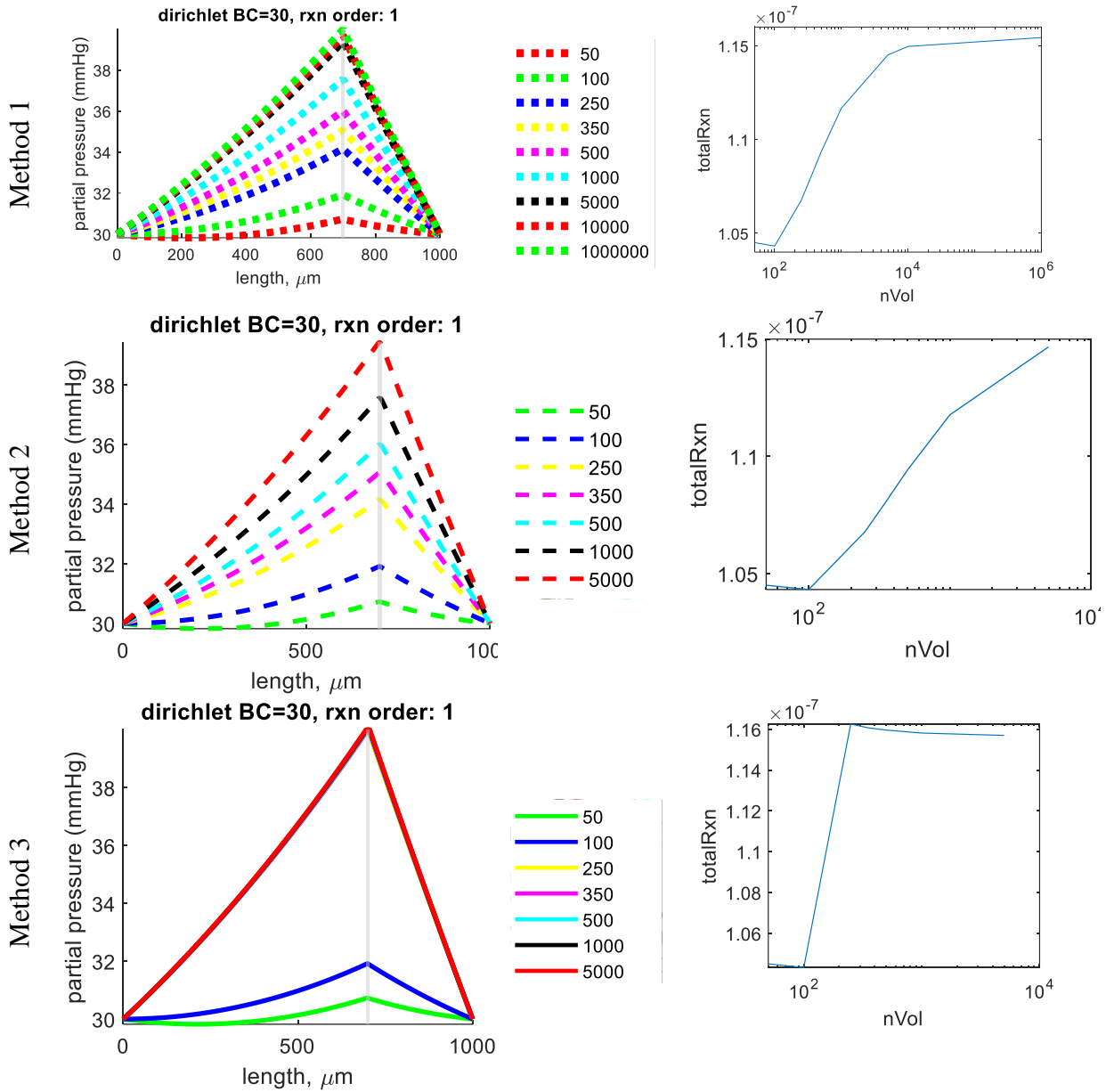


Figure 7.205. Comparison between different methods of mass transfer in a 1D model. The predictions use a Dirichlet boundary condition on the tissue edges (30mmHg) for Top) dotted lines indicate mass transfer method 1, Middle) dashed lines represent method 2, and Bottom) solid lines indicate the solution for method 3 for different volume densities. Note, method 3 converges must faster because this method effectively breaks the domain into 2 disconnected regions, making the problem structurally different and much easier to solve.

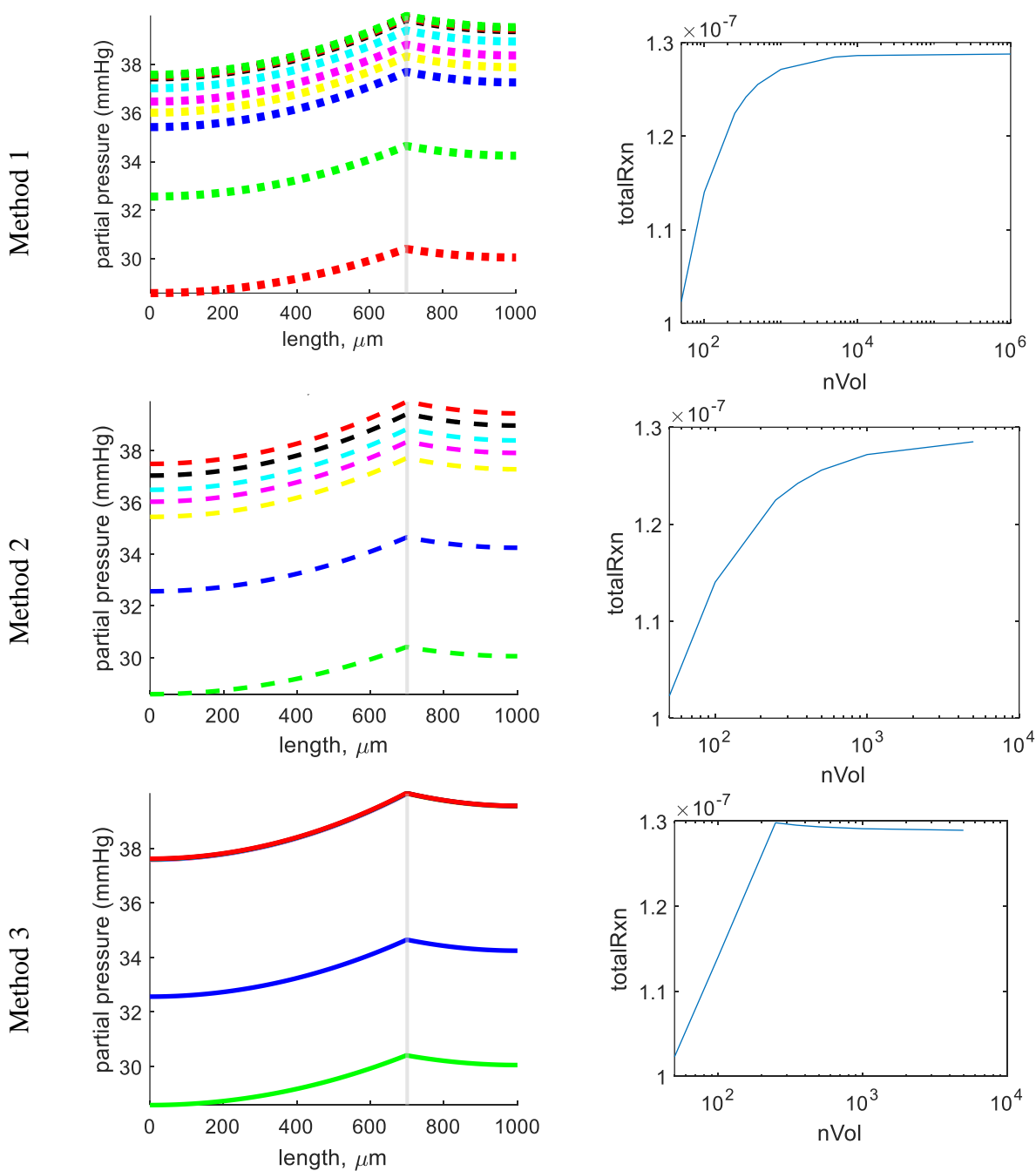


Figure 7.206. Comparison of mesh independence for insulated tissue boundaries (flux=0) and 1st order reaction).

Top) dotted lines indicate mass transfer method 1, Middle) dashed lines represent method 2, and Bottom) solid lines indicate the solution for method 3 for different volume densities. Method 3 converges must faster because this method effectively breaks the domain into 2 disconnected regions, making the problem structurally different and much easier to solve.

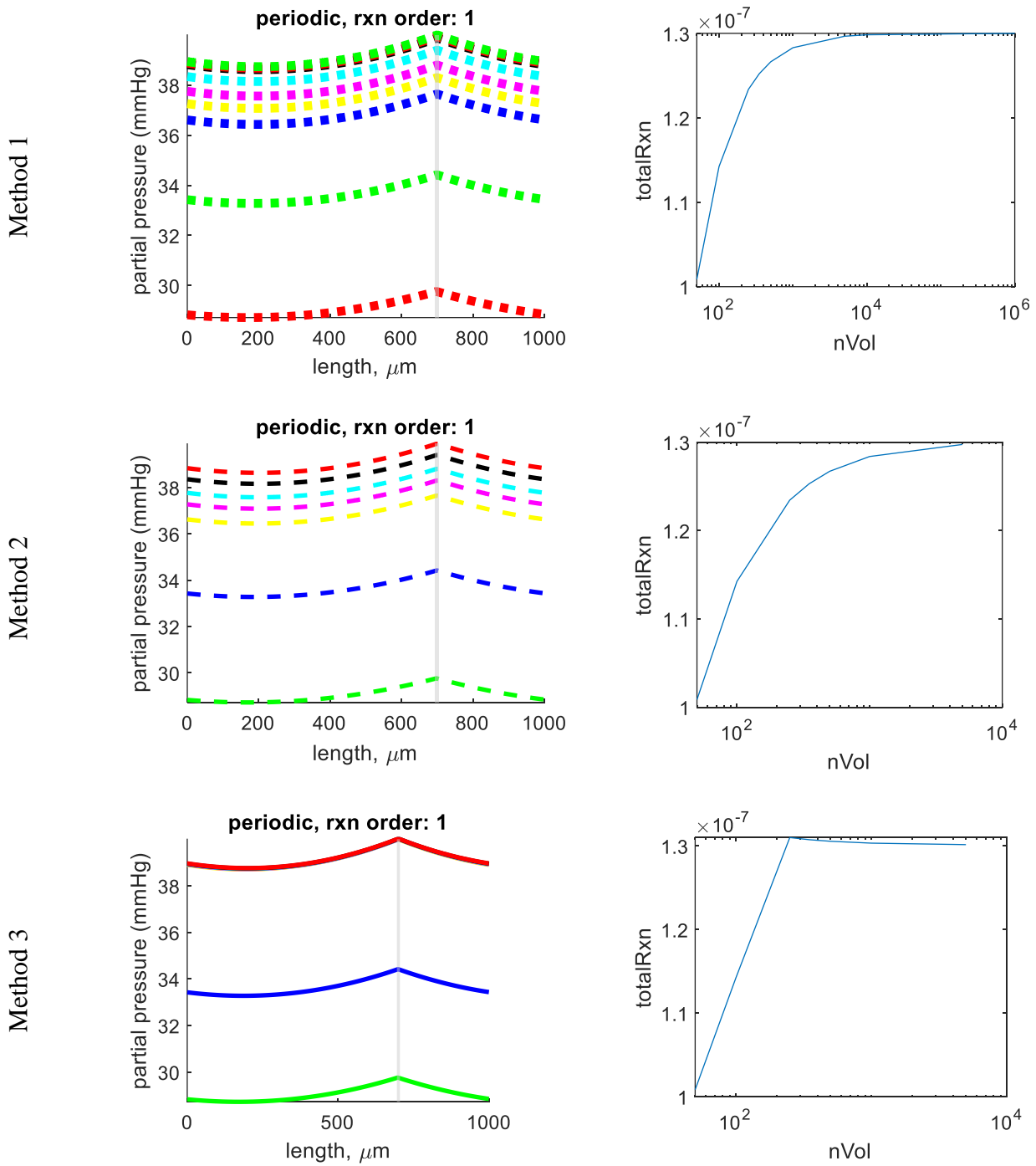


Figure 7.207. Comparison of mesh independence for periodic tissue boundary conditions and 1st order reaction.

Top) dotted lines indicate mass transfer method 1, Middle) dashed lines represent method 2, and Bottom) solid lines indicate the solution for method 3 for different volume densities. Method 3 converges must faster because this method effectively breaks the domain into 2 disconnected regions, making the problem structurally different and much easier to solve.

7.28.6 2D problem

Because the findings in 1D were inconclusive (due to the structural breaking of the problem into 2 distinct domains), it is important to investigate the problem with relevant mass transfer models (Method 1 and 3) in 2-dimensions. This formulation is still simpler and easier to investigate than the 3-dimensional counterpart yet does not suffer from domain splitting.

7.28.6.1 Diffusion in cylindrical coordinates

Conclusions:

- For a constant potential difference, $\phi_A - \phi_B$, a thicker vessel a_{large} experiences smaller resistance, and accordingly, larger fluxes
- Thinner vessels have smaller flow because a larger volume of the domain has to be overcome, but also because the resistance increases since the flow emerges from a more constricted circle with radius a_{medium}
- The gradient of the concentration profile becomes infinitely steep as a approaches 0. The flux also approaches 0. It may be plausible that the flux goes to 0 for an infinitely small point source because the space around an infinite point source is so “thin” it would have infinitely large resistance. In other words, an infinitely small point source discharges nothing for a given potential difference because the resistance becomes infinite. Note that numerical adaptations that use the cross sectional area between adjacent cells may overcome this, assuming there is no evaluation of flux at the source point, however this is an artifact of numerical error (source has 0 extent yet volume has nonzero extent) and is

not indicative of the real system. This is relevant for problems such as the Green's function.

- The discretized form of the equations is able to resolve a radius of 0, because the cross sectional area between the adjacent cells uses the radius halfway between the cells (numerical approximation). This is a mesh artifact and does not accurately represent the source as infinitely thin, but rather as the width of the first element. In other words, this does not reflect an infinitely thin source.

7.28.6.2 Problem formulation

The first approach for a 2D mass transfer model will use a single tube for feeding the domain. Using a cylindrical coordinate system, the balance is symmetric in the azimuth (θ) dimension (assuming the domain is also symmetric in θ). If the problem assumes no change in the axial (z) direction (domain and source are infinitely long), the problem is symmetric in the z dimension as well, reducing to a 1-dimensional problem (*cylindrical diffusion*):

$$D \frac{\partial}{\partial r} \left(r \frac{\partial c}{\partial r} \right) = 0 \quad (7.533)$$

Which can be integrated to yield the general form of the analytic solution (assuming diffusivity $D=1$):

$$\int \partial \left(r \frac{\partial c}{\partial r} \right) = \int 0 \partial r \quad (7.534)$$

$$r \frac{\partial c}{\partial r} = C_1$$

$$\int 1 \partial c = \int \frac{C_1}{r} \partial r$$

$$c_{cyl} = -C_1 \ln(r) + C_2$$

Which has been verified by insertion:

$$\frac{\partial}{\partial r} c = \frac{C_1}{r}$$

$$\frac{\partial}{\partial r} \left(r \frac{\partial c}{\partial r} \right) = \frac{\partial}{\partial r} \left(r \frac{C_1}{r} \right) \quad (7.535)$$

$$\frac{\partial}{\partial r} (C_1) = 0 \rightarrow \text{verified}$$

For the sake of completeness (due to the spherical implementation in other analytic problems such as the Green's method [31,164]), the same process can be performed in *spherical* coordinate system (where again D=1):

$$D \frac{\partial}{\partial r} \left(r^2 \frac{\partial c}{\partial r} \right) = 0$$

$$\int \partial \left(r^2 \frac{\partial c}{\partial r} \right) = \int 0 \partial r \quad (7.536)$$

$$r^2 \frac{\partial c}{\partial r} = C_1$$

$$\int 1 \partial c = \int C_1 r^{-2} \partial r$$

$$c = C_1 r^{-1} + C_2$$

$$c_{spere} = \frac{C_1}{r} + C_2$$

This is also validation by insertion:

$$\frac{\partial}{\partial r} c = -\frac{C_1}{r^2}$$

$$\frac{\partial}{\partial r} \left(r^2 \frac{\partial c}{\partial r} \right) = \frac{\partial}{\partial r} \left(-\frac{C_1}{r^2} r^2 \right) \quad (7.537)$$

$$\frac{\partial}{\partial r} (C_1) = 0 \rightarrow \text{verified}$$

The cylindrical implementation can also be solved with given boundary conditions (2 Dirichlet concentrations) for the particular solution and evaluated:

$$c_{cyl}(r) = -C_1 \ln(r) + C_2 \quad (7.538)$$

$$\text{for BCs: } c(a) = c_1 \text{ and } c(b) = c_2$$

$$c_1 = -C_1 \ln(a) + C_2 \quad (7.539)$$

$$c_2 = -C_1 \ln(b) + C_2 \quad (7.540)$$

Where Equation (7.540) can be simplified and inserted in to Equation (7.539):

$$c_2 + C_1 \ln(b) = C_2 \quad (7.541)$$

$$c_1 = -C_1 \ln(a) + c_2 + C_1 \ln(b)$$

$$c_1 = C_1 (\ln(b) - \ln(a)) + c_2$$

$$c_1 = C_1 \ln(b/a) + c_2 \quad (7.542)$$

$$\frac{c_1 - c_2}{\ln(b/a)} = C_1$$

Now C_1 can be substituted back into Equation (7.541) to give C_2 :

$$c_2 + \frac{c_1 - c_2}{\ln(b/a)} \ln(b) = C_2 \quad (7.543)$$

Which can now be plugged back into the original formulation to give:

$$c_{cyl}(r) = -\frac{c_1 - c_2}{\ln(b/a)} \ln(r) + c_2 + \frac{c_1 - c_2}{\ln(b/a)} \ln(b) \quad (7.544)$$

Which can be simplified to:

$$c_{cyl}(r) = -\frac{c_1 \ln\left(\frac{b}{r}\right) + c_2 \ln(r/a)}{\ln(b/a)} \quad (7.545)$$

Which, when endowed with the properties of Table 7.60, gives a reasonable profile as seen in Figure 7.208.

Table 7.60. Values used to graphically evaluate the analytic solution to diffusion equation in cylindrical coordinates.

Name	Symbol	Value
Endothelial concentration	c_1	35
Distal concentration	c_2	30
Endothelial radius	a	5
Domain radius	b	500

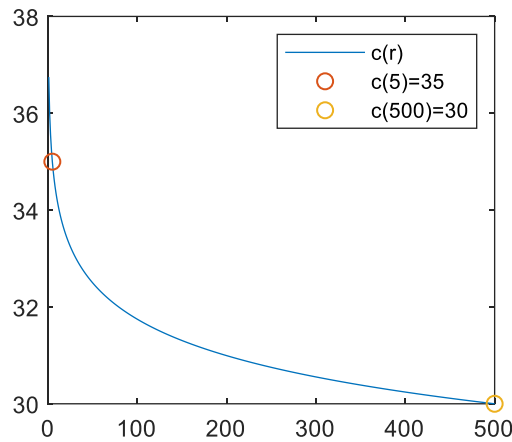


Figure 7.208. Validation of the implementation of the radial diffusion problem by verifying boundary conditions

Now that the equation has been validated, the evaluation at radius $r=0$ (the singular point) can be interrogated to investigate the robustness of the analytic method. It can be shown below that the variation of the location a (or likewise, the variation in concentration at a) results in a singularity at $r=0$. In the event the boundary condition is enforced for $a=0$, the solution is the value at c_2 along the entire domain. In other words, the analytic equation exposes a singularity around $r=0$. In the case that $a=0$, the value of c becomes constant, which is not the correct model either. This can be expressed analytically if $b = b$, and $a=0$ evaluated at $r=0$:

$$\begin{aligned}
c_{cyl}(r) &= -\frac{c_1 \ln\left(\frac{b}{r}\right) + c_2 \ln(r/a)}{\ln(b/a)} \\
c_{cyl}(r) &= -\frac{c_1 \ln\left(\frac{b}{r}\right) + c_2 \ln(r/0)}{\ln(b/0)} \\
c_{cyl}(r) &= -\frac{c_1 \ln\left(\frac{b}{r}\right)}{\ln(inf)} - c_2 \frac{\ln(inf)}{\ln(inf)} \\
\text{where } \frac{c_1 \ln\left(\frac{b}{r}\right)}{\ln(inf)} &= \frac{c_1 \ln\left(\frac{b}{r}\right)}{-inf} = 0
\end{aligned}
\tag{7.546}$$

And due to l'Hopital's rule:

$$c_2 \frac{\ln(inf)}{\ln(inf)} = c_2 \tag{7.547}$$

7.28.6.3 Implementation

Varying the model w.r.t. radius allows interrogation of how the profile changes with different models of the source. This is relevant for comparing the many models for mass transfer implemented on 1D-3D coupling paradigms, where three models are frequently used: (i) infinitely thin diameter [31–33,164,165], (ii) finite but very small[21,22,62], and (iii) true vessel diameter[23,24,85,264,267]. This variation only considers the modeling diameter, not the distribution diameter, so it does not offer insight into model (ii) which uses a combination of real

diameter (for computations) and a modified diameter (for distribution). The conclusions are stated at the top of Section 7.28.6.

Analytic implementation. The varying of vessel radius reflects boundary conditions enforcement and that the constant flux BC results in varying gradients at the vessel edge. Note, the concentration gradient gets more steep as the vessel radius decreases. This can be expressed analytically by evaluating the derivative of Equation (7.545):

$$\frac{dc}{dr} = -\frac{c_1 - c_2}{\ln\left(\frac{b}{a}\right)} \cdot \frac{1}{r} \quad (7.548)$$

Giving a flux of:

$$\begin{aligned} q &= -D \frac{dc}{dr} SA \\ &= -D \frac{c_1 - c_2}{\ln\left(\frac{b}{a}\right)} \cdot \frac{1}{r} (2\pi r) \\ &= -D \frac{c_1 - c_2}{\ln\left(\frac{b}{a}\right)} (2\pi) \\ &= \text{constant} \end{aligned} \quad (7.549)$$

In order to satisfy this constant flux, the concentration difference is inversely proportional to source radius, a . If the diameter changes and the concentration is maintained at the vessel edge, the flux will decrease as a function of diameter. This is a result of the increased resistance to reach

the opposing boundary and the lower surface area (higher resistance) between vessel interior and vessel exterior. These results can be seen in Figure 7.209 and Table 7.61.

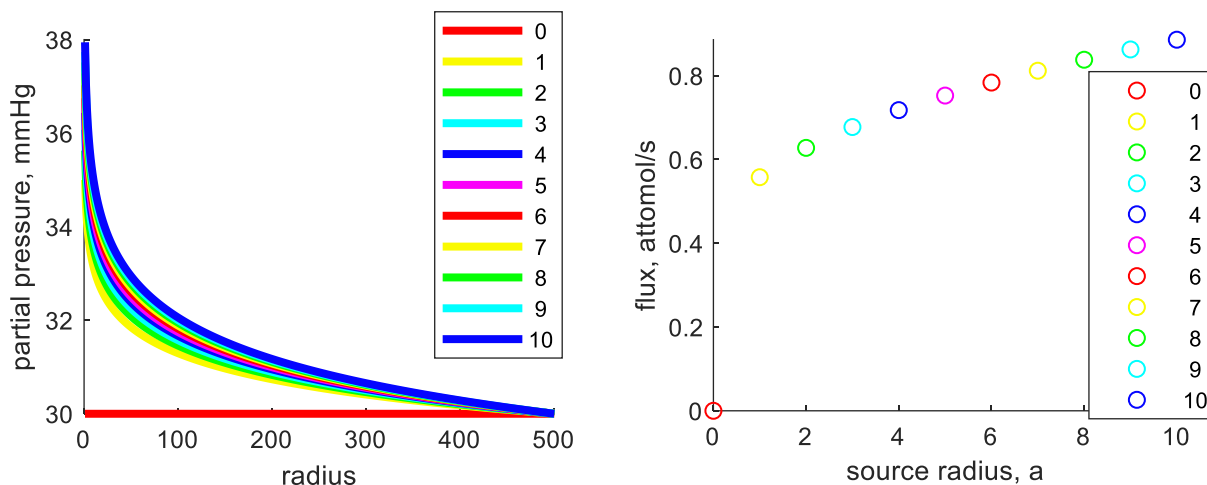


Figure 7.209. Varying the radius of the cylinder source (a) and its effect on the flux through the system.

The legend indicates the value of the vessel radius, a . Note, a radius of 0 results in nonphysical profile.

Table 7.61. Resulting flows at the initial and terminal segments when varying the vessel radius (a)

a	Q1	QN	Residual
1	5.06	5.06	0
11	8.23	8.23	0
21	9.91	9.91	0
31	11.30	11.30	0
41	12.56	12.56	0
51	13.76	13.76	0
61	14.93	14.93	0
71	16.09	16.09	0
81	17.26	17.26	0
91	18.44	18.44	0
101	19.64	19.64	0

Numerical implementation. The numerical implementation of the radial diffusion problem also shows promising results in agreement with the analytic solution. The edge detection for equation

generation for a 2D Cartesian mesh is given in Section 7.32.1. The conclusions are reviewed at the top of this section.

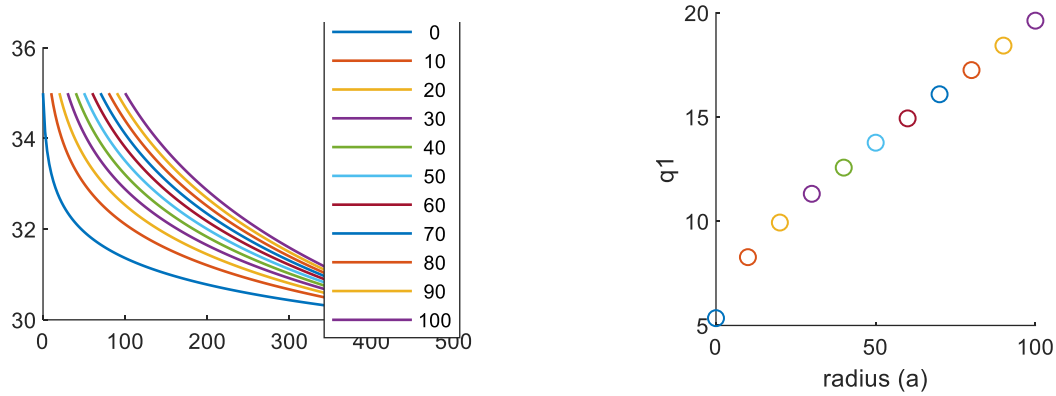


Figure 7.210. Varying the radius of the cylinder source (a) and its effect on the flux through the system using discretized implementation.

Note, the evaluation of radius $r=0$ is indicative of a diffusion problem that uses midpoint evaluation of flux computations.

Table 7.62. Resulting flows at the initial and terminal segments when varying the vessel radius (a)

a	Q1	QN	Residual
0	5.35	5.35	0.269e-12
10	8.28	8.28	5.92e-12
20	9.93	9.93	0.482e-12
30	11.31	11.31	0.620e-12
40	12.57	12.57	0.161e-12
50	13.76	13.76	0.247e-12
60	14.93	14.93	9.80e-12
70	16.09	16.09	0.194e-12
80	17.25	17.25	0.657e-12
90	18.42	18.42	8.31e-12
100	19.62	19.62	0.717e-12

Mesh independence for FVM. In order to validate the numerical method and discretization scheme, it is important to find mesh convergence with a 1D discrete grid to the analytic cylindrical diffusion problem. This can be accomplished by using a radial (cylindrical) model for the cross

sectional area between a 1D diffusion-reaction problem. This can be solved with linear spacing (Cartesian) or with logarithmic spacing (for comparison). The findings indicate that the steep gradient next to the source cause significant numerical error and a large number of volumes are necessary to converge the solution. In the case of linear spacing, the residual, defined by concentration at $100\mu\text{m}$ and the flux between a and $10\mu\text{m}$, indicates the residual is still larger than $1e-3$ after 1000 volumes. The logarithmic spacing reaches a value below $1e-3$ between 500 and 1000 volumes. The logarithmic spacing, however, rectifies this problem at a low discretization (50 volumes is already converged). In the future, this knowledge can be implemented on a linearly discretized mesh using a shape profile [165].

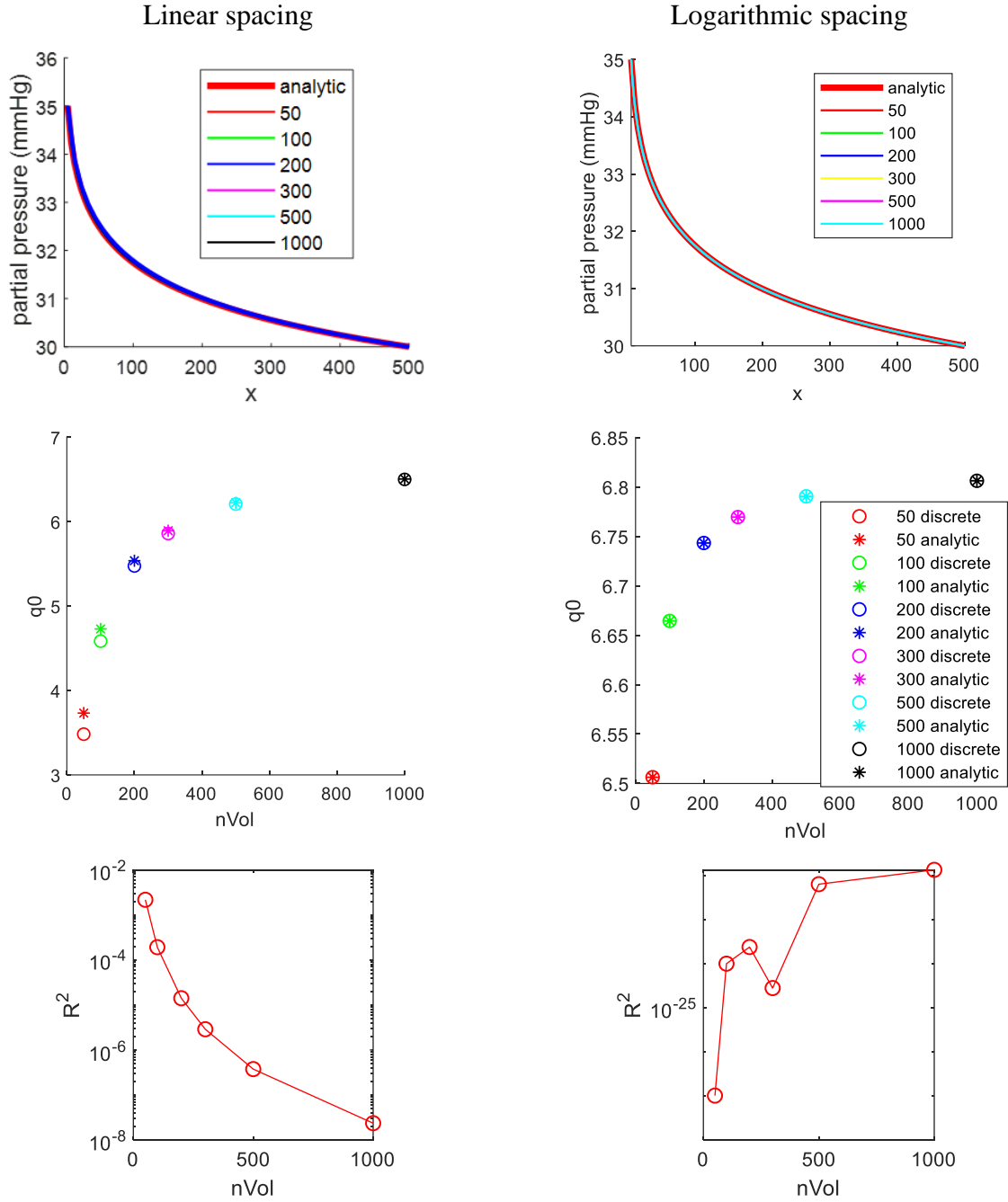


Figure 7.211. Comparison of different discretization schemes.

Top) concentration profile, Middle) difference from analytic solution at 100 μ m from the vessel wall, and Bottom) residual error as a function of discretization density (nVol). The linear spacing does not converge until ~300 volumes while the logarithmic spacing converges at a much lower density. Unfortunately, this type of geometric mesh modification destroys the Cartesian mesh logic (especially when applied to a dense network consisting of many vessels), so it will not be used in this work. Note, the discrepancies between the values in the middle row is due to the discrete calculation with differing Δx .

Numerical validation for 2D Cartesian mesh. The radial model is a good method for validating numerical implementation, but in real applications, the domain is represented by a mesh instead of an analytic domain. To model the mesh in 2D, a Cartesian grid (a mesh made entirely of rectangles) was generated. In order to investigate the robustness of this method, a test was devised that assigned boundary conditions that result in a domain similar to those in the analytic formulation above. Conclusions:

- The narrower the vessel, the steeper the gradient
- The steeper the gradient, the more volumes are needed to solve discretely
- The problem converges to a final profile rather quick (~50 elements per dimension)
- The implementation of the boundary condition on the exterior volume is insufficient to represent the geometry at $a=0$

A parametric study of mesh convergence is offered in Figure 7.212. Also depicted is a parametric study with variable values for vessel radius (a) with constant volumes. As the radius becomes thinner, the profile deviates more from the analytic solution.

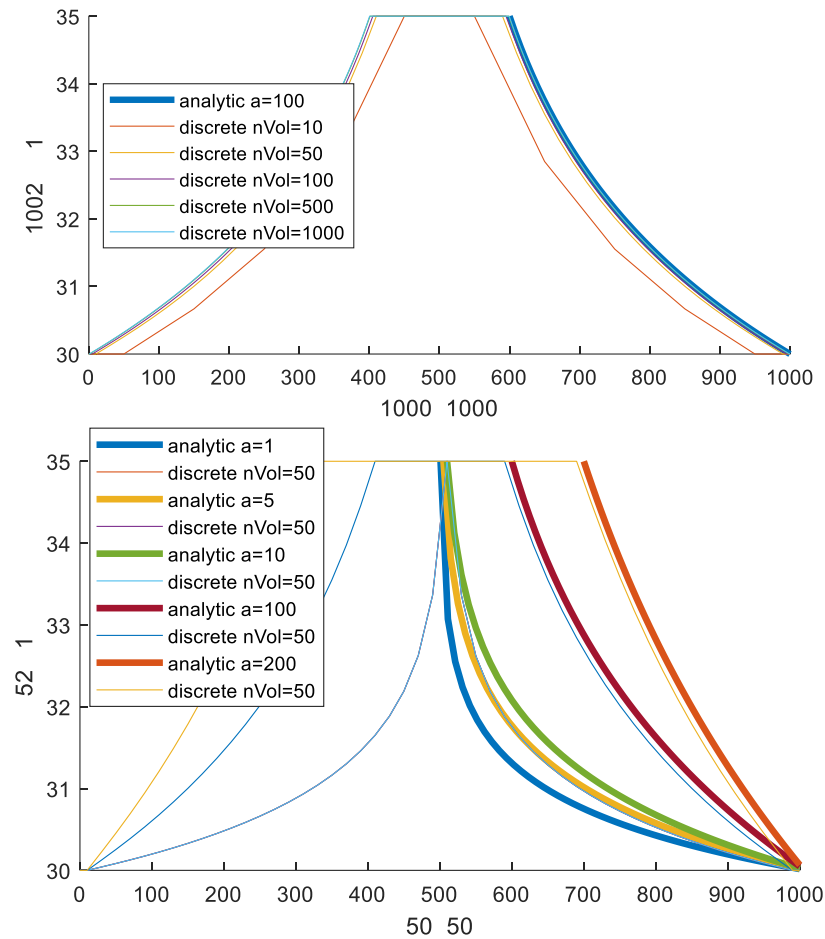


Figure 7.212. Comparison of analytic to discrete simulations of concentration profiles with varying the source vessel radius (a).

Top) different levels of discretization show a convergence quickly (<50 volumes, yellow line). Bottom) numerical implementation aligns well across many source radii. Note, the slight differences are attributed to poor discretization at the boundary edge (see Section 7.14 half-volume technique for more information).

The concentration profile of the entire domain can be visualized using Matlab's surface mesh function, where the z-coordinates of every mesh element can be assigned by the concentration. This is shown for varying volumes in Figure 7.213 and for many radii in Figure 7.214.

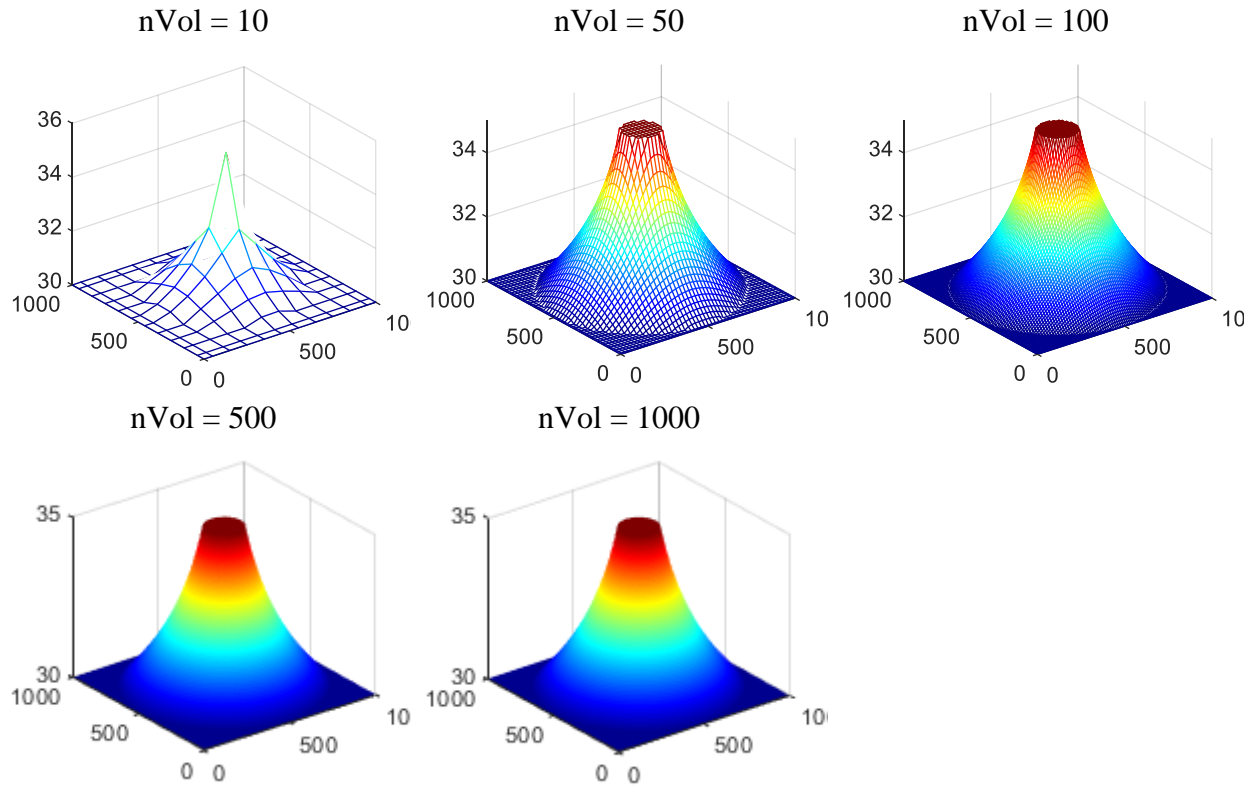


Figure 7.213. 3D profiles of the concentration simulated with numerical implementation while varying the mesh density.

The final profile is reasonably achieved by a 50x50 mesh (Vol=50).

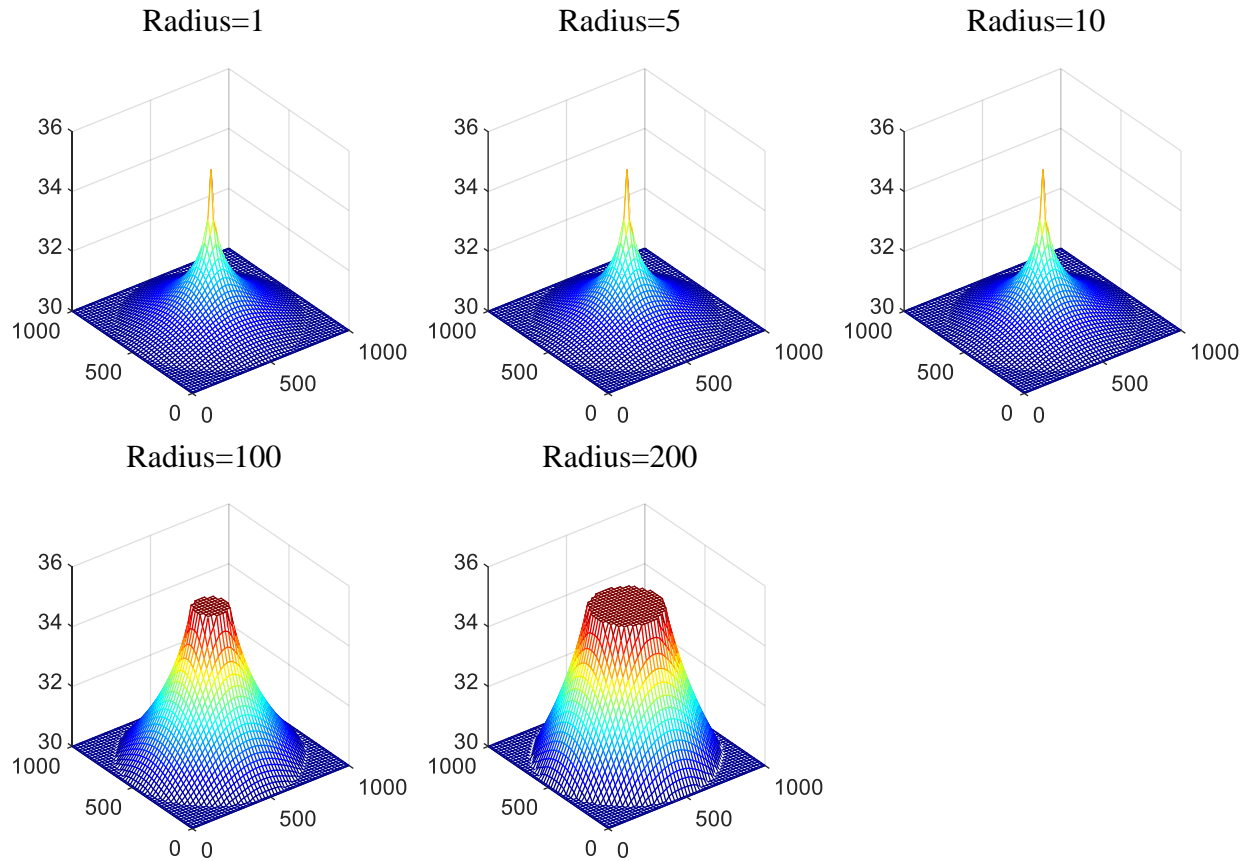


Figure 7.214. 3D profiles of the concentration simulated with numerical implementation while varying the vessel radius, a.

The residual error is also compared for the different discretization schemes. Error = $\text{abs}\{\text{analytical solution (c[800microns])} - \text{numerical value (c[800microns])}\}$

nVol	10	50	100	500	1000	5000
Error at 800 microns (mmHg)	3.37e-2	1.85e-2	9.21e-3	3.42e-3	2.69e-3	1.97e-3

7.28.7 3D problem

Simulating the 3D version of the proposed mass transfer problem can be solved with many methods. The obvious first choice would be a full, body-fitted 3D mesh of the vasculature and the tissue combined. Unfortunately, due to the size of the vasculature (large number of vessels), the

simulation of the vasculature alone is computationally prohibitive as discussed in Section 1. To overcome this barrier, many models have been developed to couple a 1D vascular model to a 3D tissue domain. These include (i) an infinitely-thin source term [31–33,164,165], (ii) a body fitted mesh between the vasculature and tissue [23,24,85,264], and (iii) 1D-3D point-centered mesh coupling [21,22,62].

The first method is useful for deriving the analytic solution, giving immediate mesh independence. Unfortunately, this method suffers from a non-physiological mass transfer model (as discussed in Section 7.28.6) and the nonlinear fixed point iterative solving methodology is time prohibitive for vascular structures larger than ~ 100 segments.

A body-fitted mesh is another method to adapt a 1D network to a 3D mesh, as it does not suffer from the limited mass transfer model. This method is unfortunately also computationally prohibitive for large structures. This is an effect of the unstructured mesh density which must be dense surrounding small vessels, leading to meshes too large to simulate at the scale of large microcirculatory sections of the mouse brain ($\sim 1\text{mm}^3$).

The third model is a point connectivity between a hexahedral mesh and an encompassed vasculature. The vasculature points discharge, through mass transfer, oxygen into a single mesh element. This method maintains numerical stability for large microcirculatory structures by avoiding dense meshing as in body-fitted meshes. Unfortunately, however, the mesh convergence can become unstable when the network has a large range of vessel diameters ($\sim 1\mu\text{m}$ to $\sim 100\mu\text{m}$). In such cases, as the mesh continues to refine and resolve microgradients between small vessels, the discharging source volumes become smaller than the large vessels. In these circumstances, a massive mass transfer flux from a pial vessel will discharge into a mesh volume of significantly

smaller size, leading to a large local gradient surrounding that volume. For more information on meshing in relationship to large gradients, refer to Section 7.28.6.3.

To incorporate the benefits of the body fitted meshes (vessel size and orientation) and the scalability of the point-connection, three new techniques have been developed relying on the Cartesian mesh logic; (i) the entire simulation will be solved in 3D using a Darcy-like flow problem, (ii) a 1D flow simulation will be projected into a 3D Cartesian domain and the convection-mass transfer-diffusion-reaction sequence will be solved entirely in a 3D mesh, and (iii) a method for connecting large vessels with a distributed vessel edge detection and smaller vessels with the point connection. After testing, only option (iii) is considered reasonable for further pursuit. Note, all simulations require a method for edge detection between a 1D vascular network and a 3D Cartesian mesh. The algorithm and implementation for this can be reviewed in Section 7.32.3.

7.28.7.1 3D blood flow (Darcy flow) and 3D convection-diffusion-rxn

The flow vector of a blood flow simulation can be simulated entirely in a mesh using a Darcy-like flow (hindered diffusion through a porous medium). First, a mesh is labeled with a mask delineating elements corresponding to the endothelial layer, blood plasma, and extravascular space. These labels will allow the program to generate equations of flux between adjacent elements. For a description of how the labeling algorithm is executed, refer to Section 7.32.3. The first step of the simulation is to define the bulk blood flow vector. To do this, the vascular-vascular flux is defined by a Darcy driven flow:

$$f_{darcy} = -KA \frac{dp}{dx} \quad (7.550)$$

Where K is related to viscosity, μ , from the HP equations, but is not necessarily the same value. In fact, later simulations will prove that when using this model, the value of K must change for each simulation and each mesh discretization size. The flux between endothelial elements and vascular elements is modeled as a mass transfer. A is the cross sectional area between two adjacent volume elements and dx is the distance between adjacent element centers. The remainder of the elements are endowed with a Dirichlet boundary condition ($p = 0$) and the faces between endothelial layers and vessel layers have no flux. The flux balances for the first simulation exist only in the vasculature nodes:

$$\begin{aligned} 0 &= \nabla \cdot f_{darcy} \\ 0 &= -KA \frac{d^2p}{dx^2} \end{aligned} \quad (7.551)$$

For the second stage of simulation, the interface between adjacent vasculature elements is endowed with a convection flux model:

$$f_{conv} = qAc, \quad q = -K \frac{dp}{dx} \quad (7.552)$$

Here, c represents the concentration of oxygen in a given element. The faces separating endothelial elements from vasculature elements undergo a mass transfer flux. The same model is also used at the endothelial - tissue interface and between adjacent endothelial elements:

$$f_{MT} = -UA \frac{dp}{dx} \quad (7.553)$$

Where U is the mass transfer coefficient of the endothelial layer. The tissue-tissue interface will employ a diffusivity flux:

$$f_{diff} = \vec{\nabla} \cdot \left(D \frac{dc}{dx} A \right) \quad (7.554)$$

Where D is the tissue permeability (diffusivity). The flux balances for the tissue domain include a reaction model:

$$f_{rxn} = -k_1 c V \quad (7.555)$$

Where k_1 is the 1st order reaction rate (metabolic rate of oxygen in brain tissue) and V is the volume of the mesh element. The balance for each tissue element then becomes:

$$0 = f_{diff} + f_{rxn} + f_{MT} \quad (7.556)$$

Note, the addition of some fluxes (like f_{MT}) is entirely dependent on the description of the faces of the given element, and these fluxes may be absent in many cases. The vascular mass balances incorporates the convection model and diffusion model:

$$0 = f_{conv} + f_{MT} \quad (7.557)$$

And the endothelial elements will undergo a singular mass transfer (hindered diffusion) flux:

$$0 = f_{MT} \quad (7.558)$$

It is also important to acknowledge that the discretization scheme is important; the mesh needs to be fine enough to resolve at least one interior voxel in all segments (for continuity). In the event the mesh is too coarse to resolve a vessel, there is a break in the network structure and no flow is possible, leading to a singular matrix. Some completed case studies are pictured below:

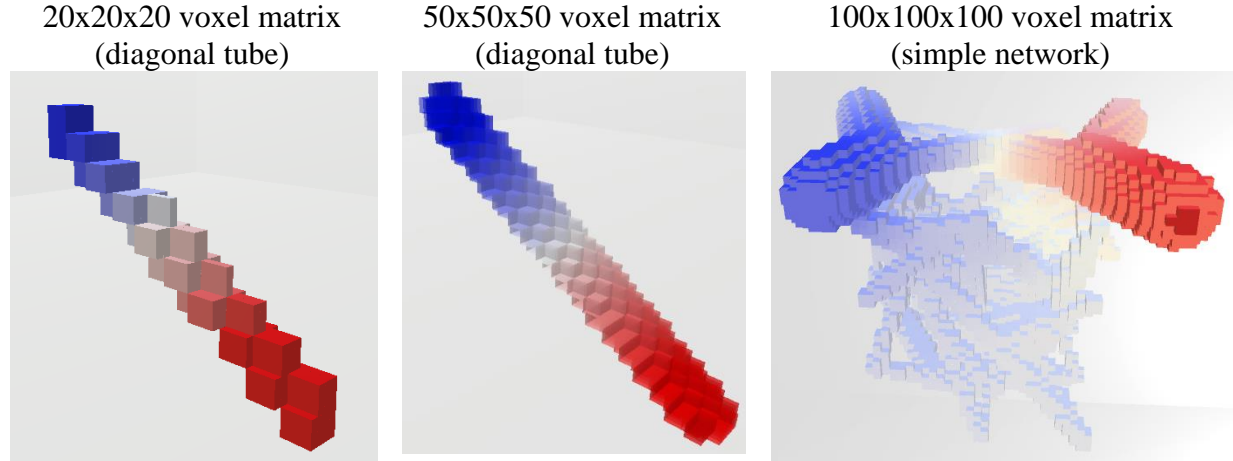


Figure 7.215. Vascular flow and pressure simulated in a full 3D voxel matrix using Darcy's law. The results show a reasonable gradient between the inlet and outlet of the systems.

Convection. The next step is to validate convection through the network using the resulting pressures and flows from the previous step. This can be accomplished by evaluating the flows across every face using the following equation:

$$f = -K \frac{dp}{dx} A \quad (7.559)$$

And the resulting convection through the voxels with nonzero bulk flow (f) is given by:

$$f_{conv} = f c_{source} \quad (7.560)$$

Unfortunately, in coarser discretization patterns, some hexahedrons are labeled as vessel elements, but only share a single neighboring vessel element. When solving the Darcy diffusion in these two elements, the trivial result is that the pressure in both elements is identical (no flow

between the elements). Elements that have no flow are visualized in blue in Figure 7.216. In this special case, a simple assignment equation in lieu of a convection equation should be written to avoid matrix singularity:

$$0 = -c_i + c_j \quad (7.561)$$

Where c_i is the blue element and p_j is the neighboring vessel element.

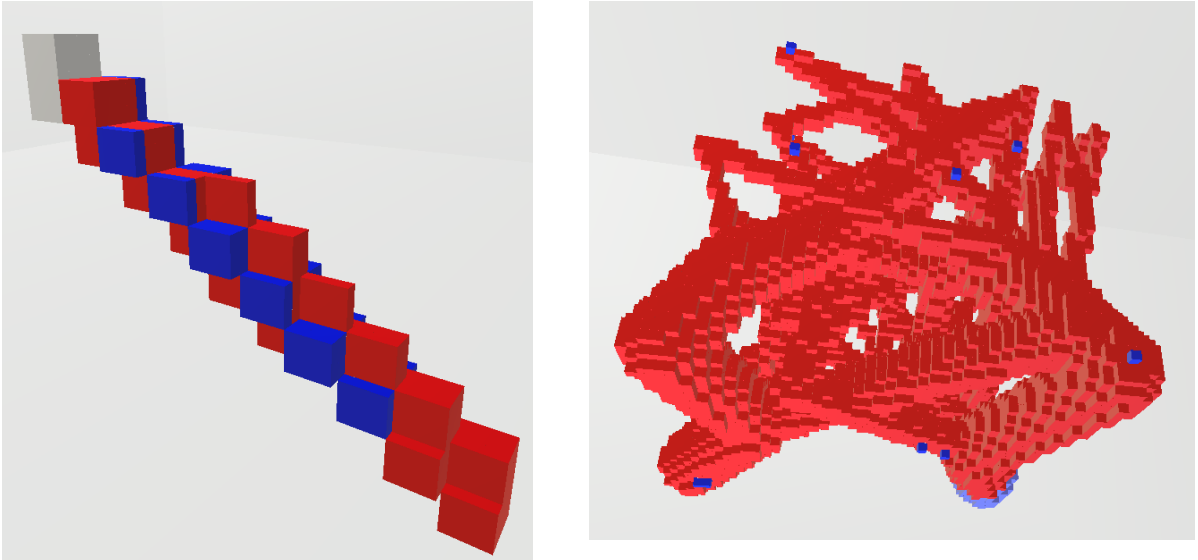


Figure 7.216. Examples of vessel interior cells that only share a single neighboring interior cell. Left) a simple vessel and Right) a simplified 1/10 KF dataset. The blue cells share the same pressure as their single neighbor and thus do not entertain a bulk convective flow.

In extreme cases where there are more than one cell isolated in a cluster, a second equation is formed that follows Equation (7.561) and is added to the original equation. This is continued until all adjacent vascular cells are accounted for. The result of the simulation is visualized in Figure 7.217. Note, the distribution is uniform everywhere due to a system devoid of reactions. Note, the

outlet BC (not necessary in convection) is still assigned due to programmatic efficiency. This outlet concentration constitutes the blue elements in Figure 7.217 Left.

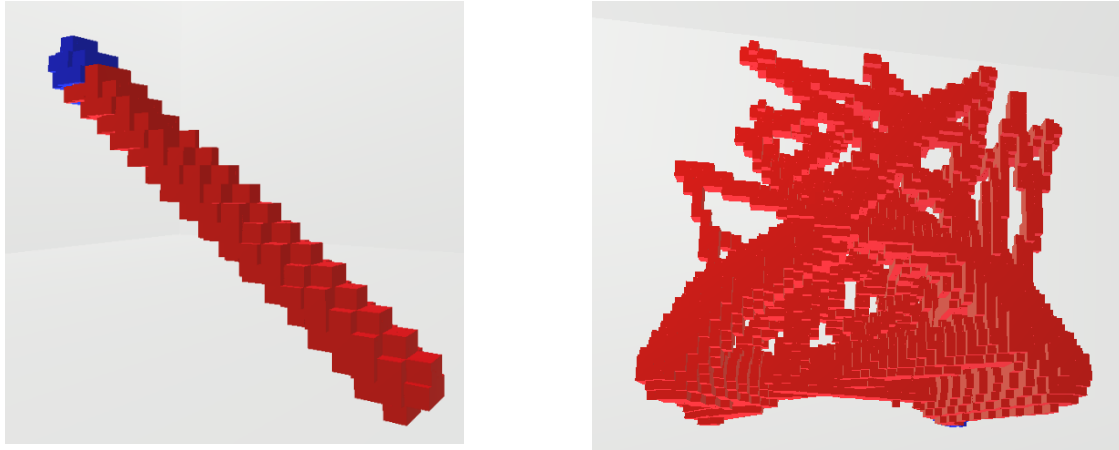


Figure 7.217. Examples of convection in networks that use a diffusive-like equation to account for orphaned cells.

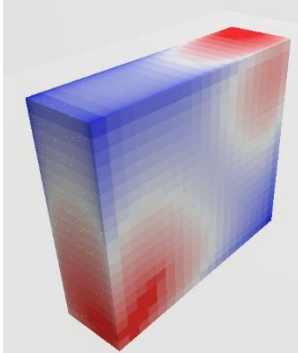
Left) a simple vessel and Right) a simplified 1/10 KF dataset. Note, the blue cells correspond to the lower Dirichlet outlet BC (symmetry is not programmed yet).

Mass transfer without reaction. The next simulation is imperative for validating the mass transfer interface. The equations for this case study follow the previous sections, except with a nonzero value for the mass transfer coefficient (U). Moreover, the tissue undergoes diffusion as follows:

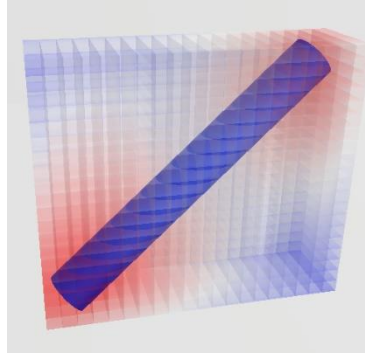
$$0 = D \frac{d^2 c}{dx^2} A \quad (7.562)$$

The tissue also has Dirichlet boundary conditions of value 0. The results have validated the simulations:

Exponential color scaling



Exponential color scaling



Linear color scaling

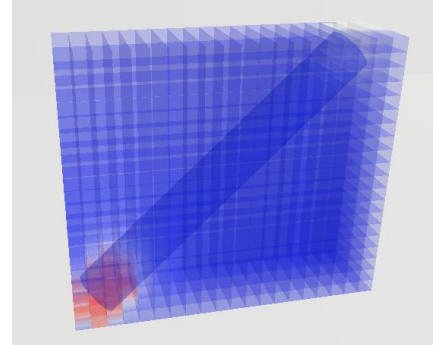


Figure 7.218. Sample case study of convection with a very low Darcy constant, K , (=very low flow).

The boundary conditions for convection were two Dirichlet BCs, hence the outlet of the vessel turns into a source for the tissue. Note, the red on the mesh wall is considered numerical error in the solution vector.

Mass transfer with reaction. The final case study involves using a tissue reaction and insulated boundary conditions on the tissue exterior. In these cases, the tissue uses the following equation:

$$0 = D \frac{d^2 c}{dx^2} - k_1 V c \quad (7.563)$$

Where V is the volume of the cell element and k_1 is the 1st order reaction rate. The results are given in Figure 7.219.

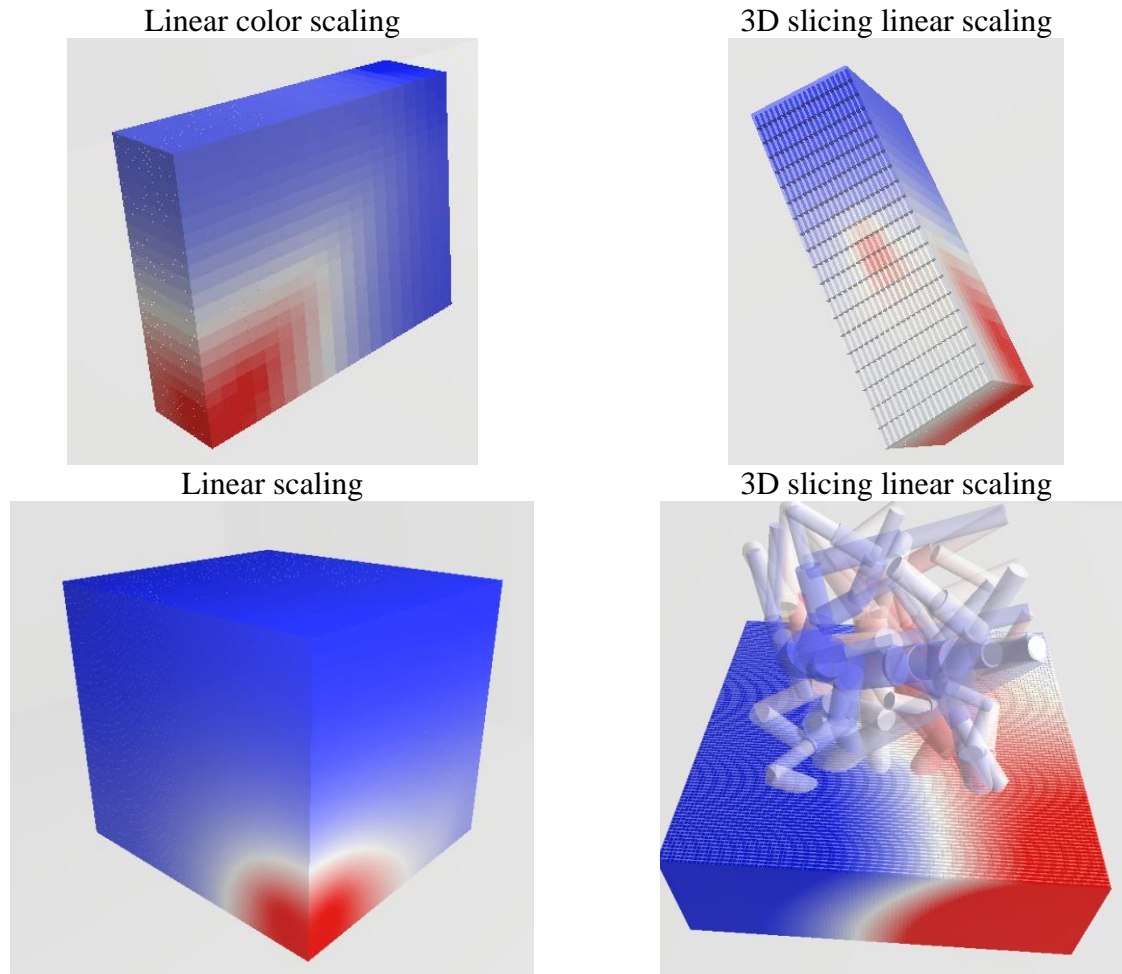


Figure 7.219. Case study for convection - mass transfer - diffusion – reaction system performed using only a 3D mesh.

The results indicate a reasonable distribution of concentration throughout the tissue. Note, the values used for constants were not validated.

In conclusion, the 3D simulation shows reasonable trends, however the validation against a 1D network flow shows discrepancy in the values. This discrepancy cannot be rectified by a simple adjustment to the Darcy coefficient, but instead requires a different flow model.

To properly discretize the network for this simulation, the networks should have a diameter no smaller than 4 microns, the wall thickness should be set to 1 micron, and the resolution of the dataset should be ~1 micron mesh cell edge length.

7.28.7.2 1D blood flow projected to 3D mesh

The previous section described a full 3D Darcy simulation of blood flow and convection-mass transfer-reaction-diffusion system. Unfortunately, this simulation found an inherent discrepancy in the blood flow model. To rectify this, a model has been created to solve a 1D blood flow model (HP) and project the flow vector onto the 3D mesh. This new model would then solve the convection-mass transfer-reaction-diffusion system with only a 3D mesh. Unfortunately, the staircasing effect of the voxelation (pixilation in 3D) causes some cells to lose mass unnecessarily as seen in Figure 7.220.

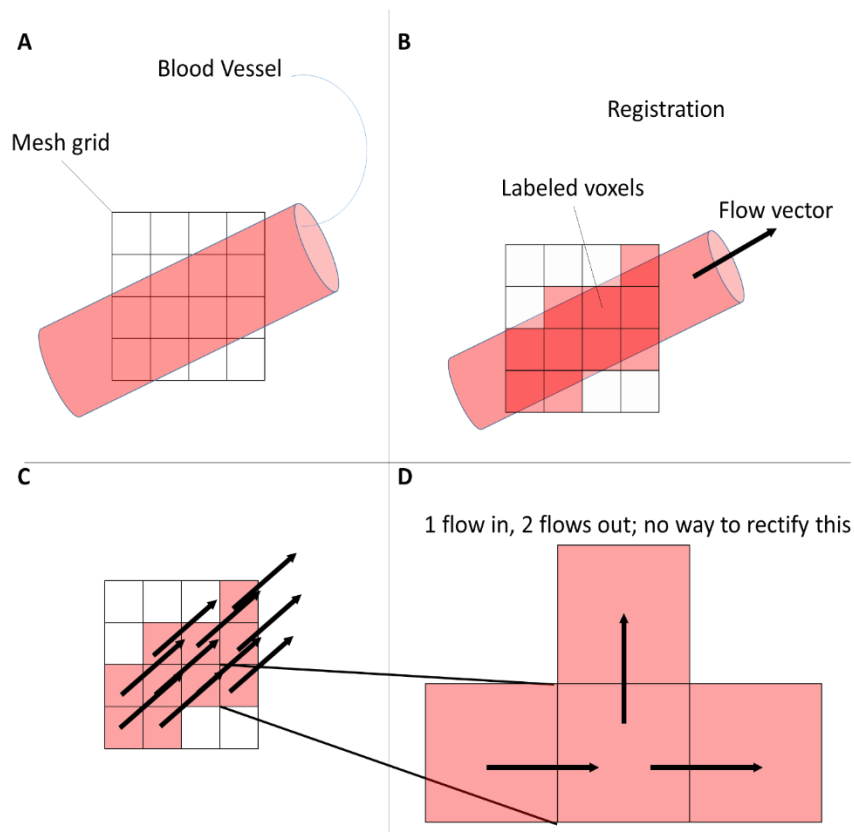


Figure 7.220. Schematic diagram of a network cylinder (with embedded flow vector) registered to a voxel matrix in 2 dimensions.

As can be seen from this simple case study, some voxels will not be able to maintain mass balance.

Suggested remedies to this problem have not been identified and thus this approach has not been investigated further.

7.28.7.3 1D blood flow, 1D convection, 3D diffusion-rxn, mass transfer

The blood flow will be modeled with Poiseuille flow throughout the vascular structures as given in [21,234] as in Equation (7.564). Biphasic blood flow can also be implemented as explained elsewhere [21,22,234] but is outside the scope of this study. Oxygen enters the domain through the vasculature where it moves via convection through the network and leaves through a mass transfer into the surrounding tissue as in Equation (7.565). The oxygen that enters the tissue domain is permitted to diffuse while being metabolized following Equation (7.566). The boundary conditions follow previously published values [22].

$$\Delta p = \alpha f; \quad 0 = \vec{\nabla} \cdot f \quad (7.564)$$

$$UA \frac{c_v - c_t}{\partial x} = \frac{\partial c_v}{dx} f \quad (7.565)$$

$$D \frac{\partial^2 c_t}{\partial x^2} + UA \frac{\partial c_v - c_t}{\partial x} = -k_{met} c_t V \quad (7.566)$$

Here, f is the bulk flow field derived from linear, U is the transmembrane permeability of the endothelial layer, A is the endothelial layer thickness, c_v is the concentration of oxygen in the vasculature, and c_t is the concentration in the tissue. In the event of biphasic blood flow and simple convective model of oxygen, the oxygen will be attached to the RBC phase of blood. More

complex models will implement the Hill equation as reviewed in Section 7.36 and published elsewhere [21,22].

The tissue oxygen concentration will be modeled as a diffusion-reaction domain with a mass transfer from the vasculature:

$$\frac{dc_o}{dt} V_{tiss} = \vec{\nabla} \cdot (D \nabla c_o) V_{tiss} - UA_{vasc} \Delta c_o - k_1 c_o V_{tiss} \quad (7.567)$$

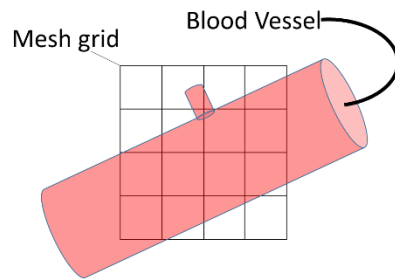
Point-centered connection. In this model, the mass transfer will be calculated using the area at each network point:

$$A_{vasc} = \sum_{i=1}^{nConnectfaces} \frac{1}{2} 2\pi r_i \quad (7.568)$$

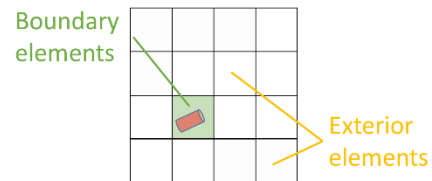
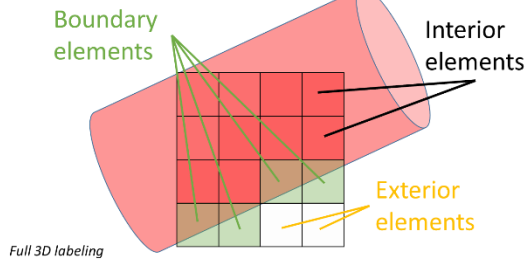
The network point will be connected to a mesh face through a single mass transfer connection, meaning that each network point will discharge a flux through a mass transfer flux into a single mesh element. This method has a drawback when the largest vessel becomes larger than a single mesh element, as the mass transfer flux overpowers the mesh element and the gradient significantly increases.

Edge detection. In order to overcome this limitation, the edge detection used in Section 7.28.7.1 (full 3D mesh simulations) will be used for large vessels (vessels spanning more than 1 mesh element) while smaller vessels will discharge using the point-centered mass transfer model. This new model will gain the advantages of full 3D resolution for larger vessels and simplified point-

centered connection for smaller vessels. The edge detection is identified following a simple test comparing the perpendicular distance between each voxel center and the vessel centerline. A fuzzy logic is used to resolve the endothelial layer, where the hard constraints of labeling based on vessel diameter can be relaxed in the case of the endothelial wall (up to $\frac{1}{2}$ the wall thickness). To create a smooth connection between adjacent segments (modeled as cylinders in 3 dimensions), a sphere with identical radius is added to the end of each vessel. More details on mesh cell labeling in 3D can be found in Section 5.5. This method accounts for 4 scenarios:

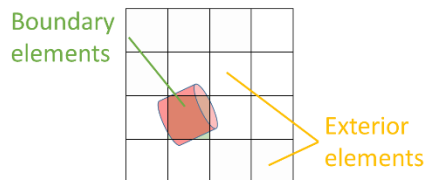
A

B Scenario 1: Vessel encompasses many mesh elements **C** Scenario 2: Vessel is smaller than 1 mesh element



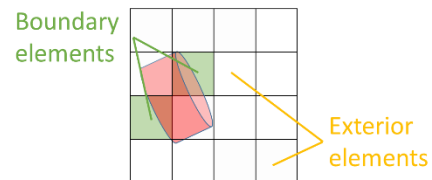
1D-3D coupling at vessel terminals

D Scenario 3: Vessel same width as 1 mesh element



1D-3D coupling at vessel terminals

E Scenario 4: Vessel same width as 2 mesh elements



1D-3D coupling at vessel terminals

Figure 7.221. Graphic depiction of four scenarios when classifying mask on mesh to determine equations.

All mesh elements are classified as vessel interior elements (red squares), vascular boundary elements (green squares) and exterior cells (white squares). These classifications later translate to equations of mass transfer (for boundary elements), assignment equations (interior elements) and diffusion-reaction equations (for exterior elements).

In order to keep track of multiple vessels that may attach to a single mesh element, a matrix is formed that holds lists of which mesh elements correspond to each vessel. This is later interrogated to develop a matrix that tracks how many mesh elements belong to the surface of each vessel and another matrix for network node connection to each mesh element. When forming the equations for the mesh elements identified as blood vessel elements, a simple Dirichlet equation is used to assign the vascular element value to the mesh element value:

$$c_{tiss}^j = c_{vasc}^i \quad (7.569)$$

Note, this mesh element does not enjoy any reaction, diffusion, or mass transfer flux.

Assignment of the exterior cells follows the mesh diffusion-reaction system outlined in Section 7.28.7.1. The mass transfer elements are assigned by interrogating the vascular segment connectivity as follows:

1. FOR i = 1 TO mesh.nVolumes DO
2. pointsForMeshCell = getPointsForMeshCell(i); //retrieves row of matrix
3. FOR j = 1 TO pointsForMeshCell
4. v1 = i; aPtIdx = pointsForCell[i];
5. meshCellsForPoint = getMeshCellsForPoint(aPtIdx); //retrieves row of matrix
6. fullSurfaceArea = nwk.getSurfaceAreForPoint(aPtIdx)
7. SA = fullSurfaceArea/meshCellsForPoint.Length;
8. coefficient = uu/dx*SA;
9. aMatrix.addFlux(v1, v1, v2, coefficient)
10. aMatrix.addFlux(v2, v1, v2, -coefficient)
11. ENDFOR
12. ENDFOR

Case Study, simple KF. Validation of the edge detection algorithm to produce meaningful simulation results at different levels of mesh density is offered. Note, all simulations converged to reasonable tolerances for mass balance overall and mass balance at each node.

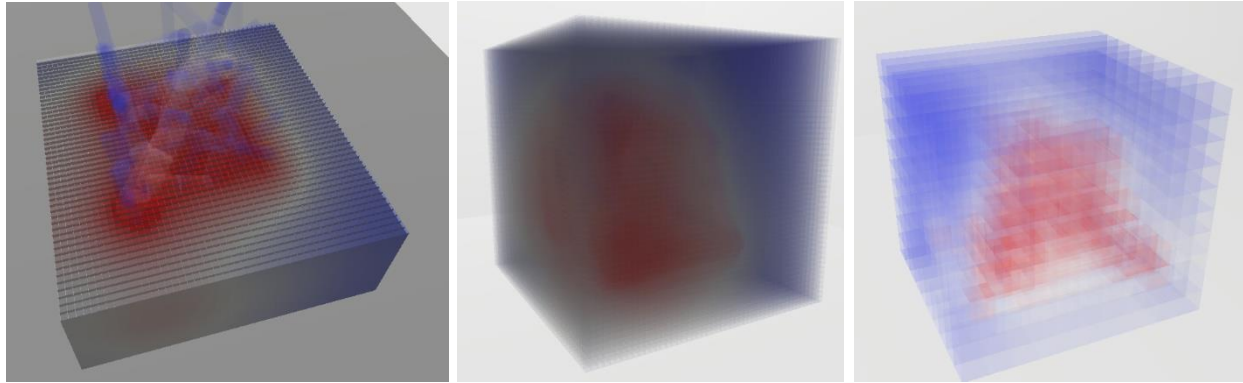


Figure 7.222. The slice (Left) and transparency (Middle) view of the oxygen distribution around the simplified synthetic network shows reasonable and consistent trends between the dense mesh (Middle) and coarse mesh (Right).

This is significant, because at the dense mesh resolution, each vascular network segment is comprised of many mesh elements, identified with edge detection, however in the coarse mesh the vessel elements discharge the entire mass transfer into a single mesh element.

7.29 Appendix AC: Models of aging brain

The aged brain suffers from a variety of vascular changes. A few have been implemented in the main manuscript but a more in-depth list is offered here binned into two categories;

(i) widely agreed upon statistics and (ii) more rarely seen morphological changes. These include widely agreed upon:

- Decrease vessel density (nSgm/mm³) of ~20%
- Decreased hematocrit by ~30%
- Increase in microocclusion occurrence (from negligible in young) to ~30%
- Increased tortuosity (in white matter penetrating arterioles, do not affect GM)
- Decrease VEGF production (specifically less production from HIF-1)
- Thickening of endothelial basement membrane (decreases wall flexibility)

A proposed:

- RBC velocity increases up to 50%
- Vessel lumen diameter decreases by up to 30% (higher percentage in pials than in microcirculation)

These values are summarized in a table:

Table 7.63. Comparison of vascular topological changes observed in the aged brain

Model	Number	Units	Citation
Vessel Number	↓5-35	%	Bullitt 2010 (pial vessels human)
	↓20	%	Faber 2011 (pial vessels mouse)
	↓20	%	Murugesan 2012 (capillaries mouse)
	↓20	%	Desjardins 2014 (microcirc rat)
	↓22	%	Wilkinson 1981 (rat cortex)
	↓27.4	%	Casey 1984 (capillaries rat)
RBC Speed	↑0.5	mm/s	Desjardins 2014 (microcirc rat)
	↑48	%	Desjardins 2014 (microcirc rat)
Hematocrit	↓32	%	Desjardins 2014 (microcirc rat)
Tortuosity increase	↑60	%	Faber 2011 (pial vessels mouse)
Vessel lumen diameter	↓30	%	Faber 2011 (pial vessels mouse)
	↑7	%	Desjardins 2014 (rat microcirculation)
Angiogenic gene expression	↓	NA	Murugesan 2012 (capillaries mouse)
Micro-occlusions	~30	%	Wang 2012 (citations within, healthy aging)
	55	%	Wang 2012 (citations within, AD)

7.29.1 Physiological background to aging brain

An explanation could be decreased ability of aged brain to respond to hypercapnic challenges (due to stiffened vascular wall, exacerbated in hypertension) coupled with the decreased hematocrit lead to an increase in temporary hypoxia. This then leads to a breaking of the pericyte from local capillaries, which normally increases vascularization and allows growth of new branches. In the aged brain, however, where VEGF is not as readily available (or is not generated in response to hypoxia by the hypoxia-inducible-factor1, HIF-1 gene) in the extracellular space, so the loss of the pericyte does not trigger vascular growth but rather vascular death.

An alternative argument could be made that the decrease in pericytic density in old age could lead to a decreased ability to upkeep the vasculature, and this leads to a decrease in vasculature. Interestingly, Guan et al (2012) claimed that the astrocytes are the primary producers of VEGF, and their density (cells/mm³) remains constant throughout life [268].

The loose tie of NO to vascular remodeling is not substantiated, as the source study only found that after a ligation the rebuilding of the network was stronger in the wild type mice than the NO knockout mice. This could simply be due to the nature of NO in dilation and vascular upkeep, not a remark on the interaction of NO with the aging process. This is not the same as the firing NO, this would be a breakdown on endothelial NOS pathway, which signals for more things than just dilation.

The loss of NO induces drastic changes in O₂ perfusion which can be a post plaque-development symptom. This could initiate a cascade leading to diameter thinning and wall stiffening, which could lead to more micro-occlusions. These occlusions lead to preliminary hypoxic micro-pockets. These pockets lead to neuronal death, decreasing local NO activity due to neurons, also the de-structuring of cell wall can occur from this source. This then leads to dysfunctional NO production, and angiogenesis is hindered greatly.

In the adult brain, when a brain region suffers hypoxia (for a variety of reasons), the pericytes, which normally cover the endothelial layer, cannot interact with the endothelial receptor which maintains their relationship (Tie-2 receptor). When this receptor is now occupied by another hypoxia-induced protein, the pericyte moves away from the endothelial cell. At the same time, VEGF (vascular endothelial growth factor) is produced in HIF-1-producing cells and becomes available to the newly opened endothelial cell which then grows a new vessel. In aging, this process is considered less available and thus the alternate route (cell death) occurs more frequently and the tissue remains underperfused. This supports the previous statement, that the neuronal loss happens after an initial hypoxic condition which can be caused by microocclusions or other effects.

One paper cited many of the degenerative mechanisms in the cerebrovasculature arise from the cellular anatomical perspective [269]. One of the clear trends explained by this manuscript is the

age-related tortuosity inherent in the white matter and how these affect mostly the deep penetrating arterioles that reach through the grey matter and dive directly to the white matter for direct supply. There was a section dedicated to a discovery made by the same group involving white-matter veins becoming occluded by collagen and unable to pass flow. This then led to degradation of the vessels and eventually a decrease in vascular density.

It was noted that in AD patients, the level of A-beta in the CSF is lower, which was attributed to degradation of the perivascular clearance pathway. This was a stretch, as the reason the perivascular space is impeded was due to the thickening of the vascular walls and a decrease in mixing in the extravascular space. Moreover, it was noted that in AD, the A-beta proteins form plaques on the vascular walls which causes degradation of the endothelial layer.

The basement membrane of vascular walls is the most robust and the last to deteriorate after endothelial apoptosis (due to any number of reasons that causes the death of endothelial cells). It was noted that “string vessels” exist, and they are the remnants of old vessels which are in decline and only the basement membrane remains. Interestingly, these vessels look incredibly shriveled and can exist at any point along a vessel, creating a “pinched” spot in the vessel like those observed in medical images but are currently considered imaging defects. The generation of string vessels can be initiated by ischemia and takes 3-5 days before the degradation of the endothelial layer begins, and takes 8 days to become a “string vessel” which takes more than 40 days to fully clear. This condition (string vessels) occurs in normal brain but is upregulated in AD.

Normally, hypoxia triggers the transcription of hypoxia-inducible-factor1 (HIF-1) that creates VEGF (vascular endothelial growth factor). In normal, healthy brain capillaries, pericyte end-processes cover the endothelial cell layer and express angiopoietin-1 which activates endothelial Tie-2 receptors which cause a cascade that maintains the endothelial cell structure. In hypoxia,

angiopoietin-2 is created which occupies this receptor and prevents the expression of angiopoietin-1, causing pericytes to move away from the endothelial layer. At this point, there is a large source of VEGF in the vicinity of the endothelial layer ignites the growth tip (similar to a growth cone in an axon) which sprouts a new vessel, bringing blood into the hypoxic region, reperfusion it, and increasing local oxygen levels.

In the cases of aging brain or AD, plaque buildup or other factors lead to a lower availability of VEGF, leading to no growth. In fact, when the VEGF is not available and the endothelial cell is exposed, it leads to cell death. In fact, healthy aging noted a considerable reduction in the response availability of HIF-1 to hypoxia [ref 81-83]. It was noted that in middle-aged rats, chronic hypoxia led to a considerable increase in vascularization but that it does not occur in aged rats.

There was a clear correlation between old age and loss of capillary density which was exacerbated by AD. In these cases, the greater loss was attributed to A-beta attached to the endothelial wall and reducing availability of VEGF to the endothelial cell.

The majority of tortuosity increases exist in the deep penetrating arterioles that feed the white matter. A capillary tortuosity increase was also measured throughout the white matter. Tortuosity has also been correlated to severity of hypertension and has been speculated to be a feedback mechanism in response to increased flow.

The diameter shifts are best described by a thickening of the vessel wall, making diameter of the exterior of the vessel larger, but the interior diameter could be larger or smaller. Multiple groups have differing opinions on this number.

The increase in RBC speed can be attributed to either a higher volumetric flow rate at a diameter equal, smaller, or slightly larger than base diameter. It could also be attributed to the same

flow rate at a smaller diameter. If the overall CBF doesn't change much and the vessel density is lowered, the result will be an increased volumetric flow rate through the remaining capillaries.

Importantly, capillary density increases between young and middle-aged subjects, but the heavily aged subjects adhered to a degradation from mid-life and at some point become significantly lower than young animals. This is important, because the age of the “aged” group in studies will heavily affect the results under this paradigm.

7.29.2 Implementation

A few models of how to implement the changes in morphology of the aged brain are proposed:

- 1) Decrease systemic hematocrit: change in inlet hematocrit BC directly
 - a. 90% saturation in RBCs (backup plan)
 - b. Calculate saturation on vessel assuming a partial pressure (actual plan)

Option A is desired for initial calculations, as it is more easy to implement. Option B is more widely used in literature, but specifics on how this is executed are not divulged.

- 2) Loss of vessel density can follow two methodologies:
 - a. A new growth can be performed with less vessels (option A)
 - b. An existing network can have these vessels removed randomly (option B)

Option A implies a balanced tree with less vessels, where option B implies a sudden change in vasculature where the vessels are lost. Note, option B gives effectively the same result as a micro-occlusions, while option A is unique.

- 3) Micro-occlusions can be applied by changing the values of resistance to a very high number for a select number of vessels between model generation and matrix formulation steps

- a. This must only be applied to vessels at a bifurcation (not block the same vessel 3 different times), model A (current model, already implemented)
- b. Only vessels leaving a bifurcation in the capillary group can be blocked by erythrocytes, this requires grouping and bifurcation finding. This means that bifurcations on the larger arteries or veins cannot be blocked. Fundamentally different than capillary reduction, which can remove venous-side branches as well

Model A is the only model that makes sense. Both models require group identification, which is not implemented on the first round of growths.

- 4) Diameter changes will be implemented either in the case file or as a computational step after loading the network but prior to the model generation step

Also, implementation of debatable topological changes, probably save for rebuttal:

- 5) The increase in tortuosity (occurs solely in the white matter): manually shift the α parameter between the steps of model generation and matrix formation to match the linear correlation between length and tortuosity (namely, 10% increase in length translates to a 10% increase in α)

For best results (stable mesh convergence), the point-centered mass transfer coupling with edge detection (see Section 7.28.7.3 for more details). Also important for convergence at highly dense mesh structures (>200x200x200 mesh volumes for a KF dataset) it is recommended to use iterative mesh refinement (see interpolation section of the Cartesian mesh documentation).

7.30 Appendix AD: Proposal of a functional hyperemia model

Functional hyperemia is the active dilation of blood flow in response to local neurological stimulation. This section details models on how such a model could be implemented in the framework of 1D-3D mesh coupling.

7.30.1 Summary

The functional hyperemia problem can be solved as a series of linear algebraic problems. The simulation can be broken up into a dynamic simulation of NO distribution in the tissue (solved with a series of linear algebraic equations in which the right hand side is updated) followed by an evaluation of the network diameter change due to the NO concentration. This is followed by a single-phasic steady-state simulation of flow and oxygen.

The results show that a simplified (single-phase instead of Hill equation for oxygen binding kinetics), distributed, first-principle mechanistic model of the oxygen can reasonably approximate the oxygen in the brain.

Nitric Oxide. When investigating functional hyperemia, there needs to be a causal link between neuronal stimulation and vasomotion. The chosen model for vasodilatory signaling is nitric oxide (NO) as sourced from neuronal nitric oxide synthase (nNOS) produced by L-arginine. NO is produced in a firing column [270] and follows a 0th order generation model and a 1st order clearance model as in Equation (7.570) and (7.571).

$$\frac{dc_{NO}}{dt} V_{tiss} = \vec{\nabla} \cdot [D \vec{\nabla} c_{NO}(t)] V_{tiss} + k_{NO}^+(t) V_{tiss} - k_{NO}^- c_{NO}(t) V_{tiss} \quad (7.570)$$

Where

$$k_{NO}^+(t_{begin}^{firing} < t < t_{end}^{firing}) = \begin{cases} \overline{k(t)} & \text{if inside cylinder} \\ 0 & \text{else} \end{cases} \quad (7.571)$$

The effect of NO on vasculature follows a model inspired by the Secomb expansion model where dilatory forces are met with resistance from elastic forces. Our model uses a 20% maximum dilation over baseline and follows a 1st order reaction model:

$$\begin{aligned} \frac{dDia}{dt} &= dilationForces - tensiveForces \\ \frac{d}{dt} &= 0.2 \left(\frac{c_{NO}(t) - c_{ref}}{c_{NO}^{max}} \right) - \frac{d - d_0}{d_0} \end{aligned} \quad (7.572)$$

Where d is the diameter of the vessel and c_{ref} is the baseline NO concentration and c_{NO}^{max} is the maximum NO concentration in the region. The dilation effect from the NO signal is independently investigated:

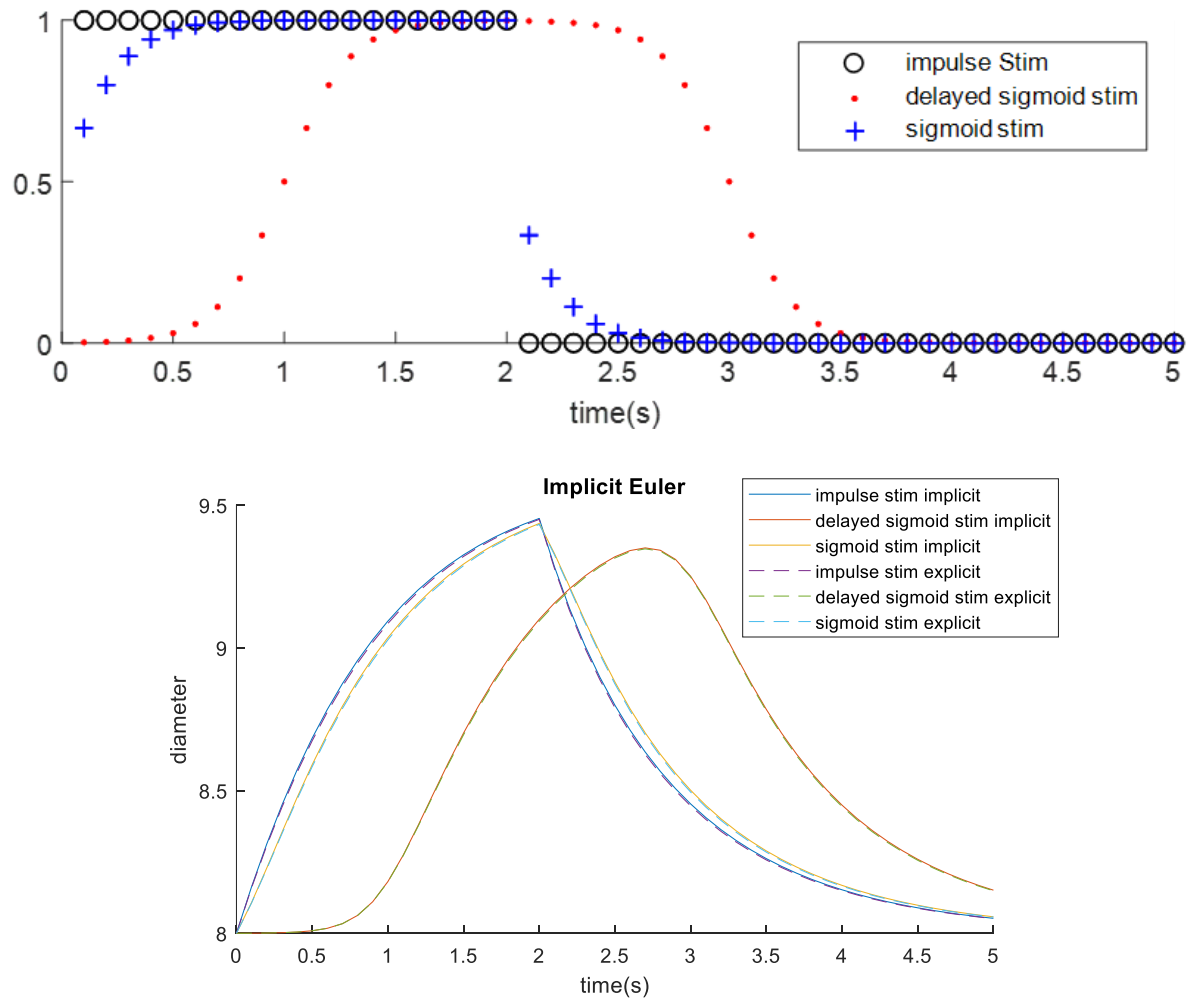


Figure 7.223. First order impulse response model as in Equation (7.572).
 Note, the response is not instantaneous, but takes time to overcome the elasticity of the membrane.

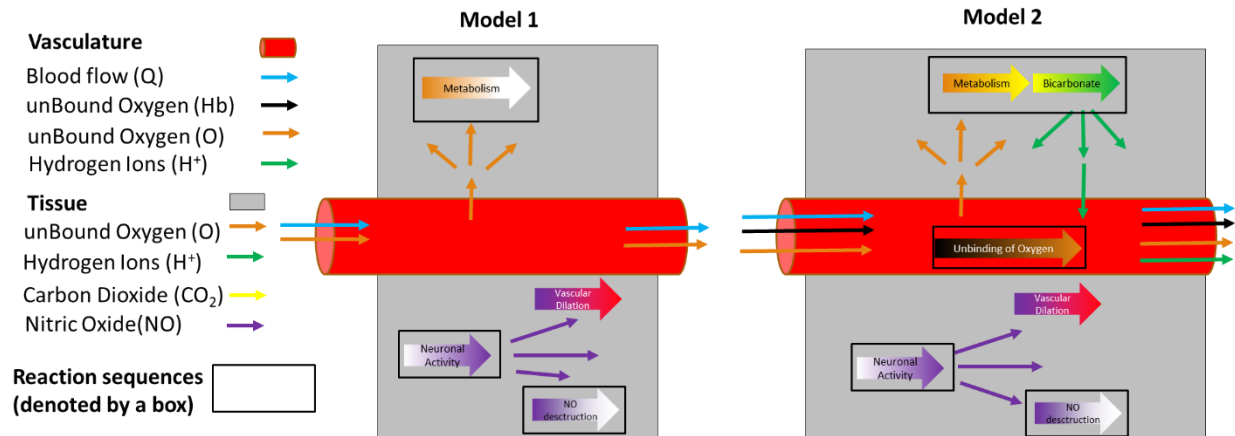


Figure 7.224. Schematic representation of the two models for functional hyperemia.

Method 1. The first method uses a simplified vascular oxygen transport model, where the nonlinear binding kinetics are neglected. The workflow for this is as follows:

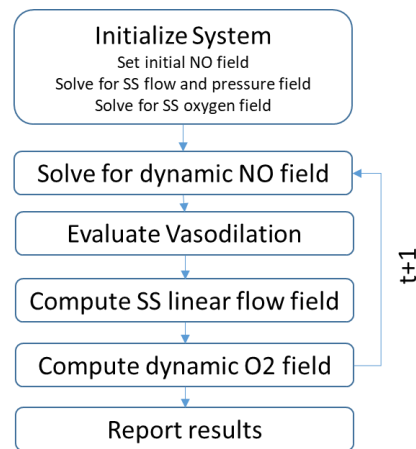


Figure 7.225. Workflow diagram for functional hyperemia for simplified vascular oxygen binding.

Method 2. The second method uses the hill equation to approximate the binding kinetics of vascular oxygen saturation. The workflow for this is as follows:

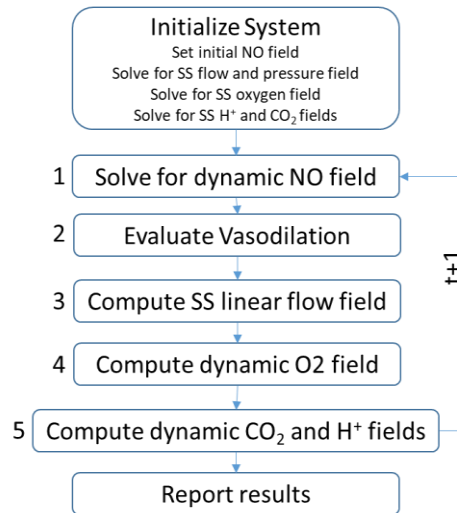


Figure 7.226. Workflow diagram for functional hyperemia for simplified vascular oxygen binding.

7.30.2 Strategy

In this chapter, the reactions in the tissue will generate a species (nitric oxide, NO, representing the neurotransmitter released during repeated neuronal firing) that will diffuse through the tissue. This species will be reacted away through the tissue at a constant rate (representing microglial and astrocytic cleaning of the extracellular matrix). This will interact with the blood vessels, causing an increase in local diameter (vasodilation). This dilation will also be transmitted upstream and downstream at a fixed rate.

The implementation will include four solving steps that follow the equations listed below. The diameter increase is based on the maximum diameter expansion of 20%, consistent with previous experimental findings [236].

The generation is a 0th order reaction and the reaction away of NO is a 1st order reaction.

1) Tissue NO concentration (dynamic)

$$\frac{dc_{NO}}{dt} V_{tiss} = \vec{\nabla} \cdot [D \vec{\nabla} c_{NO}(t)] V_{tiss} + k_{NO}^+(t) V_{tiss} - k_{NO}^- c_{NO}(t) V_{tiss}$$

Where

$$k_{NO}^+(t) = \begin{cases} \overline{k(t)} & \text{if inside cylinder,} \\ 0 & \text{else} \end{cases}, \quad k_{NO}^- = \overline{k^-} \quad (7.573)$$

And

$$\overline{k(t)} = \begin{cases} \overline{k} & 10 < t < 20 \\ 0 & \text{else} \end{cases}$$

2) Vasculature diameter response (evaluate, assuming SS):

$$d = d_0 + 0.2 \left(\frac{c_{NO}(t) - c_{ref}}{c_{NO}^{max}} \right) d_0 \quad (7.574)$$

3) vasculature Linear Flow (SS):

$$0 = \vec{\nabla} \cdot \left(\frac{1}{\alpha} \Delta p \right), \quad f = \frac{1}{\alpha} \Delta p, \quad p = \overline{p} \quad (7.575)$$

4) In tissue concentration (SS):

$$0 = \vec{\nabla} \cdot (D \vec{\nabla} c_A) V_{tiss} + U A_{vasc} \Delta c_A - k_1 c_A V_{tiss} \quad f c_A = 0 \quad (7.576)$$

In vasculature (SS):

$$0 = \vec{\nabla} \cdot (f c_A) - U A_{vasc} \Delta c_A \quad c_{O2} = \overline{c_{O2}} \quad (7.577)$$

7.30.3 Implementation

The building of dynamic equations and solving them iteratively has already been implemented for an implicit Euler method. The mathematics of the implicit Euler can be reduced to a single set of linear algebraic equations with an updating right hand side vector, as shown below.

Implicit Euler ($\dot{c} = Ac$):

$$\frac{c^N - c^o}{\Delta t} = Ac^N \quad (7.578)$$

$$c^N - c^o = \Delta t Ac^N$$

$$(I - \Delta t A)c^N = c^o$$

$$A \quad x = b$$

Note, all algebraic and algebraic-differential systems of equations will be solved using an Implicit Euler scheme. This was compared to a second order Adams Bashford method, 4th, and 5th order Runge Kutta methods all of which required significantly more timesteps for timestep independence (see Section 7.31). The libraries of PETSc linear algebraic solvers is employed using the generalized minimal residual method, GMRES, and a block Jacobi preconditioner as described in Section 7.20.6.

7.30.4 Dynamically solving a steady problem

7.30.4.1 Validation

The first validation of the dynamic Implicit Euler solver is to ensure a system already at steady-state can be integrated in time (without stimulus) and remain stationary. This will ensure the

integration method does not, itself, modify the result. Note, the steady state simulation is described in Section 7.28.7. This can be accomplished by setting the reaction kinetics to 0, the initialization of NO is set to 0.1 everywhere with flux boundary conditions and the oxygen model is solved at steady-state in each time point. The results are summarized below. The solver is a GMRES with PETSc block Jacobi preconditioning and a tolerance set to 1e-7.

In vasculature Linear Flow:

$$0 = \vec{\nabla} \cdot \left(\frac{1}{\alpha} \Delta p \right), \quad f = \frac{1}{\alpha} \Delta p, \quad p = \bar{p} \quad (7.579)$$

In tissue concentration:

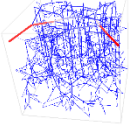
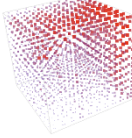
$$0 = \vec{\nabla} \cdot (D \nabla c_A) V_{tiss} + UA_{vasc} \Delta c_A V_{tiss} - k_1 c_A V_{tiss} \quad (7.580)$$

In vasculature:

$$0 = \vec{\nabla} \cdot (f c_A) V_{vasc} - UA_{vasc} \Delta c_A V_{vasc} \quad (7.581)$$

Boundary conditions:

$$f c_A = 0, \quad c_A = \bar{c}_A \quad (7.582)$$

Name	#	Convection flow (mol/min)	Solute flux (mol/ min)	Concentratio n range (mol)	Max norm of residual (mol)
ArtVenFused oXLibraries	M_LF_41		1.01E6 – 1.02E9	430.4 - 1000	2.6E-6
ArtVenFused Mesh 0 th order oXLibraries	M_LF_42		4.5e-13 – 7.9E5	143.2 – 701.4	

7.30.5 Dynamic solving of constant NO generation, vasculature unchanged

The next step is to evaluate the dynamic solving of a generation/diffusion problem over time. To evaluate the solver, mesh independence and time independence will be achieved. The solver chosen in GMRES binary with 3e-6 tolerance.

$$\frac{dc_{NO}}{dt} V_{tiss} = \vec{\nabla} \cdot [D \vec{\nabla} c_{NO}(t)] V_{tiss} + k_{NO}^+(t) V_{tiss} - k_{NO}^- c_{NO}(t) V_{tiss} \quad (7.583)$$

Where

$$k_{NO}^+(t) = \begin{cases} \overline{k(t)} & \text{if inside cylinder,} \\ 0 & \text{else} \end{cases}, \quad k_{NO}^- = \overline{k^-}$$

In vasculature Linear Flow:

$$0 = \vec{\nabla} \cdot \left(\frac{1}{\alpha} \Delta p \right), \quad f = \frac{1}{\alpha} \Delta p, \quad p = \bar{p} \quad (7.584)$$

In tissue O₂ concentration:

$$0 = \vec{\nabla} \cdot (D \vec{\nabla} c_A) V_{tiss} + UA_{vasc} \Delta c_A - k_1 c_A V_{tiss} \quad (7.585)$$

In vasculature O₂:

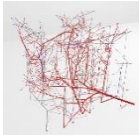
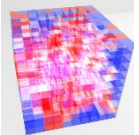
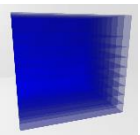
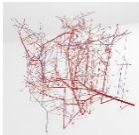
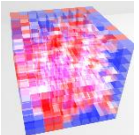
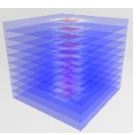
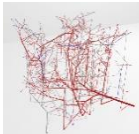
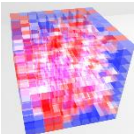
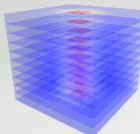
$$0 = \vec{\nabla} \cdot (f c_A) - UA_{vasc} \Delta c_A \quad (7.586)$$

Boundary conditions:

$$f c_A = 0, \quad c_A = \overline{c_A} \quad (7.587)$$

$$k_{NO}^- = 1 \frac{mol}{s} \quad k_{NO}^+ = 5 \frac{mol}{s} \quad D_{NO} = 0.001 \frac{m}{s} \quad D_{O_2} = 0.001 \frac{M}{s},$$

$$k_1 = 5$$

	#	Visualizati on network O ₂	Visualizati on mesh O ₂	Visualizati on mesh NO	Concentrati on Range O ₂ NWK (mol)	Concentrati on range O ₂ mesh (mol/mL)	Concentrati on range NO mesh (mol/mL)	Max residu al
t=0	M_LF_ 41				993.8 - 1000	7-999.4	0.1-0.1 (initial condition)	9.0E-8
t=0. 1	M_LF_ 42				993.8 - 1000	7-999.4	0.05 – 2.04	9.0E-8
t=0. 2					993.8 - 1000	7-999.4	0.025 – 2.82	9.0E-8

7.30.6 Dynamic solving of 0.04s impulse, 1 second simulation, NO generation, vasculature unchanged

The simulation engine will now be validated against an impulse response (from t= 0 – 0.4 seconds). In this case, the simulation correctly shows a rise in concentration followed by a fall in concentration emanating from the cylindrical source. Number of volume elements in each dimension is 15.

$$\frac{dc_{NO}}{dt}V_{tiss} = \vec{\nabla} \cdot [D\vec{\nabla}c_{NO}(t)]V_{tiss} + k_{NO}^+(t)V_{tiss} - k_{NO}^-c_{NO}(t)V_{tiss}$$

Where (7.588)

$$k_{NO}^+(t) = \begin{cases} \overline{k(t)} & \text{if inside cylinder,} \\ 0 & \text{else} \end{cases}, \quad k_{NO}^- = \overline{k^-}$$

In vasculature Linear Flow:

$$0 = \vec{\nabla} \cdot \left(\frac{1}{\alpha} \Delta p \right), \quad f = \frac{1}{\alpha} \Delta p, \quad p = \overline{p} \quad (7.589)$$

In tissue O₂ concentration:

$$0 = \vec{\nabla} \cdot (D\nabla c_A)V_{tiss} + UA_{vasc}\Delta c_A - k_1c_AV_{tiss} \quad (7.590)$$

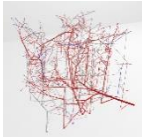
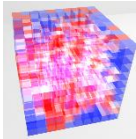
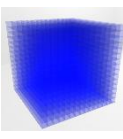

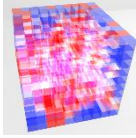
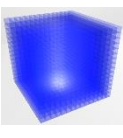
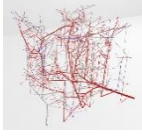
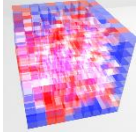
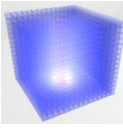
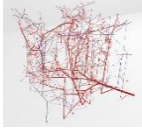
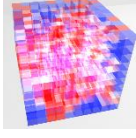
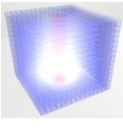

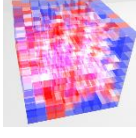
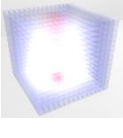

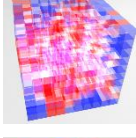
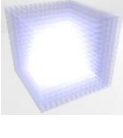

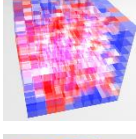
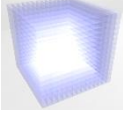

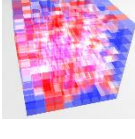
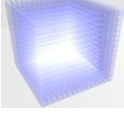
In vasculature O₂:

$$0 = \vec{\nabla} \cdot (fc_A) - UA_{vasc}\Delta c_A \quad (7.591)$$

Boundary conditions:

$$fc_A = 0, \quad c_A = \overline{c_A} \quad (7.592)$$

$$k_{NO}^- = 0.1 \frac{mol}{s} \quad k_{NO}^+ = 10 \frac{mol}{s} \quad D_{NO} = 0.1 \frac{m}{s} \quad D_{O_2} = 0.001 \frac{M}{s}, k_1 = 5$$

T	Visualization network O ₂	Visualization mesh O ₂	Visualization on mesh NO	Concentration Range O ₂ NWK (mol)	Concentration range O ₂ mesh (mol/mL)	Concentration range NO mesh (mol/mL)
0				993.8 - 1000	7-999.4	0-0 (initial condition)
0.1				993.8 - 1000	7-999.4	0.05 – 0.42
0.2				993.8 - 1000	7-999.4	0.13 – 0.55
0.3				993.8 - 1000	7-999.4	0.21 – 0.64
0.4				993.8 - 1000	7-999.4	0.28 – 0.72
0.5				993.8 - 1000	7-999.4	0.3 – 0.36
0.6				993.8 - 1000	7-999.4	0.29 – 0.30
0.7				993.8 - 1000	7-999.4	0.26 – 0.27

7.30.7 Dynamic solving of impulse NO generation, vasculature expansion

The vascular response to NO concentration must be meaningful. To model the vasodilatory signal, a previous publication reported a maximum of 20% dilation in vessels in response to neuronal firing [271]. The model chosen to represent this is a linear model of dilation in response to NO concentration as in Equation (7.594).

This simulation should exhibit a steady rise in concentration of NO over the course of the impulse and a fall after the impulse ends. The vasculature should exhibit a time-lag vasodilation that follows a linear response to the NO generation. The oxygen model should reflect a baseline oxygen and an increase in oxygen after the vasodilation, to be returned to baseline after the vasodilation withers. C_{NO}^{max} is chosen from largest value of NO at time point 0.04 seconds. These aspects of the simulation are reflected in the results, indicating the mechanisms are working properly.

$$\frac{dc_{NO}}{dt} V_{tiss} = \vec{\nabla} \cdot [D \vec{\nabla} c_{NO}(t)] V_{tiss} + k_{NO}^+(t) V_{tiss} - k_{NO}^- c_{NO}(t) V_{tiss}$$

Where (7.593)

$$k_{NO}^+(t) = \begin{cases} \overline{k(t)} & \text{if inside cylinder,} \\ 0 & \text{else} \end{cases}, \quad k_{NO}^- = \overline{k^-}$$

Vasculature diameter response (evaluate, assuming SS):

$$d = d_0 + 0.2 \left(\frac{c_{NO}(t) - c_{ref}}{c_{NO}^{max}} \right) d_0 \quad (7.594)$$

In vasculature Linear Flow:

$$0 = \vec{\nabla} \cdot \left(\frac{1}{\alpha} \Delta p \right), \quad f = \frac{1}{\alpha} \Delta p, \quad p = \bar{p} \quad (7.595)$$

In tissue O₂ concentration:

$$0 = \vec{\nabla} \cdot (D \nabla c_A) V_{tiss} + U A_{vasc} \Delta c_A - k_1 c_A V_{tiss} \quad (7.596)$$

In vasculature O₂:

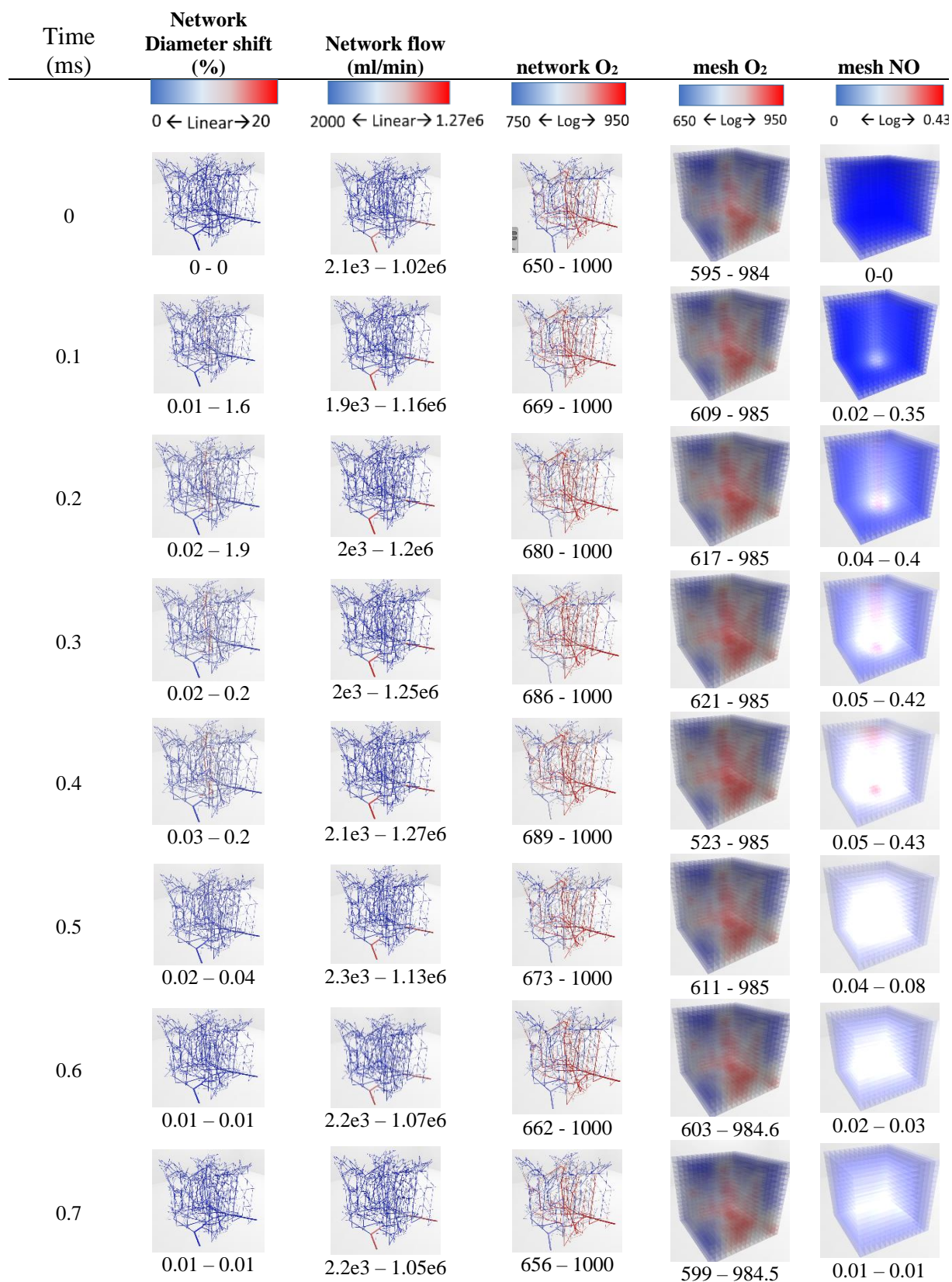
$$0 = \vec{\nabla} \cdot (f c_A) - U A_{vasc} \Delta c_A \quad (7.597)$$

Boundary conditions:

$$f c_A = 0, \quad c_A = \bar{c}_A \quad (7.598)$$

$$k_{NO}^- = 0.5 \frac{mol}{s} \quad k_{NO}^+ = 10 \frac{mol}{s} \quad D_{NO} = 0.1 \frac{m}{s} \quad D_{O_2} = 0.001 \frac{M}{s}, k_1 = 5$$

$$c_{ref} = 0; \quad c_{NO}^{max} = 0.4266$$



7.30.8 Dynamic solving of impulse NO generation, vasculature signaling

The signaling mechanism that causes upstream vessels to dilate can be modeled as a constant-rate signaling mechanism as observed experimentally [271]. The values chosen from literature for upstream dilation are 12.65 $\mu\text{m/s}$ and downstream signaling is 12.83 $\mu\text{m/s}$.

This simulation should exhibit a steady rise in concentration of NO over the course of the impulse and a fall after the impulse ends. The vasculature should exhibit a time-lag vasodilation that follows a linear response to the NO generation. The oxygen model should reflect a baseline oxygen and an increase in oxygen after the vasodilation, to be returned to baseline after the vasodilation withers. C_{NO}^{max} is chosen from largest value of NO at time point 0.04 seconds. This model has been proposed but not yet implemented.

$$\frac{dc_{NO}}{dt}V_{tiss} = \vec{\nabla} \cdot [D\vec{\nabla}c_{NO}(t)]V_{tiss} + k_{NO}^+(t)V_{tiss} - k_{NO}^-c_{NO}(t)V_{tiss}$$

Where (7.599)

$$k_{NO}^+(t) = \begin{cases} \overline{k(t)} & \text{if inside cylinder,} \\ 0 & \text{else} \end{cases}, \quad k_{NO}^- = \overline{k^-}$$

Vasculature diameter response (evaluate, assuming SS):

Direct effect of NO:

$$\text{expansion} = \phi = 0.2 \left(\frac{c_{NO}(t) - c_{ref}}{c_{NO}^{\text{max}}} \right) \quad (7.600)$$

$$d = d_0 + \phi d_0$$

Signaling:

$$\phi_{up}(x, t) = \phi$$

In vasculature Linear Flow:

$$0 = \vec{\nabla} \cdot \left(\frac{1}{\alpha} \Delta p \right), \quad f = \frac{1}{\alpha} \Delta p, \quad p = \bar{p} \quad (7.601)$$

In tissue O₂ concentration:

$$0 = \vec{\nabla} \cdot (D \vec{\nabla} c_A) V_{tiss} + U A_{vasc} \Delta c_A - k_1 c_A V_{tiss} \quad (7.602)$$

In vasculature O₂:

$$0 = \vec{\nabla} \cdot (f c_A) - U A_{vasc} \Delta c_A \quad (7.603)$$

Boundary conditions:

$$f c_A = 0, \quad c_A = \bar{c}_A \quad (7.604)$$

$$k_{NO}^- = 0.5 \frac{mol}{s} \quad k_{NO}^+ = 10 \frac{mol}{s} \quad D_{NO} = 0.1 \frac{m}{s} \quad D_{O_2} = 0.001 \frac{M}{s}, k_1 = 5$$

$$c_{ref} = 0; \quad c_{NO}^{max} = 0.4266$$

7.30.9 Dynamic biphasic blood flow

This simulation follows Section 7.30.5 except with replacing the linear blood flow model with a biphasic blood flow model. This mechanism ties the oxygen convection to the RBC phase of the blood flow. All results have been validated for reasonable flow balance error and residual errors (table not shown).

$$\frac{dc_{NO}}{dt} V_{tiss} = \vec{\nabla} \cdot [D \vec{\nabla} c_{NO}(t)] V_{tiss} + k_{NO}^+(t) V_{tiss} - k_{NO}^- c_{NO}(t) V_{tiss} \quad (7.605)$$

Where

$$k_{NO}^+(t) = \begin{cases} \overline{k(t)} & \text{if inside cylinder,} \\ 0 & \text{else} \end{cases}, \quad k_{NO}^- = \overline{k^-}$$

Vasculature diameter response (evaluate, assuming SS):

Direct effect of NO:

$$\begin{aligned} \text{expansion} = \phi &= 0.2 \left(\frac{c_{NO}(t) - c_{ref}}{c_{NO}^{max}} \right) \\ d &= d_0 + \phi d_0 \end{aligned} \quad (7.606)$$

Signaling:

$$\phi_{up}(x, t) = \phi$$

In vasculature Biphasic Blood Flow:

$$G(Q, p, h) = 0 \begin{cases} R(h, d)Q - C_1 p & = 0 \\ C_2 Q & = 0 \\ C_3(Q, d)h & = 0 \end{cases} \quad (7.607)$$

In tissue O₂ concentration:

$$0 = \vec{\nabla} \cdot (D \nabla c_A) V_{tiss} + U A_{vasc} \Delta c_A - k_1 c_A V_{tiss} \quad (7.608)$$

In vasculature O₂:

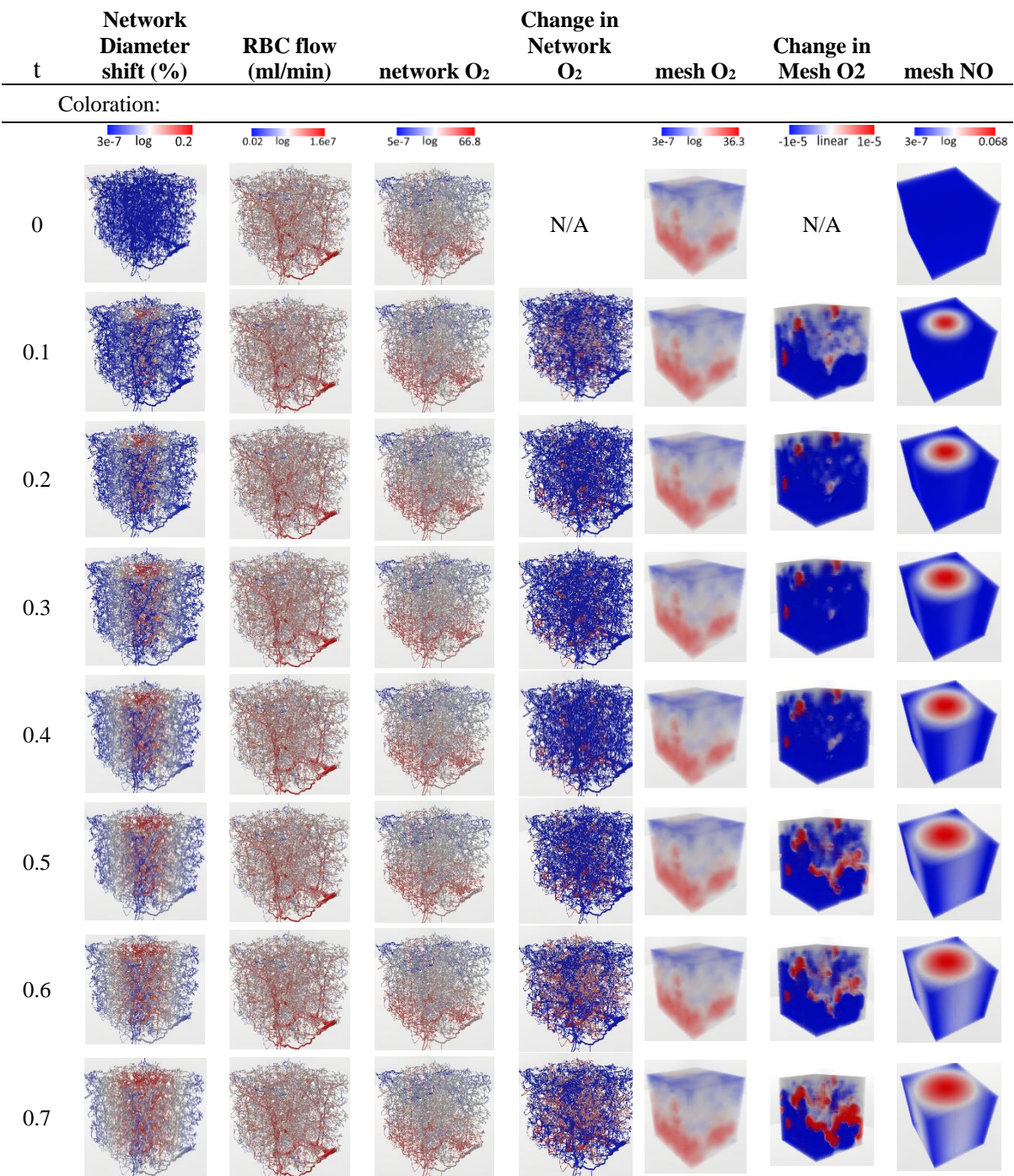
$$0 = \vec{\nabla} \cdot (f c_A) - U A_{vasc} \Delta c_A \quad (7.609)$$

Boundary conditions:

$$f c_A = 0, \quad c_A = \overline{c_A} \quad (7.610)$$

$$k_{NO}^- = 0.5 \frac{mol}{s} \quad k_{NO}^+ = 10 \frac{mol}{s} \quad D_{NO} = 0.1 \frac{m}{s} \quad D_{O_2} = 0.001 \frac{M}{s}, k_1 = 5$$

$$c_{ref} = 0; \quad c_{NO}^{max} = 0.4266$$



7.31 Appendix AE: Time integration solvers

7.31.1 Introduction

Some problems in transport phenomena are formulated as large systems of coupled partial differential equations (PDEs). To solve the spatial discretization, finite difference (FEA) or finite volume (FVM) methods have been used. In some cases, such as in functional hyperemia in the biomedical industry, these dynamic systems of equations must be integrated in time. In this example, relevant fields for integration include nitric oxide (NO) in the tissue, vessel dilation (as a function of local NO concentration), blood flow, and coupled convection-mass-transfer-diffusion-reaction system of oxygen. Using a simple implicit Euler time integration leads to a very small timestep ($<10^{-5}$ s) and can take excessively long to simulate for physiologically meaningful durations (10^1 s). With this gap in solving time, the solvers will take >500 days to solve the entire simulation.

To improve solving time, an explicit Euler method can be employed such that there is no need to solve the linear algebraic set of equations in each time step. Unfortunately, a first order explicit Euler does not hold to mass conservation without an extremely small timestep (10^{-8}). A second order Adams-Bashford, 4th and 5th order Runge-Kutta methods are possibilities to improve solving time. Below find a case studies showing that the Adams-Bashford (AB) method can be up to a single order of magnitude better than the first order implicit Euler (IE).

These more advanced methods use a linear averaging (convolution between the current timestep evaluation (y^n) and the previous timestep and, in some cases, the update guess (y^{n+1}) as in the case of RK4 and RK5. When applying these solvers to the full oxygen simulation at the

scale of a microcirculatory block in the mouse brain, the implicit Euler maintains the most robust and efficient solver.

7.31.2 Methods

Integration methods investigated include implicit Euler (IE, Equation (7.611)), 2nd order explicit Adams-Bashford (AB, Equation (7.612)), explicit 4th order Runge-Kutta (RK4, Equation (7.613)), and explicit 5th order Runge-Kutta (RK5, Equation (7.614)). The RK4 and RK5 methods are delineated nicely in a publication by Christodoulou [272] with a sample code for implementation in Matlab. Supporting Matlab codes for this example can be found in Section 7.31.4. In these equations, h is the timestep size, y is the solution vector, A is the coefficient matrix, and I is the identity matrix. In Equation (7.618), the implicit Euler equation has been simplified to a linear set of equations that can be solved in every timestep by merely updating the right hand side vector.

$$\begin{aligned} y^{n+1} &= y^n + h(A \cdot y^{n+1}) \\ (I - hA)y^{n+1} &= y^n \end{aligned} \tag{7.611}$$

$$y^{n+1} = y^n + \frac{3}{2}h(A \cdot y^n) - \frac{1}{2}h(A \cdot y^{n-1}) \tag{7.612}$$

$$y^{n+1} = y^n + \frac{k_1 + 2k_2 + 2k_3 + k_4}{6}$$

$$k_1 = hA \cdot y^n \quad k_2 = hA \cdot \left(y^n + \frac{k_1}{2}\right) \tag{7.613}$$

$$k_3 = hA \cdot \left(y^n + \frac{k_2}{2}\right) \quad k_4 = hA \cdot (y^n + k_3)$$

$$\begin{aligned}
y^{n+1} &= y^n + \frac{7k_1 + 32k_3 + 12k_4 + 32k_5 + 7k_6}{90} \\
k_1 &= hA \cdot y^n & k_2 &= hA \cdot \left(y^n + \frac{k_1}{2}\right) \\
k_3 &= hA \cdot \left(y^n + \frac{3k_1 + k_2}{16}\right) & k_4 &= hA \cdot \left(y^n + \frac{k_3}{2}\right) \\
k_5 &= hA \cdot \left(y^n + \frac{3k_2 + 6k_3 + 9k_4}{16}\right) \\
k_6 &= hA \cdot \left(y^n + \frac{k_1 + 4k_2 + 6k_3 - 12k_4 + 8k_5}{7}\right)
\end{aligned} \tag{7.614}$$

Note that the AB method requires 2 initial steps in order to integrate. To solve for the first time step, the system is initialized by integrating twice with the IE method. In more elaborate simulations where the dynamic changes are in response to a stimulus, (specifically inhomogeneous steady-state systems, or steady-state systems that are perturbed) integrator should be initialized using the steady-state solution to the system.

7.31.3 Case study 1: cyclic reaction system

The chemical reaction system pictured in Figure 7.227 can be written in component form as in Equations (7.615)-(7.617). This can also be written in Matrix form using Equation (7.618) and (7.619).

$$\frac{dA}{dt} = -(k_{AB} + k_{ca})C_A + k_{ba}C_B + k_{CA}C_C \tag{7.615}$$

$$\frac{dB}{dt} = k_{AB}C_A - (k_{ba} + k_{BC})C_B + k_{cb}C_C \tag{7.616}$$

$$\frac{dC}{dt} = k_{ac}C_A + k_{BC}C_B - (k_{CA} + k_{cb})C_C \quad (7.617)$$

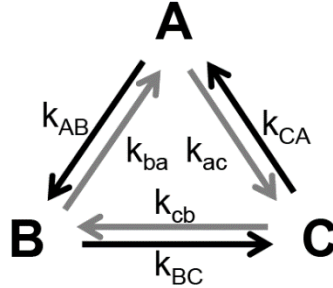


Figure 7.227. The chemical reaction system of A,B and C species used for this case study.

$$\begin{pmatrix} \frac{dA}{dt} \\ \frac{dB}{dt} \\ \frac{dC}{dt} \end{pmatrix} = \begin{bmatrix} -(k_{AB} + k_{ca}) & k_{ba} & k_{CA} \\ k_{AB} & -(k_{ba} + k_{BC}) & k_{cb} \\ k_{ac} & k_{BC} & -(k_{CA} + k_{cb}) \end{bmatrix} \begin{pmatrix} C_A \\ C_B \\ C_C \end{pmatrix} \quad (7.618)$$

$$\dot{Y} = AY \quad (7.619)$$

The time integration did not substantially differ between the four methods investigated (attributed to how short the solving time was for all cases). The accuracy, however, was enhanced between the AB, RK4, and RK5 in order as seen in Figure 7.228. Figure 7.228 reflects the time integration of species A from initial concentration fraction of 0.5.

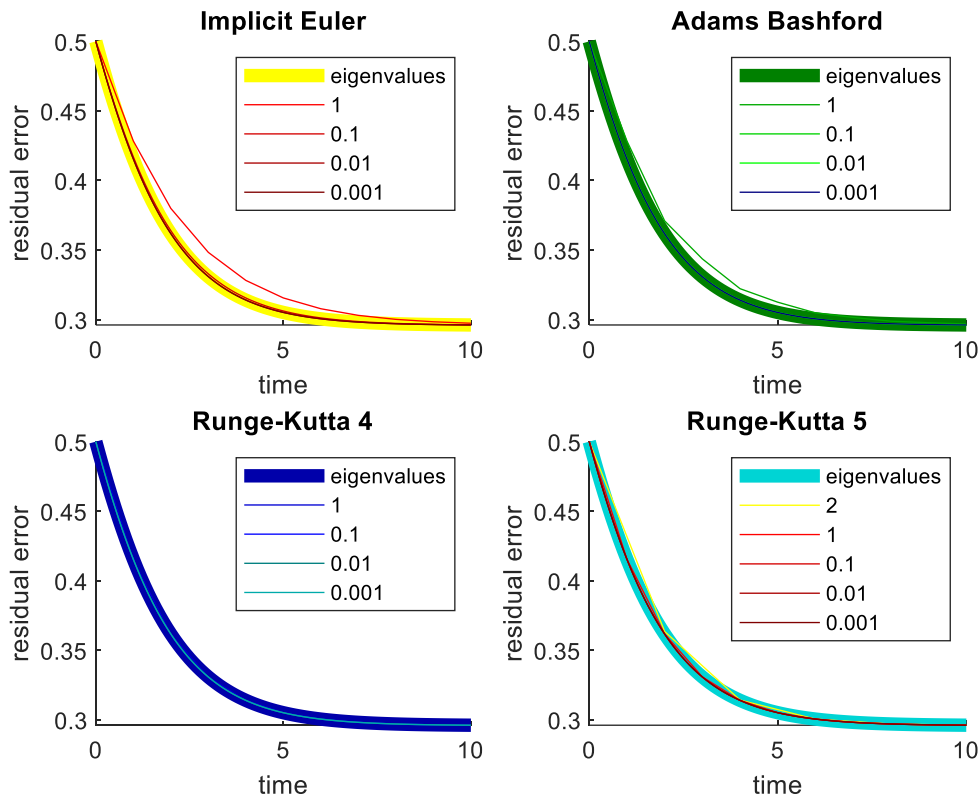


Figure 7.228. Comparison between Top Left) Implicit Euler (IE), Top Right) Adams Bashford (AB), Bottom Left) Runge-Kutta 4, and Bottom Right) Runge-Kutta 5 integrators at different timestep sizes.

The methods from best-to-worst are: RK5, RK4, AB, and finally IE. Note, RK5 was the only integrator that produced reasonable results at a timestep of $h=2$.

The different integration schemes were also compared using the maximum residual error measured at 1s intervals starting at time $t=0$ s. Among the concentrations at these time points, the error is measured as the maximum difference between the partial concentration of the analytic solution (derived from eigenvalue decomposition) and the numerical result. The comparison has been tabulated in Table 7.64. The performance of each solver is offered in Table 7.65 measured as time to solve the system.

Table 7.64. Comparative error between numerical integrators.

	h=1	h=0.1	h=0.01	h=0.001
IE	48605e-6	5930e-6	612e-6	61.357e-6
AB	45948e-6	917.9e-6	9.83e-6	9.9e-8
RK4	824.0e-6	4.558e-8	4.300e-12	1.554e-15
RK5	85.96e-6	4.81e-10	4.52e-15	1.11e-15

Table 7.65. Comparative performance between numerical integrators.

	h=1	h=0.1	h=0.01	h=0.001
IE	0.00136s	0.00114s	0.00531s	0.03209s
AB	0.00144s	0.00025s	0.00255s	0.01151s
RK4	0.00431s	0.00064s	0.00551s	0.02984s
RK5	0.00014s	0.00080s	0.00389s	0.03877s

With simple ODE systems, AB can be a more accurate integrator. Moreover, on large systems with many unknowns, an explicit solver is significantly faster than an implicit solver. These solvers were all implemented in a Delphi program and investigated for dynamic oxygen simulations yet the timestep requirements did not reflect the case study from this report. In the full implementation, a finer timestep was required for AB, RK4 and RK5. Due to this, IE remained the most efficient and robust solver.

7.31.4 Matlab implementation of different solvers on the case study

```

function testIntegrators
warning('off')
close all
A = getA();
Yinitial = getY0();
global colormp idx
nColorChoices = 4;
colormp1=linspace(1,0.5,nColorChoices); colormp2=linspace(0.5,1,nColorChoices); colormp3=linspace(0.5,1,nColorChoices);
colormp = [1,1,0;
    colormp1' zeros(nColorChoices,2);
    zeros(nColorChoices,1) colormp2' zeros(nColorChoices,1);
    zeros(nColorChoices,2) colormp3'
    zeros(nColorChoices,1) colormp2' colormp3';...
    colormp3' zeros(nColorChoices,1) colormp2']; idx = 1;

%added 9/12/2018
figure(7), subplot(2,2,1), hold on,
solveUsingEigenvalues(A,Yinitial(:,6)); solveUsingImplicitEulerMethod(A,Yinitial(:,6));
title('Implicit Euler'); ylabel('residual error'); xlabel('time'); subplot(2,2,2), hold on,
solveUsingEigenvalues(A,Yinitial(:,6)); solveUsingAdamsBashforth(A,Yinitial(:,6));
title('Adams Bashford'); ylabel('residual error'); xlabel('time'); subplot(2,2,3), hold on,
solveUsingEigenvalues(A,Yinitial(:,6)); solveUsing4THOrderRK(A,Yinitial(:,6));
title('Runge-Kutta 4'); ylabel('residual error'); xlabel('time'); subplot(2,2,4), hold on,
solveUsingEigenvalues(A,Yinitial(:,6)); solveUsing5THOrderRK(A,Yinitial(:,6));
title('Runge-Kutta 5'); ylabel('residual error'); xlabel('time');

end

% this is a new method by GH 9/12/2018 to use a multi-step adam's mashforth method to solve the ode
function Y = solveUsingAdamsBashforth(A,Y0)
disp('AB start:')
global colormp idx eigSolnAtS
stepsize = [1,1,0.01,.001]; leg = {'eigenvalues'};
for iStepsize = 1:length(stepsize)
    t = 0:stepsize(iStepsize):10;
    n = length(t);
    h = stepsize(iStepsize);
    Y(:,1) = Y0;
    Y(:,2) = (eye(3)-h*A)\Y(:,1); %implicit euler for t2
    tic,
    for loop=3:n
        Y(:,loop) = Y(:,loop-1) + 3/2*h*(A*Y(:,loop-1))-1/2*h*(A*Y(:,loop-2));
    end
    toc
    Y = Y./sum(Y(:,1)); %normalize the data
    error = eigSolnAtS - Y(:,1:1/stepsize(iStepsize):length(t)); disp(['max resid:' num2str(max(max((error))))]);
    figure(7), plot(t,Y(1,:), 'color',colormp(idx,:)); idx = idx+1;
    leg[iStepsize+1] = num2str(stepsize(iStepsize));
end
legend(leg)
end

% GH 9/14/2018 -- adding a 4th order runga kutta for evaluation
function Y = solveUsing4THOrderRK(A,Y0)
global colormp idx eigSolnAtS
disp('RK4 start:')
stepsize = [1,1,0.01,.001]; leg = {'eigenvalues'};
for iStepsize = 1:length(stepsize)
    % t = 0.1:stepsize(iStepsize):100;
    t = 0:stepsize(iStepsize):10;
    n = length(t);
    h = stepsize(iStepsize);
    Y(:,1) = Y0;
    tic,

```

```

for loop=2:n
    k1 = h*A*Y(:,loop-1);
    k2 = h*A*(Y(:,loop-1)+0.5*k1);
    k3 = h*A*(Y(:,loop-1)+0.5*k2);
    k4 = h*A*(Y(:,loop-1)+k3);
    Y(:,loop) = Y(:,loop-1) + (k1+2*k2+2*k3+k4)/6;
end
toc
Y = Y./sum(Y(:,1)); %normalize the data
error = eigSolnAtS - Y(:,1:1/stepsize(iStepsize):length(t)); disp(['max resid:' num2str(max(max((error))))]);
figure(7), plot(t,Y(1,:), 'color', colormp(idx,:)); idx = idx+1;
leg{iStepsize+1} = num2str(stepsize(iStepsize));
end
legend(leg)
end

% 5th order RK as described in:
% AN ALGORITHM USING RUNGE-KUTTA METHODS OF ORDERS 4 AND 5 FOR SYSTEMS OF ODEs by NIKOLAOS S.
CHRISTODOULOU
% http://www.teihal.gr/gen/profesor/christodoulou/Publications/An%20algorithm%20using.pdf
function Y = solveUsing5THOrderRK(A,Y0)
global colormp idx eigSolnAtS
disp('RK5 start:')
stepsize = [2,1,1,0.01,.001]; idx = 1; leg = {'eigenvalues'};
for iStepsize = 1:length(stepsize)
    % t = 0.1:stepsize(iStepsize):100;
    t = 0:stepsize(iStepsize):10;
    n = length(t);
    h = stepsize(iStepsize);
    Y(:,1) = Y0;
    tic,
    for loop=2:n
        k1 = h*A*Y(:,loop-1);
        k2 = h*A*(Y(:,loop-1)+0.5*k1);
        k3 = h*A*(Y(:,loop-1)+(3*k1+k2)/16);
        k4 = h*A*(Y(:,loop-1)+k3/2);
        k5 = h*A*(Y(:,loop-1)+(-3*k2+6*k3+9*k4)/16);
        k6 = h*A*(Y(:,loop-1)+(k1+4*k2+6*k3-12*k4+8*k5)/7);
        Y(:,loop) = Y(:,loop-1) + (7*k1+32*k3+12*k4+32*k5+7*k6)/90;
    end
    toc
    Y = Y./sum(Y(:,1)); %normalize the data
    error = eigSolnAtS - Y(:,1:1/stepsize(iStepsize):length(t)); disp(['max resid:' num2str(max(max((error))))]);
    plot(t,Y(1,:), 'color', colormp(idx,:)); idx = idx+1;
    leg{iStepsize+1} = num2str(stepsize(iStepsize));
end
legend(leg)
end

function [Y] = solveUsingEigenvalues(A,Y0)
global colormp idx eigSolnAtS
disp('ieg start:')
tic,
[X,lambda] = eig(A);
C = X\Y0;
t = 0:0.1:10;
n = length(t);
for loop = 1:n
    Y(:,loop) = X*diag(exp(diag(lambda)*t(loop)))*C;
end
toc
figure(7), plot(t,Y(1,:), 'color', colormp(idx,:), 'linewidth',5); idx = idx+1;
eigSolnAtS = Y(:,1:(1/0.1):length(t));
end

function solveUsingImplicitEulerMethod(A,Y0)
global colormp idx eigSolnAtS
% stepsize = [20,5,1,0.01,.001];

```

```

disp('IE start:');
stepsize = [1, .1, 0.01, .001]; leg = {'eigenvalues'};
for iStepsize = 1:length(stepsize)
    t = 0:stepsize(iStepsize):10;
    n = length(t);
    clear('Y')
    Y(:,1) = Y0;
    tic,
    for loop=2:n
        Y(:,loop) = (eye(3)-stepsize(iStepsize)*A)\Y(:,loop-1);
    end
    toc
    Y = Y./sum(Y(:,1)); %normalize the data
    error = eigSolnAtS - Y(:,1:1/stepsize(iStepsize):length(t)); disp(['max resid:' num2str(max(max((error))))]);
    figure(7), plot(t,Y(1,:), 'color', colormap(idx,:)); idx = idx+1;
    leg(iStepsize+1) = num2str(stepsize(iStepsize));
end
legend(leg)
end

function Y0 = getY0()
Y0=[1 0 0 .8 .3 .5 .3 .5 .2959;
    0 1 0 .2 0 .1 .2 .5 .5801;
    0 0 1 0 .7 .4 .5 0 .124];
end
function ydot = mm(t,y) %function of differential equations to be solved
global A
ydot = zeros(3,1); %set dimensions of ydot matrix, each row is a species
ydot(1) = A(1,:)*y;
ydot(2)= A(2,:)*y;
ydot(3)= A(3,:)*y;
end
function A = getA
k = [0.4 0.2 0.1 0.45 0.4 0.16]; % kAB kba kBC kcb kCA kac
A = [-(k(1)+k(6)) k(2) k(5);...
    k(1) -(k(2)+k(3)) k(4);...
    k(6) k(3) -(k(5)+k(4))];
end
function [Y] = Eigen(A,Y0)
[X,lambda] = eig(A);
C = X\Y0;
t = 0:0.1:10;
n = length(t);
for loop = 1:n
    Y(:,loop) = X*diag(exp(diag(lambda)*t(loop)))*C;
end
end

```

```

function testIntegrators
warning('off')
close all
A = getA();
Yinitial = getY0();
global colormp idx
nColorChoices = 4;
colormp1=linspace(1,0.5,nColorChoices); colormp2=linspace(0.5,1,nColorChoices); colormp3=linspace(0.5,1,nColorChoices);
colormp = [1,1,0;
           colormp1' zeros(nColorChoices,2);
           zeros(nColorChoices,1) colormp2' zeros(nColorChoices,1);
           zeros(nColorChoices,2) colormp3'
           zeros(nColorChoices,1) colormp2' colormp3';...
           colormp3' zeros(nColorChoices,1) colormp2']; idx = 1;

%added 9/12/2018
figure(7), subplot(2,2,1), hold on,
solveUsingEigenvalues(A,Yinitial(:,6)); solveUsingImplicitEulerMethod(A,Yinitial(:,6));
title('Implicit Euler'); ylabel('residual error'); xlabel('time'); subplot(2,2,2), hold on,
solveUsingEigenvalues(A,Yinitial(:,6)); solveUsingAdamsBashforth(A,Yinitial(:,6));
title('Adams Bashforth'); ylabel('residual error'); xlabel('time'); subplot(2,2,3), hold on,
solveUsingEigenvalues(A,Yinitial(:,6)); solveUsing4THOrderRK(A,Yinitial(:,6));
title('Runge-Kutta 4'); ylabel('residual error'); xlabel('time'); subplot(2,2,4), hold on,
solveUsingEigenvalues(A,Yinitial(:,6)); solveUsing5THOrderRK(A,Yinitial(:,6));
title('Runge-Kutta 5'); ylabel('residual error'); xlabel('time');

end

% this is a new method by GH 9/12/2018 to use a multi-step adam's mashforth method to solve the ode
function Y = solveUsingAdamsBashforth(A,Y0)
disp('AB start:')
global colormp idx eigSolnAtS
stepsize = [1,1,0.01,.001]; leg = {'eigenvalues'};
for iStepsize = 1:length(stepsize)
    t = 0:stepsize(iStepsize):10;
    n = length(t);
    h = stepsize(iStepsize);
    Y(:,1) = Y0;
    Y(:,2) = (eye(3)-h*A)\Y(:,1); %implicit euler for t2
    tic,
    for loop=3:n
        Y(:,loop) = Y(:,loop-1) + 3/2*h*(A*Y(:,loop-1))-1/2*h*(A*Y(:,loop-2));
    end
    toc
    Y = Y./sum(Y(:,1)); %normalize the data
    error = eigSolnAtS - Y(:,1:1/stepsize(iStepsize):length(t)); disp(['max resid:' num2str(max(max((error))))]);
    figure(7), plot(t,Y(1,:),'color',colormp(idx,:)); idx = idx+1;
    leg(iStepsize+1) = num2str(stepsize(iStepsize));
end
legend(leg)
end

% GH 9/14/2018 -- adding a 4th order runga kutta for evaluation
function Y = solveUsing4THOrderRK(A,Y0)
global colormp idx eigSolnAtS
disp('RK4 start:')
stepsize = [1,1,0.01,.001]; leg = {'eigenvalues'};
for iStepsize = 1:length(stepsize)
    % t = 0.1:stepsize(iStepsize):100;
    t = 0:stepsize(iStepsize):10;
    n = length(t);
    h = stepsize(iStepsize);
    Y(:,1) = Y0;
    tic,
    for loop=2:n
        k1 = h*A*Y(:,loop-1);
        k2 = h*A*(Y(:,loop-1)+0.5*k1);
        k3 = h*A*(Y(:,loop-1)+0.5*k2);

```

```

        k4 = h*A*(Y(:,loop-1)+k3);
        Y(:,loop) = Y(:,loop-1) + (k1+2*k2+2*k3+k4)/6;
    end
    toc
    Y = Y./sum(Y(:,1)); %normalize the data
    error = eigSolnAtS - Y(:,1:1/stepsize(iStepsize):length(t));    disp(['max resid:' num2str(max(max((error))))]);
    figure(7), plot(t,Y(1,:), 'color', colormp(idx,:)); idx = idx+1;
    leg{iStepsize+1} = num2str(stepsize(iStepsize));
end
legend(leg)
end

% 5th order RK as described in:
% AN ALGORITHM USING RUNGE-KUTTA METHODS OF ORDERS 4 AND 5 FOR SYSTEMS OF ODEs by NIKOLAOS S.
CHRISTODOULOU
% http://www.teihal.gr/gen/profesor/christodoulou/Publications/An%20algorithm%20using.pdf
function Y = solveUsing5THOrderRK(A,Y0)
global colormp idx eigSolnAtS
disp('RK5 start:')
stepsize = [2,1,.1,0.01,.001]; idx = 1; leg = {'eigenvalues'};
for iStepsize = 1:length(stepsize)
%    t = 0.1:stepsize(iStepsize):100;
    t = 0:stepsize(iStepsize):10;
    n = length(t);
    h = stepsize(iStepsize);
    Y(:,1) = Y0;
    tic,
    for loop=2:n
        k1 = h*A*Y(:,loop-1);
        k2 = h*A*(Y(:,loop-1)+0.5*k1);
        k3 = h*A*(Y(:,loop-1)+(3*k1+k2)/16);
        k4 = h*A*(Y(:,loop-1)+k3/2);
        k5 = h*A*(Y(:,loop-1)+(-3*k2+6*k3+9*k4)/16);
        k6 = h*A*(Y(:,loop-1)+(k1+4*k2+6*k3-12*k4+8*k5)/7);
        Y(:,loop) = Y(:,loop-1) + (7*k1+32*k3+12*k4+32*k5+7*k6)/90;
    end
    toc
    Y = Y./sum(Y(:,1)); %normalize the data
    error = eigSolnAtS - Y(:,1:1/stepsize(iStepsize):length(t));    disp(['max resid:' num2str(max(max((error))))]);
    plot(t,Y(1,:), 'color', colormp(idx,:)); idx = idx+1;
    leg{iStepsize+1} = num2str(stepsize(iStepsize));
end
legend(leg)
end

function [Y] = solveUsingEigenvalues(A,Y0)
global colormp idx eigSolnAtS
disp('IE start:')
tic,
[X,lambda] = eig(A);
C = X\Y0;
t = 0:0.1:10;
n = length(t);
for loop = 1:n
    Y(:,loop) = X*diag(exp(diag(lambda)*t(loop)))*C;
end
toc
figure(7), plot(t,Y(1,:), 'color', colormp(idx,:), 'linewidth',5); idx = idx+1;
eigSolnAtS = Y(:,1:(1/0.1):length(t));
end

function solveUsingImplicitEulerMethod(A,Y0)
global colormp idx eigSolnAtS
% stepsize = [20,5,.1,0.01,.001];
disp('IE start:')
stepsize = [1,.1,0.01,.001]; leg = {'eigenvalues'};
for iStepsize = 1:length(stepsize)
    t = 0:stepsize(iStepsize):10;

```

```

n = length(t);
clear('Y')
Y(:,1) = Y0;
tic,
for loop=2:n
    Y(:,loop) = (eye(3)-stepsize(iStepsize)*A)\Y(:,loop-1);
end
toc
Y = Y./sum(Y(:,1)); %normalize the data
error = eigSolnAtS - Y(:,1:1/stepsize(iStepsize):length(t)); disp(['max resid:' num2str(max(max((error))))]);
figure(7), plot(t,Y(1,:), 'color', colormap(idx,:)); idx = idx+1;
leg{iStepsize+1} = num2str(stepsize(iStepsize));
end
legend(leg)
end

function Y0 = getY0()
Y0=[1 0 0 .8 .3 .5 .3 .5 .2959;
    0 1 0 .2 0 .1 .2 .5 .5801;
    0 0 1 0 .7 .4 .5 0 .124];
end
function ydot = mm(t,y) %function of differential equations to be solved
global A
ydot = zeros(3,1); %set dimensions of ydot matrix, each row is a species
ydot(1) = A(1,:)*y;
ydot(2)= A(2,:)*y;
ydot(3)= A(3,:)*y;
end
function A = getA
k = [0.4 0.2 0.1 0.45 0.4 0.16]; % kAB kba kBC kcb kCA kac
A = [-(k(1)+k(6)) k(2) k(5);...
    k(1) -(k(2)+k(3)) k(4);...
    k(6) k(3) -(k(5)+k(4))];
end
function [Y] = Eigen(A,Y0)
[X,lambda] = eig(A);
C = X\Y0;
t = 0:0.1:10;
n = length(t);
for loop = 1:n
    Y(:,loop) = X*diag(exp(diag(lambda)*t(loop)))*C;
end
end

```

7.32 Appendix AF: Edge detection in a Cartesian mesh

7.32.1 Vessel edge detection in 2D Cartesian mesh

In order to investigate the effects of different mass transfer choices and the convergence of mesh independence, the mass transfer models from Figure 7.203 were expressed in a 2D-domain. This was achieved through a novel edge detection algorithm. The general workflow begins with defining the coordinates (I and j indices in Cartesian mesh) of the bounding box encompassing the circle. These values are calculated using the radius and the center of the circle and with the *getSurroundingCell* function. Then the distance between each cell (quadrilateral) center and the vessel center is calculated. If the cell center is close to the circle edge (line 7 below), then it is labeled as an edge volume. If it is not an edge but is inside the radius, it is labeled with as an interior node. The remainder of nodes are, by default, are labeled an exterior node. The result of implementing this algorithm for two different circles with differing radii and centers is given in Figure 7.229.

Table 7.66. Pseudocode for vascular labeling of 2D Cartesian mesh

```

1.  FUNCTION labelCells2D(cellCenter,radius)
2.    getSurroundingCell(cellCenter[X] - radius[X], cellCenter[Y] - radius[Y], minI, minJ);
3.    getSurroundingCell(cellCenter[X] + radius[X], cellCenter[Y] + radius[Y], maxI, maxJ);
4.    FOR i = minI TO maxI DO
5.      FOR j = minJ TO maxJ DO
6.        dist = norm(getCellCenter(i,j) – cellCenter);
7.        IF abs(radius – dist) < edgeThickness THEN labelMatrix[i,j] = edge;
8.        ELSEIF dist < radius THEN labelMatrix[i,j] = interior;
9.        ELSE labelMatrix[i,j] = exterior;
10.     ENDFOR
11.   ENDFOR
12.   Return labelMatrix;
13. ENDFUNCTION

```

Table 7.67. Pseudocode for getting cell index surrounding a point in 2D Cartesian mesh

```

1.  FUNCTION getSurroundingCell (ptCoord, i, j)
2.     $i = \text{floor}(\text{ptCoord}[X]/dx * nVolX) + 1$ ;       $j = \text{floor}(\text{aPtCoord}/dy * nVolY) + 1$ ;
3.  ENDFUNCTION

```

Note, the variable *edgeThickness* must be set by the user and represents the thickness of the vascular wall (endothelial layer). While the endothelial layer thickness may vary from vessel to vessel, it can be modeled by a 1-cell thick layer using $\text{norm}(dx, dy)$ as the thickness. Dx and dy correspond to the x- and y- edge lengths of a mesh cell and the *norm* is the geometric norm of the vector $\langle dx, dy \rangle$.

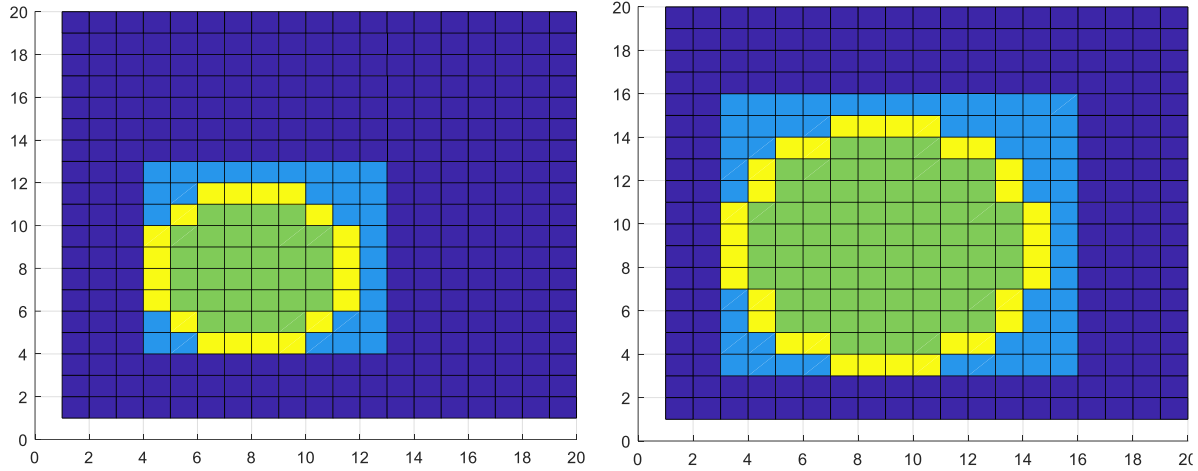


Figure 7.229. Example of edge detection algorithm visualized with Matlab.

Using the Cartesian meshing logic, (light blue) the domain of the circle can be identified. This region shortens the search area and can be interrogated as to whether each element center is outside the vascular segment (distance > radius, light blue), inside the cell (distance < radius, green) or if it is on the vessel edge (distance – radius < endothelial thickness, yellow). The results here are shown for two circles with different centers and different radii on a 10 x10 mesh.

The equations of conservation are defined differently for each group (each color in Figure 7.229) as follows:

For blue (light and dark):

$$rxn = Diffusion \quad (7.620)$$

$$R_0 = D \frac{\partial^2 c}{\partial x^2}$$

For yellow:

$$rxn = Diffusion + MT \quad (7.621)$$

$$R_0 = D \frac{\partial^2 c}{\partial x^2} - \frac{UA}{\partial x} (c_t - c_v)$$

For green:

$$(7.622)$$

$$c_i = c_{vasc}$$

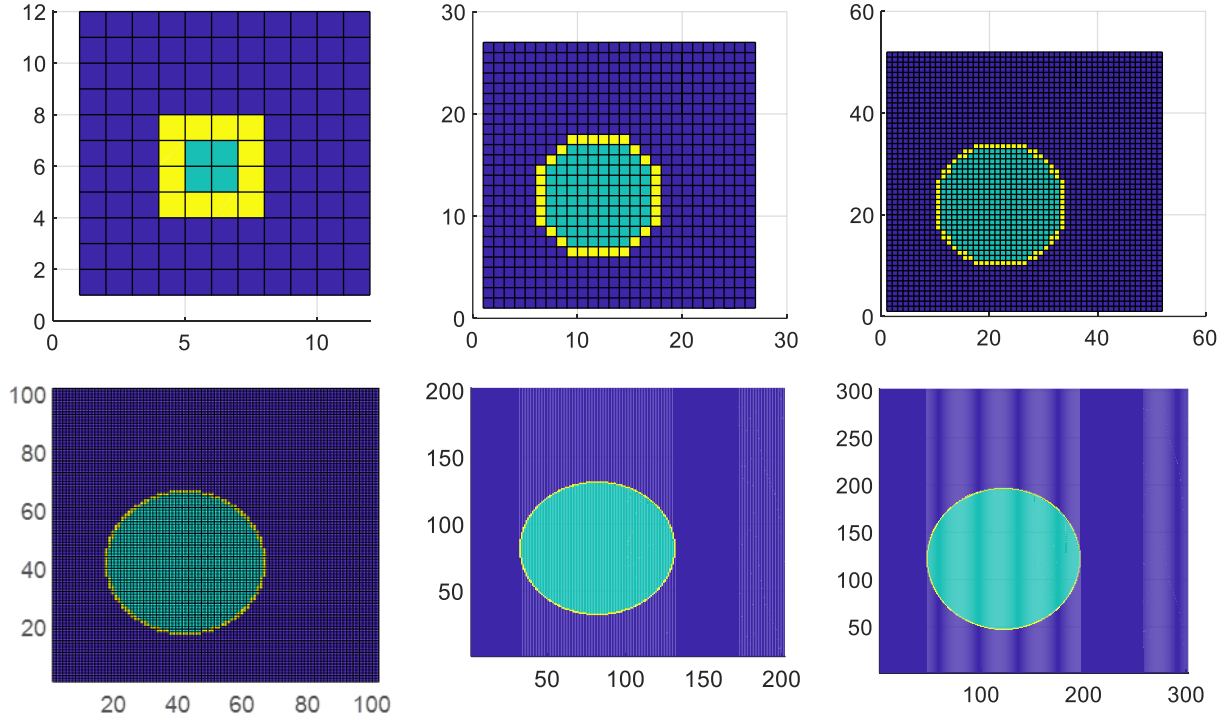


Figure 7.230. Examples of edge detection at varying mesh densities from 10x10 (top left) to 500x500 (bottom right).

Using an edge thickness of $|\langle dx, dy \rangle|$ gives a one-voxel thick edge in all cases.

7.32.2 Validating implementation with diffusion

One method of validating the edge detection is to compare the numerical solution with an analytic solution. Luckily, an analytic solution for the diffusion in a cylindrical domain is readily available. The mathematics for the analytic form follow Section 7.28.6.1 and the numerical follows:

For blue (light and dark):

(7.623)

$$R_0 = D \frac{\partial^2 c}{\partial x^2}$$

Conclusion. The analytic solution is accurately represented by the discretization method.

Implementation. To implement this step of the mesh integration and edge detection, and to simplify the programming, two mesh labels will be generated, one for the cylinder source and one for the exterior of the cylindrical domain. These will then be interrogated for equation generation.

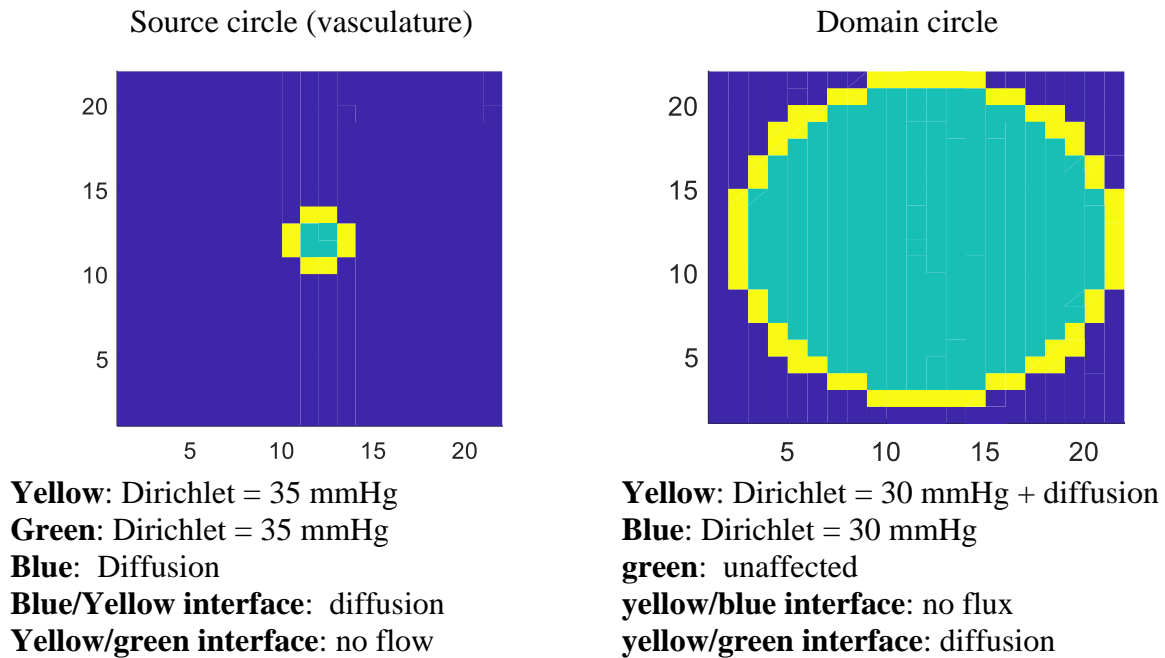


Figure 7.231. Visualization of the discrete labeling on two meshes.

Left) The source cylinder is modeled as a small circle in the center of the domain. Right) The domain edge is also modeled as a circle. Note, the off-center coloration is an artifact of Matlab surface plotting.

The two meshes will be merged and interrogated as one larger matrix as follows:

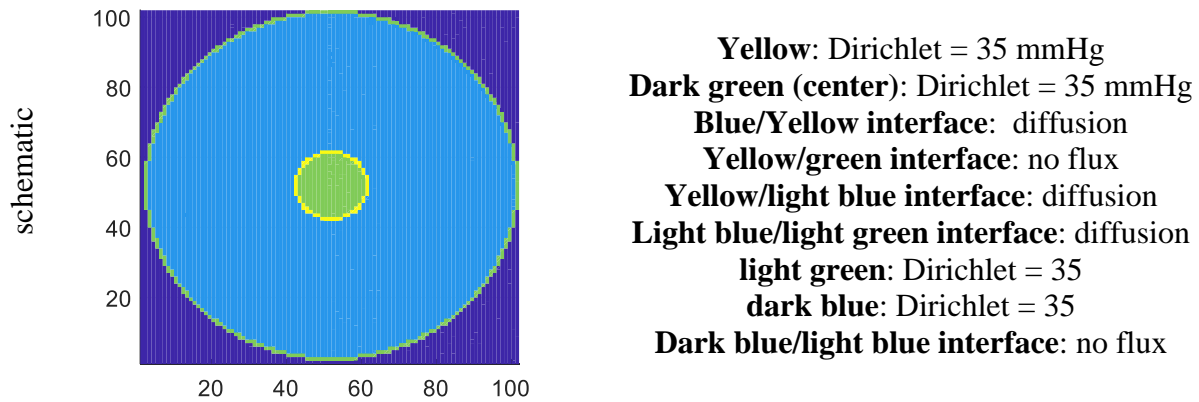


Figure 7.232. Visualization of the amalgamated meshes into a single labeled matrix. The equations are explained for each coloration as well.

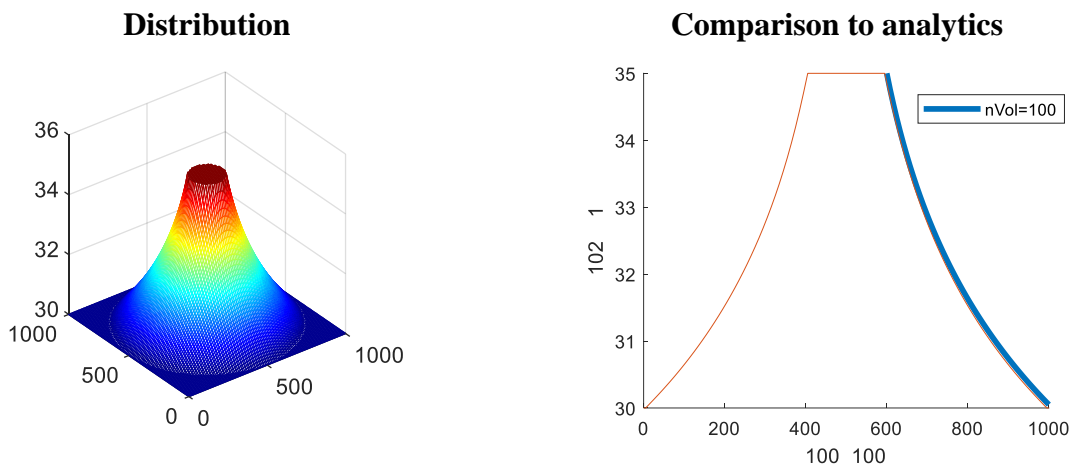


Figure 7.233. Concentration profile of the circular domain simulation. Left) The distribution is symmetric around the center axis. Right) The numerical solution matches well to the analytic solution. The very slight discrepancy is attributed to discretization error (see Section 7.28.5 half-volume technique).

7.32.3 Identifying vessel edge in 3D

In order to accurately account for mass transfer in 3D with a fine grid (grid spacing is smaller than the largest vessel diameter), the edge detection algorithm from the previous section was expanded. The calculation that solves for perpendicular distance in 2D is offered in Section 5.5 and 0. In short, every vessel can be defined by the minimum coordinates and maximum coordinates

of the two points that make up the terminals of the segment (p_1 and p_2) and the diameter. If a new point is determined to be between the first and second point, axially, then the perpendicular distance between the point and the segment characterizes whether or not the point is within the cylinder (distance < radius). Due to the gaps between adjacent segments modeled as perfect cylinders, the ends of the cylinder should be endowed with a sphere. A pseudocode has been offered in Section 5.5.

7.32.3.1 Case Studies

In order to visualize the mesh labeling, a novel rendering paradigm was generated that emphasizes the sparseness of the data (most of the mesh elements do not contain a vascular segment). Another type of data that shares this property is DICOM medical images. As such, the results of the labeling algorithm has been written to DICOM format for easy rendering in existing visualization software.

Edge detection. The result visualized below shows excellent rendering of the vascular structure. Unfortunately, when a mesh is generated obtaining all centerline points, it cuts off some of the radii and thus the mesh will be expanded for further case studies.

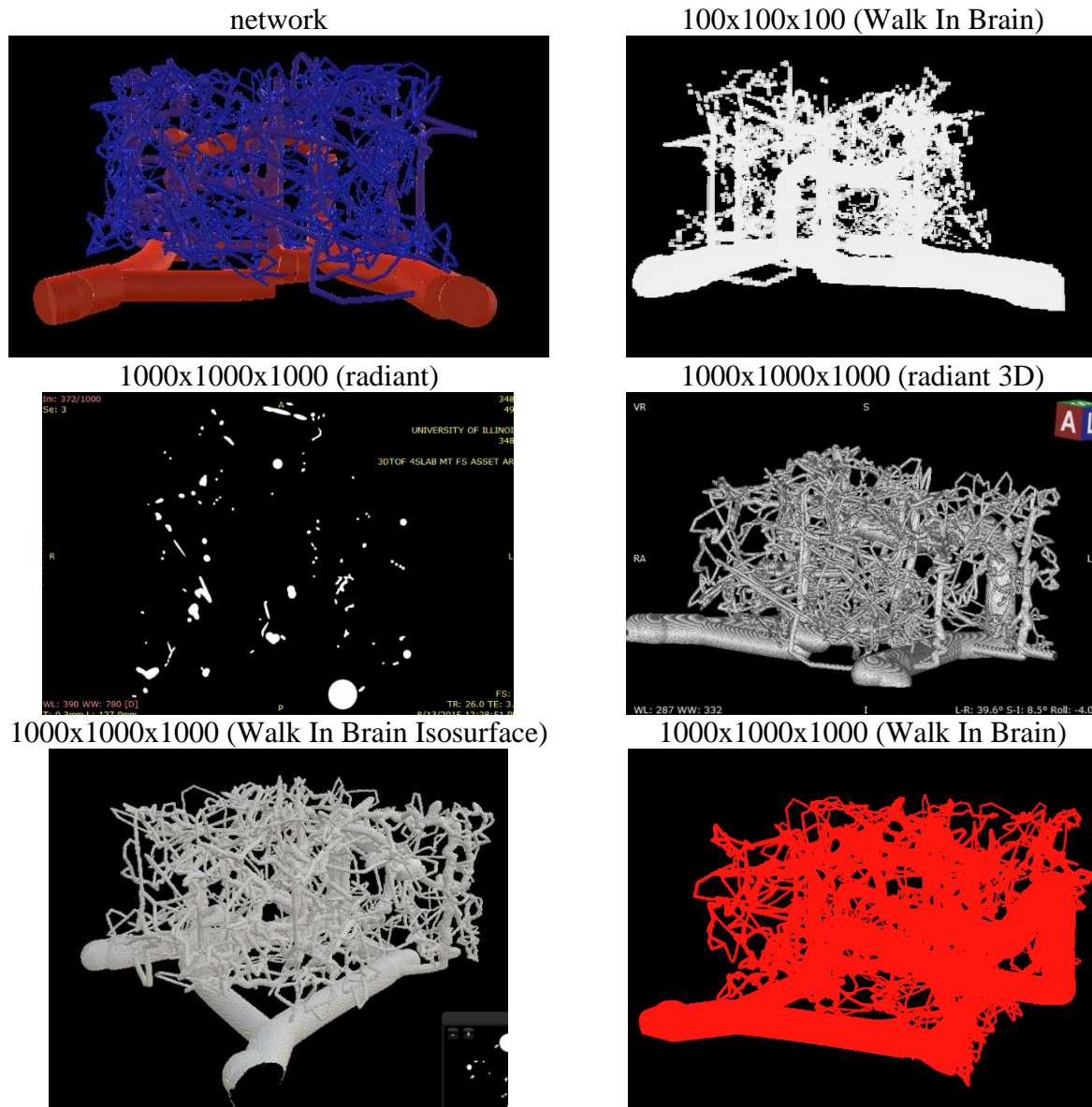
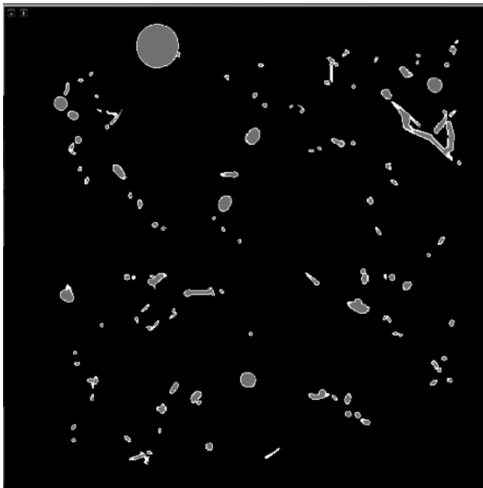


Figure 7.234. pictorial representation of the vessel labeling algorithm in 3 dimensions. The results, stored in DICOM image format and visualized in many platforms, shows excellent agreement with the original network structure (top left).

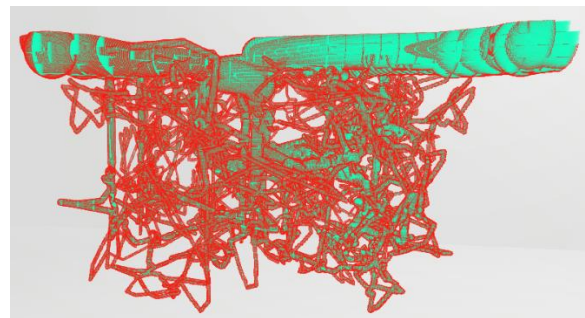
Edge detection with vessel index tracking. In order to properly align (register) the vascular segments with the mesh, it is imperative to keep track of which segment correlates to which mesh voxel. This is implemented by filling a face index vector during the isPtInCylinder procedure, where each voxel has a value corresponding to which network segment it correlates with (default

value of 0 for voxels with no vascular segment). Later, this will be expanded to a matrix and many vascular segments can be added to a single mesh element and likewise for many mesh elements for a single vessel.

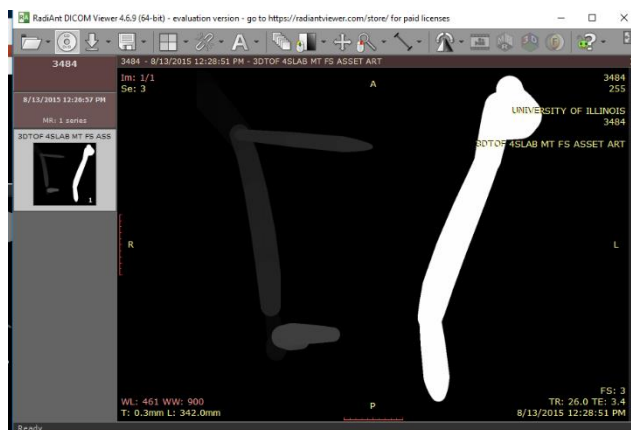
Edge detection (slice)



Edge detection (3D coloration, red=edge)



Vessel correlation (coloration corresponds to different vessels) *



Vessel correlation (coloration, note very low values are cut off) *

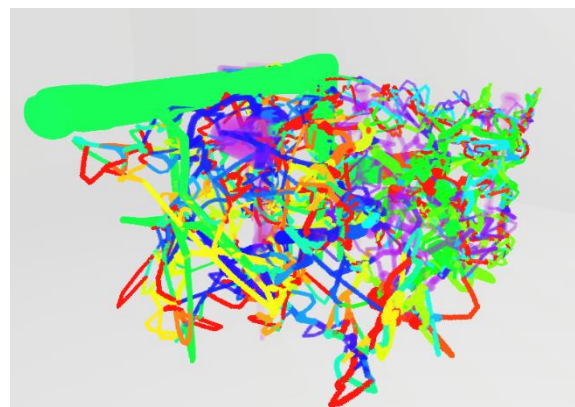


Figure 7.235. Visualization of network edge detection using DICOM viewer tools. Top Row) and the vessel index correlation (Bottom Row). Note, repeating coloration is due to limited color selection in DICOM review tools.

7.32.3.2 Simulation case studies

Further investigation of the effect of registering the voxels to a network can be seen by simulating a voxel matrix with the tagged voxels (voxels corresponding to a vessel segment) set to a nonzero Dirichlet boundary condition and the boundary of the tissue set to a Dirichlet value of 0. This case study will exemplify the accuracy of labeling the mesh and ensuring that the vessels do not hinder the simulation.

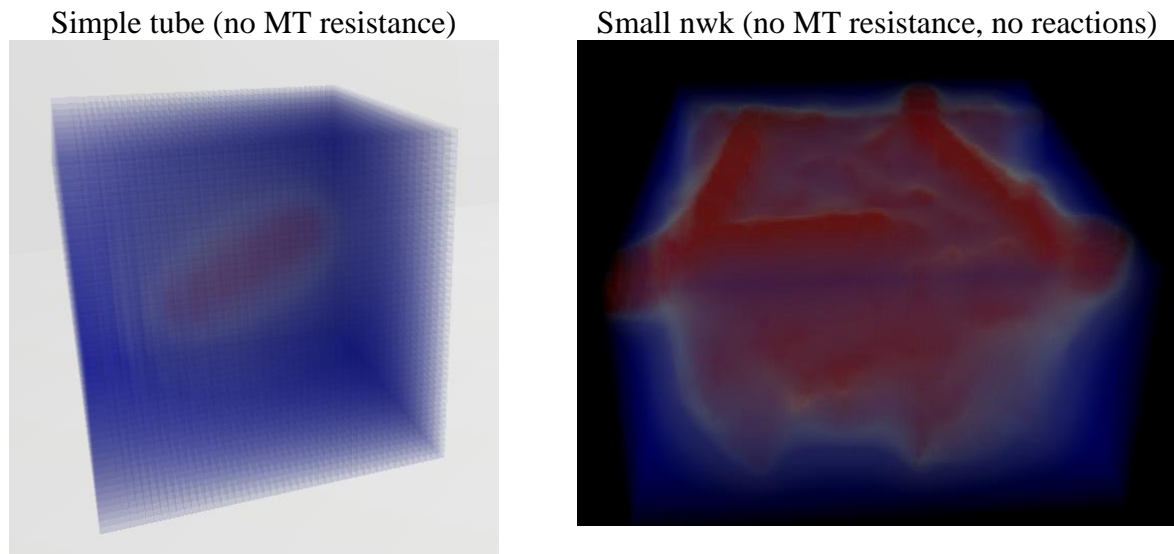
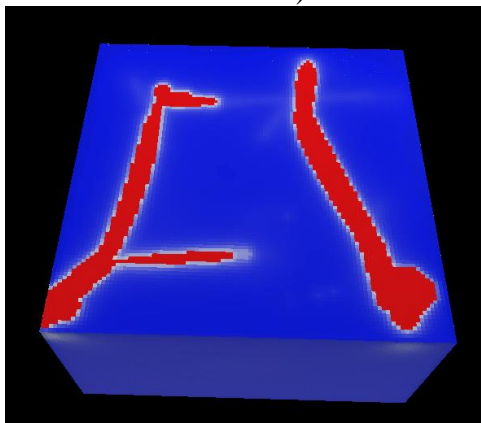


Figure 7.236. Visualization of simulation for two networks using vessel edge detection visualized with DICOM image reviewing tools.

Red indicates high concentration and blue indicates low concentration. White indicates the mean of the concentration range.

Small nwk (no MT resistance, no reactions)



Small nwk (no MT resistance, with reactions)

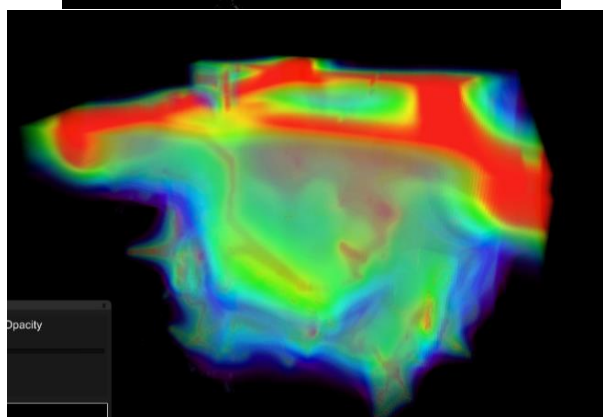
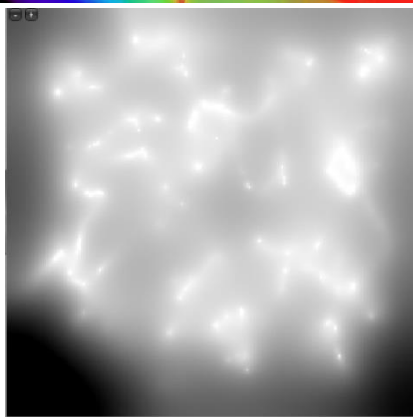
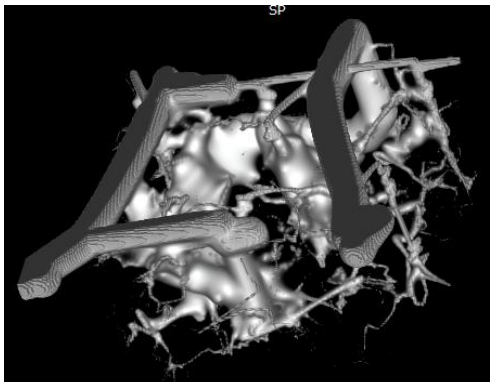
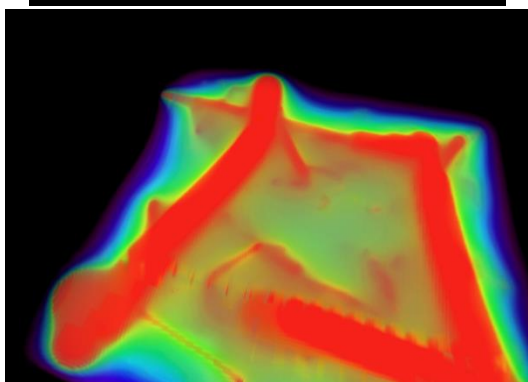
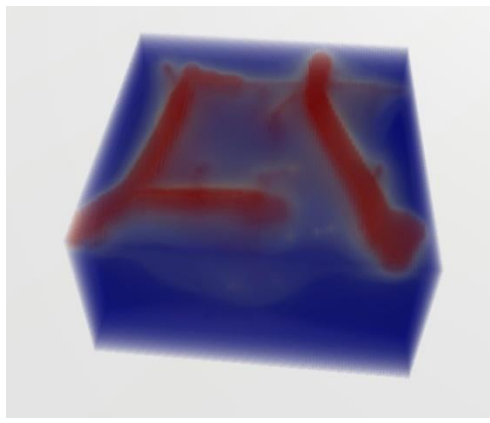












Figure 7.237. Visualization of oxygen simulation in many DICOM review tools. This proof-of-concept exemplifies the robustness of this visualization paradigm.

7.33 Appendix AG: Parametric studies (parameter sensitivity) on 1D-3D coupling

ID indicates the name for each case study. E1.1_85_Oxy_UU_0.1 refers to the E1.1 data set (E1.1), 85 Volume (85), oxygen simulation (Oxy). UU stands for mass transfer coefficient (symbol UU), and the following number is the value of UU for the simulation. Note, scaled max balance error is the infinity norm of the mass balance at point error divided by maximum flow in network.

7.33.1 Mass Transfer

ID	Pressure (mmHg)	Time to solve (s)	Scaled max network balance error	Scaled max tissue balance error	Mass transfer into tissue	Mass transfer residual error	Overall extravas- ation
E1.1_85_O xy_UU_0 .01		83.939	2.3830e-11	-1.3982e-11	6.7346e8	-9.4162e-3	-6.7346e8
E1.1_85_O xy_UU_0.1		530.60	4.0720e-12	-1.5768e-11	7.4244e8	-1.1707e-2	-7.4244e8
E1.1_85_O xy_UU_1		244.17	6.0985e-13	-1.6645e-11	7.4078e8	-1.2330e-2	-7.4078e8
E1.1_85_O xy_UU_10		158.95	2.3559e-12	-1.5941e-11	7.4106e8	-1.1812e-2	-7.4106e8
E1.1_85_O xy_UU_10 0		130.15	3.7388e-12	-1.5873e-11	7.4111e8	-1.1764e-2	-7.4111e8
E1.1_85_O xy_UU_24 00		126.30	3.5687e-12	-1.5872e-11	7.4111e8	-1.1763e-2	-7.4111e8
E1.1_85_O xy_UU_50 00		123.66	2.2339e-12	-1.5859e-11	7.4111e8	-1.1753e-2	-7.4111e8
E1.1_85_O xy_UU_1e 4		118.83	2.4861e-12	-1.5885e-11	7.4111e8	-1.177e-2	-7.4111e8
E1.1_85_O xy_UU_1e 5		89.33	2.5013e-12	-1.5857e-11	7.4111e8	-1.1752e-2	-7.4111e8
E1.1_85_O xy_UU_1e 6		110.64	3.0518e-12	-1.8681e-11	7.4111e8	-1.3845e-2	-7.4111e8

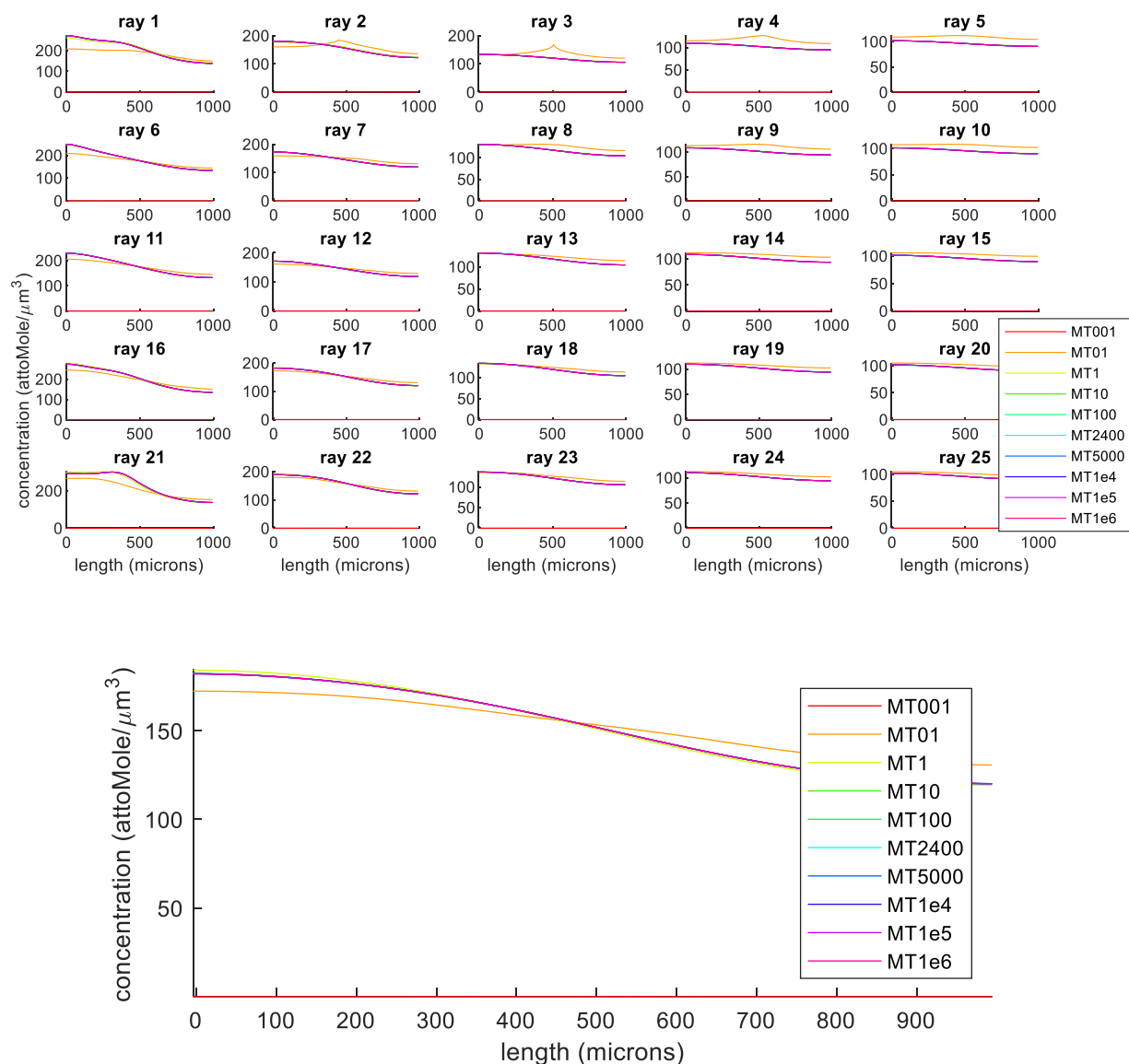
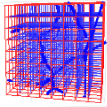
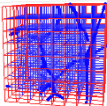
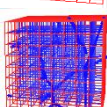
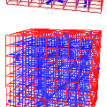
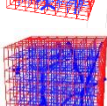
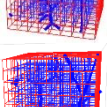
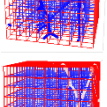
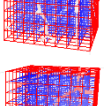
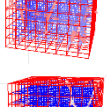


Figure 7.238. Raytraces for different rays through the 3D block with varying mass transfer coefficient corresponding to the above table.

The bottom example ray showcases ray 17 showing that the after a mass transfer coefficient of 100, all values converge to the same line. In our study, we choose 2,400 $\mu\text{m}^2/\text{s}$ (light blue). Y label is misnamed, it should be named “concentration, attoMole/ μm^3 ”.

7.33.2 Reaction (kk)

ID	Pressure (mmHg)	Time to solve (s)	Scaled max network balance error	Scaled max tissue balance error	Mass transfer into tissue	Mass transfer residual error	Overall extravas- ation
E1.1_85_ Oxy_kk_1 e-6		6591.29	2.6917e-12	4.4502e-9	3.4637e8	1.5414	-3.4637e8
E1.1_85_ Oxy_kk _1e-5		5549.15	2.6917e-12	4.4502e-9	3.4637e8	1.5414	-3.4637e8
E1.1_85_ Oxy_kk _1e-4		2263.6	7.8475e-13	4.4099e-9	6.6643e8	2.9389	-6.6643e8
E1.1_85_ Oxy_kk _1e-3		286.85	2.4577e-12	4.0255e-9	7.3439e8	2.9563	-7.3439e8
E1.1_85_ Oxy_kk _41.7e-4		122.84	3.6672e-12	2.9862e-9	7.4110e8	2.2130	-7.4110e8
E1.1_85_ Oxy_kk _0.01		93.699	4.3301e-12	1.8501e-9	7.4272e8	1.3741	-7.4272e8
E1.1_85_ Oxy_kk _0.1		26.843	1.5611e-12	3.4044e-11	7.4496e8	0.02536	-7.4496e8
E1.1_85_ Oxy_kk _1		10.347	1.7412e-12	4.7943e-16	7.4595e8	3.5763	-7.4959e8
E1.1_85_ Oxy_kk _10		3.872	3.2083e-13	-1.5974e-16	7.4628e8	-1.1921e-7	-7.4628e8

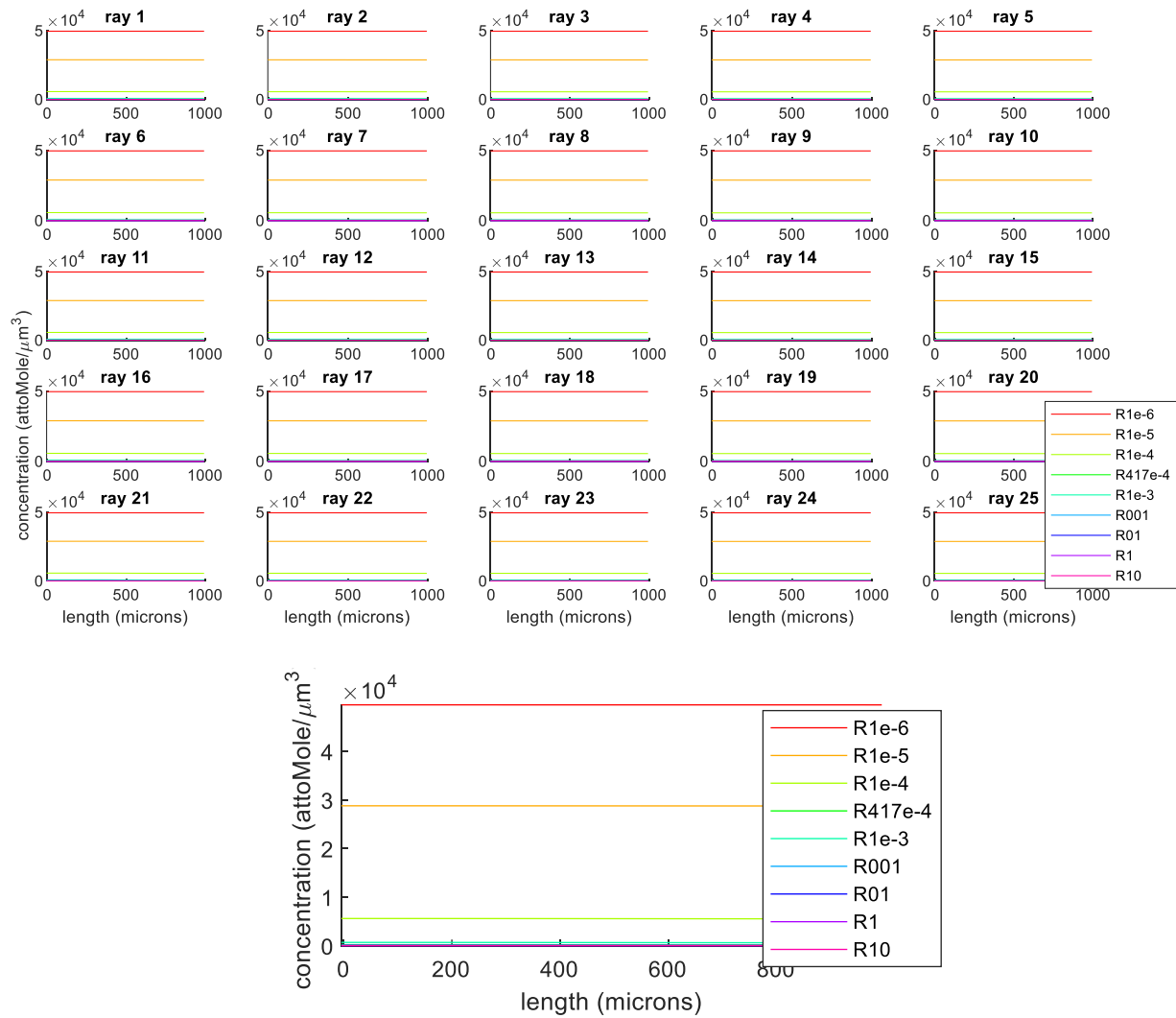
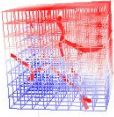
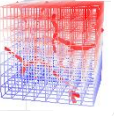
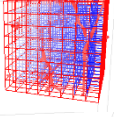
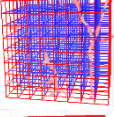
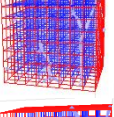
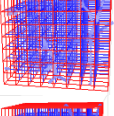
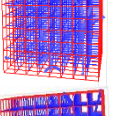
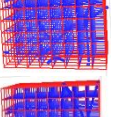
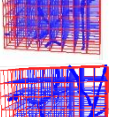
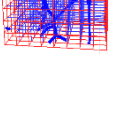


Figure 7.239. Raytraces for different rays through the 3D block with varying the metabolic rate (CMRO₂) corresponding to the above table.

The bottom example ray showcases ray 17 showing that the a value of 41.7e-4, all values converge to the same line. In our study, we choose 41.7e-4 attomole/μm³/s (dark green).

7.33.3 Diffusivity (DD)

ID	Pressure (mmHg)	Time to solve (s)	Scaled network balance error	Scaled tissue balance error	Mass transfer into tissue	Mass transfer residual error	Overall extravas- ation
E1.1_85_O xy_DD_0.1		100.73	3.6765E-12	-1.4240E-8	6.5281E8	-9.2958	-6.5281E8
E1.1_85_O xy_DD_1		71.294	3.8434E-12	-2.9301E-12	6.7235E8	-1.9701E-3	-6.7235E8
E1.1_85_O xy_DD _10		38.019	3.2959E-12	3.6755E-13	6.7818E8	2.4927E-4	-6.7818E8
E1.1_85_O xy_DD _100		34.632	2.9514E-12	1.4438E-9	7.1757E8	1.0361	-7.1757E8
E1.1_85_O xy_DD _600		68.833	3.9133E-12	4.6153E-9	7.3651E8	3.3992	-7.3651E8
E1.1_85_O xy_DD _900		86.447	3.3327E-12	4.2085E-9	7.3862E8	3.1085	-7.3862E8
E1.1_85_O xy_DD _1800		92.276	3.6672E-12	2.9862E-9	7.4111E8	2.2131	-7.4111E8
E1.1_85_O xy_DD _5000		185.53	2.0575E-13	1.3796E-9	7.4302E8	1.0251	-7.4302E8
E1.1_85_O xy_DD _1e4		395.40	1.9139E-13	7.4318E-10	7.4362E8	0.55264	-7.4362E8
E1.1_85_O xy_DD _1e5		2350.2	1.9395E-14	7.9624E-11	7.4418E8	0.059255	-7.4418E8

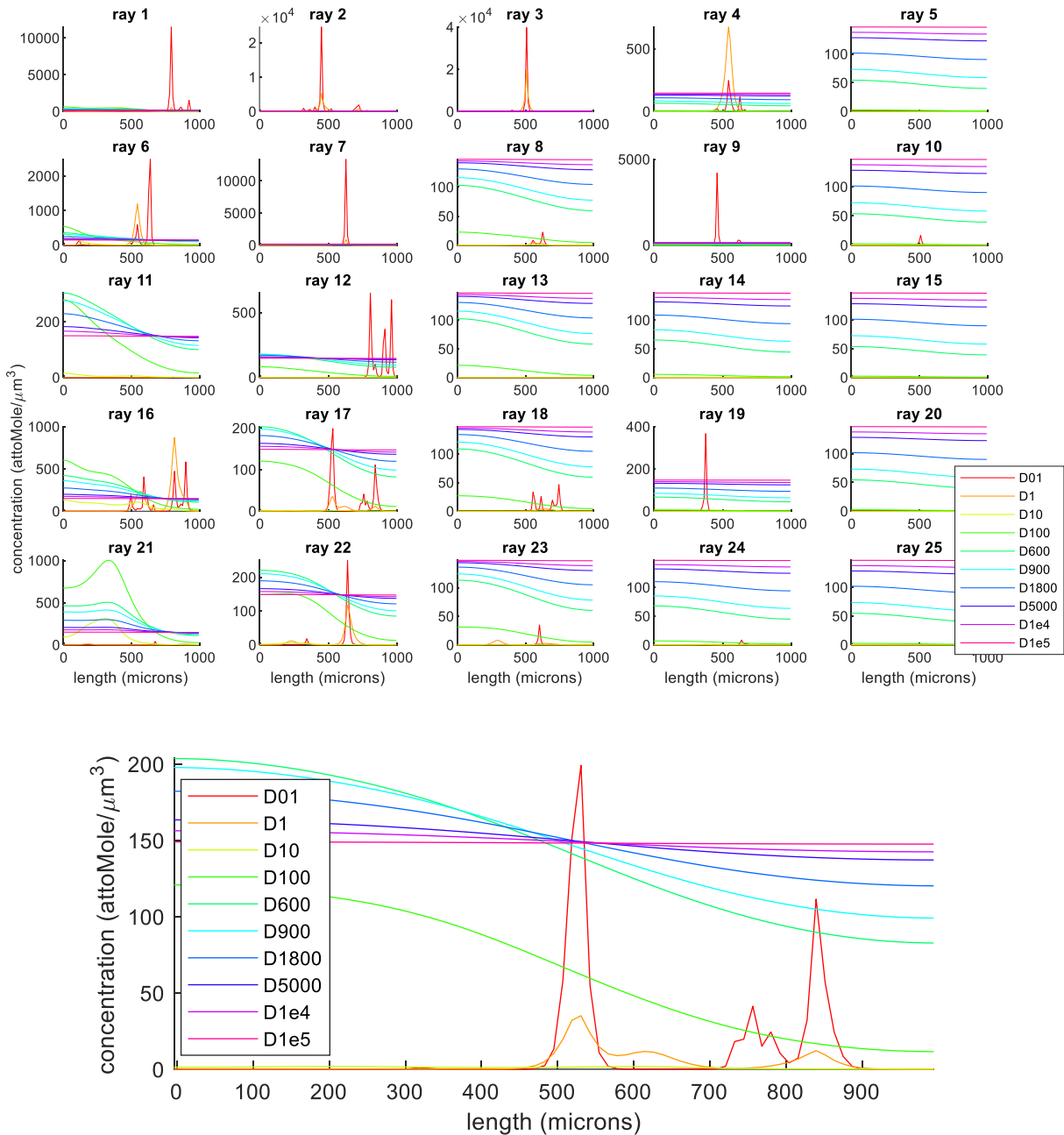
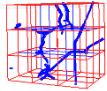
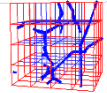
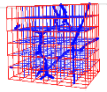
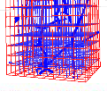
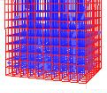
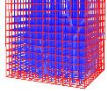
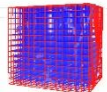
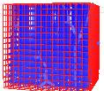
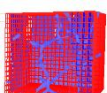


Figure 7.240. Raytraces for different rays through the 3D block with varying diffusivity corresponding to the above table.

The bottom example ray showcases ray 17. In this parametric study, the diffusivity has a great impact on the simulation results throughout the definition space. With slower diffusivity, the concentration peaks are more prominent and with faster diffusivity, the concentration profile flattens out. In our study, we choose 1,800 $\mu\text{m}^2/\text{s}$ (dark blue).

7.33.4 Volumes

ID	Pressure (mmHg)	Time to solve (s)	Scaled network balance error	Scaled tissue balance error	Mass transfer into tissue	Mass transfer residual error	Overall extravas- ation
E1.1_85_Ox y_nVol_25		4.39	3.3085E-13	-1.9628E-14	7.4096E8	-1.4543E-5	-7.4096E8
E1.1_85_Ox y_nVol_45		12.249	3.9019E-12	-5.1493E-12	7.4135E8	-3.8174E-3	-7.4135E8
E1.1_85_Ox y_nVol_65		64.409	1.3125E-12	1.6088E-16	7.4098E8	1.1921E-7	-7.4098E8
E1.1_85_Ox y_nVol_85		104.81 5	3.6672E-12	2.9862E-9	7.4111E8	2.2131	-7.4111E8
E1.1_85_Ox y_ nVol_105		328.89 4	3.7634E-12	-1.3028E-8	7.4107E8	-9.6544	-7.4107E8
E1.1_85_Ox y_ nVol_125		771.20	3.7275E-12	9.4114E-9	7.4114E8	6.9751	-7.4114E8
E1.1_85_Ox y_ nVol_145		1366.4	2.0466E-12	7.7026E-9	7.4113E8	5.7087	-7.4113E8
E1.1_85_Ox y_ nVol_165		2402.1	3.0782E-12	-1.4406E-12	7.4117E8	-1.0678E-3	-7.4117E8
E1.1_85_Ox y_ nVol_185		3589.5	4.0216E-12	-8.6794E-12	7.4115E8	-6.4328E-3	-7.4115E8

7.34 Appendix AH: Physical chemistry behind oxygen composition in brain:

This section delineates the unit conversion between partial pressure and concentration of a soluble gas at STP. It is important to note that most people calculate this conversion using the Henry constant as follows:

$$\frac{c_i}{p_i} = \frac{c_i}{p_{tot}x_i} = H \quad (7.624)$$

But in reality, the Henry constant is derived as the critical point where the fugacity of the liquid-phase material and the gas-phase material are equal. This means that if the ratio of molar fraction to absolute concentration of the gas-phase solute ($p_{tot}x_i/c_i$) increases beyond the Henry constant, precipitation occurs. This can occur in the case where the overall pressure decreases, or molar fraction increases. The currently accepted use of this equation for unit conversion between partial pressure and concentration is using the assumption that the gas is at the critical saturation limit at all times, an assumption that is never acknowledged. The actual concentration of oxygen is ~10% of the critical limit in blood using the calculation derived in this section.

7.34.1 Background

The fugacity is a method of correcting the potential energy of the system (which is usually calculated with temperature-dependent Gibbs free energy) to account for potential energy from pressure. Fugacity in low pressures can be approximated using:

$$f_i^G = Py_i = p_i \quad (7.625)$$

The fugacity of the gas phase when liquid molar fraction of bulk (x_j) approaches 1 can be approximated as a linear relationship using a constant (k) constant:

$$\lim_{x_j \rightarrow 1} f_i^L \rightarrow f_i^L = kx_i \quad (7.626)$$

When in equilibrium, the fugacities of the gas and liquid phase must be the same:

$$f_i^G = f_i^L \quad (7.627)$$

$$p_i = y_i^g P_{tot} = kx_i = Hc_i \quad (7.628)$$

$$\frac{y_i^g}{c_i} = \frac{H}{P_{tot}} \quad (7.629)$$

If the fugacity of the gas is larger than the fugacity of the liquid phase, the gas phase precipitates and bubbles off.

7.34.2 Unit conversion between concentration and partial pressure

Consider the simplest problem, only water and oxygen at STP as below. The volume is 1L.

With given molecular properties:

Table 7.68. Molecular properties of oxygen and water

Property	O ₂	H ₂ O	Symbol	units
Molecular weight:	32	18.015	w	g/mol
Density at STP	1.429 (G) 1141 (L)	997 (L)	ρ	g/L mg/ml

And the partial pressure defined as:

$$\frac{p_i}{P_{tot}} = x_i \quad (7.630)$$

Consider the total moles within the volume is given by:

$$n_{tot} = n_{H_2O} + n_{O_2} \quad (7.631)$$

$$n_{O_2} = x_{O_2} n_{tot}$$

$$n_{O_2} = x_{O_2} (n_{H_2O} + n_{O_2})$$

$$n_{O_2} (1 - x_{O_2}) = x_{O_2} n_{H_2O}$$

$$n_{O_2} = \frac{x_{O_2}}{1 - x_{O_2}} n_{H_2O} \quad (7.632)$$

$$n_{H_2O} = \frac{1 - x_{O_2}}{x_{O_2}} n_{O_2}$$

The mass (in mg) is defined by density as:

$$m_{tot} = m_{H_2O} + m_{O_2}$$

$$m_{tot} = n_{H_2O}w_{H_2O} + n_{O_2}w_{O_2}$$

$$m_{tot} = \frac{1 - x_{O_2}}{x_{O_2}} n_{O_2} w_{H_2O} + n_{O_2} M_{O_2} \quad (7.633)$$

$$m_{tot} = n_{O_2} \left(\frac{1 - x_{O_2}}{x_{O_2}} w_{H_2O} + w_{O_2} \right)$$

Given the volume of the box, the partial volumes can be found:

$$V_{tot} = V_{O_2} + V_{H_2O}$$

$$V_{tot} = \frac{w_{O_2}}{\rho_{O_2}} n_{O_2} + \frac{w_{H_2O}}{\rho_{H_2O}} n_{H_2O} \quad (7.634)$$

Using substitution, the number of oxygen molecules can be obtained as follows:

$$V_{tot} = \frac{w_{O_2}}{\rho_{O_2}} n_{O_2} + \frac{w_{H_2O}}{\rho_{H_2O}} \frac{1 - x_{O_2}}{x_{O_2}} n_{O_2}$$

$$V_{tot} = n_{O_2} \left(\frac{w_{O_2}}{\rho_{O_2}} + \frac{w_{H_2O}}{\rho_{H_2O}} \frac{1 - x_{O_2}}{x_{O_2}} \right) \quad (7.635)$$

$$\frac{n_{O_2}}{V_{tot}} = c_{O_2} = \frac{1}{\frac{w_{O_2}}{\rho_{O_2}} + \frac{w_{H_2O}}{\rho_{H_2O}} \frac{1 - x_{O_2}}{x_{O_2}}}$$

$$c_{O_2} = \frac{\rho_{O_2} \rho_{H_2O} x_{O_2}}{w_{O_2} \rho_{H_2O} x_{O_2} + \rho_{O_2} w_{H_2O} (1 - x_{O_2})}$$

Where a partial pressure boundary condition from literature and the assumption of a total pressure of STP (760 mmHg or 1 atm) can lead to the computation of the inlet concentration:

$$x_{O_2} = \frac{p_{O_2}}{P_{tot}}$$

$$c_{O_2} = \frac{\rho_{O_2} \rho_{H_2O} \frac{p_{O_2}}{P_{tot}}}{w_{O_2} \rho_{H_2O} \frac{p_{O_2}}{P_{tot}} + \rho_{O_2} w_{H_2O} \left(1 - \frac{p_{O_2}}{P_{tot}}\right)} \quad (7.636)$$

To get from c_{O_2} back to partial pressure:

Method 1:

$$\frac{c_{O_2} w_{O_2} \rho_{H_2O} x_{O_2} + c_{O_2} \rho_{O_2} w_{H_2O} (1 - x_{O_2})}{\rho_{O_2} \rho_{H_2O} x_{O_2}} = 1$$

$$x_{O_2} \frac{c_{O_2} w_{O_2} \rho_{H_2O}}{\rho_{O_2} \rho_{H_2O}} + \frac{c_{O_2} \rho_{O_2} w_{H_2O}}{\rho_{O_2} \rho_{H_2O}} - x_{O_2} \frac{c_{O_2} \rho_{O_2} w_{H_2O}}{\rho_{O_2} \rho_{H_2O}} = x_{O_2} \quad (7.637)$$

$$x_{O_2} \left(\frac{c_{O_2} w_{O_2}}{\rho_{O_2}} - \frac{c_{O_2} w_{H_2O}}{\rho_{H_2O}} \right) + \frac{c_{O_2} w_{H_2O}}{\rho_{H_2O}} = x_{O_2}$$

$$\frac{w_{H_2O}}{\rho_{H_2O}} = x_{O_2} \left(1 - \frac{c_{O_2} w_{O_2}}{\rho_{O_2}} + \frac{c_{O_2} w_{H_2O}}{\rho_{H_2O}} \right)$$

$$x_{O_2} = \frac{c_{O_2} w_{H_2O}}{\rho_{H_2O}} \left(1 - \frac{c_{O_2} w_{O_2}}{\rho_{O_2}} + \frac{c_{O_2} w_{H_2O}}{\rho_{H_2O}} \right)^{-1}$$

Method 2:

$$V_{tot} = \frac{w_{O_2}}{\rho_{O_2}} n_{O_2} + \frac{w_{H_2O}}{\rho_{H_2O}} \frac{1 - x_{O_2}}{x_{O_2}} n_{O_2}$$

$$x_{O_2} \left(V_{tot} - \frac{w_{O_2}}{\rho_{O_2}} n_{O_2} \right) = \frac{w_{H_2O}}{\rho_{H_2O}} n_{O_2} - x_{O_2} \frac{w_{H_2O}}{\rho_{H_2O}} n_{O_2}$$

$$x_{O_2} \left(V_{tot} - \frac{w_{O_2}}{\rho_{O_2}} n_{O_2} + \frac{w_{H_2O}}{\rho_{H_2O}} n_{O_2} \right) = \frac{w_{H_2O}}{\rho_{H_2O}} n_{O_2}$$

$$x_{O_2} = \frac{w_{H_2O}}{\rho_{H_2O}} c_{O_2} \left(1 - \frac{w_{O_2}}{\rho_{O_2}} c_{O_2} + \frac{w_{H_2O}}{\rho_{H_2O}} c_{O_2} \right)^{-1}$$

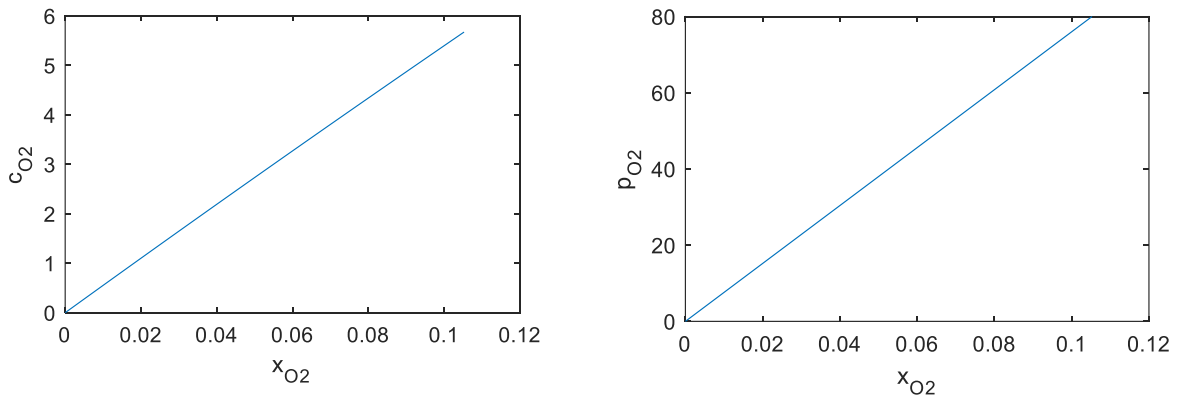


Figure 7.241. Parametric study of molar fraction of oxygen and the calculation of (left) moles of oxygen and (right) partial pressure of oxygen.

The molar concentration is slightly nonlinear yet the partial pressure is linear with respect to the molar fraction of oxygen.

```

function testPartialPressure
close all
V = 1;
densityH2O=997; densityO2=1411;
weightH2O =18.015; weightO2 =32;
pO2 = 0:0.1:80;
xO2=pO2./760;
cO2 = densityO2*densityH2O*xO2./(weightO2*densityH2O*xO2+densityO2*weightH2O*(1-xO2));
figure, plot(xO2,cO2), xlabel('x_O_2'); ylabel('c_O_2');
figure, plot(xO2,pO2), xlabel('x_O_2'); ylabel('p_O_2');
end

```

7.34.3 Conversion of CMRO₂ from medical units to moles/s

Unfortunately, the values commonly reported for the cerebral metabolic rate of oxygen (CMRO₂) are given in conventional units, but are inconvenient for mass transport simulation equations. This gives rise to a straightforward unit conversion from given values (8.2·10⁻⁶cm³O₂/cm³/s) to units (mol/s):

$8.2 \cdot 10^{-6} \frac{\text{cm}^3 \text{O}_2}{\text{cm}^3 \cdot \text{s}}$	$\frac{1141 \text{ mg}}{\text{cm}^3 \text{O}_2}$	$\frac{\text{mmol}}{32 \text{ mg}}$
---	--	-------------------------------------

Mathematical implementation:

$$\dot{r}(\text{mol/s}) = \dot{r}(\text{cm}^3 \text{O}_2 / \text{cm}^3 / \text{s}) \rho / w \quad (7.638)$$

Which results in a 0th order reaction rate of $\dot{r}=0.29238 \cdot 10^{-3} \text{mmol/cm}^3/\text{s}$. Which, when converted to our units (attomole/μm³/s) is:

$0.29238 \cdot 10^{-3} \frac{\text{mmol}}{\text{cm}^3 \cdot \text{s}}$	$\frac{1 \text{ cm}^3}{10^{12} \mu\text{m}^3}$	$\frac{10^{15} \text{ attomole}}{1 \text{ mmol}}$
--	--	---

Resulting in 0.29238 attomole/ $\mu\text{m}^3/\text{s}$. When using this reaction rate, the integral of reaction over the entire volume can be calculated:

$$\begin{aligned}
 loss &= \dot{r} \cdot V_{tiss} \\
 loss &= (0.29238 \text{ attomole}/\mu\text{m}^3/\text{s})(1.14 \cdot 10^9 \mu\text{m}^3) \\
 loss &= 0.33331 \cdot 10^9 \text{ attomole}/\text{s} \\
 loss &= 0.33331 \text{ nanomole}/\text{s}
 \end{aligned} \tag{7.639}$$

Where V_{tiss} for the E1.1 dataset is $1.14 \cdot 10^9 \mu\text{m}^3$, giving a loss of $0.33331 \cdot 10^9$ attomoles/s at steady-state, or 0.33331 nanomoles/s. If considering a 1st order reaction and an average concentration of 30 mmHg (an assumption), the reaction rate (k_1) can be approximated by:

$$\begin{aligned}
 loss &= k_1 \bar{c} V_{tiss} \\
 \text{Where } \bar{c} &= 2.1628 \cdot 10^3 \text{ attomole}/\mu\text{m}^3
 \end{aligned} \tag{7.640}$$

If the loss is assumed the same, then the reaction rate can be calculated using:

$$\begin{aligned}
 k_1 &= \frac{loss}{\bar{c} V_{tiss}} \\
 k_1 &= \frac{0.33331 \cdot 10^9 \text{ attomoles}/\text{s}}{\left(2.1628 \cdot \frac{10^3 \text{ attomole}}{\mu\text{m}^3}\right) (1.14 \cdot 10^9 \mu\text{m}^3)} \\
 k_1 &= 1.3518 \cdot 10^3 \text{ s}^{-1}
 \end{aligned} \tag{7.641}$$

7.35 Appendix A1: Calculating free oxygen concentration hemoglobin binding kinetics

Oxygen with Hemoglobin. A more complete, but costly computation of oxygen in the vasculature accounts for hemoglobin binding kinetics. The hemoglobin binding kinetics requires a complicated 4-state binding model with state-dependent reaction rates. Moreover, the model requires a biphasic model of blood to account for the red blood cell phase independent of whole blood. Hemoglobin will follow the RBC phase and oxygen will follow the plasma phase in the event of biphasic blood flow. Oxygen-bound hemoglobin (Hb_{bound}) will be modeled by convection through the RBC phase of blood and a reaction to discharge the oxygen into the free plasma. The free oxygen (O) will have a generation term (generated from detachment from hemoglobin) and will be convected through the plasma and transferred through the wall using mass transfer as seen in Equation (7.642).

$$\frac{dc_{Hb}^b}{dt} V_{vasc} = \vec{V} \cdot (f c_{Hb}^b) - r_{hill} \quad (7.642)$$

$$\frac{dc_O}{dt} V_{vasc} = r_{hill} - \vec{V} \cdot (f c_O) + UA_{vasc} \Delta c_O V_{vasc}$$

Where

$$r_{hill} = k_{eff}(c_o - c_o^{eq})V_{vasc} \quad (7.643)$$

The Hill equation (Equation (7.653)) can be used to derive the ratio of bound and total hemoglobin (Equation (7.370)), rendering:

$$r_{hill} = k_{eff} \left(c_o - \left(K_{eq} \frac{\theta}{1 - \theta} \right)^{\frac{1}{n}} \right) V_{vasc}$$

Where (7.644)

$$\theta = \frac{(c_o)^n}{K_{eq} + (c_o)^n} = \frac{1}{1 + \frac{K_{eq}}{(c_o)^n}}$$

This is a nonlinear set of equations, which can either be converged using a fixed point iteration scheme such as the one used in the calculation of biphasic blood flow (Section 7.12). More information on the Hill equation can be found in Section 7.36.

Hydrogen (pH). Hydrogen ions can be included as the metabolic product of oxygen destruction. It will be modeled as a competitive binder to hemoglobin with oxygen (decreases the binding affinity for oxygen). The flux through the tissue uses diffusion, but through vasculature it can be modeled by mass transfer and convection:

$$\frac{dc_{H^+}}{dt} V_{tiss} = \vec{\nabla} \cdot (f c_{H^+}) + U A_{vasc} \Delta c_{H^+} \quad (7.645)$$

$$K_{eq}(H^+) = \overline{k_{eq}} + \alpha c_{H^+} \quad (7.646)$$

Note, Equation (7.646) offers a hydrogen-dependent equilibrium rate. The production of hydrogen involves the bicarbonate reaction sequence as a result of increased aerobic metabolism in the tissue and enters the vasculature through mass transfer. To keep consistent with previous implementation, the equilibrium concentration will use the following definition:

$$k_{eq} = (P_{50})^n \quad (7.647)$$

Which can be substituted back into the Hill equation to become:

$$\theta = \frac{1}{1 + \left(\frac{P_{50}}{P_O}\right)^n} \quad (7.648)$$

The derivation of the equilibrium concentration is given below:

$$\begin{aligned} \theta &= \frac{c_{HbO}}{c_{HbO} + c_{Hb}} = \frac{c_{HbO}}{c_{HbO} + c_O^{RBC}} \\ (1 - \theta) &= \frac{c_{Hb}}{c_{HbO} + c_{Hb}} = \frac{c_O^{RBC}}{c_{HbO} + c_O^{RBC}} \end{aligned} \quad (7.649)$$

Combining k_{eq} and Equation (7.649) gives:

$$\begin{aligned} k_{eq} \frac{\theta}{1 - \theta} &= \frac{c_O^n c_{Hb}}{c_{HbO}} \frac{c_{HbO}/\epsilon_{HbO} + \epsilon_{Hb}}{c_{Hb}/\epsilon_{HbO} + \epsilon_{Hb}} \\ k_{eq} \frac{\theta}{1 - \theta} &= \frac{c_O^n \epsilon_{Hb}}{\epsilon_{HbO}} \frac{\epsilon_{HbO}}{\epsilon_{Hb}} = c_O^n \\ \left(k_{eq} \frac{\theta}{1 - \theta}\right)^{\frac{1}{n}} &= c_O \end{aligned} \quad (7.650)$$

This relates the concentration of oxygen to a series of values that can be looked up in a table (if θ is approximated using the old value as in [21,22]) or converged with fixed-point iteration.

7.36 Appendix AJ: The Hill Equation

The hill equation is an approximation to calculate the saturation of O to Hb. It was developed for understanding reoxygenation in the lungs in [273]. The derivation begins with a reversible reaction where the four hemoglobin (Hb) binding states are assumed to bind as in:



$$k_{eq} = \frac{k_2}{k_1} = \frac{(c_O)^n c_{Hb}}{c_{HbO}} \quad (7.652)$$

A new definition is now introduced as the saturation of O to Hb:

$$\theta = \frac{c_{HbO}}{c_{Hb} + c_{HbO}} \quad (7.653)$$

With substitution, Eq. (7.653) becomes:

$$\begin{aligned} c_{HbO} &= \frac{(c_O)^n c_{Hb}}{k_{eq}} \\ \theta &= \frac{\frac{(c_O)^n c_{Hb}}{k_{eq}}}{c_{Hb} + \frac{(c_O)^n c_{Hb}}{k_{eq}}} = \frac{\frac{(c_O)^n c_{Hb}}{k_{eq}}}{\frac{c_{Hb} k_{eq}}{k_{eq}} + \frac{(c_O)^n c_{Hb}}{k_{eq}}} = \frac{\frac{(c_O)^n c_{Hb}}{k_{eq}}}{\frac{c_{Hb} (k_{eq} + (c_O)^n)}{k_{eq}}} \end{aligned} \quad (7.654)$$

$$= \frac{(c_O)^n \epsilon_{HB}}{k_{eq}} \frac{k_{eq}}{\epsilon_{HB} (k_{eq} + (c_O)^n)}$$

$$\theta = \frac{c_{HbO}}{c_{Hb} + c_{HbO}} = \frac{(c_O)^n}{k_{eq} + (c_O)^n} = \frac{1}{1 + \frac{k_{eq}}{(c_O)^n}}$$

The majority of the computational blood oxygen level community don't use concentrations, but rather gas partial pressure (pO₂). This replaces the equilibrium concentration ratio with a partial pressure at 50% Hb saturation, known as P₅₀. When making this change to the equation, the value is also raised to the power of n :

$$\theta = \frac{1}{1 + \left(\frac{P_{50}}{pO}\right)^n} \quad (7.655)$$

7.36.1 Effects of catalyzed oxygen unbinding on Hill equation

In order to consider the effects of hydrogen competitive binding, the catalysis of the reaction must be evaluated. Considering a reversible reaction:



With an equilibrium constant (K_{eq}):

$$K_{eq} = \frac{c_A^{ss} c_B^{ss}}{c_C^{ss}} = \frac{k_1}{k_2} \quad (7.657)$$

In the case of ligand binding with multiple binding states, such as:



The constant equation accounts for the multiple binding states:

$$K_{eq} = \frac{k_1}{k_2} = \frac{c_{Hb}(c_O)^n}{c_{OHb}} \quad (7.659)$$

If we define the ratio of occupied to total receptor:

$$\theta = \frac{r_{occ}}{r_{tot}} = \frac{c_{OHb}}{c_{Hb} + c_{OHb}} = \frac{\frac{c_{Hb}(c_O)^n}{K_{eq}}}{c_{Hb} + \frac{c_{Hb}(c_O)^n}{K_{eq}}} = \frac{\frac{c_{Hb}(c_O)^n}{K_{eq}}}{\frac{c_{Hb}K_{eq} + c_{Hb}(c_O)^n}{K_{eq}}} \quad (7.660)$$

$$= \frac{c_{Hb}(c_O)^n}{K_{eq}} \frac{K_{eq}}{c_{Hb}K_{eq} + c_{Hb}(c_O)^n}$$

$$\theta = \frac{(c_O)^n}{K_{eq} + (c_O)^n} \rightarrow \text{Hill Eqn} \quad (7.661)$$

Where O is moles of free oxygen, n is number of binding sites on hemoglobin, Hb is moles of hemoglobin, and OHb is moles of bound oxygen to hemoglobin. If the forward reaction is catalyzed, the reaction rate is increased: ($\beta = \alpha/k_2$)



$$\begin{aligned} \frac{k_1 + \alpha}{k_2} &= K_{eq} + \beta \\ &= \frac{c_{Hb}(c_O)^n}{c_{OHb}} \end{aligned} \quad (7.663)$$

The ratio of occupied to total receptor is then defined as:

$$\theta = \frac{r_{bound}}{r_{tot}} = \frac{c_{OHb}}{c_{Hb} + c_{OHb}} = \frac{\frac{c_{Hb}(c_O)^n}{K_{eq} + \beta}}{c_{Hb} + \frac{c_O(c_{Hb})^n}{K_{eq} + \beta}} = \frac{\frac{c_{Hb}(c_O)^n}{K_{eq} + \beta}}{\frac{c_{Hb}(K_{eq} + \beta) + c_{Hb}(c_O)^n}{K_{eq} + \beta}} \quad (7.664)$$

$$= \frac{c_{Hb}(c_O)^n}{(K_{eq} + \beta)} \frac{(K_{eq} + \beta)}{c_{Hb}(K_{eq} + \beta) + c_{Hb}(c_O)^n}$$

$$\theta = \frac{(c_O)^n}{(K_{eq} + \beta) + (c_O)^n} \quad (7.665)$$

Given that k_2 and α are both unknown, the previous equation can be rewritten in terms of a single unknown constant:

$$\theta = \frac{(c_O)^n}{K_{eq}\gamma(H^+) + (c_O)^n}$$

or

(7.666)

$$\theta = \frac{(c_O)^n}{K_{eq} + \gamma(H^+) + (c_O)^n}$$

Where $\gamma(H^+)$ can be any function. The following is recommended: (α will be chosen to keep oxygen within measurements):

$$\gamma(H^+) = \alpha c_{H^+} \quad (7.667)$$

The result of adjusting this value effectively shifts the P_{50} value. An example of binding curve (Hill equation value) with a commonly reported P_{50} value (29.3) and an amended value (35) are compared in Figure 7.242.

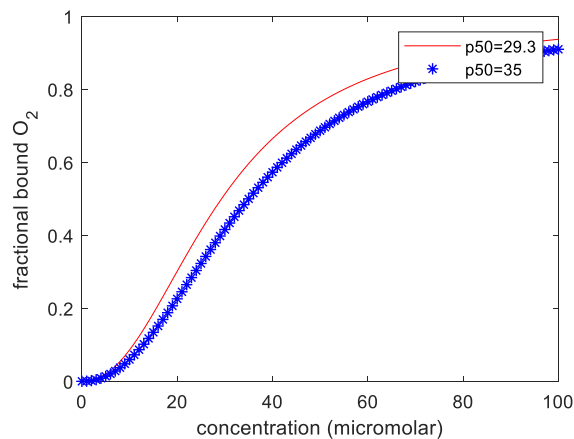


Figure 7.242. Effect of shifting the equilibrium concentration on the results of the Hill equation. A decrease in the equilibrium concentration gives the predicted effect; namely that the concentration of bound oxygen will be lower for the same amount of free oxygen.

7.37 Appendix AK: Models of the neurovascular unit

7.37.1 Idealized networks

The most well-known model of the neurovascular unit is the Krogh cylinder which has numerous implementations. It is commonly used for calculating transport parameters such as CMRO_2 and diffusivity when experimental data is available. An in-depth investigation of the Krogh model was conducted and compared to the discrete models developed by our group previously [62]. This model implies a single vessel inside a finite cylindrical medium with azimuth and axial symmetry (only varying in radial direction). Moreover, this model assumes a 0th order reaction sequence, which is not an accurate method and can lead to underestimating the consumption in the tissue. It also imposes a no-flux boundary condition on the outer edge of the domain. Due to these limitations, this model is not ideal for calculating the oxygen distribution throughout a network.

A group led by Dr. Ress made a simple model to investigate the time course of stimulation-evoked change in oxygen perfusion pressure in cat [274–276]. The electrical surrogate model used a closed voltage loop with a resistor-inductor model to account for large arterial inertia and parallel capacitor-resistor model for the capillary and venous downstream effects. In the model of two resistors, the second resistor models the increased resistance when the arteries are not dilated.

The goal of this model was to replicate dynamic experimental observations of previous studies on cat brains during functional hyperemia. The model normalized all oxygen values to inlet vascular oxygen tension (all oxygen concentration was between 0 and 1, and units for motion were dimensionless). The first resistor is upstream arterial resistance; the second resistor signifies the change in upstream resistance due to dilation. R_2 is the venous resistance, C is the compliance of

the venous system. Q is the concentration of oxygen, and U is the velocity of blood flow. The inductor accounts for the inertia of the blood as it flows into the capillary bed. R_A is the loss through mass transfer and brain metabolism (its value is very large compared to the other resistors). More details are offered regarding R_A , but they are beyond the scope of this review. The parameters for the model were either derived from literature (length, $CMRO_2$, frequency, time constant) or from maximum likelihood estimate for nonlinear parameter estimation using published data.

The model was later endowed with the nonlinear oxygen binding kinetics of the Hill equation and the blood model was advanced to a biphasic medium, yet algebraic maneuvering allowed the reduction of the blood flow back to a single phase for the purposes of deriving the intravascular oxygen content [275,276]. The new model also included a loss term for the endothelial wall which was assumed to metabolize a portion of the oxygen delivered to the tissue.

This model is governed by equations of continuity. The flux balance includes a convection in the vasculature, mass transfer across the endothelial layer, and a metabolic loss term in the tissue. Here, c_v is the concentration of oxygen in the vasculature, c_t is the oxygen concentration in tissue, U is bulk blood flow, Γ is $CMRO_2$ (metabolism of oxygen), A is the mass transfer coefficient, t is time, and z is the axial direction of the blood vessel.

$$\frac{\partial c_v(z, t)}{\partial t} + U(t) \frac{\partial c_v(z, t)}{\partial z} = -A(c_v(z, t) - c_t(z, t)) \quad (7.668)$$

$$\frac{\partial c_t(z, t)}{\partial t} = \alpha A(c_v(z, t) - c_t(z, t)) - \beta \Gamma(t) \quad (7.669)$$

$$\alpha = \frac{r_1^2}{r_2^2 - r_1^2}, \quad \beta = \frac{\alpha}{CBV} \quad (7.670)$$

At steady state (with constant flow, U), the system simplifies to:

$$c_v = 1 - \frac{\beta\Gamma}{\alpha U} z \quad (7.671)$$

$$c_t = 1 - \frac{\beta\Gamma}{\alpha A} z - \frac{\beta\Gamma}{\alpha U} z \quad (7.672)$$

The linear system lends itself to simple linear impulse response functions (corresponding to underdamped, top, critically damped, middle, and overdamped, bottom):

$$U_t = U_{t,0} \begin{cases} e^{-t/\tau} \sin(2\pi f t) \\ te^{-t/\tau} \\ e^{-t/\tau} \sinh(2\pi f t) \end{cases} \quad (7.673)$$

In the case of biphasic blood flow, the governing equations (prior to simplification) are formulated where v is the velocity, z is longitudinal direction of the vessel, vr is volume ratio (as volume per length), hct is hematocrit, pl is plasma, f is a hemoglobin binding function (assumed to be the Hill), t is tissue, A is mass transfer coefficient, Γ is $CMRO_2$ (metabolism of oxygen), bvf is blood volume fraction.

$$\frac{dc_{O_2}^{hct}}{dt} = -\frac{U(z, t)}{vr_{O_2}^{Hct}} - v(t) \frac{dc_{O_2}^{hct}}{dz} \quad (7.674)$$

$$\frac{dc_{O_2}^{pl}}{dt} = \frac{f(z, t)}{vr_{O_2}^{Hct}} - A(c_{O_2}^{pl} - c_{O_2}^{evs}) - v(t) \frac{dc_{O_2}^{pl}}{dz} \quad (7.675)$$

$$\frac{dc_{O_2}^t}{dt} = A \frac{vr_{O_2}^{pl}}{vr_{O_2}^t} (c_{O_2}^{pl} - c_{O_2}^{evs}) - \frac{vr_{O_2}^{hct} + vr_{O_2}^{pl}}{bvft vr_{O_2}^t} \Gamma(t) \quad (7.676)$$

This model has limitations of drastic spatial simplifications and the assumption that the responses are simply impulse response adaption to steady-state deviations. In the brain, however, the states are not fixed (no steady-state value). Overall, due to these limitations, this model predicts only the assumptions by the modeler cannot be used to explore novel theories.

Another approach from the Boas group [277] built a method that uses dynamic measurement data and fits parameters of a model to these measurements. This feedback-feedforward simulation is built on a 3-compartment geometry with a Windkessel model for each compartment.

Where the heart is a constant DC power supply. The system can then be described by a series of partial differential equations:

$$\frac{dV}{dt} = \nabla Q \quad (7.677)$$

$$\Delta P = (R_n + R_{n+1})Q \quad (7.678)$$

$$\frac{dV}{dt} = F_n - F_{n+1} = \frac{\Delta P}{(R_n + R_{n-1})} - \frac{\Delta P}{(R_n + R_{n+1})} \quad (7.679)$$

$$\frac{dV}{dt} = \frac{P_{n-1} - P_n}{(R_{n-1} + R_n)} - \frac{P_n - P_{n+1}}{(R_n + R_{n+1})} \quad (7.680)$$

$$A_n = \frac{V_n(t = 0)}{Q_{n-1,n}(t = 0)(R_{n-1}(t = 0) + R_n(t = 0))^{\left(\frac{1}{\beta_n}\right)}} \quad (7.681)$$

Where Q is volumetric flow rate, V is volume, P is pressure, and R is resistance. The sum of all values of R must be the resistance of the entire cerebral compartment. A_n is the diffusivity and C_n is the capacitance of the compartment. The capacitance (passive dilation) can be defined by:

$$C_n = A_n [P_n]^{(1/\beta_n)-1} \quad (7.682)$$

Which applies only to the capillaries and veins. β_n is the vascular reserve of the compartment, here taken the same for capillaries and veins. This results in:

$$Q_{n-1,n} = \frac{P_{n-1} - V_n(t)^\beta / A^\beta}{R_{n-1}(t) + R_n(t)}$$

$$Q_{n,n+1} = \frac{V_n(t)^{\alpha+\beta}}{A^\beta V_n(0)^\alpha (R_n(0) + R_{n+1}(0))}$$

$$\alpha = 2$$

The arterial dilation is an active dilation mechanism defined by:

$$\frac{V_a(t)}{V_a(0)} = \left(\frac{D_a(t)}{D_a(0)} \right)^2 \quad (7.683)$$

The time integration first updates the diameter, then computes the updated parameters starting in arteries, then capillaries, then veins. The update is given as a change in the resistance vector,

not explicitly changing the diameter or volume parameters. A further investigation into the oxygen transport was offered. The mass transfer for each compartment follows:

$$\begin{aligned}\frac{dc_{MT}}{dt} &= -K(c_{vasc}(t) - c_{tiss}(t)) \\ \frac{dc_{tot}}{dt} &= Q_{in}c_{in} - Q_{out}c_{out} - K\left(\frac{c_{in} - c_{out}}{2} - c_{mt}\right)\end{aligned}\tag{7.684}$$

Where $\frac{c_{in}-c_{out}}{2}$ represents the compartment concentration as a linear average between adjacent compartments. The physiological correctness of this definition is not clear, however, because the majority of driving force in the intravascular compartment is convection, which would make the concentration just c_{in} . This coupled with Equation (7.685) is used to calculate the $CMRO_2$ rate and the K value:

$$\frac{dc_{tot}}{dt} = \sum_n K\left(\frac{c_{in} - c_{out}}{2} - c_{mt}\right) - CMRO_2\tag{7.685}$$

Which expresses mass conservation in the tissue domain, where the $CMRO_2$ affects the entire region which is fed by mass transfer from arteries, capillaries and veins. The values of plasma and tissue saturation then follows by inverting:

$$\frac{dnO_{tot}}{dt} = \alpha p_{O_2} + Hc_{Hb}S_{O_2}\tag{7.686}$$

Where H is the Hufner number (oxygen bound per gram of Hb), S is saturation of oxygen (claimed to use the Kelman's equation, described next), and α is the oxygen solubility in plasma (Henry's Law)).

The Kelman equation was derived in 1979 to expand the Hill equation for more verbose use of point-measurements and inversion. The functions for resistance modification and $CMRO_2$ changes are implied by gamma distributions. A resistance model is fitted to the data dynamically during simulation and the $CMRO_2$ signal is given explicitly and numerically integrated.

Overall, the goal was to use a model to determine more experimental parameters than usually measured. Unfortunately, the model was simplified to a degree where the predicted (estimated) values are not very trustworthy and could be incorrect by more than an order of magnitude based on the simplicity. An example is vascular simplicity, which can cause results to vary wildly when diameter and resistance are key parameters used for estimation, where a single vessel is not applicable towards a complex microcirculatory environment.

The experimental results given in the paper show a time-course for increased flow, increased oxyHb and deoxyHb. One of the key findings was that a multi-compartment model is better than a single-compartment model, however whether this was simply due to more degrees of freedom or due to an intrinsically better model was not discussed.

Another geometrically simplified model generated by the Buxton group is the balloon model. This model focuses on accounting for dynamic changes in deoxygenated hemoglobin [278,279]. Due to the focus of this model, the dynamic equations are not generated from first-principles of transport phenomena, but are rather ad-hoc equations. Because the model does not rely on physiologically consistent geometry or transport equations, it will not be discussed in detail.

An additional model, designed for studying the effect of capillary transit time on oxygen tension, uses a gamma function for the distribution of transit time for oxygen through capillary vessels [280]. The assumed geometry is a perfectly symmetric series of parallel vascular elements with a single feeding vessel and a single draining vein. The governing equations were based on physical simplified mass balances, however the transport equations are not presented in differential form. The finding indicates an increase in capillary heterogeneity is a function of transit time, however the simplified physics of the system and simplified geometry calls this finding into question. In other words, it is not clear that this finding is a result of the model or rather the assumptions from the modeler. This model will also not be discussed in detail.

A later publication by the Boas group used an idealized network to investigate a more distributed response to functional hyperemia [267]. The group builds the Hagen Poiseuille (HP) equations using indexing, intended to bring this type of analysis to a wider audience. The blood follows HP physics of linear flow with Dirichlet BCs. The network was constructed from some crude physiological statistics to make a symmetric hierarchical graph, where length and diameters were assigned as literature constants for each hierarchy. The penetrating vessels have a single value for diameter, the precapillary arterioles are given diameters from a fractal pattern, where each branch decreases the diameter by 20%. The capillaries are set to a fixed diameter and length.

The goal of this model is to dilate some arterial vessels, allow a crude FSI-inspired vessel dilation downstream, and allow the inertia to dilate the capillaries and post-capillary venules in the network. This can be considered the *arterial dilation* model for functional hyperemia:

$$p_i = p_{IC} + \left(\frac{V_i(t)}{A_{0,i}} \right)^\beta$$

$$A_{0,i} = \frac{V_i(t=0)}{(p_i(t=0) - p_{IC})^{1/\beta}} \quad (7.687)$$

Where β is a constant (here chosen to be 2, but in a later paper chosen to be 1), V is the volume of the i^{th} vessel, and p_{IC} is the intracranial pressure. The oxygen in the vasculature was calculated using inlet blood saturation at inlet minus the outlet blood saturation. The interior vessels are later calculated using a linear averaging between in the inlet and outlet saturation. The oxygen is convected through the vasculature and undergoes mass transfer into the tissue where it is consumed. Interestingly, the partial pressure is converted to concentration using the Bunsen solubility coefficient. Note, the tissue oxygen tension is considered constant and set to 15 mmHg.

The $CMRO_2$ follows an increase that the dilation takes. Both are modeled by a dynamic input function, namely:

$$R_i(t) = R_{i,0} \left(1 - \Delta R_i \frac{H(t-\tau)^2 e^{-\left(\frac{(t-\tau)}{\sigma_w^2}\right)^2}}{\sigma_w^2 e^{-1}} \right) \quad (7.688)$$

$$CMRO_2(t) = CMRO_{2,0} \left(1 + \Delta CMRO_2 \frac{H(t-\tau)^2 e^{-\left(\frac{(t-\tau)}{\sigma_w^2}\right)^2}}{\sigma_w^2 e^{-1}} \right) \quad (7.689)$$

Where R is the i^{th} vessel resistance, H is the Heaviside function ($=0$ when $t < 0$ and $=1$ otherwise), σ_w is the width of the dilation duration (in seconds, here set to 1 s) and ΔR_i is the magnitude of the dilation, here set to 0.05. The result is an increase in flow in the local blood flow and oxygen in the dilated vessels, downstream vessels and, interestingly, a decrease in blood flow and oxygen for nearby vessels.

7.37.2 Realistic vasculature

Green function. Another approach that uses reconstructed vasculature from a frog toe webbing and cremaster muscle by Secomb uses a Green function [31,164,281]. This method uses an analytically-derived function for the oxygen concentration throughout the tissue as a function of the network architecture, using the network as a series of discrete infinitely-small point sources. In summary, the Green function solves oxygen within a finite domain, enforcing no flux boundary conditions along the surface (walls) of the domain, and distributing the oxygen source along the surface of each vessel (as opposed to point sources, although in reality point sources were actually implemented to solve the equations).

The Green function draws on many assumptions. These include:

- Metabolic rate is modeled as a 0th order reaction and is homogenous
- Flow of oxygen must be from vasculature to tissue (no resorption possible)
- Diffusivity is isotropic and constant
- Vasculature is perfectly cylindrical segments
- (only original publication) each vessel has uniform radius, flow, oxygen pressure and full hemoglobin saturation (Hill equation)

- Oxygen in plasma is negligible (all dissociation from Hb goes directly through BBB)
- Oxygen concentration can be converted to partial pressure through Henry's law (full saturation of oxygen in plasma)
- The concentration profile generated from every point source acts independent of other point sources (we know this to not be true, 2 point sources will influence local distribution not necessarily linearly)
- There is no mass transfer, it is assumed that the concentration in the plasma is the source concentration
- Many point sources can accurately capture the physics of an analytic line

The main governing equations in the extravascular space follow 0th order reaction and diffusion:

$$M = D\nabla^2 c \quad (7.690)$$

Where D is diffusivity, assumed isotropic, c is concentration of oxygen, and M is metabolic demand (loss due to reaction). M is a constant in this case, due to the above mentioned simplification on the reaction rate. The extravascular space boundary insists on a no-flux boundary condition:

$$\frac{dc}{dl} = 0 \quad (7.691)$$

Where \underline{l} is the length between the source and the boundary. Converting between partial pressure and concentration uses Henry's law (another improper assumption that assumes full saturation, see Section 7.34.2 for more information):

$$c = \alpha P_{O_2} \quad (7.692)$$

Oxygen saturation in blood is modeled using the Hill equation:

$$c_{O_2}^{vasc} = c_s f\left(\frac{p_{O_2}^{vasc}}{p_{50}}\right) \quad \text{Where} \quad f(x) = \frac{x^n}{1+x^n} \quad (7.693)$$

Because of radial symmetry, the concentration at the boundary between tissue and vasculature can be evaluated with a surface integral:

$$c(z) = \frac{1}{2\pi} \int_0^{2\pi} c \, d\theta \quad (7.694)$$

The Green function is decomposed into 3 sub-equations that will together satisfy the following conditions:

1. A point source generates a Gaussian distribution with the peak at the source point
2. The divergence within each node is equal to the metabolic loss divided by the volume (M/V, where M is assumed 1)
3. The edge of the boundary (each face of the cube) has 0 net flux across it
4. Every point on the extravascular boundary has 0 net flux across it

In order to satisfy these constraints, 3 green's functions were created (G1-G3):

- G1. Inspired by a Gaussian distribution in 3 dimensions, giving the proper shape profile to the concentration (criteria 1).
- G2. Chosen to ensure the net flux across each boundary face (faces of domain cube) constitute a net flux of 0 (integral flux is 0 on every edge, criteria 3) and that the divergence of the flux at each point is equal to the loss (Equation (7.691) and criteria 4).
- G3. An infinite double-series that satisfies the divergence-free equation (criteria 2) and that the flux at every point on the boundary edge is 0 (criteria 4). This is not enforced by G₂ which only enforces the *net* flux across each boundary edge is 0.

The next set of equations will build such a Green function. For simplicity, the equations have been written in general form:

$$G = G_1 + G_2 + G_3 \quad (7.695)$$

$$G_1 = \frac{1}{4 * \pi |x - x^*|} \quad (7.696)$$

$$G_2 = C_1 x_1 + C_2 x_2 + C_3 x_3 + D_1 x_1^2 + D_2 x_2^2 + D_3 x_3^2 \quad (7.697)$$

$$C_1 = -\frac{1}{l_2 l_3} F(l_2, l_3, x_2^*, x_3^*, x_1^*) \quad (7.698)$$

$$C_2 = -\frac{1}{l_1 l_3} F(l_1, l_3, x_1^*, x_3^*, x_2^*)$$

$$C_3 = -\frac{1}{l_1 l_2} F(l_1, l_2, x_1^*, x_2^*, x_3^*)$$

$$D_1 = \frac{1}{2l_1 l_2 l_3} (F(l_2, l_3, x_2^*, x_3^*, x_1^*) + F(l_2, l_3, x_2^*, x_3^*, l_1 - x_1^*))$$

$$D_2 = \frac{1}{2l_1 l_2 l_3} (F(l_1, l_3, x_1^*, x_3^*, x_2^*) + F(l_1, l_3, x_1^*, x_3^*, l_2 - x_2^*))$$

$$D_3 = \frac{1}{2l_1 l_2 l_3} (F(l_1, l_2, x_1^*, x_2^*, x_3^*) + F(l_1, l_2, x_1^*, x_2^*, l_3 - x_3^*))$$

$$\begin{aligned} G_3 = & \sum_0^\infty \sum_0^\infty \left[A_{m,n} \cosh(k_{m,n}(x_1 - l_1)) \right. \\ & + B_{m,n} \cosh(k_{m,n}x_1) \left. \right] \cos\left(\frac{m\pi x_2}{l_2}\right) \cos\left(\frac{n\pi x_3}{l_3}\right) \\ & + \left[A'_{m,n} \cosh(k'_{m,n}(x_2 - l_2)) \right. \\ & + B'_{m,n} \cosh(k'_{m,n}x_2) \left. \right] \cos\left(\frac{m\pi x_3}{l_3}\right) \cos\left(\frac{n\pi x_1}{l_1}\right) \\ & + \left[A''_{m,n} \cosh(k''_{m,n}(x_3 - l_3)) \right. \\ & + B''_{m,n} \cosh(k''_{m,n}x_3) \left. \right] \cos\left(\frac{m\pi x_1}{l_1}\right) \cos\left(\frac{n\pi x_2}{l_2}\right) \end{aligned} \quad (7.699)$$

$$k_{m,n} = \pi \left(\frac{m^2}{l_2^2} + \frac{n^2}{l_3^2} \right)^{\frac{1}{2}} \quad (7.700)$$

$$k'_{m,n} = \pi \left(\frac{m^2}{l_3^2} + \frac{n^2}{l_1^2} \right)^{\frac{1}{2}} \quad (7.701)$$

$$k''_{m,n} = \pi \left(\frac{m^2}{l_1^2} + \frac{n^2}{l_2^2} \right)^{1/2} \quad (7.702)$$

$$A_{m,n} = \frac{4}{l_2 l_3 k_{m,n} \sinh(k_{m,n} l_1)} \int_0^{l_2} \int_0^{l_3} \frac{\partial}{\partial x_1} (G_1 + G_2) \Big|_{x_1=0} \cos\left(\frac{m\pi x_2}{l_2}\right) \cos\left(\frac{n\pi x_3}{l_3}\right) dx_2 dx_3 \quad (7.703)$$

$$A'_{m,n} = \frac{4}{l_1 l_3 k'_{m,n} \sinh(k'_{m,n} l_2)} \int_0^{l_1} \int_0^{l_3} \frac{\partial}{\partial x_2} (G_1 + G_2) \Big|_{x_2=0} \cos\left(\frac{m\pi x_3}{l_3}\right) \cos\left(\frac{n\pi x_1}{l_1}\right) dx_1 dx_3 \quad (7.704)$$

$$A''_{m,n} = \frac{4}{l_2 l_1 k''_{m,n} \sinh(k''_{m,n} l_3)} \int_0^{l_1} \int_0^{l_2} \frac{\partial}{\partial x_3} (G_1 + G_2) \Big|_{x_3=0} \cos\left(\frac{m\pi x_1}{l_1}\right) \cos\left(\frac{n\pi x_2}{l_2}\right) dx_1 dx_2 \quad (7.705)$$

$$B_{m,n} = \frac{4}{l_2 l_3 k_{m,n} \sinh(k_{m,n} l_1)} \int_0^{l_2} \int_0^{l_3} \frac{\partial}{\partial x_1} (G_1 + G_2) \Big|_{x_1=l_1} \cos\left(\frac{m\pi x_2}{l_2}\right) \cos\left(\frac{n\pi x_3}{l_3}\right) dx_2 dx_3 \quad (7.706)$$

$$B'_{m,n} = \frac{4}{l_1 l_3 k'_{m,n} \sinh(k'_{m,n} l_2)} \int_0^{l_1} \int_0^{l_3} \frac{\partial}{\partial x_2} (G_1 + G_2) \Big|_{x_2=l_2} \cos\left(\frac{m\pi x_3}{l_3}\right) \cos\left(\frac{n\pi x_1}{l_1}\right) dx_1 dx_3 \quad (7.707)$$

$$B''_{m,n} = \frac{4}{l_2 l_1 k''_{m,n} \sinh(k''_{m,n} l_3)} \int_0^{l_1} \int_0^{l_2} \frac{\partial}{\partial x_3} (G_1 + G_2) \Big|_{x_3=l_3} \cos\left(\frac{m\pi x_1}{l_1}\right) \cos\left(\frac{n\pi x_2}{l_2}\right) dx_1 dx_2 \quad (7.708)$$

In order to evaluate G_3 , the differentiation of G_1 and G_2 is required. The linear combination of the G_1 and G_2 inside or outside the function is irrelevant, so for ease of understanding, the functions will be broken up. First, given that G_2 results in a scalar and it would not make sense for one G function to be a vector and the other to be a scalar. The vertical bars in G_1 will be assumed to stand for the norm of $(x-x^*)$. Before taking the derivative, this will be written in the Euclidean form, using chain rule:

$$g_1 = \frac{1}{4 * \pi |x - x^*|} \quad (7.709)$$

$$\frac{\partial g_1}{\partial x} = \frac{1}{4\pi} \frac{-1}{2} 2(x - x^*) [(x - x^*)^2 + (y - y^*)^2 + (z - z^*)^2]^{-3/2} \quad (7.710)$$

$$\frac{\partial g_1}{\partial x} = - \frac{(x - x^*)}{4\pi [(x - x^*)^2 + (y - y^*)^2 + (z - z^*)^2]^{3/2}}$$

$$= \frac{(x^* - x)}{4\pi [(x - x^*)^2 + (y - y^*)^2 + (z - z^*)^2]^{3/2}}$$

$$\frac{\partial g_1}{\partial y} = - \frac{(y - y^*)}{4\pi [(x - x^*)^2 + (y - y^*)^2 + (z - z^*)^2]^{3/2}}$$

$$= \frac{(y^* - y)}{4\pi [(x - x^*)^2 + (y - y^*)^2 + (z - z^*)^2]^{3/2}}$$

$$\frac{\partial g_1}{\partial z} = - \frac{(z - z^*)}{4\pi [(x - x^*)^2 + (y - y^*)^2 + (z - z^*)^2]^{3/2}}$$

$$= \frac{(z^* - z)}{4\pi [(x - x^*)^2 + (y - y^*)^2 + (z - z^*)^2]^{3/2}}$$

Evaluating at key locations (0 and L) an example is created in variable x:

$$\begin{aligned}\left.\frac{\partial G_1}{\partial x}\right|_{x=0} &= \frac{x^*}{4\pi[(-x^*)^2 + (y - y^*)^2 + (z - z^*)^2]^{\frac{3}{2}}} \\ \left.\frac{\partial G_1}{\partial x}\right|_{x=l_1} &= \frac{x^* - l_1}{4\pi[(l_1 - x^*)^2 + (y - y^*)^2 + (z - z^*)^2]^{\frac{3}{2}}}\end{aligned}\quad (7.711)$$

These equations can now be integrated over y and z dimensions as (evaluated using Matlab symbolic expressions):

$$\int_0^{l_2} \left.\frac{\partial}{\partial x_1}(G_1)\right|_{x_1=0} \cos\left(\frac{m\pi x_2}{l_2}\right) \cos\left(\frac{n\pi x_3}{l_3}\right) dy \quad (7.712)$$

Where a, b, and c are the coordinates of x^*

$$\begin{aligned}\int_0^{l_2} \int_0^{l_3} \left.\frac{\partial}{\partial x_1}(G_1)\right|_{x_1=0} \cos\left(\frac{m\pi x_2}{l_2}\right) \cos\left(\frac{n\pi x_3}{l_3}\right) dy dz &= \\ &= \frac{x^*}{4\pi} \int_0^{l_2} \int_0^{l_3} [(-x^*)^2 + (y - y^*)^2 \\ &\quad + (z - z^*)^2]^{-\frac{3}{2}} \cos\left(\frac{m\pi x_2}{l_2}\right) \cos\left(\frac{n\pi x_3}{l_3}\right) dy dz\end{aligned}\quad (7.713)$$

These integrals are evaluated numerically as no analytic solution is available. Evaluation of this integral is admittedly the most time-consuming part of solving the problem. And G_2 is simply (for example variable x_1):

$$G_2 = C_1 x_1 + C_2 x_2 + C_3 x_3 + D_1 x_1^2 + D_2 x_2^2 + D_3 x_3^2 \quad (7.714)$$

$$\frac{\partial G_2}{\partial x_1} = C_1 + 2D_1 x_1 \quad (7.715)$$

Evaluated at important points are:

$$\begin{aligned}\frac{\partial}{\partial x_1}(G_2)\Big|_{x_1=0} &= C_1 \\ \frac{\partial}{\partial x_1}(G_2)\Big|_{x_1=l_1} &= C_1 + 2D_1 l_1\end{aligned}\tag{7.716}$$

Which are now integrated (using Matlab), given that C_1 and D_1 are not dependent on x_2 or x_3 :

$$\int_0^{l_3} \frac{\partial}{\partial x_1}(G_2)\Big|_{x_1=0} \cos\left(\frac{m\pi x_2}{l_2}\right) \cos\left(\frac{n\pi x_3}{l_3}\right) dy = \frac{b \cos\left(\frac{\pi n z}{l_3}\right) \sin\left(\frac{\pi m y}{l_2}\right) (C_1)}{m\pi}\tag{7.717}$$

$$\begin{aligned}\int_0^{l_2} \int_0^{l_3} \frac{\partial}{\partial x_1}(G_2)\Big|_{x_1=0} \cos\left(\frac{m\pi x_2}{l_2}\right) \cos\left(\frac{n\pi x_3}{l_3}\right) dy dz \\ = \frac{bc \sin\left(\frac{\pi m y}{l_2}\right) \sin\left(\frac{\pi n z}{l_3}\right) (C_1)}{mn\pi^2}\end{aligned}$$

$$\begin{aligned}\int_0^{l_3} \frac{\partial}{\partial x_1}(G_2)\Big|_{x_1=l_1} \cos\left(\frac{m\pi x_2}{l_2}\right) \cos\left(\frac{n\pi x_3}{l_3}\right) dy \\ = \frac{b \cos\left(\frac{\pi n z}{l_3}\right) \sin\left(\frac{\pi m y}{l_2}\right) (C_1 + 2D_1 x)}{m\pi}\end{aligned}\tag{7.718}$$

$$\begin{aligned}\int_0^{l_2} \int_0^{l_3} \frac{\partial}{\partial x_1}(G_2)\Big|_{x_1=l_1} \cos\left(\frac{m\pi x_2}{l_2}\right) \cos\left(\frac{n\pi x_3}{l_3}\right) dy dz \\ = \frac{bc \sin\left(\frac{\pi m y}{l_2}\right) \sin\left(\frac{\pi n z}{l_3}\right) (C_1 + 2D_1 x)}{mn\pi^2}\end{aligned}$$

Case study. An example is offered with sources between at $[0,0.95,0.95]^T$ to $[0.95,0.95,0.95]^T$ and domain from the origin to $[3,3,3]^T$ (Note, the solution cannot be sampled at the source point, it will break the algorithm because the values approach infinity). Note, the concentration becomes

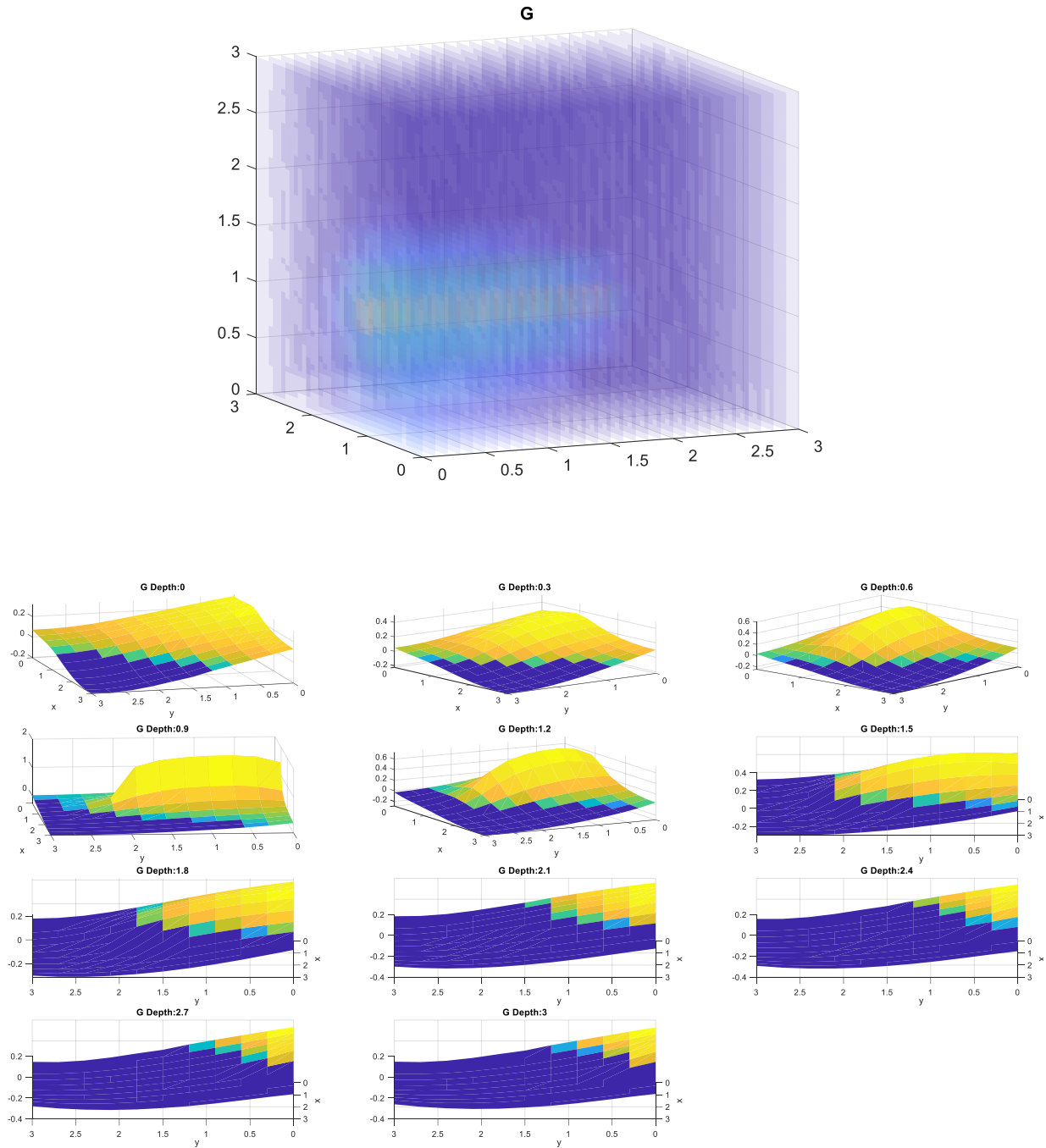


Figure 7.243. Implementation of Green's function with a short segment (Yellow region) that terminates midway through the domain.

negative at the farthest extremities, this is the origin of the G_0 term in the Green's function reconstruction; it offsets the concentration so that it no longer becomes negative. Secomb never discloses how he arrives at a value for G_0 , so it is assumed to be arbitrarily chosen to result in a reasonable lowest concentration value.

1D-3D Coupling. Some further work from Quarteroni [32,33] proposed a method similar to the Green function but with numerical implementation of boundary conditions, greatly simplifying the equations. This method uses the centerline of each vessel whose flux is computed using a finite cylinder but acts as an infinitely thin line source term. Unfortunately, this decoupling of the problem resulted in a singularity along the proposed line sources (point sources in Green function suffer from the same drawback). This method was proposed to couple 1D networks and 3D meshes in the context of water transport in the brain.

$$\begin{cases} -\nabla \cdot (k \nabla u) + q - \hat{f} \delta_\Lambda = 0 & \text{in } \Omega \\ -\frac{d}{ds} \left(\hat{k} \frac{d\hat{u}}{ds} \right) + \hat{f} & \text{in } \Lambda \end{cases}$$

$$\text{Where } \hat{u} = p_{vasc}, \hat{k} = \alpha_{HP}, k = D_{darcy} \text{ and } u = p_{tiss} \quad (7.719)$$

$$\hat{f} = \beta(\hat{u} - \bar{u})$$

\bar{u} is the arithmetic mean of $u(r)$ at $r = R$, and R is the radius of the vessel. Here, Ω is the tissue space and Λ is the infinitely-thin line source that represents the vasculature. $\hat{f} \delta_\Lambda$ is the Dirac delta function scaled by the value of mass density \hat{f} , termed the volume of blood flowing into the tissue per unit time and unit length (mL/min/mm), which is integrated over the length of the vessel

to give a flow rate. Initial boundary conditions are insulated at the tissue boundary, Dirichlet value of 0 at vessel outlet, and fixed flux inlet.

A decoupled problem is also proposed in which the vasculature values are known. Substantial work was made to prove that a solution does exist using this type of splitting, which will not be reviewed here. The largest drawback to this method is that the singularity discovered with this decoupling method is a side effect of an incorrect vascular representation; namely that vessels in real life have a finite non-zero thickness which cannot be ignored. The singularity is generated only from the modeling choice to neglect the vessel thickness and impose line sources.

While this method is convenient for having a semi-analytic solution that is mesh independent, it is computationally expensive and cannot be performed for large structures such as the capillary bed. Moreover, the iterative nature of the coupling between the tissue and the vasculature, although proven stable [33], is too time- and computationally- intensive for practical use.

Analytic applications to dynamic networks. The Payne group designed another approach using realistic structures with analytic functions. This model used realistic capillary networks and Laplace transforms to predict the time-dependent response to changes in blood flow impulses [282]. In this paper, the network topology was synthesized matching statistics reported by Cassot et al. [42]. The synthesis was derived using a previous manuscript [54]. The tissue is assumed to have diffusion, but have negligible concentration (the tissue is assumed to deplete the oxygen very rapidly such that accumulation is nearly 0). The oxygen in the vasculature is considered constant (an assumption that allows the analytic solution to be obtained). The dynamic conservation equation follows:

$$\frac{\partial c_v}{\partial t} = -Q \frac{\partial c_v}{\partial x} - UA(c_v - c_t) \quad (7.720)$$

Where c_v is the vascular concentration, c_t is the tissue concentration, U is the transmembrane mass transfer coefficient, Q is the bulk volumetric blood flow rate, t is time, x is the axial direction, and A is the surface area of the each segment in the 1D network .Given that c_{tiss} is negligible (value of 0), this reduces to:

$$\frac{\partial c_v}{\partial t} = -Q \frac{\partial c_v}{\partial x} - UA c_v \quad (7.721)$$

The manuscript took the Laplace transform of this PDE assuming at time $t=0$, the concentration everywhere is 0. This can be interpreted as computing the deviation from a steady-state value. The Laplace transform follows:

$$c(x, s) = c(0, s) e^{-\frac{(s+UA)x}{Q}} \quad (7.722)$$

The inlet oxygen is considered a unit step function (value=0 at time < 0 and value = 1 for time > 0). The magnitude is A such that $c(0,s)=A/s$. This gives the inverse Laplace of the outlet as:

$$c(L, s) = Au \left(t - \frac{L}{Q} \right) e^{-\frac{UAL}{Q}} \quad (7.723)$$

Where $u\left(t - \frac{L}{Q}\right)$ is the unit step function . This can be interpreted as the outlet concentration is the inlet function with a time offset of L/Q and a magnitude offset of $e^{-\frac{UAL}{Q}}$. The same Laplace and inverse Laplace transforms were performed on the mass balances at each node to define analytic function of time progression for the entire system. The overall loss was considered the oxygen extraction fraction (OEF). The findings primarily pointed out that the mean transit time of the network is constant irrespective of the two network sizes, however there was a decrease in OEF with an increase in average deviation of transit time (more heterogeneous transit time leads to less OEF). This stands loosely in agreement with Ostergaard [280] although earlier simulations by the same group later stood in direct opposition of the Ostergaard group [283].

While this simulation is attractive as an analytic solution, it relies on imperative assumptions regarding tissue and vascular concentration of oxygen that disallow investigation of network effects and organ wide trends that would exist in the real brain.

Another group generated a body-fitted 3D triangulated mesh of the tissue space [25]. The first manuscript from this group aimed at reconstructing microvasculature in a network and performing a transient oxygen simulation on that network and surrounding tissue. The tissue was simplified to a single 0th order reaction term in the blood transport equations:

$$\frac{\partial c_t}{\partial t} = v\nabla c_f - v\nabla c_b + \nabla \cdot D_{O_2}\nabla c_f - OC$$

Where v is blood velocity, c_f and c_b are free and bound oxygen concentration, OC is oxygen consumption, D is diffusivity and c_t is total concentration.

$$c_b = 4c_{hb}Hct \cdot \alpha(c_f) \quad (7.724)$$

Where c_{hb} is the hemoglobin concentration per RBC, Hct is the hematocrit in a given segment, and α is the hemoglobin saturation as a function of free oxygen. The deconstruction is a dual-mesh technique similar to our own:

Vasculature:

$$\frac{\partial c_b}{\partial t} = v \nabla c_b \quad \frac{\partial c_f}{\partial t} = v \nabla c_f \quad (7.725)$$

Wall:

$$\frac{\partial c_f}{\partial t} \cdot V = JA \quad J = \frac{K(c_{f,v} - c_{f,t})}{\alpha w}$$

Where V is the volume of the network node, α is the Bunsen solubility coefficient ($1.27 \text{ e-}15 \text{ } \mu\text{mol}/\mu\text{m}^3/\text{mmHg}$), w is the wall thickness. $c_{f,v}$ is the free oxygen in the vasculature, subscript t refers to the tissue domain:

$$\frac{dc_{f,t}}{dt} = \nabla \cdot (D_{O_2} \nabla c_{f,t}) - OC \quad (7.726)$$

It was also noted that the convection, due to the nonlinear nature, is solved independent of the coupled wall and tissue mesh. This is also advantageous as there is no direct way to couple a 1D to a 3D mesh, however the wall and tissue meshes are both triangulated 3D meshes, making this coupling more direct.

The tissue BCs are chosen to be reflective (no flux). The vasculature uses Neumann inflow BCs and sink efflux terms. The vasculature was segmented semi-manually through a custom process developed for this manuscript. The terminals of the vasculature are manually identified for boundary condition assignment. A Dirichlet BC is used on arteriole inlets and venous outlets, however terminals that do not connect to draining veins are set to a no flux (velocity=0). The blood flow is then solved in this network to be used for the convection in previously described oxygen simulation.

The results are converted from concentration to partial pressure using the Bunsen equation with above reported Bunsen solubility coefficient. The Bunsen solubility coefficient was chosen instead of the Henry equation due to the increased complexity of the biological tissue over a simple mixture implied by the Henry equation.

The group then ran a parametric study evaluating the effect of inlet velocity, hematocrit and oxygen consumption rate on the oxygen profile along a single vessel. They noted a higher velocity and hematocrit increases the oxygen concentration in and around the vessel towards the venous outlet. A decrease in oxygen consumption rate also increased the oxygen concentration. The model investigated different analytic solutions for the independent discrete time integration of convection, mass transfer, and diffusion. The group acknowledged a mesh dependence in the result. The results were also compared to experimental results from [284].

This model does have some limitations. For instance, these equations imply immediate saturation of oxygen between each vessel. This has previously been deemed physiologically inaccurate by the LPPD group [84]. The form of the Hill equation used for the oxygen saturation is an invertible version that was proposed by [285]. Note, the vessel wall is comprised of an

infinitely thin triangulated surface mesh which does not account for the endothelial layer thickness and resistivity.

Moreover, the structure is incomplete and suffers from poor boundary condition choice (no flux on dangling vascular terminals) implies that these segments are fully collapsed, which is not true in vivo. To align the diffusion only simulation to the analytic solution, it was necessary to cut the diffusivity in half and double the time span, something that can be interpreted as poor time-space discretization. The reported maximum velocity was 10 mm/s with a minimum ~1 mm/s in the capillaries. The differences between calculated and experimental results were attributed to the incomplete vascular network which in turn was attributed to limited spatial resolution.

It is also revealed that the 2 vasculature conservation equations are solved, then the saturation equation, then the tissue mesh. Whether the equations are iteratively refined until converged or just simply forward integrated was not disclosed. It was important to note the equations for vasculature were solved prior to enforcing the equilibrium saturation equation. The tissue diffusion-rxn system is then updated with the Galerkin's method of using basis vectors for each tetrahedron and solving the surface integral using these base vectors and the facet normals. The writing of these equations makes them seem de-coupled, however each system is coupled to each other system, so the ambiguity it difficult to decipher exactly how the time integration is performed.

The group did, however, go on to investigate the spatially-relevant models for oxygen distribution, dynamic oxygen extraction during functional hyperemia, and even produced a simulated BOLD signal using proton spin analysis [23,24,85].

7.37.3 Further information: building of the Greens functional parts in full

$$g = g_1 + g_2 + g_3 \quad (7.727)$$

$$g_1 = \frac{1}{4 * \pi |x - x^*|} \quad (7.728)$$

$$g_2 = C_1 x_1 + C_2 x_2 + C_3 x_3 + D_1 x_1^2 + D_2 x_2^2 + D_3 x_3^2 \quad (7.729)$$

$$C_1 = -\frac{1}{4\pi l_2 l_3} \left[\tan^{-1} \left\{ \frac{(l_2 - x_2^*)(l_3 - x_3^*)}{x_1 [x_1^2 + (l_2 - x_2^*)^2 + (l_3 - x_3^*)^2]^{1/2}} \right\} \right. \quad (7.730)$$

$$+ \tan^{-1} \left\{ \frac{(l_2 - x_2)x_3}{x_1 [x_1^2 + (l_2 - x_2)^2 + x_3^{*2}]^{1/2}} \right\}$$

$$+ \tan^{-1} \left\{ \frac{x_2(l_3 - x_3)}{x_1 [x_1^2 + x_2^2 + (l_3 - x_3^*)^2]^{1/2}} \right\}$$

$$+ \tan^{-1} \left\{ \frac{x_2 x_3}{x_1 [x_1^2 + x_2^2 + x_3^{*2}]^{1/2}} \right\} \Bigg]$$

$$C_2 = -\frac{1}{4\pi l_1 l_3} \left[\tan^{-1} \left\{ \frac{(l_1 - x_1)(l_3 - x_3^*)}{x_2 [(l_1 - x_1)^2 + x_2^2 + (l_3 - x_3^*)^2]^{1/2}} \right\} \right. \quad (7.731)$$

$$+ \tan^{-1} \left\{ \frac{(l_1 - x_1)x_3^*}{x_2 [(l_1 - x_1)^2 + x_2^2 + x_3^{*2}]^{1/2}} \right\}$$

$$+ \tan^{-1} \left\{ \frac{x_1(l_3 - x_3)}{x_2 [x_1^2 + x_2^2 + (l_3 - x_3^*)^2]^{1/2}} \right\}$$

$$+ \tan^{-1} \left\{ \frac{x_2 x_3}{x_2 [x_1^2 + x_2^2 + x_3^{*2}]^{1/2}} \right\} \Bigg]$$

$$C_3 = -\frac{1}{4\pi l_1 l_2} \left[\tan^{-1} \left\{ \frac{(l_1 - x_1)(l_2 - x_2)}{x_3 [(l_1 - x_1)^2 + (l_2 - x_2)^2 + x_3^{*2}]^{1/2}} \right\} \right. \quad (7.732)$$

$$+ \tan^{-1} \left\{ \frac{(l_1 - x_1)x_2}{x_3 [(l_1 - x_1)^2 + x_2^2 + x_3^{*2}]^{1/2}} \right\}$$

$$+ \tan^{-1} \left\{ \frac{x_1(l_2 - x_2)}{x_3 [x_1^2 + (l_2 - x_2)^2 + x_3^{*2}]^{1/2}} \right\}$$

$$+ \tan^{-1} \left\{ \frac{x_2 x_3}{x_3 [x_1^2 + x_2^2 + x_3^{*2}]^{1/2}} \right\} \Bigg]$$

$$D_1 = \frac{1}{8\pi l_1 l_2 l_3} \left[\tan^{-1} \left\{ \frac{(l_2 - x_2)(l_3 - x_3^*)}{x_1 [x_1^2 + (l_2 - x_2)^2 + (l_3 - x_3^*)^2]^{1/2}} \right\} \right. \quad (7.733)$$

$$+ \tan^{-1} \left\{ \frac{(l_2 - x_2)x_3^*}{x_1 [x_1^2 + (l_2 - x_2)^2 + x_3^{*2}]^{1/2}} \right\}$$

$$+ \tan^{-1} \left\{ \frac{x_2(l_3 - x_3^*)}{x_1 [x_1^2 + x_2^2 + (l_3 - x_3^*)^2]^{1/2}} \right\}$$

$$+ \tan^{-1} \left\{ \frac{x_2 x_3^*}{x_1 [x_1^2 + x_2^2 + x_3^{*2}]^{1/2}} \right\}$$

$$+ \tan^{-1} \left\{ \frac{(l_2 - x_2)(l_3 - x_3^*)}{(l_1 - x_1)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + (l_3 - x_3^*)^2]^{1/2}} \right\}$$

$$+ \tan^{-1} \left\{ \frac{(l_2 - x_2)x_3^*}{(l_1 - x_1)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + x_3^{*2}]^{1/2}} \right\}$$

$$+ \tan^{-1} \left\{ \frac{x_2(l_3 - x_3^*)}{(l_1 - x_1)[(l_1 - x_1)^2 + x_2^2 + (l_3 - x_3^*)^2]^{1/2}} \right\}$$

$$+ \tan^{-1} \left\{ \frac{x_2 x_3^*}{(l_1 - x_1)[(l_1 - x_1)^2 + x_2^2 + x_3^{*2}]^{1/2}} \right\} \Bigg]$$

$$\begin{aligned}
D_2 = \frac{1}{8\pi l_1 l_2 l_3} & \left[\tan^{-1} \left\{ \frac{(l_1 - x_1)(l_3 - x_3^*)}{x_2[(l_1 - x_1)^2 + x_2^2 + (l_3 - x_3^*)^2]^{1/2}} \right\} \right. \\
& + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_3^*}{x_2[(l_1 - x_1)^2 + x_2^2 + x_3^{*2}]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{x_1(l_3 - x_3^*)}{x_2[x_1^2 + x_2^2 + (l_3 - x_3^*)^2]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{x_2 x_3}{x_2[x_1^2 + x_2^2 + x_3^{*2}]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{(l_1 - x_1)(l_3 - x_3^*)}{(l_2 - x_2)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + (l_3 - x_3^*)^2]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_3^*}{(l_2 - x_2)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + x_3^{*2}]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{x_1(l_3 - x_3^*)}{(l_2 - x_2)[x_1^2 + (l_2 - x_2)^2 + (l_3 - x_3^*)^2]^{1/2}} \right\} \\
& \left. + \tan^{-1} \left\{ \frac{x_2 x_3^*}{(l_2 - x_2)[x_1^2 + (l_2 - x_2)^2 + x_3^{*2}]^{1/2}} \right\} \right]
\end{aligned} \tag{7.734}$$

$$\begin{aligned}
D_3 = \frac{1}{8\pi l_1 l_2 l_3} & \left[\tan^{-1} \left\{ \frac{(l_1 - x_1)(l_2 - x_2)}{x_3 [(l_1 - x_1)^2 + (l_2 - x_2)^2 + x_3^{*2}]^{1/2}} \right\} \right. \\
& + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_2}{x_3 [(l_1 - x_1)^2 + x_2^2 + x_3^2]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{x_1(l_2 - x_2)}{x_3 [x_1^2 + (l_2 - x_2)^2 + x_3^{*2}]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{x_2 x_3}{x_3 [x_1^2 + x_2^2 + x_3^{*2}]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{(l_1 - x_1)(l_2 - x_2)}{(l_3 - x_3) [(l_1 - x_1)^2 + (l_2 - x_2)^2 + (l_3 - x_3^*)^2]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_2}{(l_3 - x_3) [(l_1 - x_1)^2 + x_2^2 + (l_3 - x_3^*)^2]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{x_1(l_2 - x_2)}{(l_3 - x_3) [x_1^2 + (l_2 - x_2)^2 + (l_3 - x_3^*)^2]^{1/2}} \right\} \\
& \left. + \tan^{-1} \left\{ \frac{x_2 x_3}{(l_3 - x_3) [x_1^2 + x_2^2 + (l_3 - x_3^*)^2]^{1/2}} \right\} \right]
\end{aligned} \tag{7.735}$$

Which, when reinserted becomes:

$$\begin{aligned}
G_2 = & -\frac{x_1}{4\pi l_2 l_3} \left[\tan^{-1} \left\{ \frac{(l_2 - x_2)(l_3 - x_3)}{x_1 [x_1^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{(l_2 - x_2)x_3}{x_1 [x_1^2 + (l_2 - x_2)^2 + x_3^2]^{\frac{1}{2}}} \right\} \right. \\
& + \tan^{-1} \left\{ \frac{x_2(l_3 - x_3)}{x_1 [x_1^2 + x_2^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{x_2 x_3}{x_1 [x_1^2 + x_2^2 + x_3^2]^{\frac{1}{2}}} \right\} \Big] \\
& - \frac{x_2}{4\pi l_1 l_3} \left[\tan^{-1} \left\{ \frac{(l_1 - x_1)(l_3 - x_3)}{x_2 [(l_1 - x_1)^2 + x_2^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_3}{x_2 [(l_1 - x_1)^2 + x_2^2 + x_3^2]^{\frac{1}{2}}} \right\} \right. \\
& + \tan^{-1} \left\{ \frac{x_1(l_3 - x_3)}{x_2 [x_1^2 + x_2^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{x_2 x_3}{x_2 [x_1^2 + x_2^2 + x_3^2]^{\frac{1}{2}}} \right\} \Big] \\
& - \frac{x_3}{4\pi l_1 l_2} \left[\tan^{-1} \left\{ \frac{(l_1 - x_1)(l_2 - x_2)}{x_3 [(l_1 - x_1)^2 + (l_2 - x_2)^2 + x_3^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_2}{x_3 [(l_1 - x_1)^2 + x_2^2 + x_3^2]^{\frac{1}{2}}} \right\} \right. \\
& + \tan^{-1} \left\{ \frac{x_1(l_2 - x_2)}{x_3 [x_1^2 + (l_2 - x_2)^2 + x_3^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{x_2 x_3}{x_3 [x_1^2 + x_2^2 + x_3^2]^{\frac{1}{2}}} \right\} \Big] \\
& + \frac{1}{8\pi l_1 l_2 l_3} \left[\tan^{-1} \left\{ \frac{(l_2 - x_2)(l_3 - x_3)}{x_1 [x_1^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{(l_2 - x_2)x_3}{x_1 [x_1^2 + (l_2 - x_2)^2 + x_3^2]^{\frac{1}{2}}} \right\} \right. \\
& + \tan^{-1} \left\{ \frac{x_2(l_3 - x_3)}{x_1 [x_1^2 + x_2^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{x_2 x_3}{x_1 [x_1^2 + x_2^2 + x_3^2]^{\frac{1}{2}}} \right\} \\
& + \tan^{-1} \left\{ \frac{(l_2 - x_2)(l_3 - x_3)}{(l_1 - x_1)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} \\
& + \tan^{-1} \left\{ \frac{(l_2 - x_2)x_3}{(l_1 - x_1)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + x_3^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{x_2(l_3 - x_3)}{(l_1 - x_1)[(l_1 - x_1)^2 + x_2^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} \\
& + \tan^{-1} \left\{ \frac{x_2 x_3}{(l_1 - x_1)[(l_1 - x_1)^2 + x_2^2 + x_3^2]^{\frac{1}{2}}} \right\} \Big] x_1^2 \\
& + \frac{x_2^2}{8\pi l_1 l_2 l_3} \left[\tan^{-1} \left\{ \frac{(l_1 - x_1)(l_3 - x_3)}{x_2 [(l_1 - x_1)^2 + x_2^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_3}{x_2 [(l_1 - x_1)^2 + x_2^2 + x_3^2]^{\frac{1}{2}}} \right\} \right. \\
& + \tan^{-1} \left\{ \frac{x_1(l_3 - x_3)}{x_2 [x_1^2 + x_2^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{x_2 x_3}{x_2 [x_1^2 + x_2^2 + x_3^2]^{\frac{1}{2}}} \right\} \\
& + \tan^{-1} \left\{ \frac{(l_1 - x_1)(l_3 - x_3)}{(l_2 - x_2)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} \\
& + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_3}{(l_2 - x_2)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + x_3^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{x_1(l_3 - x_3)}{(l_2 - x_2)[x_1^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} \\
& + \tan^{-1} \left\{ \frac{x_2 x_3}{(l_2 - x_2)[x_1^2 + (l_2 - x_2)^2 + x_3^2]^{\frac{1}{2}}} \right\} \Big] \\
& + \frac{x_3^2}{8\pi l_1 l_2 l_3} \left[\tan^{-1} \left\{ \frac{(l_1 - x_1)(l_2 - x_2)}{x_3 [(l_1 - x_1)^2 + (l_2 - x_2)^2 + x_3^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_2}{x_3 [(l_1 - x_1)^2 + x_2^2 + x_3^2]^{\frac{1}{2}}} \right\} \right. \\
& + \tan^{-1} \left\{ \frac{x_1(l_2 - x_2)}{x_3 [x_1^2 + (l_2 - x_2)^2 + x_3^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{x_2 x_3}{x_3 [x_1^2 + x_2^2 + x_3^2]^{\frac{1}{2}}} \right\} \\
& + \tan^{-1} \left\{ \frac{(l_1 - x_1)(l_2 - x_2)}{(l_3 - x_3)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\}
\end{aligned}$$

$$\begin{aligned}
& + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_2}{(l_3 - x_3)[(l_1 - x_1)^2 + x_2^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{x_1(l_2 - x_2)}{(l_3 - x_3)[x_1^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} \\
& + \tan^{-1} \left\{ \frac{x_2x_3}{(l_3 - x_3)[x_1^2 + x_2^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} \Bigg]
\end{aligned}$$

$$G_3 = \sum_0^{\infty} \sum_0^{\infty} \left[A_{m,n} \cosh(k_{m,n}(x_1 - l_1)) \right. \quad (7.736)$$

$$\begin{aligned}
& + B_{m,n} \cosh(k_{m,n}x_1) \Big] \cos\left(\frac{m\pi x_2}{l_2}\right) \cos\left(\frac{n\pi x_3}{l_3}\right) \\
& + \left[A'_{m,n} \cosh(k'_{m,n}(x_2 - l_2)) \right. \\
& + B'_{m,n} \cosh(k'_{m,n}x_2) \Big] \cos\left(\frac{m\pi x_3}{l_3}\right) \cos\left(\frac{n\pi x_1}{l_1}\right) \\
& + \left[A''_{m,n} \cosh(k''_{m,n}(x_3 - l_3)) \right. \\
& + B''_{m,n} \cosh(k''_{m,n}x_3) \Big] \cos\left(\frac{m\pi x_1}{l_1}\right) \cos\left(\frac{n\pi x_2}{l_2}\right)
\end{aligned}$$

$$k_{m,n} = \pi \left(\frac{m^2}{l_2^2} + \frac{n^2}{l_3^2} \right)^{1/2} \quad k_{m,n} = \pi \left(\frac{m^2}{l_3^2} + \frac{n^2}{l_1^2} \right)^{1/2} \quad k_{m,n} = \pi \left(\frac{m^2}{l_1^2} + \frac{n^2}{l_2^2} \right)^{1/2} \quad (7.737)$$

$$\begin{aligned}
A_{m,n} = \frac{4}{l_2 l_3 k_{m,n} \sinh(k_{m,n} l_1)} \int_0^{l_2} \int_0^{l_3} \frac{\partial}{\partial x_1} (G_1 \\
+ G_2) \Big|_{x_1=0} \cos\left(\frac{m\pi x_2}{l_2}\right) \cos\left(\frac{n\pi x_3}{l_3}\right) dx_2 dx_3 \quad (7.738)
\end{aligned}$$

$$\begin{aligned}
A'_{m,n} = \frac{4}{l_1 l_3 k'_{m,n} \sinh(k'_{m,n} l_2)} \int_0^{l_1} \int_0^{l_3} \frac{\partial}{\partial x_2} (G_1 \\
+ G_2) \Big|_{x_2=0} \cos\left(\frac{m\pi x_3}{l_3}\right) \cos\left(\frac{n\pi x_1}{l_1}\right) dx_1 dx_3 \quad (7.739)
\end{aligned}$$

$$A''_{m,n} = \frac{4}{l_2 l_1 k''_{m,n} \sinh(k''_{m,n} l_3)} \int_0^{l_1} \int_0^{l_2} \frac{\partial}{\partial x_3} (G_1 + G_2) \Big|_{x_3=0} \cos\left(\frac{m\pi x_1}{l_1}\right) \cos\left(\frac{n\pi x_2}{l_2}\right) dx_1 dx_2 \quad (7.740)$$

$$B_{m,n} = \frac{4}{l_2 l_3 k_{m,n} \sinh(k_{m,n} l_1)} \int_0^{l_2} \int_0^{l_3} \frac{\partial}{\partial x_1} (G_1 + G_2) \Big|_{x_1=l_1} \cos\left(\frac{m\pi x_2}{l_2}\right) \cos\left(\frac{n\pi x_3}{l_3}\right) dx_2 dx_3 \quad (7.741)$$

$$B'_{m,n} = \frac{4}{l_1 l_3 k'_{m,n} \sinh(k'_{m,n} l_2)} \int_0^{l_1} \int_0^{l_3} \frac{\partial}{\partial x_2} (G_1 + G_2) \Big|_{x_2=l_2} \cos\left(\frac{m\pi x_3}{l_3}\right) \cos\left(\frac{n\pi x_1}{l_1}\right) dx_1 dx_3 \quad (7.742)$$

$$B''_{m,n} = \frac{4}{l_2 l_1 k''_{m,n} \sinh(k''_{m,n} l_3)} \int_0^{l_1} \int_0^{l_2} \frac{\partial}{\partial x_3} (G_1 + G_2) \Big|_{x_3=l_3} \cos\left(\frac{m\pi x_1}{l_1}\right) \cos\left(\frac{n\pi x_2}{l_2}\right) dx_1 dx_2 \quad (7.743)$$

When combining: (7.744)

$$\begin{aligned}
 G_3 = & \sum_{n=0}^{\infty} \sum_{m=0}^{\infty} \left[\frac{4}{l_2 l_3 k_{m,n} \sinh(k_{m,n} l_1)} \int_0^{l_2} \int_0^{l_3} \frac{\partial}{\partial x} (G_1 + G_2) \Big|_{x_1=0} \cos\left(\frac{m\pi x_2}{l_2}\right) \cos\left(\frac{n\pi x_3}{l_3}\right) dx_2 dx_3 \cosh\left(\pi \left(\frac{m^2}{l_2^2} + \frac{n^2}{l_3^2}\right)^{1/2} (x_1 - l_1)\right) \right. \\
 & + \frac{4}{l_2 l_3 k_{m,n} \sinh(k_{m,n} l_1)} \int_0^{l_2} \int_0^{l_3} \frac{\partial}{\partial x} (G_1 + G_2) \Big|_{x_1=l_1} \cos\left(\frac{m\pi x_2}{l_2}\right) \cos\left(\frac{n\pi x_3}{l_3}\right) dx_2 dx_3 \cosh\left(\pi \left(\frac{m^2}{l_2^2} + \frac{n^2}{l_3^2}\right)^{1/2} x_1\right) \cos\left(\frac{m\pi x_2}{l_2}\right) \cos\left(\frac{n\pi x_3}{l_3}\right) \\
 & + \left[\frac{4}{l_1 l_3 k_{m,n} \sinh(k_{m,n} l_2)} \int_0^{l_1} \int_0^{l_3} \frac{\partial}{\partial x} (G_1 + G_2) \Big|_{x_2=0} \cos\left(\frac{m\pi x_3}{l_3}\right) \cos\left(\frac{n\pi x_1}{l_1}\right) dx_1 dx_3 \cosh\left(\pi \left(\frac{m^2}{l_2^2} + \frac{n^2}{l_1^2}\right)^{1/2} (x_2 - l_2)\right) \right. \\
 & + \frac{4}{l_1 l_3 k_{m,n} \sinh(k_{m,n} l_2)} \int_0^{l_1} \int_0^{l_3} \frac{\partial}{\partial x} (G_1 + G_2) \Big|_{x_2=l_2} \cos\left(\frac{m\pi x_3}{l_3}\right) \cos\left(\frac{n\pi x_1}{l_1}\right) dx_1 dx_3 \cosh\left(\pi \left(\frac{m^2}{l_2^2} + \frac{n^2}{l_1^2}\right)^{1/2} x_2\right) \cos\left(\frac{m\pi x_3}{l_3}\right) \cos\left(\frac{n\pi x_1}{l_1}\right) \\
 & + \left[\frac{4}{l_2 l_1 k_{m,n} \sinh(k_{m,n} l_3)} \int_0^{l_1} \int_0^{l_2} \frac{\partial}{\partial x} (G_1 + G_2) \Big|_{x_3=0} \cos\left(\frac{m\pi x_1}{l_1}\right) \cos\left(\frac{n\pi x_2}{l_2}\right) dx_1 dx_2 \cosh\left(\pi \left(\frac{m^2}{l_1^2} + \frac{n^2}{l_2^2}\right)^{1/2} (x_3 - l_3)\right) \right. \\
 & \left. \left. + \frac{4}{l_2 l_1 k_{m,n} \sinh(k_{m,n} l_3)} \int_0^{l_1} \int_0^{l_2} \frac{\partial}{\partial x} (G_1 + G_2) \Big|_{x_3=l_3} \cos\left(\frac{m\pi x_1}{l_1}\right) \cos\left(\frac{n\pi x_2}{l_2}\right) dx_1 dx_2 \cosh\left(\pi \left(\frac{m^2}{l_1^2} + \frac{n^2}{l_2^2}\right)^{1/2} x_3\right) \cos\left(\frac{m\pi x_1}{l_1}\right) \cos\left(\frac{n\pi x_2}{l_2}\right) \right] \right]
 \end{aligned}$$

When combining altogether: (7.745)

$$\begin{aligned}
 G_3 = & \sum_{n=0}^{\infty} \sum_{m=0}^{\infty} \left[\frac{4}{l_2 l_3 k_{m,n} \sinh(k_{m,n} l_1)} \int_0^{l_2} \int_0^{l_3} \frac{\partial}{\partial x} \left(\frac{1}{4 + \pi|x - x'|} + G_2 \right) \Big|_{x_1=0} \cos\left(\frac{m\pi x_2}{l_2}\right) \cos\left(\frac{n\pi x_3}{l_3}\right) dx_2 dx_3 \cosh\left(\pi \left(\frac{m^2}{l_2^2} + \frac{n^2}{l_3^2}\right)^{1/2} (x_1 - l_1)\right) \right. \\
 & + \frac{4}{l_2 l_3 k_{m,n} \sinh(k_{m,n} l_1)} \int_0^{l_2} \int_0^{l_3} \frac{\partial}{\partial x} \left(\frac{1}{4 + \pi|x - x'|} + G_2 \right) \Big|_{x_1=l_1} \cos\left(\frac{m\pi x_2}{l_2}\right) \cos\left(\frac{n\pi x_3}{l_3}\right) dx_2 dx_3 \cosh\left(\pi \left(\frac{m^2}{l_2^2} + \frac{n^2}{l_3^2}\right)^{1/2} x_1\right) \\
 & \left. + \frac{n^2}{l_3^2} \right)^{1/2} x_1 \Big] \cos\left(\frac{m\pi x_2}{l_2}\right) \cos\left(\frac{n\pi x_3}{l_3}\right) \\
 & + \left[\frac{4}{l_1 l_3 k_{m,n} \sinh(k_{m,n} l_2)} \int_0^{l_1} \int_0^{l_3} \frac{\partial}{\partial x} \left(\frac{1}{4 + \pi|x - x'|} + G_2 \right) \Big|_{x_2=0} \cos\left(\frac{m\pi x_3}{l_3}\right) \cos\left(\frac{n\pi x_1}{l_1}\right) dx_1 dx_3 \cosh\left(\pi \left(\frac{m^2}{l_2^2} + \frac{n^2}{l_1^2}\right)^{1/2} (x_2 - l_2)\right) \right. \\
 & + \frac{4}{l_1 l_3 k_{m,n} \sinh(k_{m,n} l_2)} \int_0^{l_1} \int_0^{l_3} \frac{\partial}{\partial x} \left(\frac{1}{4 + \pi|x - x'|} + G_2 \right) \Big|_{x_2=l_2} \cos\left(\frac{m\pi x_3}{l_3}\right) \cos\left(\frac{n\pi x_1}{l_1}\right) dx_1 dx_3 \cosh\left(\pi \left(\frac{m^2}{l_2^2} + \frac{n^2}{l_1^2}\right)^{1/2} x_2\right) \\
 & \left. + \frac{n^2}{l_1^2} \right)^{1/2} x_2 \Big] \cos\left(\frac{m\pi x_3}{l_3}\right) \cos\left(\frac{n\pi x_1}{l_1}\right) \\
 & + \left[\frac{4}{l_2 l_1 k_{m,n} \sinh(k_{m,n} l_3)} \int_0^{l_1} \int_0^{l_2} \frac{\partial}{\partial x} \left(\frac{1}{4 + \pi|x - x'|} + G_2 \right) \Big|_{x_3=0} \cos\left(\frac{m\pi x_1}{l_1}\right) \cos\left(\frac{n\pi x_2}{l_2}\right) dx_1 dx_2 \cosh\left(\pi \left(\frac{m^2}{l_1^2} + \frac{n^2}{l_2^2}\right)^{1/2} (x_3 - l_3)\right) \right. \\
 & + \frac{4}{l_2 l_1 k_{m,n} \sinh(k_{m,n} l_3)} \int_0^{l_1} \int_0^{l_2} \frac{\partial}{\partial x} \left(\frac{1}{4 + \pi|x - x'|} + G_2 \right) \Big|_{x_3=l_3} \cos\left(\frac{m\pi x_1}{l_1}\right) \cos\left(\frac{n\pi x_2}{l_2}\right) dx_1 dx_2 \cosh\left(\pi \left(\frac{m^2}{l_1^2} + \frac{n^2}{l_2^2}\right)^{1/2} x_3\right) \\
 & \left. + \frac{n^2}{l_2^2} \right)^{1/2} x_3 \Big] \cos\left(\frac{m\pi x_1}{l_1}\right) \cos\left(\frac{n\pi x_2}{l_2}\right)
 \end{aligned}$$

[illegible]

[illegible]

[illegible]

$$\begin{aligned}
& + \tan^{-1} \left\{ \frac{x_2 x_3}{x_3 [x_1^2 + x_2^2 + x_3^2]^{1/2}} \right\} \\
& + \frac{1}{8\pi l_1 l_2 l_3} \left[\tan^{-1} \left\{ \frac{(l_2 - x_2)(l_3 - x_3)}{x_1 [x_1^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{1/2}} \right\} + \tan^{-1} \left\{ \frac{(l_2 - x_2)x_3}{x_1 [x_1^2 + (l_2 - x_2)^2 + x_3^2]^{1/2}} \right\} + \tan^{-1} \left\{ \frac{x_2(l_3 - x_3)}{x_1 [x_1^2 + x_2^2 + (l_3 - x_3)^2]^{1/2}} \right\} \right. \\
& + \tan^{-1} \left\{ \frac{x_2 x_3}{x_1 [x_1^2 + x_2^2 + x_3^2]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{(l_2 - x_2)(l_3 - x_3)}{(l_1 - x_1)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{1/2}} \right\} + \tan^{-1} \left\{ \frac{(l_2 - x_2)x_3}{(l_1 - x_1)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + x_3^2]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{x_2(l_3 - x_3)}{(l_1 - x_1)[(l_1 - x_1)^2 + x_2^2 + (l_3 - x_3)^2]^{1/2}} \right\} + \tan^{-1} \left\{ \frac{x_2 x_3}{(l_1 - x_1)[(l_1 - x_1)^2 + x_2^2 + x_3^2]^{1/2}} \right\} \Big] x_1^2 \\
& + \frac{x_2^2}{8\pi l_1 l_2 l_3} \left[\tan^{-1} \left\{ \frac{(l_1 - x_1)(l_3 - x_3)}{x_2 [(l_1 - x_1)^2 + x_2^2 + (l_3 - x_3)^2]^{1/2}} \right\} + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_3}{x_2 [(l_1 - x_1)^2 + x_2^2 + x_3^2]^{1/2}} \right\} + \tan^{-1} \left\{ \frac{x_1(l_3 - x_3)}{x_2 [x_1^2 + x_2^2 + (l_3 - x_3)^2]^{1/2}} \right\} \right. \\
& + \tan^{-1} \left\{ \frac{x_2 x_3}{x_2 [x_1^2 + x_2^2 + x_3^2]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{(l_1 - x_1)(l_3 - x_3)}{(l_2 - x_2)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{1/2}} \right\} + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_3}{(l_2 - x_2)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + x_3^2]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{x_1(l_3 - x_3)}{(l_2 - x_2)[x_1^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{1/2}} \right\} + \tan^{-1} \left\{ \frac{x_2 x_3}{(l_2 - x_2)[x_1^2 + (l_2 - x_2)^2 + x_3^2]^{1/2}} \right\} \Big] \\
& + \frac{x_3^2}{8\pi l_1 l_2 l_3} \left[\tan^{-1} \left\{ \frac{(l_1 - x_1)(l_2 - x_2)}{x_3 [(l_1 - x_1)^2 + (l_2 - x_2)^2 + x_3^2]^{1/2}} \right\} + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_2}{x_3 [(l_1 - x_1)^2 + x_2^2 + x_3^2]^{1/2}} \right\} + \tan^{-1} \left\{ \frac{x_1(l_2 - x_2)}{x_3 [x_1^2 + (l_2 - x_2)^2 + x_3^2]^{1/2}} \right\} \right. \\
& + \tan^{-1} \left\{ \frac{x_2 x_3}{x_3 [x_1^2 + x_2^2 + x_3^2]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{(l_1 - x_1)(l_2 - x_2)}{(l_3 - x_3)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{1/2}} \right\} + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_2}{(l_3 - x_3)[(l_1 - x_1)^2 + x_2^2 + (l_3 - x_3)^2]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{x_1(l_2 - x_2)}{(l_3 - x_3)[x_1^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{x_2 x_3}{(l_3 - x_3)[x_1^2 + x_2^2 + (l_3 - x_3)^2]^{1/2}} \right\} \Big] \Bigg) \Bigg|_{x_3=l_3} \cos\left(\frac{m\pi x_1}{l_1}\right) \cos\left(\frac{n\pi x_2}{l_2}\right) dx_1 dx_2 \cosh\left(\pi \left(\frac{m^2}{l_1^2} + \frac{n^2}{l_2^2}\right)^{1/2} x_3\right) \cos\left(\frac{m\pi x_1}{l_1}\right) \cos\left(\frac{n\pi x_2}{l_2}\right)
\end{aligned}$$

When combining all parts, the full Green function becomes:

$$\begin{aligned}
 G = & \frac{1}{4 \pi [x - x^*]} - \frac{x_1}{4 \pi l_2 l_3} \left[\tan^{-1} \left\{ \frac{(l_2 - x_2)(l_3 - x_3)}{x_1 [x_1^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{(l_2 - x_2)x_3}{x_1 [x_1^2 + (l_2 - x_2)^2 + x_3^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{x_2(l_3 - x_3)}{x_1 [x_1^2 + x_2^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} \right. \\
 & \left. + \tan^{-1} \left\{ \frac{x_2 x_3}{x_1 [x_1^2 + x_2^2 + x_3^2]^{\frac{1}{2}}} \right\} \right] \\
 & - \frac{x_2}{4 \pi l_1 l_3} \left[\tan^{-1} \left\{ \frac{(l_1 - x_1)(l_3 - x_3)}{x_2 [(l_1 - x_1)^2 + x_2^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_3}{x_2 [(l_1 - x_1)^2 + x_2^2 + x_3^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{x_1(l_3 - x_3)}{x_2 [x_1^2 + x_2^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} \right. \\
 & \left. + \tan^{-1} \left\{ \frac{x_2 x_3}{x_2 [x_1^2 + x_2^2 + x_3^2]^{\frac{1}{2}}} \right\} \right] \\
 & - \frac{x_3}{4 \pi l_1 l_2} \left[\tan^{-1} \left\{ \frac{(l_1 - x_1)(l_2 - x_2)}{x_3 [(l_1 - x_1)^2 + (l_2 - x_2)^2 + x_3^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_2}{x_3 [(l_1 - x_1)^2 + x_2^2 + x_3^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{x_1(l_2 - x_2)}{x_3 [x_1^2 + (l_2 - x_2)^2 + x_3^2]^{\frac{1}{2}}} \right\} \right. \\
 & \left. + \tan^{-1} \left\{ \frac{x_2 x_3}{x_3 [x_1^2 + x_2^2 + x_3^2]^{\frac{1}{2}}} \right\} \right] \\
 & + \frac{1}{8 \pi l_1 l_2 l_3} \left[\tan^{-1} \left\{ \frac{(l_2 - x_2)(l_3 - x_3)}{x_1 [x_1^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{(l_2 - x_2)x_3}{x_1 [x_1^2 + (l_2 - x_2)^2 + x_3^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{x_2(l_3 - x_3)}{x_1 [x_1^2 + x_2^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} \right. \\
 & \left. + \tan^{-1} \left\{ \frac{x_2 x_3}{x_1 [x_1^2 + x_2^2 + x_3^2]^{\frac{1}{2}}} \right\} \right. \\
 & \left. + \tan^{-1} \left\{ \frac{(l_2 - x_2)(l_3 - x_3)}{(l_1 - x_1)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{(l_2 - x_2)x_3}{(l_1 - x_1)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + x_3^2]^{\frac{1}{2}}} \right\} \right. \\
 & \left. + \tan^{-1} \left\{ \frac{x_2(l_3 - x_3)}{(l_1 - x_1)[(l_1 - x_1)^2 + x_2^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{x_2 x_3}{(l_1 - x_1)[(l_1 - x_1)^2 + x_2^2 + x_3^2]^{\frac{1}{2}}} \right\} \right] x_1^2 \\
 & + \frac{x_2^2}{8 \pi l_1 l_2 l_3} \left[\tan^{-1} \left\{ \frac{(l_1 - x_1)(l_3 - x_3)}{x_2 [(l_1 - x_1)^2 + x_2^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_3}{x_2 [(l_1 - x_1)^2 + x_2^2 + x_3^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{x_1(l_3 - x_3)}{x_2 [x_1^2 + x_2^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} \right. \\
 & \left. + \tan^{-1} \left\{ \frac{x_2 x_3}{x_2 [x_1^2 + x_2^2 + x_3^2]^{\frac{1}{2}}} \right\} \right. \\
 & \left. + \tan^{-1} \left\{ \frac{(l_1 - x_1)(l_3 - x_3)}{(l_2 - x_2)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_3}{(l_2 - x_2)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + x_3^2]^{\frac{1}{2}}} \right\} \right. \\
 & \left. + \tan^{-1} \left\{ \frac{x_1(l_3 - x_3)}{(l_2 - x_2)[x_1^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{x_2 x_3}{(l_2 - x_2)[x_1^2 + (l_2 - x_2)^2 + x_3^2]^{\frac{1}{2}}} \right\} \right] \\
 & + \frac{x_3^2}{8 \pi l_1 l_2 l_3} \left[\tan^{-1} \left\{ \frac{(l_1 - x_1)(l_2 - x_2)}{x_3 [(l_1 - x_1)^2 + (l_2 - x_2)^2 + x_3^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_2}{x_3 [(l_1 - x_1)^2 + x_2^2 + x_3^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{x_1(l_2 - x_2)}{x_3 [x_1^2 + (l_2 - x_2)^2 + x_3^2]^{\frac{1}{2}}} \right\} \right. \\
 & \left. + \tan^{-1} \left\{ \frac{x_2 x_3}{x_3 [x_1^2 + x_2^2 + x_3^2]^{\frac{1}{2}}} \right\} \right. \\
 & \left. + \tan^{-1} \left\{ \frac{(l_1 - x_1)(l_2 - x_2)}{(l_3 - x_3)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_2}{(l_3 - x_3)[(l_1 - x_1)^2 + x_2^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} \right. \\
 & \left. + \tan^{-1} \left\{ \frac{x_1(l_2 - x_2)}{(l_3 - x_3)[x_1^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} + \tan^{-1} \left\{ \frac{x_2 x_3}{(l_3 - x_3)[x_1^2 + x_2^2 + (l_3 - x_3)^2]^{\frac{1}{2}}} \right\} \right] +
 \end{aligned}$$

[illegible]

[illegible]

[illegible]

$$\begin{aligned}
& + \tan^{-1} \left\{ \frac{x_2 x_3}{x_3 [x_1^2 + x_2^2 + x_3^2]^{1/2}} \right\} \\
& + \frac{1}{8\pi l_1 l_2 l_3} \left[\tan^{-1} \left\{ \frac{(l_2 - x_2)(l_3 - x_3)}{x_1 [x_1^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{1/2}} \right\} + \tan^{-1} \left\{ \frac{(l_2 - x_2)x_3}{x_1 [x_1^2 + (l_2 - x_2)^2 + x_3^2]^{1/2}} \right\} + \tan^{-1} \left\{ \frac{x_2(l_3 - x_3)}{x_1 [x_1^2 + x_2^2 + (l_3 - x_3)^2]^{1/2}} \right\} \right. \\
& + \tan^{-1} \left\{ \frac{x_2 x_3}{x_1 [x_1^2 + x_2^2 + x_3^2]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{(l_2 - x_2)(l_3 - x_3)}{(l_1 - x_1)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{1/2}} \right\} + \tan^{-1} \left\{ \frac{(l_2 - x_2)x_3}{(l_1 - x_1)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + x_3^2]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{x_2(l_3 - x_3)}{(l_1 - x_1)[(l_1 - x_1)^2 + x_2^2 + (l_3 - x_3)^2]^{1/2}} \right\} + \tan^{-1} \left\{ \frac{x_2 x_3}{(l_1 - x_1)[(l_1 - x_1)^2 + x_2^2 + x_3^2]^{1/2}} \right\} \Big] x_1^2 \\
& + \frac{x_2^2}{8\pi l_1 l_2 l_3} \left[\tan^{-1} \left\{ \frac{(l_1 - x_1)(l_3 - x_3)}{x_2 [(l_1 - x_1)^2 + x_2^2 + (l_3 - x_3)^2]^{1/2}} \right\} + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_3}{x_2 [(l_1 - x_1)^2 + x_2^2 + x_3^2]^{1/2}} \right\} + \tan^{-1} \left\{ \frac{x_1(l_3 - x_3)}{x_2 [x_1^2 + x_2^2 + (l_3 - x_3)^2]^{1/2}} \right\} \right. \\
& + \tan^{-1} \left\{ \frac{x_2 x_3}{x_2 [x_1^2 + x_2^2 + x_3^2]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{(l_1 - x_1)(l_3 - x_3)}{(l_2 - x_2)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{1/2}} \right\} + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_3}{(l_2 - x_2)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + x_3^2]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{x_1(l_3 - x_3)}{(l_2 - x_2)[x_1^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{1/2}} \right\} + \tan^{-1} \left\{ \frac{x_2 x_3}{(l_2 - x_2)[x_1^2 + (l_2 - x_2)^2 + x_3^2]^{1/2}} \right\} \Big] \\
& + \frac{x_3^2}{8\pi l_1 l_2 l_3} \left[\tan^{-1} \left\{ \frac{(l_1 - x_1)(l_2 - x_2)}{x_3 [(l_1 - x_1)^2 + (l_2 - x_2)^2 + x_3^2]^{1/2}} \right\} + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_2}{x_3 [(l_1 - x_1)^2 + x_2^2 + x_3^2]^{1/2}} \right\} + \tan^{-1} \left\{ \frac{x_1(l_2 - x_2)}{x_3 [x_1^2 + (l_2 - x_2)^2 + x_3^2]^{1/2}} \right\} \right. \\
& + \tan^{-1} \left\{ \frac{x_2 x_3}{x_3 [x_1^2 + x_2^2 + x_3^2]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{(l_1 - x_1)(l_2 - x_2)}{(l_3 - x_3)[(l_1 - x_1)^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{1/2}} \right\} + \tan^{-1} \left\{ \frac{(l_1 - x_1)x_2}{(l_3 - x_3)[(l_1 - x_1)^2 + x_2^2 + (l_3 - x_3)^2]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{x_1(l_2 - x_2)}{(l_3 - x_3)[x_1^2 + (l_2 - x_2)^2 + (l_3 - x_3)^2]^{1/2}} \right\} \\
& + \tan^{-1} \left\{ \frac{x_2 x_3}{(l_3 - x_3)[x_1^2 + x_2^2 + (l_3 - x_3)^2]^{1/2}} \right\} \Big] \Bigg|_{x_3=l_3} \cos \left(\frac{m\pi x_1}{l_1} \right) \cos \left(\frac{n\pi x_2}{l_2} \right) dx_1 dx_2 \cosh \left(\pi \left(\frac{m^2}{l_1^2} + \frac{n^2}{l_2^2} \right)^{1/2} x_3 \right) \cos \left(\frac{m\pi x_1}{l_1} \right) \cos \left(\frac{n\pi x_2}{l_2} \right)
\end{aligned}$$

7.38 Appendix AL: Models for synthesizing vascular networks

Many models for computer synthesis of microcirculatory models have been proposed. Some of these models are reviewed here to compare to the model developed in this thesis.

One group began a pioneered a vascular synthesis method that was derived from principles of angiogenesis [53]. It employed a model of hypoxia-induced VEGF production that spawned a stochastic decision variable as to whether to increase tortuosity or sprout a bifurcation. In each iteration, the growth territory size increases to allow for new vessels. This step also consists of a perfusion mapping simulation, where the synthesis of new vasculature ensures a specified level of blood perfusion carries oxygen to all tissue regions efficiently (all levels are larger than hypoxic levels).

Two methods of spatially-defined growth were used, each describes the oxygen consumption within the domain; a uniform distribution and a Gaussian distribution.

One of the main limitations of this work is that the growth seems anatomical from the description, however almost all of the growth parameters are based on PDFs that have not derived from empirical or physiological observation. Moreover, this method is not capable of growing structures at the scale of the mouse hemisphere, as every round of growth requires an oxygen simulation, and spatial extent of oxygen simulations are still limited to small regions ($\sim 1\text{mm}^3$).

Another group that worked on vascular synthesis for a region of brain matter focused on the capillary bed [37,54,283,286] without growing the pial or penetrating vessels. This group used statistics from the previously mentioned Cassot [42] reconstruction in order to derive new capillary beds with statistically similar properties.

The PDF calculated in Cassot was binned into 10 micron intervals and the goal was to evaluate four different algorithms for network generation for statistical comparison. All methods begin with random number generation inside a cube and three methods were proposed in how to connect the points.

The shortest arc method (SAM) connects each node to a close node with a maximum of 3 connections per node (bifurcation). The bifurcation rule is enforced by pruning excess connections in a post-processing step. This method was claimed to underestimate the lengths over 100 microns (attributed to lack of tortuosity).

The Gamma distribution method (GAM) samples the length of each arc from a gamma distribution (chosen because it seems to resemble experimental distributions). This method was better than the SAM for estimating the length distribution but could not capture the short lengths very well, and thus did not match the topology correctly. The model for the distribution was given and the parameters were hand-chosen:

$$f(x) = a^b x^{b-1} \frac{e^{-ax}}{\Gamma(b)} \quad (7.746)$$

The spanning tree method (STM) was taken from graph theory with the goal of growing two trees that will eventually be connected. The algorithm places all the nodes first, then begins with the shortest segment and attaches segments based on proximity to the nearest node. It was noted that this method is the first of the methods to allow tortuous segments (continuations). This did not, however, give a perfect fit to the length distribution. To adapt it further, a method of randomly

pruning the network was implemented that removed segments in the undesired length range. This method improved the distribution.

The modified STM (MSTM) method was effectively the same as the STM method except after the tree is grown, artificial nodes are added very close to bifurcation points to skew the PDF towards the short-segment region (more short segments). This managed a good fit for the data from Cassot. Linear flow without hematocrit was calculated for all 4 network types and it was concluded that the hemodynamic properties were somewhat similar for all networks. The total flow was highest in STM yet the ratio of flow to surface area (considered the network “performance”) was highest in the MSTM method.

Some groups have attempted homogenization techniques to simplify the large size of the capillary network using a Cartesian mesh and an anisotropic diffusion tensor. This simplification uses anisotropy to represent the non-uniform vascular flow by altering local resistances (using a diffusivity tensor instead of a constant value). One such group used architecture based on graph theory [36]. A lot of emphasis was placed on deriving the radially symmetric (spherical to be exact) decay of the source (penetrating arterial terminal) on the neighboring mesh cells. The inverse relationship between distance and magnitude of the source strength on the surrounding tissue was fit with a linear portion which defined the unique decay profile for each terminal. The nearest 3 cells in all directions were considered affected, but at the end of the day the decay was considered averaged within each cell, evaluated at the cell center.

The vasculature obeys Hagen Poiseuille flow subject to the Pries in-vitro modified viscosity formula (see Section 7.12.1 for more information on biphasic blood flow or hematocrit-dependent viscosity). The capillary continuum is modeled with Darcy’s law, however the permeability coefficient is a function of effective viscosity and edge length:

$$K_{eff} = \frac{\bar{K}}{\mu_{eff}} h \quad (7.747)$$

The coupling has a potential:

$$P(r) = p_{ref} - \frac{\mu_{ref} q_s}{4\pi \bar{K}} \left(\frac{1}{r} - \frac{1}{l_{ref}} \right) \quad (7.748)$$

Where p_{ref} is pressure at arbitrary distance l_{ref} from source. Note, $r \leq l_{ref}$. l_{ref} refers to the reference length and p_{ref} to the reference pressure. Metrics were developed to compare the artificial vascular network to the continuum capillary mesh. The validation showed that the partial source redistribution method can substantially improve the distribution patterns and they claim that the point source drop-off calculation is necessary to reduce error in the entire capillary continuum. The computational efficiency also is assessed and concluded that the continuum method is a significant reduction of the capillary network when using the more complex mathematical implementation.

One limitation in this method is that the derivation claims an imperative assumption that l_{ref} is much smaller than the edge length of the mesh ($l_{ref} \ll h$), something that will cause problems with mesh refinement. Moreover, all the statistics from this method were validated against a geometry that is highly planar (thickness \ll length x width), causing significant skewing of the statistics, topological properties, and flow patterns of this network.

A second group to use a homogenization technique used coupling between pial and penetrating vessels through an inhomogeneous tissue block endowed with an anisotropic diffusion tensor [37,286]. Once the diffusion tensor was calculated (using the eigenvalue decomposition of the

overall flow vector within the element), the tissue was endowed with Darcy's law coupled to the Hagen- Poiseuille flow in the vessels through a finite source term dispensing into the local mesh volume with a resistive boundary coupling.

$$\begin{aligned}\Delta p &= \alpha f \\ \vec{\nabla} \cdot f + S &= 0\end{aligned}\tag{7.749}$$

Where viscosity is Pries in vivo (see biphasic blood flow report).

$$\nabla \cdot (K \nabla p) - S = 0\tag{7.750}$$

Where S is the flow between meshes, also known as a source.

$$S_i = \sum_{i=1}^{nTerm} G_i \Delta p\tag{7.751}$$

Where G_i is the conductance between the given terminal and the surrounding mesh element. The lateral flow observed in the group's results is caused by the proclivity of flow to the path of least resistance in a continuum, which in this case is the shortest path between the end of the penetrating arterioles and ascending venules. Physiologically consistent capillaries would extend numerous discrete paths - an important note that cannot be accurately captured by this method.

Each case study was run for a single penetrating arteriole occlusion and an ascending venule occlusion. The results found that the pressure and velocity were reduced near the occluded vessel but were relatively unaffected elsewhere.

While this model is seemingly approachable, it does not capture the description of vascular flow through the tissue appropriately, nor does the model seem to be amenable to a biphasic blood flow splitting model. Moreover, the mesh cell density is significantly larger than microvascular network models, endowing these simulations with significantly *larger* than the original counterparts. This is seemingly in opposition to the goal of the models.

7.39 Appendix AM: Data inventory

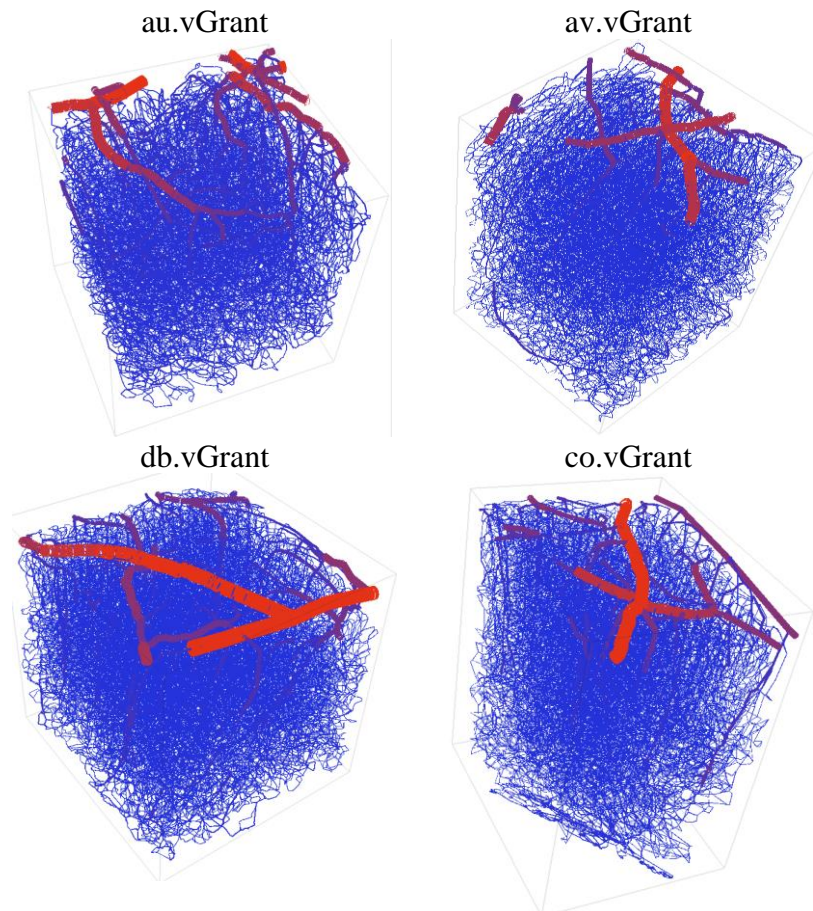
All data is saved in `nwk.Grant` in `grantsThesisNode`.

7.39.1 Empirical Kleinfeld networks

Networks with boundary conditions directory: `grantsInventory\EKF_SeriesNetworks\`

Statistics for each network: `grantsInventory\EKF_SeriesNetworks\statistics\`

Biphasic blood flow results: `grantsInventory\EKF_SeriesNetworks\simulationResults\`



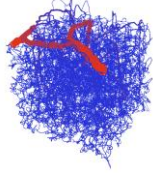
7.39.2 Generation 1, biphasic blood flow paper data

Networks with boundary conditions directory: *grantsInventory\SKF_SeriesI*

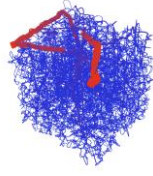
Statistics for each network: *grantsInventory\SKF_SeriesI*

Biphasic blood flow results: *grantsInventory\SKF_SeriesI\simulationResults*

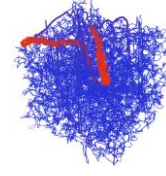
Au1.tortuous (E1.1)
au.1.tortuous.resultsLinninger2015
Pries_In_Vitro



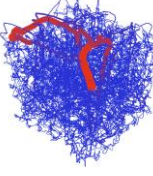
Au2.tortuous (E1.2)
au.2.tortuous.resultsLinninger2015
Pries_In_Vitro



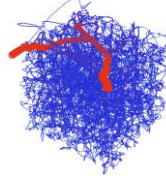
Au3.tortuous (E1.3)
au.3.tortuous.resultsLinninger2015
Pries_In_Vitro



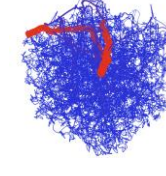
Au4.tortuous (E1.4)
au.4.tortuous.resultsLinninger2015
Pries_In_Vitro



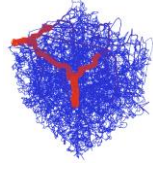
Au5.tortuous (E1.5)
au.5.tortuous.resultsLinninger2015
Pries_In_Vitro



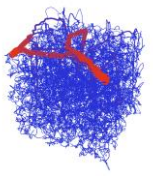
Au6.tortuous (E1.6)
au.6.tortuous.resultsLinninger2015
Pries_In_Vitro



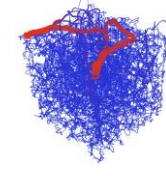
Au7.tortuous (E1.7)
au.7.tortuous.resultsLinninger2015
Pries_In_Vitro



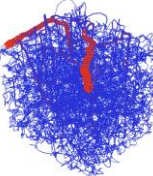
Au8.tortuous (E1.8)
au.8.tortuous.resultsLinninger2015
Pries_In_Vitro



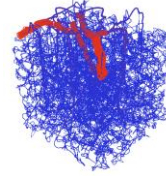
Au9.tortuous (E1.9)
au.9.tortuous.resultsLinninger2015
Pries_In_Vitro



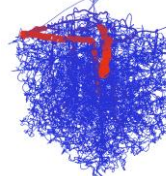
Au10.tortuous (E1.10)
au.10.tortuous.resultsLinninger2015
5Pries_In_Vitro



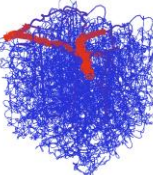
Au11.tortuous (E1.11)
au.11.tortuous.resultsLinninger2015
5Pries_In_Vitro



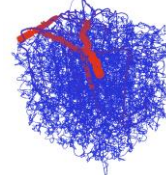
Au12.tortuous (E1.12)
au.12.tortuous.resultsLinninger2015
5Pries_In_Vitro



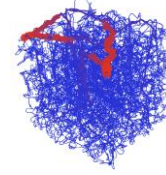
Au13.tortuous (E1.13)
au.13.tortuous.resultsLinninger2015
5Pries_In_Vitro



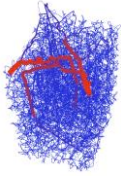
Au14.tortuous (E1.14)
au.14.tortuous.resultsLinninger2015
5Pries_In_Vitro



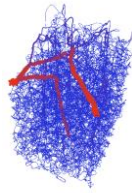
Au15.tortuous (E1.15)
au.15.tortuous.resultsLinninger2015
5Pries_In_Vitro



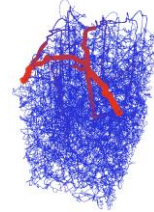
Av1.tortuous (E2.1)
av.1.tortuous.resultsLinninger2015
Pries_In_Vitro



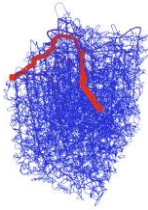
Av2.tortuous (E2.2)
av.2.tortuous.resultsLinninger2015
Pries_In_Vitro



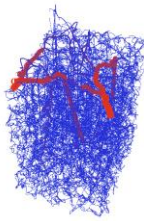
Av3.tortuous (E2.3)
av.3.tortuous.resultsLinninger2015
Pries_In_Vitro



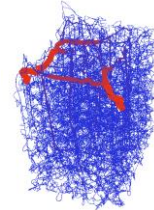
Av4.tortuous (E2.6)
av.4.tortuous.resultsLinninger2015
Pries_In_Vitro



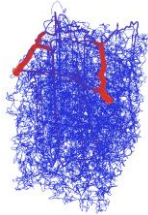
Av5.tortuous (E2.5)
av.5.tortuous.resultsLinninger2015
Pries_In_Vitro



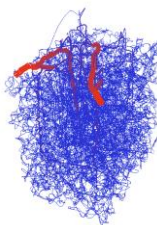
Av6.tortuous (E2.4)
av.6.tortuous.resultsLinninger2015
Pries_In_Vitro



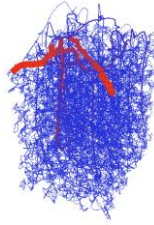
Av7.tortuous (E2.7)
av.7.tortuous.resultsLinninger2015
Pries_In_Vitro



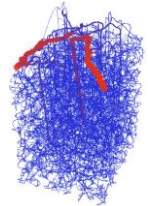
Av8.tortuous (E2.8)
av.8.tortuous.resultsLinninger2015
Pries_In_Vitro



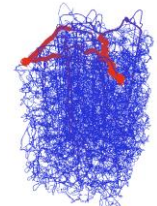
Av9.tortuous (E2.9)
av.9.tortuous.resultsLinninger2015
Pries_In_Vitro



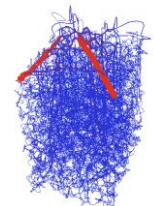
Av10.tortuous (E2.10)
av.10.tortuous.resultsLinninger2015
Pries_In_Vitro



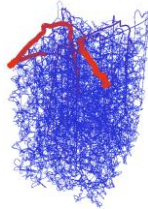
Av11.tortuous (E2.11)
av.11.tortuous.resultsLinninger2015
Pries_In_Vitro



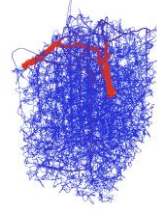
Av12.tortuous (E2.12)
av.12.tortuous.resultsLinninger2015
Pries_In_Vitro



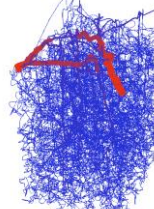
Av13.tortuous (E2.13)
av.13.tortuous.resultsLinninger2015
Pries_In_Vitro



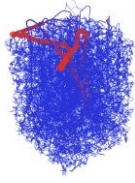
Av14.tortuous (E2.14)
av.14.tortuous.resultsLinninger2015
Pries_In_Vitro



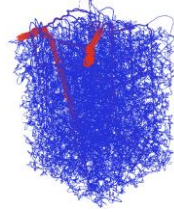
Av15.tortuous (E2.15)
av.15.tortuous.resultsLinninger2015
Pries_In_Vitro



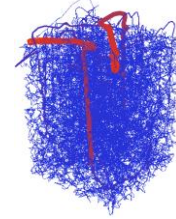
Co1.tortuous (E3.1)
co.1.tortuous.resultsLinninger2015
Pries_In_Vitro.



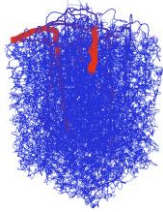
Co2.tortuous (E3.2)
co.2.tortuous.resultsLinninger2015
Pries_In_Vitro.



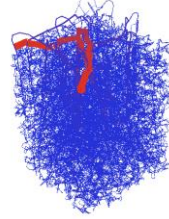
Co3.tortuous (E3.3)
co.3.tortuous.resultsLinninger2015
Pries_In_Vitro.



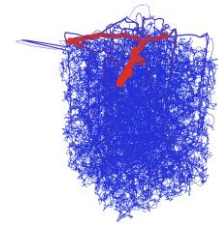
Co4.tortuous (E3.4)
co.4.tortuous.resultsLinninger2015
Pries_In_Vitro.



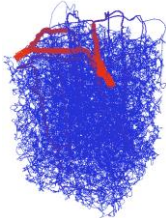
Co5.tortuous (E3.5)
co.5.tortuous.resultsLinninger2015
Pries_In_Vitro.



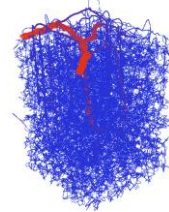
Co6.tortuous (E3.6)
co.6.tortuous.resultsLinninger2015
Pries_In_Vitro.



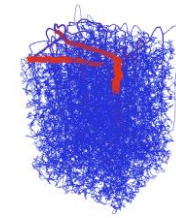
Co7.tortuous (E3.7)
co.7.tortuous.resultsLinninger2015
Pries_In_Vitro.



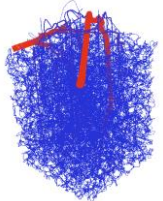
Co8.tortuous (E3.8)
co.8.tortuous.resultsLinninger2015
Pries_In_Vitro.



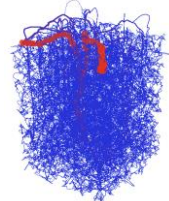
Co9.tortuous (E3.9)
co.9.tortuous.resultsLinninger2015
Pries_In_Vitro.



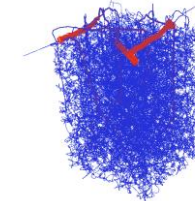
Co10.tortuous (E3.10)
co.10.tortuous.resultsLinninger2015
Pries_In_Vitro.



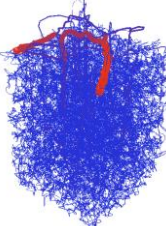
Co11.tortuous (E3.11)
co.11.tortuous.resultsLinninger2015
Pries_In_Vitro.



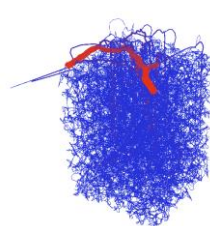
Co12.tortuous (E3.12)
co.12.tortuous.resultsLinninger2015
Pries_In_Vitro.



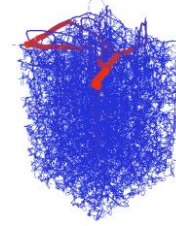
Co13.tortuous (E3.13)
co.13.tortuous.resultsLinninger2015
Pries_In_Vitro.



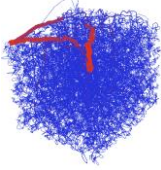
Co14.tortuous (E3.14)
co.14.tortuous.resultsLinninger2015
Pries_In_Vitro.



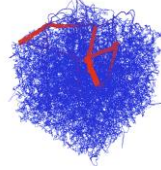
Co15.tortuous (E3.15)
co.15.tortuous.resultsLinninger2015
Pries_In_Vitro.



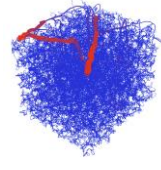
Db1.tortuous (E4.1)
db.1.tortuous.resultsLinninger2015
Pries_In_Vitro



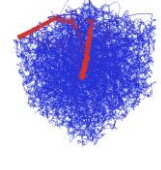
Db2.tortuous (E4.2)
db.2.tortuous.resultsLinninger2015
Pries_In_Vitro



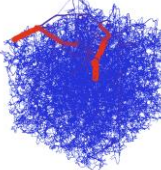
Db3.tortuous (E4.3)
db.3.tortuous.resultsLinninger2015
Pries_In_Vitro



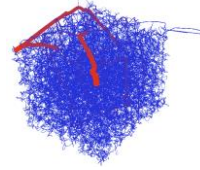
Db4.tortuous (E4.4)
db.4.tortuous.resultsLinninger2015
Pries_In_Vitro



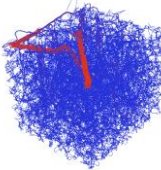
Db5.tortuous (E4.5)
db.5.tortuous.resultsLinninger2015
Pries_In_Vitro



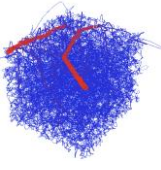
Db6.tortuous (E4.6)
db.6.tortuous.resultsLinninger2015
Pries_In_Vitro



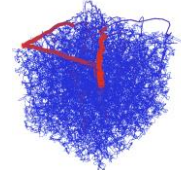
Db7.tortuous (E4.7)
db.7.tortuous.resultsLinninger2015
Pries_In_Vitro



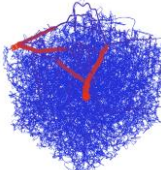
Db8.tortuous (E4.8)
db.8.tortuous.resultsLinninger2015
Pries_In_Vitro



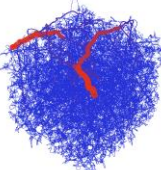
Db9.tortuous (E4.9)
db.9.tortuous.resultsLinninger2015
Pries_In_Vitro



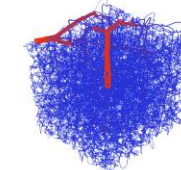
Db10.tortuous (E4.10)
db.10.tortuous.resultsLinninger2015
5Pries_In_Vitro



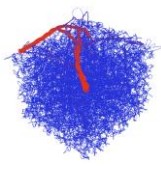
Db11.tortuous (E4.11)
db.11.tortuous.resultsLinninger2015
5Pries_In_Vitro



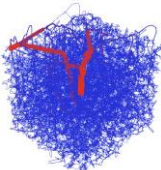
Db12.tortuous (E4.12)
db.12.tortuous.resultsLinninger2015
5Pries_In_Vitro



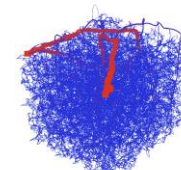
Db13.tortuous (E4.13)
db.13.tortuous.resultsLinninger2015
5Pries_In_Vitro



Db14.tortuous (E4.14)
db.14.tortuous.resultsLinninger2015
5Pries_In_Vitro



Db15.tortuous (E4.15)
db.15.tortuous.resultsLinninger2015
5Pries_In_Vitro



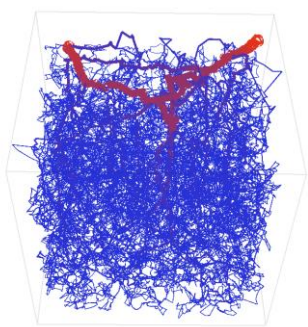
7.39.3 Synthetic Kleinfeld second generation, 100-series, paper 2 (growth paper 1)

Networks with boundary conditions directory: *grantsInventory\ SKF_Serise2_100Series*

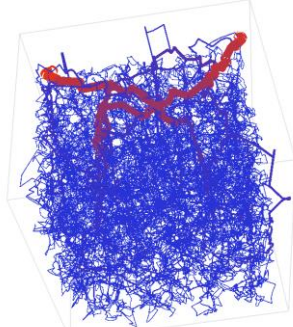
Statistics for each network: *grantsInventory\ SKF_Serise2_100Series*

Biphasic blood flow results: *grantsInventory\ SKF_Serise2_100Series *

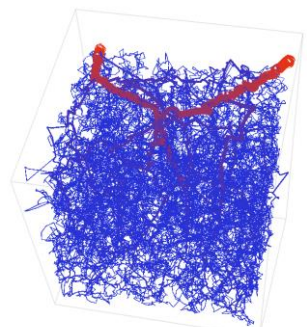
S1.101.tourtuous (E1.101)



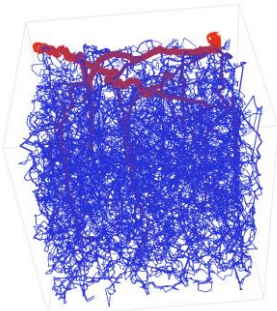
S1.102.tourtuous (E1.102)



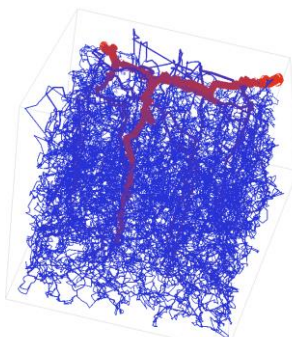
S1.103.tourtuous (E1.103)



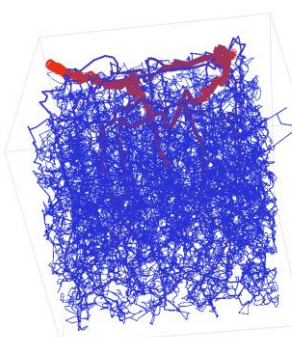
S1.104.tourtuous (E1.104)



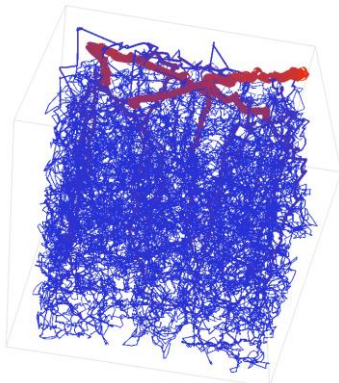
S1.105.tourtuous (E1.105)



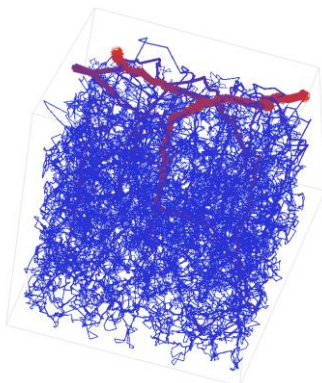
S1.106.tourtuous (E1.106)



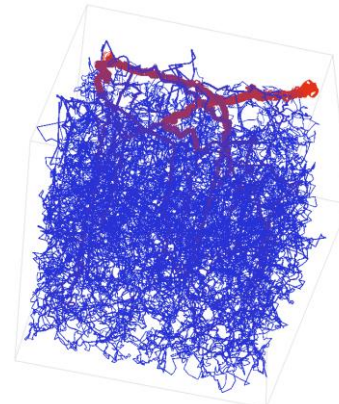
S1.107.tourtuous (E1.107)



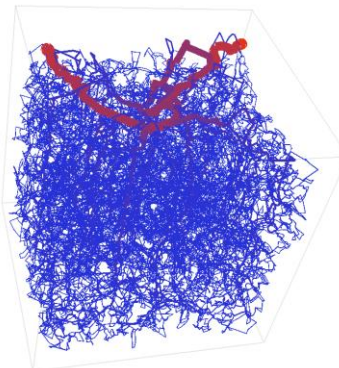
S1.108.tourtuous (E1.108)



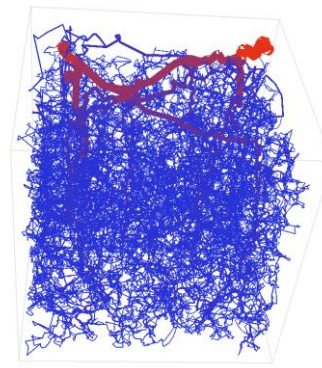
S1.109.tourtuous (E1.109)



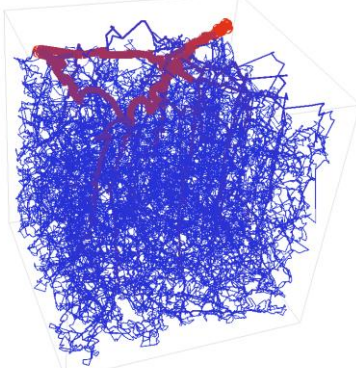
S1.110.tourtuous (E1.110)



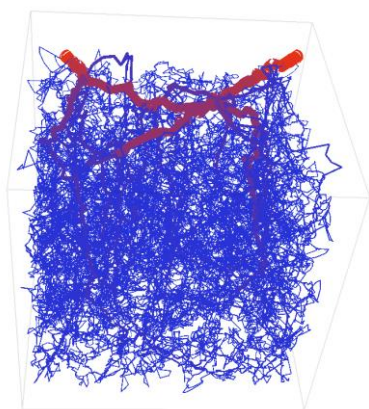
S1.111.tourtuous (E1.111)



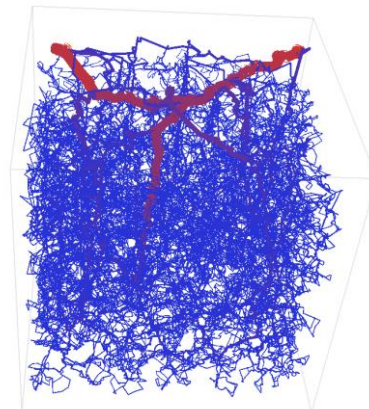
S1.112.tourtuous (E1.112)



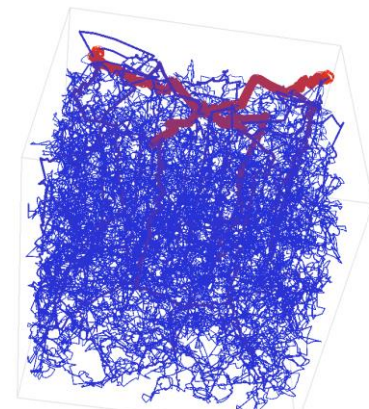
S1.113.tourtuuous (E1.113)



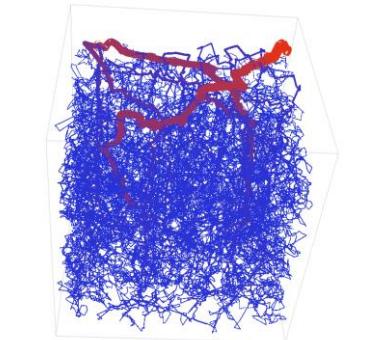
S1.114.tourtuuous (E1.114)



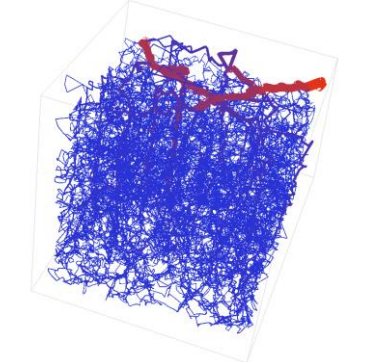
S1.115.tourtuuous (E1.115)



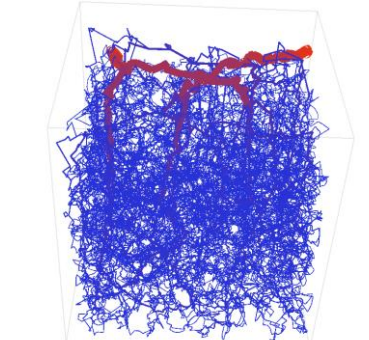
S1.116.tourtuuous (E1.116)



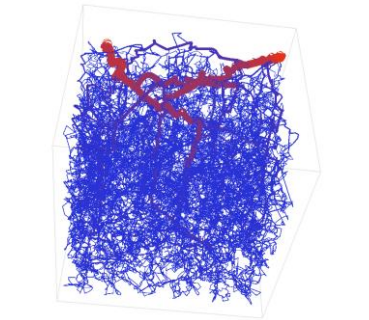
S1.117.tourtuuous (E1.117)



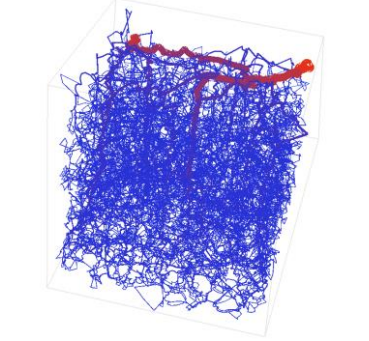
S1.118.tourtuuous (E1.118)



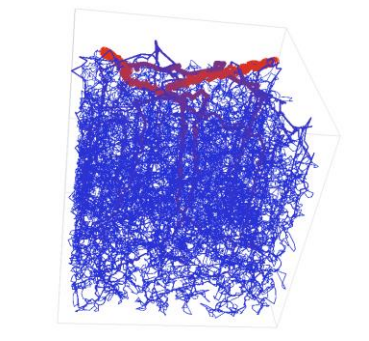
S1.119.tourtuuous (E1.119)



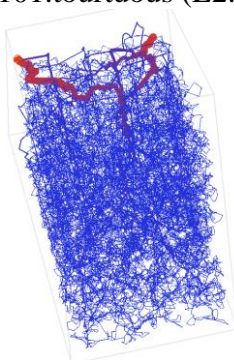
S1.120.tourtuuous (E1.120)



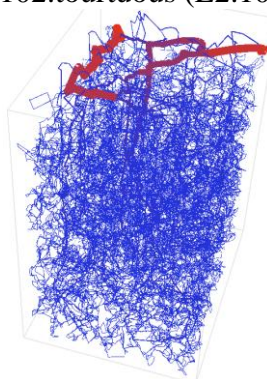
S1.121.tourtuuous (E1.121)



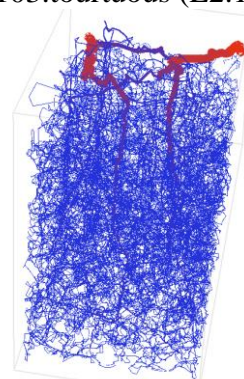
S2.101.tourtuous (E2.101)



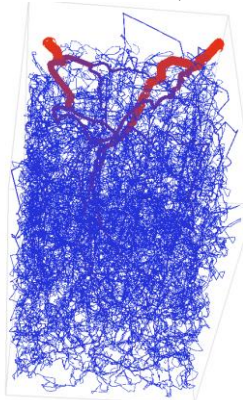
S2.102.tourtuous (E2.102)



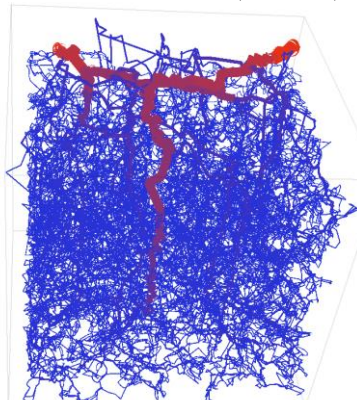
S2.103.tourtuous (E2.103)



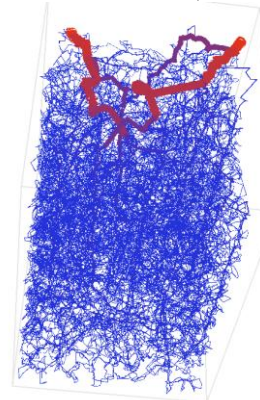
S2.104.tourtuous (E2.104)



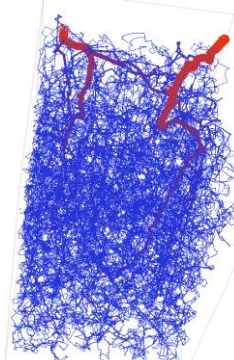
S2.105.tourtuous (E2.105)



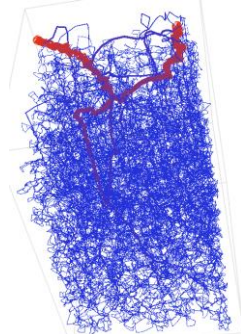
S2.106.tourtuous (E2.106)



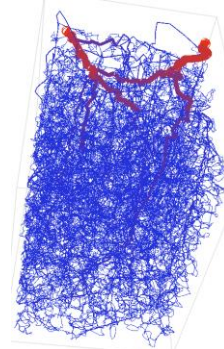
S2.107.tourtuous (E2.107)



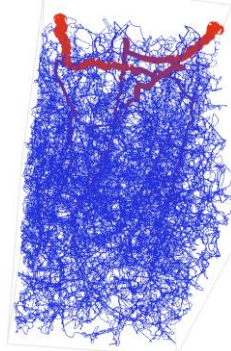
S2.108.tourtuous (E2.108)



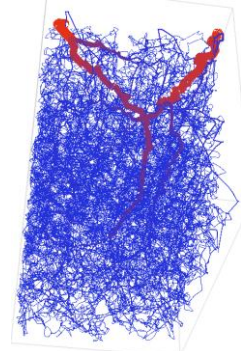
S2.109.tourtuous (E2.109)



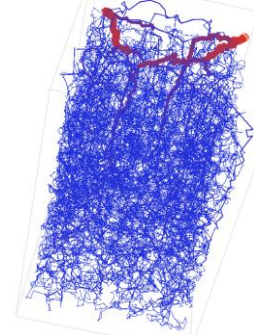
S2.110.tourtuous (E2.110)



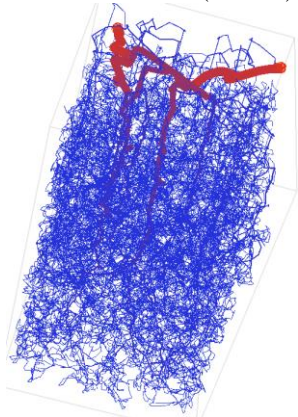
S2.111.tourtuous (E2.111)



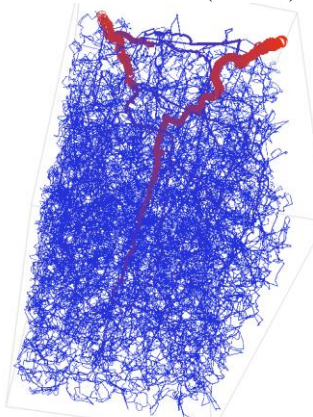
S2.112.tourtuous (E2.112)



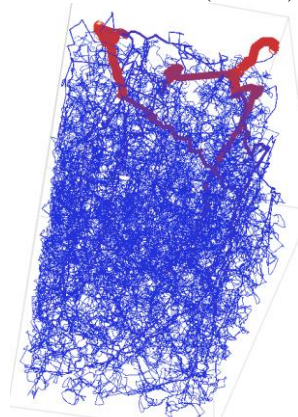
S2.113.tourtuous (E2.113)



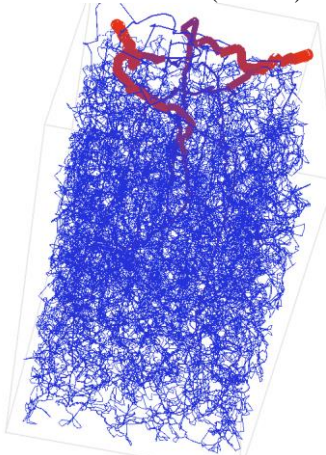
S2.114.tourtuous (E2.114)



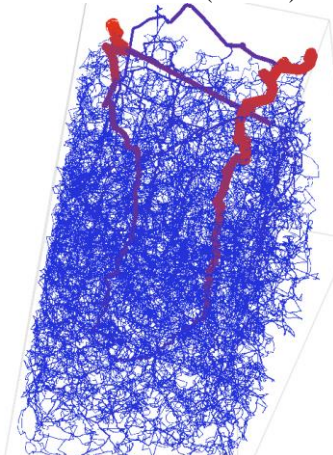
S2.115.tourtuous (E2.115)



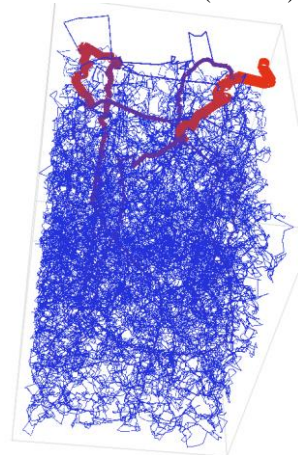
S2.116.tourtuous (E2.116)



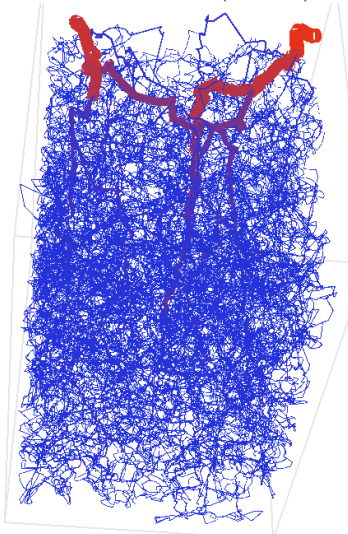
S2.117.tourtuous (E2.117)



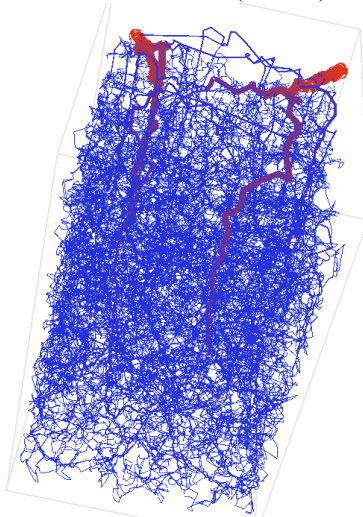
S2.118.tourtuous (E2.118)



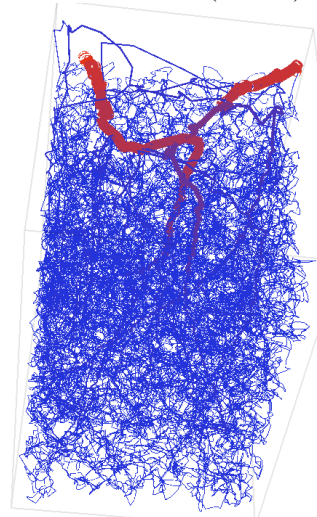
S2.119.tourtuous (E2.119)



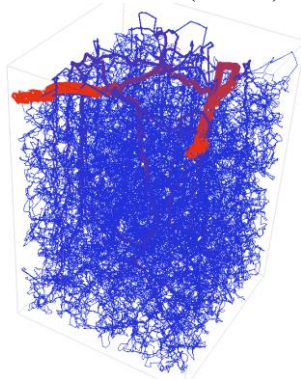
S2.120.tourtuous (E2.120)



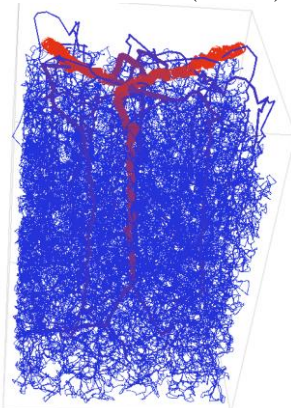
S2.121.tourtuous (E2.121)



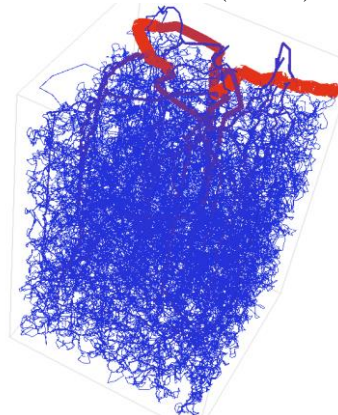
S3.101.tourtuuous (E3.101)



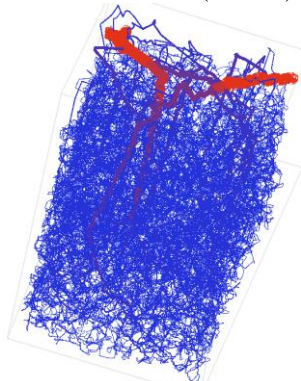
S3.102.tourtuuous (E3.102)



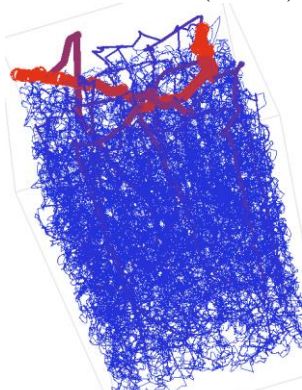
S3.103.tourtuuous (E3.103)



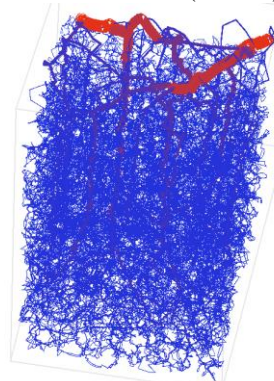
S3.104.tourtuuous (E3.104)



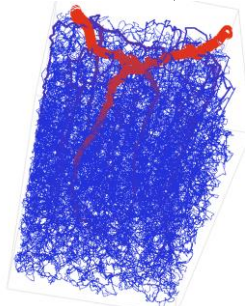
S3.105.tourtuuous (E3.105)



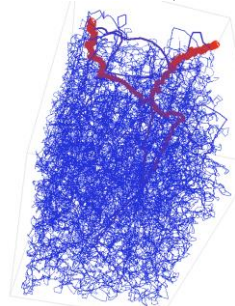
S3.106.tourtuuous (E3.106)



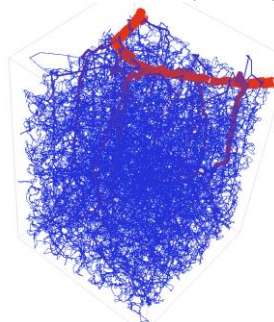
S3.107.tourtuuous (E3.107)



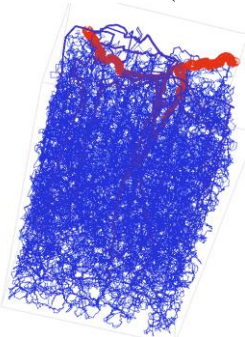
S3.108.tourtuuous (E3.108)



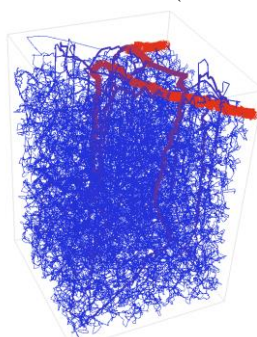
S3.109.tourtuuous (E3.109)



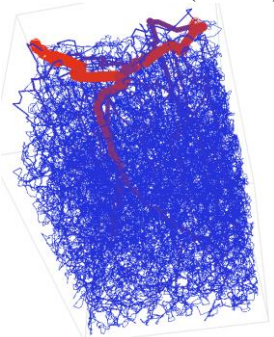
S3.110.tourtuuous (E3.110)



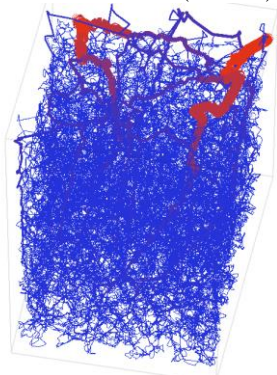
S3.111.tourtuuous (E3.111)



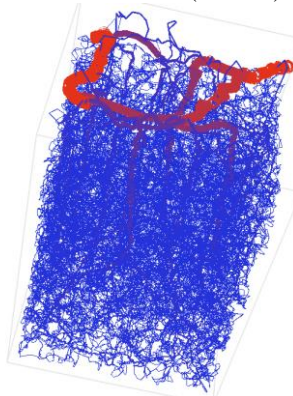
S3.112.tourtuuous (E3.112)



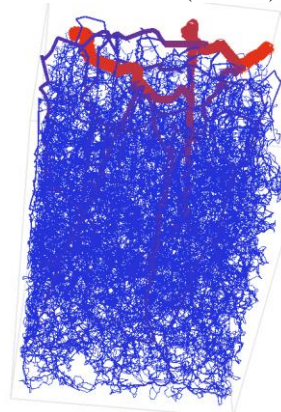
S3.113.tourtuuous (E3.113)



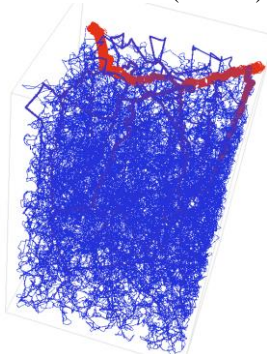
S3.114.tourtuuous (E3.114)



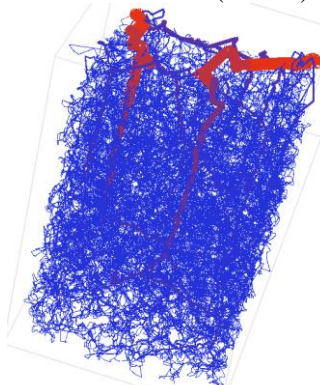
S3.115.tourtuuous (E3.115)



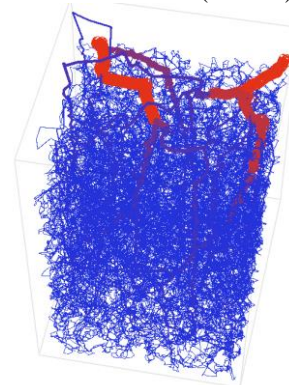
S3.116.tourtuuous (E3.116)



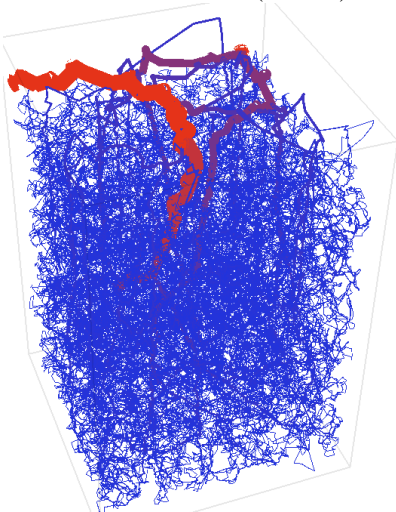
S3.117.tourtuuous (E3.117)



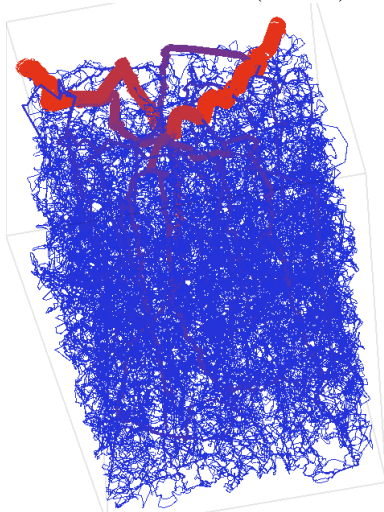
S3.118.tourtuuous (E3.118)



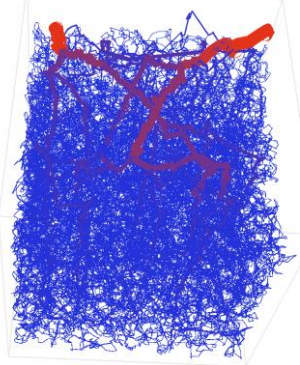
S3.119.tourtuuous (E3.119)



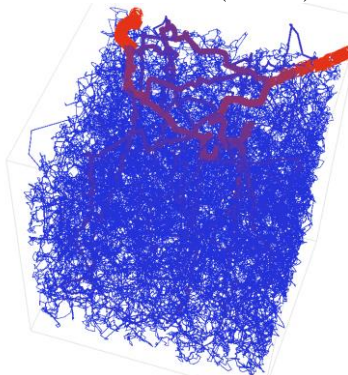
S3.120.tourtuuous (E3.120)



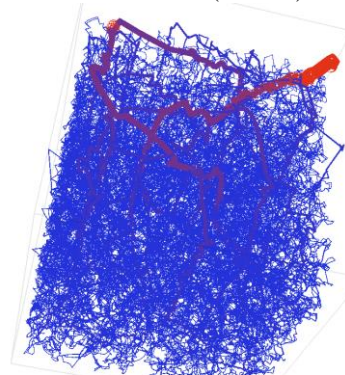
S4.101.tourtuuous (E4.101)



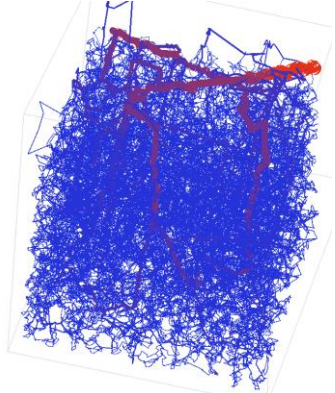
S4.102.tourtuuous (E4.102)



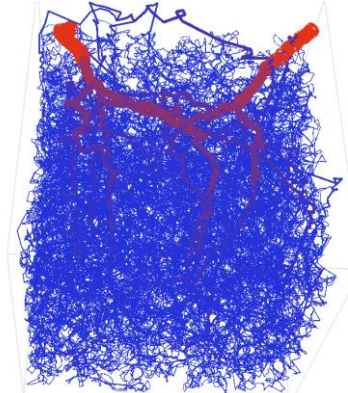
S4.103.tourtuuous (E4.103)



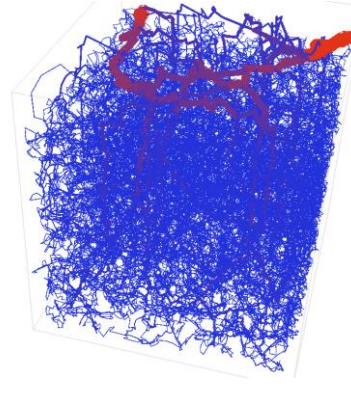
S4.104.tourtuuous (E4.104)



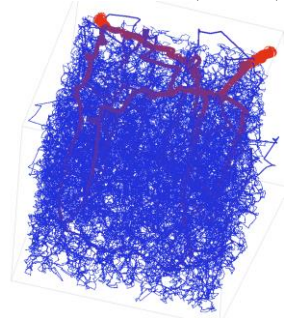
S4.105.tourtuuous (E4.105)



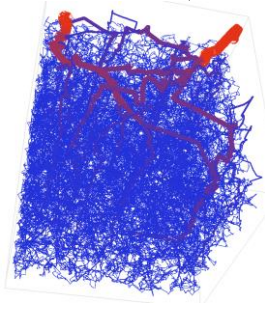
S4.106.tourtuuous (E4.106)



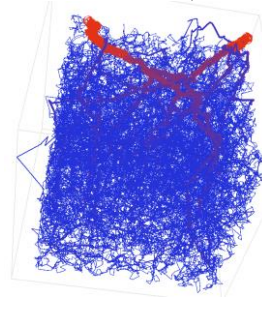
S4.107.tourtuuous (E4.107)



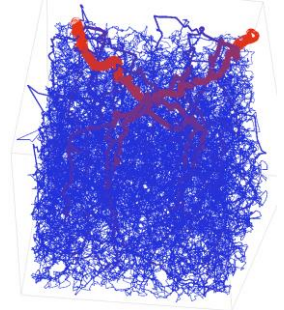
S4.108.tourtuuous (E4.108)



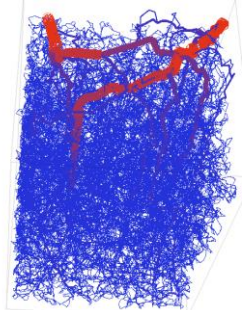
S4.109.tourtuuous (E4.109)



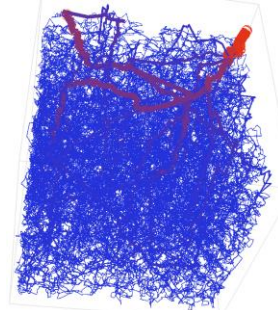
S4.110.tourtuuous (E4.110)



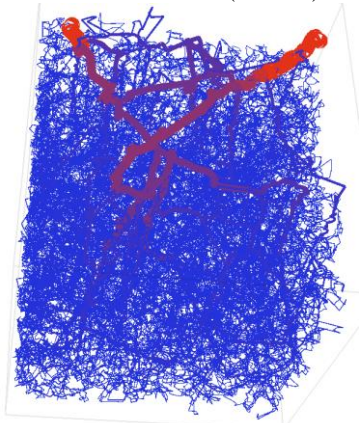
S4.111.tourtuuous (E4.111)



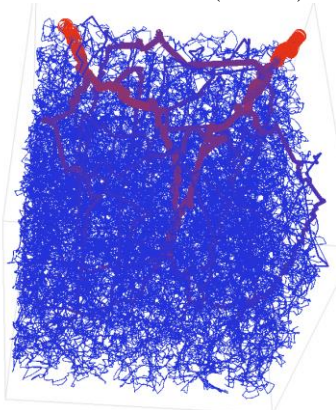
S4.112.tourtuuous (E4.112)



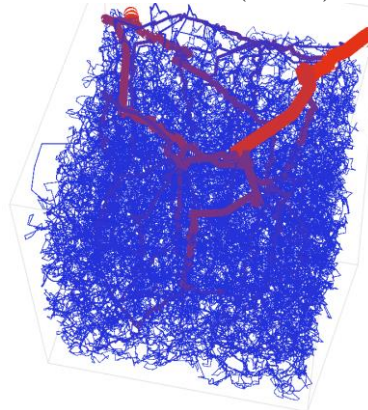
S4.113.tourtuous (E4.113)



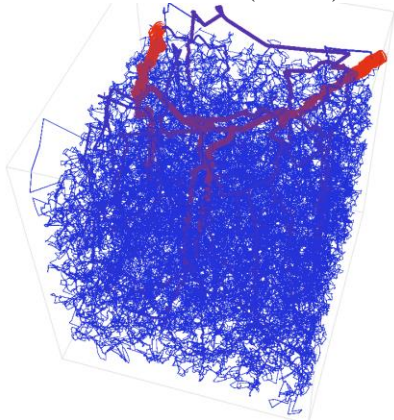
S4.114.tourtuous (E4.114)



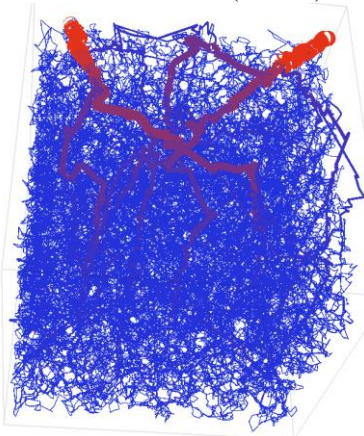
S4.115.tourtuous (E4.115)



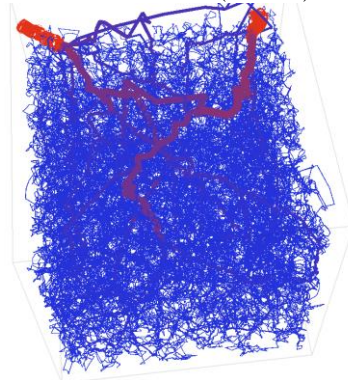
S4.116.tourtuous (E4.116)



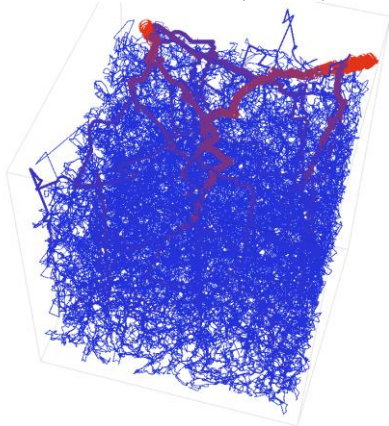
S4.117.tourtuous (E4.117)



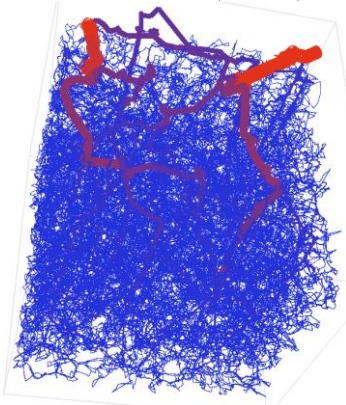
S4.118.tourtuous (E4.118)



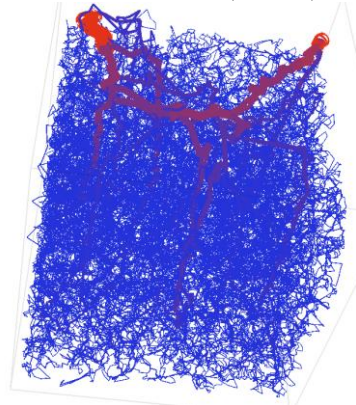
S4.119.tourtuous (E4.119)



S4.120.tourtuous (E4.120)



S4.121.tourtuous (E4.121)

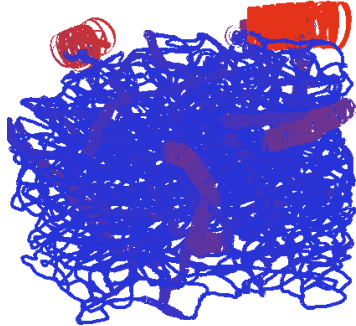


7.39.4 Empirical Boas

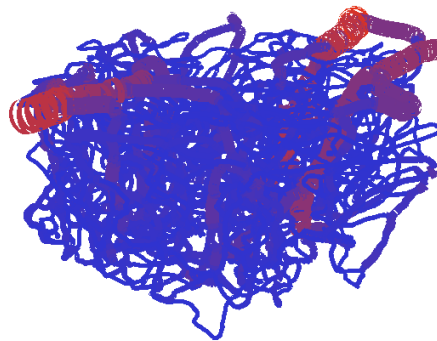
Networks with boundary conditions directory: *grantsInventory\EB_SeriesNetworks*

Statistics for each network: *grantsInventory\EB_SeriesNetworks\Gagnon*

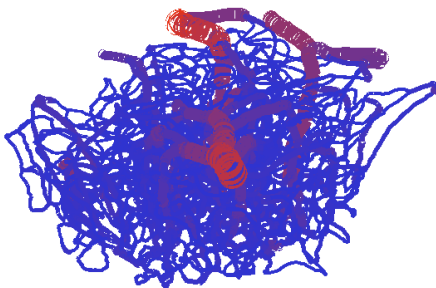
Gagnon_1\correctedGraph.cs31



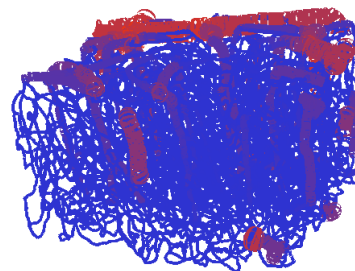
Gagnon_3\correctedGraph.cs31



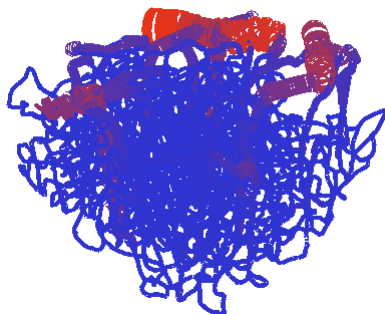
Gagnon_4\correctedGraph.cs31



Gagnon_5\correctedGraph.cs31



Gagnon_6\correctedGraph.cs31

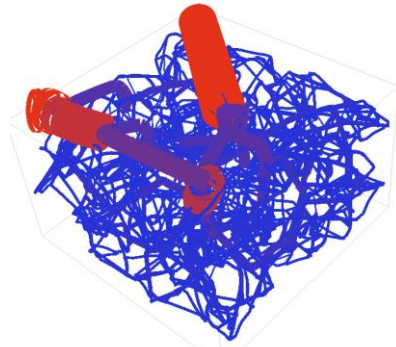


7.39.5 Synthetic Boas second generation, 100-series

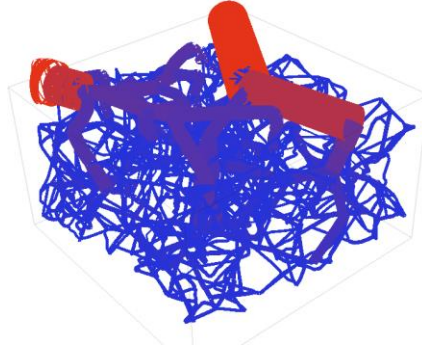
Networks with boundary conditions directory: *grantsInventory\SB_Series2_100Series*

Statistics for each network: *grantsInventory\SB_Series2_100Series*

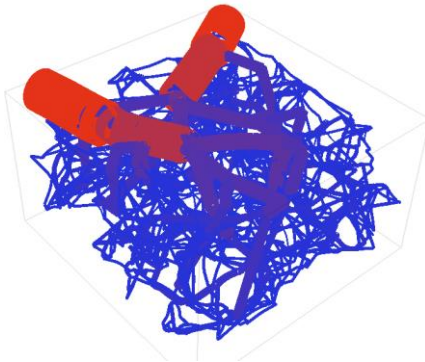
SB1.201.tortuous (email1)
Accompanied by .art, .vein and regular
network.



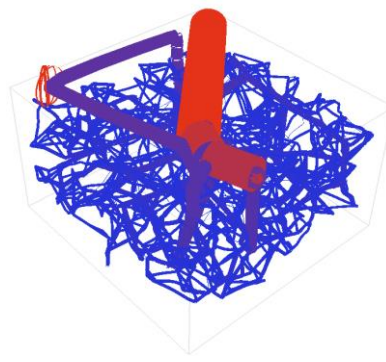
SB1.202.tortuous (email1)
Accompanied by .art, .vein and regular
network.



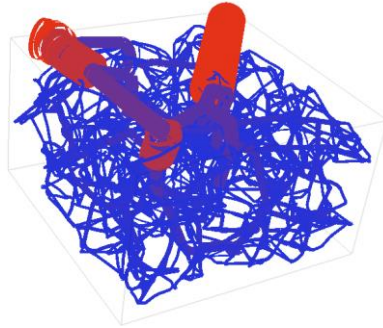
SB1.203.tortuous (email1)
Accompanied by .art, .vein and regular
network.



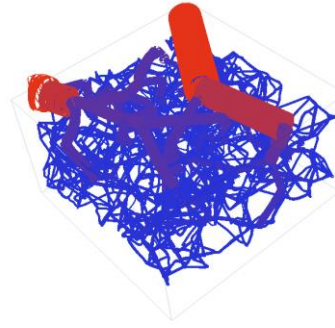
SB1.204.tortuous (email1)
Accompanied by .art, .vein and regular
network.



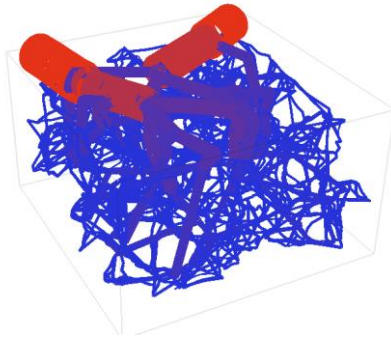
SB1.201.tortuous (email1)
Accompanied by .art, .vein and
regular network.



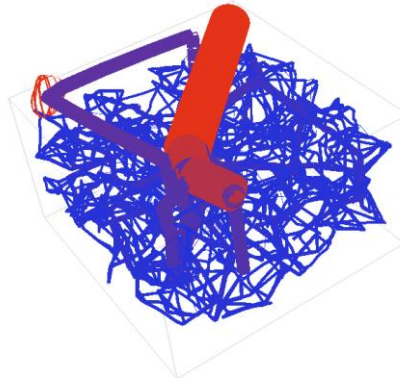
SB1.202.tortuous (email1)
Accompanied by .art, .vein and regular
network.



SB1.203.tortuous (email1)
Accompanied by .art, .vein and
regular network.



SB1.204.tortuous (email1)
Accompanied by .art, .vein and regular
network.

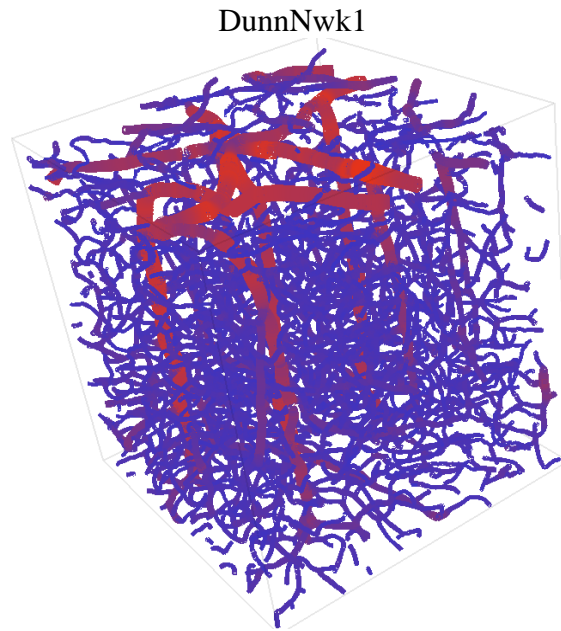


7.39.6 Empirical Dunn

Networks with boundary conditions directory: *grantsInventory\ED_SeriesNetworks*

Statistics for each network: *grantsInventory\ED_SeriesNetworks*

Biphasic blood flow results: *grantsInventory\ED_SeriesNetworks*



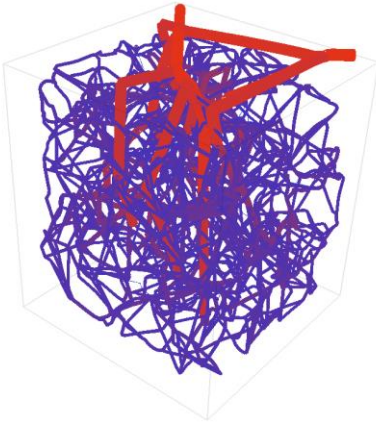
7.39.7 Synthetic Dunn second generation, 100-series

Networks with boundary conditions directory: *grantsInventory\SD_Series2_100Series*

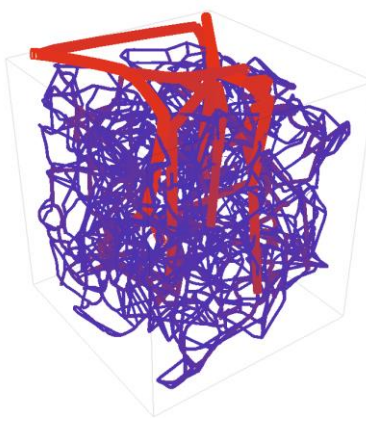
Statistics for each network: *grantsInventory\SD_Series2_100Series*

Biphasic blood flow results: *grantsInventory\SD_Series2_100Series*

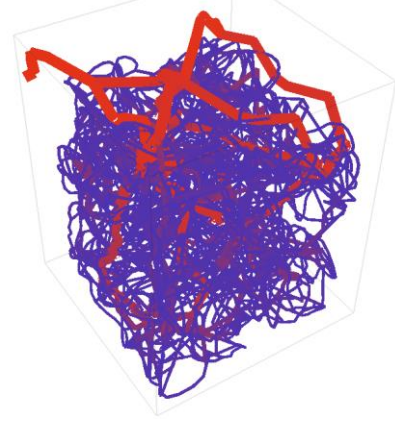
SD1.202.tortuous
Accompanied by .art, .vein and
regular network.



SD1.203.tortuous
Accompanied by .art, .vein and
regular network.



SD1.269.tortuous



7.39.8 Mouse 1 (Sled reconstruction)

MCA biphasic blood flow paper

Network Location: *grantsInventory\SledNetworks\MCAArteriesOnly_forBiphasicPaper*

Biphasic blood flow results file name:

MCAV1.resizedx1000.correcedDia.resultsLinninger2015Pries_In_Vitro.cs31

MCA biphasic blood flow paper

Network Location: *grantsInventory\SledNetworks\MCAArteriesOnly_forBiphasicPaper*

Biphasic blood flow results file name:

CortexGrowthWithClosureCleaned.AddedTortuosity.resultsLinninger2015Pries_In_Vitro.cs31

Hemisphere

All hemisphere structures are put in a directory with all supporting files, including each stage of growth (pial, pial and penetrators, and pial/penetrators/capillaries prior to closure).

Network Location: *grantsInventory\SledNetworks\MCATerritoryV1_forBiphasicPaper*

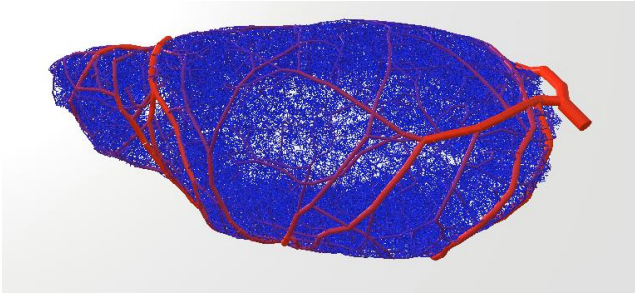
Biphasic blood flow results file name:

hemisphereArtVenClosureWithClosurev3.AddedTortuosity.resultsLinninger2015Pries_In_Vitro

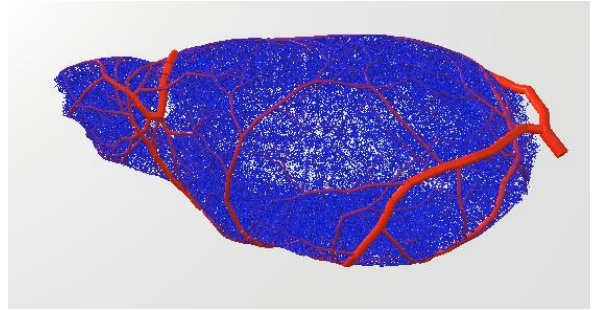
Linear flow results: *hemisphereArtVenClosureWithClosurev3.results.cs31*

Mouse Model	Name	Type	Directory
Mouse 1	M1_H1	hemisphere	Hemisphere1\
Mouse 1	M1_H2	hemisphere	Hemisphere2\
Mouse 1	M1_H3	hemisphere	Hemisphere3\
Mouse 1	M1_H4	hemisphere	Hemisphere4\
Mouse 1	M1_H5	hemisphere	Hemisphere5\
Mouse 1	M1_H6	hemisphere	Hemisphere6\
Mouse 1	M1_H7	hemisphere	Hemisphere7\

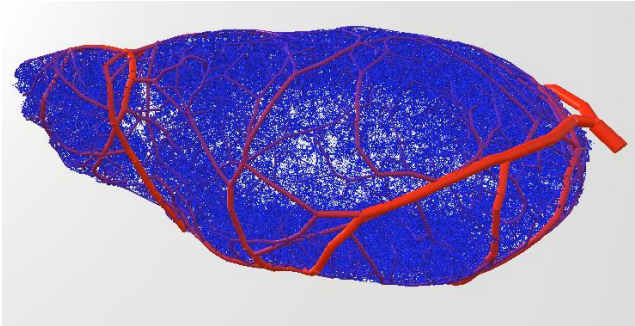
M1_H1



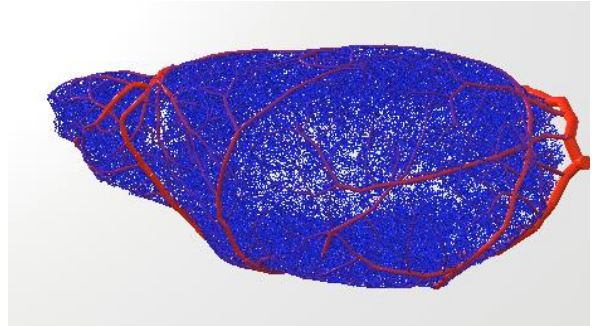
M1_H2



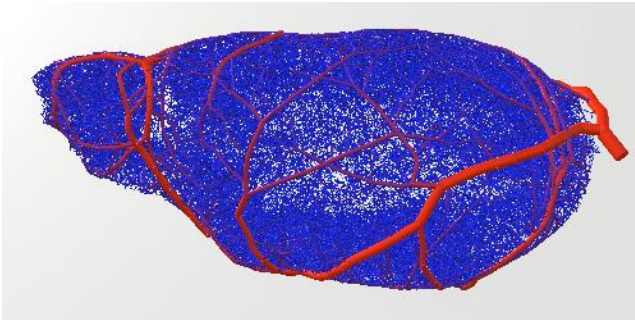
M1_H3



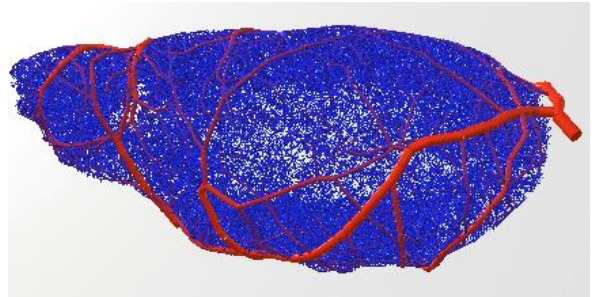
M1_H4



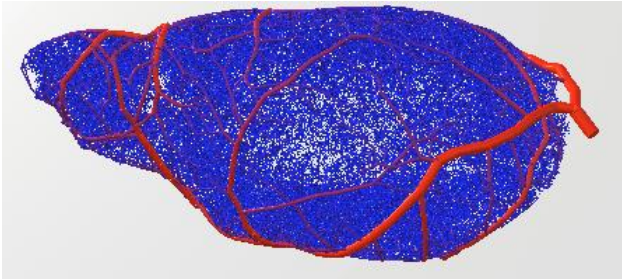
M1_H5



M1_H6



M1_H7



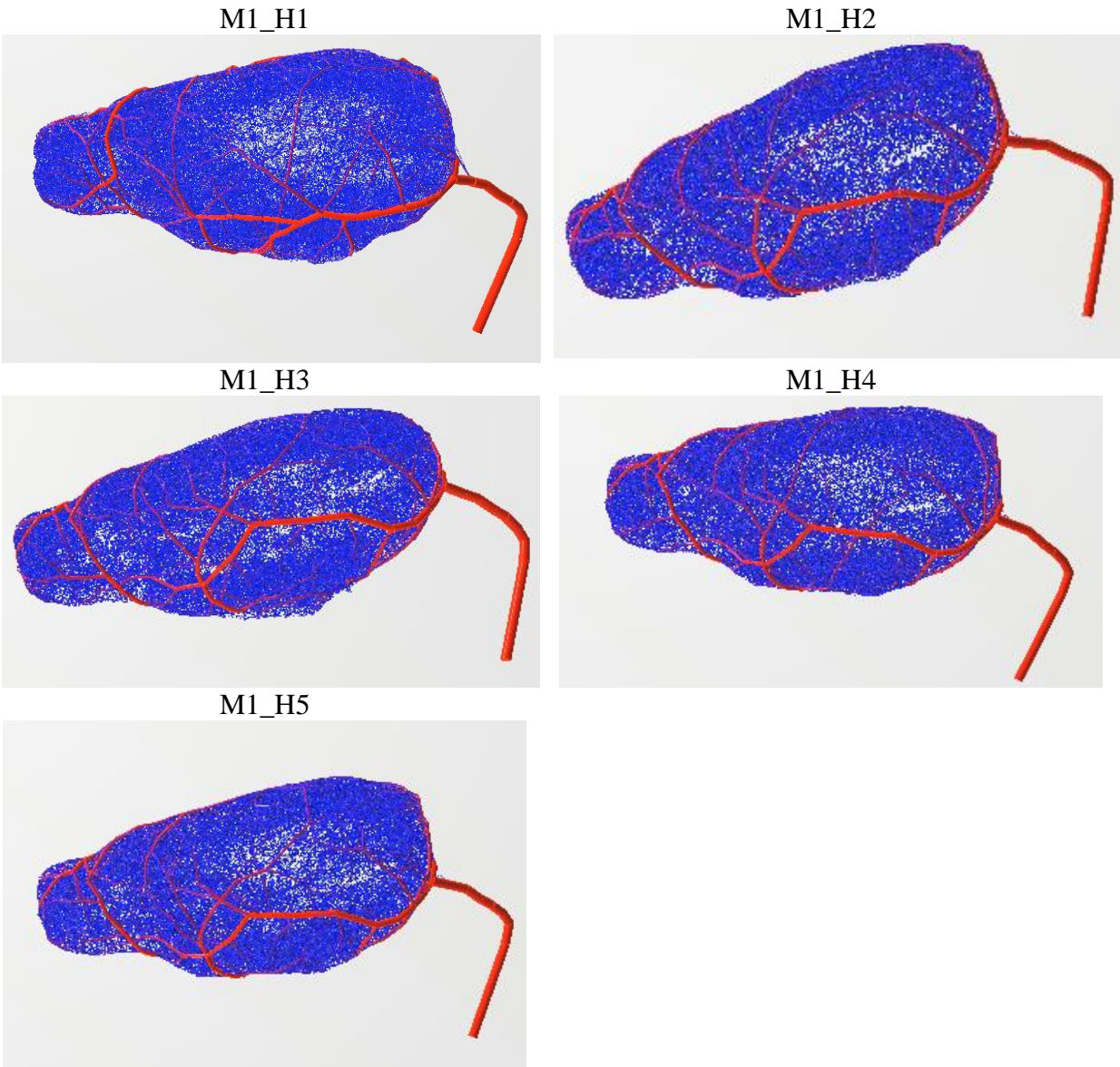
7.39.9 Mouse 2 hemispheres (reconstructed from Allen Brain Institute images)

All hemisphere structures are put in a directory with all supporting files, including each stage of growth (pial, pial and penetrators, and pial/penetrators/capillaries prior to closure).

Network Location:

grantsInventory\ AllenBrainInstituteReconstruction\MCATerritoryV1_forBiphasicPaper

Mouse Model	Name	Type	Directory
Mouse 2	M2_H1	hemisphere	Hemisphere1\
Mouse 2	M2_H2	hemisphere	Hemisphere2\
Mouse 2	M2_H3	hemisphere	Hemisphere3\
Mouse 2	M2_H4	hemisphere	Hemisphere4\
Mouse 2	M2_H5	hemisphere	Hemisphere5\



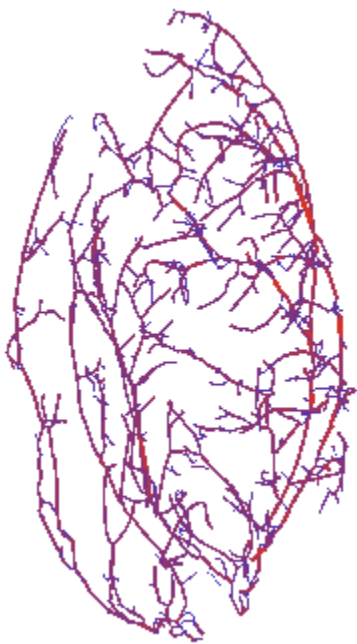
Quick Reference of pial surface areas:

	Surf Area Mouse 1	Surf Area Mouse 2
MCA	39.468	48.347
ACA	20.73	25.376
PCA	15.873	20.036
total:	76.071	93.759

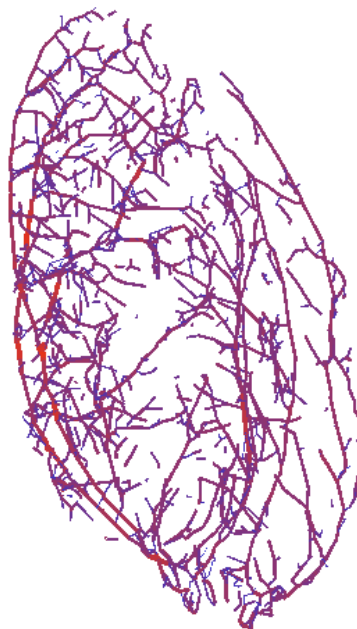
7.39.10 Horowitz heart reconstructions

File locations: *grantsInventory\Horowitz*

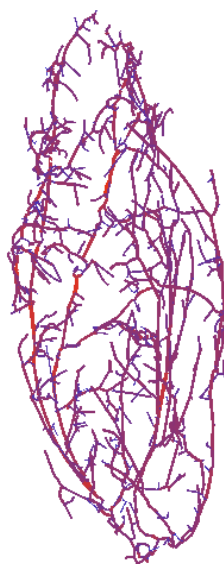
M226\M226FINAL



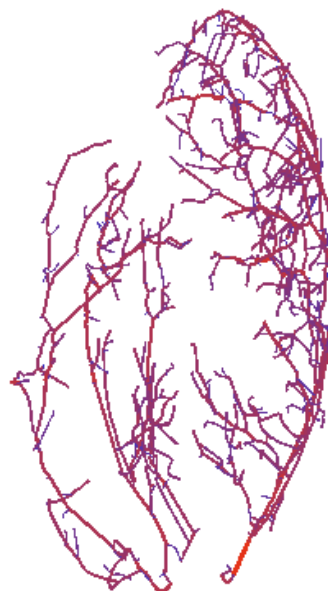
M226\M226Original



M262\M262FINAL



M285\M285FINAL



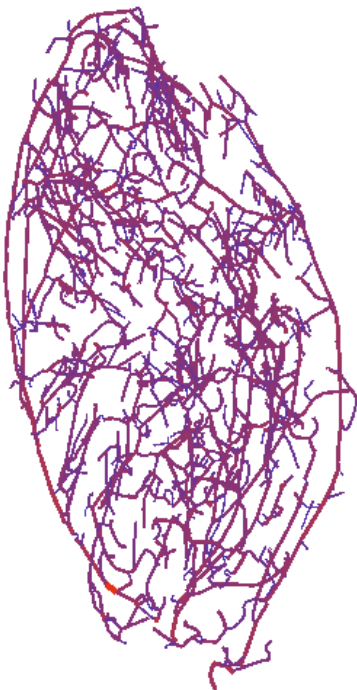
M286\M286FINAL



M315\M315.Final



M316\FINAL

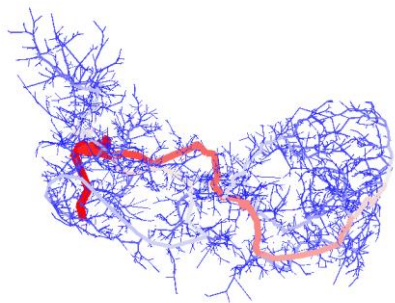


7.39.11 Human pial growth

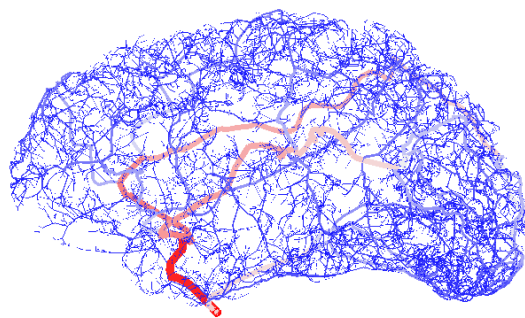
Numerous trials of the human pial growth from backbone were investigated. These are listed below.

File locations: *grantsInventory\HumanGrowthTesting*

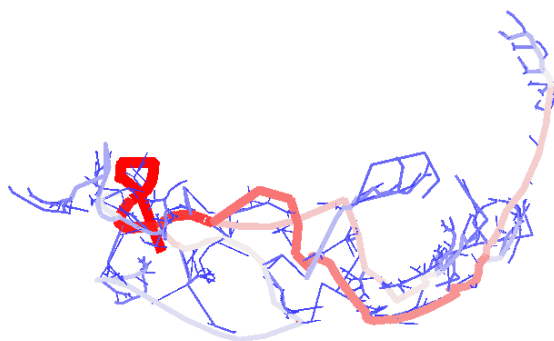
coarseRun\KTHHemisphereV4.cs31.pials9.cs31



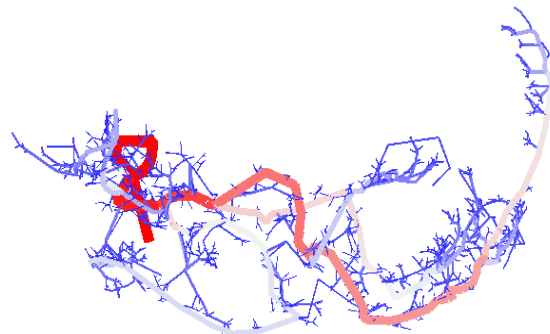
Hemisphere_attempt1\HemisphereV2.cs31.pials35



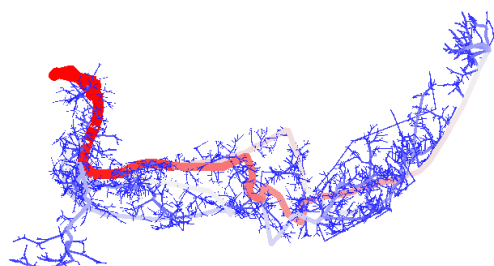
fineBrainRun\KTHHemisphereV4.cs31.pials1



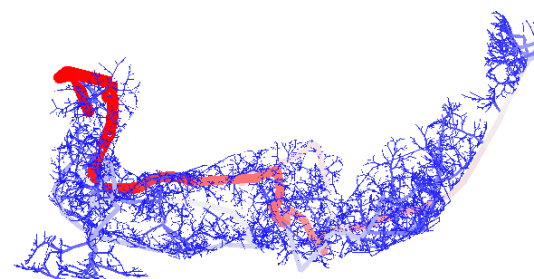
fineBrainRun\KTHHemisphereV4.cs31.pials2



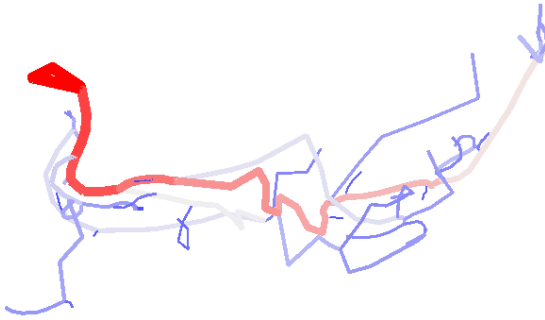
fineBrainRun\KTHHemisphereV4.cs31.pials3



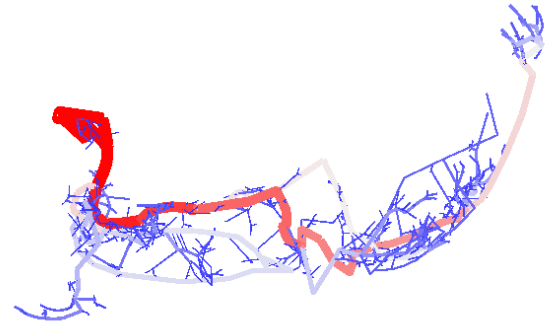
fineBrainRun\KTHHemisphereV4.cs31.pials4



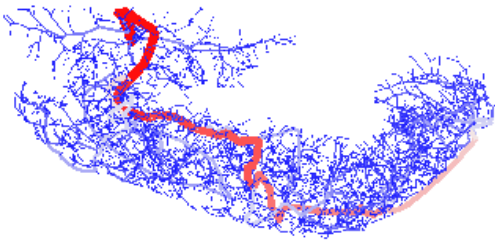
hemisphereV4\KTHHemisphereV4



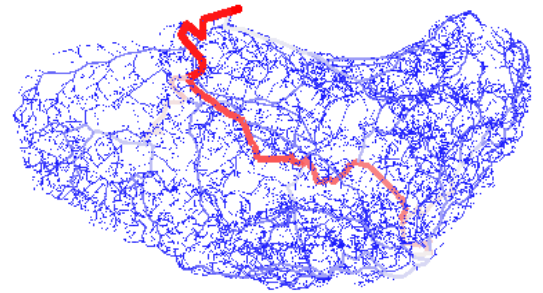
hemisphereV4\KTHHemisphereV4.cs31.pials1



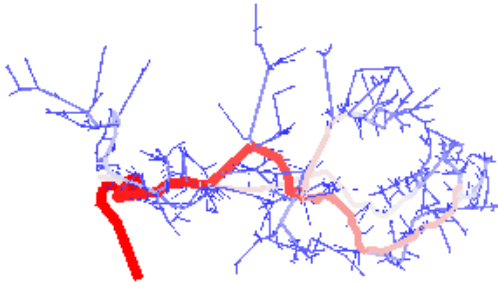
hemisphereV4\KTHHemisphereV4.cs31.pials13



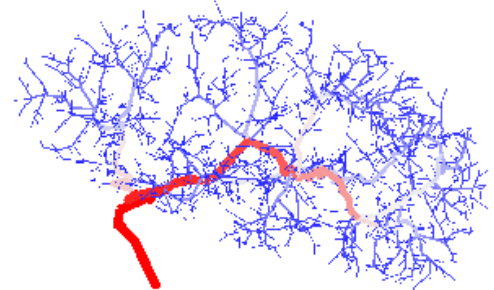
hemisphereV4\KTHHemisphereV4.cs31.pials36



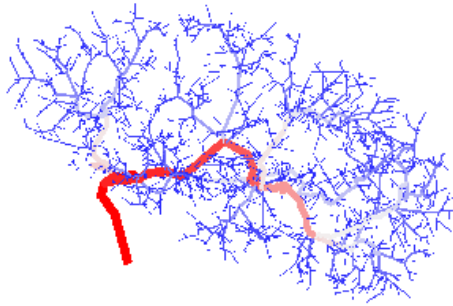
mcaV5 lowDensity\KTHHemisphereV5.cs31.pials1



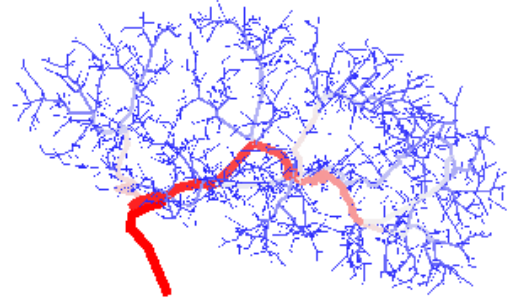
mcaV5 lowDensity\KTHHemisphereV5.cs31.pials10



mcaV5 lowDensity\KTHHemisphereV5.cs31.pials25



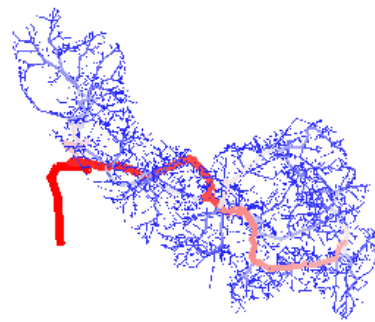
mcaV5 lowDensity\KTHHemisphereV5.cs31.pials50



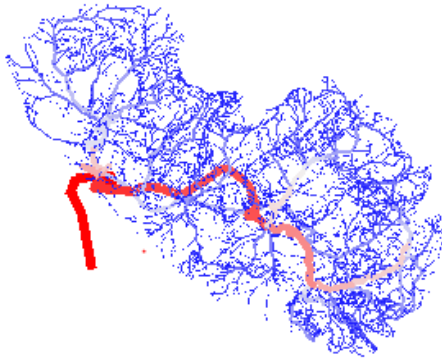
mcaV5 fullDensity\KTHHemisphereV5.cs31.pials1



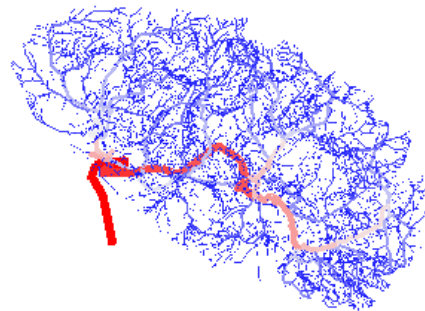
mcaV5 fullDensity\KTHHemisphereV5.cs31.pials10



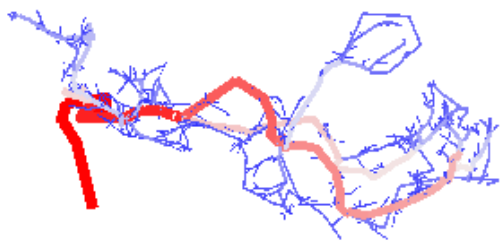
mcaV5 fullDensity\KTHHemisphereV5.cs31.pials25



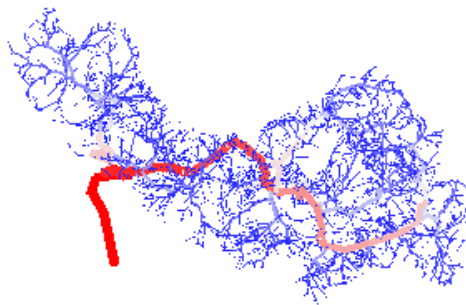
mcaV5 fullDensity\KTHHemisphereV5.cs31.pials50



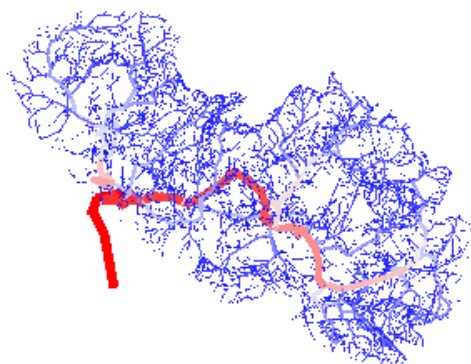
mcaV5 fullDensity2\KTHHemisphereV5.cs31.pials1



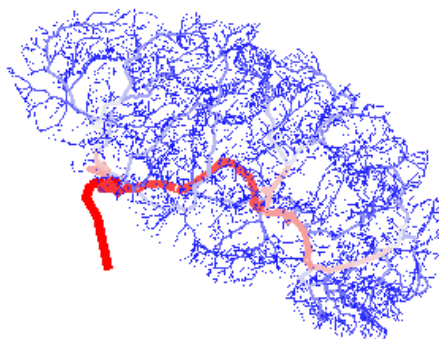
mcaV5 fullDensity2\KTHHemisphereV5.cs31.pials10



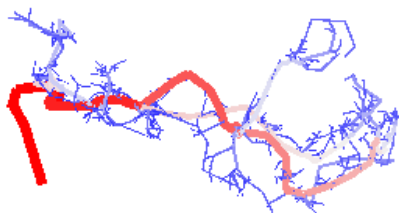
mcaV5 fullDensity2\KTHHemisphereV5.cs31.pials25



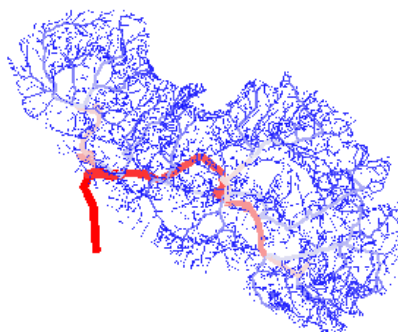
mcaV5 fullDensity2\KTHHemisphereV5.cs31.pials50



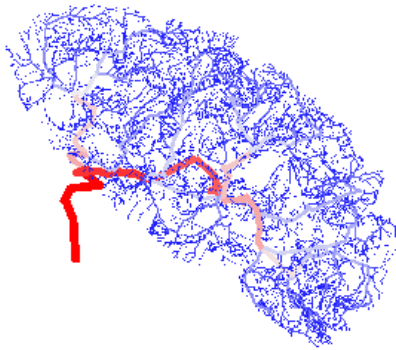
mcaV5 fullDensity3\KTHHemisphereV5.cs31.pials1



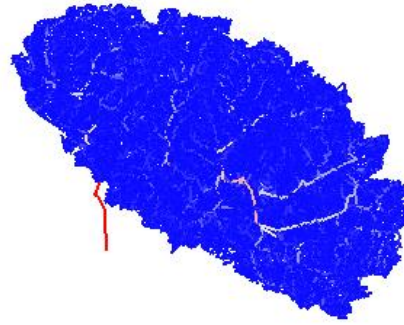
mcaV5 fullDensity3\KTHHemisphereV5.cs31.pials25



mcaV5 fullDensity3\KTHHemisphereV5.cs31.pials50



mcaV5 fullDensity3\KTHHemisphereV5.pials.Capillaries



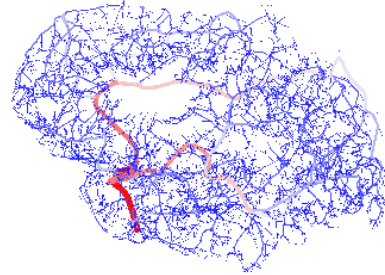
moreDenseHemisphere\HemisphereV3.cs31.pials1



aca_V3



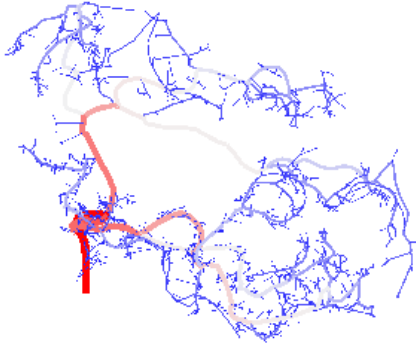
moreDenseHemisphere\HemisphereV3.cs31.pials17



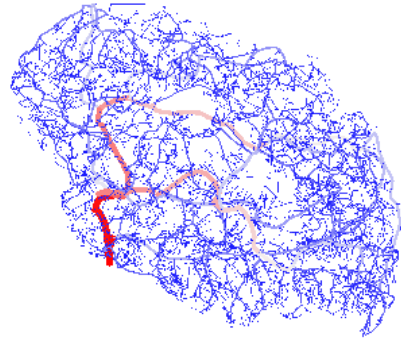
hemisphereV3.cs4



HemisphereV3.cs31.pials1



HemisphereV3.cs31.pials19



kt.cleaned.cs31



MCA_V5.Connected.cleanedflipped.final



MCA_V5



KTHHemisphereV5



8 Cited Literature

1. Peers C, Dallas ML, Boycott HE, Scragg JL, Pearson HA, Boyle JP. Hypoxia and Neurodegeneration. *Ann N Y Acad Sci*. 2009 Oct 1;1177(1):169–77.
2. Peers C, Pearson HA, Boyle JP. Hypoxia and Alzheimer’s disease. *Essays Biochem*. 2007 Aug 10;43:153–64.
3. Bullitt E, Zeng D, Mortamet B, Ghosh A, Aylward SR, Lin W, et al. The effects of healthy aging on intracerebral blood vessels visualized by magnetic resonance angiography. *Neurobiol Aging*. 2010;31(2):290–300.
4. Moeini M, Lu X, Avti PK, Damseh R, Bélanger S, Picard F, et al. Compromised microvascular oxygen delivery increases brain tissue vulnerability with age. *Sci Rep*. 2018 May 29;8(1):8219.
5. Silasi G, She J, Boyd JD, Xue S, Murphy TH. A Mouse Model of Small-Vessel Disease that Produces Brain-Wide-Identified Microocclusions and Regionally Selective Neuronal Injury. *J Cereb Blood Flow Metab*. 2015 May 1;35(5):734–8.
6. Wang M, Iliff JJ, Liao Y, Chen MJ, Shinseki MS, Venkataraman A, et al. Cognitive Deficits and Delayed Neuronal Loss in a Mouse Model of Multiple Microinfarcts. *J Neurosci*. 2012 Dec 12;32(50):17948.
7. Okamoto Y, Yamamoto T, Kalaria RN, Senzaki H, Maki T, Hase Y, et al. Cerebral hypoperfusion accelerates cerebral amyloid angiopathy and promotes cortical microinfarcts. *Acta Neuropathol (Berl)*. 2012 Mar 1;123(3):381–94.
8. Desjardins M, Berti R, Lefebvre J, Dubeau S, Lesage F. Aging-related differences in cerebral capillary blood flow in anesthetized rats. *Neurobiol Aging*. 2014 Aug 1;35(8):1947–55.
9. Gagnon L, Sakadžić S, Lesage F, Musacchia JJ, Lefebvre J, Fang Q, et al. Quantifying the Microvascular Origin of BOLD-fMRI from First Principles with Two-Photon Microscopy and an Oxygen-Sensitive Nanoprobe. *J Neurosci*. 2015 Feb 25;35(8):3663–75.
10. Lorthois S, Cassot F, Lauwers F. Simulation study of brain blood flow regulation by intra-cortical arterioles in an anatomically accurate large human vascular network. Part II: Flow variations induced by global or localized modifications of arteriolar diameters. *NeuroImage*. 2011 Feb 14;54(4):2840–53.
11. Kim JH, Ress D. Arterial impulse model for the BOLD response to brief neural activation. *NeuroImage*. 2016 Jan 1;124:394–408.
12. Griffeth VEM, Buxton RB. A theoretical framework for estimating cerebral oxygen metabolism changes using the calibrated-BOLD method: Modeling the effects of blood volume distribution, hematocrit, oxygen extraction fraction, and tissue signal properties on the BOLD signal. *NeuroImage*. 2011 Sep 1;58(1):198–212.
13. Boas DA, Jones SR, Devor A, Huppert TJ, Dale AM. A vascular anatomical network model of the spatio-temporal response to brain activation. *NeuroImage*. 2008 Apr 15;40(3):1116–29.
14. Zhong J, Kennan RP, Fulbright RK, Gore JC. Quantification of intravascular and extravascular contributions to BOLD effects induced by alteration in oxygenation or intravascular contrast agents. *Magn Reson Med*. 1998 Oct 1;40(4):526–36.

15. Sweeney PW, Walker-Samuel S, Shipley RJ. Insights into cerebral haemodynamics and oxygenation utilising in vivo mural cell imaging and mathematical modelling. *Sci Rep*. 2018 Jan 22;8(1):1373.
16. Lu H, Golay X, Pekar JJ, Zijl PCM van. Functional magnetic resonance imaging based on changes in vascular space occupancy. *Magn Reson Med*. 2003 Aug 1;50(2):263–74.
17. Hernández-Torres E, Kassner N, Forkert ND, Wei L, Wiggermann V, Daemen M, et al. Anisotropic cerebral vascular architecture causes orientation dependency in cerebral blood flow and volume measured with dynamic susceptibility contrast magnetic resonance imaging. *J Cereb Blood Flow Metab*. 2017 Mar 1;37(3):1108–19.
18. Boxerman JL, Bandettini PA, Kwong KK, Baker JR, Davis TL, Rosen BR, et al. The intravascular contribution to fmri signal change: monte carlo modeling and diffusion-weighted studies in vivo. *Magn Reson Med*. 1995 Jul 1;34(1):4–10.
19. Markuerkiaga I, Barth M, Norris DG. A cortical vascular model for examining the specificity of the laminar BOLD signal. *NeuroImage*. 2016 May 15;132:491–8.
20. Levin JM, Frederick B deB, Ross MH, Fox JF, von Rosenberg HL, Kaufman MJ, et al. Influence of baseline hematocrit and hemodilution on BOLD fMRI activation. *Magn Reson Imaging*. 2001 Oct 1;19(8):1055–62.
21. Gould IG, Linninger AA. Hematocrit Distribution and Tissue Oxygenation in Large Microcirculatory Networks. *Microcirculation*. 2015 Jan 1;22(1):1–18.
22. Gould IG, Tsai P, Kleinfeld D, Linninger A. The capillary bed offers the largest hemodynamic resistance to the cortical blood supply. *J Cereb Blood Flow Metab*. 2017 Jan 1;37(1):52–68.
23. Gagnon L, Smith AF, Boas DA, Devor A, Secomb TW, Sakadžić S. Modeling of Cerebral Oxygen Transport Based on In vivo Microscopic Imaging of Microvascular Network Structure, Blood Flow, and Oxygenation. *Front Comput Neurosci*. 2016;10.
24. Gagnon L, Sakadžić S, Lesage F, Mandeville ET, Fang Q, Yaseen MA, et al. Multimodal reconstruction of microvascular-flow distributions using combined two-photon microscopy and Doppler optical coherence tomography. *Neurophotonics*. 2015 Mar;2(1):015008.
25. Fang Q, Sakadžić S, Ruvinskaya L, Devor A, Dale AM, Boas DA. Oxygen Advection and Diffusion in a Three Dimensional Vascular Anatomical Network. *Opt Express*. 2008 Oct 27;16(22):17530–41.
26. Blinder P, Tsai PS, Kaufhold JP, Knutsen PM, Suhl H, Kleinfeld D. The cortical angiome: an interconnected vascular network with noncolumnar patterns of blood flow. *Nat Neurosci*. 2013 Jul;16(7):889–97.
27. Hartung G, Vesel C, Morley R, Alaraj A, Sled J, Kleinfeld D, et al. Simulations of blood as a suspension predicts a depth dependent hematocrit in the circulation throughout the cerebral cortex. *PLOS Comput Biol*. 2018 Nov 19;14(11):e1006549.
28. Ghanavati S, Yu LX, Lerch JP, Sled JG. A perfusion procedure for imaging of the mouse cerebral vasculature by X-ray micro-CT. *J Neurosci Methods*. 2014 Jan 15;221(Supplement C):70–7.
29. Ghaffari M, Tangen K, Alaraj A, Du X, Charbel FT, Linninger AA. Large-scale subject-specific cerebral arterial tree modeling using automated parametric mesh generation for blood flow simulation. *Comput Biol Med*. 2017 Dec 1;91:353–65.

30. Holter KE, Kehlet B, Devor A, Sejnowski TJ, Dale AM, Omholt SW, et al. Interstitial solute transport in 3D reconstructed neuropil occurs by diffusion rather than bulk flow. *Proc Natl Acad Sci*. 2017 Sep 12;114(37):9894–9.
31. Hsu R, Secomb TW. A Green's function method for analysis of oxygen delivery to tissue by microvascular networks. *Math Biosci*. 1989 Sep 1;96(1):61–78.
32. D'Angelo C. Finite element approximation of elliptic problems with Dirac measure terms in weighted spaces: applications to one-and three-dimensional coupled problems. *SIAM J Numer Anal*. 2012;50(1):194–215.
33. D'ANGELO C, Quarteroni A. On the coupling of 1d and 3d diffusion-reaction equations: application to tissue perfusion problems. *Math Models Methods Appl Sci*. 2008;18(08):1481–1504.
34. Ghaffari M, Hsu C-Y, Linninger AA. Automatic reconstruction and generation of structured hexahedral mesh for non-planar bifurcations in vascular networks. In: *Computer Aided Chemical Engineering*. Elsevier; 2015. p. 635–640.
35. Ghaffari M, Alaraj A, Du X, Zhou XJ, Charbel FT, Linninger AA. Quantification of near-wall hemodynamic risk factors in large-scale cerebral arterial trees. *Int J Numer Methods Biomed Eng*. 2018;34(7):e2987.
36. Peyrounette M, Davit Y, Quintard M, Lorthois S. Multiscale modelling of blood flow in cerebral microcirculation: Details at capillary scale control accuracy at the level of the cortex. *PLOS ONE*. 2018 Jan 11;13(1):e0189474.
37. El-Bouri WK, Payne SJ. Multi-scale homogenization of blood flow in 3-dimensional human cerebral microvascular networks. *J Theor Biol*. 2015 Sep 7;380:40–7.
38. Li B, Esipova TV, Sencan I, Kılıç K, Fu B, Desjardins M, et al. More homogeneous capillary flow and oxygenation in deeper cortical layers correlate with increased oxygen extraction. *Elife*. 2019;8.
39. Schmid F, Tsai PS, Kleinfeld D, Jenny P, Weber B. Depth-dependent flow and pressure characteristics in cortical microvascular networks. *PLOS Comput Biol*. 2017 Feb 14;13(2):e1005392.
40. Lorthois S, Cassot F, Lauwers F. Simulation study of brain blood flow regulation by intra-cortical arterioles in an anatomically accurate large human vascular network: Part I: methodology and baseline flow. *NeuroImage*. 2011 Jan 15;54(2):1031–42.
41. Blinder P, Tsai PS, Kaufhold JP, Knutsen PM, Suhl H, Kleinfeld D. The cortical angiome: an interconnected vascular network with noncolumnar patterns of blood flow. *Nat Neurosci*. 2013 Jul;16(7):889–97.
42. Cassot F, Lauwers F, Fouard C, Prohaska S, Lauwers-Cances V. A Novel Three-Dimensional Computer-Assisted Method for a Quantitative Study of Microvascular Networks of the Human Cerebral Cortex. *Microcirculation*. 2006;13(1):1–18.
43. Lauwers F, Cassot F, Lauwers-Cances V, Puwanarajah P, Duvernoy H. Morphometry of the human cerebral cortex microcirculation: General characteristics and space-related profiles. *NeuroImage*. 2008 Feb 1;39(3):936–48.
44. Lorthois S, Cassot F, Lauwers F. Simulation study of brain blood flow regulation by intra-cortical arterioles in an anatomically accurate large human vascular network. Part II: Flow variations induced by global or localized modifications of arteriolar diameters. *NeuroImage*. 2011 Feb 14;54(4):2840–53.

45. Tsai PS, Kaufhold JP, Blinder P, Friedman B, Drew PJ, Karten HJ, et al. Correlations of Neuronal and Microvascular Densities in Murine Cortex Revealed by Direct Counting and Colocalization of Nuclei and Vessels. *J Neurosci*. 2009 Nov 18;29(46):14553–70.
46. Kidoguchi K, Tamaki M, Mizobe T, Koyama J, Kondoh T, Kohmura E, et al. In Vivo X-Ray Angiography in the Mouse Brain Using Synchrotron Radiation. *Stroke*. 2006 Jul 1;37(7):1856–61.
47. Dyer EL, Roncal WG, Prasad JA, Fernandes HL, Gürsoy D, Andrade VD, et al. Quantifying Mesoscale Neuroanatomy Using X-Ray Microtomography. *eNeuro*. 2017 Sep 1;4(5):ENEURO.0195-17.2017.
48. Bicer T, Gursoy D, Kettimuthu R, Foster IT, Ren B, De Andrede V, et al. Real-Time Data Analysis and Autonomous Steering of Synchrotron Light Source Experiments. In: 2017 IEEE 13th International Conference on e-Science (e-Science). Auckland: IEEE; 2017. p. 59–68.
49. Zagzoule M, Marc-Vergnes J-P. A global mathematical model of the cerebral circulation in man. *J Biomech*. 1986 Jan 1;19(12):1015–22.
50. Cebal JR, Castro MA, Soto O, Löhner R, Alperin N. Blood-flow models of the circle of Willis from magnetic resonance data. *J Eng Math*. 2003 Dec;47(3/4):369–86.
51. Blanco PJ, Pivello MR, Urquiza SA, Feijóo RA. On the potentialities of 3D–1D coupled models in hemodynamics simulations. *J Biomech*. 2009 May 11;42(7):919–30.
52. Blanco PJ, Watanabe SM, Dari EA, Passos MARF, Feijóo RA. Blood flow distribution in an anatomically detailed arterial network model: criteria and algorithms. *Biomech Model Mechanobiol*. 2014 Nov 1;13(6):1303–30.
53. Schneider M, Hirsch S, Weber B, Székely G. Physiologically Based Construction of Optimized 3-D Arterial Tree Models. In: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2011. Springer, Berlin, Heidelberg; 2011. p. 404–11.
54. Su S-W, Catherall M, Payne S. The Influence of Network Structure on the Transport of Blood in the Human Cerebral Microvasculature. *Microcirculation*. 2012;19(2):175–87.
55. Bui AV, Manasseh R, Liffman K, Šutalo ID. Development of optimized vascular fractal tree models using level set distance function. *Med Eng Phys*. 2010 Sep 1;32(7):790–4.
56. Karch R, Neumann F, Neumann M, Schreiner W. A three-dimensional model for arterial tree representation, generated by constrained constructive optimization. *Comput Biol Med*. 1999 Jan;29(1):19–38.
57. Karch R, Neumann F, Neumann M, Schreiner W. Staged Growth of Optimized Arterial Model Trees. *Ann Biomed Eng*. 2000 May 1;28(5):495–511.
58. Karch R, Neumann F, Podesser BK, Neumann M, Szawłowski P, Schreiner W. Fractal Properties of Perfusion Heterogeneity in Optimized Arterial Trees: A Model Study. *J Gen Physiol*. 2003 Sep 1;122(3):307–22.
59. Schreiner W, Neumann F, Karch R, Neumann M, Roedler SM, End A. Shear Stress Distribution in Arterial Tree Models, Generated by Constrained Constructive Optimization. *J Theor Biol*. 1999 May 7;198(1):27–45.
60. Schreiner W, Neumann F, Neumann M, Karch R, End A, Roedler SM. Limited Bifurcation Asymmetry in Coronary Arterial Tree Models Generated by Constrained Constructive Optimization. *J Gen Physiol*. 1997 Feb 1;109(2):129–40.

61. Schreiner W, Karch R, Neumann M, Neumann F, Roedler SM, Heinze G. Heterogeneous Perfusion is a Consequence of Uniform Shear Stress in Optimized Arterial Tree Models. *J Theor Biol.* 2003;3(220):285–301.
62. Linninger AA, Gould IG, Marinnan T, Hsu C-Y, Chojecki M, Alaraj A. Cerebral Microcirculation and Oxygen Tension in the Human Secondary Cortex. *Ann Biomed Eng.* 2013 Nov 1;41(11):2264–84.
63. Hartung G, Vesel C, Morley R, Alaraj A, Sled J, Kleinfeld D, et al. Simulations of blood as a suspension predicts a depth dependent hematocrit in the circulation throughout the cerebral cortex. *PLOS Comput Biol.* (In Print).
64. Tellegen BDH. A general network theorem, with applications. *Philips Res Rept.* 1952;7:259–69.
65. Wartmann M. Process Network Optimality. Carnegie Mellon University; 2010.
66. Cassot F, Lauwers F, Lorthois S, Puwanarajah P, Cances-Lauwers V, Duvernoy H. Branching patterns for arterioles and venules of the human cerebral cortex. *Brain Res.* 2010 Feb 8;1313:62–78.
67. Murray CD. The Physiological Principle of Minimum Work: I. The Vascular System and the Cost of Blood Volume. *Proc Natl Acad Sci.* 1926 Mar 1;12(3):207–14.
68. Hsu C-Y, Schneller B, Alaraj A, Flannery M, Zhou XJ, Linninger A. Automatic recognition of subject-specific cerebrovascular trees. *Magn Reson Med.* 2016 Dec 27;1(77):398–410.
69. Hsu C-Y, Ghaffari M, Alaraj A, Flannery M, Zhou XJ, Linninger A. Gap-free segmentation of vascular networks with automatic image processing pipeline. *Comput Biol Med.* 2017 Mar 1;82:29–39.
70. Ghanavati S, Yu LX, Lerch JP, Sled JG. A perfusion procedure for imaging of the mouse cerebral vasculature by X-ray micro-CT. *J Neurosci Methods.* 2014 Jan 15;221(Supplement C):70–7.
71. Xiong B, Li A, Lou Y, Chen S, Long B, Peng J, et al. Precise Cerebral Vascular Atlas in Stereotaxic Coordinates of Whole Mouse Brain. *Front Neuroanat.* 2017 Dec 19;11.
72. Gamma E, Helm R, Johnson R, Vlissides J. Design Patterns: Elements of Reusable Object-Oriented Software. Pearson Education; 1994. 457 p.
73. Schmid F, Barrett MJP, Jenny P, Weber B. Vascular density and distribution in neocortex. *NeuroImage.* 2017 Jun 29;
74. Kidoguchi K, Tamaki M, Mizobe T, Koyama J, Kondoh T, Kohmura E, et al. In Vivo X-Ray Angiography in the Mouse Brain Using Synchrotron Radiation. *Stroke.* 2006 Jul 1;37(7):1856–61.
75. Ma Y, Hof PR, Grant SC, Blackband SJ, Bennett R, Slate L, et al. A three-dimensional digital atlas database of the adult C57BL/6J mouse brain by magnetic resonance microscopy. *Neuroscience.* 2005 Jan 1;135(4):1203–15.
76. Badea A, Ali-Sharief AA, Johnson GA. Morphometric analysis of the C57BL/6J mouse brain. *NeuroImage.* 2007 Sep 1;37(3):683–93.
77. Kovačević N, Henderson JT, Chan E, Lifshitz N, Bishop J, Evans AC, et al. A Three-dimensional MRI Atlas of the Mouse Brain with Estimates of the Average and Variability. *Cereb Cortex.* 2005 May 1;15(5):639–45.

78. Diem AK, Tan M, Bressloff NW, Hawkes C, Morris AWJ, Weller RO, et al. A Simulation Model of Periarterial Clearance of Amyloid- β from the Brain. *Front Aging Neurosci.* 2016;8.
79. Natt O, Watanabe T, Boretius S, Radulovic J, Frahm J, Michaelis T. High-resolution 3D MRI of mouse brain reveals small cerebral structures in vivo. *J Neurosci Methods.* 2002 Oct 30;120(2):203–9.
80. Clavaguera F, Bolmont T, Crowther RA, Abramowski D, Frank S, Probst A, et al. Transmission and spreading of tauopathy in transgenic mouse brain. *Nat Cell Biol.* 2009 Jul;11(7):909–13.
81. Lefort S, Tómm C, Floyd Sarria J-C, Petersen CCH. The Excitatory Neuronal Network of the C2 Barrel Column in Mouse Primary Somatosensory Cortex. *Neuron.* 2009 Jan 29;61(2):301–16.
82. DeFelipe J, Alonso-Nanclares L, I. Arellano J. Microstructure of the neocortex: comparative aspects. *J Neurocytol.* 2002;31(3–5):299–316.
83. Gould IG, Tsai P, Kleinfeld D, Linninger A. The capillary bed offers the largest hemodynamic resistance to the cortical blood supply. *J Cereb Blood Flow Metab.* 2017 Jan 1;37(1):52–68.
84. Gould IG, Linninger AA. Hematocrit Distribution and Tissue Oxygenation in Large Microcirculatory Networks. *Microcirculation.* 2015 Jan 1;22(1):1–18.
85. Gagnon L, Sakadžić S, Lesage F, Musacchia JJ, Lefebvre J, Fang Q, et al. Quantifying the Microvascular Origin of BOLD-fMRI from First Principles with Two-Photon Microscopy and an Oxygen-Sensitive Nanoprobe. *J Neurosci.* 2015 Feb 25;35(8):3663–75.
86. Coumey C, Berg M, Ho H, Hunter P. Computational Simulation of Blood Flow and Drug Transportation in a Large Vasculature. In: Joldes GR, Doyle B, Wittek A, Nielsen PMF, Miller K, editors. *Computational Biomechanics for Medicine.* Springer International Publishing; 2016. p. 133–42.
87. Linninger AA, Tangen K, Hsu C-Y, Frim D. Cerebrospinal Fluid Mechanics and Its Coupling to Cerebrovascular Dynamics. *Annu Rev Fluid Mech.* 2016;48(1):219–57.
88. Linninger AA, Xu C, Tangen K, Hartung G. Starling forces drive intracranial water exchange during normal and pathological states. *Croat Med J.* 2017 Dec;58(6):384–94.
89. Gagnon L, Sakadžić S, Lesage F, Musacchia JJ, Lefebvre J, Fang Q, et al. Quantifying the Microvascular Origin of BOLD-fMRI from First Principles with Two-Photon Microscopy and an Oxygen-Sensitive Nanoprobe. *J Neurosci.* 2015 Feb 25;35(8):3663–75.
90. Gould IG, Tsai P, Kleinfeld D, Linninger A. The capillary bed offers the largest hemodynamic resistance to the cortical blood supply. *J Cereb Blood Flow Metab.* 2017 Jan 1;37(1):52–68.
91. Iadecola C. Neurovascular regulation in the normal brain and in Alzheimer’s disease. *Nat Rev Neurosci.* 2004 May;5(5):347–60.
92. Fox PT, Raichle ME. Focal physiological uncoupling of cerebral blood flow and oxidative metabolism during somatosensory stimulation in human subjects. *Proc Natl Acad Sci.* 1986 Feb 1;83(4):1140–4.
93. Kim JH, Ress D. Arterial impulse model for the BOLD response to brief neural activation. *NeuroImage.* 2016 Jan 1;124:394–408.
94. Uludağ K, Blinder P. Linking brain vascular physiology to hemodynamic response in ultra-high field MRI. *NeuroImage.* 2017 Feb 22;

95. Lorthois S, Cassot F, Lauwers F. Simulation study of brain blood flow regulation by intra-cortical arterioles in an anatomically accurate large human vascular network. Part II: Flow variations induced by global or localized modifications of arteriolar diameters. *NeuroImage*. 2011 Feb 14;54(4):2840–53.
96. Drake-Holland AJ, Laird JD, Noble MI, Spaan JA, Vergroesen I. Oxygen and coronary vascular resistance during autoregulation and metabolic vasodilation in the dog. *J Physiol*. 1984 Mar 1;348(1):285–99.
97. Hudetz AG. Blood Flow in the Cerebral Capillary Network: A Review Emphasizing Observations with Intravital Microscopy. *Microcirculation*. 1997 Jan 1;4(2):233–52.
98. Johnson PC. Autoregulation of blood flow. *Circ Res*. 1986 Nov 1;59(5):483–95.
99. Strandgaard S. Autoregulation of cerebral blood flow in hypertensive patients. The modifying influence of prolonged antihypertensive treatment on the tolerance to acute, drug-induced hypotension. *Circulation*. 1976 Apr 1;53(4):720–7.
100. Lassen NA. Cerebral Blood Flow and Oxygen Consumption in Man. *Physiol Rev*. 1959 Apr 1;39(2):183–238.
101. Strandgaard S, Paulson OB. Cerebral autoregulation. *Stroke*. 1984 May 1;15(3):413–6.
102. Paulson OB, Strandgaard S, Edvinsson L. Cerebral autoregulation. *Cerebrovasc Brain Metab Rev*. 1990;2(2):161–92.
103. Panerai RB. Assessment of cerebral pressure autoregulation in humans - a review of measurement methods. *Physiol Meas*. 1998;19(3):305.
104. Olufsen MS, Nadim A, Lipsitz LA. Dynamics of cerebral blood flow regulation explained using a lumped parameter model. *Am J Physiol-Regul Integr Comp Physiol*. 2002 Feb;282(2):R611–22.
105. Olufsen M, Tran H, Ottesen J. Modeling Cerebral Blood Flow Control During Posture Change from Sitting to Standing. *Cardiovasc Eng*. 2004 Mar;4(1):47–58.
106. Lu K, Clark JW, Ghorbel FH, Robertson CS, Ware DL, Zwischenberger JB, et al. Cerebral autoregulation and gas exchange studied using a human cardiopulmonary model. In *IEEE*; 2003. p. 395–7.
107. Sakadžić S, Mandeville ET, Gagnon L, Musacchia JJ, Yaseen MA, Yucel MA, et al. Large arteriolar component of oxygen delivery implies a safe margin of oxygen supply to cerebral tissue. *Nat Commun*. 2014 Dec 8;5:5734.
108. Schmid F, Tsai PS, Kleinfeld D, Jenny P, Weber B. Depth-dependent flow and pressure characteristics in cortical microvascular networks. *PLOS Comput Biol*. 2017 Feb 14;13(2):e1005392.
109. Lorthois S, Cassot F, Lauwers F. Simulation study of brain blood flow regulation by intra-cortical arterioles in an anatomically accurate large human vascular network: Part I: methodology and baseline flow. *NeuroImage*. 2011 Jan 15;54(2):1031–42.
110. Payne SJ, Lucas C. Oxygen delivery from the cerebral microvasculature to tissue is governed by a single time constant of approximately 6 seconds. *Microcirculation*. 2018 Feb 1;25(2):n/a–n/a.
111. Gagnon L, Sakadžić S, Lesage F, Mandeville ET, Fang Q, Yaseen MA, et al. Multimodal reconstruction of microvascular-flow distributions using combined two-photon microscopy and Doppler optical coherence tomography. *Neurophotonics*. 2015 Mar;2(1):015008.

112. Boas DA, Jones SR, Devor A, Huppert TJ, Dale AM. A vascular anatomical network model of the spatio-temporal response to brain activation. *NeuroImage*. 2008 Apr 15;40(3):1116–29.
113. Lorthois S, Cassot F. Fractal analysis of vascular networks: Insights from morphogenesis. *J Theor Biol*. 2010 Feb 21;262(4):614–33.
114. Obrist D, Weber B, Buck A, Jenny P. Red blood cell distribution in simplified capillary networks. *Philos Trans R Soc Lond Math Phys Eng Sci*. 2010 Jun 28;368(1921):2897–918.
115. Gould IG, Linninger AA. Hematocrit Distribution and Tissue Oxygenation in Large Microcirculatory Networks. *Microcirculation*. 2015 Jan 1;22(1):1–18.
116. Linninger AA, Gould IG, Marinnan T, Hsu C-Y, Chojecki M, Alaraj A. Cerebral Microcirculation and Oxygen Tension in the Human Secondary Cortex. *Ann Biomed Eng*. 2013 Nov 1;41(11):2264–84.
117. Ghanavati S, Yu LX, Lerch JP, Sled JG. A perfusion procedure for imaging of the mouse cerebral vasculature by X-ray micro-CT. *J Neurosci Methods*. 2014 Jan 15;221(Supplement C):70–7.
118. Chugh BP, Lerch JP, Yu LX, Pienkowski M, Harrison RV, Henkelman RM, et al. Measurement of cerebral blood volume in mouse brain regions using micro-computed tomography. *NeuroImage*. 2009 Oct 1;47(4):1312–8.
119. Marxen M, Thornton MM, Chiarot CB, Klement G, Koprivnikar J, Sled JG, et al. MicroCT scanner performance and considerations for vascular specimen imaging. *Med Phys*. 2004 Feb;31(2):305–13.
120. Ghanavati S, Lerch JP, Sled JG. Automatic anatomical labeling of the complete cerebral vasculature in mouse models. *NeuroImage*. 2014 Jul 15;95:117–28.
121. Ghaffari M, Hsu C-Y, Linninger AA. Automatic Reconstruction and Generation of Structured Hexahedral Mesh for Non-planar Bifurcations in Vascular Networks. In: Gernaey KV, Huusom JK, Gani R, editors. *Computer Aided Chemical Engineering*. Elsevier; 2015. p. 635–40.
122. Hsu C-Y, Ghaffari M, Alaraj A, Flannery M, Zhou XJ, Linninger A. Gap-free segmentation of vascular networks with automatic image processing pipeline. *Comput Biol Med*. 2017 Mar 1;82:29–39.
123. Hsu C-Y, Schneller B, Alaraj A, Flannery M, Zhou XJ, Linninger A. Automatic recognition of subject-specific cerebrovascular trees. *Magn Reson Med*. 2016 Dec 27;1(77):398–410.
124. Tsai PS, Kaufhold JP, Blinder P, Friedman B, Drew PJ, Karten HJ, et al. Correlations of Neuronal and Microvascular Densities in Murine Cortex Revealed by Direct Counting and Colocalization of Nuclei and Vessels. *J Neurosci*. 2009 Nov 18;29(46):14553–70.
125. Shih AY, Driscoll JD, Drew PJ, Nishimura N, Schaffer CB, Kleinfeld D. Two-Photon Microscopy as a Tool to Study Blood Flow and Neurovascular Coupling in the Rodent Brain. *J Cereb Blood Flow Metab*. 2012 Jul 1;32(7):1277–309.
126. Kaufhold JP, Tsai PS, Blinder P, Kleinfeld D. Vectorization of optically sectioned brain microvasculature: Learning aids completion of vascular graphs by connecting gaps and deleting open-ended segments. *Med Image Anal*. 2012 Aug;16(6):1241–58.
127. Pries AR, Neuhaus D, Gaehtgens P. Blood viscosity in tube flow: dependence on diameter and hematocrit. *Am J Physiol*. 1992 Dec;263(6 Pt 2):H1770-1778.

128. Yang J, Yoo SS, Lee T-R. Effect of fractional blood flow on plasma skimming in the microvasculature. *Phys Rev E*. 2017 Apr 25;95(4):040401.
129. Yang J, Pak YE, Lee T-R. Predicting bifurcation angle effect on blood flow in the microvasculature. *Microvasc Res*. 2016 Nov 1;108:22–8.
130. Lee T-R, Yoo SS, Yang J. Generalized plasma skimming model for cells and drug carriers in the microvasculature. *Biomech Model Mechanobiol*. 2017 Apr 1;16(2):497–507.
131. Tellegen BDH. A General Network Theorem With Applications. *Phillips Res Rep*. 1952;7:259–69.
132. Miller GF, Penke L. The evolution of human intelligence and the coefficient of additive genetic variance in human brain size. *Intelligence*. 2007 Mar 1;35(2):97–114.
133. Pardridge WM. Blood–brain barrier delivery. *Drug Discov Today*. 2007 Jan 1;12(1):54–61.
134. Abbott NJ, Patabendige AAK, Dolman DEM, Yusof SR, Begley DJ. Structure and function of the blood–brain barrier. *Neurobiol Dis*. 2010 Jan 1;37(1):13–25.
135. Hartung G, Linninger A. The image-based cerebrovascular growth algorithm. 2019.
136. Hartung G, Alaraj A, Linninger A. Chapter 21 - Walk-In Brain: Virtual Reality Environment for Immersive Exploration and Simulation of Brain Metabolism and Function. In: Martín M, Eden MR, Chemmangattuvalappil NG, editors. *Computer Aided Chemical Engineering*. Elsevier; 2016. p. 649–58. (Tools For Chemical Product Design; vol. 39).
137. Desai B, Hobbs J, Hartung G, Xu G, Gokaslan ZL, Linninger A, et al. Image-guidance technology and the surgical resection of spinal column tumors. *J Neurooncol*. 2017 Feb 1;131(3):425–35.
138. Zweifach BW, Lipowsky HH. Quantitative studies of microcirculatory structure and function. III. Microvascular hemodynamics of cat mesentery and rabbit omentum. *Circ Res*. 1977 Sep 1;41(3):380–90.
139. Oshio K, Watanabe H, Song Y, Verkman AS, Manley GT. Reduced cerebrospinal fluid production and intracranial pressure in mice lacking choroid plexus water channel Aquaporin-1. *FASEB J*. 2004 Nov 8;19(1):76–8.
140. Morimoto S, Cassell MD, Beltz TG, Johnson AK, Davisson RL, Sigmund CD. Elevated Blood Pressure in Transgenic Mice With Brain-Specific Expression of Human Angiotensinogen Driven by the Glial Fibrillary Acidic Protein Promoter. *Circ Res*. 2001 Aug 17;89(4):365–72.
141. Maeda K, Mies G, Oláh L, Hossmann K-A. Quantitative Measurement of Local Cerebral Blood Flow in the Anesthetized Mouse Using Intraperitoneal [¹⁴C]Iodoantipyrine Injection and Final Arterial Heart Blood Sampling. *J Cereb Blood Flow Metab*. 2000 Jan 1;20(1):10–4.
142. Gertz K, Priller J, Kronenberg G, Fink KB, Winter B, Schröck H, et al. Physical Activity Improves Long-Term Stroke Outcome via Endothelial Nitric Oxide Synthase–Dependent Augmentation of Neovascularization and Cerebral Blood Flow. *Circ Res*. 2006 Nov 10;99(10):1132–40.
143. DeFelipe J, Alonso-Nanclares L, I. Arellano J. Microstructure of the neocortex: comparative aspects. *J Neurocytol*. 2002;31(3–5):299–316.
144. Lefort S, Tómm C, Floyd Sarria J-C, Petersen CCH. The Excitatory Neuronal Network of the C2 Barrel Column in Mouse Primary Somatosensory Cortex. *Neuron*. 2009 Jan 29;61(2):301–16.

145. Karch R, Neumann F, Neumann M, Schreiner W. A three-dimensional model for arterial tree representation, generated by constrained constructive optimization. *Comput Biol Med.* 1999 Jan;29(1):19–38.
146. Dorr A, Sled JG, Kabani N. Three-dimensional cerebral vasculature of the CBA mouse brain: A magnetic resonance imaging and micro computed tomography study. *NeuroImage.* 2007 May;35(4):1409–23.
147. Xiong B, Li A, Lou Y, Chen S, Long B, Peng J, et al. Precise Cerebral Vascular Atlas in Stereotaxic Coordinates of Whole Mouse Brain. *Front Neuroanat.* 2017 Dec 19;11.
148. Nishimura N, Schaffer CB, Friedman B, Lyden PD, Kleinfeld D. Penetrating arterioles are a bottleneck in the perfusion of neocortex. *Proc Natl Acad Sci.* 2007 Jan 2;104(1):365–70.
149. Diem AK, Tan M, Bressloff NW, Hawkes C, Morris AWJ, Weller RO, et al. A Simulation Model of Periarterial Clearance of Amyloid- β from the Brain. *Front Aging Neurosci.* 2016;8.
150. Clavaguera F, Bolmont T, Crowther RA, Abramowski D, Frank S, Probst A, et al. Transmission and spreading of tauopathy in transgenic mouse brain. *Nat Cell Biol.* 2009 Jul;11(7):909–13.
151. Natt O, Watanabe T, Boretius S, Radulovic J, Frahm J, Michaelis T. High-resolution 3D MRI of mouse brain reveals small cerebral structures in vivo. *J Neurosci Methods.* 2002 Oct 30;120(2):203–9.
152. Krogh A. *The Anatomy and Physiology of Capillaries.* Yale University Press; 1922. 304 p.
153. Dintenfass L. *Blood Viscosity.* Springer Science & Business Media; 1985. 502 p.
154. Klitzman B, Duling B. Microvascular hematocrit and red cell in resting and contracting striated muscle. Vol. 237. 1979. H481 p.
155. Lipowsky HH, Usami S, Chien S. In vivo measurements of “apparent viscosity” and microvessel hematocrit in the mesentery of the cat. *Microvasc Res.* 1980 May 1;19(3):297–319.
156. Pries AR, Secomb TW, Gessner T, Sperandio MB, Gross JF, Gaehtgens P. Resistance to blood flow in microvessels in vivo. *Circ Res.* 1994 Nov 1;75(5):904–15.
157. Liu R, Li Z, Kleinfeld D. Adaptive optics with direct wavefront sensing from microvessels enables two-photon imaging of deep cortical layers. *Print.*
158. Pries AR, Secomb TW. Microvascular blood viscosity in vivo and the endothelial surface layer. *Am J Physiol - Heart Circ Physiol.* 2005 Dec 1;289(6):H2657–64.
159. Chebbi R. Dynamics of blood flow: modeling of the Fåhræus–Lindqvist effect. *J Biol Phys.* 2015 Jun 1;41(3):313–26.
160. Pries AR, Ley K, Claassen M, Gaehtgens P. Red cell distribution at microvascular bifurcations. *Microvasc Res.* 1989 Jul 1;38(1):81–101.
161. Rasmussen PM, Secomb TW, Pries AR. Modeling the hematocrit distribution in microcirculatory networks: A quantitative evaluation of a phase separation model. *Microcirculation.* 2018 Mar 10;n/a-n/a.
162. Tang Z, Lee JH. Effects of Different Hematocrit Levels on Glucose Measurements With Handheld Meters for Point-of-Care Testing. *Arch Pathol Lab Med.* 2000;124:6.

163. Gagnon L, Smith AF, Boas DA, Devor A, Secomb TW, Sakadžić S. Modeling of Cerebral Oxygen Transport Based on In vivo Microscopic Imaging of Microvascular Network Structure, Blood Flow, and Oxygenation. *Front Comput Neurosci.* 2016;10.
164. Secomb TW, Hsu R, Park EYH, Dewhirst MW. Green's Function Methods for Analysis of Oxygen Delivery to Tissue by Microvascular Networks. *Ann Biomed Eng.* 2004 Nov 1;32(11):1519–29.
165. Gjerde IG, Kumar K, Nordbotten JM, Wohlmuth B. Splitting method for elliptic equations with line sources. *ArXiv Prepr ArXiv181012979.* 2018;
166. Mintun MA, Lundstrom BN, Snyder AZ, Vlassenko AG, Shulman GL, Raichle ME. Blood flow and oxygen delivery to human brain during functional activity: theoretical modeling and experimental data. *PNAS.* 2001;98(12):6859–64.
167. Truskey G, Yuan F, Katz D. *Transport Phenomena in Biological Systems.* Pearson Prentice Hall;
168. Balay S, Abhyankar S, Adams MF, Brown J, Brune P, Buschelman K, et al. PETSc Web page [Internet]. 2019.
169. Balay S, Abhyankar S, Adams MF, Brown J, Brune P, Buschelman K, et al. PETSc Users Manual [Internet]. Argonne National Laboratory; 2019. Report No.: ANL-95/11-Revision 3.11.
170. Linninger A, Hartung G, Badr S, Morley R. Mathematical synthesis of the cortical circulation for the whole mouse brain-part I: theory and image integration. *Comput Biol Med.* (In Print).
171. Faber JE, Zhang H, Lassance-Soares RM, Prabhakar P, Najafi AH, Burnett MS, et al. Aging causes collateral rarefaction and increased severity of ischemic injury in multiple tissues. *Arterioscler Thromb Vasc Biol.* 2011;31(8):1748–1756.
172. Murugesan N, Demarest TG, Madri JA, Pachter JS. Brain regional angiogenic potential at the neurovascular unit during normal aging. *Neurobiol Aging.* 2012;33(5):1004–e1.
173. Casey MA, Feldman ML. Aging in the rat medial nucleus of the trapezoid body. III. Alterations in capillaries. *Neurobiol Aging.* 1985;6(1):39–46.
174. Wilkinson J, Hopewell J, Reinhold H. A quantitative study of age-related changes in the vascular architecture of the rat cerebral cortex. *Neuropathol Appl Neurobiol.* 1981;7(6):451–462.
175. Hsu C-Y, Linninger A. Vesselness Filter [Internet]. Laboratory for Product and Process Design, UIC; 2017.
176. Wang Z, Chi Y, Huang W, Venkatesh SK, Tian Q, Oo T, et al. Comparisons of centerline extraction methods for liver blood vessels in imageJ and 3D slicer. *APSIPA ASC.* 2010;276–279.
177. Lee TC, Kashyap RL, Chu CN. Building Skeleton Models via 3-D Medial Surface Axis Thinning Algorithms. *CVGIP Graph Models Image Process.* 1994 Nov 1;56(6):462–78.
178. Kollmannsberger P, Kerschnitzki M, Repp F, Wagermaier W, Weinkamer R, Peter Fratzl. The small world of osteocytes: connectomics of the lacuno-canalicular network in bone. *New J Phys.* 2017;19(7):073019.
179. Ghanavati S, Lerch JP, Sled JG. Automatic anatomical labeling of the complete cerebral vasculature in mouse models. *NeuroImage.* 2014 Jul 15;95:117–28.
180. Hartung G, Alaraj A, Linninger A. Chapter 21 - Walk-In Brain: Virtual Reality Environment for Immersive Exploration and Simulation of Brain Metabolism and Function. In: Martín M, Eden MR,

- Chemangattuvalappil NG, editors. Computer Aided Chemical Engineering. Elsevier; 2016. p. 649–58. (Tools For Chemical Product Design; vol. 39).
181. Fischl B, et al. FreeSurfer. Boston, MA; 2017.
 182. Yushkevich PA, Piven J, Cody Hazlett H, Gimpel Smith R, Ho S, Gee JC, et al. User-Guided 3D Active Contour Segmentation of Anatomical Structures: Significantly Improved Efficiency and Reliability. *Neuroimage*. 2006;31(3):1116–1128.
 183. Linninger A, Hartung GA, Liu BP, Mirkov S, Tangen K, Lukas RV, et al. Modeling the diffusion of D-2-hydroxyglutarate from IDH1 mutant gliomas in the central nervous system. *Neuro-Oncol*. 2018 Aug 2;20(9):1197–206.
 184. Politis A, Sweetman B, Linninger A. File formats for unstructured computational meshes. LPPD; 2008.
 185. Press WH, Teukolsky SA, Vetterling WT, Flannery BP. Numerical Recipes 3rd Edition: The Art of Scientific Computing. Cambridge University Press; 2007. 1195 p.
 186. Shrirang A, Brown J, Constantinescu E, Debojyoti G, Smith B, Zhang H. A modern scalable ODE/DAE solver library. *ArXiv Prepr*. 2018;
 187. Zenos M, Kulkarni K, Linninger A. Iterative methods for solving algebraic systems. LPPD; 2004.
 188. Ghaffari M, Linninger A. Eigen-analysis for the solution of linear dynamics systems. LPPD; 2017.
 189. Park C-S, Linninger A. Introduction to polynomial approximation – Lagrange polynomials. LPPD; 2017.
 190. Sweetman B, Linninger A. Hyperbolic (wave) and parabolic (diffusion) partial differential equations and their applications to vasculature dynamics. LPPD; 2008.
 191. Wartmann M. Process Network Optimality. Carnegie Mellon University; 2010.
 192. Bird R, Stewart W, Lightfoot E. Transport Phenomena (Second ed.). John Wiley & Sons; 2001.
 193. Vázquez BYS, Cabrales P, Tsai AG, Intaglietta M. Nonlinear cardiovascular regulation consequent to changes in blood viscosity. *Clin Hemorheol Microcirc*. 2011 Jan 1;49(1–4):29–36.
 194. Heuser G, Opitz R. A Couette viscometer for short time shearing of blood. *Biorheology*. 1980 Jan 1;17(1–2):17–24.
 195. Gaehtgens P. Flow of blood through narrow capillaries: Rheological mechanisms determining capillary hematocrit and apparent viscosity. *Biorheology*. 1980 Jan 1;17(1–2):183–9.
 196. Pries AR, Secomb TW, Gaehtgens P. Biophysical aspects of blood flow in the microvasculature. *Cardiovasc Res*. 1996 Oct 1;32(4):654–67.
 197. Pries AR, Secomb TW, Gessner T, Sperandio MB, Gross JF, Gaehtgens P. Resistance to blood flow in microvessels in vivo. *Circ Res*. 1994 Nov 1;75(5):904–15.
 198. Secomb TW, Pries AR. Blood viscosity in microvessels: Experiment and theory. *Comptes Rendus Phys*. 2013 Jun 1;14(6):470–8.

199. Pries AR, Secomb TW. Microvascular blood viscosity in vivo and the endothelial surface layer. *Am J Physiol - Heart Circ Physiol*. 2005 Dec 1;289(6):H2657–64.
200. Kiani MF, Hudetz AG. A semi-empirical model of apparent blood viscosity as a function of vessel diameter and discharge hematocrit. *Biorheology*. 1991 Jan 1;28(1–2):65–73.
201. Charm S, Kurland G. Blood flow and microcirculation. *Am Heart J*. 1974 Dec 1;88(6):243.
202. Pries AR, Ley K, Claassen M, Gaehtgens P. Red cell distribution at microvascular bifurcations. *Microvasc Res*. 1989 Jul 1;38(1):81–101.
203. Lipowsky HH, Usami S, Chien S. In vivo measurements of “apparent viscosity” and microvessel hematocrit in the mesentery of the cat. *Microvasc Res*. 1980 May 1;19(3):297–319.
204. Pries AR, Secomb TW, Gessner T, Sperandio MB, Gross JF, Gaehtgens P. Resistance to blood flow in microvessels in vivo. *Circ Res*. 1994 Nov 1;75(5):904–15.
205. Pries AR, Secomb TW, Gaehtgens P. Biophysical aspects of blood flow in the microvasculature. *Cardiovasc Res*. 1996 Oct 1;32(4):654–67.
206. Secomb TW, Pries AR. Blood viscosity in microvessels: Experiment and theory. *Comptes Rendus Phys*. 2013 Jun 1;14(6):470–8.
207. Pries AR, Secomb TW. Microvascular blood viscosity in vivo and the endothelial surface layer. *Am J Physiol - Heart Circ Physiol*. 2005 Dec 1;289(6):H2657–64.
208. Blood flow and microcirculation: By Stanley E. Charm, Sc.D., and George S. Kurland, M.D., New York, 1974, John Wiley & Sons, Inc., 243 pp. \$18.00. *Am Heart J*. 1974 Dec 1;88(6):815.
209. Haynes RH. Physical basis of the dependence of blood viscosity on tube radius. *Am J Physiol-Leg Content*. 1960 Jun 1;198(6):1193–200.
210. Whitmore R. A theory of blood flow in small vessels. *J Appl Physiol*. 1967;22(4):767–71.
211. Kiani MF, Hudetz AG. A semi-empirical model of apparent blood viscosity as a function of vessel diameter and discharge hematocrit. *Biorheology*. 1991 Jan 1;28(1–2):65–73.
212. Reinke W, Gaehtgens P, Johnson PC. Blood viscosity in small tubes: effect of shear rate, aggregation, and sedimentation. *Am J Physiol*. 1987 Sep;253(3 Pt 2):H540-7.
213. Bayliss LE. The axial drift of the red cells when blood flows in a narrow tube. *J Physiol*. 1959;149(3):593–613.
214. Krogh A. The progress of physiology. *Am J Physiol-Leg Content*. 1929;90(2):243–51.
215. Fourman J, Moffat DB. The effect of intra-arterial cushions on plasma skimming in small arteries. *J Physiol*. 1961;158(2):374–80.
216. Dellimore JW, Dunlop MJ, Canham PB. Ratio of cells and plasma in blood flowing past branches in small plastic channels. *Am J Physiol - Heart Circ Physiol*. 1983 May 1;244(5):H635–43.
217. Klitzman B, Johnson PC. Capillary network geometry and red cell distribution in hamster cremaster muscle. *Am J Physiol-Heart Circ Physiol*. 1982 Feb 1;242(2):H211–9.

218. Fenton BM, Carr RT, Cokelet GR. Nonuniform red cell distribution in 20 to 100 μm bifurcations. *Microvasc Res.* 1985 Jan 1;29(1):103–26.
219. Fenton BM, Wilson DW, Cokelet GR. Analysis of the effects of measured white blood cell entrance times on hemodynamics in a computer model of a microvascular bed. *Pflüg Arch.* 1985 Apr 1;403(4):396–401.
220. Chien S, Tvetenstrand CD, Epstein MA, Schmid-Schonbein GW. Model studies on distributions of blood cells at microvascular bifurcations. *Am J Physiol - Heart Circ Physiol.* 1985 Apr 1;248(4):H568–76.
221. Schmid-Schönbein GW, Usami S, Skalak R, Chien S. The interaction of leukocytes and erythrocytes in capillary and postcapillary vessels. *Microvasc Res.* 1980 Jan 1;19(1):45–70.
222. Sarelius IH, Sinclair JD. Effects of small changes of blood volume on oxygen delivery and tissue oxygenation | *American Journal of Physiology-Heart and Circulatory Physiology.* *J Physiol-Heart Circ Physiol.* 1981;
223. Fournier RL. *Basic Transport Phenomena in Biomedical Engineering*, Fourth Edition. CRC Press; 2017. 557 p.
224. Balogh P, Bagchi P. A computational approach to modeling cellular-scale blood flow in complex geometry. *J Comput Phys.* 2017 Apr 1;334:280–307.
225. Balogh P, Bagchi P. Direct Numerical Simulation of Cellular-Scale Blood Flow in 3D Microvascular Networks. *Biophys J.* 2017 Dec 19;113(12):2815–26.
226. Balogh P, Bagchi P. Analysis of red blood cell partitioning at bifurcations in simulated microvascular networks. *Phys Fluids.* 2018 May 1;30(5):051902.
227. Klitzman B, Duling B. Microvascular hematocrit and red cell in resting and contracting striated muscle. Vol. 237. 1979. H481 p.
228. Tang Z, Lee JH. Effects of Different Hematocrit Levels on Glucose Measurements With Handheld Meters for Point-of-Care Testing. *Arch Pathol Lab Med.* 2000;124:6.
229. Spiess A-N, Neumeyer N. An evaluation of R^2 as an inadequate measure for nonlinear models in pharmacological and biochemical research: a Monte Carlo approach. *BMC Pharmacol.* 2010;10(1):6.
230. Desai B, Hobbs J, Hartung G, Xu G, Gokaslan ZL, Linninger A, et al. Image-guidance technology and the surgical resection of spinal column tumors. *J Neurooncol.* 2017 Feb 1;131(3):425–35.
231. Hsu C-Y, Ghaffari M, Alaraj A, Flannery M, Zhou XJ, Linninger A. Gap-free segmentation of vascular networks with automatic image processing pipeline. *Comput Biol Med.* 2017 Mar 1;82:29–39.
232. Ghaffari M, Hsu C-Y, Linninger AA. Automatic Reconstruction and Generation of Structured Hexahedral Mesh for Non-planar Bifurcations in Vascular Networks. In: Gernaey KV, Huusom JK, Gani R, editors. *Computer Aided Chemical Engineering*. Elsevier; 2015. p. 635–40.
233. Linninger AA, Tangen K, Hsu C-Y, Frim D. Cerebrospinal Fluid Mechanics and Its Coupling to Cerebrovascular Dynamics. *Annu Rev Fluid Mech.* 2016;48(1):219–57.
234. Hartung G, Vesel C, Morley R, Alaraj A, Sled J, Kleinfeld D, et al. Simulations of blood as a suspension predicts a depth dependent hematocrit in the circulation throughout the cerebral cortex. *PLOS Comput Biol.* (In Print).

235. Thrane AS, Thrane VR, Zeppenfeld D, Lou N, Xu Q, Nagelhus EA, et al. General anesthesia selectively disrupts astrocyte calcium signaling in the awake mouse cortex. *Proc Natl Acad Sci*. 2012;109(46):18974–18979.
236. Drew PJ, Shih AY, Kleinfeld D. Fluctuating and sensory-induced vasodynamics in rodent cortex extend arteriole capacity. *Proc Natl Acad Sci*. 2011 May 17;108(20):8473–8.
237. Holtmaat A, Bonhoeffer T, Chow DK, Chuckowree J, De Paola V, Hofer SB, et al. Long-term, high-resolution imaging in the mouse neocortex through a chronic cranial window. *Nat Protoc*. 2009;4(8):1128.
238. Mattson DL. Comparison of arterial blood pressure in different strains of mice. *Am J Hypertens*. 2001 May 1;14(5):405–8.
239. Zhao M, Charbel FT, Alperin N, Loth F, Clark M. Improved phase-contrast flow quantification by three-dimensional vessel localization. *Magn Reson Imaging*. 2000;18(6):697–706.
240. Quateroni A, Sacco R, Saleri F. *Numerical Mathematics*. Secaucus, NJ, USA: Springer-Verlag New York, Inc; 2006.
241. Wright GB, Javed M, Montanelli H, Trefethen LN. Extension of Chebfun to Periodic Functions. *SIAM J Sci Comput*. 2015 Jan;37(5):C554–73.
242. Hsu C-Y, Ghaffari M, Hartung G, Park C-S, Rashidisabet H, Linninger A. *Fourier series: mathematical concepts and applications*. UIC LPPD; 2018.
243. Quateroni A, Sacco R, Saleri F. *Numerical Mathematics*. Secaucus, NJ, USA: Springer-Verlag New York, Inc; 2006.
244. Wright GB, Javed M, Montanelli H, Trefethen LN. Extension of Chebfun to Periodic Functions. *SIAM J Sci Comput*. 2015 Jan;37(5):C554–73.
245. T. A. Driscoll, N. Hale, and L. N. Trefethen, editors, *Chebfun Guide*, Pafnuty Publications, Oxford, 2014.
246. Quateroni A, Sacco R, Saleri F. *Numerical Mathematics* [Internet]. 2nd ed. Berlin Heidelberg: Springer-Verlag; 2007. (Texts in Applied Mathematics).
247. Balay S, Gropp WD, McInnes LC, Smith BF. Efficient Management of Parallelism in Object Oriented Numerical Software Libraries. In: Arge E, Bruaset AM, Langtangen HP, editors. *Modern Software Tools in Scientific Computing*. Birkhäuser Press; 1997. p. 163–202.
248. Saad Y. *Iterative methods for sparse linear systems*. Vol. 82. siam; 2003.
249. Anzt H, Dongarra J, Flegar G, Higham NJ, Quintana-Ortí ES. Adaptive precision in block-Jacobi preconditioning for iterative sparse linear system solvers. *Concurr Comput Pract Exp*. 2019;31(6):e4460.
250. Axelsson O, Polman B. On approximate factorization methods for block matrices suitable for vector and parallel processors. *Linear Algebra Its Appl*. 1986;77:3–26.
251. Starling EH. On the absorption of fluids from the connective tissue spaces. *J Physiol*. 1896;19(4):312–326.
252. Levick JR, Michel CC. Microvascular fluid exchange and the revised Starling principle. *Cardiovasc Res*. 2010 Jul 15;87(2):198–210.

253. Linninger AA, Xu C, Tangen K, Hartung G. Starling forces drive intracranial water exchange during normal and pathological states. *Croat Med J*. 2017 Dec;58(6):384–94.
254. Patankar S. Numerical heat transfer and fluid flow. CRC press; 2018.
255. Zhang L, Moon J, Park CS, Linninger A. Newton, steepest descent and trust region methods. Chicago, USA: UIC; 2005.
256. Biegler LT. Nonlinear Programming: Concepts, Algorithms, and Applications to Chemical Processes. SIAM; 2010. 411 p.
257. Horbinski C. What do we know about IDH1/2 mutations so far, and how do we use it? *Acta Neuropathol (Berl)*. 2013 May 1;125(5):621–36.
258. Hartung G, Alaraj A, Linninger A. Chapter 21 - Walk-In Brain: Virtual Reality Environment for Immersive Exploration and Simulation of Brain Metabolism and Function. In: Martín M, Eden MR, Chemmangattuvalappil NG, editors. *Computer Aided Chemical Engineering*. Elsevier; 2016. p. 649–58. (Tools For Chemical Product Design; vol. 39).
259. Golub AS, Barker MC, Pittman RN. Microvascular oxygen tension in the rat mesentery. *Am J Physiol-Heart Circ Physiol*. 2008 Jan;294(1):H21–8.
260. Lancaster JR. A Tutorial on the Diffusibility and Reactivity of Free Nitric Oxide. *Nitric Oxide*. 1997 Feb 1;1(1):18–30.
261. Mintun MA, Lundstrom BN, Snyder AZ, Vlassenko AG, Shulman GL, Raichle ME. Blood flow and oxygen delivery to human brain during functional activity: theoretical modeling and experimental data. *PNAS*. 2001;98(12):6859–64.
262. Wood J, Garthwaite J. Models of the diffusional spread of nitric oxide: Implications for neural nitric oxide signalling and its pharmacological properties. *Neuropharmacology*. 1994 Nov 1;33(11):1235–44.
263. Truskey G, Yuan F, Katz D. Transport Phenomena in Biological Systems. Pearson Prentice Hall;
264. Fang Q, Sakadžić S, Ruvinskaya L, Devor A, Dale AM, Boas DA. Oxygen Advection and Diffusion in a Three Dimensional Vascular Anatomical Network. *Opt Express*. 2008 Oct 27;16(22):17530–41.
265. Shen Q, Ren H, Duong TQ. BCF, BOLD, CBV, and CMRO2 fMRI Signal Temporal Dynamics at 500-ms Resolution. *J Magn Imaging*. 2008;27:599–606.
266. Popel S, Pittman N, Ellsworth L. Rate of oxygen loss from arterioles is an order of magnitude higher than expected. :4.
267. Boas DA, Jones SR, Devor A, Huppert TJ, Dale AM. A vascular anatomical network model of the spatio-temporal response to brain activation. *NeuroImage*. 2008 Apr 15;40(3):1116–29.
268. Zhang R, Kadar T, Sirimanne E, MacGibbon A, Guan J. Age-related memory decline is associated with vascular and microglial degeneration in aged rats. *Behav Brain Res*. 2012 Dec 1;235(2):210–7.
269. Brown WR, Thore CR. Review: Cerebral microvascular pathology in ageing and neurodegeneration. *Neuropathol Appl Neurobiol*. 2011 Feb 1;37(1):56–74.

270. Buxhoeveden DP, Casanova MF. The minicolumn hypothesis in neuroscience. *Brain*. 2002 May 1;125(5):935–51.
271. Cai C, Fordsmann JC, Jensen SH, Gesslein B, Lønstrup M, Hald BO, et al. Stimulation-induced increases in cerebral blood flow and local capillary vasoconstriction depend on conducted vascular responses. *Proc Natl Acad Sci*. 2018 Jun 19;115(25):E5796–804.
272. Christodoulou NS. An algorithm using Runge-Kutta methods of orders 4 and 5 for systems of ODEs. *Int J Numer Methods Appl*. 2009;2(1):47–57.
273. Hill AV. The possible effects of the aggregation of the molecules of haemoglobin on its dissociation curves. *J Physiol*. 1910;40:4–7.
274. Ress D, Thompson JK, Rokers B, Khan R, Huk AC. A model for transient oxygen delivery in cerebral cortex. *Front Neuroenergetics*. 2009;1:3.
275. Kim JH, Ress D. Arterial impulse model for the BOLD response to brief neural activation. *NeuroImage*. 2016 Jan 1;124:394–408.
276. Kim JH, Khan R, Thompson JK, Ress D. Model of the transient neurovascular response based on prompt arterial dilation. *J Cereb Blood Flow Metab*. 2013;33(9):1429–1439.
277. Huppert TJ, Allen MS, Benav H, Jones PB, Boas DA. A multicompartiment vascular model for inferring baseline and functional changes in cerebral oxygen metabolism and arterial dilation. *J Cereb Blood Flow Metab*. 2007;27(6):1262–1279.
278. Buxton RB, Uludağ K, Dubowitz DJ, Liu TT. Modeling the hemodynamic response to brain activation. *NeuroImage*. 2004 Jan 1;23:S220–33.
279. Griffeth VEM, Buxton RB. A theoretical framework for estimating cerebral oxygen metabolism changes using the calibrated-BOLD method: Modeling the effects of blood volume distribution, hematocrit, oxygen extraction fraction, and tissue signal properties on the BOLD signal. *NeuroImage*. 2011 Sep 1;58(1):198–212.
280. Jespersen SN, Østergaard L. The Roles of Cerebral Blood Flow, Capillary Transit Time Heterogeneity, and Oxygen Tension in Brain Oxygenation and Metabolism. *J Cereb Blood Flow Metab*. 2012 Feb 1;32(2):264–77.
281. Secomb T, Hsu R. Analysis of oxygen delivery to tissue by microvascular networks. *Oxyg Transp Tissue X*. 1988;95–103.
282. Park CS, Payne SJ, others. A model of oxygen dynamics in the cerebral microvasculature and the effects of morphology on flow and metabolism. 2014;
283. Park Chang Sub, Payne Stephen J. A generalized mathematical framework for estimating the residue function for arbitrary vascular networks. *Interface Focus*. 2013 Apr 6;3(2):20120078.
284. Lipowsky HH. Microvascular rheology and hemodynamics. *Microcirculation*. 2005;12(1):5–15.
285. Lobdell DD. An invertible simple equation for computation of blood O₂ dissociation relations. *J Appl Physiol*. 1981;50(5):971–973.
286. Payne SJ, El-Bouri WK. Modelling dynamic changes in blood flow and volume in the cerebral vasculature. *NeuroImage*. 2018 Aug 1;176:124–37.

9 Vita

Education

University of Illinois at Chicago (UIC)

Doctor of philosophy in Bioengineering

GPA: 3.83/4.0

completed 11/2019

walking 5/2020

University of Illinois at Chicago (UIC)

Master of Science in Neural Bioengineering

GPA: 4.0/4.0

May 2015

University of Illinois at Chicago (UIC)

Bachelor of Science in Bioengineering with a Neural Engineering concentration

Minor: Mechanical Engineering

August 2012

Research Interest

I am deeply interested in computational research including simulations, big data analysis and unique data visualizations. Specifically, deriving/implementing new simulation approaches that investigate mechanistic trends and *causation* for mechanisms of the aging brain. I am also familiar with biofiltration of nanoparticles and immune system reactions to invasive bodies.

Research Experience

Current Projects:

6/17 – 12/19 **Deterministic simulation of oxygen in aging brain - Dr. Andreas Linninger**

The aging brain is known to exhibit different mechanisms of vascular degradation which can be implemented using our vascular growth algorithm. Recent measurements of changes in brain tissue oxygen tension allow the numerical comparison of changes in vasculature with tissue hypoxic micro-pocket formation. The contribution of individual age-related vascular dysfunction can then be investigated.

- Generated a novel simulation platform based on a cartesian mesh masking technique that surpassed the previous supercomputer record for largest extravascular simulation of brain tissue using only a personal computer
- The new paradigm allowed stable integration of 1D-3D coupling capable of investigating oxygen transport across the blood brain barrier at the scale of the whole-brain (~100x larger than previously possible)
- Larger simulation domain enabled simulation of age-related vascular changes without significant impact from boundary conditions at domain edges

Past Projects:

6/17 – 5/19 **Simulation of neurovascular dynamics of oxygen transport and vasomotion in functional hyperemia - Dr. Andreas Linninger**

Investigation of mechanistic interactions between the extravascular neuronal environment during repeated activation gives unprecedented investigation of differing mechanisms of delivery. Classic theories of passive blood flow response and cellular theories of active dissociation in the presence of elevated pH render differing extravascular oxygen profiles.

Simulations of large volumes of tissue ($>1\text{ mm}^3$) of the complete neurovascular unit are the largest of their kind and allow unique perspective of mechanistic interactions between parts.

- Object oriented program design in Pascal, Matlab, and C++ for model generation, multilingual data analysis, and visualization
- Derived, implemented and compared numerous time integrators for comparison

1/17 – 4/17 Optimization for deriving cerebrovascular simulation results directly from blood flow measurements - Dr. Andreas Linninger

Formulated and implemented a method that deterministically minimizes the difference between simulation results and empirical measurements. This included the nominal (steady-state) condition as well as transient results derived in the time domain and in the Fourier domain.

- Formulated optimization problem minimizing error between experiments and simulations that used an entire simulation as constraints and was able to be simplified to a single linear algebraic set of equations
- Implemented in both object Matlab and Object Pascal for comparison
- Formulated in Fourier domain for advanced filtering

4/15 – 8/17 Biphasic blood flow simulations - Dr. Andreas Linninger

Investigating the effects of nonlinear red blood cell skimming in the cerebral microcirculation at the scale of the whole brain. A vascular synthesis algorithm was adapted and expanded to grow and validate synthetic microcirculatory networks up to 1/5 of the mouse brain. This investigation discovered a correlation between cortical depth and red blood cell concentration in the vasculature.

- Implemented fixed-point iterative and successive overrelaxation procedures for solving very large (>1 million) systems of highly nonlinear equations in less than 1 hour on standard desktop computer
- Debugged and optimized growth algorithm to drastically reduce computational time and allow algorithm to expand beyond 10% density limit

9/14 – 1/17 Big data visualization - Dr. Andreas Linninger

Expanded a preliminary application of visualization toolkit (VTK), an open-source C++ library for enhanced graphics visualizations, to include 3D medical image rendering and 3D reconstruction and simulation visualization.

- Object oriented programming in C++ utilizing VTK for 3D immersive data visualization

8/11 – 5/15 Molecular pharmacodynamic model development of cellular reactions to carbon nanotubes in-vivo - Dr. G. Ali Mansoori

11/12-5/15 Challenged the accepted theory of in-vivo carbon nanotube toxicity. Compiled a decision tree to determine the toxicity of any characterized nanoparticle influenced by the newly developed nanotube toxicity model.

8/11-11/12 Reviewed the last 40 years of nanoparticle research in vivo to discern governing factors of toxicity and filtration of nanoparticles out of the body.

9/13 – 9/14 Fast axonal transport simulation - Dr. Gerardo Morfini

Created a full-scale stochastic simulation of the subcellular axonal transport mechanisms. visual simulation of axonal transport with which we can test different theories of mechanisms behind

fast axonal transport. Programming in Matlab utilizing GUIs, active drawing plots, and optimized matrix computations allows parametric studies of changes in axonal transport mechanisms and quantitatively compare with empirical results.

- Developed and optimized matrix computations in Matlab to investigate transient vesicular transport stochastically

9/11 **Neural Engineering Lab** (September 2011) - **Dr. John Hetling PhD**

Created a device to control a computer cursor via arm movement sensors. First student team to ever accomplish this task. Project was heavily reliant on data collection and analysis.

- Used continuous neural recordings, filtered and amplified through analog circuitry and digitized using an Arduino
- Utilized preexisting mouse motherboard to convert digital Arduino signal into windows-ready mouse movements

9/10 – 4/11 **Senior Design** "An Acoustic Signal Acquisition System for a Fetal Heart Rate Monitoring Device" (April 26, 2011) - **Heartsounds Inc.**

Created an apparatus to acquire and filter in real time up to 3 separate fetal heart rate signals during pregnancy and delivery. Digitized filtered analog signal was processed through a blind-source separation algorithm to differentiate multiple individual fetal heartbeats. Adapted new scientific practice (silicon-infused piezoelectric microphones) to acquire such a signal efficiently (SNR of at least 10 dB) with extreme cost-efficiency. Project was heavily reliant on data collection. A prototype was fabricated and performed exceeding expectations.

- Created custom gel-infused piezoelectric microphones and artificial heartbeat signal in phantom model

1/10-8/10 **University of Chicago Human Neuroscience Department - Dr. Ana Solodkin**

PhD Bioengineer and Information Technologist/Programmer – HIPPA

Certified

Utilized magnetic fields to induce brain-derived neural stimulation which elicited hand and foot motion. Upon the elicitation of motion, the lower leg and forearm neural potentials were measured in both control subjects and subjects suffering from degenerative nerve disorders. The results created a system to quantitatively determine patient nerve degeneration.

- Compiled and debugged code executed on open science grid computers using unix command line (RHE Linux)
- Responsible for lab-wide bioengineering-related troubleshooting

7/09-9/09 **UIC Pathology Department Medical Branch - Dr. Virgilia Macias PhD**

Research Project Patient Admission Clerk – HIPPA Certified

- Recruited patients into Pathology Department research programs

4/09-9/09 **UIC Pathology Department (West Campus) - Dr. Andre Balla PhD**

Research Assistant

This study was designed to examine the effects of flax seed oil on ovarian cancer from chicken subjects.

- Created ovarian tissue microarrays
- Increased productivity 800% with notably higher consistency than previous assistants improving project results for medical research team

9/08-9/09 **UIC Pathology Department (West Campus) - Dr. Virgilia Macias PhD**

Research Assistant

Evaluated transferrin receptor expression in relation to prostate cancer.

- Identified and marked on digital microarrays cellular level cancerous prostate tissue requiring attention to detail

Publications

In preparation

G Hartung, S Badr, M Moeini, F Lesage, and A Linninger. "Multi-scale simulation of cerebral blood flow and oxygen exchange for the mouse cortex."

G Hartung, S Badr, and A Linninger. "Mathematical synthesis of the cortical circulation for the whole mouse brain-part II. microvascular closure and topological matching."

A Linninger, **G Hartung**, J Marek, M Hoeller, A Alaraj. "Global blood flow patterns and trends in the entire mouse hemisphere."

Published

2019 Park, CS, **G Hartung**, A Alaraj, X Du, FT Charbel, and AA Linninger. "Quantification of blood flow patterns in the cerebral arterial circulation of individual (human) subjects." *International journal for numerical methods in biomedical engineering*. 2019.

2019 A Linninger, **G Hartung**, S Badr, and R Morley. "Mathematical synthesis of the cortical circulation for the whole mouse brain-part I. theory and image integration." *Computers in Biology and Medicine*. 2019.

2018 **G Hartung**, C Vesel, R Morley, A Alaraj, J Sled, D Kleinfeld, A Linninger. "Simulations of blood as a suspension predicts a depth dependent hematocrit in the circulation throughout the cerebral cortex." *PLoS Computational Biology*, 14(11), e1006549, 2018.

2018 A Linninger, **G Hartung**, BP Liu, S Mirkov, K Tangen, RV Lukas, D Unruh, CD James, JN Sarkaria, C Horbinski. "Modeling the diffusion of D-2hydroxyglutarate from IDH1 mutant gliomas in the central nervous system." *Journal of Neuro-Oncology*, 20(9) 1197-1206, 2018.

2017 A Linninger, C Xu, K Tangen, **G Hartung**. "Starling forces drive intracranial water exchange during normal and pathological states." *Croatian Medical Journal*, 58, 384-394, 2017

2017 B Desai, J Hobbs, **G Hartung**, G Xu, ZL Gokaslan, A Linninger, A Mehta. "Image-guidance technology and the surgical resection of spinal column tumors." *Journal of Neuro-Oncology*, 131(3), 425-436, 2017.

2016 **G Hartung**, A Alaraj, A Linninger. "Walk-In Brain: virtual reality environment for immersive exploration and simulation of brain metabolism and function." *Tools for Chemical Product Design*, 649-658, 2016.

2013 **G Hartung**, G. Ali Mansoori, "In Vivo General Trends, Filtration and Toxicity of Nanoparticles." *J Nanomaterials & Molecular Nanotechnology*, 2(3) 1-21, 2013 (Invited Paper)

4/13 **G Hartung**, "Fullerenes; Possible to Evacuate Post-Administration or Are They Just Toxic?" *Undergraduate Bioengineering Student Journal of the University of Illinois at Chicago*, 4(1):9-14, 2013. (Invited Paper)

Posters and Presentations

- 7/18 **Grant Hartung**, R Morley, A Linninger. *Whole brain simulation to elucidate mechanisms of functional hyperemia. Poster presented* at 8th World Congress of Biomechanics, Dublin, Ireland.
- 6/18 A Linninger, **Grant Hartung**, R Morley. *In silico mouse brain synthesis, blood brain barrier, water transport and homeostasis. Poster presented* at 8th World Congress of Biomechanics, New London, New Hampshire.
- 4/17 **Grant Hartung**, G Xu, A Linninger. *Immersive medical media – a platform for dynamic exploration for automatic, subject-specific atlases from standard medical images. Poster presented* at Brain 2017, Berlin, Germany.
- 5/15 **Grant Hartung**, S Alford, A Linninger. *Quantification of charge dependence on transport through ion channels. Poster presented* at UIC Research Forum 2015, Chicago, Illinois.
- 4/13 **Grant Hartung**, G. Ali Mansoori, *In vivo biofiltration: emerging trends of nanoparticles". Poster presented* at Midwest Biomedical Engineering Career Conference 2013, Chicago, Illinois.
- 6/13 **Grant Hartung**, G. Ali Mansoori, *In vivo biofiltration: emerging trends of nanoparticles". Poster presented* at Biosensors & Bioelectronics - 2013, Northbrook, Illinois

Teaching Experiences

07/11 - 08/12 **Teaching Assistant and Bioengineering Coordinator for Introduction to Engineering course at UIC**

- General bioengineering liaison for incoming freshman and transfer students
- Created and Implemented bioengineering specific curriculum
- Taught detailed guidance and comprehensive assessments of bioengineering field

Professional Affiliations

- BioMedical Engineering Society Student Member (10/08 – 9/10)
- Society of Automotive Engineers Student Member (5/10-present)
- Journal of Neural Engineering (Registered 2/12)
- Bioengineering Organizational Alliance (08/11-5-12)
- UIC Alumni Association - (May 2013-Present)

Skills

Computer: Designing and assembling custom computers, network assembly, and maintenance, general IT

Proficient in: Object Pascal (Delphi), Matlab, C++, Acrobat, FileZilla FTP, Powerpoint, Excel, Word, ITK-Snap, dll creation and linking, VTK

Intermediate knowledge in: C#, HTML, Autodesk 3D Studios Max, linux (RHEL and Ubuntu), LabView, Adobe Dreamweaver, Freesurfer, autocad, solidworks

Wood: lathing, carpentry (both design and fabrication), routing

Plumbing: design and fabrication of intricate in home and irrigation systems of both PVC and copper

Electrical: 12V DC, 120V AC, 240V AC and 3-phase AC design and fabrication of entire rooms and systems, knowledge with Remote-Control systems for both design and fabrication, soldering, prototype circuit board design and installation experience

Machinist: Lathing, Milling, Welding (Arc, Mig, Tig, and Oxy-Acetylene for steel and Mig, Arc, and Oxy-Acetylene for aluminum), precision drilling (drill press, lathe or mill), notching
Auto mechanic, landscaper, repairman, carpenter, woodworker, and mason

Awards

June 18, 2013 **Best Poster Award - OMICS Group 2013 Biosensors & Bioelectronics Conference**

April 13, 2012 **Chancellor's Student Service Award**

April 18, 2008 **Best New Society (BMES)**

Non-Relevant Work Experience

8/12-8/13 **Franck's Construction**

Intensive handyman work focusing on carpentry. Main focus of work was rehabilitating houses and rebuilding houses. The work included the use of intricate masonry, carpentry, roofing, painting, plumbing, electrical and interior design work. The work at times extended to gardening, landscaping, machining, and even automotive repair work.

SAE Mini Baja Team Chief fabricator and machinist (07/09-06/11)

Co-captain (06/11– 8/12)

Led and was head of recruitment for an undergraduate engineering design and build team which annually created a fully operational, full size, manned off-road vehicle. Competed in multiple SAE sponsored week long international competitions against over 100 different universities in design and operation events. The custom prototypes required multiple custom metal frame, steering, and drivetrain parts for which I was the machinist, fabricator and welder for the team.