TLISMNI/ADAMS ALGORITHM FOR THE SOLUTION OF THE DIFFERENTIAL/ALGEBRAIC EQUATIONS OF CONSTRAIND DYNAMICAL SYSTEMS

Ahmed A. Shabana (shabana@uic.edu)
Dayu Zhang (dyzhang@mail.nwpu.edu.cn)
Gengxiang Wang (gengxiangwang@gmail.com)

[1]Department of Mechanical and Industrial Engineering, University of Illinois at Chicago, 842 West Taylor Street, Chicago, Illinois 60607
[2]National Key Laboratory of Aerospace Flight Dynamics, School of Astronautics, Northwestern Polytechnical University, Xi'an 710072, Shaanxi, P.R. China
[3]Faculty of Mechanical and Precision Instrument Engineering, Xi'an University of Technology, Xi'an 710048, Shaanxi, P.R. China

ABSTRACT

This paper examines the performance of the 3rd and 4th order implicit Adams methods in the framework of the **t**wo-**l**oop **i**mplicit **s**parse **m**atrix **n**umerical **i**ntegration (TLISMNI) method in solving the differential/algebraic equations (DAE's) of heavily constrained dynamical systems. The variable-step size TLISMNI/Adams method proposed in this investigation avoids numerical force differentiation, ensures satisfying the nonlinear algebraic constraint equations at the position, velocity, and acceleration levels, and allows using sparse matrix techniques for the efficiently solving the dynamical equations. The iterative outer loop of the TLISMNI/Adams method is aimed at achieving the convergence of the implicit integration formulae used to solve the independent *differential equations* of motion, while the inner loop is used to ensure the convergence of the iterative procedure used to satisfy the *algebraic constraint equations*. To solve the independent differential equations, two different implicit Adams integration formulae are examined in this investigation; a 3rd order implicit Adams-Moulton formula with a 2nd order explicit predictor Adams Bashforth formula, and a 4th order implicit Adams-Moulton formula with a 3rd order explicit predictor Adams Bashforth formula. A standard Newton-Raphson algorithm is used to satisfy the nonlinear algebraic constraint equations at the position level. The constraint equations at the velocity and acceleration levels are linear, and therefore, there is no need for an iterative procedure to solve for the dependent velocities and accelerations. The algorithm used for the error check and step-size change is described. The performance of the TLISMNI/Adams algorithm developed in this investigation is evaluated by comparison with the explicit predictor-corrector Adams method which has a variable-order and variable-step size. Simple and heavily constrained dynamical systems are used to evaluate the accuracy, robustness, damping characteristics, and effect of the outer-loop iterations of the proposed implicit schemes. The results obtained in this investigation show that the TLISMNI methods proposed in this study can be more efficient for stiff systems because of their ability to damp out high-frequency oscillations. Explicit integration methods, on the other hand, can be more efficient in the case on non-stiff systems.

Keywords: Differential/algebraic equations; implicit integration; constrained dynamical systems; multibody systems; Adams integration methods.

1. INTRODUCTION

The motion of multibody systems (MBS) is governed by a system of differential/algebraic equations (DAE's) [3]. The differential equations define the equations of motion, while the algebraic equations define the nonlinear kinematic constraints in the MBS application. These constraint equations describe mechanical joints as well as specified motion trajectories. In general MBS algorithms, the algebraic equations cannot always be easily eliminated, particularly in the case of closed-loops for which it is difficult to explicitly write the dependent coordinates as functions of the independent joint coordinates. For this reason, general MBS algorithms are designed as DAE's solvers and allow for the use of an iterative procedure to solve for the nonlinear kinematic constraint equations at the position level. In this investigation, the numerical process of determining the dependent variables from the system degrees of freedom is referred to as the *inner loop*, while the iterative process for determining the independent variables (coordinates and velocities) using the implicit integration formulae is referred to as the *outer loop*. An explicit *numerical integration scheme* can always be as a special case of the implicit methods used in this investigation by limiting the number of iterations of the outer loop to one.

Both explicit and implicit methods are used to solve the resulting MBS equations [6, 7, 11, 12, 14]. Explicit methods, such as the explicit predictor-corrector Adams method with variable order and variable step size, are widely used in the solution of the MBS DAE's [15]. Explicit methods, however, are not suited for solving stiff equations which characterize MBS applications, especially when flexible bodies with very high stiffness are considered. Such explicit methods often fail or become inefficient when stiff systems are analyzed. On the other hand, while implicit methods can be more efficient in solving stiff differential equations, existing implicit methods have several drawbacks that include the possibility of damping out some important details if used by inexperienced user. In the case of DAE's that characterize MBS applications, existing implicit methods suffer from serious problems that can lead to deterioration of the method efficiency and robustness, and to violation of basic mechanics principles. For example, most existing implicit methods require force differentiation, do not satisfy the constraint equations at all levels, and do not allow for exploiting sparse matrix techniques. Numerical differentiation can source of serious numerical problems when general MBS algorithms are considered, particularly when the system includes deformable bodies with high stiffness. The numerical differentiation of the flexible body forces is prone to errors that can lead to deterioration of the accuracy of the solution or even to divergence problems. Not satisfying the nonlinear algebraic constraint equations at the position, velocity, and acceleration levels is violation of the principle of mechanics. The basic Lagrange-D'Alembert principle as well as other principles used in the derivation of the equations that govern the motion of dynamical systems are based on the assumption that the constraint equations are satisfied at all levels regardless of whether or not these constraint equations are eliminated using the embedding technique or kept in the formulation using the technique of Lagrange multipliers [6, 7, 11, 12, 14]. Furthermore, adopting sparse matrix techniques is necessary for obtaining efficient and accurate solution of heavily constrained and complex dynamical systems [8, 9]. However, most existing implicit methods require the differentiation of both the differential and algebraic equations in order to determine the Jacobian matrix associated with the unknown variables. This Jacobian matrix, used in the iterative Newton-Raphson algorithms, is not, in general, a sparse matrix, and therefore, efficient sparse matrix techniques that also require significantly smaller array space cannot be effectively exploited.

In order to address these concerns regarding the use of implicit integrators in MBS simulations, the **t**wo-**l**oop **i**mplicit **s**parse **m**atrix **n**umerical **i**ntegration (TLISMNI) method was proposed [1, 5, 13]. Unlike other implicit numerical integration algorithms, the TLISMNI method does not require the numerical differentiation of the forces, ensures that the constraint equations are satisfied at all levels, and allows for effectively using sparse matrix techniques at all stages of the function evaluation. The TLISMNI method is designed to have two iterative loops; the outer and inner loops. The outer loop is designed to achieve convergence of the implicit integration formulae (differential equations), while the inner loop is designed to ensure that the nonlinear holonomic kinematic algebraic constraint equations are not violated. The algebraic equations at the velocity and acceleration levels are linear equations and do not require the use of an iterative procedure in order to obtain their solution. While the TLSIMNI method has been used in solving a large number of MBS applications, it has been only used with 2nd order integration methods such as Hilber–Hughes–Taylor (HHT), Trapezoidal, and BDF methods [1]. Furthermore, no analysis has been provided to examine the effectiveness of the outer loop of the TLISMNI algorithm.

It is the objective of this investigation to develop a TLISMNI algorithm based on the implicit 3rd and 4th order Adams integration methods and compare the performance of the new algorithm with the TLISMNI algorithm based on the 2nd order integration formula defined by the trapezoidal method as well as with the explicit predictor-corrector variable-order Adams method [15]. In the algorithm developed in this study two different Adams integration formulae are used; the 2nd order explicit Adams-Bashforth formula is used as the predictor for the 3rd order implicit corrector Adams-Moulton method, and the 3rd order explicit Adams-Bashforth formula is used as the predictor for the 4th order implicit corrector Adams-Moulton formula [2]. Because of the nature of the large scale problems and the intensive computations required to evaluate the equations of motion, the number of iterations of the outer loop is kept small to increase the efficiency of the method. During the outer iterations, the time step is kept constant, the predictor formula is not used after the first iteration, and if the outer iterations do not converge, the time step is reduced before restarting the iterations. Because the iterations for the position analysis cannot be in general avoided, particularly in the case of MBS with closed loops, use of a Newton-Raphson algorithm for the inner loop ensures that the nonlinear algebraic constraint equations are satisfied. The error check used to determine the convergence of the outer loop involves only the independent variables, and the dependent variables (coordinates and velocity) are determined using the nonlinear algebraic equations, thereby ensuring that these algebraic equations are satisfied and there is no violation of the basic physics principles (17, 12]. The effectiveness of the outer loop iterations is evaluated in this investigation, and the results obtained have shown that such outer loop iterations are necessary in order to achieve the convergence and robustness of the TLISMNI/Adams algorithm proposed in this investigation. For the first outer loop iteration, the error is estimated based on the difference between the corrected and predicted solutions, and for subsequent outer loop iterations, the error is estimated based on the difference between the current iteration corrector solution and the corrector solution of the previous iteration. The effect of the time step size and error tolerance on the damping characteristics of the proposed algorithm is evaluated in order to shed light on the disadvantages and advantages of using implicit numerical integration methods. Several numerical examples, including examples that represent large scale and heavily constrained systems, are used in order to properly asses the effect of increasing the order of the integration formula and the effectiveness of the outer loop iterations.

2. ADAMS INTEGRATION FORMULAE

Low order implicit integration methods such as Newmark and HHT are widely used in the field of computational mechanics to solve challenging physics and engineering problems. While low-order implicit methods can be very effective in solving many problems, they must be used with care because of the inherent nature of numerical damping and inability to capture accurately high speed and highly nonlinear rotational motion. Nonetheless, in dynamical systems with stiff deformable bodies and/or springs, the high frequency oscillations may have a negligible effect on the solution. Accurate explicit methods attempt to capture these high frequency oscillations; making the method very inefficient or in many cases fail. For this reason, a general purpose MBS algorithm that is based only on an explicit solver can have serious limitations and can fail in solving many practical problems. Implicit integrators, on the other hand, have the ability to damp out high frequency signals, and therefore, such implicit methods can be much more efficient in solving many engineering problems in which the effect of high-frequency oscillations on the solution accuracy is negligible. Because the decision of which signals are important and should not be filtered out depends mainly on the experience of the analyst, it is important to provide the two options (explicit and implicit solvers) in general-purpose MBS software. The stiff problems in which the explicit integrator works, regardless of whether or not the solution is efficiently obtained, can be used as reference solution to assess the accuracy of the implicit integration method. Such reference solutions, which can serve as a guide, are particularly important when the widely used general-purpose MBS software are adopted by inexperienced analysts, as it is the case in industries which heavily rely on the virtual testing ground for the product design and performance evaluation. In this section, the integration formulae used in this investigation are presented since they will be later referenced in this paper. More details regarding the derivation of these formulae can be found in standard numerical analysis texts [2, 10].

The TLISMNI method has been proposed and used with 2nd order integration formulae such as HHT and the trapezoidal methods. In this investigation, the performance of two different

Adams integration formulae is examined; the implicit 3rd and 4th order Adams methods are used and the results obtained are compared with the results obtained using the 2nd order trapezoidal method and the results obtained using the explicit variable-order Adams method [15]. In the procedure described in this paper, it is assumed that the equations of motion can be converted to the first-order state-space form and written as $\dot{\mathbf{y}} = \mathbf{f}(\mathbf{y}, t)$, where $\mathbf{y} = \begin{bmatrix} \mathbf{q}^T & \dot{\mathbf{q}}^T \end{bmatrix}^T$, t is time, and $\mathbf{q}$ and $\dot{\mathbf{q}}$ are, respectively, the vector of the system coordinates and velocities. It follows that $\mathbf{y}(t) = \mathbf{y}_0 + \int_0^t \mathbf{f}(\mathbf{y}, t)\, dt$. The integral in this equation can be approximated using different methods leading to the general expression for $p+1$ step methods, including Adams method, as $\mathbf{y}_{n+1} = \sum_{j=0}^{p} a_j \mathbf{y}_{n-j} + h \sum_{j=-1}^{p} b_j \mathbf{f}\left(t_{n-j}, \mathbf{y}_{n-j}\right)$, where $p \geq 0$, $n \geq p$, $\mathbf{y}_n$ is the solution at time t_n, h is the time step size, and a_j and b_j are constants whose values and number define the method and order to be used with the condition that either $a_p \neq 0$ or $b_p \neq 0$ [2]. As explained by Atkinson, the coefficients a_j and b_j can be determined using the method of undetermined coefficients or the method of numerical integration. The 2nd order explicit Adams-Bashforth predictor is defined by the formula

$$\left(\mathbf{y}_{n+1}\right)_{pr} = \mathbf{y}_n + \frac{h}{2}\left(3\dot{\mathbf{y}}_n - \dot{\mathbf{y}}_{n-1}\right) \tag{1}$$

where $\mathbf{y}_n = \mathbf{y}(t_n)$, $\dot{\mathbf{y}}_n = \dot{\mathbf{y}}(t_n, \mathbf{y}_n)$, and subscript pr refers to predictor. The 3rd order implicit Adams-Moulton formula is given by

$$\left(\mathbf{y}_{n+1}\right)_{cr} = \mathbf{y}_n + \frac{h}{12}\left(5\dot{\mathbf{y}}_{n+1} + 8\dot{\mathbf{y}}_n - \dot{\mathbf{y}}_{n-1}\right) \tag{2}$$

where subscript cr refers to corrector. In this investigation, the explicit 2nd order Adams-Bashforth formula is used as the predictor and the 3rd order Adams-Moulton formula is used as the corrector for the first outer loop iteration in the TLISMNI/Adams algorithm. After the end of the iteration, the error is estimated as the norm of the vector $\mathbf{e}_{n+1} = (\mathbf{y}_{n+1})_{cr} - (\mathbf{y}_{n+1})_{pr}$. If the $|\mathbf{e}_{n+1}| = |(\mathbf{y}_{n+1})_{cr} - (\mathbf{y}_{n+1})_{pr}| \le \varepsilon$, where ε is a given error tolerance, the solution is assumed to converge, and no other outer loop iterations are performed. If $|\mathbf{e}_{n+1}| = |(\mathbf{y}_{n+1})_{cr} - (\mathbf{y}_{n+1})_{pr}| > \varepsilon$, other outer iterations are performed using the iterative corrector formula

$$(\mathbf{y}_{n+1})_{cr}^{k} = \mathbf{y}_n + \frac{h}{12}\left(5\dot{\mathbf{y}}_{n+1}^{k-1} + 8\dot{\mathbf{y}}_n - \dot{\mathbf{y}}_{n-1}\right) \tag{3}$$

In this equation, superscript k refers to the outer loop iteration number, and $\dot{\mathbf{y}}_{n+1}^{k-1} = \dot{\mathbf{y}}_{n+1}\left(t_{n+1}, \mathbf{y}_{n+1}^{k-1}\right)$. Convergence of the outer loop iterations is achieved for $k > 1$, if $|\mathbf{e}_{n+1}| = \left|(\mathbf{y}_{n+1})_{cr}^{k} - (\mathbf{y}_{n+1})_{cr}^{k-1}\right| \le \varepsilon$. During these outer loop iterations, the time step h is kept constant. If the outer loop iterations converge, the error is analyzed to determine whether or not the time step size h can be increased. If the outer loop iterations do not converge, the time step h is reduced and the outer loop iterations are restarted with the smaller h .

In addition to the 3rd order implicit Adams method, the 4th order implicit Adams-Moulton corrector formula $(\mathbf{y}_{n+1})_{cr} = \mathbf{y}_n + (h/24)(9\dot{\mathbf{y}}_{n+1} + 19\dot{\mathbf{y}}_n - 5\dot{\mathbf{y}}_{n-1} + \dot{\mathbf{y}}_{n-2})$ is used with the explicit 3rd order Adams-Bashforth predictor formula $(\mathbf{y}_{n+1})_{pr} = \mathbf{y}_n + (h/12)(23\dot{\mathbf{y}}_n - 16\dot{\mathbf{y}}_{n-1} + 5\dot{\mathbf{y}}_{n-2})$. The local truncation error in these higher-order predictor and corrector formulae are defined as in the case of the 3rd order implicit method previously described in this section.

The effect of increasing the order of the integration formula is evaluated by comparing the numerical results obtained using the two different TLISMNI/Adams schemes with the results obtained using the explicit variable-order and variable-step size Adams predictor-corrector method as well as the results obtained using the TLISMNI/trapezoidal implementation. For the 2nd order implicit trapezoidal method, the explicit midpoint method $\left(\mathbf{y}_{n+1}\right)_{pr} = \mathbf{y}_{n-1} + 2h\dot{\mathbf{y}}_n$ is used as the predictor. The implicit trapezoidal corrector is $\left(\mathbf{y}_{n+1}\right)_{cr} = \mathbf{y}_n + \left(h/2\right)\left(\dot{\mathbf{y}}_n + \dot{\mathbf{y}}_{n+1}\right)$. The outer-loop iterations in the TLISMNI method in the case of the trapezoidal formula are defined as $\left(\mathbf{y}_{n+1}\right)_{cr}^{k} = \mathbf{y}_n + \left(h/2\right)\left(\dot{\mathbf{y}}_n + \dot{\mathbf{y}}_{n+1}^{k-1}\right)$.

Since in the case of multistep methods, information from previous steps are needed, the single-step Runge-Kutta method is used at the beginning of the integration to obtain the information at three time points that are required for using the implicit Adams and trapezoidal integration formulae as well as the predictors used with these formulae. Very small time steps are used with Runge-Kutta method in order to guarantee the accuracy of the solution history to be used with the integration formulae that will be examined in this investigation. As previously mentioned, in the TLISMNI algorithm used in this investigation, the outer loop iterations are used to achieve the convergence of the implicit numerical integration formulae, while the inner loop iterations are used to satisfy the algebraic constraint equations used in the evaluation of the vector $\dot{\mathbf{y}}_{n+1}$ required in the integration formula. The evaluation of $\dot{\mathbf{y}}_{n+1}$ using the equations of motion and the algebraic constraint equations is discussed in the following section.

3. CONSTRAINED DYNAMICAL SYSTEMS

The motion of many physics and engineering systems is governed by a system of differential/algebraic equations which are presented in a general form in this section for reference to be made in later sections of the paper. It is also explained in this section, how the inner TLISMNI loop can be designed to obtain a sparse matrix structure in the computer implementation of the dynamic equations in general MBS algorithms.

For a constrained dynamical system, the differential equations define the 2nd order equations of motion, while the algebraic equations define the MBS motion constraints. In MBS applications, the equations of motion are often highly nonlinear due to the geometric nonlinearities that arise from the finite rotations of the system components and the geometric and possibly material nonlinearities that arise, respectively, from the use of nonlinear strain displacement relationships and the use of nonlinear material models. The constraint equations, which define mechanical joints and specified motion trajectories, are also highly nonlinear. In general, the MBS differential and the algebraic equations can be written, respectively, in the following forms $\mathbf{M}\ddot{\mathbf{q}} + \mathbf{C}_{\mathbf{q}}^{\mathrm{T}}\boldsymbol{\lambda} = \mathbf{Q}$, and $\mathbf{C}(\mathbf{q}, t) = \mathbf{0}$ [11, 14, 12], where $\mathbf{M}$ is the system mass matrix, $\mathbf{q}$ is the system generalized coordinate vector, $\mathbf{C}$ is the constraint function vector, t is time, $\mathbf{C}_{\mathbf{q}}$ is the Jacobian matrix of the kinematic constraint equations, $\boldsymbol{\lambda}$ is the Lagrange multiplier vector, and $\mathbf{Q}$ is the generalized force vector that include all external, elastic, Coriolis and centrifugal forces. The first and second time-derivatives of the constraint equations $\mathbf{C}(\mathbf{q}, t) = \mathbf{0}$ define, respectively, the constraint equations at the velocity and acceleration levels as $\mathbf{C}_{\mathbf{q}}\dot{\mathbf{q}} = -\mathbf{C}_t$, and $\mathbf{C}_{\mathbf{q}}\ddot{\mathbf{q}} = \mathbf{Q}_d$. In these equations, $\mathbf{C}_t$ is the partial derivative of the constraint equations with respect to time, and $\mathbf{Q}_d$ is a vector that absorbs terms which are quadratic in the velocities. While the constraint equations at the position level can be highly nonlinear functions of the coordinates, the constraint equations at the velocity and acceleration levels are linear in the velocities and accelerations, respectively. Combining the equations of motion $\mathbf{M}\ddot{\mathbf{q}} + \mathbf{C}_{\mathbf{q}}^{\mathrm{T}}\boldsymbol{\lambda} = \mathbf{Q}$ and the constraint equations at the acceleration level $\mathbf{C}_{\mathbf{q}}\ddot{\mathbf{q}} = \mathbf{Q}_d$ in one matrix equation, one obtains

$$\begin{bmatrix} \mathbf{M} & \mathbf{C}_{\mathbf{q}}^{T} \\ \mathbf{C}_{\mathbf{q}} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{q}} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{Q} \\ \mathbf{Q}_{d} \end{bmatrix} \tag{4}$$

This equation, which has a sparse coefficient matrix, can be solved for the acceleration vector $\ddot{\mathbf{q}}$ and the vector of Lagrange multipliers $\boldsymbol{\lambda}$. The solution of this matrix equation ensures that the constraint equations at the acceleration level are satisfied. Because the preceding matrix equation is linear in the acceleration vector $\ddot{\mathbf{q}}$ and the vector of Lagrange multipliers $\boldsymbol{\lambda}$, there is no need for the use of an iterative procedure. For a given set of initial conditions on the coordinates $\mathbf{q}$ and the velocities $\dot{\mathbf{q}}$, the Jacobian matrix of the kinematic constraint equations $\mathbf{C}_{\mathbf{q}}$ can be evaluated and used to define a set of independent and dependent coordinates $\mathbf{q}_{i}$ and $\mathbf{q}_{d}$, respectively [16, 17, 12]. Knowing the independent coordinates $\mathbf{q}_{i}$, the nonlinear algebraic constraint equations can be solved iteratively using a Newton-Raphson algorithm to determine the dependent coordinates using the equation

$$\begin{bmatrix} \mathbf{C}_{\mathbf{q}}^{l} \\ \mathbf{I}_{i} \end{bmatrix} \Delta\mathbf{q}^{l} = \begin{bmatrix} -\mathbf{C}^{l} \\ \mathbf{0} \end{bmatrix} \tag{5}$$

where superscript l refers to the Newton-iteration number, $\Delta\mathbf{q}$ are the Newton differences, and $\mathbf{I}_{i}$ is a Boolean matrix which has ones corresponding to the location of the independent coordinates to ensure $\Delta\mathbf{q}_{i} = \mathbf{0}$ and zeros in all other locations, that is $\mathbf{I}_{i}\mathbf{q} = \mathbf{q}_{i}$. Because the coefficient matrix in the preceding equation is sparse, sparse matrix techniques can be used during the Newton iterations. Equation 5 represents the inner loop of the TLISMNI algorithm used in this study. Once the dependent coordinates are determined and knowing the independent velocities, the dependent velocities can be calculated from the linear sparse matrix equation

$$\begin{bmatrix} \mathbf{C}_{\mathbf{q}} \\ \mathbf{I}_{i} \end{bmatrix} \dot{\mathbf{q}} = \begin{bmatrix} -\mathbf{C}_{t} \\ \dot{\mathbf{q}}_{i} \end{bmatrix} \tag{6}$$

Knowing the coordinates and velocities, Eq. 4 can be constructed and used to determine the vector of unknown accelerations and Lagrange multipliers.

During the process of the numerical solution, the integration formula used defines the vector $\mathbf{y}_{n+1} = \left[(\mathbf{q}_i)_{n+1}^T \quad (\dot{\mathbf{q}}_i)_{n+1}^T \right]^T$. This vector is used in the equations described in this section to evaluate the vector $\dot{\mathbf{y}}_{n+1} = \left[(\dot{\mathbf{q}}_i)_{n+1}^T \quad (\ddot{\mathbf{q}}_i)_{n+1}^T \right]^T$ that appear in the implicit corrector integration formula. Using the equations presented in this and preceding sections, the steps of the TLISMNI/Adams algorithm can be outlined as described in the following section.

4. TLISMNI/ADAMS ALGORITHM

Using the equations described in the preceding two sections, a TLISMNI/Adams algorithm can be developed. The steps of this algorithm can be summarized, using the implicit 3rd order Adams method as an example, as follows:

1. Given the initial coordinates and velocities $\mathbf{q}$ and $\dot{\mathbf{q}}$, respectively, the constraint Jacobian matrix $\mathbf{C}_\mathbf{q}$ is evaluated and used with a Gaussian elimination method to identify the independent coordinates (degrees of freedom) $\mathbf{q}_i$ and independent velocities $\dot{\mathbf{q}}_i$. The state vector $\mathbf{y}_n = \left[(\mathbf{q}_i)_n^T \quad (\dot{\mathbf{q}}_i)_n^T \right]^T$ with $n = 0$, in the case of the initial conditions, is used as input to the numerical integration routine.
2. At the initial time, a single-step Runge-Kutta method is used in the numerical integration with very small time step in order to determine history information required by the multistep predictor and corrector formulae used in this investigation.

3. Having the information needed from previous time steps, the 2nd order explicit predictor Adams-Bashforth formula $(\mathbf{y}_{n+1})_{pr} = \mathbf{y}_n + (h/2)(3\dot{\mathbf{y}}_n - \dot{\mathbf{y}}_{n-1})$ is used to predict the solution $\mathbf{y}_{n+1} = \left[(\mathbf{q}_i)_{n+1}^T \quad (\dot{\mathbf{q}}_i)_{n+1}^T\right]^T$. Using this predicted solution, the equations of motion summarized in the preceding section are used to evaluate $\dot{\mathbf{y}}_{n+1} = \left[(\dot{\mathbf{q}}_i)_{n+1}^T \quad (\ddot{\mathbf{q}}_i)_{n+1}^T\right]^T$.
4. The Adams-Moulton corrector formula $(\mathbf{y}_{n+1})_{cr} = \mathbf{y}_n + (h/12)(5\dot{\mathbf{y}}_{n+1} + 8\dot{\mathbf{y}}_n - \dot{\mathbf{y}}_{n-1})$ is used to evaluate the solution. If the error $|\mathbf{e}_{n+1}| = \left|(\mathbf{y}_{n+1})_{cr} - (\mathbf{y}_{n+1})_{pr}\right| \le \varepsilon$, where ε is a given error tolerance, then convergence is achieved and no other outer loop iterations are required. In this case, go to Step 6.
5. If $|\mathbf{e}_{n+1}| = \left|(\mathbf{y}_{n+1})_{cr} - (\mathbf{y}_{n+1})_{pr}\right| > \varepsilon$, $\dot{\mathbf{y}}_{n+1}$ obtained at the predictor step is denoted as $\dot{\mathbf{y}}_{n+1}^1$, and the TLISMNI outer loop iterations start to evaluate $(\mathbf{y}_{n+1})_{cr}^k = \mathbf{y}_n + (h/12)(5\dot{\mathbf{y}}_{n+1}^{k-1} + 8\dot{\mathbf{y}}_n - \dot{\mathbf{y}}_{n-1})$, where $\dot{\mathbf{y}}_{n+1}^{k-1}$ is evaluated using $(\mathbf{y}_{n+1})_{cr}^{k-1}$ for $k > 1$ using the TLISMNI inner loop and the motion and algebraic constraint equations summarized in the preceding section. During the outer loop iterations, the time step h is kept constant. Convergence is achieved if $|\mathbf{e}_{n+1}| = \left|(\mathbf{y}_{n+1})_{cr}^k - (\mathbf{y}_{n+1})_{cr}^{k-1}\right| \le \varepsilon$. If convergence criteria is not satisfied after specified number of TLISMNI outer loop iterations n_{it}, go to Step 7, otherwise go to Step 6.
6. If convergence is achieved, check whether or not the $t \ge t_f$, where t_f is the specified end of simulation time. If $t \ge t_f$, then the simulation is stopped, otherwise, the error is used to

determine whether or not the step size h can be increased according to the criterion described in the following section. If $t < t_f$, update the solution history and go to Step 3.

7. If convergence is not achieved using the time step h, the time step is reduced according to the criterion described in the following section. Keeping the solution history the same, go to Step 3.

In this investigation, the same algorithm is used for the implicit 4th order Adams formula with the 3rd order Adams-Bashforth predictor, and the implicit trapezoidal corrector with the explicit midpoint predictor, as previously mentioned in this paper. In the three cases, the TLISMNI algorithm is implemented to take advantage of the sparsity of the matrices presented in the preceding section. Furthermore, in such an algorithm numerical force differentiation is not required and the constraint equations are satisfied at all levels. In the algorithm used in this investigation, the time step is halved if convergence is not achieved in the inner loop during the Newton-Raphson iterations used to ensure that the position constraint equations are satisfied.

5. ERROR AND TIME STEP SELECTION CRITERIA

In the TLISMNI/Adams algorithm developed in this investigation, the order of the integration formulae is not varied. A variable step size is used in order to improve the convergence characteristics and efficiency of the algorithm. During the outer loop iterations, however, the step size is kept constant until convergence is achieved or fails. If the outer loop iterations converge, the step size is kept the same or increased depending on the magnitude of the error. In this section, the 3rd order Adams-Moulton method is used in the discussion of the error and time step selection criteria. A similar procedure can be used with the implicit 4th order Adams formula. It is important, however, to point out that the use of other criteria for the error check and time step selection may be more appropriate for specific applications, and therefore, the criteria presented in this section should not be viewed as the only or most optimum ones for all MBS applications when a TLISMNI algorithm is used.

For the 3rd order implicit Adams-Moulton method used in this investigation, the truncation error is estimated as $T_{n+1} = (1/24) h^4 \left| d^4 \mathbf{y}_{n+1} / dt^4 \right|$ (Atkinson, 1978). Using this truncation error equation and the solution history, one can use backward differentiation formula to write

$$T_{n+1} = (1/24) h \left| \dot{\mathbf{y}}_{n+1} - 3\dot{\mathbf{y}}_n + 3\dot{\mathbf{y}}_{n-1} - \dot{\mathbf{y}}_{n-2} \right| \tag{7}$$

In the case of convergence, $\left|\mathbf{e}_{n+1}\right| = \left| (\mathbf{y}_{n+1})_{cr} - (\mathbf{y}_{n+1})_{pr} \right| \le \varepsilon$, and if $\left|\mathbf{e}_{n+1}\right| \le \alpha \times T_{n+1}$, where α is a coefficient that can include a safety factor, the new time step h can be increased using the equation

$$h = \frac{24\left|\mathbf{e}_{n+1}\right|}{\left|D_{4y}\right|} \tag{8}$$

This is with the assumption that $D_{4y} = \left| \dot{\mathbf{y}}_{n+1} - 3\dot{\mathbf{y}}_n + 3\dot{\mathbf{y}}_{n-1} - \dot{\mathbf{y}}_{n-2} \right| / \left(h^3/8 \right) > 0$. If $T_{n+1} > \left|\mathbf{e}_{n+1}\right| > \alpha \times T_{n+1}$, the time step can be kept the same for the next step. Atkinson (1978) proposed a time-step selection strategy for the trapezoidal method. Using an approach similar to the one proposed by Atkinson, the new step size can be selected using the equation $\left(h/h_o\right)^4 \left|\mathbf{e}_{n+1}\right| = \varepsilon h/2$, where h_o is the previous time step . Therefore, a more efficient method for selecting the new time step is $h = \left(\varepsilon h_o^4 / 2\left|\mathbf{e}_{n+1}\right| \right)^{1/3}$. In the algorithm used in this investigation, the time step is not allowed to increase more than five times, that is, h is selected such that $h \le 5h_o$. In order to avoid repeated calculation of D_{4y}, in the TLISMNI/Adams algorithm used in this investigation, the time step is selected according to

$$h = \text{Maximum}\left(5h_o, \left(\frac{\varepsilon h_o^4}{2|\mathbf{e}_{n+1}|}\right)^{1/3}\right) \tag{9}$$

On the other hand, if the outer iterations do not converge, $|\mathbf{e}_{n+1}| = \left|(\mathbf{y}_{n+1})_{cr} - (\mathbf{y}_{n+1})_{pr}\right| > \varepsilon$, and if $|\mathbf{e}_{n+1}| > T_{n+1}$, The new step size h can be decreased using Eq. 8, $h = 24|\mathbf{e}_{n+1}| / |D_{4y}|$, with the assumption that $D_{4y} > 0$. In order to ensure that the step size will be significantly reduced and avoid the calculation of D_{4y}, the following equation is used:

$$h = \text{Maximum}\left(\frac{h_o}{2}, \left(\frac{\varepsilon h_o^4}{2|\mathbf{e}_{n+1}|}\right)^{1/3}\right) \tag{10}$$

The initial time step can be selected using an approach similar to the one proposed by Atkinson [2] for the trapezoidal method. Following Atkinson's approach, the initial time step h_i can be selected according to $\varepsilon h_i/2 = h_i^4 |D_{4y}|/24$. This equation can be used to determine the new time step as $h_i = \left(12\varepsilon/|D_{4y}|\right)^{1/3}$. This method for determining the time step requires the evaluation of D_{4y} only once at the beginning of the simulation. The truncation error formulae for the 3rd order Adams-Bashforth predictor and the 4th order Adams-Moulton corrector formulae are given, respectively, as $T_{n+1} = (3/8)h^4 \left|d^4\mathbf{y}_{n+1}/dt^4\right|$ and $T_{n+1} = (19/720)h^5 \left|d^5\mathbf{y}_{n+1}/dt^5\right|$. Using these truncation errors, a procedure similar to the one described for the 3rd order Adams formula can be used [2].

A similar procedure is used in the case of the 2nd order trapezoidal method which has a different truncation error defined as $T_{n+1} = (1/12)h^3 \left|d^3\mathbf{y}_{n+1}/dt^3\right|$, which can be written using the solution at previous time points as

$$T_{n+1} = (1/3)\,h\left|\dot{\mathbf{y}}_{n+1} - 2\dot{\mathbf{y}}_n + \dot{\mathbf{y}}_{n-1}\right| \tag{11}$$

In the case of the convergence of the trapezoidal method, the new time step size is determined using the equation

$$h = \text{Maximum}\left(5h_o, \sqrt{\frac{\varepsilon h_o^3}{2\left|\mathbf{e}_{n+1}\right|}}\right) \tag{12}$$

In the case of divergence of the trapezoidal method, the new time step is evaluated according to

$$h = \text{Maximum}\left(\frac{h_o}{2}, \sqrt{\frac{\varepsilon h_o^3}{2\left|\mathbf{e}_{n+1}\right|}}\right) \tag{13}$$

The initial time step in the case of the trapezoidal method is evaluated as $h_i = \sqrt{6\varepsilon / \left|D_{3y}\right|}$, where $D_{3y} = \left|\dot{\mathbf{y}}_{n+1} - 2\dot{\mathbf{y}}_n + \dot{\mathbf{y}}_{n-1}\right| / \left(h^2/4\right)$ [2]. The trapezoidal method is implemented in the TLISMNI framework [1] in order to allow comparison with the new TLISMNI/Adams algorithm. The results of the new algorithm are also compared with the explicit predictor-corrector Adams method with variable-order and variable time step.

6. NUMERICAL RESULTS

In this section, examples are presented in order to evaluate the implementation of the TLISMNI/Adams algorithm proposed in this study. These examples include mass-spring systems and heavily constrained MBS vehicle model. The mass-spring systems will allow clearly investigating the effect of varying the problem stiffness. The heavily constrained MBS vehicle model, which includes a large number of differential and algebraic equations and relatively large number of bodies is designed first as a non-stiff model and is used to demonstrate the efficiency of the explicit methods and the less significant effects of the tolerances and the outer-loop iterations when the implicit integration methods are used.

6.1 Mass-Spring System

Figure 1 shows three different masses m_1, m_2 and m_3 connected by three springs which have stiffness coefficients k_1, k_2 and k_3, respectively. The three masses are assumed to have values $m_1 = m_2 = m_3 = 1\text{kg}$, while the stiffness are assumed to have different coefficients that produce slow, moderate, and fast oscillations. In this example, the stiffness coefficients are varied in the range 1.0×10^6 N/m - 1.0×10^{12} N/m. These values for the mass and stiffness coefficients produce different solutions that have frequencies in the range 1.0×10^3 rad/s - 1.0×10^6 rad/s. All the masses are assumed to have the same initial displacement of 0.01 m. In order to examine the implementation of the proposed TLISMNI/Adams algorithm, the system is modeled using a general three-dimensional MBS algorithm in which Euler parameters are used to describe the orientation of the masses. Each mass is connected to the ground by a prismatic joint, ensuring a single degree of freedom for each mass-spring system. Including the ground body, the system motion is described using 28 absolute Cartesian coordinates (7 coordinates for each body including the four Euler parameters), and 25 algebraic constraint equations that describe 4 Euler parameter constraint equations, 3 prismatic joints (15 equations), and 6 ground constraint equations for the ground body. In this example, the effect of the stick-slip friction is not considered. To evaluate the effect of the change of the error tolerance, the number of outer-loop iterations, and the initial time step, a reference model is first used. In the reference model, all the springs are assumed to have the same stiffness coefficients, that is, $k_1 = k_2 = k_3 = 1.0\times10^6$ N/m, the relative error tolerance is assumed 10^{-6}, and the initial time step is assumed 10^{-4} s. The simulation time is assumed to be 2

s. As shown in Fig. 2, the reference model produces a solution that agrees well with the analytical solution of the problem when all integration methods considered in this investigation are used. The numerical simulations also showed that all integration methods have comparable CPU times in the case of the reference model with low stiffness coefficient for all springs. While simulations were performed for two complete seconds, in order to show clearly the solution details and avoid obfuscated figures, some of the results are presented for only 0.2 s. For the smaller time window, no changes in the accuracy of the solution are observed over the longer simulation period.

Relative Error Tolerance The computational efficiency of the integration methods is tested by changing the relative error tolerance while other parameters are fixed. Table 1 shows that the CPU times of all methods are comparable in the case of loose error tolerance, while they differ when the tolerance is tightened. In this table, ε_a refers to the absolute error tolerance which is used only for the explicit Adams method, ε_r is the relative error tolerance, k is the stiffness of all springs, h_0 is the initial time step, and n_{it} is the maximum number of the outer-loop iterations allowed. No outer-loop iterations are used for the explicit Adams, maximum of three outer-loop iterations are used for the implicit Adams, and maximum of four outer-loop iterations are used for the implicit trapezoidal. The CPU time in this table is measured with respect to the 4th order implicit Adams method. The results of Table 1 show that the 4th order Adams method is more efficient than the other two TLISMNI methods, and the explicit Adams method remains the most efficient method for this non-stiff system. Figures 3 and 4 show the solutions obtained for the two cases of the relative error tolerance presented in Table 1. Since all masses have the same solutions, the results in these figures are presented only for the displacement of the first mass. As clearly shown by the results of these figures, when a loose error tolerance is used, the implicit methods damp out the solution. This damping effect is significantly reduced as the error tolerance is tightened, as it is clear from the two cases of $\varepsilon_r = 10^{-4}$ (Fig. 3), and $\varepsilon_r = 10^{-7}$ (Fig. 4). The ability of the implicit method to damp out solutions can be advantageous when it is desired in some applications to filter out high-frequency and less-significant signals.

Initial Time Step Table 2 compares the efficiency of different integration methods when the initial time step is changed. As previously mentioned, all the methods used in this investigation have a variable time step that is selected according to the error in the solution. The results presented in Table 2 show that the 4^{th} order implicit Adams method has the same degree of efficiency as the explicit Adams method and both of these methods are more efficient compared to the implicit 3^{rd} order Adams and trapezoidal methods. It was also observed that reducing the time step leads to a reduction of the maximum number of the outer-loop iterations required by the 4^{th} order Adams method. When using $h_0 = 10^{-5}$, the maximum number of outer-loop iterations required by the 4^{th} order TLISMNI Adams method reduces to only two iterations. Because in the implementation used in this investigation the reduction in the time step is controlled to be a percentage of the previous time step, the initial time step proposed by the user can have an effect on the efficiency of the method as shown by the results presented in Table 2. In the algorithm used in this study, the time step is halved if the inner-loop that ensures that the algebraic constraint equations are satisfied at the position level, fails to converge.

Number of the Outer-Loop Iterations If the maximum number of outer-loop iterations n_{it} is set to one, the implicit method has only the inner-loop iterations used to ensure that the constraint equations are satisfied at the position level. In this case, no iterations are used with the integration corrector formula. If convergence is not achieved, the time step is reduced and the predictor formula is used again. When the outer-loop iterations are performed, the predictor formula is not used, and the corrector formula previous iteration results are used. Table 3 shows the effect of changing the maximum number of the outer-loop iterations. The results presented in this table show that when n_{it} is equal to one, the trapezoidal method fails for the given numerical parameters selected. As the maximum number of outer-loop iterations increases for this non-stiff system, the 4th order Adams method becomes as efficient as the explicit Adams method which has a variable order [15]. Increasing n_{it} beyond a certain number does not contribute to improvement in the efficiency of the methods since the methods converge with a number of iterations less than the user-specified n_{it}. The numerical simulations also show that if very low n_{it} is used, the time step of the implicit methods is reduced, making the method less efficient. Therefore, it is important to select the proper number of maximum outer-loop iterations. Such a number, however, depends on the problem being investigated and the frequency contents in the solution.

Stiffness Coefficients To evaluate the performance of the implicit integrators in the case of stiff systems, the stiffness coefficients of the springs are assumed to have values of $k_1 = 1.0\times10^6$ N/m, $k_2 = 1.0\times10^8$ N/m, and $k_3 = 1.0\times10^{12}$ N/m. The solution of the displacement of the first mass remains the same as previously reported. Figures 5 and 6 show the displacements of the second and third mass, when all other parameters used for the reference model, are kept the same ($h_0 = 10^{-4}$, $\varepsilon_a = 10^{-6}$, and $\varepsilon_r = 10^{-6}$). The results presented in Figs. 5 and 6 clearly show how the implicit integration methods damp out the solution in the case of stiff problems. For the moderate stiffness, as in the case of the second mass shown in Fig. 5, all the implicit integration methods considered in this investigation have comparable effect on filtering out higher frequencies. Nonetheless, the difference in damping effect becomes significant in the case of high stiffness as evident by the results of the third mass. The results presented in Fig. 6 also show that solutions can be obtained using the implicit integrators, while the explicit Adams method fails in the case of a highly stiff system. In the case of this stiff system, Tables 4 - 6 shows the CPU times when ε_r is varied, h_0 is varied, and n_{it} is varied, respectively. Figures 7 – 12 show the results of the displacements of the second and third masses that correspond to the three cases of $h_0 = 10^{-4}$, $\varepsilon_a = 10^{-6}$, and $\varepsilon_r = 10^{-4}$; $h_0 = 10^{-4}$, $\varepsilon_a = 10^{-6}$, and $\varepsilon_r = 10^{-7}$; and $h_0 = 10^{-5}$, $\varepsilon_a = 10^{-6}$, and $\varepsilon_r = 10^{-6}$. All the implicit methods solutions can converge, while the explicit method can fail if more accuracy is demanded by tightening the relative error. Nonetheless, a good agreement between the solutions obtained the explicit Adams method and the 4th order implicit Adams is found whenever the explicit Adams method converges. The results presented in these figures clearly demonstrate the ability of the implicit integration methods to damp out high frequency oscillations when proper numerical parameters are selected. These methods, as expected, can also be much more efficient as compared to the explicit methods which are not suited for solving stiff problems. In some stiff problems, as shown in this numerical study, the explicit Adams fails to obtain the solution for the entire simulation. For example, the explicit Adams method fails in the case of the numerical parameters $h_0 = 10^{-4}$, $\varepsilon_a = 10^{-6}$, and $\varepsilon_r = 10^{-7}$, as shown in Table 4.

6.2 HMMWV Vehicle Model

In order to further test the performance of the TLISMNI/Adams integration method in the case of a complex MBS DAE's system, a heavily constrained vehicle model is considered. The model considered in this section is the **h**igh-**m**obility **m**ulti-purpose **w**heeled **v**ehicle (HMMWV) which is designed for off-road operations [4]. The vehicle model, which is shown in Fig. 13 and has front and rear suspensions, tires, and steering system, consists of 50 bodies connected by different types of joints and bushings elements. Table 7 shows the vehicle inertia and geometric properties, while Table 8 shows the types of joints used and the number of nonlinear algebraic constraint equations resulting from the use of each joint type. The model has 350 absolute coordinates because of the use of Euler parameters as orientation coordinates, 320 nonlinear algebraic constraint equations, and 30 degrees of freedom. The algebraic constraint equations include 6 ground, 50 Euler-parameters, 81 spherical joint, 85 revolute joint, 42 rigid joint, 36 cylindrical joint, 1 relative angular velocity, 11 gear, 1 relative coordinate projection, 2 rack and pinion, and 5 dot-product constraint equations. The front and rear suspension springs are assumed to have stiffness coefficients of 1.67071×10^{5} N/m and 3.02619×10^{5} N/m respectively, respectively, and undeformed lengths of 0.33934m and 0.31572m, respectively. Figures 14 – 15 show the nonlinear damping force-velocity relationships of the suspension dashpots. In addition to the suspension elements, the vehicle model has 4 bushing elements. The bushing stiffness coefficients in three perpendicular directions are assumed to be 2.0×10^{6} N/m, 2.0×10^{6} N/m, and 2.0952×10^{6} N/m, the damping coefficients in these three directions are assumed to be 1.0×10^{3} N.s/m. Figures 16 – 18 show the nonlinear bushing torque-angular displacement relationships. The damping coefficients associated with these bushing rotational coordinates is assumed to be 5.7296×10^{-2} N.m.s/rad. The vehicle is assumed to have an initial velocity of 18 km/h and the simulation time is assumed to be for 10 s. Figure 19 shows the forward position of the center of mass of the HMMWV chassis as a function of time, while Figure 20 shows the vertical position of the same point. The results in Fig. 19 show the effect of friction which is a source of energy dissipation that results in reducing the vehicle forward velocity. Table 9 shows the effect of changing the relative error ε_r on the CPU time when considering the methods discussed in this investigation, while Table 10 shows the effect of changing the maximum number of the outer-loop iterations n_{it}. Because of the heavy chassis of this vehicle, the model frequency content is not very high, and consequently, the explicit Adams integrator becomes faster. It was found that by increasing the bushing stiffness coefficients to 2.0×10^{12} N/m, 2.0×10^{12} N/m, and 2.0952×10^{12} N/m, the implicit methods become on the average 85% faster than the explicit Adams method. For this model increasing the bushing damping coefficients did not improve the performance of the implicit methods since the bushing damping contributed to damping out the high frequency signals, and therefore, for this particular example, the performance of the explicit methods can improve as the result of damping out the high frequency signals. Numerical experimentation has also shown that there is no efficiency improvement gained by increasing the number of outer loop iterations beyond 3 when the implicit methods are used.

7. SUMMARY AND CONCLUSIONS

Most existing implicit numerical integration methods used for solving MBS application problems are not suited for the solution of very large systems; particularly those systems that include flexible bodies which may have a very large number of generalized coordinates. When flexible bodies with high stiffness are considered, numerical differentiation of the forces can be source of errors. Furthermore, some existing implicit integration methods do not exploit sparse matrix techniques required for the efficient solution of large scale MBS applications, and do not ensure that the kinematic constraint equations are satisfied at position, velocity, and acceleration levels. This paper addresses these computational challenges by developing a new TLISMNI/Adams algorithm in which the 3rd and 4th order implicit Adams-Moulton formulae are used as the correctors, respectively, with the explicit 2nd and 3rd order Adams-Bashforth formulae as predictor. The proposed methods, therefore, have a constant order, while the step size is allowed to vary in order to enhance the efficiency and convergence characteristics of the methods when used in the DAE's solution of heavily constrained dynamical systems. The variable-step size TLISMNI method proposed in this investigation avoids numerical differentiation of forces, ensures that the nonlinear algebraic constraint equations are satisfied at all levels, and exploits sparse matrix techniques for the efficient solution of the dynamical system equations. During the iterative outer loop iterations, the time step is not allowed to vary until convergence is achieved or fails. The inner loop is used to ensure the convergence of the iterative procedure used to satisfy the holonomic algebraic constraint equations. As discussed in this paper, the holonomic constraint equations at the velocity and acceleration levels are always linear and do not require an iterative procedure to solve for the dependent velocities and accelerations. The criteria used for the error check and step-size selection is described. The performance of the TLISMNI/Adams algorithm developed in this investigation is evaluated by comparison with the 2nd order implicit trapezoidal method and the variable-order explicit predictor-corrector Adams method [15]. The numerical results obtained in this investigation are used to shed light on the advantages and drawbacks of the implicit integrators when used in the analysis of constrained dynamical systems. Two examples that include mass-spring systems and heavily constrained MBS vehicle model are considered in this study. The mass-spring systems allowed to clearly investigate the effect of varying the problem stiffness, while the heavily constrained MBS vehicle model, which includes a large number of differential and algebraic equations and relatively large number of bodies is a non-stiff model and is used to demonstrate the efficiency of the explicit methods and the less significant effects of the tolerances and the outer-loop iterations when the implicit integration methods are used. The results obtained using the two examples considered in this investigation show that the TLISMNI methods proposed in this study can be more efficient for stiff systems because of their ability to damp out high-frequency oscillations. The use of the iterative procedure allows for damping out the high-frequency oscillations despite the fact that the proposed TLISMNI methods do not include numerical damping as in the case of the HHT method. The results obtained in this study also show that explicit integration methods, such as the explicit Adams method, can be more efficient in the case on non-stiff systems.

Stiff systems are characterized by widely-separated eigenvalue solutions, and not necessarily with high frequency content. A variable time-step integration method is appropriate in the case of sudden changes in response characteristics such as in the jump phenomenon of nonlinear systems, stiffening and/or softening effects, or stick-slip phenomena. All the methods used in the paper, including the explicit Adams method, are variable time-step methods. Future investigations will focus on examining the effectiveness of the proposed integration methods in the case of discontinuities, an important scenario which has not been addressed in this investigation which is mainly focused on the application of the proposed methods to stiff systems not subjected to discontinuities.

REFERENCE

1. Aboubakr, A.K., and Shabana, A.A., 2015, "Efficient and Robust Implementation of the TLISMNI Method", *Journal of Sound and Vibration*, Vol. 353, pp. 220 - 242.
2. Atkinson, K. E., 1978, *An Introduction to Numerical Analysis*, John Wiley and Sons, New York.
3. Brenan, K.E., Campbell, S.L., and Petzold, L.R., 1989, *Numerical Solution of Initial Value Problems in Differential Algebraic Equations*, North-Holland, New York.
4. Grujicic, M., Marvi, H., Arakere, G., Bell, W., and Haque, I., 2010, The Effect of up-Armoring of the High-Mobility Multi-Purpose Wheeled Vehicle (HMMWV) on the off-Road Vehicle Performance, *Multidiscipline Modeling in Materials and Structures,* 6(2), pp. 229–256.
5. Hussein, B.A., and Shabana, A.A., 2011, "Sparse Matrix Implicit Numerical Integration of the Stiff Differential/Algebraic Equations: Implementation", *Nonlinear Dynamics*, Vol. 65, pp. 369-382.
6. Huston, R.L., 1990, *Multibody Dynamics*, Butterworth-Heineman, Boston.
7. Magnus, K., 1978, *Dynamics of Multibody Systems*, Springer, Berlin.
8. Orlandea, N.V., 2008, "From Newtonian Dynamics to Sparse Tableaux Formulation and Multibody Dynamics", *IMechE Journal of Multibody Dynamics*, Vol. 222, pp. 301 – 314.
9. Orlandea, N., Chace, M.A., and Calahan, D.A., 1977, "A Sparsity Oriented Approach to Dynamic Analysis and Design of Mechanical Systems", *ASME Journal of Engineering for Industry*, Vol. 99, pp. 773 – 784.
10. Press W, Teukolsky S, Vetterling W, Flannery B., 1992, *Numerical Recipes in FORTRAN*, Cambridge University Press, New York.
11. Roberson, R.E., and Schwertassek, R., 1988, *Dynamics of Multibody Systems*, Springer, Berlin.

12. Shabana, A.A., 2010, *Computational Dynamics*, Wiley & Sons, Chichester, U.K.
13. Shabana, A.A., and Hussein, B.A., 2009, "A Two-Loop Sparse Matrix Numerical Integration Procedure for the Solution of Differential/Algebraic Equations: Application to Multibody Systems", *Sound and Vibration*, Vol. 327, pp. 557-563.
14. Schiehlen, W.O. (Ed), 1990, *Multibody System Handbook*, Springer-Verlag, Berlin.
15. Shampine, L., and Gordon, M., 1975, *Computer Solution of ODE: The Initial Value Problem*, Freeman, San Francisco.
16. Uicker, J.J., Ravani, B., and Sheth, P.N., 2013, *Matrix Methods in the Design Analysis of Mechanisms and Multibody Systems*, Cambridge University Press, New York.
17. Wehage, R.A., 1980, "Generalized Coordinate Partitioning in Dynamic Analysis of Mechanical Systems", Ph.D. Dissertation, University of Iowa, Iowa City.

Abbreviations

BDF	Backward differentiation formula
CPU	Central processing unit
DAE	Differential/algebraic equations
HHT	Hilber–Hughes–Taylor
HMMWV	High-mobility multi-purpose wheeled vehicle
MBS	Multibody system
TLISMNI	Two-loop implicit sparse matrix numerical integration

Nomenclature

$\mathbf{C}$	Vector of constraint functions
$\mathbf{C_q}$	Constraint Jacobian matrix
$\mathbf{C}_t$	Partial derivative of the constraint functions with respect to time
$\mathbf{e}$	Error vector
$\mathbf{f}$	Time derivative of the state vector
h	Time step size
h_0	Initial time step
k_i	Stiffness coefficient of spring i
$\mathbf{M}$	System mass matrix
m_i	Mass of mass i
n_{it}	Number of outer-loop iterations
$\mathbf{Q}$	Force vector

$\mathbf{Q}_d$	Constraint quadratic velocity vector
$\mathbf{q}$	Vector of system generalized coordinate
$\mathbf{q}_d$	Vector of dependent coordinates
$\mathbf{q}_i$	Vector of independent coordinates or degrees of freedom
$\dot{\mathbf{q}}$	Velocity vector
$\ddot{\mathbf{q}}$	Acceleration vector
T_k	Truncation error at step k
t	Time
t_f	End of the simulation time
$\mathbf{y}$	State Vector
ε	Error tolerance
ε_a	Absolute error
ε_r	Relative error
$\boldsymbol{\lambda}$	Vector of Lagrange multipliers

Table 1. Effect of the relative error tolerance on CPU time $\left(k = 10^6 \text{ N/m}, h_0 = 10^{-4}, \varepsilon_a = 10^{-6}\right)$

Method	$\varepsilon_r = 10^{-4}$	$\varepsilon_r = 10^{-7}$
Explicit Adams	1	0.67
Implicit Adams (2nd order)	1	1.34
Implicit Adams (4th order)	1	1
Implicit Trapezoidal	1.3	1.67

Table 2. Effect of initial time step on the CPU time $\left(k = 10^6 \text{ N/m}, \varepsilon_a = 10^{-6}, \varepsilon_r = 10^{-6}\right)$

Method	$h_0 = 10^{-3}$	$h_0 = 10^{-5}$
Explicit Adams	1	1
Implicit Adams (3rd order)	2	1.67
Implicit Adams (4th order)	1	1
Implicit Trapezoidal	2	1.67

Table 3. Effect of the maximum outer-loop iterations on the CPU time
$\left(k = 10^6 \text{ N/m}, \varepsilon_a = 10^{-6}, \varepsilon_r = 10^{-6}, h_0 = 10^{-4}\right)$

Method	$n_{it} = 1$	$n_{it} = 2$	$n_{it} = 3$	$n_{it} = 4$	$n_{it} = 5$	$n_{it} = 6$	$n_{it} = 7$
Explicit Adams	0.34	0.34	1	1	1	1	1
Implicit Adams (3rd order)	3.34	0.67	1.5	1.5	1.5	1.5	1.5
Implicit Adams (4th order)	1	1	1	1	1	1	1
Implicit Trapezoidal	Failed	1.85	5.5	2	2	2	2

Table 4. Effect of the relative error tolerance on CPU time $\left(h_0 = 10^{-4}, \varepsilon_a = 10^{-6}\right)$

Method	$\varepsilon_r = 10^{-4}$	$\varepsilon_r = 10^{-6}$	$\varepsilon_r = 10^{-7}$
Explicit Adams	2.2	1.54	Failed
Implicit Adams (3rd order)	0.75	1.85	2.53
Implicit Adams (4th order)	1	1	1
Implicit Trapezoidal	0.93	2.18	2.77

Table 5. Effect of initial time step on the CPU time $\left(\varepsilon_a = 10^{-6}, \varepsilon_r = 10^{-6}\right)$

Method	$t_0 = 10^{-3}$	$t_0 = 10^{-4}$	$t_0 = 10^{-5}$
Explicit Adams	Failed	1.23	2.46
Implicit Adams (3rd order)	2.66	1.90	1.47
Implicit Adams (4th order)	1	1	1
Implicit Trapezoidal	3.36	2.17	5.18

Table 6. Effect of the maximum outer-loop iterations on the CPU time $\left(\varepsilon_a = 10^{-6}, \varepsilon_r = 10^{-6}, h_0 = 10^{-4}\right)$

Method	$n_{it} = 1$	$n_{it} = 2$	$n_{it} = 3$	$n_{it} = 4$	$n_{it} = 5$	$n_{it} = 6$	$n_{it} = 7$
Explicit Adams	1	1	1	1	1	1	1
Implicit Adams (3rd order)	1	0.13	0.13	0.03	0.03	0.03	0.03
Implicit Adams (4th order)	1	0.98	0.12	0.06	0.04	0.04	0.04
Implicit Trapezoidal	Failed	Failed	1	0.29	0.16	0.16	0.16

Table 7. Vehicle inertia and geometric properties

Components	Mass (kg)	I_{xx} (kg.m^2)	I_{yy} (kg.m^2)	I_{zz} (kg.m^2)
Ground	1	1	1	1
Front sub-frame	50.000	1.0000×10^{-6}	1.0000×10^{-6}	1.0000×10^{-6}
Rear sub-frame	45.359	2.9264×10^{-4}	2.9264×10^{-4}	2.9264×10^{-4}
Chassis	2086.5	1.0785×10^{3}	2.9557×10^{3}	3.5702×10^{3}
Front left and right suspensions				
Upright	3.6382	4.0800×10^{-2}	4.2300×10^{-2}	8.3400×10^{-3}
Upper arm	5.4431	2.3300×10^{-2}	3.6100×10^{-2}	1.3200×10^{-2}
Lower arm	16.329	0.14688	0.23163	0.11852
Upper strut	5.0000	1.0000×10^{-6}	1.0000×10^{-6}	1.0000×10^{-6}
Lower strut	5.0000	1.0000×10^{-6}	1.0000×10^{-6}	1.0000×10^{-6}
Tire	68.039	1.1998	1.7558	1.1998
Tie rod	0.5545	5.7854×10^{-3}	5.7854×10^{-3}	1.7745×10^{-5}
Tri pot	1.9851	1.1019×10^{-3}	1.1019×10^{-3}	8.1390×10^{-4}
Drive shaft	4.2175	0.16599	0.16599	6.9283×10^{-4}
Spindle	1.1028	4.7790×10^{-4}	4.7790×10^{-4}	4.9628×10^{-4}
Rear left and right suspensions				
Upright	3.6382	4.0800×10^{-2}	4.2300×10^{-2}	8.3400×10^{-3}
Upper arm	5.9320	0.068400	0.091400	0.024000
Lower arm	16.287	0.29036	0.51811	0.23229
Upper strut	5.0000	1.0000×10^{-6}	1.0000×10^{-6}	1.0000×10^{-6}
Lower strut	5.0000	1.0000×10^{-6}	1.0000×10^{-6}	1.0000×10^{-6}
Tire	68.039	1.1998	1.7558	1.1998
Tie rod	2.0412	4.2200×10^{-2}	4.2200×10^{-2}	1.9000×10^{-4}
Tri pot	2.0307	1.1383×10^{-3}	1.1383×10^{-3}	8.4254×10^{-4}
Drive shaft	6.0495	0.24565	0.24565	1.4463×10^{-3}
Spindle	1.5046	7.7539×10^{-4}	7.7539×10^{-4}	9.2387×10^{-4}
Steering system				
Steering wheel	2.1500	2.5892×10^{-2}	2.5892×10^{-2}	5.1629×10^{-2}
Steering column	2.7985	6.4319×10^{-2}	6.4319×10^{-2}	3.0466×10^{-4}
Intermediate shaft	2.2535	3.2103×10^{-2}	3.2103×10^{-2}	2.4336×10^{-4}
Steering shaft	2.0182	2.3725×10^{-2}	2.3725×10^{-2}	2.2182×10^{-4}
Rack	3.4424	0.23089	0.23089	2.6961×10^{-4}
Pinion	0.5000	1.0000×10^{-6}	1.0000×10^{-6}	1.0000×10^{-6}

Table 8. Joints used in HMMWV Model

Joint type	Number of constraints
Generalized-Coordinate-Constraint	6
Euler parameter constraint equations	50
Spherical-Joints	81
Revolute-(Pin)-Joints	85
Rigid-(Bracket)-Joints	42
Cylindrical-Joints	36
Relative-Angular-Velocity-Constraint	1
Gear-Constraint-Element	11
Relative-Coordinate-Projection-Constraint	1
Rack-Pinion-Constraint	2
Dot-Product-Constraint	5

Table 9 Effect of the relative error on HMMWV CPU time $\left(t_0 = 10^{-3}, \varepsilon_a = 10^{-5}\right)$

Method	$\varepsilon_r = 10^{-5}$	$\varepsilon_r = 10^{-6}$	$\varepsilon_r = 10^{-7}$
Explicit Adams	1	1	1
Implicit Adams (3rd order)	1.31	1.31	1.31
Implicit Adams (4th order)	1.59	1.59	1.57
Implicit Trapezoidal	1.48	1.48	1.42

Table 10 Effect of the number of outer-loop iterations on the HMMWV CPU time $\left(\varepsilon_a = 10^{-5}, \varepsilon_r = 10^{-5}, t_0 = 10^{-3}\right)$

Method	$n_{it} = 1$	$n_{it} = 2$	$n_{it} = 3$	$n_{it} = 4$	$n_{it} = 5$
Explicit Adams	1	1	1	1	1
Implicit Adams (3rd order)	1	0.82	0.76	0.75	0.76
Implicit Adams (4th order)	1	0.71	0.52	0.52	0.52
Implicit Trapezoidal	1	0.92	0.76	0.77	0.77

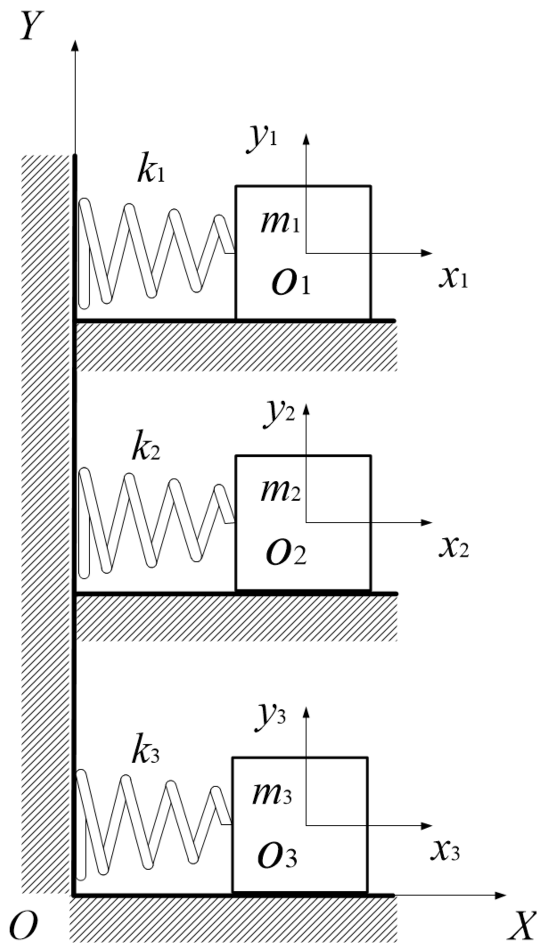
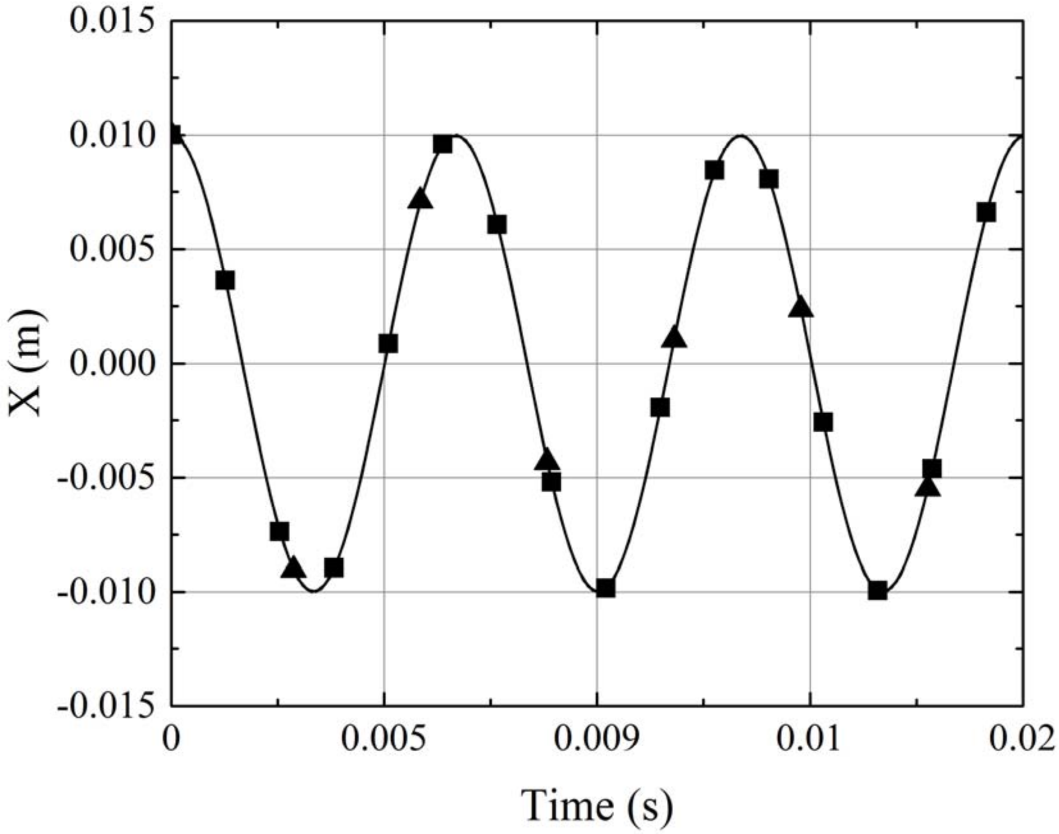
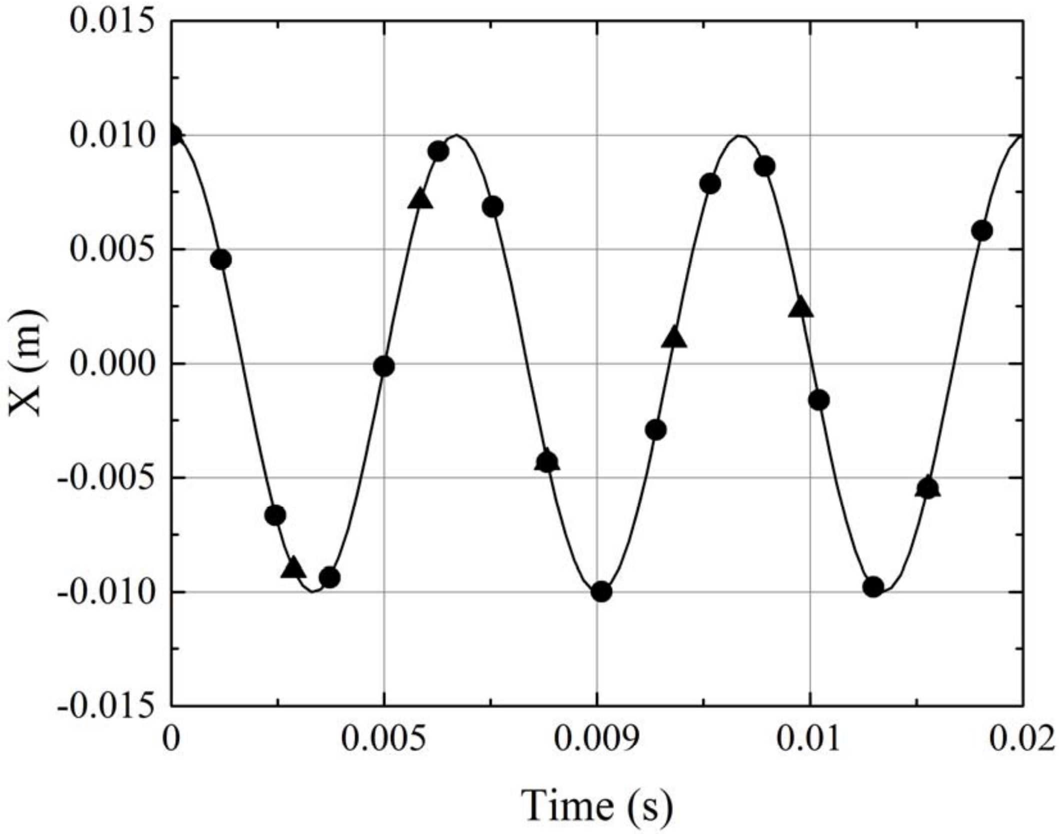


Figure 1. Mass Spring System Model



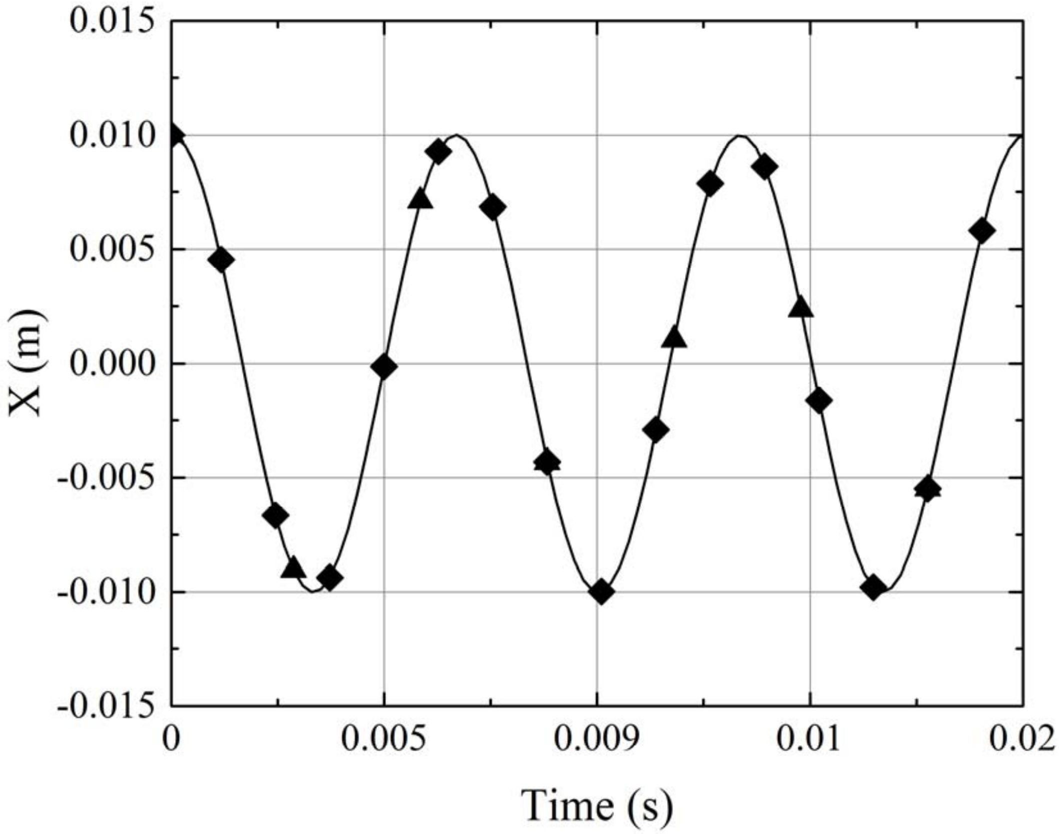
(a)

(Explicit Adams Method, 4^{th} Order Implicit Adams Method)



(b)

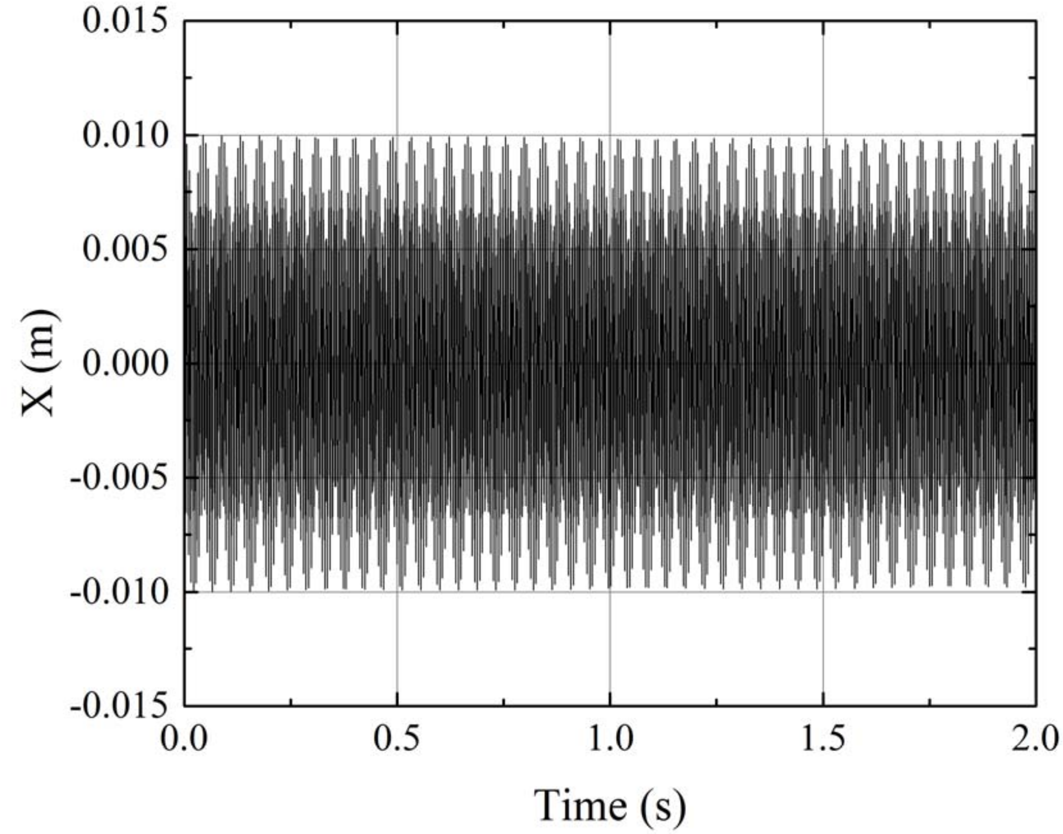
(—●— 3rd Order Implicit Adams Method, —▲— 4th Order Implicit Adams Method)



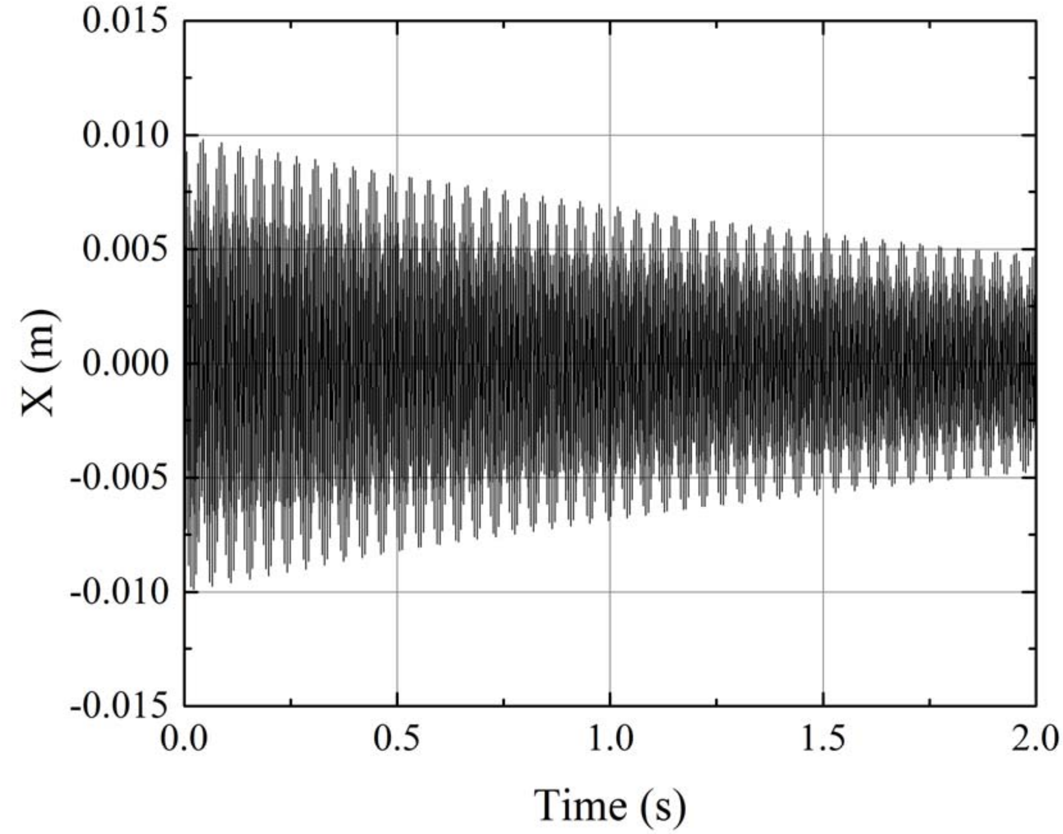
(c)

(Implicit Trapezoidal Method, 4^{th} Order Implicit Adams Method)

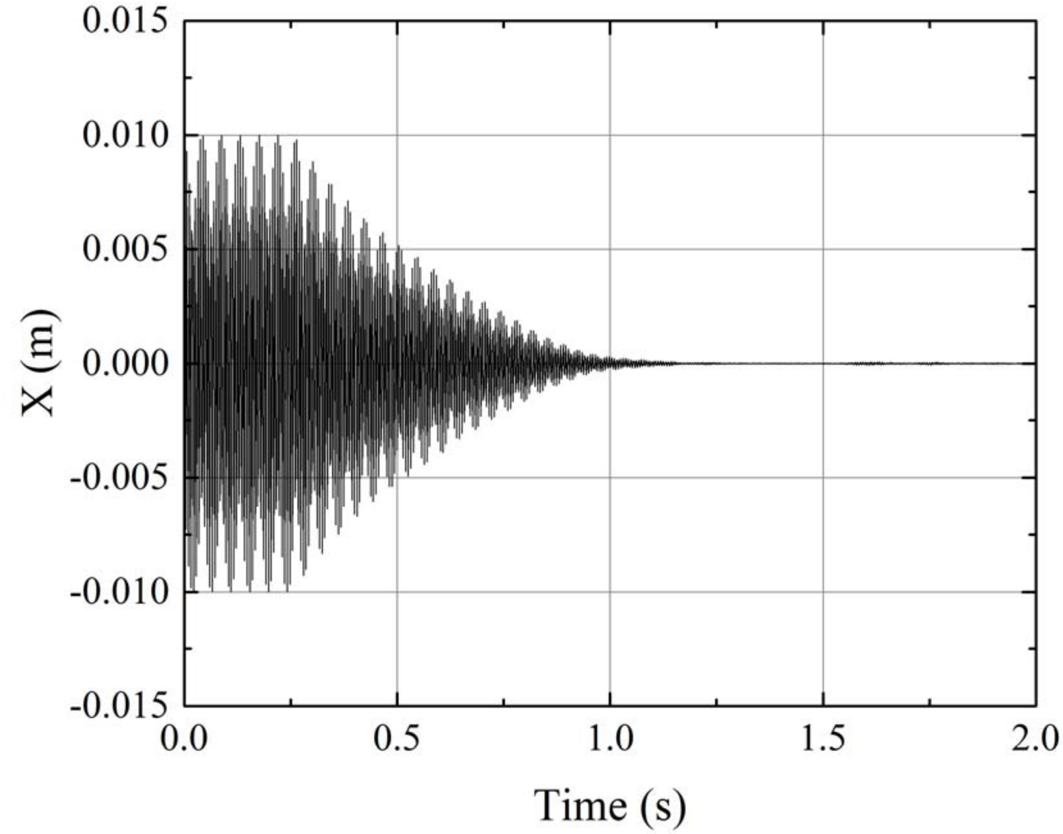
Figure 2. Coordinate of Mass 1 ($k_1 = k_2 = k_3 = 10^6$ N/m)



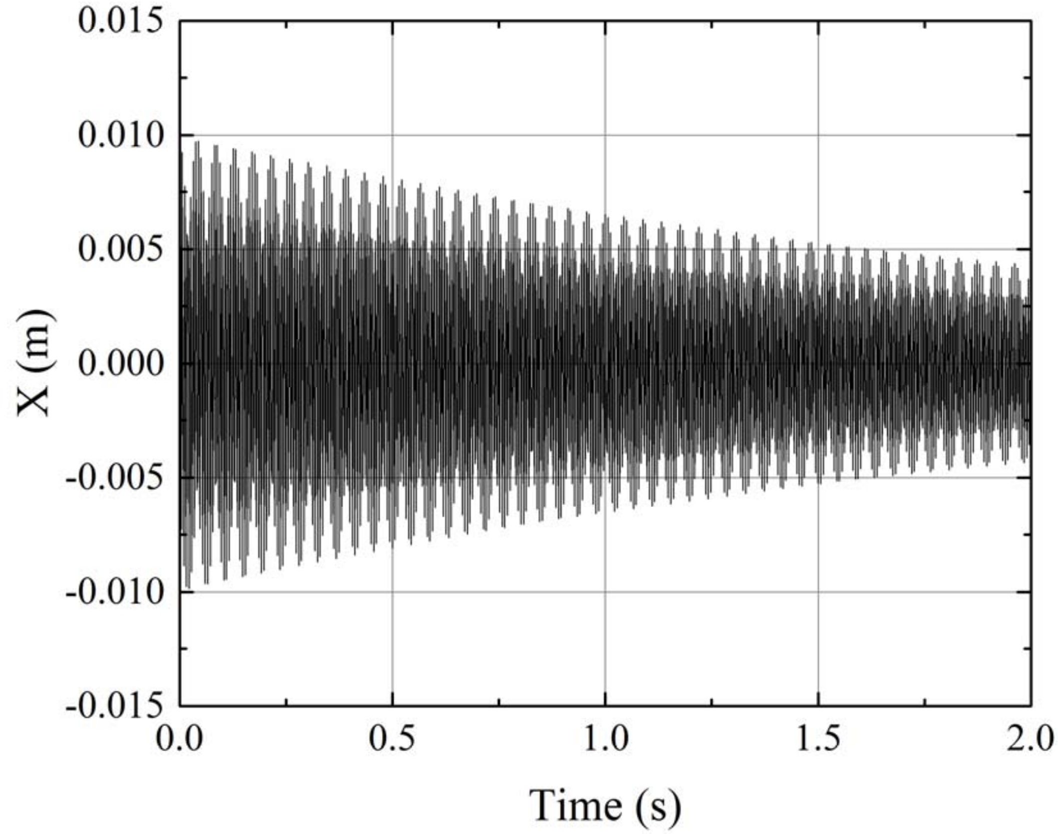
(a) Explicit Adams Method



(b) 3rd Order Implicit Adams Method

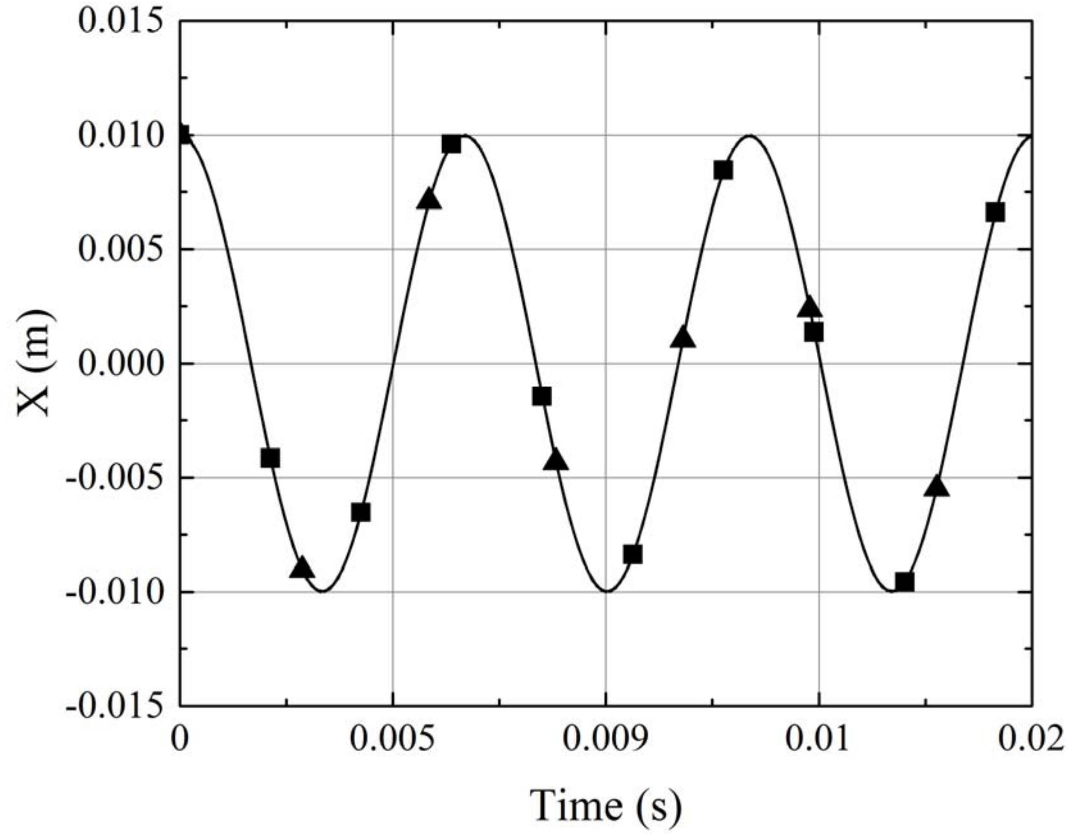


(c) 4^{th} Order Implicit Adams Method



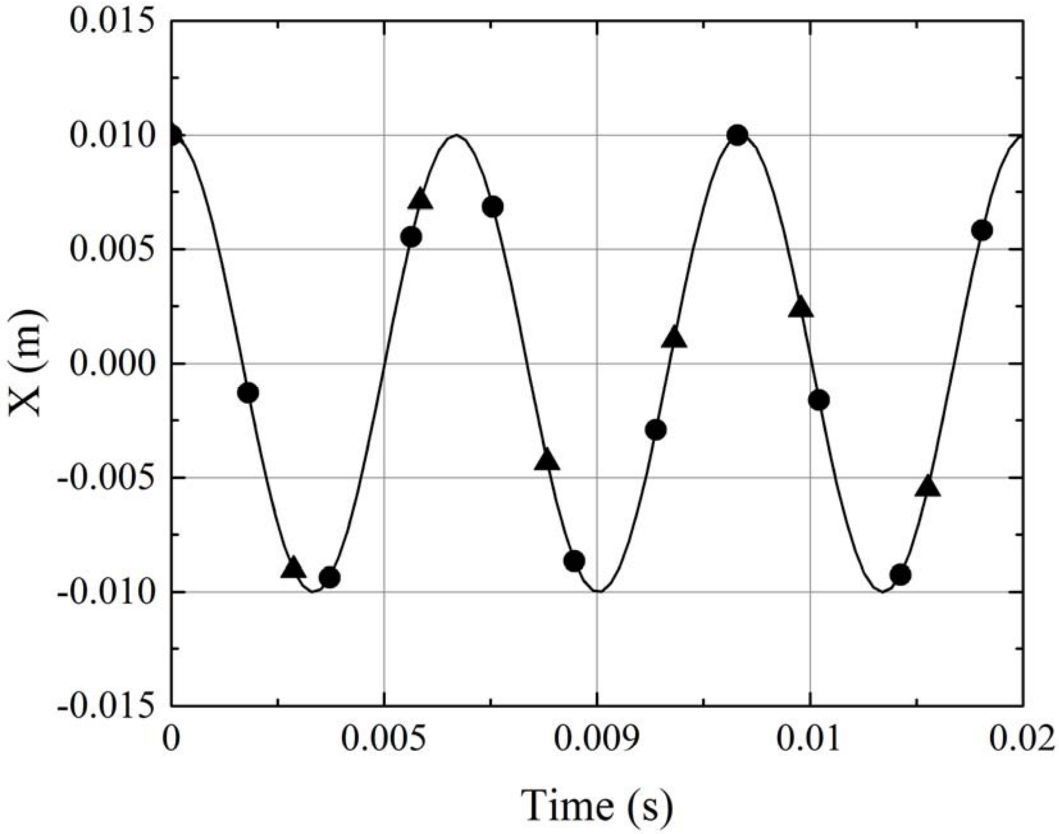
(d) Implicit Trapezoidal Method

Figure 3. Coordinate of Mass 1 ($\varepsilon_r = 10^{-4}$, $k_1 = k_2 = k_3 = 10^6$ N/m)



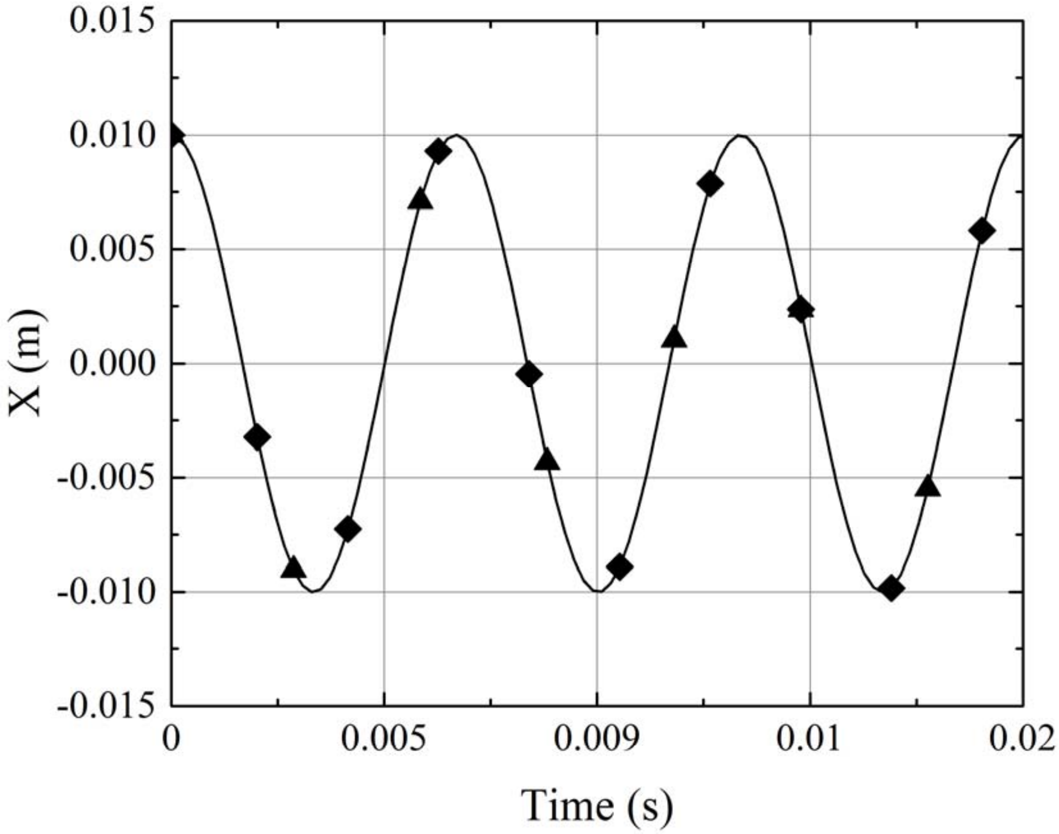
(a)

(—■—Explicit Adams Method, —▲—4^{th} Order Implicit Adams Method)



(b)

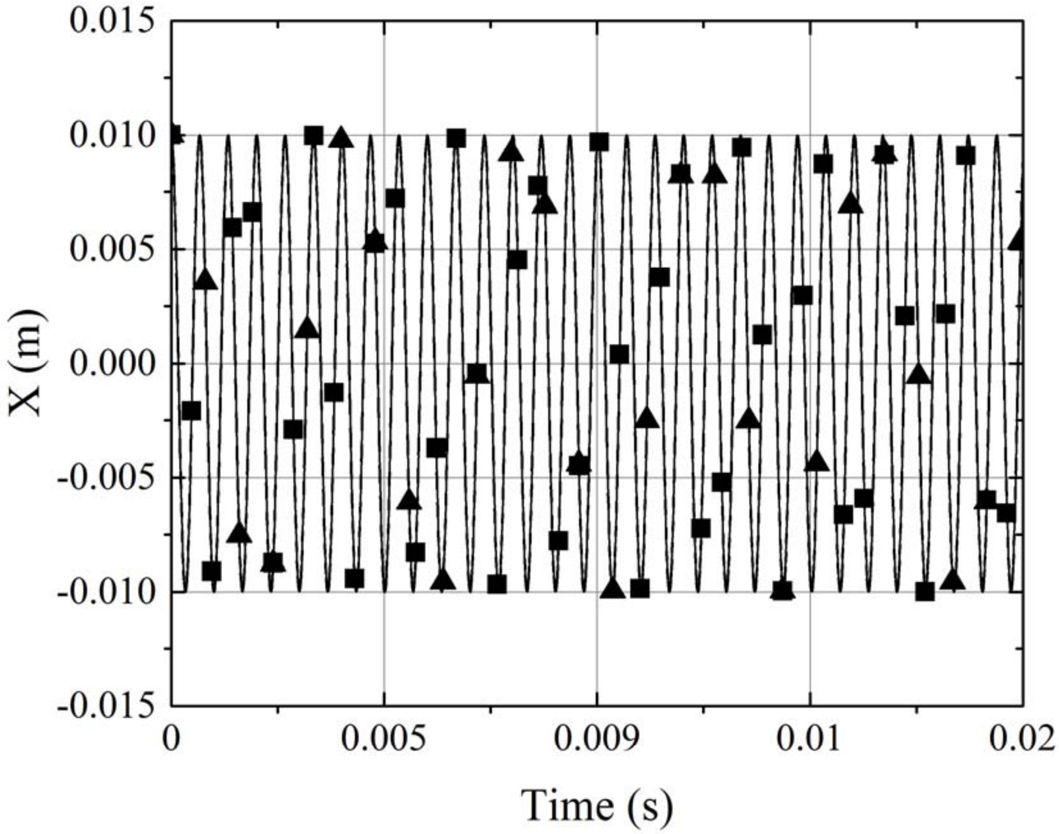
(3rd Order Implicit Adams Method, 4th Order Implicit Adams Method)



(c)

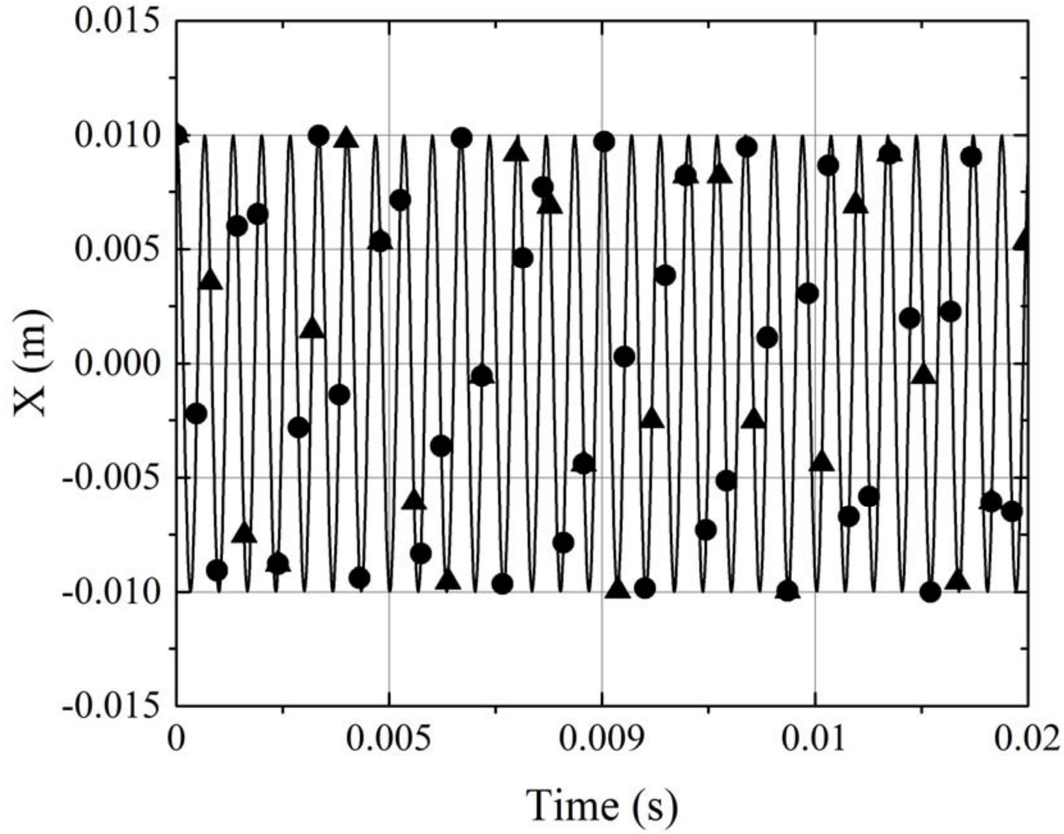
(Implicit Trapezoidal Method, 4th Order Implicit Adams Method)

Figure 4. Coordinate of Mass 1 ($\varepsilon_r = 10^{-7}$, $k_1 = k_2 = k_3 = 10^6$ N/m)



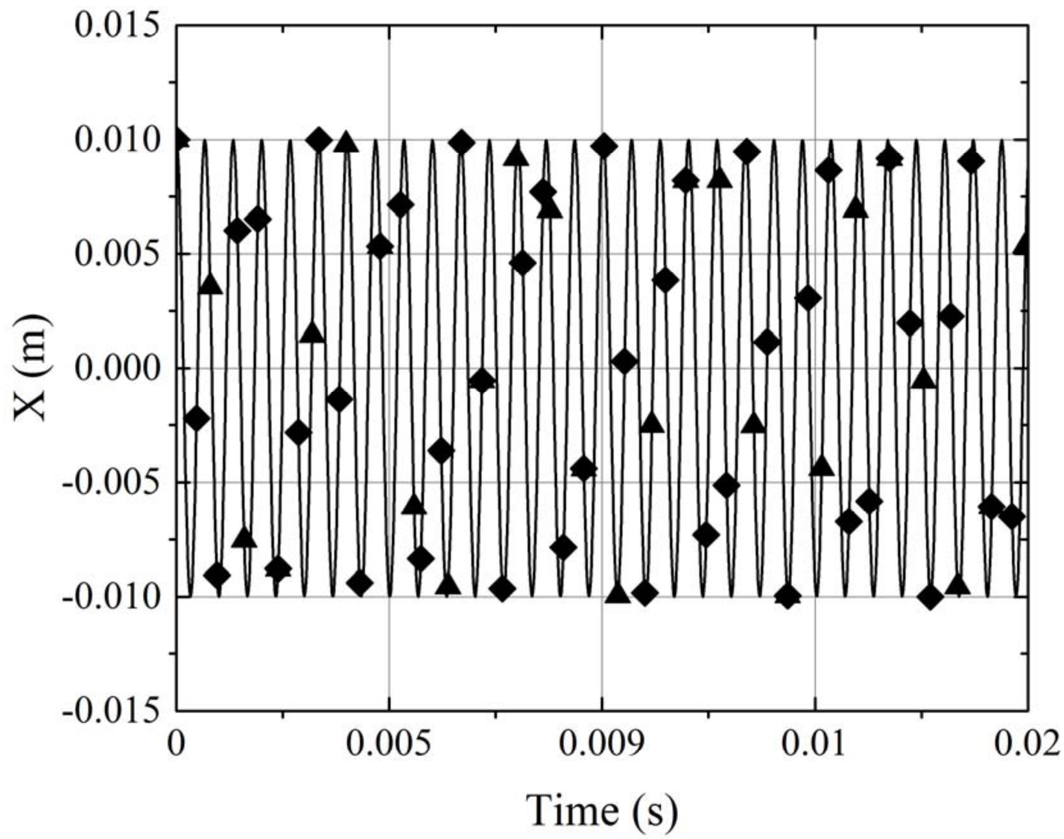
(a)

(Explicit Adams Method, 4th Order Implicit Adams Method)



(b)

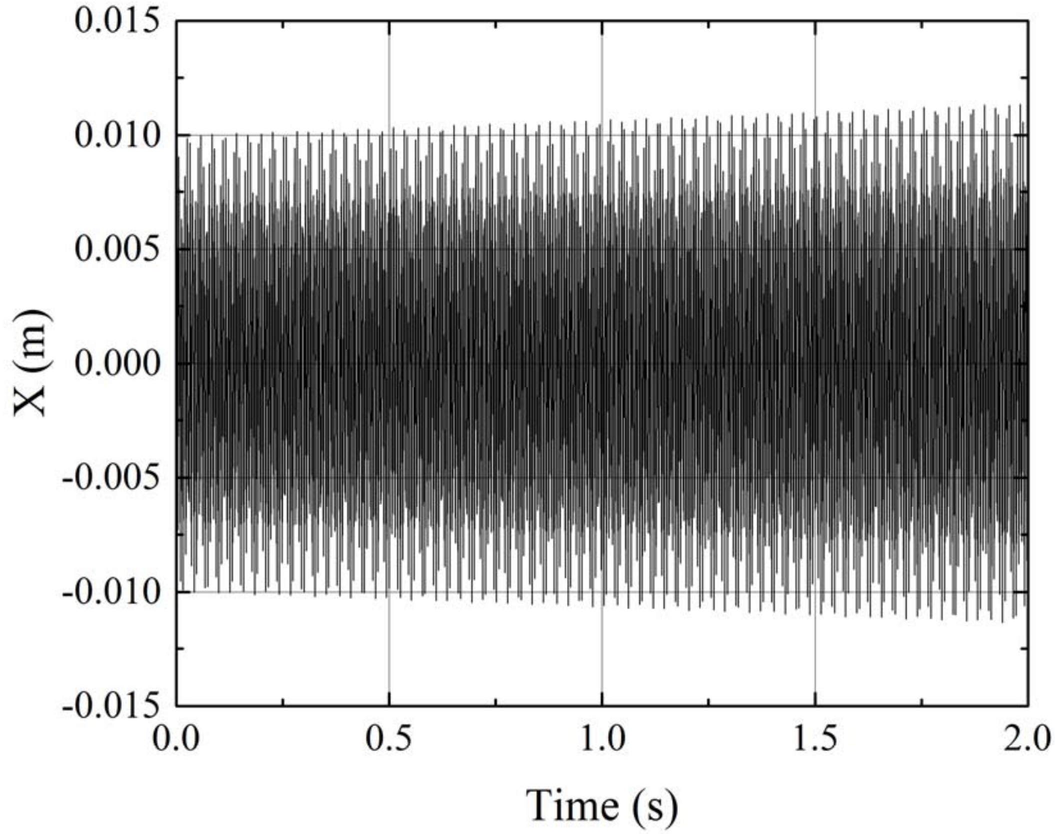
(3rd Order Implicit Adams Method, 4th Order Implicit Adams Method)



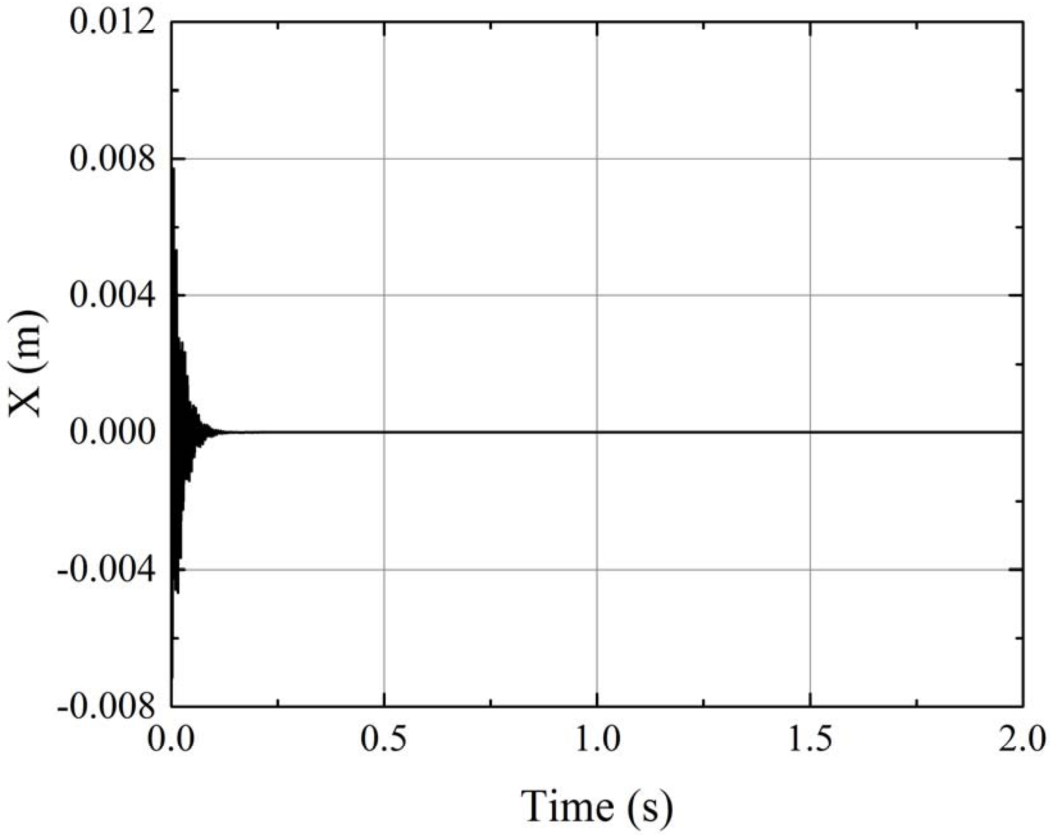
(c)

(Implicit Trapezoidal Method, 4th Order Implicit Adams Method)

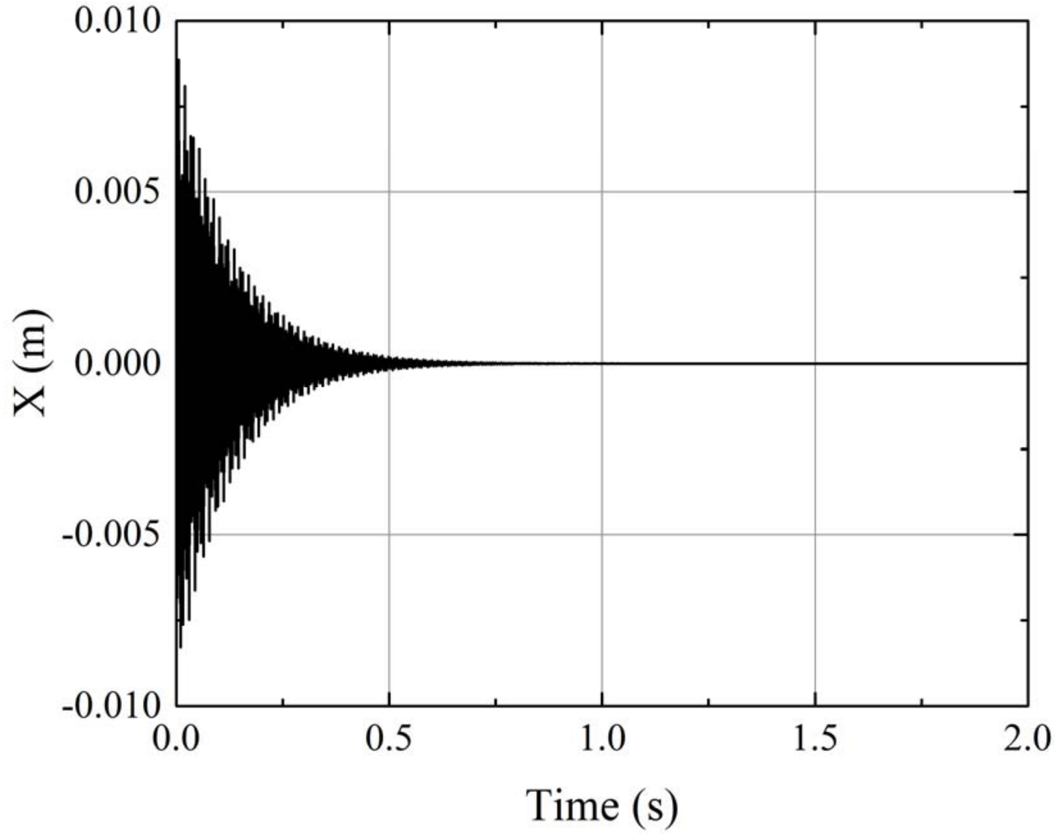
Figure 5. Coordinate of Mass 2 ($k_2 = 10^8$ N/m)



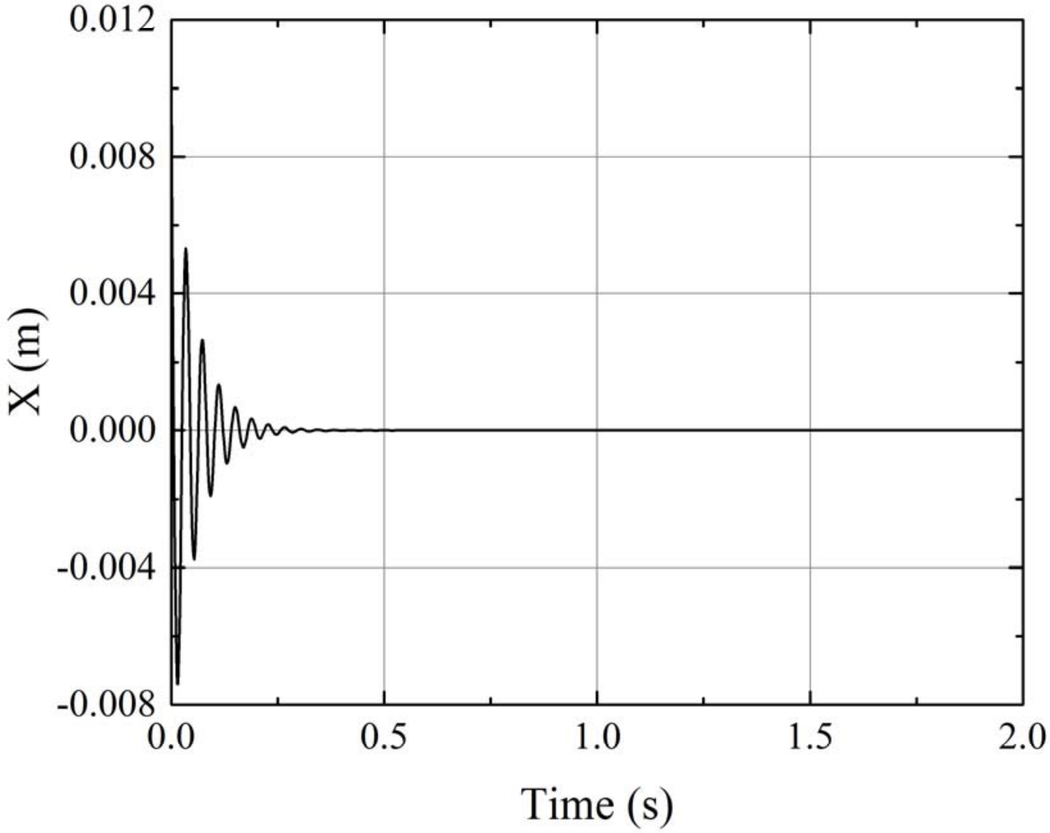
(a) Explicit Adams Method



(b) 3rd Order Implicit Adams Method

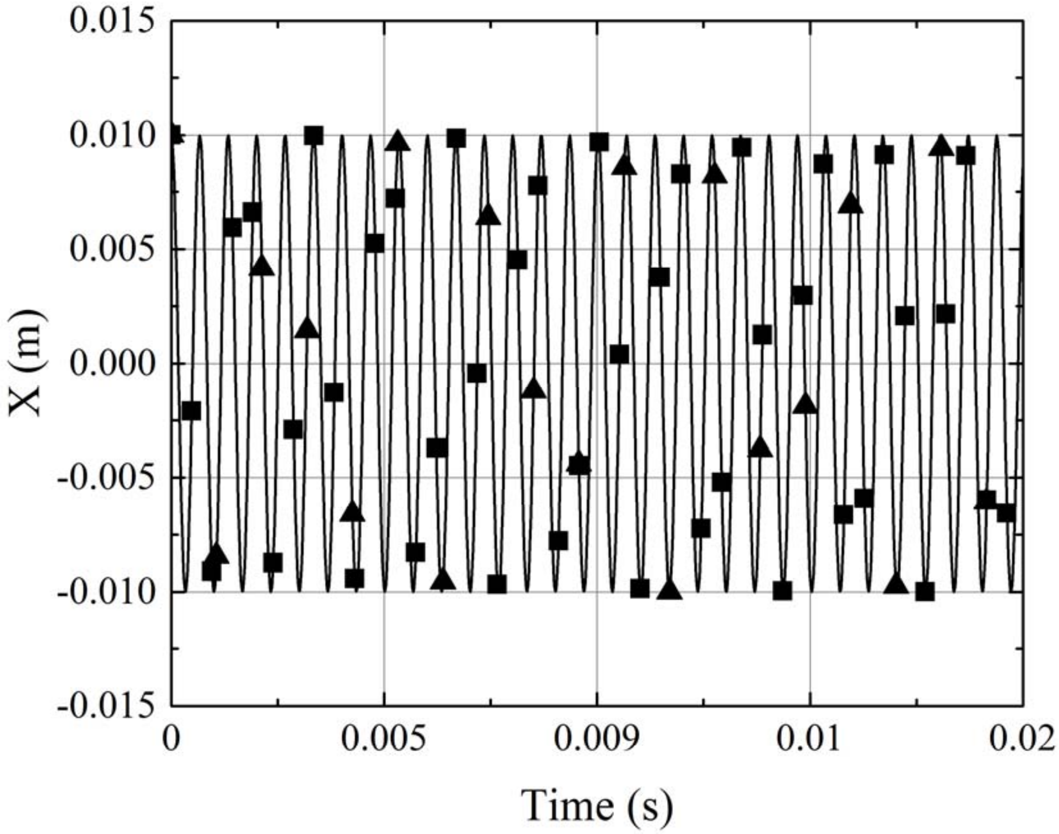


(c) 4th Order Implicit Adams Method



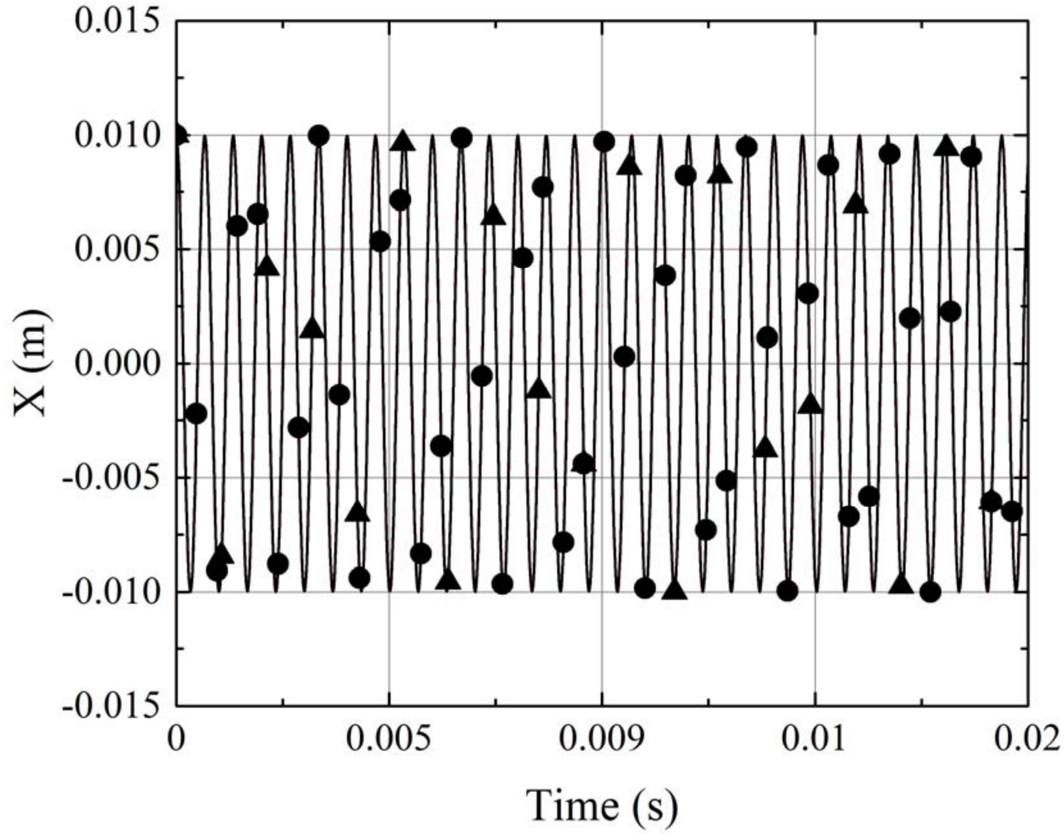
(d) Implicit Trapezoidal Method

Figure 6. Coordinate of Mass 3 ($k_3 = 10^{12}$ N/m)



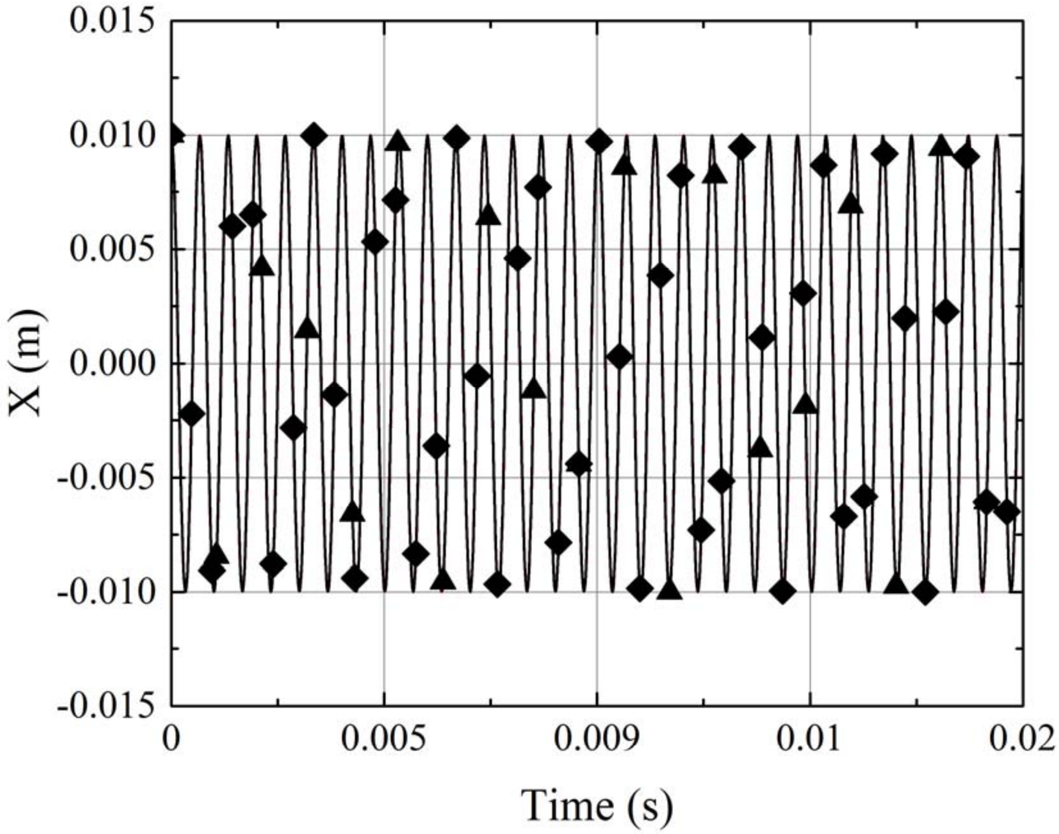
(a)

(—■—Explicit Adams Method, —▲— 4th Order Implicit Adams Method)



(b)

(3rd Order Implicit Adams Method, 4th Order Implicit Adams Method)



(c)

(Implicit Trapezoidal Method, 4th Order Implicit Adams Method)

Figure 7. Coordinate of Mass 2 ($\varepsilon_r = 10^{-4}$, $k_2 = 10^8$ N/m)

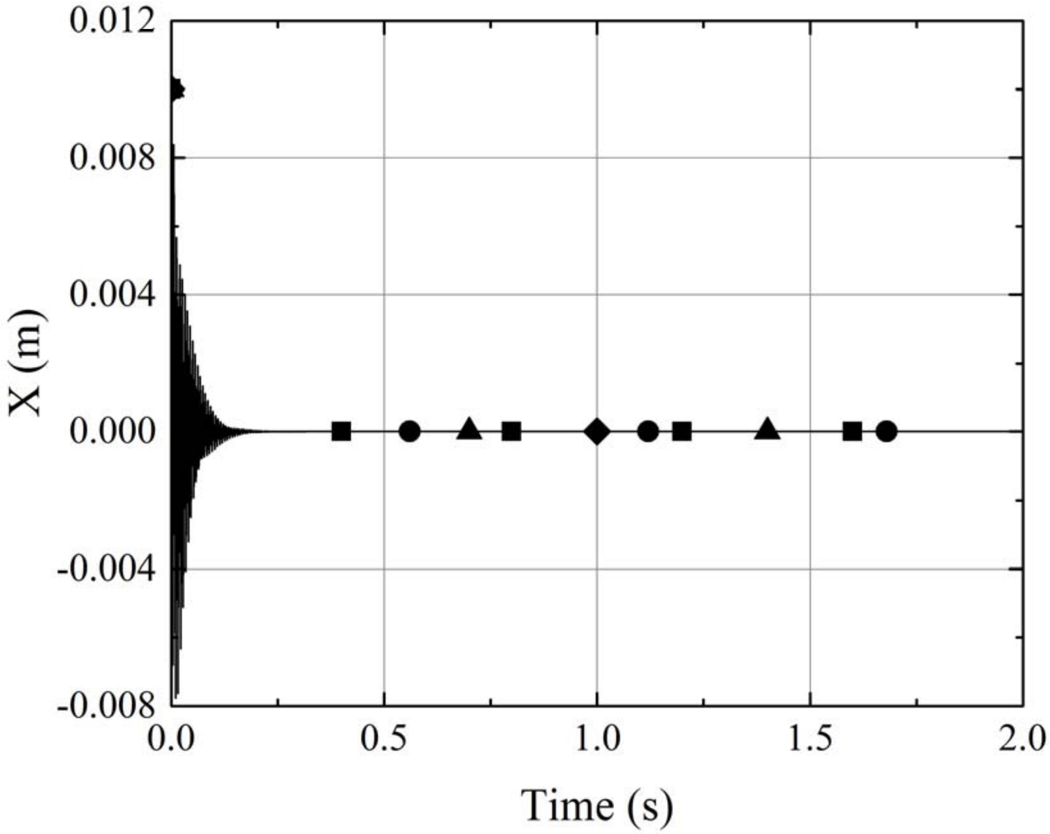
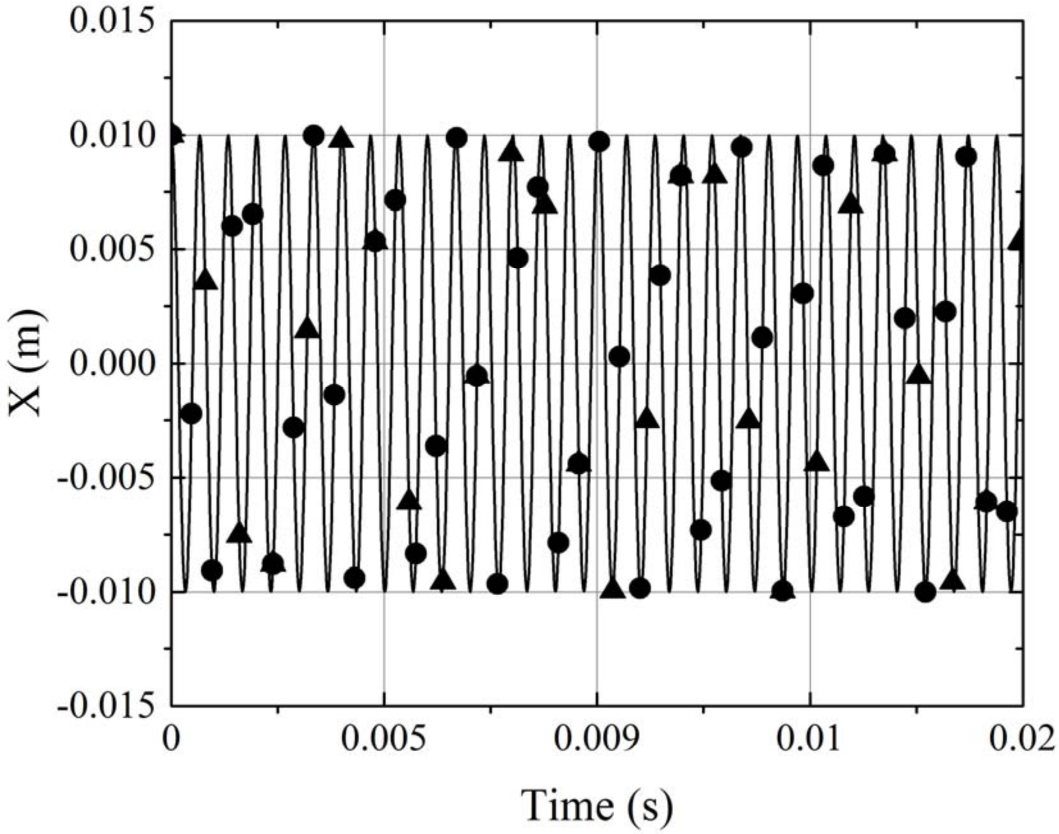


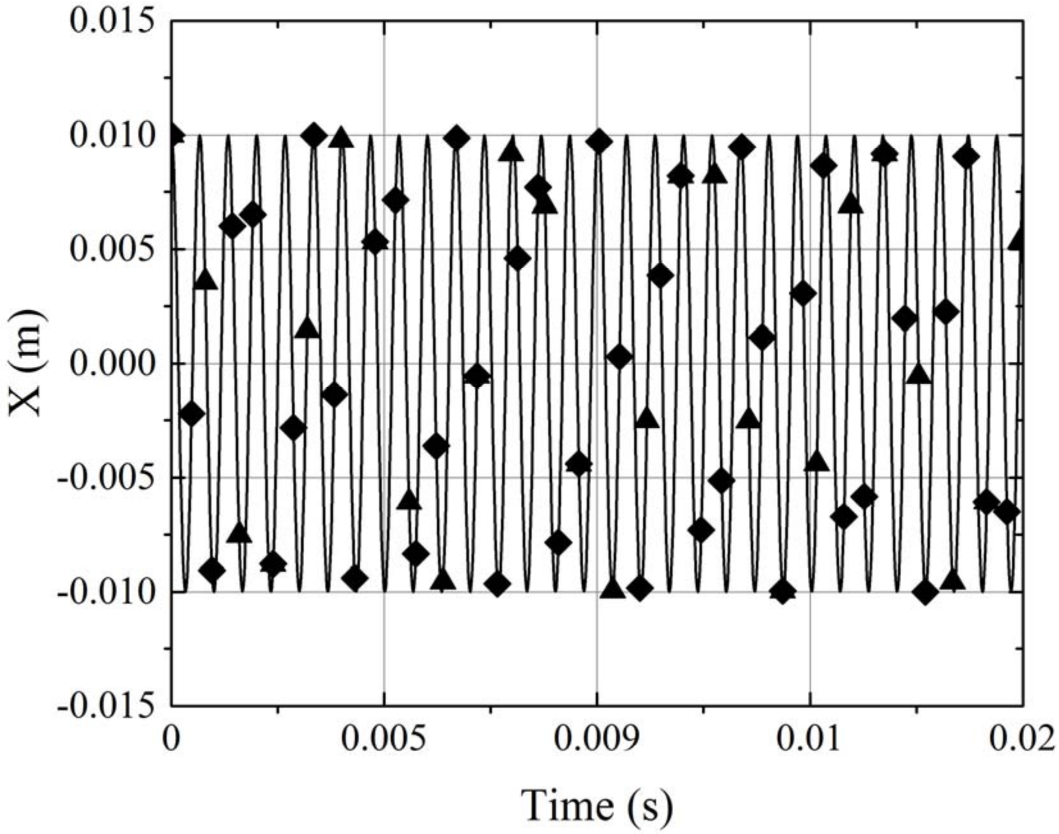
Figure 8. Coordinate of Mass 3 ($\varepsilon_r = 10^{-4}$, $k_3 = 10^{12}$ N/m)

(■ Explicit Adams Method, ● 3rd Order Implicit Adams Method, ▲ 4th Order Implicit Adams Method, ◆ Implicit Trapezoidal Method)



(a)

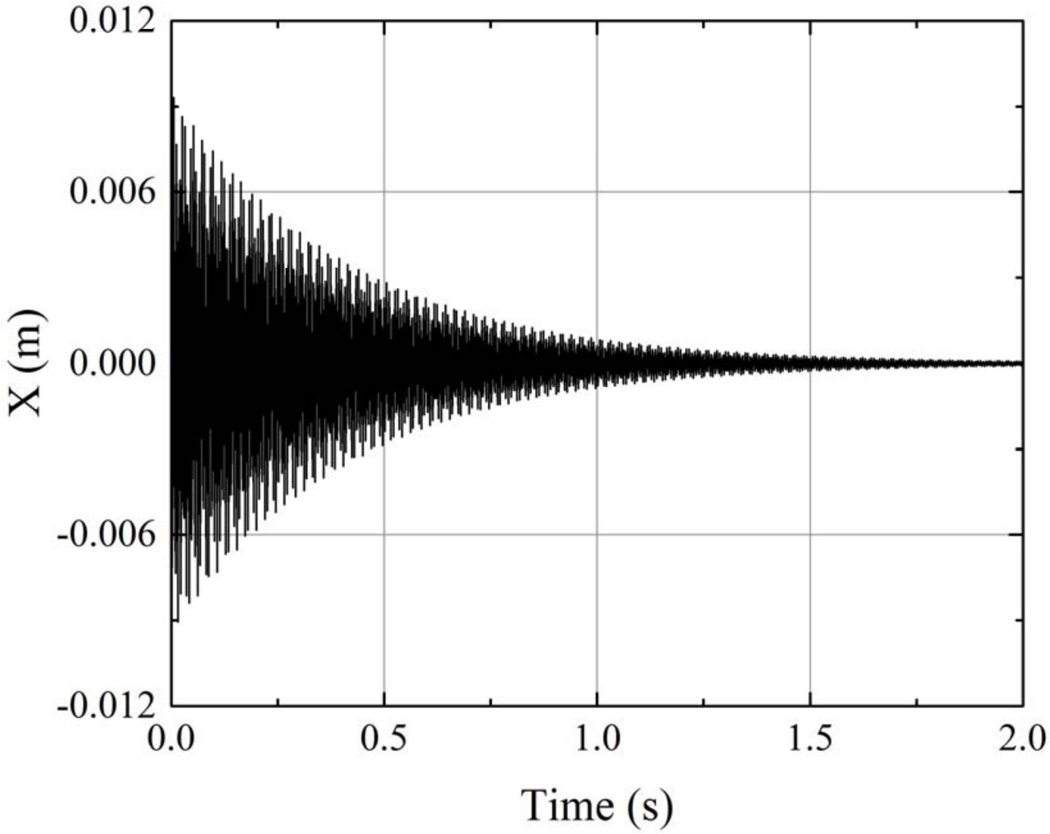
(3rd Order Implicit Adams Method, 4th Order Implicit Adams Method)



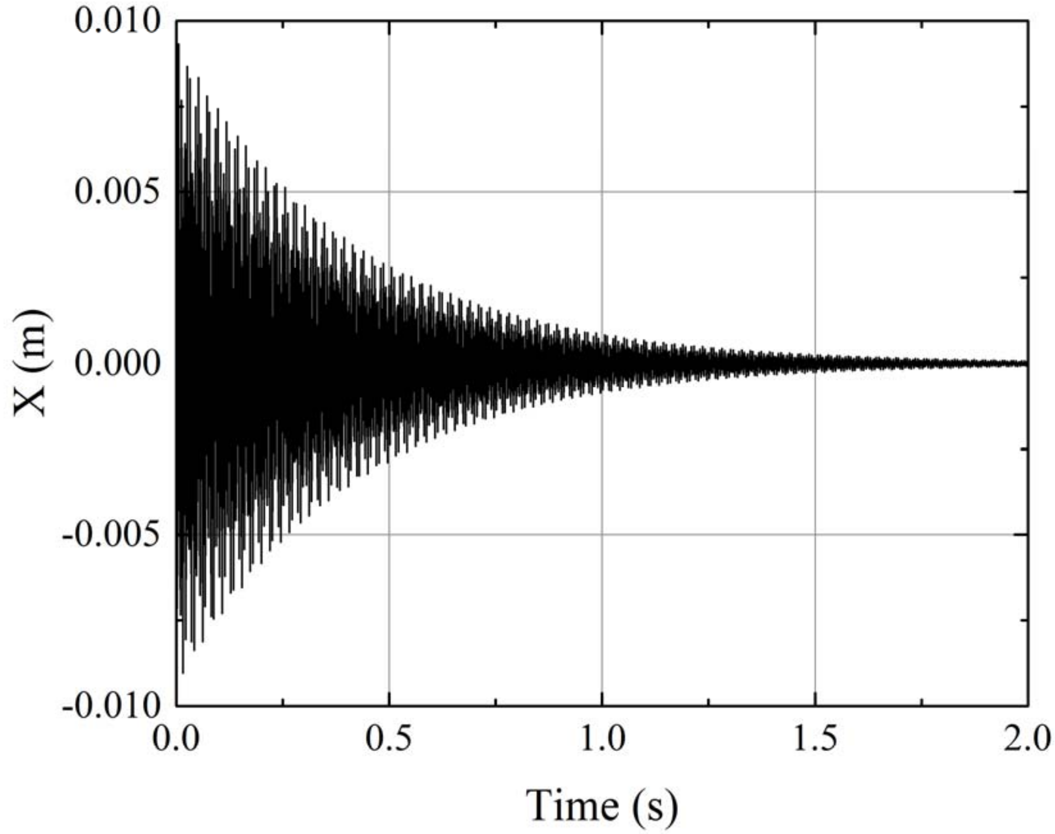
(b)

(Implicit Trapezoidal Method, 4th Order Implicit Adams Method)

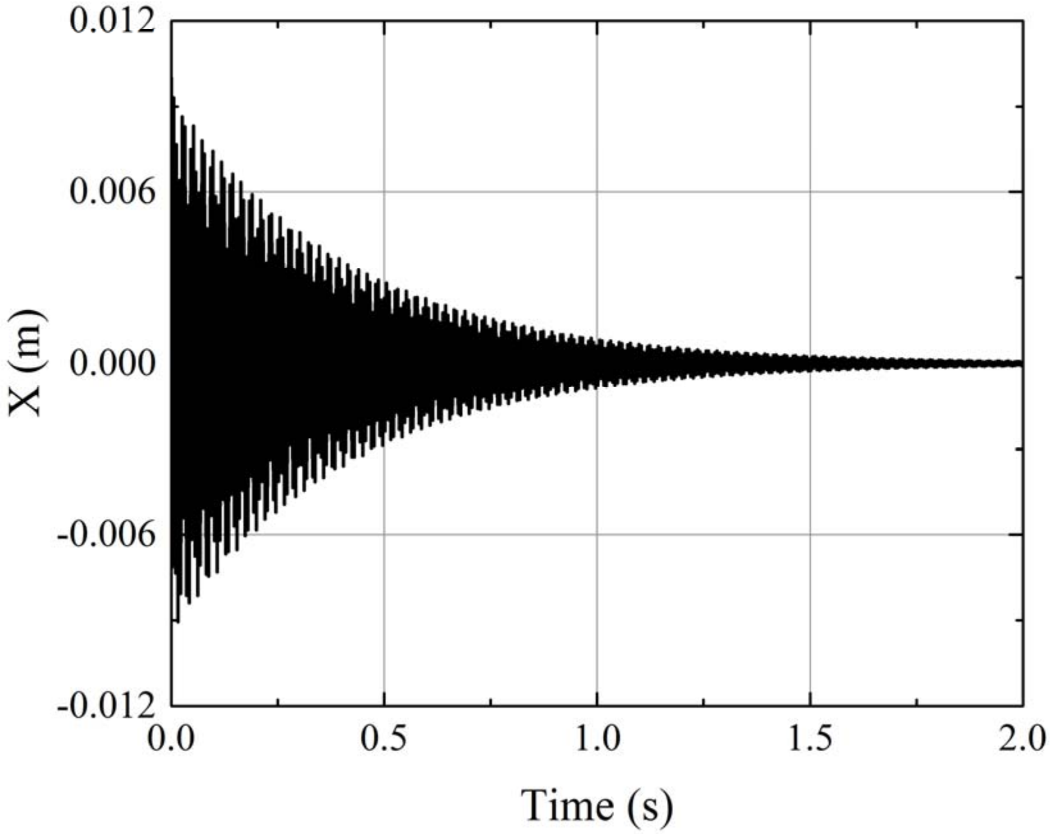
Figure 9. Coordinate of Mass 2 ($\varepsilon_r = 10^{-7}$, $k_2 = 10^8$ N/m)



(a) 3rd Order Implicit Adams Method

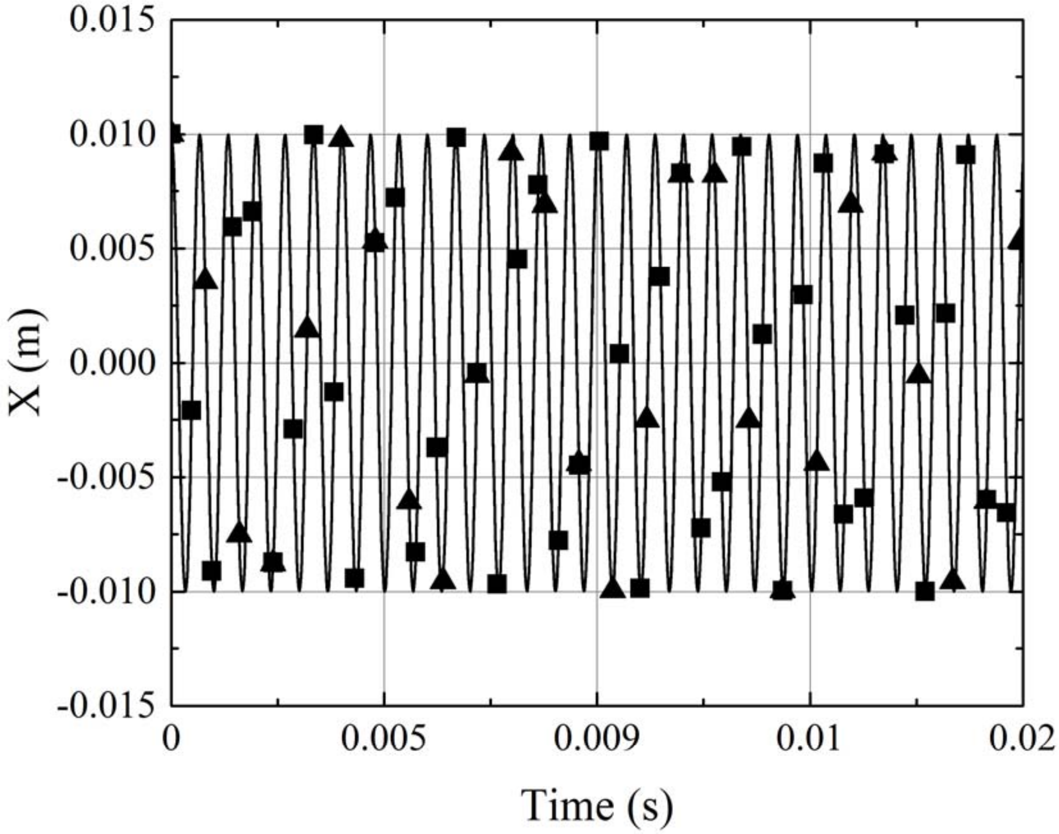


(b) 4th Order Implicit Adams Method



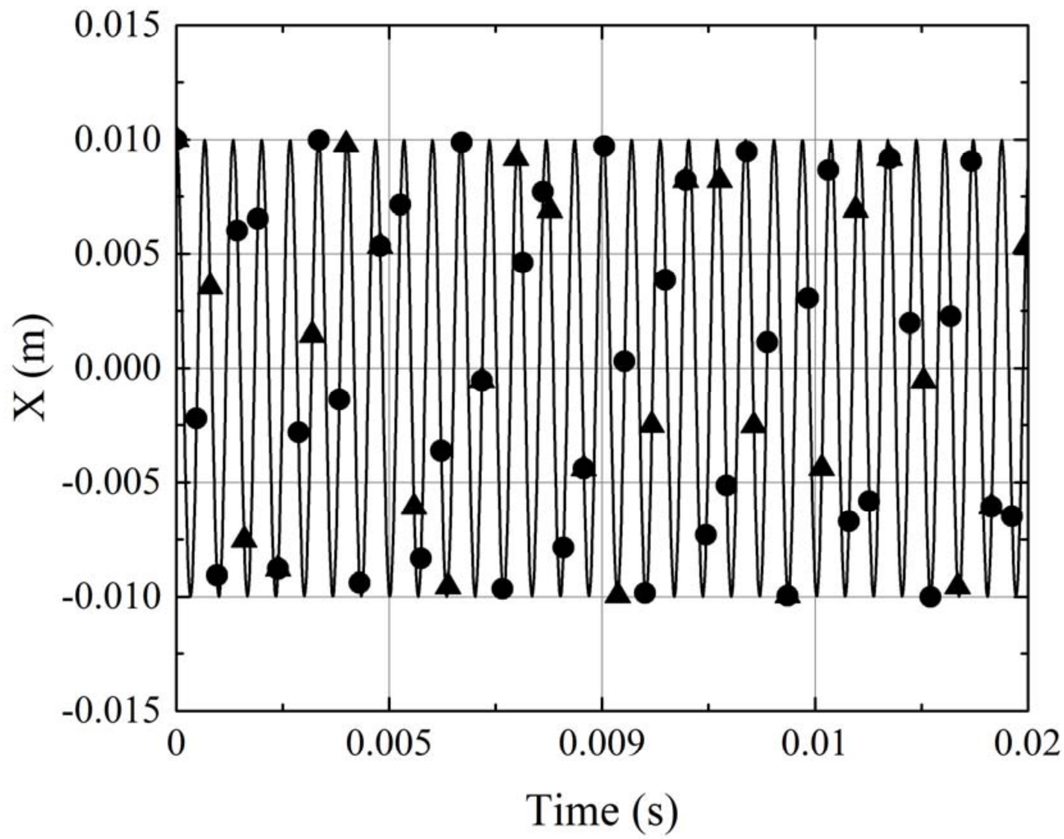
(c) Implicit Trapezoidal Method

Figure 10. Coordinate of Mass 3 ($\varepsilon_r = 10^{-7}$, $k_3 = 10^{12}$ N/m)



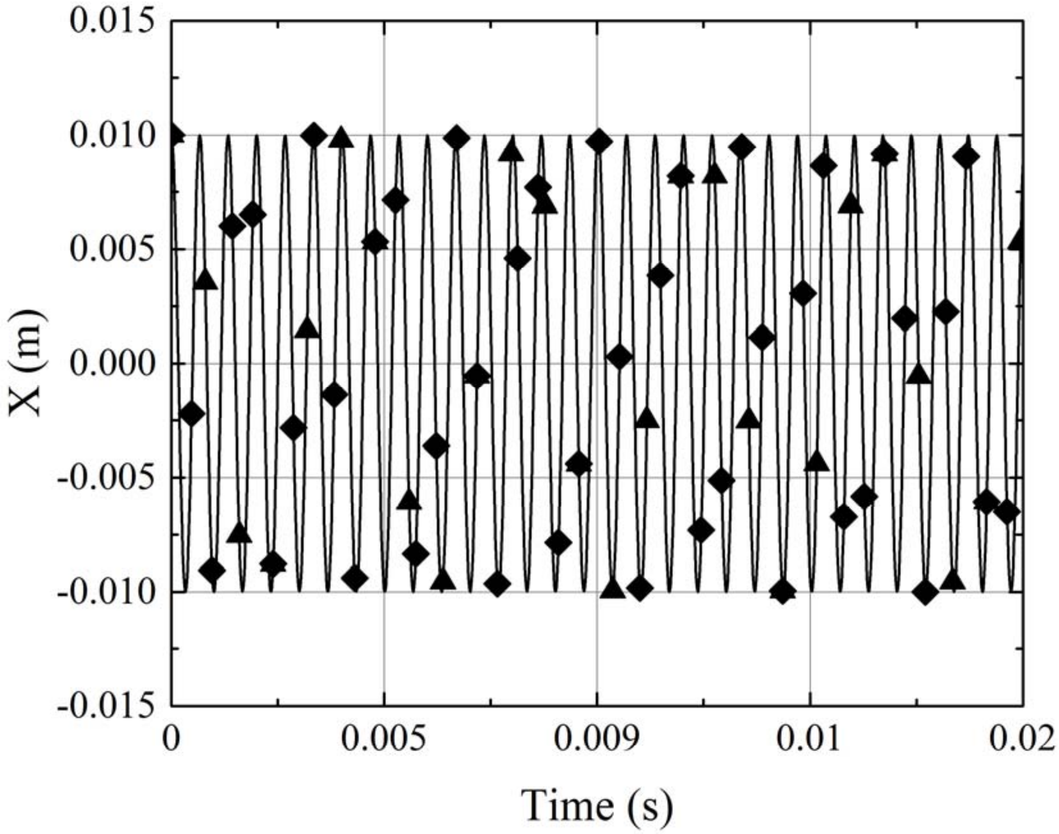
(a)

(—■— Explicit Adams Method, —▲— 4^{th} Order Implicit Adams Method)



(b)

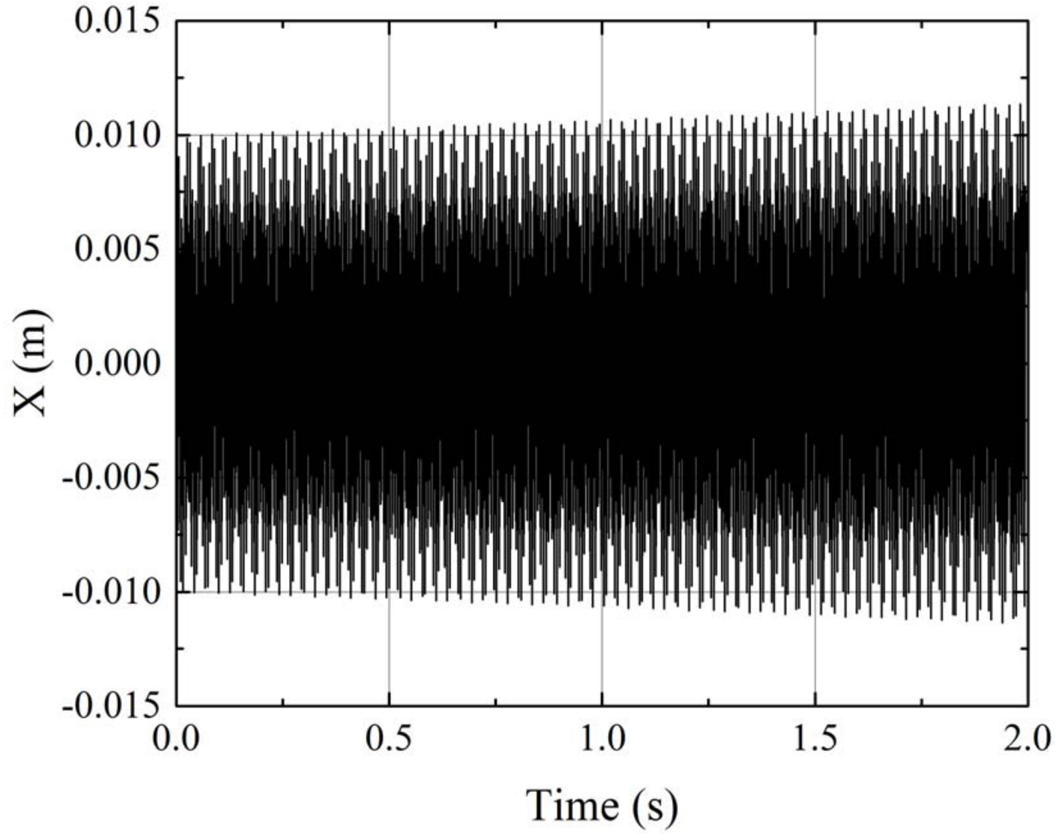
(3rd Order Implicit Adams Method, 4th Order Implicit Adams Method)



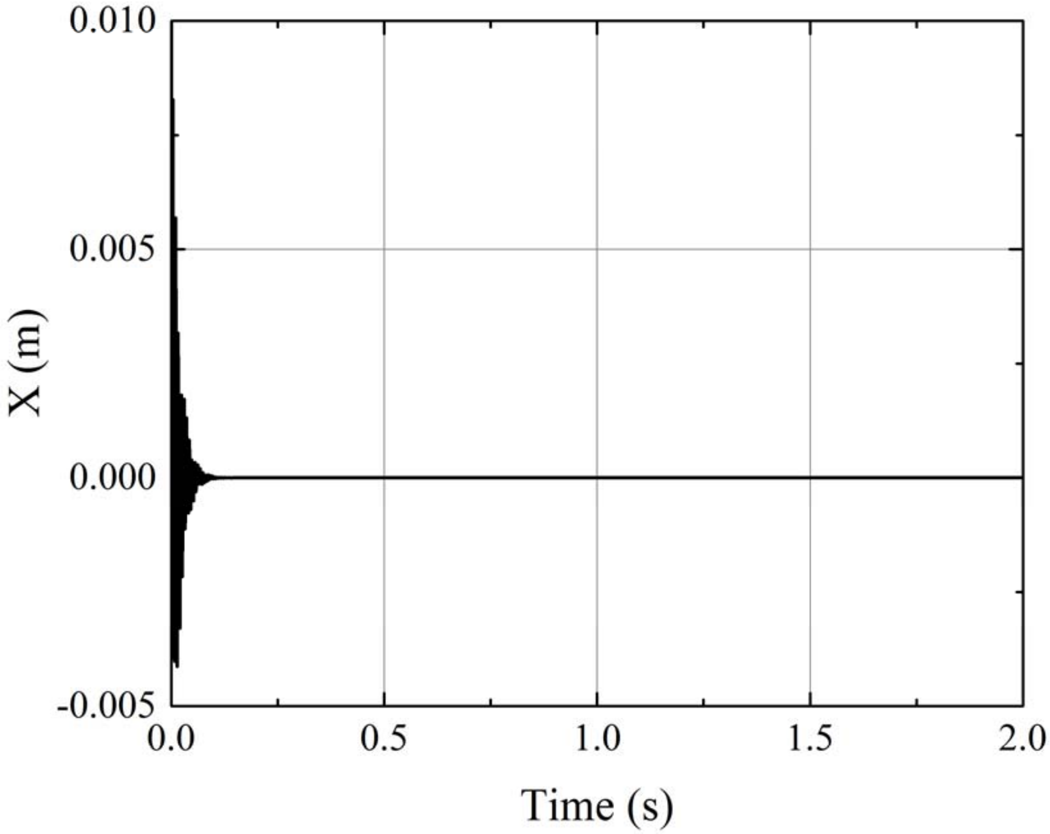
(c)

(Implicit Trapezoidal Method, 4^{th} Order Implicit Adams Method)

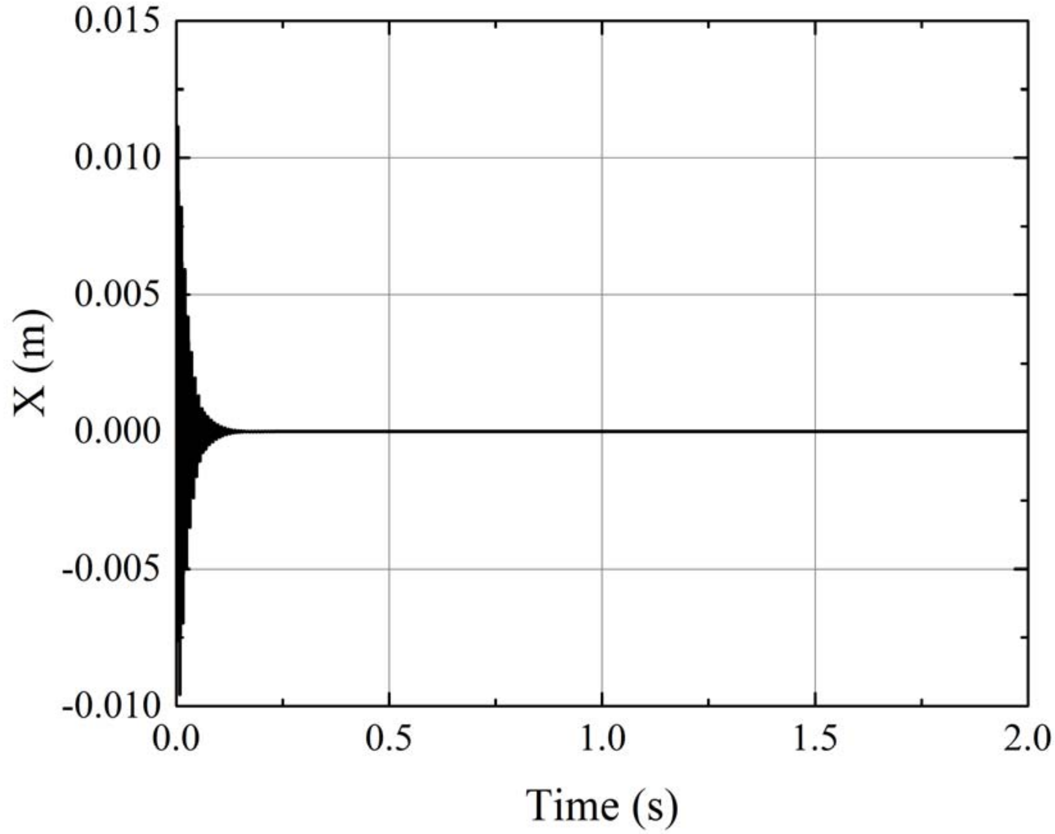
Figure 11. Coordinate of Mass 2 ($t_0 = 10^{-5}$, $k_2 = 10^8$ N/m)



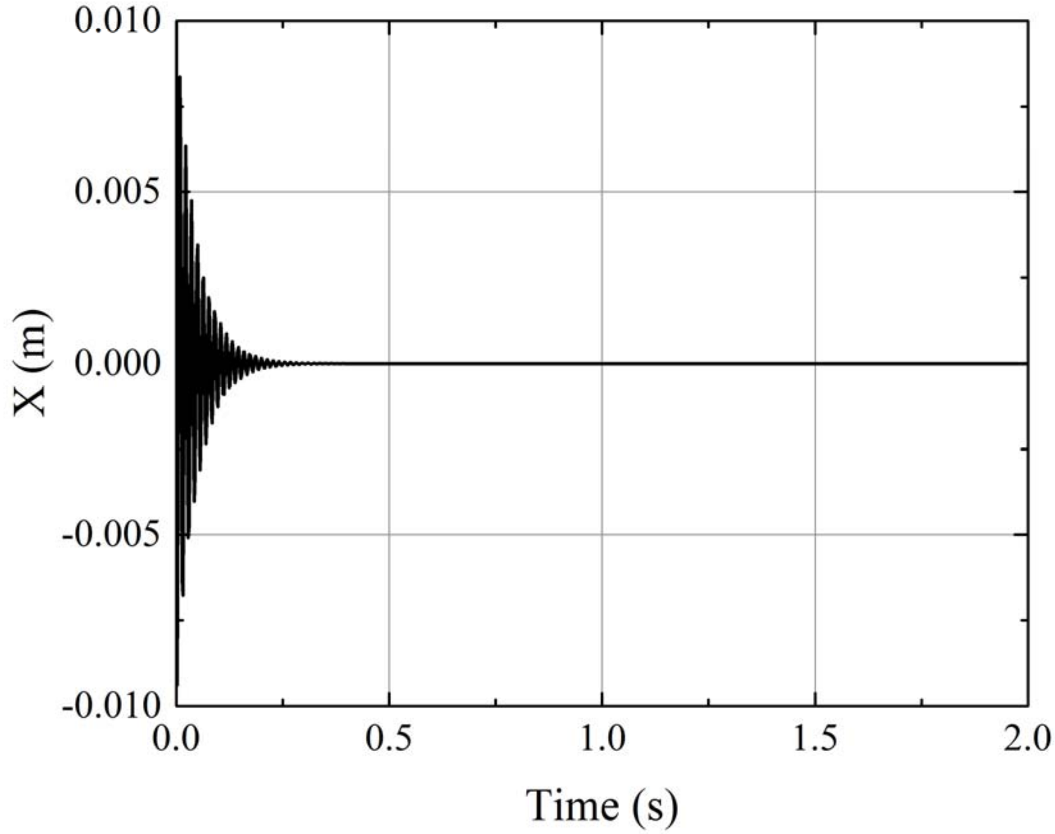
(a) Explicit Adams Method



(b) 3^{rd} Order Implicit Adams Method



(c) 4th Order Implicit Adams Method



(d) Implicit Trapezoidal Method

Figure 12. Coordinate of Mass 3 ($t_0 = 10^{-5}$, $k_3 = 10^{12}$ N/m)

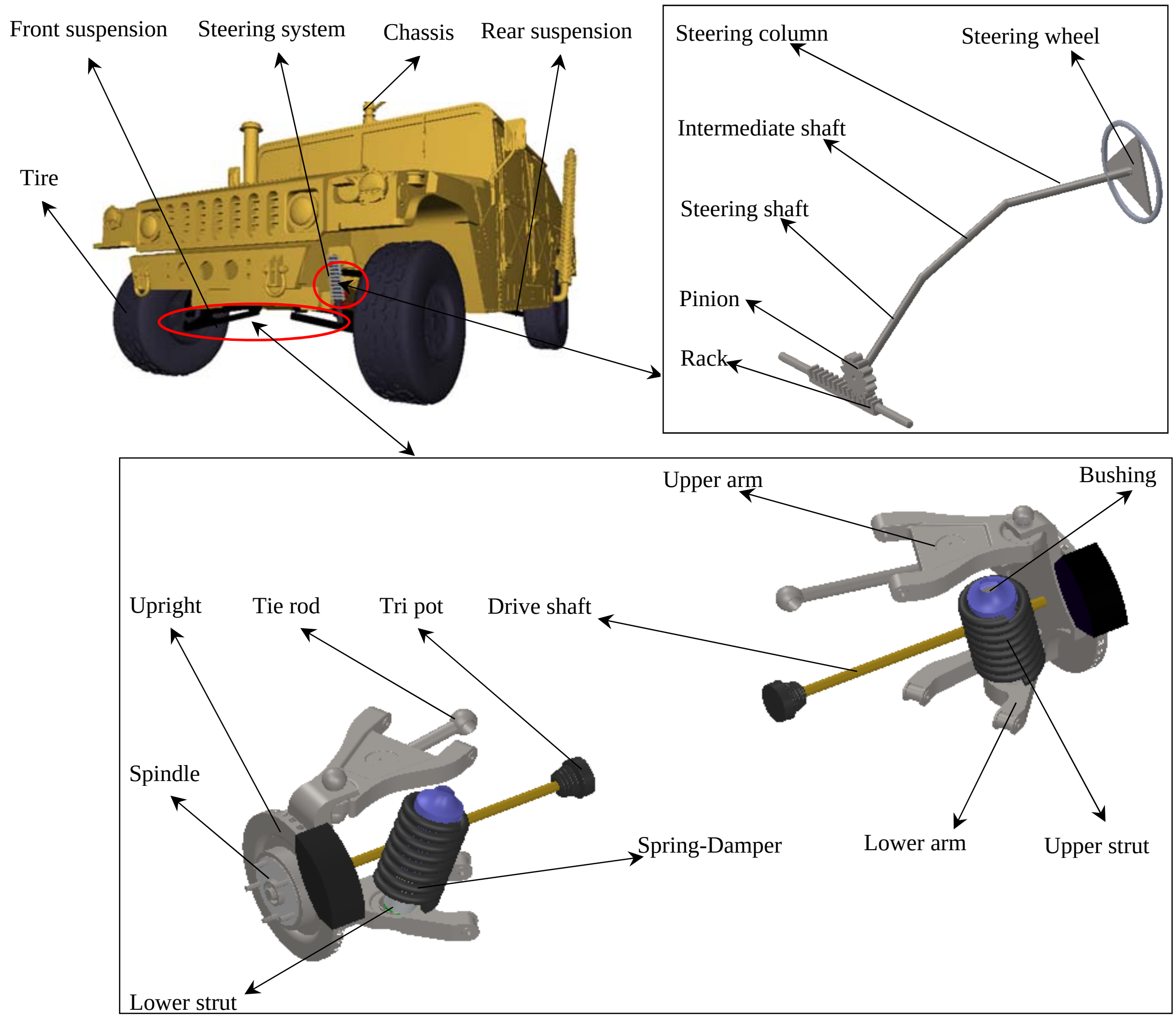


Figure. 13 HMMWV Model

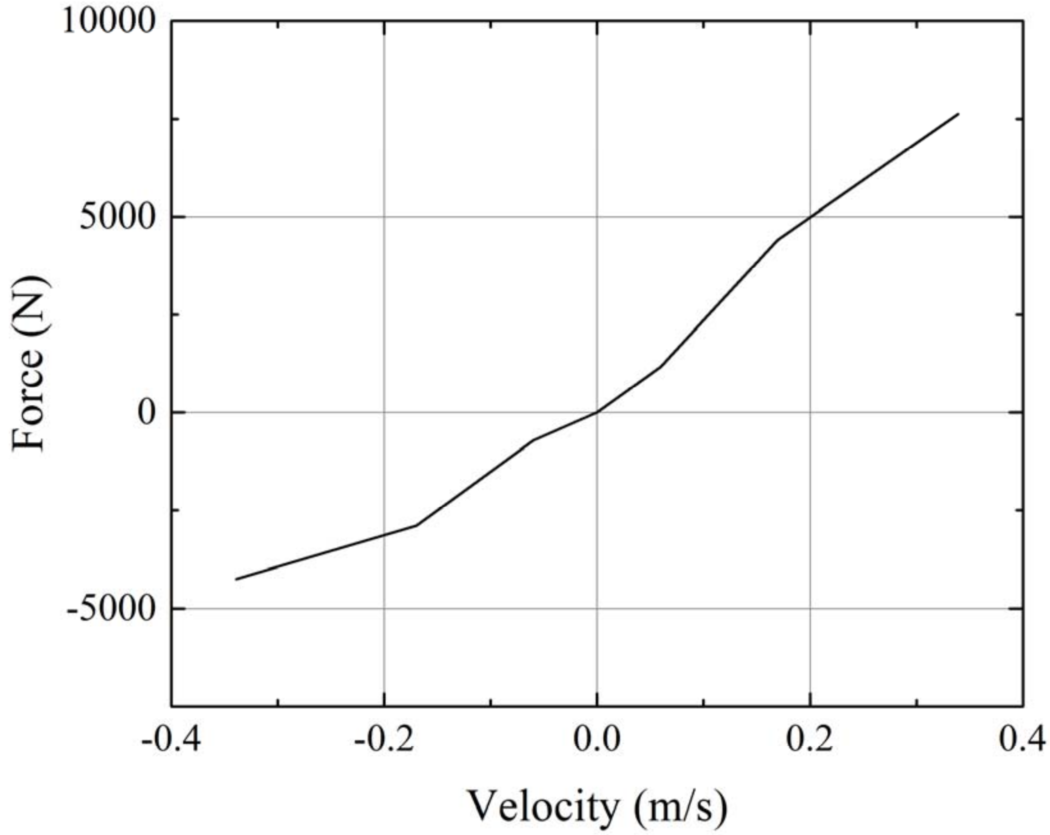


Figure. 14 Damping Coefficient of the Front Suspension Damper

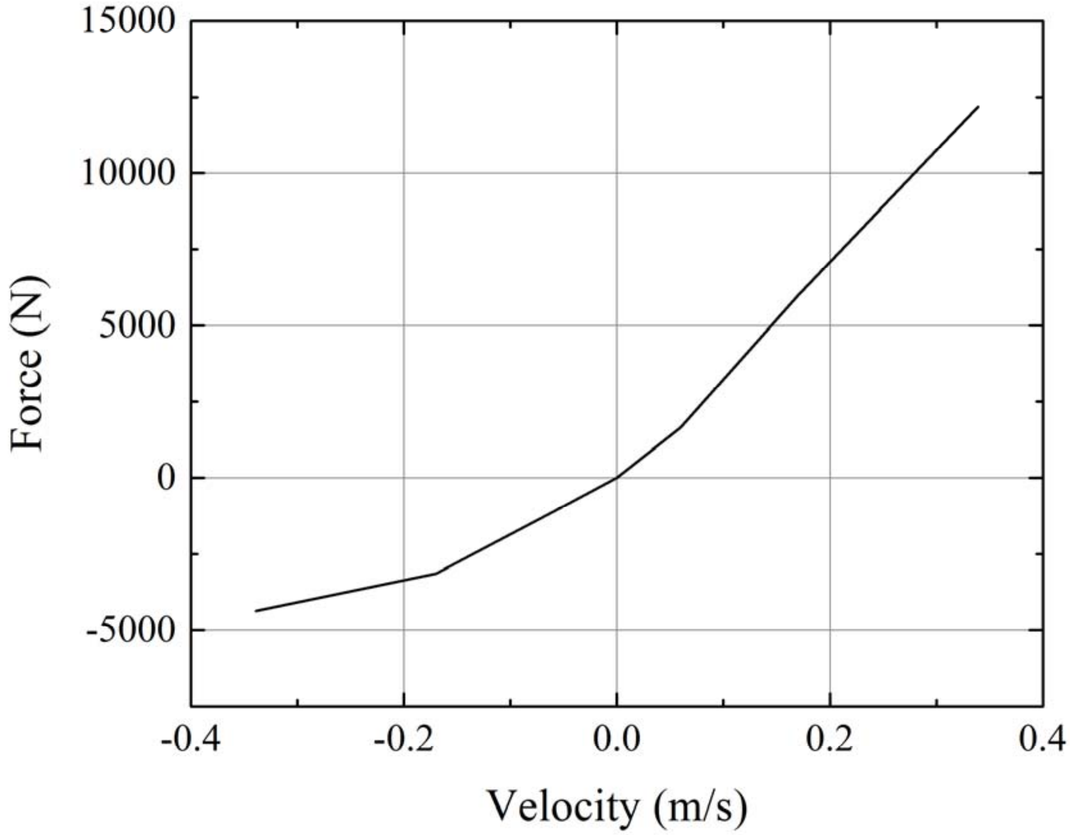


Figure. 15 Damping Coefficient of the Rear Suspension Damper

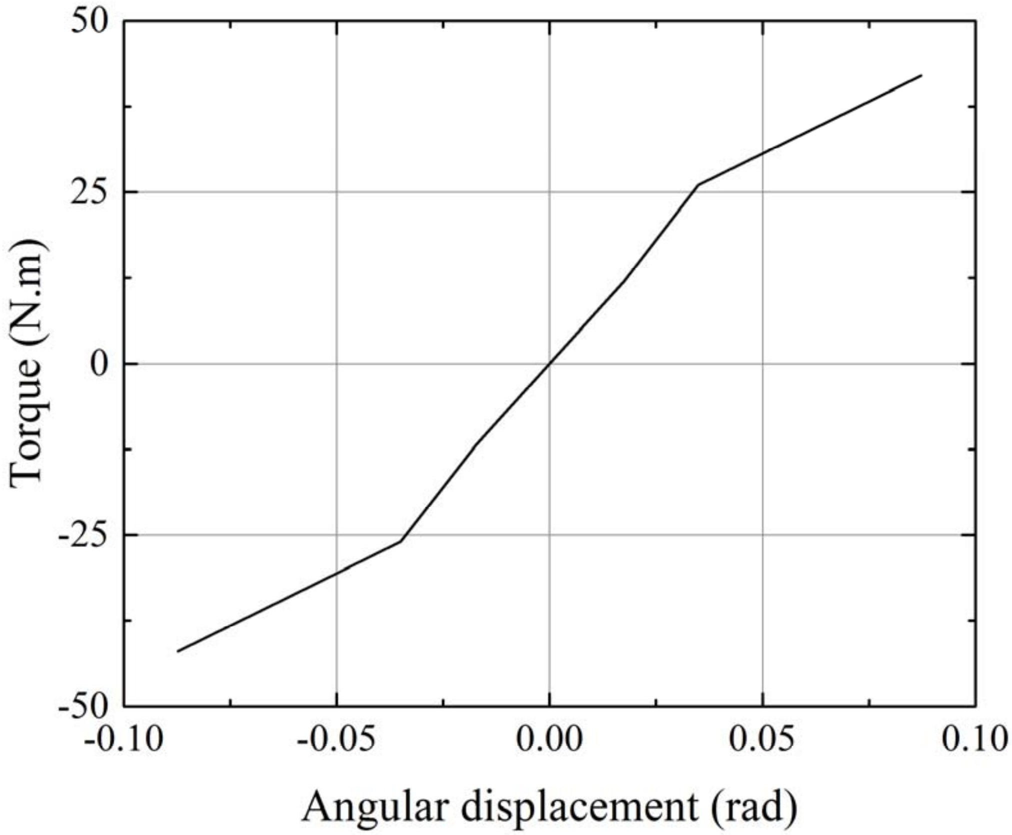


Figure. 16 Bushing Stiffness Coefficient (rotation about *X*-Axis)

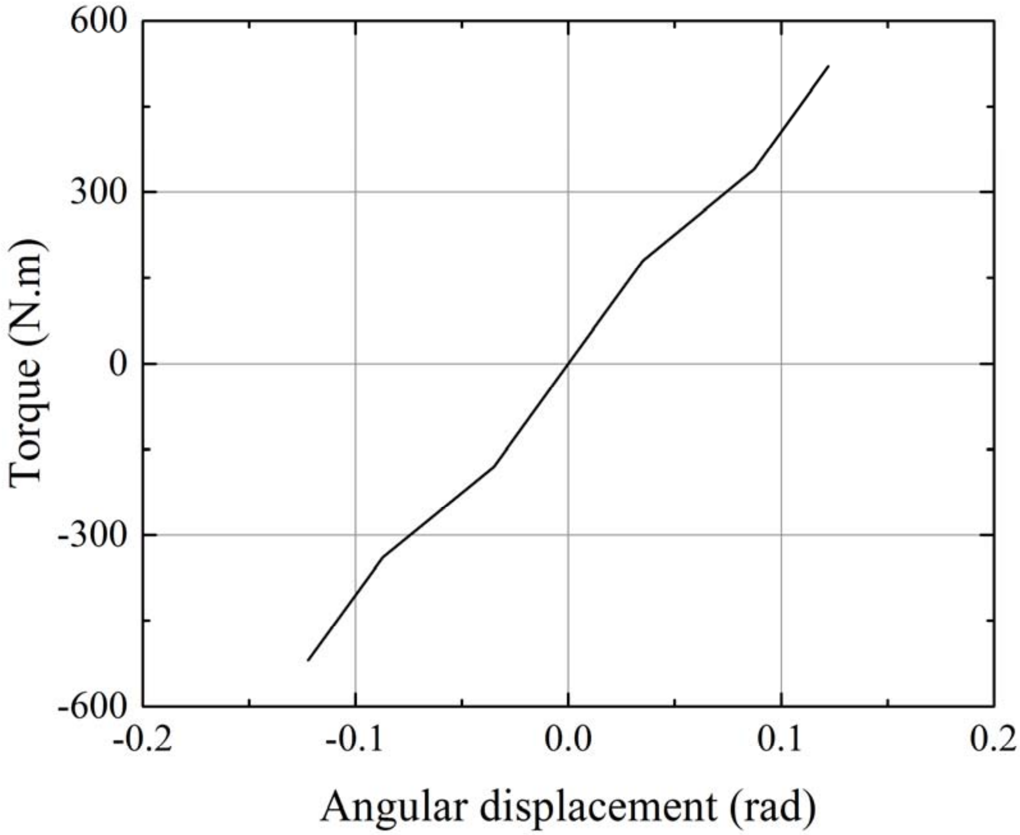


Figure. 17 Bushing Stiffness Coefficient (Rotation about *Y*-Axis)

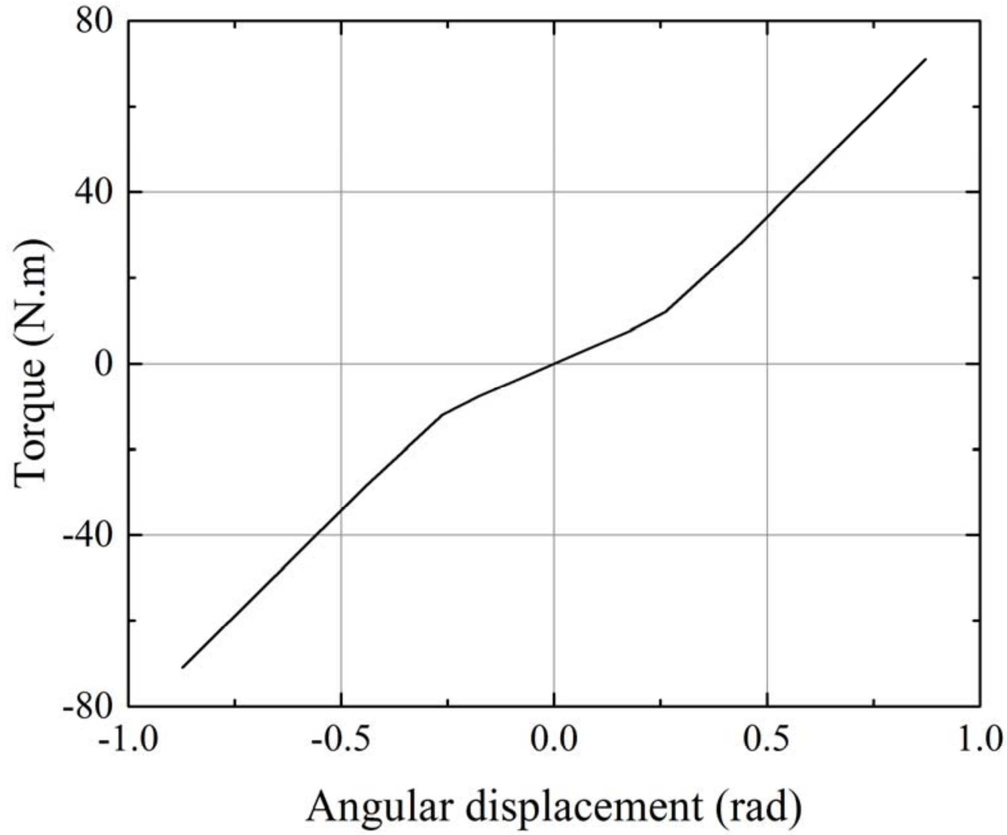


Figure. 18 Bushing Stiffness Coefficient (Rotation about *Z*-Axis)

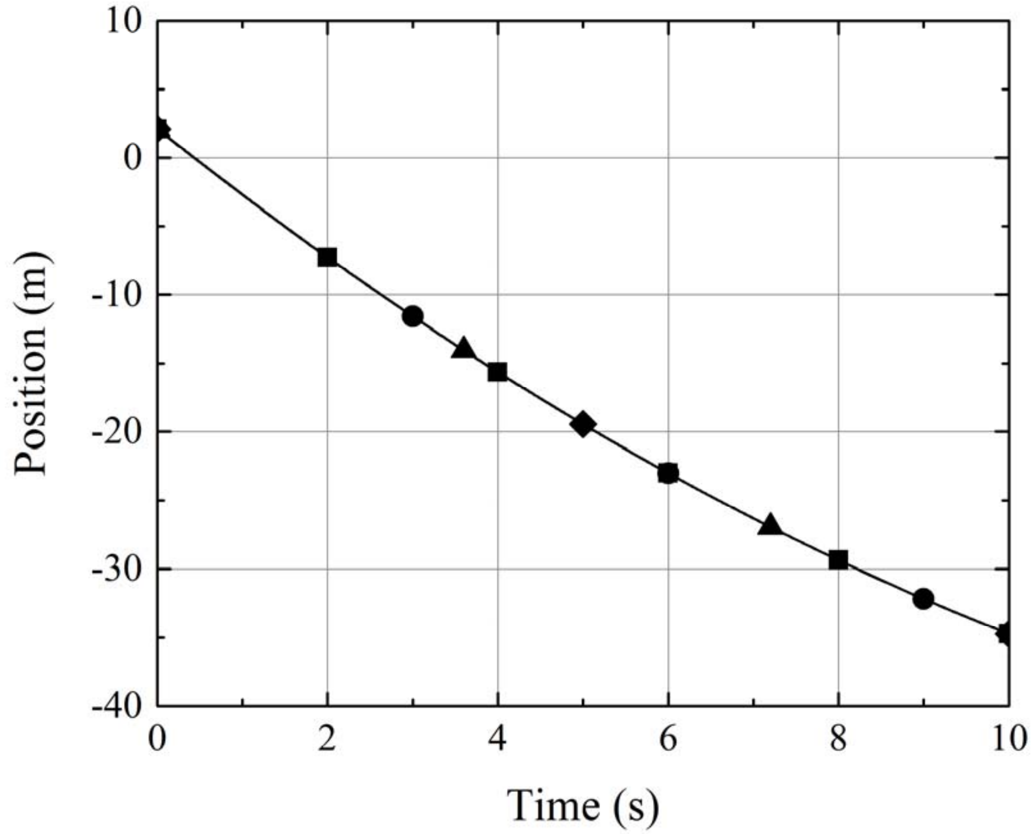


Figure.19 Forward Position of the Chassis Center of Mass

(—■— Explicit Adams Method, —●— 3rd Order Implicit Adams Method, —▲— 4th Order Implicit Adams Method, —◆— Implicit Trapezoidal Method)

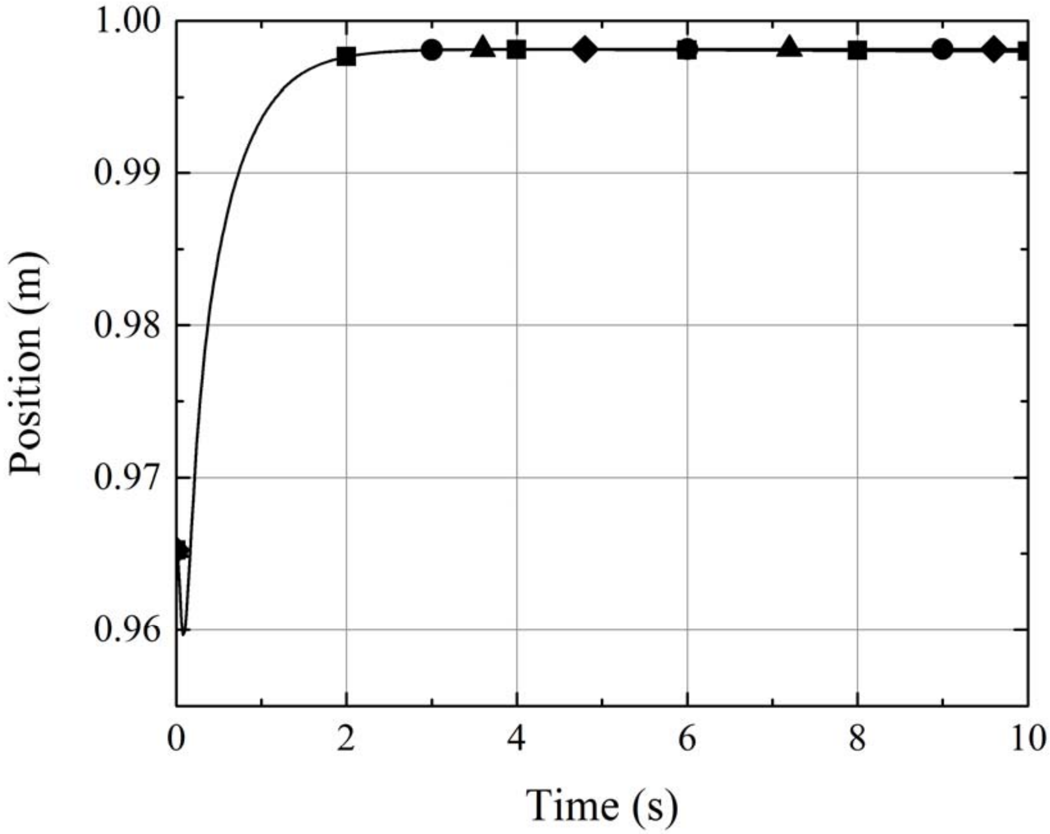


Figure. 20 Vertical Position of the Chassis Center of Mass

(—■— Explicit Adams Method, —●— 3rd Order Implicit Adams Method, —▲— 4th Order Implicit Adams Method, —◆— Implicit Trapezoidal Method)