

DETC2005-85096

ON THE USE OF THE HHT METHOD IN THE CONTEXT OF INDEX 3 DIFFERENTIAL
ALGEBRAIC EQUATIONS OF MULTIBODY DYNAMICS

Dan Negrut*

Mathematics and Computer Science Division
Argonne National Laboratory
Argonne, Illinois 60439
Email: negrut@mcs.anl.gov

Rajiv Rampalli

MSC.Software
Ann Arbor, Michigan, 48105
Email: rajiv.rampalli@mscsoftware.com

Gisli Ottarsson

MSC.Software
Ann Arbor, Michigan, 48105
Email: gisli.ottarsson@mscsoftware.com

Anthony Sajdak

MSC.Software
Ann Arbor, Michigan, 48105
Email: anthony.sajdak@mscsoftware.com

ABSTRACT

The paper presents theoretical and implementation aspects related to a new numerical integrator available in the 2005 version of the MSC.ADAMS/Solver C++. The starting point for the new integrator is the Hilber-Hughes-Taylor method (HHT, also known as α -method) that has been widely used in the finite element community for more than two decades. The method implemented is tailored to answer the challenges posed by the numerical solution of index 3 Differential Algebraic Equations that govern the time evolution of a multi-body system. The proposed integrator was tested with more than 1,600 models prior to its release in the 2005 version of the simulation package MSC.ADAMS. In this paper an all-terrain-vehicle model with flexible chassis is used to prove the good efficiency and accuracy of the method.

GENERAL CONSIDERATIONS ABOUT THE HHT METHOD

The HHT method [1] is widely used in the structural dynamics community for the numerical integration of a linear set

of second Order Differential Equations (ODE). This problem is obtained at the end of a finite element discretization. Provided the finite element approach is linear, the equations of motion assume the form

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{C}\dot{\mathbf{q}} + \mathbf{K}\mathbf{q} = \mathbf{F}(t) \quad (1)$$

The $p \times p$ mass, damping, and stiffness matrices, \mathbf{M} , \mathbf{C} , and \mathbf{K} , respectively, are constant, the force $\mathbf{F} \in \mathbb{R}^p$ depends on time t , and $\mathbf{q} \in \mathbb{R}^p$ is the set of generalized coordinates used to represent the configuration of the mechanical system.

A precursor of the HHT method is the Newmark method [2], in which a family of integration formulas that depend on two parameters β and γ is defined:

$$\mathbf{q}_{n+1} = \mathbf{q}_n + h\dot{\mathbf{q}}_n + \frac{h^2}{2} [(1 - 2\beta)\ddot{\mathbf{q}}_n + 2\beta\ddot{\mathbf{q}}_{n+1}], \quad (2a)$$

*Address all correspondence to this author.

$$\dot{\mathbf{q}}_{n+1} = \dot{\mathbf{q}}_n + h[(1 - \gamma)\ddot{\mathbf{q}}_n + \gamma\ddot{\mathbf{q}}_{n+1}]. \quad (2b)$$

These formulas are used to discretize at time t_{n+1} the equations of motion (1)

$$\mathbf{M}\ddot{\mathbf{q}}_{n+1} + \mathbf{C}\dot{\mathbf{q}}_{n+1} + \mathbf{K}\mathbf{q}_{n+1} = \mathbf{F}_{n+1} \quad (2c)$$

Note that based on Eqs. (2a) and (2b), \mathbf{q}_{n+1} and $\dot{\mathbf{q}}_{n+1}$ are functions of the acceleration $\ddot{\mathbf{q}}_{n+1}$, which in Eq. (2c) remains the sole unknown quantity that is eventually computed as the solution of a linear system. This method is implicit and A-stable (stable in the whole left-hand plane) [3], provided [1]

$$\gamma \geq 1/2 \quad \beta \geq \frac{(\gamma + \frac{1}{2})^2}{4}. \quad (3)$$

The only combination of β and γ that leads to second order accuracy is $\gamma = \frac{1}{2}$, and $\beta = \frac{1}{4}$. The method obtained is the trapezoidal method, which is therefore both A-stable and second order. The drawback of the trapezoidal formula is that it does not induce any numerical damping in the solution, which makes it impractical for problems that have (a) high frequency oscillations that are of no interest, or (b) parasitic high frequency oscillations that are a byproduct of the finite element discretization process [4]. The major drawback of the Newmark family was that it could not provide a formula that was A-stable, second order, and displayed a desirable level of numerical damping. The compromise for the Newmark family was to keep the A-stability and have numerical damping at the expense of the integration order, which went down to one.

The HHT method came as an improvement as it preserved the A-stability and numerical damping properties, while achieving second order accuracy when used in conjunction with the second order linear ODE problem of Eq. (1). The idea proposed in [1] actually does not pertain the expression of the Newmark integration formulas, but rather the form of the discretized equations of motion in (2c). The new equation in which the integration formulas of Eqs. (2a) and (2b) are substituted is

$$\mathbf{M}\ddot{\mathbf{q}}_{n+1} + (1 + \alpha)(\mathbf{C}\dot{\mathbf{q}}_{n+1} + \mathbf{K}\mathbf{q}_{n+1}) - \alpha(\mathbf{C}\dot{\mathbf{q}}_n + \mathbf{K}\mathbf{q}_n) = \mathbf{F}(\tilde{t}_{n+1}) \quad (4a)$$

where

$$\tilde{t}_{n+1} = t_n + (1 + \alpha)h \quad (4b)$$

As indicated in [4], the HHT method will possess the advertised stability and order properties provided that $\alpha \in [-\frac{1}{3}, 0]$, and

$$\gamma = \frac{1 - 2\alpha}{2} \quad \beta = \frac{(1 - \alpha)^2}{4} \quad (5)$$

The smaller the value of α , the more damping is induced in the numerical solution. Note that in the limit, the choice $\alpha = 0$ leads to the trapezoidal method; *i.e.*, no numerical damping is introduced in the solution.

THE INDEX 3 DAE OF MULTI-BODY DYNAMICS

The state of a multi-body system at the position level is represented in this paper by an array $\mathbf{q} = [q_1, \dots, q_n]^T$ of generalized coordinates. The velocity of the system is described by the array of generalized velocities $\dot{\mathbf{q}} = [\dot{q}_1, \dots, \dot{q}_n]^T$. For each body i its position is described by the vector $\mathbf{r}_i = [x_i, y_i, z_i]^T$ that locates the center of mass; the body orientation is given by the array of local 3-1-3 Euler angles [5], $\mathbf{e}_i = [\psi_i, \theta_i, \phi_i]^T$. Consequently, for a mechanical system containing n_b bodies, $\mathbf{q} = [\mathbf{r}_1^T \mathbf{e}_1^T \dots \mathbf{r}_{n_b}^T \mathbf{e}_{n_b}^T]^T \in \mathbb{R}^p$, $p = 6n_b$. This set of position generalized coordinates may be augmented with deformation modes when flexible bodies are present in the model. In order to keep notation simple the assumption made in the presentation is that there are no flexible bodies in the model. Flexible bodies do not pose a problem as shown in the numerical experiments section.

In any constrained mechanical system, joints connecting bodies restrict their relative motion and impose constraints on the generalized coordinates. Kinematic constraints are then formulated as algebraic expressions involving generalized coordinates,

$$\Phi(\mathbf{q}, t) = [\Phi_1(\mathbf{q}, t) \dots \Phi_m(\mathbf{q}, t)]^T = \mathbf{0} \quad (6a)$$

where m is the total number of constraint equations that must be satisfied by the generalized coordinates throughout the simulation. It is assumed here that the m constraint equations are independent. Although the implementation of the proposed method handles non-holonomic constraints, to keep the presentation simpler the case of holonomic constraints is assumed in what follows.

Differentiating Eq. (6a) with respect to time leads to the velocity kinematic constraint equation

$$\Phi_{\mathbf{q}}(\mathbf{q}, t)\dot{\mathbf{q}} + \Phi_t(\mathbf{q}, t) = \mathbf{0} \quad (6b)$$

where the over-dot denotes differentiation with respect to time

and the subscript denotes partial differentiation, $\Phi_{\mathbf{q}} = \left[\frac{\partial \Phi_i}{\partial q_j} \right]$, for $1 \leq i \leq m$, and $1 \leq j \leq n$. Finally, the acceleration kinematic constraint equations are obtained by differentiating Eq. (6b) with respect to time,

$$\Phi_{\mathbf{q}}(\mathbf{q}, t) \ddot{\mathbf{q}} + (\Phi_{\mathbf{q}}(\mathbf{q}, t) \dot{\mathbf{q}})_{\mathbf{q}} \dot{\mathbf{q}} + 2\Phi_{\mathbf{q}t}(\mathbf{q}, t) \dot{\mathbf{q}} + \Phi_{tt}(\mathbf{q}, t) = \mathbf{0} \quad (6c)$$

Equations (6a)–(6c) characterize the admissible motion of the mechanical system.

The state of the mechanical system changes in time under the effect of applied forces. The time evolution of the system is governed by the Lagrange multiplier form of the constrained equations of motion [6],

$$\mathbf{M}(\mathbf{q}) \ddot{\mathbf{q}} + \Phi_{\mathbf{q}}^T(\mathbf{q}) \lambda = \mathbf{Q}(\dot{\mathbf{q}}, \mathbf{q}, t) \quad (6d)$$

where $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{p \times p}$ is the generalized mass, and $\mathbf{Q}(\dot{\mathbf{q}}, \mathbf{q}, t) \in \mathbb{R}^p$ is the action (as opposed to the reaction $\Phi_{\mathbf{q}}^T(\mathbf{q}) \lambda$) force acting on the generalized coordinates $\mathbf{q} \in \mathbb{R}^p$. These equations are neither linear, nor ordinary differential as is the case in Eq. (1), first and foremost because the solution $\mathbf{q}(t)$ must also satisfy the kinematic constraint equations in Eq. (6a). These constraint equations lead in Eq. (6d), to the presence of the reaction force $\Phi_{\mathbf{q}}^T(\mathbf{q}) \lambda$, where $\lambda \in \mathbb{R}^m$ is the Lagrange multiplier associated with the kinematic constraints.

In addition to the equations of motion and kinematic constraint equations, there are several classes of equations that need to be considered in a general purpose mechanical simulation package:

1. *User defined variables*, which can technically be regarded as aliases or definition equations. A set of n_v user defined variables $\mathbf{V} \in \mathbb{R}^{n_v}$ is typically specified through an equation of the form

$$\mathbf{V} - \mathbf{v}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \lambda, \mathbf{V}, \mathbf{F}, t) = \mathbf{0}_{n_v} \quad (7a)$$

and which during the solution sequence are solved (or rather evaluated) simultaneously with the equations of motion and the kinematic constraint equations. Here $\mathbf{v} \in \mathbb{R}^{n_v}$ is a user defined function that depends on other system states as indicated in Eq. (7a).

2. *External force definition*, \mathbf{F} , which allow the user to more conveniently define the set of n_f applied forces $\mathbf{F} \in \mathbb{R}^{n_f}$ that act on the system. This is the mechanism through which a complex tire model can be hooked up with a vehicle model, or the avenue through which the user can define his/her own

bushing elements, custom non-linear dampers, friction, etc.

$$\mathbf{F} - \mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \lambda, \mathbf{V}, \mathbf{F}, t) = \mathbf{0}_{n_f} \quad (7b)$$

Equations (6a)–(6d) comprise a system of index 3 DAE [7]. It is known that differential-algebraic equations are not ordinary differential equations [8]. Analytical solutions of Eqs. (6a) and (6d) automatically satisfy Eqs. (6b) and (6c), but this is no longer true for numerical solutions. In general, the task of obtaining a numerical solution of the DAE of Eqs. (6a)–(6d) is substantially more difficult and prone to intense numerical computation than that of solving ordinary differential equations (ODE). For an account of relevant work in the area of numerical integration methods for the DAE of Multi-body Dynamics the reader is referred to [3, 7, 9–12] and references therein.

The theory and attractive features associated with the HHT method have been derived in conjunction with a linear second order ODE. The only similarity between Eqs. (1) and (6d) is that they are both second order, and qualitatively obtained from Newton's second law. In [13] and more recently [14], for the purpose of stability and convergence analysis the constrained equations of motion are tackled in a stabilized index 2 DAE framework. The HHT method is also discussed in [15] and more recently in [16], where the proposed implementation is based on a technique that accounts for violations in the position and velocity constraints in a stabilization framework similar to the one proposed in [17]. There are also several Runge-Kutta based approaches for highly oscillatory mechanical system simulation that, like the HHT method, display the attractive attribute of selectively damping frequency at the high end of the spectrum. In [18], a Singly Diagonal Runge-Kutta (SDIRK) based method allows the user to choose, within certain bounds, the diagonal value in the formula, and thus control the amount of numerical damping associated with the algorithm. The role of the diagonal element in the formula becomes very similar to the role of the α parameter in the HHT method. An approach based on additive Runge-Kutta methods that has the potential to accurately handle highly oscillatory multi-body dynamics simulation was introduced in [19], and further discussed in [20]. These novel Runge-Kutta based algorithms are mathematically very sound, but more time is required for them to achieve, vis-a-vis industrial strength applications, the level of acceptance currently associated with the well established HHT method.

THE PROPOSED ALGORITHM

The index 3 DAE problem of multi-body dynamics is neither linear, nor ordinary differential in nature and the HHT method is thus applied for a different class of problems than what it was originally designed for. Rather than approaching the solution

within an index 2 framework [13, 14] or using a stabilization approach [15, 16], the proposed algorithm uses the implicit Newmark formulas to discretize the equations of motion and requires that the position-level kinematic constraint equations be satisfied at the end of each time step. This is a direct index 3 approach and it requires at each integration time step the solution of a non-linear system of equations. The theoretical foundation of this method is provided firstly by the stability and convergence results in [21] and [22], and secondly by the fact that as pointed out in [4], the HHT method is equivalent with a three-step implicit multi-step integration formula.

At the cornerstone of the proposed algorithm lies the simple idea upon which the HHT method is built: recycle the Newmark integration formulas, and slightly change the equations of motion to account for the set of forces acting on the system at two consecutive integration points. By means of the Newmark formulas, the collection of differential and algebraic equations are discretized to obtain an algebraic non-linear system that is solved by means of a modified-Newton algorithm.

The unknowns of interest are the generalized positions, velocities, and accelerations \mathbf{q} , $\dot{\mathbf{q}}$, and $\ddot{\mathbf{q}}$, respectively, the Lagrange multipliers λ , the applied force states \mathbf{F} , the user-defined variables (aliases) \mathbf{V} . Note that the generalized force \mathbf{Q} is obtained by projecting (via a linear transformation) the force states \mathbf{F} along the generalized coordinates \mathbf{q} ; *i.e.*, $\mathbf{Q} = \Pi \mathbf{F}$, where the projection operator $\Pi = \Pi(\mathbf{q})$ depends on the choice of generalized coordinates.

Looking at Eq. (4a), the idea behind the HHT discretization is that the α -scaling is done in conjunction with the overall forces applied on the system. The current value of the force is first scaled by $1 + \alpha$, while the value of the force at the previous time step is subtracted after being scaled by α . According to Eq. (6d), for the equations of motion in the multi-body formulation, the force is computed as $\Phi_{\mathbf{q}}^T \lambda - \mathbf{Q}(\dot{\mathbf{q}}, \mathbf{q}, t)$; *i.e.*, everything that is not explicitly depending on acceleration $\ddot{\mathbf{q}}$. Therefore, in the spirit of the original HHT formulation the discretization of the multi-body dynamics equations of motion yields

$$(\mathbf{M}\ddot{\mathbf{q}})_{n+1} + (1 + \alpha) (\Phi_{\mathbf{q}}^T \lambda - \mathbf{Q})_{n+1} - \alpha (\Phi_{\mathbf{q}}^T \lambda - \mathbf{Q})_n = \mathbf{0} \quad (8)$$

For notational simplicity, when obvious, the dependency of some quantities on \mathbf{q} and/or $\dot{\mathbf{q}}$ and/or time t will be omitted as was done in Eq. (8). From an implementation standpoint it is more advantageous to scale the previous equation by $(1 + \alpha)$ to obtain the equivalent form

$$\frac{1}{1 + \alpha} (\mathbf{M}\ddot{\mathbf{q}})_{n+1} + (\Phi_{\mathbf{q}}^T \lambda - \mathbf{Q})_{n+1} - \frac{\alpha}{1 + \alpha} (\Phi_{\mathbf{q}}^T \lambda - \mathbf{Q})_n = \mathbf{0} \quad (9a)$$

The Newmark integration formulas are used in an index 3 DAE direct approach to discretize the remaining differential and algebraic equations to obtain

$$\Phi(\mathbf{q}_{n+1}, t_{n+1}) = \mathbf{0} \quad (9b)$$

$$\mathbf{V}_{n+1} - \mathbf{v}(\mathbf{q}_{n+1}, \dot{\mathbf{q}}_{n+1}, \ddot{\mathbf{q}}_{n+1}, \lambda_{n+1}, \mathbf{V}_{n+1}, \mathbf{F}_{n+1}, t_{n+1}) = \mathbf{0} \quad (9c)$$

$$\mathbf{F}_{n+1} - \mathbf{f}(\mathbf{q}_{n+1}, \dot{\mathbf{q}}_{n+1}, \ddot{\mathbf{q}}_{n+1}, \lambda_{n+1}, \mathbf{V}_{n+1}, \mathbf{F}_{n+1}, t_{n+1}) = \mathbf{0} \quad (9d)$$

Everywhere in Eq. (9), the Newmark integration formulas of Eq. (2) are used to express \mathbf{q} and $\dot{\mathbf{q}}$ as a function of $\ddot{\mathbf{q}}$. A Newton-like algorithm [23] is used to solve the resulting system of non-linear equations for the set of unknowns (in this order) $\ddot{\mathbf{q}}$, λ , \mathbf{V} , and \mathbf{F} . The iterative method requires at each iteration (k) the solutions of the linear system

$$\begin{bmatrix} \hat{\mathbf{M}} & \Phi_{\mathbf{q}}^T & 0 & -\mathbf{I} \\ \Phi_{\mathbf{q}} & 0 & 0 & 0 \\ -(\mathbf{v}_{\ddot{\mathbf{q}}} + \gamma h \mathbf{v}_{\dot{\mathbf{q}}} + \beta h^2 \mathbf{v}_{\mathbf{q}}) & -\mathbf{v}_{\lambda} & \mathbf{I} - \mathbf{v}_{\mathbf{V}} & -\mathbf{v}_{\mathbf{F}} \\ -(\mathbf{f}_{\ddot{\mathbf{q}}} + \gamma h \mathbf{f}_{\dot{\mathbf{q}}} + \beta h^2 \mathbf{f}_{\mathbf{q}}) & -\mathbf{f}_{\lambda} & -\mathbf{f}_{\mathbf{V}} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \Delta \ddot{\mathbf{q}} \\ \Delta \lambda \\ \Delta \mathbf{V} \\ \Delta \mathbf{F} \end{bmatrix}^{(k)} = \begin{bmatrix} -\mathbf{c}_1 \\ -\mathbf{c}_2 \\ -\mathbf{c}_3 \\ -\mathbf{c}_4 \end{bmatrix}^{(k)} \quad (10)$$

where \mathbf{c}_i are the residuals in satisfying the set of discretized equations of motion, constraint equations, variable definition equations, and applied force definition equations, respectively:

$$\begin{aligned} \mathbf{c}_1 &= \frac{1}{1 + \alpha} (\mathbf{M}\ddot{\mathbf{q}})_{n+1} + (\Phi_{\mathbf{q}}^T \lambda - \mathbf{Q})_{n+1} - \frac{\alpha}{1 + \alpha} (\Phi_{\mathbf{q}}^T \lambda - \mathbf{Q})_n, \\ \mathbf{c}_2 &= \frac{1}{\beta h^2} \Phi(\mathbf{q}, t), \\ \mathbf{c}_3 &= \mathbf{V} - \mathbf{v}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \lambda, \mathbf{V}, t), \\ \mathbf{c}_4 &= \mathbf{F} - \mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \mathbf{V}, t). \end{aligned}$$

Likewise, the matrix $\hat{\mathbf{M}}$ in Eq. (10) is defined as

$$\hat{\mathbf{M}} = \frac{\partial \mathbf{e}_1}{\partial \ddot{\mathbf{q}}} = \mathbf{M} - h\gamma \frac{\partial \mathbf{Q}}{\partial \dot{\mathbf{q}}} + \beta h^2 \left[(\mathbf{M}\ddot{\mathbf{q}})_{\mathbf{q}} + (\Phi_{\mathbf{q}}^T \lambda)_{\mathbf{q}} - \frac{\partial \mathbf{Q}}{\partial \mathbf{q}} \right]. \quad (11)$$

Unless otherwise specified, all the quantities in \mathbf{e}_1 through \mathbf{c}_4 above are evaluated at time t_{n+1} . Finally, note that the non-linear equations associated with the position kinematic constraints are

scaled by $\frac{1}{\beta h^2}$ in order to improve the conditioning of the coefficient matrix in Eq. (10). This is a compromise reached after considering the following alternatives: (a) have the level-zero positions, \mathbf{q} be the unknowns (replacing $\dot{\mathbf{q}}$; then some entries in the Jacobian matrix in Eq. (10) will have to be divided by βh^2 ; (b) have $\ddot{\mathbf{q}}$ be the unknown, but then the second row in the Jacobian matrix comes multiplied by βh^2 ; (c) do as in (b), except that the set of positions kinematic constraint equations are scaled by $\frac{1}{\beta h^2}$. Alternative (a) is what the default GSTIFF integrator currently uses in the MSC.ADAMS simulation package [24] (here entries get divided by a factor $\beta_0 h$ rather than βh^2 , as the second order equations of motion are reduced to an equivalent first order system of differential equations). On numerous occasions this has been observed to be the cause of numerical problems once the step-size becomes very small and consequently some entries in the Jacobian become extremely large. A bad Jacobian condition number ensues, and the quality of the Newton corrections becomes poor. The alternative (b) was not embraced due to the fact that the problem at (a) plagues in a more subtle way this approach as well. If h becomes very small, the second row of the Jacobian matrix is scaled by βh^2 , which practically makes all the entries in this row very small, from where the same large condition number situation ensues. Alternative (c) proved a good solution since typically the type of error that one sees in satisfying the position kinematic constraint equations is very small. It is never that these constraint equations are problematic in a simulation but rather some discontinuity in the model that causes the step-size h to assume small values. But if h is small, when advancing the simulation the position constraint violation stays very small, and the value of \mathbf{c}_2 in Eq. (11) always remains very reasonable.

With the corrections computed as the solution of the linear system of Eq. (10), the numerical solution is improved at each iteration as $\ddot{\mathbf{q}}^{(k+1)} = \ddot{\mathbf{q}}^{(k)} + \Delta\ddot{\mathbf{q}}^{(k)}$, $\lambda^{(k+1)} = \lambda^{(k)} + \Delta\lambda^{(k)}$, $\mathbf{V}^{(k+1)} = \mathbf{V}^{(k)} + \Delta\mathbf{V}^{(k)}$, $\mathbf{F}^{(k+1)} = \mathbf{F}^{(k)} + \Delta\mathbf{F}^{(k)}$. The following sections present in detail the answer to three key questions; (a) when is the computed solution accurate enough, (b) how to select the integration step-size h , and (c) when to stop the Newton-like iterative process that computes at each integration step the unknowns $\ddot{\mathbf{q}}$, λ , \mathbf{F} , and \mathbf{V} . Recall that once $\dot{\mathbf{q}}$ is available, Eqs. (2a), (2b) are used to evaluate \mathbf{q} , and $\dot{\mathbf{q}}$, respectively.

IMPLEMENTATION DETAILS FOR PROPOSED ALGORITHM

Estimating the local integration error in the HHT method

Since an approximation of the *global* error at time t_{n+1} can not be obtained in general, the goal is to produce an approximation of the *local* integration error in advancing the simulation from step n to $n+1$. Once the *local* integration error is avail-

able, an algorithm is implemented to ensure that this error stays smaller than a user prescribed tolerance.

The strategy for estimating the local integration error in positions is based on a linearization of the equations of motion in Eq. (6d) along with an asymptotic expansion of the solution \mathbf{q} . The approximation of this *local* integration error for HHT method is similar to the approach proposed in [25] for the Newmark method. The discussion is going to focus on Eq. (4a), since locally a linearization of Eq. (6d) leads to the previous form. For the purpose of computing the local integration error, the usual assumption is that the configuration at time t_n ; *i.e.*, $(\mathbf{q}_n, \dot{\mathbf{q}}_n, \ddot{\mathbf{q}}_n)$, is perfectly consistent. That is, it satisfies the equations of motion, along with the time derivatives of the equations of motion. The focus is exclusively on computing the error associated with advancing the simulation from t_n to t_{n+1} using the HHT method. Since the configuration is considered to be consistent at time t_n , it will satisfy the equations of motion, that is,

$$\mathbf{M}\ddot{\mathbf{q}}_n + \mathbf{C}\dot{\mathbf{q}}_n + \mathbf{K}\mathbf{q}_n = \mathbf{F}_n \quad (12)$$

By the same token, it will also satisfy the first time derivative of the equations of motion at time t_n , that is,

$$\mathbf{M}\ddot{\dot{\mathbf{q}}}_n + \mathbf{C}\dot{\mathbf{q}}_n + \mathbf{K}\dot{\mathbf{q}}_n = \dot{\mathbf{F}}_n \quad (13)$$

where $\ddot{\dot{\mathbf{q}}}_n$ formally represents the time derivative of the acceleration at time t_n . The fact that this quantity is typically not available is set aside for the time being and revisited later.

The Newmark integration formula of Eq. (2) is rewritten in the equivalent form

$$\mathbf{q}_{n+1} = \mathbf{q}_n + h\dot{\mathbf{q}}_n + \frac{h^2}{2}\ddot{\mathbf{q}}_n + \beta h^2 \mathbf{x} \quad (14a)$$

$$\dot{\mathbf{q}}_{n+1} = \dot{\mathbf{q}}_n + h\ddot{\mathbf{q}}_n + h\gamma \mathbf{x} \quad (14b)$$

$$\ddot{\mathbf{q}}_{n+1} = \ddot{\mathbf{q}}_n + \mathbf{x} \quad (14c)$$

The quantity of interest is now the unknown \mathbf{x} , the change in the value of acceleration from time t_n to t_{n+1} , which becomes central to the task of computing an estimation of the *local* integration

error

$$\delta_{n+1} = \mathbf{q}_{n+1} - \tilde{\mathbf{q}}_{n+1}. \quad (15)$$

Here $\tilde{\mathbf{q}}_{n+1}$ is the *exact* solution of the initial value problem (IVP)

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{C}\dot{\mathbf{q}} + \mathbf{K}\mathbf{q} = \mathbf{F} \quad (16)$$

that starts in the configuration $(\mathbf{q}_n, \dot{\mathbf{q}}_n, \ddot{\mathbf{q}}_n)$ at $t = t_n$. Using Taylor's Theorem, $\tilde{\mathbf{q}}_{n+1}$ is obtained as

$$\tilde{\mathbf{q}}_{n+1} = \mathbf{q}_n + h\dot{\mathbf{q}}_n + \frac{h^2}{2}\ddot{\mathbf{q}}_n + \frac{h^3}{6}\dddot{\mathbf{q}}_n + O(h^4) \quad (17)$$

The local integration error δ_{n+1} becomes available as soon the acceleration correction \mathbf{x} is available. In order to obtain an estimation for \mathbf{x} , from Eqs. (14) and (4a),

$$\begin{aligned} & \mathbf{M}(\ddot{\mathbf{q}}_n + \mathbf{x}) + (1 + \alpha)[\mathbf{C}(\dot{\mathbf{q}}_n + h\ddot{\mathbf{q}}_n + h\gamma\mathbf{x}) \\ & + \mathbf{K}(\mathbf{q}_n + h\dot{\mathbf{q}}_n + \frac{h^2}{2}\ddot{\mathbf{q}}_n + \beta h^2\mathbf{x})] - \alpha(\mathbf{C}\dot{\mathbf{q}}_n + \mathbf{K}\mathbf{q}_n) \\ & = \mathbf{F}_n + (1 + \alpha)\dot{\mathbf{F}}_n h + O(h^2) \end{aligned} \quad (18)$$

where Taylor's theorem was used to expand $\mathbf{F}(t_n + (1 + \alpha)h)$. Using Eqs. (12) and (13), and ignoring the $O(h^2)$ term $\mathbf{K}\frac{h^2}{2}\ddot{\mathbf{q}}_n$ leads to

$$[\mathbf{M} + (1 + \alpha)h\gamma\mathbf{C} + (1 + \alpha)\beta h^2\mathbf{K}]\mathbf{x} = (1 + \alpha)\mathbf{M}\ddot{\mathbf{q}}_n h + O(h^2) \quad (19)$$

Denoting $\mathbf{D} = \mathbf{M} + (1 + \alpha)h\gamma\mathbf{C} + (1 + \alpha)\beta h^2\mathbf{K}$, since $\mathbf{D}^{-1} = \mathbf{M}^{-1} + O(h) \cdot \mathbf{I}_p$ the equation

$$\mathbf{D}\mathbf{x} = (1 + \alpha)\mathbf{M}\ddot{\mathbf{q}}_n h + O(h^2) \quad (20)$$

leads to

$$\mathbf{x} = (1 + \alpha)\ddot{\mathbf{q}}_n h + O(h^2) \quad (21)$$

Using Eq. (14a),

$$\mathbf{q}_{n+1} = \mathbf{q}_n + h\dot{\mathbf{q}}_n + \frac{h^2}{2}\ddot{\mathbf{q}}_n + \beta h^3(1 + \alpha)\ddot{\mathbf{q}}_n + O(h^4) \quad (22)$$

Based on Eqs. (22) and (17)

$$\delta_{n+1} = \mathbf{q}_{n+1} - \tilde{\mathbf{q}}_{n+1} = h^3 \left[\beta(1 + \alpha) - \frac{1}{6} \right] \cdot \ddot{\mathbf{q}}_n + O(h^4) \quad (23)$$

Substituting for $\ddot{\mathbf{q}}_n$ from Eq. (21) and dropping the higher order terms leads to

$$\delta_{n+1} = \left[\beta - \frac{1}{6(1 + \alpha)} \right] h^2 \mathbf{x} \quad (24)$$

which provides an effective way of computing the local integration error since all the quantities that enter this equation are available at the end of the corrector stage.

The Accuracy Test

With the local truncation error in positions \mathbf{q} obtained as indicated in Eqs. (24) the numerical integrator needs to certify at time t_{n+1} the accuracy of the newly computed solution. With $\mathbf{q} \in \mathbb{R}^p$, using for $1 \leq i \leq p$ the notation $c_i = \left[\beta - \frac{1}{6(1 + \alpha)} \right] \ddot{\mathbf{q}}_{i,n}$, and dropping the terms of order h^3 or higher in Eq. (23) yields

$$\delta_{i,n+1} = c_i \cdot h^3 \quad (25)$$

Since c_i is not known, the integration error at the end of the one time step is actually approximated as

$$\delta_{i,n+1} = \left[\beta - \frac{1}{6(1 + \alpha)} \right] h^2 \cdot \mathbf{x}_i \quad 1 \leq i \leq p \quad (26)$$

Based on this value of the local integration error, a *composite error* first needs to be defined. The proposed form is

$$e \equiv \sqrt{\frac{1}{p} \left[\left(\frac{\delta_{1,n+1}}{Y_1^q} \right)^2 + \dots + \left(\frac{\delta_{p,n+1}}{Y_p^q} \right)^2 \right]} \quad (27)$$

where $Y_i^q = \max(1, \max_{j=1, \dots, n+1} |q_{i,j}|)$, and $\delta_{i,n+1}$, $1 \leq i \leq p$,

is the i^{th} component of the quantity computed in Eq. (26). The composite error is compared with the user prescribed error denoted here by ϵ . If $e \leq \epsilon$ the integration time step is accepted, otherwise it is rejected. Using the notation

$$\Psi \equiv \frac{p\epsilon^2}{\left[\beta - \frac{1}{6(1+\alpha)}\right]^2} \quad (28)$$

the error test is equivalently expressed as

$$\|\mathbf{x}\|^2 \leq \frac{\Psi}{h^4}, \quad (29)$$

where $\|\cdot\|$ represents a weighted norm [26] defined as $\|\mathbf{x}\| \equiv \left[\sum_{i=1}^p \left(\frac{x_i}{Y_i^q}\right)^2\right]^{\frac{1}{2}}$. If the condition of Eq. (29) is satisfied at the end of the corrector stage, the integration step is accepted, otherwise it is rejected.

The Step-Size Selection

Step-size selection plays a central role in the numerical integration algorithm. If $e_{new} \ll \epsilon$, effort is wasted in computing a solution that unnecessarily exceeds the user demanded accuracy. In this case, a more aggressive step-size selection would result in larger values for h with potentially big CPU savings at the end of the simulation. At the other end of the spectrum, a step-size selection mechanism that is too aggressive leads to many integration time steps at the end of which the user accuracy demands are not met. The effort to perform such an integration step is wasted, as the integration step is discarded for a new attempt with a more conservative step-size h . To strike the right note, every time the integration step-size is chosen the goal is for the error at the end of the integration step to be precisely equal to the one deemed acceptable by the user, and quantitatively defined by ϵ .

Key in the selection of a new step-size is Eq. (25), which indicates that the composite error is proportional to the cube of the step-size h . Ideally, the new step-size h_{new} is selected such that

$$\epsilon = h_{new}^3 \left[\frac{1}{p} \sum_{i=1}^p \left(\frac{C_i}{Y_i}\right)^2 \right]^{\frac{1}{2}}$$

Therefore, $\frac{\epsilon}{\epsilon} = \frac{h^3}{h_{new}^3}$, from where

$$h_{new} = h \frac{\epsilon^{\frac{1}{3}}}{\left\{ \left[\frac{1}{p} \sum_{i=1}^p \left(\frac{\delta_{i,n+1}}{Y_i}\right)^2 \right]^{\frac{1}{2}} \right\}^{\frac{1}{3}}} \quad (30)$$

Based on Eq. (24),

$$h_{new} = h \frac{s \cdot \epsilon^{\frac{1}{3}}}{\left[\frac{1}{\sqrt{p}} \left(\beta - \frac{1}{6(1+\alpha)} \right) h^2 \cdot \|\mathbf{x}\| \right]^{\frac{1}{3}}}$$

where a safety factor $s = 0.9$ was used to scale the value of the new step-size [9].

Defining $\Theta = \frac{\|\mathbf{x}\|^2 \cdot h^4}{\Psi}$, the previous equation finally leads to

$$h_{new} = \frac{s h}{\Theta^{\frac{1}{6}}} \quad (31)$$

The Correction Stage

It remains to determine a sensible stopping criteria for the corrector iterative process, and the issue is how accurate \mathbf{x} of Eqs. (14) should be computed. This quantity is obtained as the solution of an iterative Newton-like algorithm that requires at least one evaluation of the residuals in Eq. (11), followed by a forward/backward substitution to retrieve the corrections in the unknowns. However, one corrector iteration might be as expensive as doing all of the above but preceded by a full blown evaluation and factorization of the coefficient matrix of the linear system of Eq. (10). These operations are expensive and should be kept to a minimum.

It is important to get an accurate solution of the non-linear discretization system as a sloppy solution would adversely impact the quality and stability of the numerical solution. In this context, suppose that \mathbf{x} is approximated by $\mathbf{x}^{(k)}$, *i.e.*, the value obtained after k corrector iterations. Equation (26) indicates that the integration error e is computed based on the value \mathbf{x} , and therefore $\mathbf{x}^{(k)}$ will lead to a value of the error $e^{(k)}$. Thus, it is important to have a good approximation $\mathbf{x}^{(k)}$ for \mathbf{x} , if the algorithm is to produce a meaningful approximation $e^{(k)}$ of the composite integration error e .

A second reason for having an accurate solution is that the stability and convergence results associated with a numerical integrator are derived under the assumption that the numerical solution is computed to the specifications of the integration formula; *i.e.*, there is no room left for errors in finding the numerical solution at

the end of one integration step. Finding an approximate solution translates into solving a different initial value problem, which can be close or far from the original problem based on how accurate the non-linear system of Eq. (9) is solved, and the nature of the original initial value problem itself.

The corrector stopping criteria adopted here is that the relative difference between e and $e^{(k)}$ will be smaller than a threshold value denoted by c . A typical value recommended in the literature is $c = 0.001$ [9].

The local integration error at the end of one time step was shown to be $e = \left[\beta - \frac{1}{6(1+\alpha)} \right] \frac{h^2}{\sqrt{p}} \|\mathbf{x}\|$. After iteration k , the approximation obtained is $e^{(k)} = \left[\beta - \frac{1}{6(1+\alpha)} \right] \frac{h^2}{\sqrt{p}} \|\mathbf{x}^{(k)}\|$. The question is what should k be such that $e^{(k)}$ is close to e within 0.1% ($c = 0.001$); *i.e.*,

$$\left| \frac{e - e^{(k)}}{e} \right| \leq c \quad (32)$$

Since e is not available, the test above is replaced by

$$\left| \frac{e - e^{(k)}}{\varepsilon} \right| \leq c \quad (33)$$

where ε is the user prescribed error. Note that the goal of the step-size control is to keep e as close as possible to ε , so replacing Eq. (32) with Eq. (33) is acceptable. Therefore,

$$|e - e^{(k)}| \leq \left| \beta - \frac{1}{6(1+\alpha)} \right| \frac{h^2}{\sqrt{p}} \|\mathbf{x} - \mathbf{x}^{(k)}\| \quad (34)$$

and an approximation for $\|\mathbf{x} - \mathbf{x}^{(k)}\|$ is needed. Since for the Newton-like method employed the convergence is linear, there is a constant ξ that for convergence must satisfy $0 \leq \xi < 1$, such that [23]

$$\|\Delta \mathbf{x}^{(k+1)}\| \leq \xi \cdot \|\Delta \mathbf{x}^{(k)}\| \quad (35)$$

where $\Delta \mathbf{x}^{(k)}$ represents the correction at iteration k , $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \Delta \mathbf{x}^{(k)}$. Then,

$$\mathbf{x}^{(k)} - \mathbf{x} = \left[\mathbf{x}^{(k)} - \mathbf{x}^{(k+1)} \right] + \left[\mathbf{x}^{(k+1)} - \mathbf{x}^{(k+2)} \right] + \dots$$

$$\begin{aligned} \Rightarrow \|\mathbf{x}^{(k)} - \mathbf{x}\| &\leq \|\Delta \mathbf{x}^{(k)}\| + \|\Delta \mathbf{x}^{(k+1)}\| + \dots \\ &\leq \|\Delta \mathbf{x}^{(k)}\| (1 + \xi + \xi^2 + \dots) \end{aligned}$$

which leads to

$$\|\mathbf{x} - \mathbf{x}^{(k)}\| \leq \|\Delta \mathbf{x}^{(k)}\| \cdot \frac{1}{1 - \xi} \quad (36)$$

The value ξ is going to be approximated by

$$\xi \approx \xi_k = \frac{\|\Delta \mathbf{x}^{(k)}\|}{\|\Delta \mathbf{x}^{(k-1)}\|} \quad (37)$$

Based on Eq. (36)

$$\|\mathbf{x} - \mathbf{x}^{(k+1)}\| \leq \|\Delta \mathbf{x}^{(k)}\| \cdot \frac{\xi}{1 - \xi} \quad (38)$$

which indicates that the distance between the true solution \mathbf{x} , and the approximation of the solution obtained after applying k corrections; *i.e.* $\mathbf{x}^{(k+1)}$, is bounded from above by a quantity proportional to the norm of the most recent correction $\|\Delta \mathbf{x}^{(k)}\|$. Going back to Eq. (34),

$$|e - e^{(k)}| \leq \left| \beta - \frac{1}{6(1+\alpha)} \right| \frac{h^2}{\sqrt{p}} \|\Delta \mathbf{x}^{(k)}\| \cdot \frac{\xi}{1 - \xi} \quad (39)$$

The condition of Eq. (33) is then satisfied as soon as

$$\frac{1}{\varepsilon} \left| \beta - \frac{1}{6(1+\alpha)} \right| \frac{h^2}{\sqrt{p}} \|\Delta \mathbf{x}^{(k)}\| \frac{\xi}{1 - \xi} \leq c$$

Equivalently,

$$\left(\frac{\xi}{1 - \xi} \right)^2 \|\Delta \mathbf{x}^{(k)}\|^2 \leq c^2 \cdot \frac{\Psi}{h^4} \quad (40)$$

Note that at the right of the inequality sign are quantities that remain constant during the corrector iterative process, while at the left are quantities that change at each iteration. Likewise, note that the stopping criteria of Eq. (40) can only be used at the end of the second iteration since it is only then that an approximation of the convergence rate ξ can be produced. In other words, the proposed approach will not be able to stop the iterative process

after the first iteration. This is not a matter of great concern since models as simple as a one body pendulum are already non-linear.

The Prediction Stage

For the Newton-like algorithm used to find the solution of Eq. (11), a good starting point is essential both for convergence, and reducing the effort to find the approximation of the solution at time t_{n+1} . In [4], the generalized accelerations prediction is obtained by taking $\ddot{\mathbf{q}}_{n+1} = \ddot{\mathbf{q}}_n$, which is equivalent to setting $\mathbf{x} = \mathbf{0}$ in Eq. (14). This strategy is replaced by one based on polynomial extrapolation, in which currently a polynomial of order up to three is used to produce an initial guess for all the unknowns. The approach used in [4] is thus obtained by setting the degree of the interpolation polynomial to zero. The polynomial extrapolation is based on Newton divided differences and it uses Horner's scheme for evaluation of the interpolant at time t_{n+1} [26]. The degree of the interpolant can be adjusted on the fly, and the mechanism to control this takes into account the smoothness of the solution.

Summary of key HHT Formulas

The answers to the questions (a) what is the stopping criteria for the non-linear discretization algebraic system, (b) how is the integration error computed, and (c) how is the step-size controlled, is summarized below.

Notation:

$$\Psi \equiv \frac{p\epsilon^2}{\left[\beta - \frac{1}{6(1+\alpha)}\right]^2} \quad \Theta = \frac{\|\mathbf{x}\|^2 \cdot h^4}{\Psi} \quad (41a)$$

Prediction: Performed based on divided differences (Newton interpolation and Horner's scheme for extrapolation at t_{n+1}).

Correction: Linear convergence rate allows for computation of ξ (see Eq. (37)). Stopping criteria:

$$\left(\frac{\xi}{1-\xi}\right)^2 \|\Delta \mathbf{x}^{(k)}\|^2 \leq c^2 \frac{\Psi}{h^4}, \quad (c = 0.001) \quad (41b)$$

Accuracy Check: Performed after corrector converged,

$$\Theta \leq 1 \quad (41c)$$

Step-Size Selection: With a safety factor $s = 0.9$,

$$h_{new}^{(q)} = \frac{s h}{\Theta^{\frac{1}{6}}} \quad (41d)$$



Figure 1. ALL TERRAIN VEHICLE.

Finally, dense output is supported for the differential variables $\dot{\mathbf{q}}$ by linear interpolation, and third order Hermite interpolation [26], for $\dot{\mathbf{q}}$, and \mathbf{q} .

NUMERICAL EXPERIMENTS

The proposed algorithm has been extensively tested with more than 1,600 mechanical systems of various complexity. In this paper one model containing a flexible body will be used to compare the results and efficiency of the HHT and GSTIFF integrators. The latter is the default integrator in the ADAMS [24] simulation package.

An All-Terrain-Vehicle Model with Flexible Chassis

The model in Fig. 1 is an all-terrain vehicle (ATV) with a flexible frame. Beneath the ATV is a 4-post shaker device, which is used to simulate a durability event represented by a simple-harmonic translational displacement constraint at each of the four posts

The frame component in Fig. 1 is an MSC.ADAMS flexible body modal representation [27] that was created with the MSC.Patran finite element (FE) package; it contains 134 modes (from 56.7 Hz to 13.1 kHz). The von Mises stresses in the flexible body will be recovered with ADAMS/Durability to identify critical stress locations in the frame.

Front and rear suspension linkages, which utilize rigid parts and spring damper force components to provide stiffness and damping effects, are connected to the frame with idealized joints and joint primitives (spherical, revolute, inline, etc.). The steering system has a motion constraint that applies a rotational displacement function, causing the front wheels to turn left-to-right in a sinusoidal fashion. The tires of the vehicle interact with the shaker by means of 3-dimensional solid-to-solid contact forces. A rider weight of 230 pounds has been approximated with lumped masses and distributed between the steering column assembly and frame as 30 pounds and 200 pounds, respectively. Remaining parts in the model that are attached to the frame, such as the engine, are modeled as lumped masses and are connected

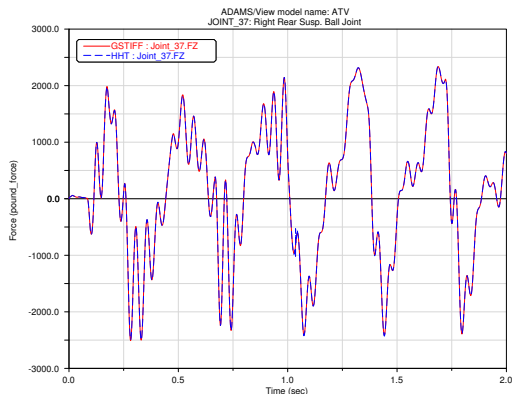


Figure 2. COMPARISON OF VERTICAL REACTION FORCE IN REAR SUSPENSION BALL JOINT.

using fixed joints.

Altogether there are 28 moving parts, 43 joints, 5 motions and 1 flexible body leading to a total number of 532 equations for the HHT integrator, and 927 equations for the Index-3 GSTIFF (I3) formulation. The MSC.ADAMS modeling units used for this experiment were pound_force, pound_mass, inch, and second. The duration of the simulation was 2.0 seconds, which allows the ATV to move up and down several times. The GSTIFF integrator was run with $ERROR=1e-4$, while HHT was run with $ERR=1e-5$. A maximum time step ($HMAX=5e-4$) was specified for each integrator, allowing the MSC.ADAMS output step size to be somewhat coarser ($4e-3$) so as to help minimize the burden of downstream stress-recovery hotspots computations during results post-processing operations.

The simulation CPU time for GSTIFF was 367.08 seconds, while HHT completed the simulation in 122.47 seconds. The simulation was run using MSC.ADAMS 2005 on a Windows 2000 laptop computer with a single 2.20 GHz Pentium4 CPU and 1Gb RAM.

There is excellent agreement in location of critical stresses as obtained for this simulation from ADAMS/Durability and presented in Table 1; the difference in peak stress between the two sets of results is less than 0.5%. The table lists the first 10 hotspots for this model. What is remarkable is that both integrators produced the same top ten list, the nodes being provided in the first column of the table. The second and third columns of the table contain the value of the stress in GSTIFF and HHT, expressed in $lbf/inch^2$. The next column contains the relative difference in values; the last two columns contain the simulation time at which the corresponding stress value was observed. Note that GSTIFF and HHT are identical in this regard.

The Z-component of the reaction force in the spherical joint that connects the right half of the rear suspension component to the frame is presented in Fig. 2. The plot shows good correlation between the force results obtained with the GSTIFF and HHT integrators.

Table 1. COMPARISON OF TOP 10 VON MISES STRESS HOTSPOTS

POINT NO.	Stress GSTIFF	Stress HHT	RelErr [%]	Time[s] GSTIFF	Time[s] HHT
23956	300738	300768	0.01%	1.104	1.104
26654	204503	204274	-0.11%	0.746	0.746
33918	204075	204191	0.06%	0.206	0.206
46577	200688	200354	-0.17%	0.746	0.746
34560	198106	197887	-0.11%	0.211	0.211
46580	193729	192929	-0.41%	0.326	0.326
30060	190386	189911	-0.25%	0.692	0.692
24704	173882	173355	-0.30%	0.281	0.281
36156	171990	172746	0.44%	1.446	1.446
23007	170191	170375	0.11%	1.100	1.100

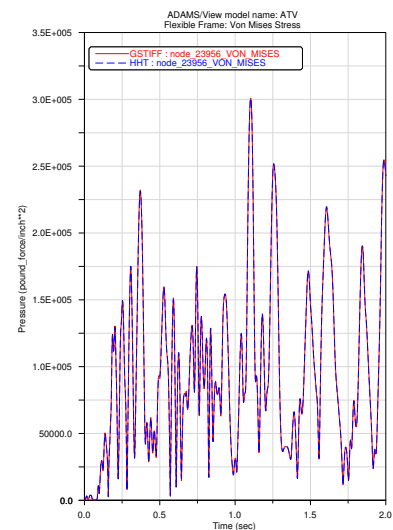


Figure 3. COMPARISON OF STRESS AT HOTSPOT NODES ON FLEXIBLE FRAME.

The von Mises stress histories at two of the hotspot locations are shown in Fig. 3. The plots once again show good correlation between integrators.

Finally, Fig. 4 presents the time variation of the angular velocity of the engine assembly, and is an indicator of the severity of the ATV pitching behavior. Figure 5 confirms that the difference between the angular velocity computed with GSTIFF and HHT integrators is less than 1.2%.

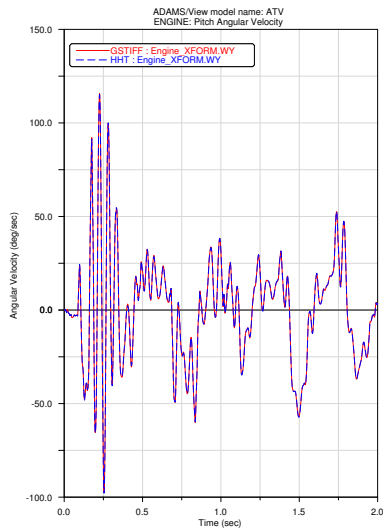


Figure 4. ANGULAR VELOCITY FOR ENGINE PITCH.

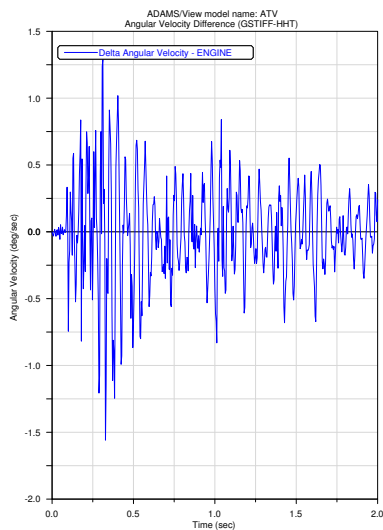


Figure 5. GSTIFF-HHT ANGULAR VELOCITY DIFFERENCE FOR ENGINE PITCH.

CONCLUSIONS

The HHT method used in structural dynamics was adapted in this paper for the numerical solution of index 3 differential algebraic equations of multi-body dynamics. Strategies for corrector stopping criteria, and error and step-size control were presented in detail. An all-terrain-vehicle model indicates that simulation speed is about three times faster when compared with the default BDF-based integrator used in ADAMS [24]. An explanation for the improved performance is based on three key observations. (a) The most time consuming part of simulation is the computation of the Jacobian associated with the non-linear discretization system. The proposed algorithm contains heuristics

to reduce as much as possible the number of Jacobian evaluations. Unlike the reference BDF integrator in which terms of the integration Jacobian can become disproportionately large as a result of a scaling by the inverse of the step-size, the proposed integrator employs a different approach where certain values are multiplied (never divided) by the step-size prior to populating the Jacobian. As long as the step-size does not significantly change over several consecutive time steps, this approach better supports the recycling of the Jacobian. (b) When compared with the BDF Jacobian, the HHT Jacobian is numerically better conditioned, which leads for large problems to more reliable corrections in the Newton-like iterative approach. Typically, this results in a smaller number of corrector iterations. This desirable attribute is further enhanced by the following observation; since certain partial derivatives are scaled by the step-size h , or by h^2 prior to populating the Jacobian, small errors in these partial derivatives are going to have a lesser negative effect on the overall quality of the Jacobian. (c) On one hand, the BDF formulas of order higher than one contain regions of instability in the left plane. The higher the order, the smaller the region of stability. On the other hand, BDF intrinsically is designed to maximize the integration order/step-size. Due to these two conflicting attributes, particularly for models that are mechanically stiff (models with stiff springs, flex bodies that lead to systems with large eigenvalues close to the imaginary axis) there are many instances when an order/step-size choice lands the BDF integrator outside the stability region [9]. These integration time-steps typically end up being rejected, and smaller step-sizes are required to advance the simulation. This is a non-issue with the HHT method, which is a fixed low order method with good stability properties in the whole left plane.

It should be pointed out that there are situations where BDF type formulas are going to work significantly faster. These are the cases where BDF can sustain a high integration order throughout the simulation. If the model simulated allows BDF to work at order 5 or 6, the HHT method can not produce a solution of similar quality in comparable CPU time because the low order integration formulas employed. However, this scenario is not common, as most real life large models contain discontinuities or stiff mechanical components that typically limit the BDF integration order to 1 or 2.

The accuracy of the results is good, occasional spikes in accelerations and reaction forces being explained by the use of a variable step integration algorithm for the solution of an index 3 DAE problem, an operation that is conjectured in [7] to further reduce the order of an already low order method. Quantitatively, the simulation results can be improved by decreasing the user-specified integration error; qualitatively, the results could be improved by (a) using a Runge-Kutta method as proposed in [19], (b) using the generalization of the HHT method as proposed in [29], or (c) reducing the index of the problem in an approach similar to the one proposed in [30].

REFERENCES

- [1] Hilber, H. M., Hughes, T. J. R., and Taylor, R. L., 1977. "Improved numerical dissipation for time integration algorithms in structural dynamics". *Earthquake Eng. and Struct. Dynamics*, **5**, pp. 283–292.
- [2] Newmark, N. M., 1959. "A method of computation for structural dynamics". *Journal of the Engineering Mechanics Division, ASCE*, pp. 67–94.
- [3] Ascher, U. M., and Petzold, L. R., 1998. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. SIAM, Philadelphia, PA.
- [4] Hughes, T. J. R., 1987. *Finite Element Method - Linear Static and Dynamic Finite Element Analysis*. Prentice-Hall, Englewood Cliffs, New Jersey.
- [5] Shabana, A. A., 1994. *Computational Dynamics*. John Wiley & Sons.
- [6] Haug, E. J., 1989. *Computer-Aided Kinematics and Dynamics of Mechanical Systems*. Prentice-Hall, Englewood Cliffs, New Jersey.
- [7] 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.
- [8] Petzold, L. R., 1982. "Differential-algebraic equations are not ODE's". *SIAM J. Sci., Stat. Comput.*, **3**(3).
- [9] Hairer, E., and Wanner, G., 1991. *Solving Ordinary Differential Equations*, Vol. II of *Computational Mathematics*. Springer-Verlag.
- [10] Potra, F. A., 1993. "Implementation of linear multi-step methods for solving constrained equations of motion". *SIAM. Numer. Anal.*, **30**(3).
- [11] Lubich, C., Nowak, U., Pohle, U., and Engstler, C., 1995. "Mexx - numerical software for the integration of constrained mechanical multibody systems". *Mechanics of Structures and Machines*, **23**, pp. 473–495.
- [12] Eich-Sollner, E., and Fuhrer, C., 1998. *Numerical Methods in Multibody Dynamics*. Teubner-Verlag, Stuttgart.
- [13] Cardona, A., and Geradin, M., 1994. "Numerical integration of second order differential-algebraic systems in flexible mechanics dynamics". In *Computer-Aided Analysis of Rigid and Flexible Mechanical Systems*, M. F. O. S. Pereira and J. A. C. Ambrosio, eds., Kluwer Academic Publishers.
- [14] Yen, J., Petzold, L., and Raha, S., 1996. A time integration algorithm for flexible mechanism dynamics: the DAE α -method. Tech. Rep. TR96-024, Dept. of Comp. Sci., University of Minnesota.
- [15] de Jalon, J. G., and Bayo, E., 1994. *Kinematic and Dynamic Simulation of Multi-body Systems. The real time challenge*. Springer-Verlag, Berlin.
- [16] Cuadrado, J., Dopico, D., Naya, M. N., and Gonzalez, M., 2004. "Penalty, semi-recursive and hybrid methods for MBS Real-Time dynamics in the context of structural integrators". *Multibody Systems Dynamics*(12), pp. 117–132.
- [17] Baumgarte, J., 1972. "Stabilization of constraints and integrals of motion in dynamical systems". *Computer Methods in Applied Mechanics and Engineering*, **1**, pp. 1–16.
- [18] Owren, B., and Simonsen, H., 1995. "Alternative integration methods for problems in structural dynamics". *Structural Dynamics. Comput. Meth. in Appl. Mech. and Eng.*, **122**, pp. 1–10.
- [19] Jay, L. O., 1998. "Structure preservation for constrained dynamics with super partitioned additive runge-kutta methods". *SIAM J. Sci. Comput.*, **20**, pp. 416–446.
- [20] Schaub, M., and Simeon, B., 2002. "Automatic h-scaling for the efficient time integration of stiff mechanical systems". *Multibody System Dynamics*, **8**, pp. 329–345.
- [21] Lötstedt, C., and Petzold, L., 1996. "Numerical solution of nonlinear differential equations with algebraic constraints I: Convergence results for backward differentiation formulas". *Mathematics of Computation*, **174**, pp. 491–516.
- [22] Brenan, K., and Engquist, B. E., 1988. "Backward differentiation approximations of nonlinear differential/algebraic systems". *Mathematics of Computation*, **51**(184), pp. 659–676.
- [23] Dennis, J., and Schnabel, R., 1983. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, Englewood Cliffs, NJ.
- [24] MSCsoftware, 2005. ADAMS User Manual.
- [25] Zienkiewicz, O., and Xie, Y., 1991. "A simple error estimator and adaptive time-stepping procedure for dynamic analysis". *Earthquake Engineering and Structural Dynamics*, **20**, pp. 871–887.
- [26] Atkinson, K. E., 1989. *An introduction to numerical analysis*, second ed. John Wiley & Sons Inc., New York.
- [27] Craig, R. R., and Bampton, M. C. C., 1968. "Coupling of substructures for dynamics analyses". *AIAA Journal*, **6**(7), pp. 1313–1319.
- [28] Gear, C. W., 1971. *Numerical Initial Value Problems of Ordinary Differential Equations*. Prentice-Hall, Englewood Cliffs, NJ.
- [29] Chung, J., and Hulbert, G. M., 1993. "A time integration algorithm for structural dynamics with improved numerical dissipation: the generalized- α method". *Transactions of ASME, Journal of Applied Mechanics*, **60**(2), pp. 371–375.
- [30] Gear, C. W., Gupta, G., and Leimkuhler, B., 1985. "Automatic integration of the eulerlagrange equations with constraints". *J. Comp. Appl. Math.*, **12**, pp. 77–90.