

Variable Step Implicit Numerical Integration of Stiff Multibody Systems

D. Negrut*, E. J. Haug*, M. Iancu**

* - Department of Mechanical Engineering

** - Department of Mathematics

The University of Iowa, Iowa City, IA 52242

Abstract. This paper presents a variable step size implicit numerical integration algorithm for dynamic analysis of stiff multibody systems. Stiff problems are very common in real world applications, and their numerical treatment by means of explicit integration is cumbersome or infeasible. Until recently, implicit numerical integration of the equations of motion of stiff mechanical systems has been problematic. Prior attempts to solve this problem stopped at the level of theoretical considerations, or resulted in numerical methods that lacked generality and robustness. A better theoretical understanding of the process of state space reduction, along with the ability to efficiently generate the needed derivative information for implicit integration, has allowed for the development of robust and general implicit numerical methods. The benefit of variable step size implicit integration is shown using a stiff double pendulum model. An experimental general purpose code that performs implicit integration is capable to successfully treat a large class of rigid body models.

1 The DAE of Multibody Dynamics

Dynamic analysis of a constrained mechanical system model requires the numerical integration of a set of differential-algebraic equations (DAE). This type of problem is known to exhibit numerical difficulties, and it is prone to intense computation effort. There are many specific methods for the numerical solution of DAE of multibody dynamics, some of them outlined in greater detail by Potra (1994) and Haug et al. (1997, b).

In this paper, the numerical solution of the differential-algebraic problem is obtained by reducing it to a set of ordinary differential equations (ODE). The reduction is based on state space parametrization, the evolution of the mechanical system being described by a number of independent variables equal to the number of degrees of freedom of the mechanical system. The existing theory for the numerical solution of ODE can then be employed for the solution of the resulting state space ordinary differential equations (SSODE). Once the solution of the SSODE is available, the homeomorphism induced by the parametrization considered is used to recover the solution of the differential-algebraic problem.

If $\mathbf{q} = [q_1, q_2, \dots, q_n]^T$ is the vector of generalized coordinates used to model a mechanical system subject to the constraints in Eq. 1, the parametrization considered in this paper is defined by a subset of the generalized coordinates. They are called independent generalized coordinates, and are denoted by $\mathbf{v} \in \mathfrak{R}^{\text{ndof}}$, where $\text{ndof} = n - m$.

$$\Phi(\mathbf{q}) \equiv [\Phi_1(\mathbf{q}), \Phi_2(\mathbf{q}), \dots, \Phi_m(\mathbf{q})]^T = \mathbf{0} \quad (1)$$

Details of the partitioning of \mathbf{q} into independent and dependent vectors \mathbf{v} and $\mathbf{u} \in \mathfrak{R}^m$, respectively, are provided by Haug et al. (1997,b).

Differentiating Eq. 1 with respect to time yields the kinematic velocity equation,

$$\Phi_{\mathbf{q}}(\mathbf{q})\dot{\mathbf{q}} = \mathbf{0} \quad (2)$$

where subscript denotes partial differentiation; i.e., $\Phi_{\mathbf{q}} = \left[\frac{\partial \Phi_1}{\partial q_j} \right]$, and an over dot denotes differentiation with respect to time. Finally, differentiating Eq. 2 with respect to time yields the kinematic acceleration equation,

$$\Phi_{\mathbf{q}}(\mathbf{q})\ddot{\mathbf{q}} = -(\Phi_{\mathbf{q}}\dot{\mathbf{q}})_{\mathbf{q}}\dot{\mathbf{q}} \equiv \boldsymbol{\tau}(\mathbf{q}, \dot{\mathbf{q}}) \quad (3)$$

Equations 1 through 3 characterize the admissible motion of the mechanical system.

The state of a mechanical system changes in time under the influence of both applied and constraint forces. The Lagrange multiplier form of the constrained equations of motion for mechanical systems is (Haug, 1989),

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \Phi_{\mathbf{q}}^T(\mathbf{q})\boldsymbol{\lambda} = \mathbf{Q}^A(t, \mathbf{q}, \dot{\mathbf{q}}) \quad (4)$$

The solution of Eq. 4 must satisfy Eqs.1 through 3; i.e., the kinematic constraint equations at the position, velocity, and acceleration levels. Direct integration of the ODE of Eqs.3 and 4 is not the proper way to treat the dynamic analysis problem, since there is no guarantee that the solution obtained will satisfy the constraint equations at the position and velocity levels. Generally, the solution obtained by direct integration will drift away from the manifold defined by the constraints of Eqs. 1 and 2.

In the generalized coordinate partitioning method (Haug et al., 1997,b), only independent coordinates are integrated at each time step. The dependent quantities; i.e. \mathbf{u} , $\dot{\mathbf{u}}$, $\ddot{\mathbf{u}}$, and $\boldsymbol{\lambda}$, are then recovered using the following partitioned form of the Eqs. 1 through 4:

$$\mathbf{M}^{vv}(\mathbf{u}, \mathbf{v})\ddot{\mathbf{v}} + \mathbf{M}^{vu}(\mathbf{u}, \mathbf{v})\ddot{\mathbf{u}} + \Phi_{\mathbf{v}}^T(\mathbf{u}, \mathbf{v})\boldsymbol{\lambda} = \mathbf{Q}^v(t, \mathbf{u}, \mathbf{v}, \dot{\mathbf{u}}, \dot{\mathbf{v}}) \quad (5)$$

$$\mathbf{M}^{uv}(\mathbf{u}, \mathbf{v})\ddot{\mathbf{v}} + \mathbf{M}^{uu}(\mathbf{u}, \mathbf{v})\ddot{\mathbf{u}} + \Phi_{\mathbf{u}}^T(\mathbf{u}, \mathbf{v})\boldsymbol{\lambda} = \mathbf{Q}^u(t, \mathbf{u}, \mathbf{v}, \dot{\mathbf{u}}, \dot{\mathbf{v}}) \quad (6)$$

$$\Phi_{\mathbf{v}}(\mathbf{u}, \mathbf{v})\ddot{\mathbf{v}} + \Phi_{\mathbf{v}}\ddot{\mathbf{u}} = \boldsymbol{\tau}(\mathbf{u}, \mathbf{v}, \dot{\mathbf{u}}, \dot{\mathbf{v}}) \quad (7)$$

$$\Phi_{\mathbf{v}}(\mathbf{u}, \mathbf{v})\dot{\mathbf{v}} + \Phi_{\mathbf{u}}(\mathbf{u}, \mathbf{v})\dot{\mathbf{u}} = \mathbf{0} \quad (8)$$

$$\Phi(\mathbf{u}, \mathbf{v}) = \mathbf{0} \quad (9)$$

In this approach, Eq. 5 is used to integrate the independent generalized coordinates, while Eqs. 6 through 9 are used to determine the dependent variables. Theoretically, based on the assumption that the constraints in Eq. 1 are independent at an assembled configuration $\mathbf{q}^0 = [\mathbf{u}^0, \mathbf{v}^0]$ of the mechanism, the implicit function theorem (Corwin and Szczerba, 1982), guarantees that Eq. 9 can be solved for \mathbf{u} as a differentiable function of \mathbf{v} ; i.e.,

$$\mathbf{u} = \mathbf{g}(\mathbf{v}) \quad (10)$$

where the function $\mathbf{g}(\mathbf{v})$ has as many continuous derivatives as does the constraint function $\Phi(\mathbf{q})$ in Eq. 9, and Eq. 10 is valid in some open neighborhood of \mathbf{u}^0 and \mathbf{v}^0 . Since, as a consequence of the partitioning algorithm the coefficient matrix of $\dot{\mathbf{u}}$ in Eq. 8 is nonsingular, $\dot{\mathbf{u}}$ can be determined uniquely as a function of \mathbf{v} and $\dot{\mathbf{v}}$, where Eq. 10 is used to eliminate explicit dependence on \mathbf{u} . Next, Eq. 7 can be used to uniquely determine $\ddot{\mathbf{u}}$ as a function of \mathbf{v} , $\dot{\mathbf{v}}$, and $\ddot{\mathbf{v}}$, where results of Eqs. 8 and 10 are substituted. Since the coefficient matrix of $\boldsymbol{\lambda}$ in Eq. 6 is nonsingular, $\boldsymbol{\lambda}$ can be determined uniquely as a function of \mathbf{v} , $\dot{\mathbf{v}}$, and $\ddot{\mathbf{v}}$, using previously derived results. Finally, each of the preceding results may be substituted into Eq. 5, yielding a second order SODE in \mathbf{v} (Haug, 1989),

$$\hat{\mathbf{M}}(\mathbf{u}, \mathbf{v})\ddot{\mathbf{v}} = \hat{\mathbf{Q}}(\mathbf{u}, \mathbf{v}, \dot{\mathbf{u}}, \dot{\mathbf{v}}) \quad (11)$$

where

$$\hat{\mathbf{M}} = \mathbf{M}^{vv} - \mathbf{M}^{vu}\Phi_{\mathbf{u}}^{-1}\Phi_{\mathbf{v}} - (\Phi_{\mathbf{u}}^{-1}\Phi_{\mathbf{v}})^T[\mathbf{M}^{uv} - \mathbf{M}^{uu}\Phi_{\mathbf{u}}^{-1}\Phi_{\mathbf{v}}] \quad (12)$$

$$\hat{\mathbf{Q}} = \mathbf{Q}^v - \mathbf{M}^{vu} \Phi_u^{-1} \tau + (\Phi_u^{-1} \Phi_v)^T [\mathbf{Q}^u - \mathbf{M}^{uu} \Phi_u^{-1} \tau] \quad (13)$$

As shown by Haug (1989), the coefficient matrix in Eq. 12 is nonsingular in some open neighborhood of \mathbf{v}^0 . Therefore, Eq. 1 can be solved for $\tilde{\mathbf{v}}$ and any explicit integrator can be employed to integrate for the independent velocities $\tilde{\mathbf{v}}$, and then the independent coordinates \mathbf{v} . Finally, Eqs. 8 and 9 allow for the computation of dependent velocities and positions respectively. Thus, one step of explicit integration is completed.

2 Implicit Integration of SODE

Implicit integration of the DAE of multibody dynamics follows the same theoretical development outlined in the preceding section for explicit integration. The DAE is reduced to an SODE, which in turn is integrated using an implicit formula. The difficulty that hinders the immediate application of this idea is the complicated form of the SODE in Eq. 11. An implicit integration formula applied to discretize the SODE in Eq. 11 would result in a set of nonlinear algebraic equations that must be solved at each time step. Whenever a Newton type method is employed for this purpose, the Jacobian needs to be provided and this is not readily available.

Practically, as in the case of explicit integration, the SODE is not explicitly reduced to the form in Eq. 11, rather the computation is focused on Eq. 5, the equation that after substitution of the dependent variables leads to Eq. 11. Discretization of Eq. 5 results in the following algebraic nonlinear system:

$$\begin{aligned} \Psi(\tilde{\mathbf{v}}_{n+1}) \equiv & \mathbf{M}^{vv}(\mathbf{u}_{n+1}, \mathbf{v}_{n+1}) \ddot{\mathbf{v}}_{n+1} + \mathbf{M}^{vu}(\mathbf{u}_{n+1}, \mathbf{v}_{n+1}) \ddot{\mathbf{u}}_{n+1} + \Phi_v^T(\mathbf{u}_{n+1}, \mathbf{v}_{n+1}) \lambda_{n+1} \\ & - \mathbf{Q}^v(\mathbf{u}_{n+1}, \mathbf{v}_{n+1}, \dot{\mathbf{u}}_{n+1}, \dot{\mathbf{v}}_{n+1}) = \mathbf{0} \end{aligned} \quad (14)$$

Without loss of generality, the implicit integration formulas used to discretize Eq. 5 are assumed to have the form

$$\mathbf{v}_{n+1} = \tilde{\mathbf{v}}_{n+1} + \beta h^2 \ddot{\mathbf{v}}_{n+1} \quad (15)$$

$$\dot{\mathbf{v}}_{n+1} = \tilde{\dot{\mathbf{v}}}_{n+1} + \gamma h \ddot{\mathbf{v}}_{n+1} \quad (16)$$

where $\tilde{\mathbf{v}}_{n+1}$ and $\tilde{\dot{\mathbf{v}}}_{n+1}$ are expressions depending on past values of positions and/or velocities, h is the current step size, and β and γ are coefficients depending on the integration formula being used. Any multistep implicit integration formula can be cast into the form of Eqs. 15 and 16. For a Runge-Kutta type integration formula, some slight modifications are necessary (Haug et al. 1997, a).

Since at each time step, the nonlinear system of Eq. 14 is solved for the independent accelerations using a Newton-type method, the Jacobian $\Psi_{\ddot{\mathbf{v}}}$ must be provided. As all the variables in Eq. 14 are functions of the independent accelerations, the chain rule of differentiation and Eqs. 6 through 9, and 15 and 16 are used to evaluate the Jacobian of the algebraic system.

Independent generalized positions and velocities depend on independent acceleration through the integration formulas of Eqs. 15 and 16. Their derivatives with respect to the independent accelerations are readily obtained as

$$\mathbf{v}_{\ddot{\mathbf{v}}} = \beta h^2 \mathbf{I} \quad (17)$$

$$\dot{\mathbf{v}}_{\ddot{\mathbf{v}}} = \gamma h \mathbf{I} \quad (18)$$

where the subscript $n+1$, which refers to the time step at which the solution is sought, is suppressed for convenience.

The derivatives of the dependent variables; i.e., positions, velocities, accelerations, and Lagrange multipliers, with respect to the independent accelerations are obtained by repeatedly applying the chain rule

of differentiation to Eqs. 6 through 9, along with the results in Eqs. 17 and 18. Details of the computations of these derivatives can be found in the work of Haug et al. (1997, b, c). Once the required derivatives are computed, they are substituted into the expression for the Jacobian $\Psi_{\dot{\mathbf{v}}}$, to obtain

$$\begin{aligned} \Psi_{\dot{\mathbf{v}}} = & \mathbf{M}^{vv} + \mathbf{M}^{vu}\mathbf{H} + \mathbf{H}^T[\mathbf{M}^{uv} + \mathbf{M}^{uu}\mathbf{H}] + \gamma h \{ \mathbf{M}^{vu}\mathbf{N} + \mathbf{H}^T\mathbf{S} - \mathbf{Q}_v^v - \mathbf{Q}_u^v\mathbf{H} \} \\ & + \beta h^2 \left\{ \left[(\mathbf{M}^{vv}\ddot{\mathbf{v}})_u + (\mathbf{M}^{vu}\ddot{\mathbf{u}})_u \right] \mathbf{H} + \left[(\mathbf{M}^{vv}\ddot{\mathbf{v}})_v + (\mathbf{M}^{vu}\ddot{\mathbf{u}})_v \right] + \mathbf{M}^{vu}\mathbf{L} + \mathbf{H}^T\mathbf{R} \right. \\ & \left. + (\Phi_v^T\lambda)_u \mathbf{H} + (\Phi_v^T\lambda)_v - \mathbf{Q}_u^v\mathbf{H} - \mathbf{Q}_v^v - \mathbf{Q}_u^v\mathbf{J} \right\} \end{aligned} \quad (19)$$

The matrix quantities in Eq. 19 are as defined by Haug et al. (1997, b). Serban and Haug (1997) have presented an effective way to compute derivative information required by the computation of the Jacobian in Eq. 19, for a large class of mechanical system. The basic idea of their approach is to create a library of joint and force elements. For these elements, the derivative information is generated once and afterwards used for complex mechanical models.

3 Numerical Considerations in Integration of DAE of Multibody Dynamics

The focus of this paper is on variable step implicit integration for the simulation of stiff mechanical systems. Two such implementations are available at this time; one based on a singly diagonal implicit Runge-Kutta method (SDIRK) of order 4 that is due to Hairer and Wanner (1996) and second based on the trapezoidal rule. The latter method is analyzed in this paper. In order to assess its performance, the proposed method is compared with two other existing alternatives for the integration of multibody dynamics problems. One is a constant step implementation of the same trapezoidal method, and the second is a constant step size, second order Adams-Bashforth-Moulton explicit method. The constant step size implementations are used to illustrate the advantages of using variable step size integration. The explicit integrator makes clear the limitations of this approach when simulating the motion of stiff mechanical systems.

3.1 Explicit Integration of SODE

The key issue in explicit integration of SODE is the computation of the independent accelerations. Once an efficient strategy for obtaining the independent accelerations is defined, any explicit integration formula can be employed to integrate for the independent velocities and positions. The dependent variables are then computed following the sequence described in Subsection 1.2. One step of integration is complete when the generalized positions and velocities have been computed.

There are several efficient ways in which the independent accelerations can be computed (Negrut et al., 1996, Serban et al., 1997). The concern of this paper is however with the behavior of the integrator when simulating stiff mechanical problems, and the interest primarily lies in monitoring step size limitations that are characteristic to the explicit integration of such problems. The details about obtaining the independent accelerations are thus skipped. Once the independent accelerations are available, the Adams-Bashforth-Moulton formulas (Atkinson, 1989) of Eq. 20 are used to integrate for the independent velocities and positions. The reason for choosing this integration formula is that, like the trapezoidal method, it is a second order formula.

$$y_{n+1}^{(0)} = y_n + hf(x_n, y_n) \quad (20')$$

$$y_{n+1} = y_n + \frac{h}{2} [f(x_{n+1}, y_{n+1}^{(0)}) + f(x_n, y_n)] \quad (20'')$$

3.2 Implicit Integration of SODE

While the Adams-Bashforth-Moulton formula amounts to an initial prediction using forward Euler followed by one step of correction (one iteration) with the trapezoidal method, implicit integration requires the exact solution of the algebraic system obtained after discretization by the trapezoidal method.

In the constant step size implementation of the implicit algorithm, at the beginning of a new time step “1” the values of the independent accelerations are assumed to be identical to the values at the old time step “0”. The trapezoidal integration formula is applied with these accelerations to compute independent velocities and positions at time step “1”. All the required quantities to evaluate Ψ in Eq. 14 are then computed. Based on the value of the residual Ψ , a correction is applied to the independent accelerations. This stage requires the evaluation of the Jacobian in Eq. 19 in a Newton approach. If the Euclidean norm of the correction is smaller than a prescribed value, convergence is accepted. Otherwise, with the new set of corrected accelerations at step “1”, the loop is done once again.

For the numerical experiments presented in this paper, Newton’s method is used

In the variable step size implementation, the step size control is based on a pair of embedded formulas that give an estimate of the local error. This approach follows the idea of Hairer et al. (1993). Whenever a step size h has been chosen, the embedded method provides a second approximations \hat{y}_1 of the solution, and $y_1 - \hat{y}_1$ is taken as an estimate of the error. The goal is that the error satisfy componentwise

$$|y_{1i} - \hat{y}_{1i}| < e_i \quad e_i = \text{Atol}_i + \max(|y_{0i}|, |y_{1i}|) \cdot \text{Rtol}_i \quad (21)$$

where Atol_i and Rtol_i are the desired tolerances for the i^{th} component of the solution. A measure of the

error is provided by $\text{err} = \sqrt{\frac{1}{n} \sum_{i=1}^n \left(\frac{y_{1i} - \hat{y}_{1i}}{e_i} \right)^2}$. Other norms, such as the max norm, may also be used. The error err is compared to 1, in order to find the optimal step size. If p and \hat{p} are the orders of the two methods, and $q = \min(p, \hat{p})$, from the behavior of the error $\text{err} \approx C \cdot h^{q+1}$, and from $1 \approx C \cdot h_{\text{opt}}^{q+1}$, the optimal step size is obtained as

$$h_{\text{opt}} = h \cdot \left(\frac{1}{\text{err}} \right)^{1/(q+1)} \quad (22)$$

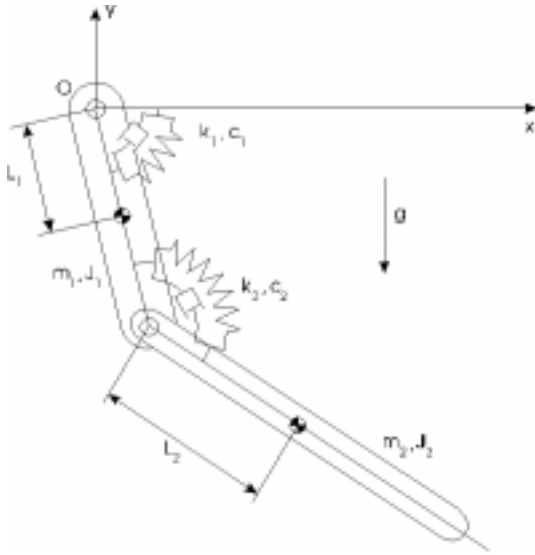
A safety factor fac multiplies the computed h_{opt} , so that the error will be acceptable at the next time step, with high probability. If the computed error err is smaller than one, the step is accepted and a new step is taken with $h_{\text{new}} = \text{fac} \cdot h_{\text{opt}}$. Otherwise, the step is rejected and the computations are repeated with the new step size h_{new} . Details regarding the strategy of choosing a proper value for the safety factor fac are provided by Hairer et al. (1993).

In the present paper, the trapezoidal method is embedded with the backward Euler method, the safety factor is chosen $\text{fac} = 0.85$, and integration is done in local extrapolation mode.

4 Numerical Results

The integrators discussed in the preceding section have been implemented and tested with a set of mechanical system models. Because of space limitations, only one mechanism is presented, namely a stiff double pendulum. Two different sets of initial conditions are considered for the simulation. Results of numerical experiments with more complex mechanical systems that are relevant from an engineering standpoint can be found in the work of Iancu et al. (1997).

The double pendulum shown in Fig. 1 is a two body, two degree of freedom planar mechanical system. The mechanism is modeled using fully Cartesian coordinates; i.e., the x and y coordinates of the centers of the mass and the angles θ that define the orientation of the local centroidal reference frame with respect to the global reference frame. Stiffness is added to the system by means of two rotational spring-damper-actuators (RSDA). The parameters of the problem are provided in Table 1, in SI units



Body	1	2
L	1.0	1.5
m	3.0	0.3
k	400.0	300.0
c	15.0	10,000.0

Table 1. Parameters for Double Pendulum

Figure 1 Double Pendulum Mechanism

The values of the mass and damping coefficient of the second pendulum ensure the stiff character of the problem. The relatively small value of the damping between ground and the first pendulum allows a gross motion of the mechanism that will continuously induce excitation in the motion of the second pendulum. The orientation of the second pendulum will experience very small amplitude, highly damped oscillations about the orientation of the first pendulum. In fact, due to the large amount of damping between the two pendulums, the evolution of the double pendulum will resemble the motion of a single pendulum with a mass of $m_1 + m_2$ and length $2(L_1 + L_2)$.

The initial conditions for the test problem are provided in Table 2. The first row contains position information, and the second contains velocity information. The initial conditions were chosen such that the mechanism does not experience extreme accelerations.

Body 1			Body 2		
x coordinate	y coordinate	θ coordinate	x coordinate	y coordinate	θ coordinate
1.0	0.0	2π	3.44888874	-0.38822858	$23\pi/12$
0.0	0.0	0.0	0.0	0.0	0.0

Table 2. Initial Conditions for the Double Pendulum

A first set of numerical experiments is aimed at demonstrating the validity of the proposed implicit integrator. For this, the constant step size implementations, explicit and implicit, are run for several time steps to determine the dependency of the integration error upon the simulation step size. Results are plotted in Figs. 2 and 3, where EXPL refers to the explicit integrator, and IMPL to the implicit one. The results show the maximum error in absolute value over any time step for the orientation of the second pendulum. In order to compute the error, the solution of the simulation for a certain step size is compared to an “exact” solution obtained with a constant step size fourth order Runge-Kutta method. The step size for the “exact” solution is 10^{-8} . The length of the simulation is 0.1 seconds.

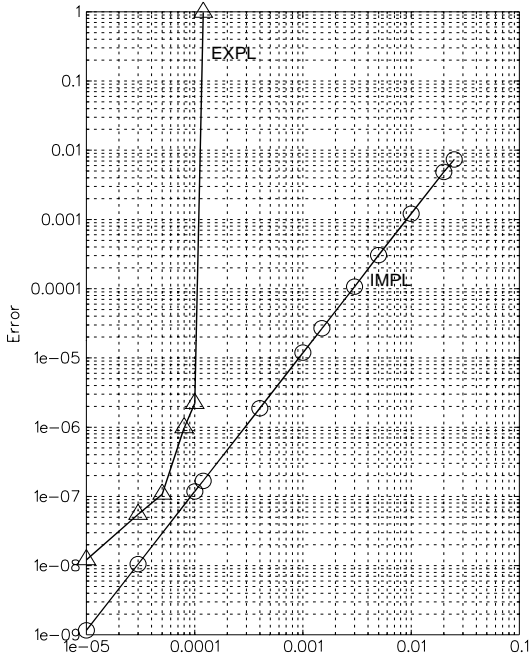


Figure 2 Position Error vs. Step Size

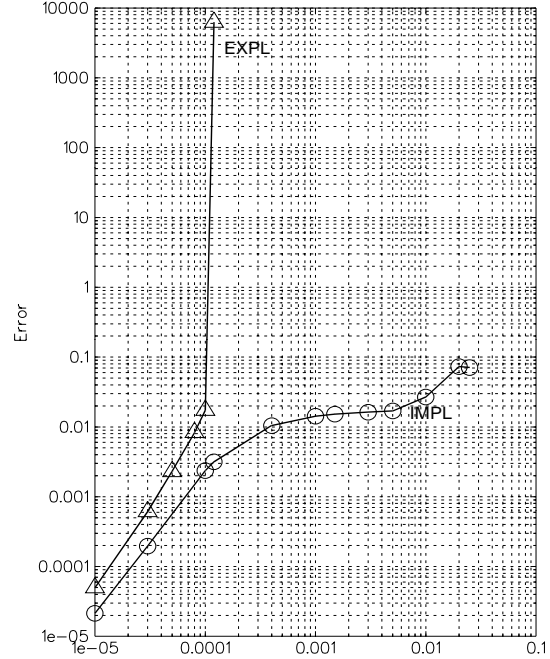


Figure 3 Velocity Error vs. Step Size

The plots show very good agreement with what theory predicts, when working with second order integrators. Since in the state space approach the DAE is reduced to an ODE, and the orientation angle of the second pendulum is one of the independent generalized coordinates being integrated, asymptotically the integration error should vary with the square of the step size. As showed by the plots in Figs. 2 and 3, for small step sizes, this is precisely what happens for both integrators. The straight line variation of error with step size for the implicit integrator very convincingly indicates second order behavior and validates the proposed algorithm.

The largest time step achieved for the explicit integrator is 0.00012. Beyond this value, the explicit integrator becomes unstable. For step sizes larger than 0.025 seconds, the implicit integrator fails when recovering the independent accelerations by exceeding the preimposed number of iterations. The largest step size ratio for the two integrators is 200, in the favor of the implicit integrator. This relatively large value suggests that the problem simulated is stiff.

Figure 4 plots error versus tolerance for the variable step size implicit integrator. The plot reports the maximum error for the orientation angle of the second pendulum over the simulation interval. The length of the simulation is 0.1 seconds, and the solution is compared to the “exact” solution obtained as described before. All simulations were run with $Atol_i=Rtol_i$ in Eq. 21. The plot validates the step size control mechanism proposed in Subsection 3.2. It can be seen that when setting a certain tolerance at the position and velocity level, step sizes chosen by the integrator ensure the desired accuracy.

One interesting conclusion is that while at the position level the accuracy requirements are met over a large range of tolerances, for velocity this ceases to be the case. This suggests that for high accuracy requirements, the low order trapezoidal method used to integrate for the motion of the system should be replaced with a higher order method, embedded with a different formula. However, it can be seen that for the range of accuracy 10^{-5} through 10^{-2} ; i.e., the accuracy requirements of engineering applications, the step size controller behaves extremely well.

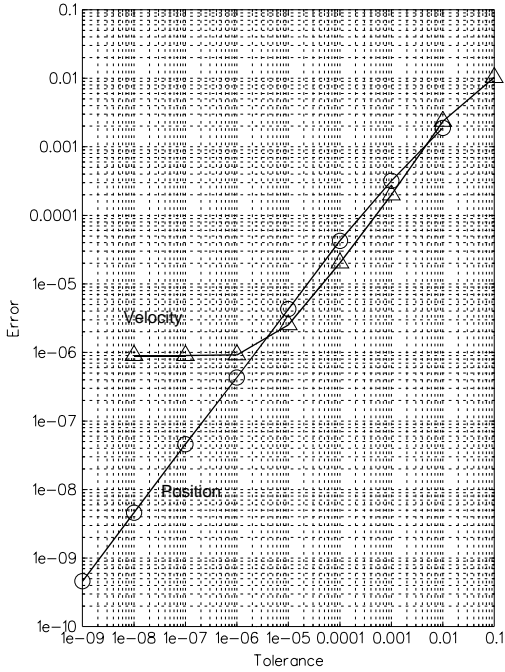


Figure 4 Variable Step Integration

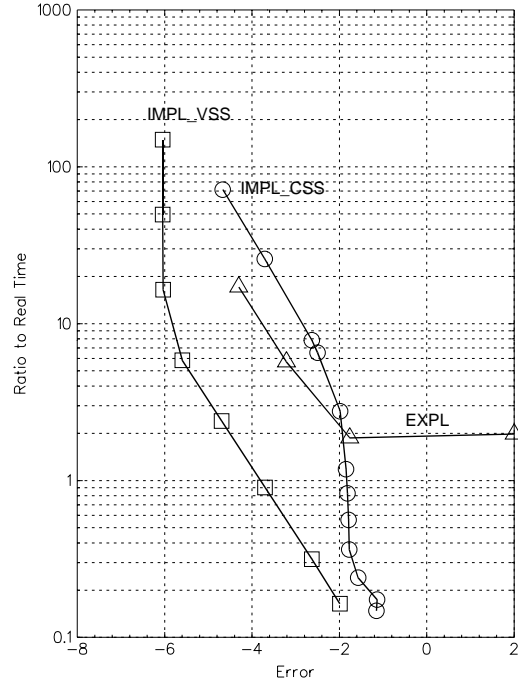


Figure 5 Precision-Work Diagram

Finally, in Fig. 5 a precision-work diagram is provided; IMPL_VSS and IMPL_CSS refer to the variable and constant step implicit integrators respectively, while EXPL represents results obtained with the explicit integrator. On the x-axis is the log of the maximum integration error for the orientation of the second pendulum over the simulation interval. The same “exact” solution is used as for the results in Figs. 2 through 4. On the y-axis is displayed the CPU time required by the simulation. The CPU time is first divided by the length of the simulation, to obtain the so called real time ratio. Thus, everything that is under the horizontal 1 line runs faster than real time. The length of the simulation is 0.1 seconds, the results refer to errors in velocity, and the tolerances for the variable step implicit integrator were set to be $A_{tol}=R_{tol}$. Whenever a simulation is run, the tolerance at the velocity level is precisely set to be one order lower than at the position level. The reason is that if the tolerances at the position and velocity level are set to be the same, step size rejections are almost always due to violation in accuracy at the velocity level. The results in Fig. 5 suggest that at comparable integration errors, the benefit of using the variable step over a constant step implementation of the trapezoidal method is a speed-up of the order of 25. This value holds practically over the whole range of errors desired in engineering applications. The horizontal line for EXPL is a consequence of the loss of stability of the explicit integrator, while the vertical line for IMPL_VSS when tolerances tighter than 10^{-6} are imposed is, as in Fig. 4, a demonstration of the limitation of the low order method. The variable step implicit integrator clearly expands the range of medium errors obtained with relatively small CPU effort. Note that, while for IMPL_CSS, the CPU increases abruptly for errors less than 10^{-2} , in the range 10^{-5} through 10^{-2} , IMPL_VSS displays a linear precision-work diagram with a moderate slope directly related to the order of the method.

A second set of numerical experiments is performed using the same model under a different set of initial conditions. The goal is to initially induce extreme accelerations in the system and monitor the integration errors at the end of a longer simulation. Considering the large value of the damping between the two pendulums, the goal is met by setting a large initial angular velocity on body 2, and taking the simulation interval to be 3.6 seconds long. The new initial conditions are provided in Table 3 in SI units. The first row contains position information, while the second contains velocity information.

Body 1			Body 2		
x coordinate	y coordinate	θ coordinate	x coordinate	y coordinate	θ coordinate
1.0	0.0	2π	3.5	0.0	2π
0.0	0.0	0.0	0.0	90.0	60.0

Table 3. Initial Conditions for the Double Pendulum

The analysis is carried out by comparing the values of the generalized coordinates and velocities at the end of the simulation with a set of exact values. The “exact” values are generated using a fourth order Runge-Kutta method with a step size of 10^{-8} . The variable step size implicit integrator is then run with a set of tolerances to generate the plot in Fig. 6. In Fig. 7 is provided a precision-work diagram for all the integrators.

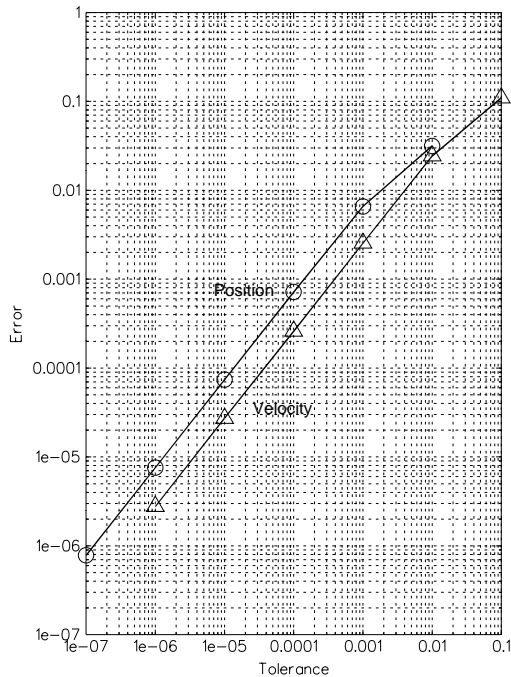


Figure 6 Variable Step Integration

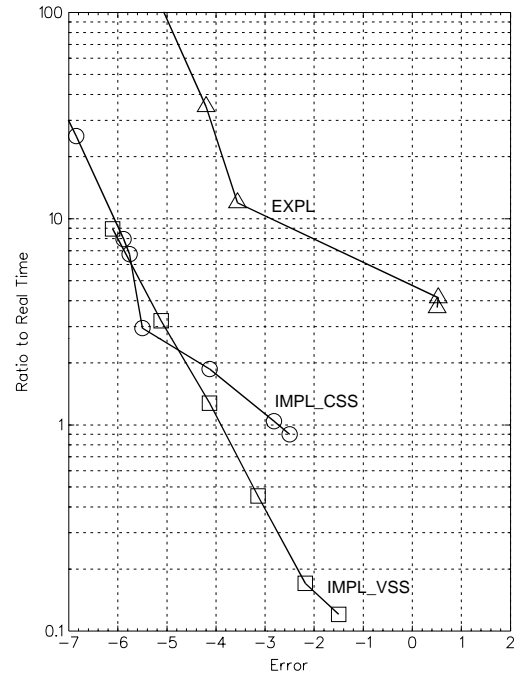


Figure 7 Precision-Work Diagram

The results in Fig. 6 plot the largest error over all generalized coordinates at the position and velocity levels at the end of the simulation interval. The plot shows that the step size control mechanism is sound, in the sense that the accuracy requirements are met.

The largest error over all the generalized positions at the end of the simulation is plotted against the normalized CPU time of the simulation in Fig. 7 in a precision-work diagram. The results indicate that for the new set of initial conditions, the benefit of working with the variable step size implicit implementation is less significant than previously observed. The speed-up of 25 obtained for the first set of initial conditions reduces now to 3 up to 4 for accuracy in the range 10^{-3} through 10^{-2} . Furthermore, for certain values of tolerance in the range 10^{-6} through 10^{-5} , the constant step size outperforms the variable step size implementation. This behavior is due to extreme initial accelerations of the order of 10^6 that fade away in an interval smaller than 10^{-3} seconds. When the constant step size implementation is run with a step size of $3e-4$, at the first time step the values of the accelerations are already of the order of 10^3 , while after one more step they are of the order of 10^2 . On the other hand, when the tolerance is set to 10^{-6} , the variable step size implementation takes 484 steps to arrive at the simulation time that the constant step size implementation reaches after one step. The constant step size implementation displays large errors at the beginning of the simulation that for this highly damped model are not going to have impact on the results in

Fig. 7. In this plot, errors at the end of a relatively long (when compared to the duration of the transients) simulation interval are reported. For this overdamped model, had the maximum error over the simulation interval been reported, the results would have looked like the ones in Fig. 5, when the variable step implementation significantly outperforms the constant step size implementation. Finally, the largest step size attained by the constant step implicit integrator is 0.0012 seconds. The results in Fig. 7 suggest that the explicit integrator compares poorly with the implicit integrators; for this integrator loss of stability is observed for step sizes larger than 0.00003 seconds.

5 Conclusions

The variable step integrator based on the implicit trapezoidal method is a reliable and efficient algorithm for mathematically speaking low accuracy precision. These precisions are in most of the cases at the level of accuracy desired in engineering applications. Confirming the theoretical results and practical wisdom, the step size controller built around a pair of low order embedded formulas is not satisfactory when medium or high accuracy is desired. The performance of the implicit integrators (variable and constant size implementations) is expected to improve by replacing the current Newton-Raphson method with a quasi-Newton method to recover independent accelerations. The work on this issue is in progress. Finally, the numerical results obtained validate the proposed implicit integrators, and recommend them for the dynamic analysis of stiff mechanical systems. The results also reveal the limitations of explicit integration in dealing with this class of problems.

Acknowledgement

This research was funded by the US Army Tank Automotive Command Center (TACOM) through the Automotive Research Center (Department of Defense contract number DAAE07-94-C-R094).

6 References

- Atkinson, K., E., 1989, *An Introduction to Numerical Analysis*, Second edition, Wiley, New York
- Corwin, L., J., Szczarba, R., H., 1982, *Multivariable Calculus*, Marcel Dekker, New York
- Hairer, E., Norsett, S., P., Wanner, G., 1993, *Solving Ordinary Differential Equations I: Nonstiff Problems*, Second Edition, Springer-Verlag, Berlin
- Hairer, E., Wanner, G., 1996, *Solving Ordinary Differential Equations II: Stiff and Differential Algebraic Problems*, Second Edition, Springer-Verlag, Berlin
- Haug, E., J., 1989, *Computer Aided Kinematics and Dynamics of Mechanical Systems Volume I: Basic Methods*, Allyn-Bacon, Boston
- Haug, E., J., Negrut, D., Engstler, C., 1997, "Runge-Kutta Implicit Methods for the Integration of DAE of Multibody Dynamics", work in progress
- Haug, E., J., Negrut, D., Iancu, M., 1997, "A State-Space Based Implicit Integration Algorithm for Differential-Algebraic Equations of Multibody Dynamics", to appear, *Mechanics of Structures and Machines*
- Haug, E., J., Negrut, D., Iancu, M., Implicit Integration of the Equations of Multibody Dynamics: State Space Form, NATO ASI on Computational Methods in Mechanisms, Bulgaria, 1997
- Iancu, M., Haug, E., J., Negrut, D., 1997, Implicit Numerical Integration of the Equations of Stiff Multibody Dynamics: Descriptor Form, NATO ASI on Computational Methods in Mechanisms, Bulgaria, 1997
- Negrut, D., Serban, R., Potra, F., A., 1996, "A Topology Based Approach for Exploiting Sparsity in Multibody Dynamics : Joint Formulation", to appear, *Mechanics of Structures and Machines*
- Potra, F., A., 1994, "Numerical Methods for Differential-Algebraic Equations with Application to Real-Time Simulation of Mechanical Systems", *ZAMM*, 74(94), pp. 177-187
- Serban, R., Haug, E., J., 1997, "Kinematic and Kinetic Derivatives in Multibody System Analysis", NATO ASI on Computational Methods in Mechanisms, Bulgaria, 1997
- Serban, R., Negrut, D., Potra, F., A., Haug, E., J., 1997, "A Topology Based Approach for Exploiting Sparsity in Multibody Dynamics in Cartesian Formulation", to appear, *Mechanics of Structures and Machines*