A Rosenbrock–Nystrom state space implicit approach for the dynamic analysis of mechanical systems: I—theoretical formulation

A Sandu[1], **D Negruţ**[2]*, **E J Haug**[3], **F A Potra**[4] and **C Sandu**[5]
[1]Department of Computer Science, Michigan Technological University, Houghton, MI, USA
[2]MSCsoftware, Ann Arbor, MI, USA
[3]Department of Mechanical Engineering, The University of Iowa, Iowa City, IA, USA
[4]Department of Mathematics and Statistics, University of Maryland, Baltimore County, Baltimore, MD, USA
[5]Department of Mechanical Engineering—Engineering Mechanics, Michigan Technological University, Houghton, MI, USA

Abstract: When performing dynamic analysis of a constrained mechanical system, a set of index three differential-algebraic equations (DAE) describes the time evolution of the system. The paper presents a state space based method for the numerical solution of the resulting DAE. A subset of so-called independent generalized coordinates, equal in number to the number of degrees of freedom of the mechanical system, is used to express the time evolution of the mechanical system. The second-order state space ordinary differential equations (SSODE) that describe the time variation of independent coordinates are numerically integrated using a Rosenbrock-type formula specialized to second-order systems of differential equations. Rosenbrock methods are known to be efficient for medium accuracy integration of stiff systems; they do not require the solution of non-linear systems for the stage values, and possess optimal linear stability properties for stiff integration. The computation of exact Jacobians needed by Rosenbrock formulas is discussed in the context of multibody systems. The companion paper [**19**] discusses a choice of method coefficients based on a four-stage L-stable Rosenbrock formula and presents numerical results.

Keywords: multibody dynamics, differential-algebraic equations, state space form, Rosenbrock methods

NOTATION

$(a)_{ij}$, $(c)_{ij}$, $(b)_i$, $(m)_i$	Rosenbrock–Nystrom method coefficients
$\mathbf{e}$	vector of Euler parameters
err	local truncation error estimator
g	explicit coordinate dependence function
J_1	Jacobian w.r.t. y
J_2	Jacobian w.r.t. y'
k_i	stage vector of Rosenbrock method
ℓ_i	stage vector of Rosenbrock–Nystrom method
m	number of constraints
$\mathbf{M}$	system mass matrix
$\hat{\mathbf{M}}$	reduced mass matrix
nb	number of bodies
$ndof$	number of degrees of freedom
$\mathbf{p}$	vector of positions
$\mathbf{q}$	vector of generalized coordinates
$\mathbf{Q}^A$	generalized external forces
$\hat{\mathbf{Q}}$	reduced external forces
$\mathbf{u}$	vector of dependent coordinates
$\mathbf{v}$	vector of independent coordinates
y_n	solution of Rosenbrock method at t_n
$(\alpha)_{ij}$, $(\gamma)_{ij}$, $(\delta)_{ij}$, $(\theta)_{ij}$	Rosenbrock–Nystrom method coefficients
Φ	kinematic constraints
λ	vector of Lagrange multipliers
μ	dependent coordinates mapping
ν	independent coordinates mapping
τ	acceleration constraint right-hand function

The MS was received on 17 January 2003 and was accepted after revision for publication on 17 July 2003.

**Corresponding author: MSCsoftware, 2300 Traverwood Drive, Ann Arbor, MI 48105, USA.*

1 INTRODUCTION

In this paper, the state of a multibody system at the position level is represented by an array $\mathbf{q} = [q_1, \ldots, q_n]^T$ of generalized coordinates. The velocity of the system is described

by the array of generalized velocities $\dot{\mathbf{q}} = [\dot{q}_1, \ldots, \dot{q}_n]^T$. There is a multitude of ways in which the set of generalized coordinates and velocities can be selected [**1–3**]. The generalized coordinates used in this paper are Cartesian coordinates for position and Euler parameters for orientation of body centroidal reference frames. Thus, for each body i the position of the body is described by the vector $\mathbf{p}_i = [x_i, y_i, z_i]^T$, while the orientation is given by the array of Euler parameters [**2**], $\mathbf{e}_i = [e_{i0}, e_{i1}, e_{i2}, e_{i3}]^T$. Consequently, for a mechanical system containing nb bodies,

$$\mathbf{q} = [\, \mathbf{p}_1^T \quad \mathbf{e}_1^T \quad \ldots \quad \mathbf{p}_{nb}^T \quad \mathbf{e}_{nb}^T \,]^T \in \mathbf{R}^{7nb} \tag{1}$$

When compared with the alternative of using a set of relative generalized coordinates, the coordinates considered are convenient because of the rather complex formalism employed to obtain the Jacobian information required for implicit integration.

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

$$\Phi(\mathbf{q}) = [\, \Phi_1(\mathbf{q}) \quad \ldots \quad \Phi_m(\mathbf{q}) \,]^{\mathbf{T}} = \mathbf{0} \tag{2a}$$

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. The number of degrees of freedom $ndof$ is thus the difference between the number of generalized coordinates and the number of constraints $ndof = n - m$.

Differentiating equation (2a) with respect to time leads to the velocity kinematic constraint equation

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

where the over dot denotes differentiation with respect to time and the subscript denotes partial differentiation, $\Phi_{\mathbf{q}} = \partial(\Phi_1 \cdots \Phi_m)/\partial(q_1 \cdots q_n)$. The acceleration kinematic constraint equations are obtained by differentiating equation (2b) with respect to time

$$\Phi_{\mathbf{q}}(\mathbf{q})\ddot{\mathbf{q}} = -[\Phi_{\mathbf{q}}(\mathbf{q})\dot{\mathbf{q}}]_{\mathbf{q}}\dot{\mathbf{q}} \equiv \tau(\mathbf{q}, \dot{\mathbf{q}}) \tag{2c}$$

Equations (2a)–(2c) 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 [**2**]

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \Phi_{\mathbf{q}}^T(\mathbf{q})\lambda = \mathbf{Q}^A(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{t}) \tag{2d}$$

where $\mathbf{M}(\mathbf{q}) \in \mathbf{R}^{n \times n}$ is the symmetric system mass matrix, $\lambda \in \mathbf{R}^m$ is the array of Lagrange multipliers that account for workless constraint forces, and $\mathbf{Q}^A(\mathbf{q}, \dot{\mathbf{q}}, t) \in \mathbf{R}^n$ represents the generalized applied force that may depend on the generalized coordinates, their first time derivatives, and time.

Equations (2a)–(2d) comprise a system of differential-algebraic equations (DAE). It is known that differential-algebraic equations are not ordinary differential equations [**4**]. Analytical solutions of equations (2a) and (2d) automatically satisfy equations (2b) and (2c), but this is no longer true for numerical solutions. In general, the task of obtaining a numerical solution of the DAE of equations (2a)–(2d) is substantially more difficult and prone to intense numerical computation than that of solving ordinary differential equations (ODE). In general, the numerical solution method employed to find the time evolution of a mechanical system falls in one of the following categories: (1) stabilization methods, (2) projection methods, and (3) state space methods. For a review of the literature on numerical integration methods for the solution of the DAE of multibody dynamics the reader is referred to [**5–9**]. While the stabilization and projection methods are typically more expedient, the theoretical foundation for these methods is either very complex, or is not completely understood yet. The state space methods are more rigorous since the DAE problem is eventually reduced to an ODE problem for which the theory is well understood and the numerical software is readily available. Likewise, there is a series of applications like active control of mechanical systems, vibration analysis, and so on, where a state space representation of the mechanical system is necessary. In this context, note that until recently the SSODE solution of the multibody dynamics DAE problem was formalized exclusively in the context of explicit integration. Based on the approach proposed in reference [**10**], this paper introduces for the first time the idea of using an efficient implicit Rosenbrock method for the solution of the SSODE problem.

Expressing equations (2c)–(2d) in matrix form as

$$\begin{bmatrix} \mathbf{M}(\mathbf{q}) & \Phi_{\mathbf{q}}(\mathbf{q})^T \\ \Phi_{\mathbf{q}}(\mathbf{q}) & \mathbf{0} \end{bmatrix} \begin{bmatrix} \ddot{\mathbf{q}} \\ \lambda \end{bmatrix} = \begin{bmatrix} \mathbf{Q}^A(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{t}) \\ \tau(\mathbf{q}, \dot{\mathbf{q}}) \end{bmatrix} \tag{3}$$

it results that equations (2a), (2b), and (3) must be satisfied by the numerical solution to be constructed. This is a set of index three DAEs. The differential index three means that the constraints must be differentiated three times to transform the DAE into a system of ODEs. A comprehensive discussion of DAEs and the index concept is provided in reference [**11**].

The index three DAE system (2a), (2b), and (3) is first reduced to an SSODE, following an approach proposed in reference [**12**]. The starting point for such a method is a partitioning of $\mathbf{q}$ in equation (1) in independent and dependent coordinates $\mathbf{v} \in \mathbf{R}^{ndof}$, and $\mathbf{u} \in \mathbf{R}^m$, $ndof = n - m$. The partitioning is based on two mappings v: $\{1, 2, \ldots, ndof\} \rightarrow S_{\text{indep}}$ and μ: $\{1, 2, \ldots, m\} \rightarrow S_{\text{dep}}$, with $S_{\text{indep}} \cup S_{\text{dep}} = \{1, 2, \ldots, n\}$ and $S_{\text{indep}} \cap S_{\text{dep}} = \emptyset$.

$$\mathbf{v}[i] = \mathbf{q}[v(i)],\ 1 \leqslant i \leqslant ndof, \text{ and } \mathbf{u}[j] = \mathbf{q}[\mu(j)],\ 1 \leqslant j \leqslant m \tag{4}$$

The partitioning is such that the sub-Jacobian of the constraints with respect to $\mathbf{u}$ is nonsingular

$$\det(\Phi_{\mathbf{u}}(\mathbf{q})) \neq 0 \tag{5}$$

Such a partition can be found starting with a set of consistent generalized coordinates $\mathbf{q}_0$; that is, which satisfy equation (2a). The constraint Jacobian matrix $\Phi_{\mathbf{q}}$ is evaluated and numerically factored, using the Gauss–Jordan algorithm with full pivoting [**13**]

$$\Phi_{\mathbf{q}}(\mathbf{q}_0) \mapsto (\text{Gauss–Jordan}) \mapsto [\Phi_{\mathbf{u}}(\mathbf{q}_0)|\Phi_{\mathbf{v}}(\mathbf{q}_0)] \tag{6}$$

Based on this partitioning, equations (2a)–(2c) can be rewritten in the associated partitioned form [**2**]

$$\mathbf{M}^{\mathbf{vv}}(\mathbf{u}, \mathbf{v})\dot{\mathbf{v}} + \mathbf{M}^{\mathbf{vu}}(\mathbf{u}, \mathbf{v})\ddot{\mathbf{u}} + \Phi_{\mathbf{v}}^{\mathbf{T}}(\mathbf{u}, \mathbf{v})\lambda = \mathbf{Q}^{\mathbf{v}}(\mathbf{u}, \mathbf{v}, \dot{\mathbf{u}}, \dot{\mathbf{v}}) \tag{7a}$$

$$\mathbf{M}^{\mathbf{uv}}(\mathbf{u}, \mathbf{v})\dot{\mathbf{v}} + \mathbf{M}^{\mathbf{uu}}(\mathbf{u}, \mathbf{v})\ddot{\mathbf{u}} + \Phi_{\mathbf{u}}^{\mathbf{T}}(\mathbf{u}, \mathbf{v})\lambda = \mathbf{Q}^{\mathbf{u}}(\mathbf{u}, \mathbf{v}, \dot{\mathbf{u}}, \dot{\mathbf{v}}) \tag{7b}$$

$$\Phi(\mathbf{u}, \mathbf{v}) = 0 \tag{7c}$$

$$\Phi_{\mathbf{u}}(\mathbf{u}, \mathbf{v})\dot{\mathbf{u}} + \Phi_{\mathbf{v}}(\mathbf{u}, \mathbf{v})\dot{\mathbf{v}} = 0 \tag{7d}$$

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

The partitioning of equations (7a)–(7e) is induced by the partitioning of the generalized coordinates in equation (4). For example, $\mathbf{M}^{\mathbf{vu}}[i, j] = \mathbf{M}[v(i), \mu(j)]$, for $1 \leqslant i \leqslant ndof$, $1 \leqslant j \leqslant n - ndof$, while $\mathbf{Q}^{\mathbf{u}}[j] = \mathbf{Q}[\mu(j)]$, for $1 \leqslant j \leqslant n - ndof$. Likewise, a partial with respect to the dependent set of coordinates $\mathbf{u}$ is obtained by gathering the columns $\mu(1)$ through $\mu(m)$ of the derivative with respect to the generalized coordinates $\mathbf{q}$. The remaining *ndof* columns provide the partial with respect to the independent coordinates $\mathbf{v}$.

The condition of equation (5) and the implicit function theorem [**14**] guarantee that equation (7c) can be solved for $\mathbf{u}$ as a function of $\mathbf{v}$

$$\mathbf{u} = \mathbf{g}(\mathbf{v}) \tag{8}$$

where the function $\mathbf{g}(\mathbf{v})$ has as many continuous derivatives as does the constraint function $\Phi(\mathbf{q})$. For all but the most simple mechanical systems, an analytical expression for the function $\mathbf{g}(\mathbf{v})$ cannot be determined. However, for any consistent configuration $\mathbf{q}_0 = (\mathbf{u}_0, \mathbf{v}_0)$, a value $\bar{\mathbf{u}}$ can be found for each $\bar{\mathbf{v}}$ in a small enough neighborhood of $\mathbf{v}_0$. The value $\bar{\mathbf{u}}$ is computed by keeping $\mathbf{v} = \bar{\mathbf{v}}$ in equation (7c) constant and solving for $\mathbf{u} = \bar{\mathbf{u}}$. Because of the form the joint constraint equations assume when used in conjunction with a Cartesian representation, $\bar{\mathbf{u}}$ is typically the solution of a system of nonlinear equations.

The system of DAE in equations (7a), (7b), and (7e) is reduced to an SSODE, through a sequence of steps that use information provided by equations (7c) and (7d). First, since the coefficient matrix of $\dot{\mathbf{u}}$ in equation (7d) is nonsingular, $\dot{\mathbf{u}}$ can be determined as a function of $\dot{\mathbf{v}}$ and $\mathbf{v}$, where equation (8) is used to eliminate explicit dependence on $\mathbf{u}$. Next, equation (7e) uniquely determines $\ddot{\mathbf{u}}$ as a function of $\mathbf{v}$, $\dot{\mathbf{v}}$, and $\ddot{\mathbf{v}}$, where results from equations (7d) and (8) are substituted. Since the coefficient matrix of λ in equation (7b) is nonsingular, λ 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 is substituted into equation (7a) to obtain the SSODE in the independent generalized coordinates $\mathbf{v}$ [**12**]

$$\hat{\mathbf{M}}(\mathbf{v})\ddot{\mathbf{v}} = \hat{\mathbf{Q}}(\mathbf{t}, \mathbf{v}, \dot{\mathbf{v}}) \tag{9}$$

where

$$\hat{\mathbf{M}} = \mathbf{M}^{\mathbf{vv}} - \mathbf{M}^{\mathbf{vu}}\Phi_{\mathbf{u}}^{-1}\Phi_{\mathbf{v}} - \Phi_{\mathbf{v}}^{\mathbf{T}}\Phi_{\mathbf{u}}^{-\mathbf{T}}[\mathbf{M}^{\mathbf{uv}} - \mathbf{M}^{\mathbf{uu}}\Phi_{\mathbf{u}}^{-1}\Phi_{\mathbf{v}}] \tag{10a}$$

$$\hat{\mathbf{Q}} = \mathbf{Q}^{\mathbf{v}} - \mathbf{M}^{\mathbf{vu}}\Phi_{\mathbf{u}}^{-1}\tau - \Phi_{\mathbf{v}}^{\mathbf{T}}\Phi_{\mathbf{u}}^{-\mathbf{T}}[\mathbf{Q}^{\mathbf{u}} - \mathbf{M}^{\mathbf{uu}}\Phi_{\mathbf{u}}^{-1}\tau] \tag{10b}$$

The SSODE (9) is well defined, due to the following property [**2**].

Lemma 1 *For any* $\mathbf{v} \in \mathbf{R}^{ndof}$, *the matrix* $\hat{\mathbf{M}}(\mathbf{v})$ *of equation* (*10a*) *is positive definite.*

2 ROSENBROCK INTEGRATION FORMULAS FOR SECOND-ORDER SYSTEMS

For the initial value problem (IVP), $y' = f(t, y)$, $y(t_0) = y_0$, an s-stage Rosenbrock method is defined as [**11**]

$$y_{n+1} = y_n + \sum_{i=1}^{s} b_i k_i \tag{11a}$$

$$k_i = hf(t_n + \alpha_{ih}, y_n + \sum_{j=1}^{i-1} \alpha_{ij} k_j) + \gamma_i h^2 \frac{\partial f}{\partial t}(t_n, y_n) + hJ \sum_{j=1}^{i} \gamma_{ij} k_j \tag{11b}$$

where the number of stages s and the coefficients b_i, α_{ij}, and γ_{ij} are chosen to obtain a desired order of consistency and stability, $J = f_y(t_n, y_n)$, $\alpha_i = \sum_{j=1}^{i-1} \alpha_{ij}$, and $\gamma_i = \sum_{j=1}^{i} \gamma_{ij}$. For reasons of computational efficiency the coefficients γ_{ii} are identical for all stages; that is, $\gamma_{ii} = \gamma$ for all $i = 1, \ldots, s$. Note that formally $\alpha_{ii} = 0$, $1 \leqslant i \leqslant s$.

For the purpose of error control in the generic Rosenbrock method a second approximation of the solution at the current time step is used to produce an estimate of the local error. This second approximation $\hat{y}_{n+1}$ is usually of lower order and it uses the same stage values k_i with a different set of coefficients $\hat{b}_i$

$$\hat{y}_{n+1} = y_n + \sum_{i=1}^{s} \hat{b}_i k_i \tag{12}$$

To apply a generic Rosenbrock formula for the solution of the SSODE of multibody dynamics of equation (9), by

multiplying from the left with $\hat{\mathbf{M}}^{-1}$ the second-order SSODE of equation (9) can be locally reduced to the form

$$y'' = f(t, y, y') \tag{13a}$$

Note that for the purpose of introducing the Rosenbrock–Nystrom approach and without any loss of generality in the formula above the vector $\mathbf{v}$ was replaced with a scalar quantity y. As the interest is only in determining the coefficients of the Rosenbrock–Nystrom method, the fact that this formula is used to find a scalar or vector numerical solution of an ODE problem is irrelevant.

The previous second-order system of ODE is transformed into a standard first-order ODE problem

$$\begin{bmatrix} y \\ y' \end{bmatrix}' = \begin{bmatrix} y' \\ f(t, y, y') \end{bmatrix} \tag{13b}$$

Applying the generic method of equations (11a) and (11b) to this first-order ODE system yields

$$\begin{bmatrix} y_{n+1} \\ y'_{n+1} \end{bmatrix} = \begin{bmatrix} y_n \\ y'_n \end{bmatrix} + \sum_{i=1}^{s} b_i \begin{bmatrix} k_i \\ \ell_i \end{bmatrix} \tag{14a}$$

$$\begin{aligned} \begin{bmatrix} k_i \\ \ell_i \end{bmatrix} &= h \begin{bmatrix} y'_n + \sum_{j=1}^{i-1} \alpha_{ij} \ell_j \\ f(t_n + \alpha_i h, y_n + \sum_{j=1}^{i-1} \alpha_{ij} k_j, y'_n + \sum_{j=1}^{i-1} \alpha_{ij} \ell_j) \end{bmatrix} \\ &\quad + \gamma_i h^2 \begin{bmatrix} 0 \\ \dfrac{\partial f}{\partial t}(t_n, y_n, y'_n) \end{bmatrix} \\ &\quad + h \begin{bmatrix} 0 & I \\ J_1 & J_2 \end{bmatrix} \sum_{j=1}^{i} \gamma_{ij} \begin{bmatrix} k_j \\ \ell_j, \end{bmatrix} \end{aligned} \tag{14b}$$

where

$$J_1 = \frac{\partial f}{\partial y}(t_n, y_n, y'_n) \qquad \text{and} \qquad J_2 = \frac{\partial f}{\partial y'}(t_n, y_n, y'_n) \tag{15}$$

In order to obtain a numerical method to directly integrate equation (13a), the 'y-stages' k_i are eliminated to express the formula only in terms of the 'y'-stages' ℓ_i. Defining $\beta_{ij} = \alpha_{ij} + \gamma_{ij}$, the first row of equation (14b) is

$$k_i = h y'_n + h \sum_{j=1}^{i} \beta_{ij} \ell_j, \qquad i = 1, \ldots, s \tag{16}$$

In the second row of equation (14b), the sum $\sum_{j=1}^{i-1} \alpha_{ij} k_j$ comes in as the second argument of $f(\cdot, \cdot, \cdot)$. Defining $\delta_{ij} = \sum_{m=j}^{i-1} \alpha_{im} \beta_{mj}$ and using equation (16) and the summation interchange procedure, this sum is expressed in terms of ℓ_j as

$$\begin{aligned} \sum_{j=1}^{i-1} \alpha_{ij} k_j &= \sum_{j=1}^{i-1} \alpha_{ij} \left(h y'_n + h \sum_{m=1}^{j} \beta_{jm} \ell_m \right) \\ &= h \alpha_i y'_n + h \sum_{j=1}^{i-1} \delta_{ij} \ell_j \end{aligned}$$

With the above substitution, the second row of equation (14b) becomes

$$\begin{aligned} \ell_i &= hf \left(t_n + \alpha_i h, y_n + h \alpha_i y'_n + h \sum_{j=1}^{i-1} \delta_{ij} \ell_j, y'_n + \sum_{j=1}^{i-1} \alpha_{ij} \ell_j \right) \\ &\quad + \gamma_i h^2 \frac{\partial f}{\partial t}(t_n, y_n, y'_n) + h J_1 \sum_{j=1}^{i} \gamma_{ij} k_j + h J_2 \sum_{j=1}^{i} \gamma_{ij} \ell_j \end{aligned} \tag{17}$$

Defining $\theta_{ij} = \sum_{m=j}^{i} \gamma_{im} \beta_{mj}$ and substituting the whole sum in ks by a sum in ℓs

$$\sum_{j=1}^{i} \gamma_{ij} k_j = h \gamma_i y'_n + h \sum_{j=1}^{i} \theta_{ij} \ell_j$$

equation (17) becomes the stage relation

$$\begin{aligned} \ell_i &= hf \left(t_n + \alpha_i h, y_n + h \alpha_i y'_n + h \sum_{j=1}^{i-1} \delta_{ij} \ell_j, y'_n + \sum_{j=1}^{i-1} \alpha_{ij} \ell_j \right) \\ &\quad + \gamma_i h^2 \left[\frac{\partial f}{\partial t}(t_n, y_n, y'_n) + J_1 y'_n \right] \\ &\quad + h^2 J_1 \sum_{j=1}^{i} \theta_{ij} \ell_j + h J_2 \sum_{j=1}^{i} \gamma_{ij} \ell_j \end{aligned} \tag{18}$$

From a computational point of view equation (18) has the following interpretation: at stage i ($1 \leqslant i \leqslant s$) the quantity ℓ_i is to be found as the solution of a linear system

$$S_i \ell_i = RHS$$

where $S_i = I - h \gamma_{ii} J_2 - h^2 \theta_{ii} J_1$. Since $\alpha_{ii} = 0$, $1 \leqslant i \leqslant s$, it follows that $\theta_{ii} = \gamma_{ii} \beta_{ii} = \gamma_{ii}(\alpha_{ii} + \gamma_{ii}) = \gamma^2$. Thus, the linear systems to be solved at each stage have the same matrix

$$S_i = S = I - h \gamma J_2 - h^2 \gamma^2 J_1$$

Substituting ks by ℓs in equation (14a) and denoting $\mu_i = \sum_{j=i}^{s} b_j \beta_{ji}$ leads to

$$y_{n+1} = y_n + h y'_n + h \sum_{i=1}^{s} \mu_i \ell_i \tag{19a}$$

$$y'_{n+1} = y'_n + \sum_{i=1}^{s} b_i \ell_i \tag{19b}$$

Using matrix notation for the coefficients [e.g. (α_{ij}) is the matrix whose entries are the α-coefficients of the method] δ, θ, and μ are expressed as

$$(\delta_{ij}) = (\alpha_{ij}) \cdot (\beta_{ij}), \qquad (\theta_{ij}) = (\gamma_{ij}) \cdot (\beta_{ij}),$$
$$(\mu_i) = (b_i) \cdot (\beta_{ij})$$

To summarize, the following linearly implicit method for the second-order system (13a) is defined as

$$y_{n+1} = y_n + hy'_n + h\sum_{i=1}^{s} \mu_i \ell_i \tag{20a}$$

$$y'_{n+1} = y'_n + \sum_{i=1}^{s} b_i \ell_i \tag{20b}$$

$$Y_i = y_n + h\alpha_i y'_n + h\sum_{j=1}^{i-1} \delta_{ij} \ell_j \tag{20c}$$

$$Y'_i = y'_n + \sum_{j=1}^{i-1} \alpha_{ij} \ell_j \tag{20d}$$

$$\begin{aligned}\ell_i = {} & hf(t_n + \alpha_i h, Y_i, Y'_i) \\ & + \gamma_i h^2 \left[\frac{\partial f}{\partial t}(t_n, y_n, y'_n) + J_1 y'_n\right] \\ & + h^2 J_1 \sum_{j=1}^{i} \theta_{ij} \ell_j + hJ_2 \sum_{j=1}^{i} \gamma_{ij} \ell_j\end{aligned} \tag{20e}$$

It can be seen in equation (18) that matrix-vector multiplications are needed. Because of the presence of both J_1 and J_2, the classical transformation removes only the multiplications with one of the Js. Substituting $z_i = \sum_{j=1}^{i} \gamma_{ij} \ell_j$ into the method, equations (20a–20e) leads to the following.

Theorem 1 *Let (α_{ij}), (γ_{ij}), (b_i), and $(\hat{b}_i)$ be the coefficients of an s-stage embedded Rosenbrock method given by equations (11a) and (11b). The associated Rosenbrock–Nystrom method is defined as*

$$\begin{aligned} y_{n+1} &= y_n + hy'_n + h\sum_{i=1}^{s} \mu_i z_i, \\ \hat{y}_{n+1} &= y_n + hy'_n + h\sum_{i=1}^{s} \hat{\mu}_i z_i \end{aligned} \tag{21a}$$

$$y'_{n+1} = y'_n + \sum_{i=1}^{s} m_i z_i, \qquad \hat{y}'_{n+1} = y'_n + \sum_{i=1}^{s} \hat{m}_i z_i \tag{21b}$$

$$Y_i = y_n + h\alpha_i y'_n + h\sum_{j=1}^{i-1} \theta_{ij} z_j \tag{21c}$$

$$Y'_i = y'_n + \sum_{j=1}^{i-1} a_{ij} z_j \tag{21d}$$

$$S = I - h\gamma J_2 - h^2\gamma^2 J_1 \tag{21e}$$

$$\begin{aligned} S \cdot z_i = {} & h\gamma f(t_n + \alpha_i h, Y_i, Y'_i) \\ & + h^2\gamma\gamma_i \left[\frac{\partial f}{\partial t}(t_n, y_n, y'_n) + J_1 y'_n\right] \\ & + \gamma \sum_{j=1}^{i-1} c_{ij} z_j + h^2 \gamma J_1 \sum_{j=1}^{i-1} \delta_{ij} z_j \end{aligned} \tag{21f}$$

where the new coefficients are

$$\begin{aligned} (a_{ij}) &= (\alpha_{ij}) \cdot (\gamma_{ij})^{-1} \\ (c_{ij}) &= \gamma^{-1} I - (\gamma_{ij})^{-1} \\ (\delta_{ij}) &= (\gamma_{ij}) + (\gamma_{ij}) \cdot (\alpha_{ij}) \cdot (\gamma_{ij})^{-1} \\ (\theta_{ij}) &= (\alpha_{ij}) + (\alpha_{ij})^2 \cdot (\gamma_{ij})^{-1} \\ (m_i) &= (b_i) \cdot (\gamma_{ij})^{-1} \\ (\hat{m}_i) &= (\hat{b}_i) \cdot (\gamma_{ij})^{-1} \\ (\mu_i) &= (b_i) + (b_i) \cdot (\alpha_{ij}) \cdot (\gamma_{ij})^{-1} \\ (\hat{\mu}_i) &= (\hat{b}_i) + (\hat{b}_i) \cdot (\alpha_{ij}) \cdot (\gamma_{ij})^{-1} \\ (\gamma_i) &= (1, \ldots, 1) \cdot (\gamma_{ij})^T \\ (\alpha_i) &= (1, \ldots, 1) \cdot (\alpha_{ij})^T \end{aligned}$$

Proof. By direct substitution. □

Note that in equations (21a) and (21b), values for a second, and typically lower-order, approximation is provided at time t_{n+1} for both the solution $\hat{y}_{n+1}$ and its first derivative $\hat{y}'_{n+1}$. These values are used for step-size control, as indicated in equation (12).

3 PROVIDING ANALYTICAL INTEGRATION JACOBIANS

A successful implementation of the Rosenbrock family of formulas introduced above depends upon the ability to provide exact derivative information. Equation (15) indicates the derivatives that must be computed. The numerical solution of a dynamic analysis problem carried out in the proposed framework of a Rosenbrock implicit integrator requires exact computation of the derivatives of independent acceleration with respect to independent positions and velocities

$$\mathbf{J}_1 = \ddot{\mathbf{v}}_{\mathbf{v}}, \qquad \mathbf{J}_2 = \ddot{\mathbf{v}}_{\dot{\mathbf{v}}} \tag{22}$$

To obtain these derivatives, first differentiating equation (7a) with respect to $\mathbf{v}$ yields

$$\begin{aligned} & \mathbf{M}^{\mathbf{vv}}\mathbf{J}_1 + (\mathbf{M}^{\mathbf{vv}}\ddot{\mathbf{v}})_{\mathbf{v}} + (\mathbf{M}^{\mathbf{vv}}\ddot{\mathbf{v}})_{\mathbf{u}}\mathbf{u}_{\mathbf{v}} + \mathbf{M}^{\mathbf{vu}}\ddot{\mathbf{u}}_{\mathbf{v}} \\ & \quad + (\mathbf{M}^{\mathbf{vu}}\ddot{\mathbf{u}})_{\mathbf{v}} + (\mathbf{M}^{\mathbf{vu}}\ddot{\mathbf{u}})_{\mathbf{u}}\mathbf{u}_{\mathbf{v}} + \Phi_{\mathbf{v}}^{\mathbf{T}}\lambda_{\mathbf{v}} + (\Phi_{\mathbf{v}}^{\mathbf{T}}\lambda)_{\mathbf{v}} \\ & \quad + (\Phi_{\mathbf{v}}^{\mathbf{T}}\lambda)_{\mathbf{u}}\mathbf{u}_{\mathbf{v}} = \mathbf{Q}_{\mathbf{v}}^{\mathbf{v}} + \mathbf{Q}_{\mathbf{u}}^{\mathbf{v}}\mathbf{u}_{\mathbf{v}} + \mathbf{Q}_{\dot{\mathbf{u}}}^{\mathbf{v}}\dot{\mathbf{u}}_{\mathbf{v}} \end{aligned} \tag{23}$$

The quantities $\mathbf{u}_{\mathbf{v}}$, $\dot{\mathbf{u}}_{\mathbf{v}}$, $\ddot{\mathbf{u}}_{\mathbf{v}}$, and $\lambda_{\mathbf{v}}$ are obtained by taking partials with respect to $\mathbf{v}$ of equations (7c), (7d), (7e), and (7b), respectively. This is an exercise in chain rule differentiation that yields [**15**, **16**]

$$\mathbf{u}_{\mathbf{v}} = -\Phi_{\mathbf{u}}^{-1}\Phi_{\mathbf{v}} \equiv \mathbf{H} \tag{24a}$$

$$\dot{\mathbf{u}}_{\mathbf{v}} = -\Phi_{\mathbf{u}}^{-1}[(\Phi_{\mathbf{q}}\dot{\mathbf{q}})_{\mathbf{v}} + (\Phi_{\mathbf{q}}\dot{\mathbf{q}})_{\mathbf{u}}\mathbf{H}] \equiv \mathbf{J} \tag{24b}$$

$$\ddot{\mathbf{u}}_{\mathbf{v}} = \mathbf{H}\mathbf{J}_1 + \mathbf{L} \tag{24c}$$

$$\lambda_{\mathbf{v}} = -\Phi_{\mathbf{u}}^{-T}[\mathbf{R} + (\mathbf{M}^{\mathbf{uv}} + \mathbf{M}^{\mathbf{uu}}\mathbf{H})\mathbf{J}_1] \tag{24d}$$

where

$$\mathbf{L} = \Phi_{\mathbf{u}}^{-1}\{[\tau_{\mathbf{u}} - (\Phi_{\mathbf{q}}\ddot{\mathbf{q}})_{\mathbf{u}}]\mathbf{H} + \tau_{\mathbf{v}} + \tau_{\dot{\mathbf{u}}}\mathbf{J} - (\Phi_{\mathbf{q}}\dot{\mathbf{q}})_{\mathbf{v}}\} \quad (25)$$

$$\mathbf{R} = [(\Phi_{\mathbf{u}}^{T}\lambda)_{\mathbf{u}} + (\mathbf{M}^{\mathbf{u}}\ddot{\mathbf{q}})_{\mathbf{u}} - \mathbf{Q}_{\mathbf{u}}^{u}]\mathbf{H} - \mathbf{Q}_{\mathbf{v}}^{u} - \mathbf{Q}_{\dot{\mathbf{u}}}^{u}\mathbf{J} + (\Phi_{\mathbf{u}}^{T}\lambda)_{\mathbf{v}} + (\mathbf{M}^{\mathbf{u}}\ddot{\mathbf{q}})_{\mathbf{v}} + \mathbf{M}^{\mathbf{uu}}\mathbf{L}, \quad \text{with } \mathbf{M}^{\mathbf{u}} = [\mathbf{M}^{\mathbf{uv}}, \mathbf{M}^{\mathbf{uu}}] \quad (26)$$

Substituting the expressions for $\mathbf{u}_{\mathbf{v}}$, $\dot{\mathbf{u}}_{\mathbf{v}}$, $\ddot{\mathbf{u}}_{\mathbf{v}}$, and $\lambda_{\mathbf{v}}$ into equation (23) and denoting $\mathbf{M}^{\mathbf{v}} = [\mathbf{M}^{\mathbf{vv}}, \mathbf{M}^{\mathbf{vu}}]$ yields

$$\hat{\mathbf{M}}\mathbf{J}_1 = \mathbf{Q}_{\mathbf{v}}^{\mathbf{v}} + \mathbf{Q}_{\mathbf{u}}^{\mathbf{v}}\mathbf{H} + \mathbf{Q}_{\dot{\mathbf{u}}}^{\mathbf{v}}\mathbf{J} - [\mathbf{M}^{\mathbf{vu}}\mathbf{L} + \mathbf{H}^{\mathrm{T}}\mathbf{R} + (\Phi_{\mathbf{v}}^{\mathrm{T}}\lambda)_{\mathbf{u}}\mathbf{H} + (\Phi_{\mathbf{v}}^{\mathrm{T}}\lambda)_{\mathbf{v}} + (\mathbf{M}^{\mathbf{v}}\ddot{\mathbf{q}})_{\mathbf{v}} + (\mathbf{M}^{\mathbf{v}}\ddot{\mathbf{q}})_{\mathbf{u}}\mathbf{H}] \quad (27)$$

According to Lemma 1, the coefficient matrix in this multiple right side linear system is positive definite. Therefore, equation (27) properly defines the derivative $\mathbf{J}_1$.

Computation of $\mathbf{J}_2 = \ddot{\mathbf{v}}_{\dot{\mathbf{v}}}$ follows the steps taken for the computation of $\mathbf{J}_1$. Taking the derivative of equation (7a) with respect to $\dot{\mathbf{v}}$ yields

$$\mathbf{M}^{\mathbf{vv}}\mathbf{J}_2 + \mathbf{M}^{\mathbf{vu}}\ddot{\mathbf{u}}_{\dot{\mathbf{v}}} + \Phi_{\mathbf{v}}^{T}\lambda_{\dot{\mathbf{v}}} = \mathbf{Q}_{\mathbf{u}}^{\mathbf{v}}\dot{\mathbf{u}}_{\dot{\mathbf{v}}} + \mathbf{Q}_{\dot{\mathbf{v}}}^{\mathbf{v}} \quad (28)$$

All derivatives in this expression are available, except the quantities $\mathbf{J}_2$, $\dot{\mathbf{u}}_{\dot{\mathbf{v}}}$, $\ddot{\mathbf{u}}_{\dot{\mathbf{v}}}$, and $\lambda_{\dot{\mathbf{v}}}$. The last three derivatives are obtained by taking partial derivatives with respect to the independent velocity $\dot{\mathbf{v}}$ of equations (7d), (7e), and (7b). By repeatedly applying the chain rule of differentiation these derivatives are obtained as

$$\dot{\mathbf{u}}_{\dot{\mathbf{v}}} = \mathbf{H} \quad (29a)$$

$$\ddot{\mathbf{u}}_{\dot{\mathbf{v}}} = \mathbf{N} + \mathbf{H}\mathbf{J}_2 \quad (29b)$$

$$\lambda_{\dot{\mathbf{v}}} = \Phi_{\mathbf{u}}^{-T}[\mathbf{Q}_{\dot{\mathbf{u}}}^{\mathbf{u}}\mathbf{H} + \mathbf{Q}_{\dot{\mathbf{v}}}^{\mathbf{u}} - \mathbf{M}^{\mathbf{uu}}\mathbf{N} - (\mathbf{M}^{\mathbf{uv}} + \mathbf{M}^{\mathbf{uu}}\mathbf{H})\mathbf{J}_2] \quad (29c)$$

where

$$\mathbf{N} = \Phi_{\mathbf{u}}^{-1}\left(\tau_{\dot{\mathbf{u}}}\mathbf{H} + \tau_{\dot{\mathbf{v}}}\right) \quad (30)$$

Substituting these results into equation (28), the derivative of independent accelerations with respect to independent velocities is obtained as the solution of the multiple right-side system of linear equations

$$\hat{\mathbf{M}}\mathbf{J}_2 = \mathbf{W} - \mathbf{H}^{T}\mathbf{X} \quad (31)$$

where

$$\mathbf{W} = \mathbf{Q}_{\dot{\mathbf{u}}}^{\mathbf{v}}\mathbf{H} + \mathbf{Q}_{\dot{\mathbf{v}}}^{\mathbf{v}} - \mathbf{M}^{\mathbf{vu}}\mathbf{N}$$

$$\mathbf{X} = -(\mathbf{Q}_{\dot{\mathbf{u}}}^{\mathbf{u}}\mathbf{H} + \mathbf{Q}_{\dot{\mathbf{v}}}^{\mathbf{u}} - \mathbf{M}^{\mathbf{uu}}\mathbf{N})$$

With $\hat{\mathbf{M}}$ positive definite, equation (31) properly defines the derivative $\mathbf{J}_2$.

Up to this point, a general framework has been provided in this section to analytically express the integration Jacobian required by the Rosenbrock family of integration formulas. Although rather involved in form, these derivatives are obtained in a straightforward way. Furthermore, they are generic, in the sense that they apply to any mechanical system simulation. It remains to provide all the ingredients that explicitly or implicitly enter the right side of equations (27) and (31). These derivatives change according to the modeling elements used to represent a mechanical system. What makes the approach viable is the fact that even these derivatives can be generated in a completely generic way. This is solely a mechanical system modeling task that hinges upon the fact that, in multibody dynamics, all modeling elements are broken down into primitives. Providing required derivative information for these primitives in Cartesian coordinates is tractable. Derivative information for primitives is then combined to produce derivatives for complex modeling entities. To illustrate this, consider the joints used to connect bodies in a mechanical system model. The vast majority of them can be obtained starting from four simple constraint primitives [**2**]:

- Φ^{d1} Dot-1 constraint primitive, imposes that two body-fixed non zero vectors $\mathbf{a}_i$ and $\mathbf{a}_j$ belonging to bodies i and j respectively, should be perpendicular at all times; that is, $\Phi^{d1}(\mathbf{a}_i, \mathbf{a}_j) = \mathbf{a}_i^{T}\mathbf{a}_j = 0$.
- Φ^{d2} Dot-2 constraint primitive, imposes that one body-fixed vector $\mathbf{a}_i$ and a vector $\mathbf{d}_{ij}$ defined by two body-fixed points P_i and P_j should be perpendicular at all times; that is, $\Phi^{d2}(\mathbf{a}_i, \mathbf{d}_{ij}) = \mathbf{a}_i^{T}\mathbf{d}_{ij} = 0$.
- Φ^{s} Point constraint primitive, imposes that two body-fixed points P_i and P_j belonging to bodies i and j respectively, should coincide at all times; that is, $\Phi^{S}(P_i, P_j) = P_i \equiv P_j$.
- Φ^{dist} Distance constraint primitive, imposes that the distance between two body-fixed points P_i and P_j belonging to bodies i and j should stay constant and equal to $C > 0$ at all times; that is, $\Phi^{dist}(P_i, P_j, C) = dist(P_i, P_j) = C$.

In this context, a universal joint that allows two relative degrees of freedom between the constrained bodies is defined by requiring that a Dot-1 and a point constraint primitive be simultaneously satisfied throughout the simulation. Likewise, a spherical joint that allows three degrees of freedom (rotational) between the constrained bodies is simply a point constraint primitive, while a revolute joint is the assembly of two Dot-1 and one point constraint primitives.

Analysing the order of derivatives used to compute the required derivative information $\mathbf{J}_1$ and $\mathbf{J}_2$, it can be seen that the highest order is three, and it appears as a result of taking partials of the right side of the acceleration kinematic constraint equation. Consequently, derivatives of Φ^{d1}, Φ^{d2}, Φ^{s}, and Φ^{dist} should be implemented up to order three. Deriving and coding expressions for all derivatives for the modeling primitives up to order three is a one time effort. For details about how these derivatives are obtained for constraint primitives, inertia elements, and forces the reader is referred to reference [**17**]. For the scope of the present

paper, it suffices to assume that the derivatives required to analytically compute the Rosenbrock integration Jacobian are readily available.

4 THE INTEGRATION ALGORITHM

The implementation of a Rosenbrock–Nystrom based method is summarized in Algorithm 1.

Algorithm 1

1. Initialize simulation
2. Set integration tolerance
3. While (time < time-end) do
4. Set macro-step
5. Get integration Jacobian
6. Factor integration Jacobian
7. Get time derivative
8. Resolve stages $1, \ldots, s$
9. Get solution. Check accuracy. Determine new step size
10. Recover dependent generalized coordinates
11. Check partition
12. End do

Step 1 initializes the simulation. Based on user-provided values at time t_0, a consistent set of initial conditions $(\mathbf{u}_0, \mathbf{v}_0, \dot{\mathbf{u}}_0, \dot{\mathbf{v}}_0)$; that is, satisfying equations (7c) and (7d), is determined and simulation starting and ending times are defined. User-defined absolute and relative integration tolerances are read in and used to control error in both independent position $\mathbf{v}$, and velocity $\dot{\mathbf{v}}$. Step 4 backs up the system configuration to be used upon a rejected time step. During Step 5, the integration Jacobian is evaluated. Since obtaining $\mathbf{J}_1$ and $\mathbf{J}_2$ is a costly operation, equations (27) and (31) suggest that $\mathbf{z}_i$ is more efficiently computed if the stage linear system of equation (21f) is replaced with an equivalent one obtained by formally multiplying the original linear system with the positive definite matrix $(1/\gamma)\hat{\mathbf{M}}$. The new linear system assumes the form

$$\Pi \mathbf{z}_i = \mathbf{r}_i \tag{32}$$

where, with $\hat{\mathbf{M}}$ and $\hat{\mathbf{Q}}$ defined in equations (10a) and (10b)

$$\Pi = \frac{1}{\gamma^2}\hat{\mathbf{M}} \cdot \mathbf{S} = \frac{1}{\gamma^2}\hat{\mathbf{M}} - \frac{h}{\gamma}\hat{\mathbf{M}}\mathbf{J}_2 - h^2\hat{\mathbf{M}}\mathbf{J}_1 \tag{33}$$

$$\begin{aligned}\mathbf{r}_i = {} & \frac{h}{\gamma}\hat{\mathbf{Q}}(t_n + \alpha_i h, \mathbf{V}_i, \dot{\mathbf{V}}_i) \\ & + h^2\frac{\gamma_i}{\gamma}\left[\hat{\mathbf{M}}\frac{\partial \ddot{\mathbf{v}}}{\partial t}(t_n, \mathbf{v}_n, \dot{\mathbf{v}}_n) + \hat{\mathbf{M}}\mathbf{J}_1\dot{\mathbf{v}}_n\right] \\ & + \frac{1}{\gamma}\hat{\mathbf{M}}\sum_{j=1}^{i-1} c_{ij}\mathbf{z}_j + \frac{h^2}{\gamma}\hat{\mathbf{M}}\mathbf{J}_1\sum_{j=1}^{i-1}\delta_{ij}\mathbf{z}_j\end{aligned} \tag{34}$$

Thus, the linear systems in equations (27) and (31) need not be solved for $\mathbf{J}_1$ and $\mathbf{J}_2$. Finding only the right side of these two linear systems suffices to compute the coefficient matrix Π and right side $\mathbf{r}_i$ at each integration stage. In reference [**10**] it is shown that the matrix Π of equation (32) is the integration Jacobian that also appears in the context of multistep implicit integration of the DAE of multibody dynamics. This observation allows for a unitary implementation of implicit methods, based on either singly diagonal implicit Runge–Kutta formulas or on multistep BDF-type formulas.

The matrix Π is factored during Step 6. The dimension of this matrix is equal to the number of degrees of freedom of the mechanical system model, and is typically small.

During Step 7, the quantity $\hat{\mathbf{M}}\partial\ddot{\mathbf{v}}/\partial t$ needed to compute the stage right side $\mathbf{r}_i$ is evaluated in the consistent configuration $(\mathbf{u}_n, \mathbf{v}_n, \dot{\mathbf{u}}_n, \dot{\mathbf{v}}_n)$ from the beginning of each macro-step. As the position kinematic constraints in equation (2a) are assumed time independent, $\hat{\mathbf{M}}$ does not depended on time. Therefore, using equation (9) and equation (10b)

$$\hat{\mathbf{M}}\frac{\partial \ddot{\mathbf{v}}}{\partial t} = \frac{\partial}{\partial t}\hat{\mathbf{Q}}(t, \mathbf{v}_n, \dot{\mathbf{v}}_n) = \mathbf{Q}_t^{\mathbf{v}} + \mathbf{H}^T\mathbf{Q}_t^{\mathbf{u}} \tag{35}$$

The simplifying assumptions made in computing $\hat{\mathbf{M}}\partial\ddot{\mathbf{v}}/\partial t$ in equation (35) are that the kinematic constraint equations are time independent and holonomic. The first assumption is to quantitatively simplify the presentation. Otherwise, the algorithm would step by step follow the derivation for the time-independent case, with the caveat that terms of the form Φ_t, $\Phi_{\mathbf{u}t}$, Φ_{tt}, and so on, would have to be accounted for. As a result, all derivatives would be more complicated. The second assumption is made because covering the nonholonomic constraint case is a qualitatively different process, which is not targeted by this paper. For a thorough account of the nonholonomic scenario, the reader is referred to reference [**18**]. Finally, note that in the case of scleronomic mechanical systems, $\hat{\mathbf{M}}\partial\ddot{\mathbf{v}}/\partial t = 0$.

Next, the stage variables $\mathbf{z}_i$ of equation (21f) are computed. During each of the four stages of the Rosenbrock–Nystrom algorithm, some or all of the following steps are taken:

(a) Obtain consistent configuration at position and velocity levels;
(b) Compute stage generalized forces $\hat{\mathbf{Q}}_i$;
(c) Compute stage right side $\mathbf{r}_i$ as in equation (35);
(d) Solve stage linear system of equation (32) to obtain $\mathbf{z}_i$

Defining $\mathbf{V}_i = \mathbf{v}_n + h\alpha_i\dot{\mathbf{v}}_n + h\sum_{j=1}^{i-1}\theta_{ij}\mathbf{z}_j$ as indicated by equation (21c), during substep (a) the dependent generalized coordinates $\mathbf{U}^i$ are the solution of the nonlinear system $\Phi(\mathbf{U}^i, \mathbf{V}^i) = \mathbf{0}$. The matrix $\Phi_{\mathbf{u}}$, along with its factorization, is at the cornerstone of the algorithm, and equation (5) guarantees that $\mathbf{U}^i$ is properly defined. Likewise, denoting the stage-independent velocities as $\dot{\mathbf{V}}^i = \dot{\mathbf{v}}_n + \sum_{j=1}^{i-1}\alpha_{ij}\mathbf{z}_j$, the stage-dependent velocities $\dot{\mathbf{U}}_i$ are, as in equation (21d), the solution of the linear system $\Phi_{\mathbf{u}}(\mathbf{U}^i, \mathbf{V}^i)\dot{\mathbf{U}}^i = -\Phi_{\mathbf{v}}(\mathbf{U}^i, \mathbf{V}^i)\dot{\mathbf{V}}^i$. Finally, note that the stage forces $\hat{\mathbf{Q}}_i = \hat{\mathbf{Q}}(t_n + \alpha_i h, \mathbf{V}_i, \dot{\mathbf{V}}_i)$ are computed during sub-step (b), as in equation (10b), using the factorization of the matrix $\Phi_{\mathbf{u}}(\mathbf{U}^i, \mathbf{V}^i)$ available at the end of sub-step (a).

Owing to the particular choice of coefficients defining the Rosenbrock–Nystrom formula and the way in which the code was implemented, each of the four stages has its own particularities. Thus,

- Stage 1 in Step 8 of the algorithm marks the beginning of a new integration step, or equivalently the end of the prior one. Therefore the system is in an assembled configuration and substep (a) above is skipped. During this stage, the matrix Π of equation (32) is evaluated, and generalized accelerations are obtained as a byproduct of the process. Thus, to obtain Π, the matrix $\hat{\mathbf{M}}$ is computed and the dependent constraint sub-Jacobian $\Phi_{\mathbf{u}}$ is factored. The latter is then used to obtain the matrices **H**, **J**, **L**, and **N** of Section 3. Owing to the choice of a constant diagonal element γ for the original Rosenbrock method, the matrix Π is constant and needs to be factored only once. The remaining stages re-use this factorization.
- Stages 2 through s of the Rosenbrock–Nystrom formula follow exactly the sub-steps (a) through (d) outlined above. For stages that do not require an additional force computation $\hat{\mathbf{Q}}_i$ there is no need to provide consistent position and velocity configurations, consequently sub-step (a) is skipped.

During Step 9, the position level independent generalized coordinates at time step $n+1$ are computed according to equation (21a) as $\mathbf{v}_{n+1} = \mathbf{v}_n + h\dot{\mathbf{v}}_n + h\sum_{i=1}^{s} \mu_i \mathbf{z}_i$. Likewise, according to equation (21b), the velocity level independent generalized coordinates are computed as $\dot{\mathbf{v}}_{n+1} = \dot{\mathbf{v}}_n + \sum_{i=1}^{s} b_i \mathbf{z}_i$. The accuracy of the solution is verified using a second approximation of the solution at time $n+1$. The less accurate solution is provided by the embedded order three formula $\hat{\mathbf{v}}_{n+1} = \mathbf{v}_n + h\dot{\mathbf{v}}_n + h\sum_{i=1}^{s} \hat{\mu}_i \mathbf{z}_i$ and $\hat{\dot{\mathbf{v}}}_{n+1} = \dot{\mathbf{v}}_n + \sum_{i=1}^{s} \hat{b}_i \mathbf{z}_i$, and it is used for the purpose of step-size control.

Step 10 computes the dependent positions $\mathbf{u}_{n+1}$ such that they satisfy

$$\|\Phi(\mathbf{u}_{n+1}, \mathbf{v}_{n+1})\|_\infty < tol$$

Dependent velocities are obtained as the solution of the linear system

$$\Phi_{\mathbf{u}}(\mathbf{u}_{n+1}, \mathbf{v}_{n+1})\dot{\mathbf{u}}_{n+1} = -\Phi_{\mathbf{v}}(\mathbf{u}_{n+1}, \mathbf{v}_{n+1})\dot{\mathbf{v}}_{n+1}$$

Note that the factorization of the dependent constraint sub-Jacobian

$$\Phi_{\mathbf{u}}(\mathbf{u}_{n+1}, \mathbf{v}_{n+1})$$

is available, since it was used to compute the dependent generalized positions.

The condition number of dependent constraint sub-Jacobian is used during Step 11 to check the partitioning of the generalized coordinates. The current partitioning is re-used as long as the condition number of the current dependent constraint sub-Jacobian does not exceed by 25 per cent the value of the condition number produced by the most recent partitioning. This empirical value was obtained after performing simulations for various models with different values of this parameter.

5 CONCLUSIONS

A generalized coordinate partitioning based state space implicit integration method is presented for dynamic analysis of multibody systems. The time integration of the resulting state space ODE is based on a Rosenbrock–Nystrom formulation, which is also developed in the paper. Rosenbrock methods are known to be efficient for medium accuracy integration of stiff systems. They do not require the solution of nonlinear systems for the stage values. They also possess optimal linear stability properties for stiff integration. The variant proposed here is specialized for the integration of second-order systems resulting from multibody dynamics analysis, in that it does not explicitly solve for velocities. The computation of exact Jacobians needed by Rosenbrock formulas is discussed in the context of multibody systems. The companion paper [**19**] presents a choice of method coefficients based on a four-stage L-stable Rosenbrock formula and presents numerical results.

ACKNOWLEDGEMENT

This research was supported in part by the US Army Tank-Automotive Research, Development, and Engineering Center (DoD contract number DAAE07-94-R094), a multi-university center led by the University of Michigan. Florian Potra was supported in part by the National Science Foundation under Grant No. 0139701.

REFERENCES

1 **Featherstone, R.** The calculation of robot dynamics using articulated-bodyinertias. *Int. J. Robotics Res.*, 1983, **2**(1), 13–30.
2 **Haug, E. J.** *Computer-Aided Kinematics and Dynamics of Mechanical Systems*, 1989 (Allyn and Bacon, Boston, London, Sydney, Toronto).
3 **Garcia de Jalon, J.** and **Bayo, E.** *Kinematic and Dynamic Simulation of Multibody Systems: The Real-Time Challenge*, 1994 (Springer-Verlag, New York, Mechanical Engineering Series).
4 **Petzold, L. R.** Differential–algebraic equations are not ODE's. *SIAM J. Sci., Stat. Comput.*, 1982, **3**(3), 367–384.
5 **Ascher, U. M.** and **Petzold, L. R.** *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*, 1998 (Society for Industrial & Applied Mathematics, Philadelphia).
6 **Eich-Sollner, E.** and **Fuhrer, C.** *Numerical Methods in Multibody Dynamics*, 1998 (Teubner-Verlag, Stuttgart).
7 **Lubich, C., Nowak, U., Pohle, U.** and **Engstler, C.** MEXX — Numerical software for the integration of constrained mechanical multibody systems. *Mec. Structures and Machines*, 1995, **23**, 473–495.

8 **Potra, F. A.** Implementation of linear multistep methods for solving constrained equations of motion. *SIAM. Numer. Anal.*, 1993, **30**(3), 474–489.

9 **Simeon, B.** MBSPACK—Numerical Integration Software for Constrained Mechanical Motion. *Surveys on Mathematics for Industry*, 1995, 5, 169–202.

10 **Negrut, D.** On the implicit integration of differential-algebraic equations of multibody dynamics. *Ph.D. Thesis*, The University of Iowa, 1998.

11 **Hairer, E., Norsett, S. P.** and **Wanner, G.** *Solving Ordinary Differential Equations I. Nonstiff Problems*, 1993 (Springer-Verlag, Berlin Heidelberg New York).

12 **Wehage, R. A.** and **Haug, E. J.** Generalized coordinate partitioning for dimension reduction in analysis of constrained dynamic systems. *J. Mech. Design* 1982, 104, 247–255.

13 **Atkinson, K. E.** *An Introduction to Numerical Analysis*, 1989 (John Wiley & Sons, New York).

14 **Corwin, L. J.** and **Szczarba, R. H.** *Multivariable Calculus*, 1982 (Marcel Dekker, New York).

15 **Haug, E. J., Negrut, D.** and **Iancu, M.** A state-space based implicit integration algorithm for differential-algebraic equations of multibody dynamics. *Mech. Struct. and Mach.*, 1997, **25**(3), 311–334.

16 **Haug, E. J., Negrut, D.** and **Iancu, M.** In *Implicit Integration of the Equations of Multibody Dynamics* (Eds J. Angeles and E. Zakhariev), volume 161, 1997 (NATO ASI Series: Springer-Verlag).

17 **Serban, R.** Dynamic and sensitivity analysis of multibody systems. *Ph.D. Thesis*, The University of Iowa, 1998.

18 **Rabier, P.** and **Rheinboldt, W. C.** *Nonholonomic Motion of Rigid Mechanical Systems from a DAE Viewpoint*, 2000 (Society for Industrial & Applied Mathematics, Philadelphia).

19 **Negrut, D., Sandu, A., Haug, E. J., Potra, F. A.** and **Sandu, C.** A Rosenbrock–Nystrom state space implicit approach for the dynamic analysis of mechanical systems: II — method and numerical examples. *Proc. Instn Mech. Engrs, Part K: J. Multibody Dynamics*, 2003, **217**, 273–281.