MULTIBODY DYNAMICS 2005, ECCOMAS Thematic Conference
J.M. Goicolea, J. Cuadrado, J.C. García Orden (eds.)
Madrid, Spain, 21–24 June 2005

HIGHER-ORDER ACCURACY IN CONSERVATIVE TIME INTEGRATION WITH EMPHASIS ON ELASTIC TRUSS STRUCTURES

Edward Graham* **and Gordan Jelenić***

*UGS, Parker's House, 46 Regent St.,
Cambridge CB2 1DP, United Kingdom
e-mail: ed_ward_graham@yahoo.co.uk
* Department of Civil Engineering, University of Rijeka.
V.C. Emina 5, Rijeka, Republic of Croatia
e-mail: gordan@gradri.hr

Keywords: conservative time integration, higher-order accuracy, relative equilibria

Abstract. *In this paper, we investigate a general approach to developing higher-order accurate algorithms applied to multi-particle problems (which are closely related to truss structures) that conserve linear and angular momenta in the absence of external forces. The fulfilment of angular momentum conservation in particular satisfies an inherent non-singularity requirement of the matrices in this formula. For a structure with N degrees of freedom and no external forces, such algorithms can have up to 2N(N-1) unspecified parameters (within the matrices); that is to say, this formula allows for up to 2N(N-1) matrix entries to be freely defined without affecting momentum conservation.*

Taking conservation of momenta as a starting point, we show how algorithms of arbitrarily high order of accuracy can be derived without recourse to additional degrees of freedom or extra stages of calculation, by specifying the remaining parameters appropriately. We also show how these algorithms can be further made to preserve states of physical relative equilibria (under the relevant starting conditions) and exhibit the property of time-symmetry, whilst remaining momentum-conserving and arbitrarily accurate. A practical drawback to higher-order instances of these algorithms, however, is that the parameter matrices are no longer sparse (in terms of the finite-element discretisation), although certain modifications to the Newton-Raphson procedure can be made that recover sparsity at the expense of iterative convergence.

Any of these algorithms can easily be made to conserve energy also by means of a simple modification to one of the parameter matrices, but this has a detrimental effect on the ability of the modified algorithm to preserve the paths of relative equilibria. Algorithms that conserve energy in this manner are therefore not as desirable as may first appear, as this modification does affect the numerical sensitivity of the scheme, and can result in a lack of robustness (i.e. no solution returned) with ill-conditioned problems, in particular those involving stiff structures. We discuss the theoretical and practical shortcomings of such schemes, and suggest areas for further investigation.

1 Introduction

In this paper, the possibilities for higher-order accuracy in the design of algorithms to solve non-linear, dynamic problems of structures involving large displacements are investigated. We start by considering *systems with symmetries*, which provide two vector constants of motion, namely the total linear momentum and the total angular momentum, each of which gives information about the qualitative and the quantitative nature of the solution. The importance of algorithmic linear and angular momentum conservation in regard to accuracy was noted by Betsch and Steinmann [9], and classic examples of such algorithms are given by Simo and co-workers [26, 27].

As a consequence of this conservation, an additional property of systems with symmetries is the existence of families of fully integrable solutions, each induced by a particular combination of initial conditions. These are known as *relative equilibrium states*, and give further information about the stability of the system: see Simo *et al.* [24, 25] for a detailed account. Algorithms that conserve momenta can be designed to preserve these relative equilibrium states (when the initial conditions arise) and thus gain additional numerical stability. An analysis of two popular time-integration schemes in this regard is given by Gonzalez and Simo [11] and further discussion on the importance of preservation of relative equilibrium orbits is provided by Armero and Romero [4]. Examples of algorithms designed to preserve relative equilibria include the energy-momentum algorithm of Simo and Tarnow [26], and subsequent algorithms that dissipate energy by Armero and co-workers [3, 4, 5].

One further property of dynamical systems in physics that we touch upon briefly is that of time-symmetry [29], which is closely related to the uniqueness of a continuous dynamic response. In the discrete case, however, it is not certain that, at any given point on the solution, negating the time-step would recover the solution given at the previous point in time. Algorithms which guarantee this are described as *time-symmetric*. An early citation of the importance of this property in the engineering context is due to Argyris *et al.* [1], and further examples of such algorithms relevant to our work are the energy-conserving algorithm proposed independently by Simo and Gonzalez [23] and Reich [21] as well as the symplectic mid-point rule [16, 22, 26, 27] and the so-called assumed distance method [7].

It is widely accepted that an algorithm should be at least second-order accurate (e.g. [17]), and the energy-momentum algorithms of Simo *et al.* mentioned earlier all satisfy this requirement. Various ways to increase the order of accuracy have been proposed. In *composition methods*, greater accuracy is achieved by computing intermediate results at additional points within a single time-step; example algorithms include those given independently by Yoshida [29], Forest [10] and Tarnow and Simo [28]. The procedure involves stepping backwards in time, using a larger time-step size than the original algorithm, which makes the principle less attractive for algorithms that are not time-symmetric, and increases the risk of failure during the non-linear iteration process. Using *finite elements in time*, the accuracy can be prescribed by the degree of the polynomial basis functions chosen; example algorithms include those of Betsch and Steinmann [8] for non-linear dynamics. These schemes bear close resemblance to Gauss Runge-Kutta methods, as described in the Appendix of [8].

Both of these strategies to improve accuracy entail additional computation cost, due to the

calculation of intermediate results or the presence of extra degrees of freedom in the temporal domain of the problem. A third approach, aimed at avoiding this additional cost, is based on *Taylor series expansions* of the state variables; in linear dynamics, this method is equivalent to using Padé approximations to the exact solution. Early work was done along these lines for non-linear analysis by Argyris *et al.* [1, 2], who presented *arbitrarily* accurate algorithms that are time-symmetric, although not conservative. They were followed by LaBudde and Greenspan who produced arbitrarily accurate schemes that also conserve energy and angular momentum for a central-force problem [19], and similar energy-conserving schemes for the N-body problem [20]. These algorithms are not time-symmetric, however, and do not preserve the orbits of relative equilibria when higher than second-order accurate. In both cases, small time-step sizes were necessary to ensure convergence of the non-linear solution procedure.

To achieve our goal of designing algorithms with desirable accuracy characteristics, we will follow this approach. Firstly, however, we aim to equip the algorithms with the properties of energy and momentum conservation, preservation of relative equilibrium states and time symmetry, when the relevant physical principles apply. Subject to retaining these properties of the system, we seek to maximise the order of accuracy of the algorithms we design, and present criteria by which it may be achieved. Where this is not possible, we discuss the importance of conservation and order of accuracy with a view to retaining the most important of these properties. Specifically, we deal with non-linear elasticity of multi-element truss structures, with the view that any progress made in this area will highlight avenues of research for time integration involving problems of two- and three-dimensional continua.

2 Equations of motion

Let $\boldsymbol{r}(\boldsymbol{X},t)$, $\dot{\boldsymbol{r}}(\boldsymbol{X},t) \in I\!R^3$ be the position and velocity at time t of a point $x \in \mathcal{B}$, where $\mathcal{B}_0 \subset I\!R^3$ is the initial volume of the continuum; let $\rho(\boldsymbol{X},t)$ represent the current density of the material, with $\rho_0(\boldsymbol{X}) \equiv \rho(\boldsymbol{X},0)$ the initial density; and let $\boldsymbol{X}$ be the position vector of x in the reference configuration. Introducing *spatial* discretisation of the position vector, we have

$$\boldsymbol{r}(\boldsymbol{X},t) = \boldsymbol{N}(\boldsymbol{X})\boldsymbol{R}(t) \tag{2.1}$$

where $\boldsymbol{N}(\boldsymbol{X}) \in I\!R^{3\times 3N}$ is a matrix of shape functions and $\boldsymbol{R}(t) = \langle \boldsymbol{r}^1(t) \ \ldots \ \boldsymbol{r}^N(t) \rangle \in I\!R^{3N}$ a vector of nodal positions, with N the number of spatial nodes used in the discretisation. (Here and throughout the paper we use the notation $\langle\cdot\rangle$ to describe a column vector.) In this work, we consider systems comprising linear (two-noded) bar elements, with the standard equilibrium equation

$$\boldsymbol{M}\ddot{\boldsymbol{R}} + \boldsymbol{\mathcal{F}}\boldsymbol{R} = \boldsymbol{F}, \tag{2.2}$$

where $\boldsymbol{M}$ is the positive-definite global (symmetric) mass matrix, $\boldsymbol{F}$ is an external force which is in this work taken to be derivable from the work function $U(\boldsymbol{R},t)$ (i.e. $\boldsymbol{F}(\boldsymbol{R},t) = \nabla_{\boldsymbol{R}} U(\boldsymbol{R},t)$), and the symmetric matrix $\boldsymbol{\mathcal{F}}$ has submatrices $\boldsymbol{\mathcal{F}}^{ij} \in I\!R^{3\times 3}$ such that

$$\boldsymbol{\mathcal{F}}^{ij} := \begin{cases} \left(\sum_{k=1}^{N} f_{ik}(l_{ik})\right)\boldsymbol{I}_3 & : \quad i = j, \\ -f_{ij}(l_{ij})\boldsymbol{I}_3 & : \quad i \neq j, \end{cases} \tag{2.3}$$

where $f_{ij} = \frac{\phi^{ij\,\prime}(l_{ij})}{l_{ij}}$ for the elemental strain energy function ϕ^{ij} (ϕ denotes the strain energy of the system) and distance l_{ij}, with i and j denoting a pair of nodes in the finite element mesh. For linear systems, $\boldsymbol{\mathcal{F}}$ is constant, and we have $\boldsymbol{\mathcal{F}}\boldsymbol{R} = \boldsymbol{K}\boldsymbol{U}$, where $\boldsymbol{K}$ is the *stiffness matrix* and

$\boldsymbol{U}$ is a vector of nodal displacements. Here, we emphasise that $\boldsymbol{\mathcal{F}}$ is a function of $\boldsymbol{R}$. In view of our forthcoming time-integration schemes, we split (2.2) into a coupled first-order system with momenta

$$\boldsymbol{P} = \boldsymbol{M}\dot{\boldsymbol{R}} = \langle \boldsymbol{p}^1 \ \ldots \ \boldsymbol{p}^N \rangle \tag{2.4}$$

and positions $\boldsymbol{R}$ as the primary variables. Thus we arrive at

$$\begin{aligned} \dot{\boldsymbol{P}} + \boldsymbol{\mathcal{F}}\boldsymbol{R} &= \boldsymbol{F}, \\ \dot{\boldsymbol{R}} &= \boldsymbol{M}^{-1}\boldsymbol{P} \end{aligned} \tag{2.5}$$

which gives the equations of motion for an assembly of bar elements.

The conservation of momenta ($\boldsymbol{\mathcal{L}} = \int_{\mathcal{B}_0} \rho_0 \dot{\boldsymbol{r}} \mathrm{d}V$ and $\boldsymbol{\mathcal{J}} = \int_{\mathcal{B}_0} \boldsymbol{r} \times \rho_0 \dot{\boldsymbol{r}} \mathrm{d}V$) for a dynamic system with symmetries gives rise to a set of *relative equilibrium states* as solutions to system (2.5) when $\boldsymbol{F} = \boldsymbol{0}$. These are described in detail in [4, 11, 24, 25] and references therein; in the context of truss structures, the motion is such that the structure rotates as a rigid body, with (constant) angular velocity $\boldsymbol{\omega}_0$, where the velocity of the centre of mass $\boldsymbol{v}_0^{\mathrm{c}}$ is aligned with the axis of rotation, which should coincide with one of the principal axes of inertia of the assembly. The initial conditions for relative equilibrium states are given by

$$\begin{aligned} \boldsymbol{\mathcal{F}}_0 \overline{\boldsymbol{R}}_0 + \boldsymbol{\Omega}_0 \boldsymbol{M} \overline{\boldsymbol{V}}_0 &= \boldsymbol{0}, \\ \overline{\boldsymbol{V}}_0 &= \boldsymbol{\Omega}_0 \overline{\boldsymbol{R}}_0, \end{aligned} \tag{2.6}$$

where the *relative* position and velocity vectors $\overline{\boldsymbol{R}}$ and $\overline{\boldsymbol{V}}$ (with respect to the position and velocity of the centre of mass) are given by

$$\begin{aligned} \bar{\boldsymbol{r}}^i &:= \boldsymbol{r}^i - \boldsymbol{r}^{\mathrm{c}};\ 1 \leq i \leq N, \quad \overline{\boldsymbol{R}} := \langle \bar{\boldsymbol{r}}^1 \ \ldots \ \bar{\boldsymbol{r}}^N \rangle \quad \text{and} \\ \dot{\boldsymbol{R}}(0) &= \boldsymbol{V}_0,\ \dot{\boldsymbol{r}}^{\mathrm{c}}(0) = \boldsymbol{v}_0^{\mathrm{c}},\ \overline{\boldsymbol{V}}_0 := \langle \boldsymbol{v}_0^1 - \boldsymbol{v}_0^{\mathrm{c}} \ \ldots \ \boldsymbol{v}_0^N - \boldsymbol{v}_0^{\mathrm{c}} \rangle, \end{aligned}$$

and the skew-symmetric matrix $\boldsymbol{\Omega}_0$ is given by

$$\boldsymbol{\Omega}_0 := \begin{pmatrix} \widehat{\boldsymbol{\omega}}_0 & \boldsymbol{0}_3 & \cdots & \boldsymbol{0}_3 \\ \boldsymbol{0}_3 & \widehat{\boldsymbol{\omega}}_0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \boldsymbol{0}_3 \\ \boldsymbol{0}_3 & \cdots & \boldsymbol{0}_3 & \widehat{\boldsymbol{\omega}}_0 \end{pmatrix} \in I\!R^{3N \times 3N}, \tag{2.7}$$

where the skew-symmetric operator $\widehat{(\cdot)} : I\!R^3 \to I\!R^{3\times 3}$ is defined such that

$$\widehat{\boldsymbol{u}} := \begin{pmatrix} 0 & -u_3 & u_2 \\ u_3 & 0 & -u_1 \\ -u_2 & u_1 & 0 \end{pmatrix} \quad \Longleftrightarrow \quad \widehat{\boldsymbol{u}}\boldsymbol{v} \equiv \boldsymbol{u} \times \boldsymbol{v} \ \ \forall\, \boldsymbol{u}, \boldsymbol{v} \in I\!R^3. \tag{2.8}$$

3 Algorithm derivation

We now describe a family of single-step time-integration schemes to solve system (2.5) approximately, that can be specialised to conserve various constants of motion. We choose to express our algorithms in the following form:

$$\begin{aligned} \frac{1}{\Delta t}\Big(\boldsymbol{P}_\Delta + \boldsymbol{\mathcal{G}}\boldsymbol{P}_{1/2}\Big) &= -\boldsymbol{\mathcal{X}}\boldsymbol{R}_{1/2} + \boldsymbol{F}_a, \\ \frac{1}{\Delta t}\Big(\boldsymbol{R}_\Delta - \boldsymbol{\mathcal{G}}^T\boldsymbol{R}_{1/2}\Big) &= \boldsymbol{\mathcal{M}}^{-1}\boldsymbol{P}_{1/2} - \boldsymbol{V}_a, \end{aligned} \tag{3.1}$$

where $\boldsymbol{\mathcal{G}}$, $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{M}}$ are the parameter matrices (with $\boldsymbol{\mathcal{M}}$ necessarily non-singular), $\boldsymbol{F}_a$ and $\boldsymbol{V}_a$ are algorithmic force and velocity vectors that pertain to the external force; and here and throughout the paper, $(\cdot)_{1/2} := \frac{1}{2}[(\cdot)_n + (\cdot)_{n+1}]$ and $(\cdot)_\Delta := (\cdot)_{n+1} - (\cdot)_n$. Each matrix will consist of unit submatrices:

$$\boldsymbol{\mathcal{G}}^{ij} = g^{ij}\boldsymbol{I}_3, \quad \boldsymbol{\mathcal{X}}^{ij} = x^{ij}\boldsymbol{I}_3 \quad \text{and} \quad \boldsymbol{\mathcal{M}}^{ij} = \mu^{ij}\boldsymbol{I}_3. \tag{3.2}$$

We will refer to this family of momentum-conserving algorithms collectively as Algorithm MC. Note that for the specific case where $\boldsymbol{\mathcal{G}} := \boldsymbol{0}_{3N}$, $\boldsymbol{\mathcal{M}} := \boldsymbol{M}$, $\boldsymbol{F}_a := \boldsymbol{F}$ and $\boldsymbol{V}_a := \boldsymbol{0}$, we have

$$\begin{aligned} \frac{1}{\Delta t}\boldsymbol{P}_\Delta &= -\boldsymbol{\mathcal{X}}\boldsymbol{R}_{1/2} + \boldsymbol{F}, \\ \frac{1}{\Delta t}\boldsymbol{R}_\Delta &= \boldsymbol{M}^{-1}\boldsymbol{P}_{1/2}, \end{aligned} \tag{3.3}$$

which is the form of several familiar time-integration schemes (e.g. [6, 26, 27]), each distinguished by its definition of $\boldsymbol{\mathcal{X}}$.

From (3.1) we can also form the equation

$$\boldsymbol{Z}_\Delta = \boldsymbol{\mathcal{C}}_{n+1}\boldsymbol{Z}_{1/2} + \boldsymbol{Z}_a,$$

where

$$\boldsymbol{\mathcal{C}}_{n+1} := \begin{pmatrix} \boldsymbol{\mathcal{G}}^T & \Delta t\boldsymbol{\mathcal{M}}^{-1} \\ -\Delta t\boldsymbol{\mathcal{X}} & -\boldsymbol{\mathcal{G}} \end{pmatrix} \text{ and } \boldsymbol{Z}_a := \Delta t \begin{Bmatrix} -\boldsymbol{V}_a \\ \boldsymbol{F}_a \end{Bmatrix}. \tag{3.4}$$

In a similar way, we may write

$$\boldsymbol{Z}_{n+1} = \boldsymbol{\mathcal{B}}_{n+1}\boldsymbol{Z}_n + \boldsymbol{Z}_F,$$

with

$$\boldsymbol{\mathcal{B}}_{n+1} := \begin{pmatrix} \boldsymbol{\mathcal{A}} & \boldsymbol{\mathcal{B}} \\ \boldsymbol{\mathcal{C}} & \boldsymbol{\mathcal{D}} \end{pmatrix} \text{ and } \boldsymbol{Z}_F := \begin{Bmatrix} -\boldsymbol{R}_F \\ \boldsymbol{P}_F \end{Bmatrix}. \tag{3.5}$$

This leads to

$$\boldsymbol{\mathcal{B}}_{n+1} = [\boldsymbol{I}_{6N} - \tfrac{1}{2}\boldsymbol{\mathcal{C}}_{n+1}]^{-1}[\boldsymbol{I}_{6N} + \tfrac{1}{2}\boldsymbol{\mathcal{C}}_{n+1}] \quad \text{and} \quad \boldsymbol{Z}_F = [\boldsymbol{I}_{6N} - \tfrac{1}{2}\boldsymbol{\mathcal{C}}_{n+1}]^{-1}\boldsymbol{Z}_a, \tag{3.6}$$

and combining (3.4) and (3.6) yields $\{\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}, \boldsymbol{P}_F, \boldsymbol{R}_F\}$ in terms of $\{\boldsymbol{\mathcal{G}}, \boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{M}}, \boldsymbol{F}_a, \boldsymbol{V}_a\}$, provided all the relevant inverses exist.

4 Properties of the algorithm

We now present the conditions under which the algorithm will conserve momenta, the total energy, preserve relative equilibria and be time-symmetric. Relevant proofs can be found in [12].

4.1 Conservation of momenta and energy

Proposition 1 *Algorithm MC gives the discrete linear momentum derivative as*

$$\frac{1}{\Delta t}\boldsymbol{\mathcal{L}}_\Delta = \sum_{i=1}^{N} \boldsymbol{F}_a^i \tag{4.1}$$

provided that

$$\sum_{i=1}^{N} \boldsymbol{\mathcal{G}}^{ij} = \sum_{i=1}^{N} \boldsymbol{\mathcal{X}}^{ij} = \boldsymbol{0}_3 \; \forall \; 1 \leq j \leq N. \tag{4.2}$$

Thus linear momentum is conserved whenever $\sum_{i=1}^{N} \boldsymbol{F}_a^i = \boldsymbol{0}$.

Proposition 2 *Algorithm MC gives the discrete angular momentum derivative as*

$$\frac{1}{\Delta t} \boldsymbol{\mathcal{J}}_{\Delta} = \sum_{i=1}^{N} \left(\boldsymbol{r}_{1/2}^i \times \boldsymbol{F}_a^i + \boldsymbol{p}_{1/2}^i \times \boldsymbol{v}_a^i \right) \tag{4.3}$$

provided that

$$\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{X}}^T \quad \text{and} \quad \boldsymbol{\mathcal{M}} = \boldsymbol{\mathcal{M}}^T. \tag{4.4}$$

Thus angular momentum is conserved whenever $\boldsymbol{F}_a = \boldsymbol{V}_a = \boldsymbol{0}$.

The total number of free parameters remaining in $\boldsymbol{\mathcal{G}}$, $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{M}}$ after conservation of momenta has been secured is $2N^2 - N$; hence Algorithm MC is a fully general form for single-step momentum-conserving algorithms, provided that (4.2) and (4.4) are satisfied. Note that this means $\boldsymbol{\mathcal{G}}$ and $\boldsymbol{\mathcal{X}}$ are singular, and $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{M}}$ are symmetric.

The discrete total energy at time-step n is given as

$$H_n = \frac{1}{2} \boldsymbol{P}_n \cdot \boldsymbol{M}^{-1} \boldsymbol{P}_n + \phi_n - U_n. \tag{4.5}$$

For Algorithm MC to be energy-conserving, we therefore require

$$H_{n+1} = H_n \tag{4.6}$$

to hold for all n under the appropriate conditions.

Proposition 3 *Algorithm MC gives the discrete energy derivative as*

$$\frac{1}{\Delta t} H_{\Delta} = -\frac{U_{\Delta}^{\mathrm{NC}}}{\Delta t} \tag{4.7}$$

if $\boldsymbol{\mathcal{X}} := \kappa \overline{\boldsymbol{\mathcal{X}}}$ *for some matrix* $\overline{\boldsymbol{\mathcal{X}}}$*, where* κ *is defined by*

$$\begin{aligned} &A\kappa + B \; = \; 0 \quad \text{for} \quad A \; = \; -\Delta t \overline{\boldsymbol{\mathcal{X}}} \boldsymbol{R}_{1/2} \cdot \boldsymbol{M}^{-1} \boldsymbol{P}_{1/2} \quad \text{and} \\ &B \; = \; \phi_{\Delta} - U_{\Delta}^{\mathrm{C}} + (\Delta t \boldsymbol{F}_a - \boldsymbol{\mathcal{G}} \boldsymbol{P}_{1/2}) \cdot \boldsymbol{M}^{-1} \boldsymbol{P}_{1/2}, \end{aligned} \tag{4.8}$$

with U^{C} *and* U^{NC} *denoting the conservative and non-conservative parts of the potential function* U*. Thus energy is conserved whenever* $U^{\mathrm{NC}} = 0$.

Equation (4.8) therefore fixes one more of the free parameters; thus *families of single-step energy-momentum algorithms can have up to* $2N^2 - N - 1$ *free parameters, excluding those related to the external force.*

There exists an alternative procedure to using the above single scalar condition to establish

energy conservation [15, 21, 23, 26], which is in many ways beneficial, but this procedure is necessarily only second-order accurate and is for this reason ommited from consideration in this paper. A more thorough analysis of the two approaches to energy conservation is presented in [14].

One of the serious flaws in using a single scalar condition to establish energy conservation is that, for typical $\boldsymbol{\mathcal{G}}$, $\boldsymbol{\mathcal{M}}$ and $\overline{\boldsymbol{\mathcal{X}}}$, it does not guarantee an energy-conserving solution for all possible configurations and time-step sizes (see e.g. [27]), since κ cannot always be defined by (4.8). Numerical experiments invariably show that adding the energy conservation in this way cannot improve the performance of the underlying scheme when it experiences numerical instability due to the total energy blow-up: results have shown that instead of the energy blow-up, the procedure then fails to provide a solution (i.e., the non-linear solution procedure does not converge) [12, 18].

4.2 Preservation of relative equilibria

We have the exact solution of a relative equilibrium problem as

$$\overline{\boldsymbol{R}}(t) = \exp(t\boldsymbol{\Omega}_0)\overline{\boldsymbol{R}}_0, \quad \dot{\overline{\boldsymbol{R}}}(t) = \exp(t\boldsymbol{\Omega}_0)\overline{\boldsymbol{V}}_0 \quad \text{and} \quad \boldsymbol{r}^{\mathrm{c}}(t) \;=\; \boldsymbol{r}^{\mathrm{c}}_0 + t\boldsymbol{v}^{\mathrm{c}}_0$$

with $\exp(t\hat{\boldsymbol{\Omega}}_0) \equiv \mathrm{diag}[\exp(t\widehat{\boldsymbol{\omega}}_0)]$, where $\overline{\boldsymbol{R}}_0$ and $\overline{\boldsymbol{V}}_0$ given by

$$\overline{\boldsymbol{R}}_0 = \boldsymbol{R}_0 - \boldsymbol{R}^{\mathrm{c}}_0, \;\; \boldsymbol{R}^{\mathrm{c}}_0 := \langle \underbrace{\boldsymbol{r}^{\mathrm{c}}_0 \;\; \ldots \;\; \boldsymbol{r}^{\mathrm{c}}_0}_{N \text{ times}} \rangle, \;\; \overline{\boldsymbol{V}}_0 = \boldsymbol{V}_0 - \boldsymbol{V}^{\mathrm{c}}_0 \;\text{ and }\; \boldsymbol{V}^{\mathrm{c}}_0 := \langle \underbrace{\boldsymbol{v}^{\mathrm{c}}_0 \;\; \ldots \;\; \boldsymbol{v}^{\mathrm{c}}_0}_{N \text{ times}} \rangle.$$

We note from Section 2 that $\boldsymbol{\omega}_0 \times \boldsymbol{v}^{\mathrm{c}}_0 = \boldsymbol{0}$, and that $\boldsymbol{R}_0$ and $\boldsymbol{V}_0$ are defined by (2.6). A *relative equilibrium path* is now defined as the discrete solution

$$\begin{aligned} &\boldsymbol{R}_k = \boldsymbol{R}^{\mathrm{c}}_k + \exp(\lambda_{\mathrm{r}} k\Delta t\boldsymbol{\Omega}_0)\overline{\boldsymbol{R}}_0, \quad \boldsymbol{P}_k = \boldsymbol{M}\Big(\boldsymbol{V}^{\mathrm{c}}_0 + \exp(\lambda_{\mathrm{r}} k\Delta t\boldsymbol{\Omega}_0)\overline{\boldsymbol{V}}_0\Big) \quad \text{and} \\ &\boldsymbol{R}^{\mathrm{c}}_k = \boldsymbol{R}^{\mathrm{c}}_0 + \lambda_{\mathrm{t}} k\Delta t\boldsymbol{V}^{\mathrm{c}}_0 \;\; \forall \;\; k, \end{aligned} \tag{4.9}$$

where λ_{r} and λ_{t} are constants. Note that such a definition of the relative equilibrium path is rather loose, since it considers the translational and the rotational motion to be governed by the independent constants λ_{r} and λ_{t}, which do not have to be equal to one another. Such a path is not necessarily only a time re-parametrisation of the actual solution (the latter would correspond to $\lambda_{\mathrm{r}} = \lambda_{\mathrm{t}}$), as would be the case if motion were only translational or only rotational. For Algorithm MC to produce paths of relative equilibria under initial conditions (2.6), we require that the solution given in (4.9) be inserted into the algorithm without conflict for $k = n$ and $k = n+1$, for certain values of λ_{r} and λ_{t}. We now introduce the notation

$$(\cdot)^{\mathrm{RE}} := (\cdot)\Big|_{\substack{\boldsymbol{R}_n,\, \boldsymbol{P}_n,\, \boldsymbol{R}_{n+1},\, \boldsymbol{P}_{n+1} \text{ defined by (4.9)},\\ \boldsymbol{R}_0,\, \boldsymbol{V}_0 \text{ defined by (2.6) and } \boldsymbol{F} = \boldsymbol{0}}}$$

for a given quantity $(\cdot)$, to denote the value taken when a relative equilibrium solution is in effect.

Proposition 4 *Under initial conditions* (2.6)*, Algorithm MC produces paths of relative equilibria provided that*

$$\begin{gathered} \left(\boldsymbol{\mathcal{X}}^{\mathrm{RE}} - \lambda_{\mathrm{r}}\frac{\tan(\frac{1}{2}\theta)}{\frac{1}{2}\theta}\boldsymbol{\mathcal{F}}_0\right)\overline{\boldsymbol{R}}_0 \;=\; \boldsymbol{0}, \quad \left(\lambda_{\mathrm{r}}\frac{\tan(\frac{1}{2}\theta)}{\frac{1}{2}\theta}\boldsymbol{\mathcal{M}}^{\mathrm{RE}} - \boldsymbol{M}\right)\overline{\boldsymbol{V}}_0 \;=\; \boldsymbol{0}, \\ \boldsymbol{\mathcal{G}}^{\mathrm{RE}} \;=\; \boldsymbol{0}_{3N}, \quad (\boldsymbol{M} - \lambda_{\mathrm{t}}\boldsymbol{\mathcal{M}}^{\mathrm{RE}})\boldsymbol{V}^{\mathrm{c}}_0 \;=\; \boldsymbol{0} \quad \text{and} \quad \boldsymbol{F}^{\mathrm{RE}}_a \;=\; \boldsymbol{V}^{\mathrm{RE}}_a \;=\; \boldsymbol{0} \end{gathered} \tag{4.10}$$

for some fixed $\lambda_{\rm r}$, $\lambda_{\rm t}$, where $\theta := \lambda_{\rm r}||\boldsymbol{\omega}_0||\Delta t$ and $-\pi < \theta < \pi$, and that for all n it gives a unique solution for $\boldsymbol{R}_{n+1}$ and $\boldsymbol{P}_{n+1}$ given $\boldsymbol{R}_n$, $\boldsymbol{P}_n$ and Δt.

Equation (4.10) therefore gives criteria that $\boldsymbol{\mathcal{G}}$, $\boldsymbol{\mathcal{X}}$, $\boldsymbol{\mathcal{M}}$, $\boldsymbol{F}_a$ and $\boldsymbol{V}_a$ must satisfy under initial conditions (2.6) for the algorithm to produce a relative equilibrium path.

It is possible to design algorithms to capture the relative equilibrium solution *exactly*. For example, by defining $\boldsymbol{\mathcal{G}}$, $\boldsymbol{\mathcal{X}}$ and $\boldsymbol{\mathcal{M}}$ such that

$$\begin{aligned} &\boldsymbol{\mathcal{G}}^{\rm RE} = \boldsymbol{0}_{3N} \quad \boldsymbol{\mathcal{X}}^{\rm RE} = \frac{\tan(\frac{1}{2}\omega_0\Delta t)}{\frac{1}{2}\omega_0\Delta t}\boldsymbol{\mathcal{F}}_0 \quad \text{and} \quad \boldsymbol{\mathcal{M}}^{\rm RE} = \boldsymbol{M} + c\Delta t^2\boldsymbol{\mathcal{F}}_0, \\ &\text{where} \quad c = \frac{\frac{1}{2}\omega_0\Delta t - \tan(\frac{1}{2}\omega_0\Delta t)}{(\omega_0\Delta_t)^2\tan(\frac{1}{2}\omega_0\Delta t)} \quad \text{and} \quad \omega_0 := ||\boldsymbol{\omega}_0||, \end{aligned} \tag{4.11}$$

it can be seen that (4.10) is satisfied with $\lambda_{\rm r} = \lambda_{\rm t} = 1$. (Note from (2.8) that $\boldsymbol{\Omega}_0^3 = -\omega_0^2\boldsymbol{\Omega}_0$, and that $\boldsymbol{\mathcal{F}}_0\overline{\boldsymbol{R}}_0 = -\boldsymbol{\Omega}_0\boldsymbol{M}\overline{\boldsymbol{V}}_0$ and $\overline{\boldsymbol{V}}_0 = \boldsymbol{\Omega}_0\overline{\boldsymbol{R}}_0$ from (2.6).) This idea is investigated further in [14].

4.3 Conservation of energy and the preservation of relative equilibria

It can be shown that any algorithm that produces paths of relative equilibria also conserves energy along those paths, thus no inherent conflict arises from having conservation of energy and preservation of relative equilibria within the same algorithm. For global energy conservation as defined in Proposition 3, however, it can be seen that A and B from (4.8) satisfy $A^{\rm RE} = B^{\rm RE} = 0$, and there is no obvious value for

$$\lim_{\substack{A \to A^{\rm RE}, \\ B \to B^{\rm RE}}} \left\{ \frac{-B}{A} \right\}$$

that is uniquely defined regardless of how $A \to A^{\rm RE}$ and $B \to B^{\rm RE}$. Therefore *it is not possible for algorithms that conserve energy via* (4.8) *to preserve relative equilibria*, in that the resulting algorithm will be ill-defined under relative equilibrium conditions. Regarding elemental energy conservation, the energy-momentum mid-point algorithm [15, 21, 23, 26] does preserve relative equilibria [11], although the exact trajectory is not recovered (i.e., $\lambda_{\rm r} \neq \lambda_{\rm t}$). It is shown in [14] that *the energy-momentum mid-point algorithm is the only algorithm of the form* (3.1) *that both conserves energy and preserves relative equilibria.* In particular, *higher-order accurate algorithms of this form cannot both conserve energy and preserve relative equilibria*.

4.4 Time symmetry

An algorithm is described as time-symmetric if, at any given configuration $\boldsymbol{Z}_{n+1}$, applying a negative time-step of $-\Delta t$ recovers the previous configuration $\boldsymbol{Z}_n$ [1, 29]. Since $\boldsymbol{Z}_{n+1} = \boldsymbol{\mathcal{B}}_{n+1}\boldsymbol{Z}_n + \boldsymbol{Z}_F$, an algorithm is thus time-symmetric if

$$\begin{aligned} \boldsymbol{Z}_{n+1} &= \boldsymbol{\mathcal{B}}(\boldsymbol{Z}_{n+1}, \boldsymbol{Z}_n, \Delta t)\boldsymbol{Z}_n + \boldsymbol{Z}_F(\boldsymbol{Z}_{n+1}, \boldsymbol{Z}_n, \Delta t) \quad \Longleftrightarrow \\ \boldsymbol{Z}_n &= \boldsymbol{\mathcal{B}}(\boldsymbol{Z}_n, \boldsymbol{Z}_{n+1}, -\Delta t)\boldsymbol{Z}_{n+1} + \boldsymbol{Z}_F(\boldsymbol{Z}_n, \boldsymbol{Z}_{n+1}, -\Delta t). \end{aligned} \tag{4.12}$$

We now introduce for any quantity $(\cdot)$ the notation

$$(\cdot)^{\rm TS} := (\cdot)\Big|_{\boldsymbol{Z}_{n+1} \leftrightarrow \boldsymbol{Z}_n,\, \Delta t \leftrightarrow -\Delta t}.$$

Proposition 5 *Algorithm MC is time-symmetric if*

$$\boldsymbol{\mathcal{X}}^{\mathrm{TS}} = \boldsymbol{\mathcal{X}},\ \boldsymbol{\mathcal{M}}^{\mathrm{TS}} = \boldsymbol{\mathcal{M}},\ \boldsymbol{\mathcal{G}}^{\mathrm{TS}} = -\boldsymbol{\mathcal{G}},\ \boldsymbol{F}_a^{\mathrm{TS}} = \boldsymbol{F}_a \text{ and } \boldsymbol{V}_a^{\mathrm{TS}} = \boldsymbol{V}_a. \tag{4.13}$$

5 Local accuracy analysis

We now analyse the local accuracy characteristics of Algorithm MC, and investigate its capacity for higher-order accuracy when applied to non-linear problems. This section is elaborated in more detail in [14].

We define the *local error vector* as

$$\boldsymbol{\epsilon} := \boldsymbol{Z}_{n+1} - \boldsymbol{Z}(t_{n+1}) \quad \text{when} \quad \boldsymbol{Z}_n = \boldsymbol{Z}(t_n), \tag{5.1}$$

with $\boldsymbol{Z} = \langle \boldsymbol{R}\ \boldsymbol{P} \rangle$ as before: throughout this section, we will assume the solution at time-step n to be exact, i.e. $\boldsymbol{Z}_n = \boldsymbol{Z}(t_n)$. We also define the *residual vector*

$$\boldsymbol{g}(\boldsymbol{X}) := \boldsymbol{\mathcal{B}}(\boldsymbol{X}, \boldsymbol{Z}_n, \Delta t)\boldsymbol{Z}_n + \boldsymbol{Z}_F(\boldsymbol{X}, \boldsymbol{Z}_n, \boldsymbol{F}, \Delta t) - \boldsymbol{X}, \tag{5.2}$$

where $\boldsymbol{\mathcal{B}}$ and $\boldsymbol{Z}_F$ were introduced in Section 3. Consequently we have

$$\boldsymbol{g}(\boldsymbol{Z}_{n+1}) = \boldsymbol{0} \text{ and hence } \boldsymbol{g}[\boldsymbol{Z}(t_{n+1})] = \boldsymbol{g}(\boldsymbol{Z}_{n+1} - \boldsymbol{\epsilon}) = -\nabla \boldsymbol{g}(\boldsymbol{Z}_{n+1})\boldsymbol{\epsilon} + \mathcal{O}(||\boldsymbol{\epsilon}||^2),$$

where $\nabla \boldsymbol{g}$ is the *Jacobian matrix*. Given that $\boldsymbol{g}(\boldsymbol{X}) \in \mathcal{O}(1)$ (i.e. $\mathcal{O}[\Delta t^0]$) for general $\boldsymbol{X}$, we have $\nabla \boldsymbol{g}(\boldsymbol{Z}_{n+1}) \in \mathcal{O}(1)$ also; thus the dependence of $\boldsymbol{g}[\boldsymbol{Z}(t_{n+1})]$ on Δt reveals the size of the local error $\boldsymbol{\epsilon}$.

Introducing the abbreviations

$$\boldsymbol{\zeta} := \langle\, \boldsymbol{Z}(t_{n+1})\ \boldsymbol{Z}(t_n)\ \Delta t\,\rangle \quad \text{and} \quad \tilde{\boldsymbol{\zeta}} := \langle\, \boldsymbol{Z}(t_{n+1})\ \boldsymbol{Z}(t_n)\ \boldsymbol{F}\ \Delta t\,\rangle,$$

setting $\boldsymbol{X} = \boldsymbol{Z}(t_{n+1})$ in (5.2) to get

$$\boldsymbol{g}[\boldsymbol{Z}(t_{n+1})] := \boldsymbol{\mathcal{B}}(\boldsymbol{\zeta})\boldsymbol{Z}_n + \boldsymbol{Z}_F(\tilde{\boldsymbol{\zeta}}) - \boldsymbol{Z}(t_{n+1}), \tag{5.3}$$

and assuming that $\boldsymbol{Z}(t)$ is analytic in a neighbourhood of t_n, we have

$$\boldsymbol{Z}(t_{n+1}) = \boldsymbol{Z}(t_n + \Delta t) = \sum_{s=0}^{\infty} \frac{\boldsymbol{Z}^{(s)}(t_n)}{s!} \Delta t^s, \tag{5.4}$$

where $(\cdot)^{(s)} \equiv \frac{\mathrm{d}^s}{\mathrm{d}t^s}\{(\cdot)\}$. We now express (2.5) in matrix form as

$$\dot{\boldsymbol{Z}} = \boldsymbol{\Psi}\boldsymbol{Z} + \tilde{\boldsymbol{F}}, \quad \text{where} \quad \boldsymbol{\Psi}(t) = \begin{pmatrix} \boldsymbol{0}_{3N} & \boldsymbol{M}^{-1} \\ -\boldsymbol{\mathcal{F}} & \boldsymbol{0}_{3N} \end{pmatrix} \text{ and } \tilde{\boldsymbol{F}}(t) = \begin{Bmatrix} \boldsymbol{0} \\ \boldsymbol{F} \end{Bmatrix}, \tag{5.5}$$

and derive the series solution of $(5.5)_1$ for known data at time t_n. By repeated differentiation of $(5.5)_1$, we can express the derivative $\boldsymbol{Z}^{(s+1)}$ in terms of lower-order derivatives of $\boldsymbol{Z}$ and $\boldsymbol{\Psi}$, i.e.

$$\begin{aligned} \ddot{\boldsymbol{Z}} &= \dot{\boldsymbol{\Psi}}\boldsymbol{Z} + \boldsymbol{\Psi}\dot{\boldsymbol{Z}} + \dot{\tilde{\boldsymbol{F}}}, \quad \boldsymbol{Z}^{(3)} = \ddot{\boldsymbol{\Psi}}\boldsymbol{Z} + 2\dot{\boldsymbol{\Psi}}\dot{\boldsymbol{Z}} + \boldsymbol{\Psi}\ddot{\boldsymbol{Z}} + \ddot{\tilde{\boldsymbol{F}}}, \\ \boldsymbol{Z}^{(4)} &= \boldsymbol{\Psi}^{(3)}\boldsymbol{Z} + 3\ddot{\boldsymbol{\Psi}}\dot{\boldsymbol{Z}} + 3\dot{\boldsymbol{\Psi}}\ddot{\boldsymbol{Z}} + \boldsymbol{\Psi}\boldsymbol{Z}^{(3)} + \tilde{\boldsymbol{F}}^{(3)}, \ \ldots \end{aligned}$$

Summarising this procedure, we have

$$\boldsymbol{Z}^{(s+1)} = \sum_{r=0}^{s} \binom{s}{r} \boldsymbol{\Psi}^{(s-r)} \boldsymbol{Z}^{(r)} + \tilde{\boldsymbol{F}}^{(s)}, \tag{5.6}$$

where $\binom{s}{r}$ represents the binomial coefficient $\frac{s!}{(s-r)!r!}$. Inserting (5.6) into (5.4) gives

$$\begin{aligned}
\boldsymbol{Z}(t_{n+1}) &= \boldsymbol{Z}_n + \sum_{s=0}^{\infty} \frac{\boldsymbol{Z}_n^{(s+1)}}{(s+1)!} \Delta t^{s+1} \\
&= \boldsymbol{Z}_n + \sum_{s=0}^{\infty} \frac{\Delta t^{s+1}}{(s+1)!} \left[\sum_{r=0}^{s} \binom{s}{r} \boldsymbol{\Psi}_n^{(s-r)} \boldsymbol{Z}_n^{(r)} + \tilde{\boldsymbol{F}}_n^{(s)} \right] \\
&= \dots \\
&= \boldsymbol{Z}_n + \sum_{s=0}^{\infty} \frac{\Delta t^{s+1}}{(s+1)!} \left(\boldsymbol{\Psi}_n^{(s)} \boldsymbol{Z}_n + \tilde{\boldsymbol{F}}_n^{(s)} \right) \\
&\quad + \sum_{s=0}^{\infty} \frac{\Delta t^{s+2}}{(s+2)!} \sum_{r=0}^{s} \binom{s+1}{r+1} \boldsymbol{\Psi}_n^{(s-r)} \left(\boldsymbol{\Psi}^{(r)} \boldsymbol{Z}_n + \tilde{\boldsymbol{F}}^{(r)} \right) \\
&\quad + \sum_{s=0}^{\infty} \frac{\Delta t^{s+3}}{(s+3)!} \sum_{r=0}^{s} \binom{s+2}{r+2} \boldsymbol{\Psi}_n^{(s-r)} \sum_{q=0}^{r} \binom{r+1}{q+1} \boldsymbol{\Psi}^{(r-q)} \boldsymbol{Z}_n^{(q+1)}.
\end{aligned}$$

The expression for $\boldsymbol{Z}(t_{n+1})$ can now be seen to consist of a sum of terms of the form

$$\begin{aligned}
&\left[\sum_{s_0=0}^{\infty} \frac{\Delta t^{s_0+m+1}}{(s_0+m+1)!} \sum_{s_1=0}^{s_0} \binom{s_0+m}{s_1+m} \boldsymbol{\Psi}_n^{(s_0-s_1)} \sum_{s_2=0}^{s_1} \binom{s_1+m-1}{s_2+m-1} \boldsymbol{\Psi}^{(s_1-s_2)} \dots \right. \\
&\left. \dots \sum_{s_m=0}^{s_{m-1}} \binom{s_{m-1}+1}{s_m+1} \boldsymbol{\Psi}^{(s_{m-1}-s_m)} \right] \boldsymbol{\mathcal{V}}
\end{aligned} \tag{5.7}$$

for $m = 0, 1, 2 \dots$, where $\boldsymbol{\mathcal{V}}$ stands for either $\boldsymbol{\Psi}^{(s_m)} \boldsymbol{Z}_n$ or $\tilde{\boldsymbol{F}}^{(s_m)}$. We can write (5.7) more compactly as

$$\sum_{s_0=0}^{\infty} \frac{\Delta t^{s_0+m+1}}{(s_0+m+1)!} \left(\prod_{i=0}^{m-1} \left[\sum_{s_{i+1}=0}^{s_i} \binom{s_i+m-i}{s_{i+1}+m-i} \boldsymbol{\Psi}_n^{(s_i-s_{i+1})} \right] \right) \boldsymbol{\mathcal{V}},$$

with $s_{-1} := -\infty$ in the case $m = 0$. Thus we can express the solution to $(5.5)_1$ at time t_{n+1} in (relatively) compact form as

$$\boldsymbol{Z}(t_{n+1}) = \boldsymbol{\mathcal{B}}^{\mathrm{e}} \boldsymbol{Z}_n + \boldsymbol{Z}_F^{\mathrm{e}}, \tag{5.8}$$

where

$$\begin{aligned}
\boldsymbol{\mathcal{B}}^{\mathrm{e}} &:= \sum_{m=0}^{\infty} \left[\sum_{s_0=0}^{\infty} \frac{\Delta t^{s_0+m+1}}{(s_0+m+1)!} \left(\prod_{i=0}^{m-1} \left[\sum_{s_{i+1}=0}^{s_i} \binom{s_i+m-i}{s_{i+1}+m-i} \boldsymbol{\Psi}_n^{(s_i-s_{i+1})} \right] \boldsymbol{\Psi}_n^{(s_m)} \right) \right] \\
&\quad + \boldsymbol{I}_{6N} \quad \text{and} \\
\boldsymbol{Z}_F^{\mathrm{e}} &:= \sum_{m=0}^{\infty} \left[\sum_{s_0=0}^{\infty} \frac{\Delta t^{s_0+m+1}}{(s_0+m+1)!} \left(\prod_{i=0}^{m-1} \left[\sum_{s_{i+1}=0}^{s_i} \binom{s_i+m-i}{s_{i+1}+m-i} \boldsymbol{\Psi}_n^{(s_i-s_{i+1})} \right] \tilde{\boldsymbol{F}}_n^{(s_m)} \right) \right].
\end{aligned} \tag{5.9}$$

(Note that for $m = 0$, we have $\prod_{i=0}^{m-1}(\cdot)_i := \boldsymbol{I}_{6N}$.) Equations (5.8) and (5.9) thus give the exact solution to the semi-discrete equations of motion (2.5), to which we can compare the solutions obtained from our algorithms. From (5.3), (5.8) and (5.9) we can see that the order of $\boldsymbol{g}[\boldsymbol{Z}(t_{n+1})]$ with respect to Δt is governed by how closely $\boldsymbol{\mathcal{B}}(\boldsymbol{\zeta})$ and $\boldsymbol{Z}_F(\tilde{\boldsymbol{\zeta}})$ match $\boldsymbol{\mathcal{B}}^{\mathrm{e}}$ and $\boldsymbol{Z}_F^{\mathrm{e}}$, respectively. Using (5.3), we see that since $\boldsymbol{Z}_n \in \mathcal{O}(1)$,

$$\boldsymbol{g}[\boldsymbol{Z}(t_{n+1})] \in \mathcal{O}(\Delta t^{p+1}) \iff \boldsymbol{\mathcal{B}}(\boldsymbol{\zeta}) - \boldsymbol{\mathcal{B}}^{\mathrm{e}} \in \mathcal{O}(\Delta t^{p+p_1}),\ \boldsymbol{Z}_F(\tilde{\boldsymbol{\zeta}}) - \boldsymbol{Z}_F^{\mathrm{e}} \in \mathcal{O}(\Delta t^{p+p_2}),$$

where $p_1,\ p_2 \in \mathscr{Z}^+$ are such that $\mathrm{Min}\{p_1, p_2\} = 1$, and thus

$$\boldsymbol{\epsilon} \in \mathcal{O}(\Delta t^{p+1}) \iff \boldsymbol{\mathcal{B}}(\boldsymbol{\zeta}) - \boldsymbol{\mathcal{B}}^{\mathrm{e}} \in \mathcal{O}(\Delta t^{p+p_1}),\ \boldsymbol{Z}_F(\tilde{\boldsymbol{\zeta}}) - \boldsymbol{Z}_F^{\mathrm{e}} \in \mathcal{O}(\Delta t^{p+p_2}). \tag{5.10}$$

Equation (5.10) thus contains the criteria for Algorithm MC to be p^{th}-order accurate. Expressed in terms of parameters $\boldsymbol{\mathcal{G}},\ \boldsymbol{\mathcal{X}},\ \boldsymbol{\mathcal{M}},\ \boldsymbol{F}_a$ and $\boldsymbol{V}_a$, this becomes

$$\begin{aligned} \boldsymbol{\epsilon} \in \mathcal{O}(\Delta t^{p+1}) \iff & \boldsymbol{\mathcal{B}}_s^{\mathrm{e}} - \frac{1}{2}\sum_{r=0}^{s} \boldsymbol{\mathcal{C}}_r \boldsymbol{\mathcal{B}}_{s-r}^{\mathrm{e}} - \frac{1}{2}\boldsymbol{\mathcal{C}}_s = \begin{cases} \boldsymbol{I}_{6N} & : s = 0, \\ \boldsymbol{0}_{6N} & : s = 1, \ldots, p \end{cases} \\ & \text{and} \quad \boldsymbol{Z}_{F,s}^{\mathrm{e}} - \frac{1}{2}\sum_{r=0}^{s} \boldsymbol{\mathcal{C}}_r \boldsymbol{Z}_{F,s-r}^{\mathrm{e}} = \boldsymbol{Z}_{a,s} \ \text{ for } \ s = 0, \ldots, p, \end{aligned} \tag{5.11}$$

where $\boldsymbol{\mathcal{B}}_s^{\mathrm{e}}$ and $\boldsymbol{Z}_{F,s}^{\mathrm{e}}$ are the matrix coefficients to Δt^s in 5.9. From 5.9 and 5.11, criteria for p^{th}-order accuracy follow uniquely provided $(p-1)^{\mathrm{th}}$-order accuracy has been secured. Suppose Algorithm MC is $(p-1)^{\mathrm{th}}$-order accurate for $p \geq 0$. Since $\boldsymbol{\mathcal{B}}_0^{\mathrm{e}} = \boldsymbol{I}_{6N}$ from 5.9 we have, from (5.11),

$$\boldsymbol{\mathcal{C}}_s = \boldsymbol{\mathcal{B}}_s^{\mathrm{e}} - \frac{1}{2}\sum_{r=0}^{s-1} \boldsymbol{\mathcal{C}}_r \boldsymbol{\mathcal{B}}_{s-r}^{\mathrm{e}} \quad \text{and} \quad \boldsymbol{Z}_{a,s} = \boldsymbol{Z}_{F,s}^{\mathrm{e}} - \frac{1}{2}\sum_{r=0}^{s} \boldsymbol{\mathcal{C}}_r \boldsymbol{Z}_{F,s-r}^{\mathrm{e}} \tag{5.12}$$

as the requirements for p^{th}-order accuracy. Since $\boldsymbol{\mathcal{C}}_r;\ 0 \leq r \leq s-1$ and $\boldsymbol{Z}_{F,s}$, $\boldsymbol{\mathcal{B}}_s^{\mathrm{e}};\ 0 \leq s < \infty$ are all known, (5.12) can be immediately solved (in order) to furnish $\boldsymbol{\mathcal{C}}_s$ and $\boldsymbol{Z}_{a,s}$. Hence with appropriate choices for $\boldsymbol{\mathcal{G}},\ \boldsymbol{\mathcal{X}},\ \boldsymbol{\mathcal{M}},\ \boldsymbol{F}_a$ and $\boldsymbol{V}_a$, *Algorithm MC can be made arbitrarily accurate*. The criteria for accuracy up to fifth order are given in Table 1, and are continued up to eighth order in [12].

From Table 1, we see immediately that *for time-integration schemes with constant* $\boldsymbol{\mathcal{G}} := \boldsymbol{\mathcal{G}}_0$ *or* $\boldsymbol{\mathcal{M}} := \boldsymbol{\mathcal{M}}_0$, *the limit is second-order accuracy for problems with general strain energy functions* $\phi(l)$, which is consistent with our results in [13]. We also see that higher-order schemes will not retain the sparsity of time-integration schemes of the form (3.3), and will thus be computationally more expensive.

We now verify that the accuracy requirements given in this section do not conflict with the conservation conditions from Section 4 by encapsulating the fact that higher-order accuracy does not hinder conservation of a physical quantity in the following result:

Proposition 6 *Let* $\boldsymbol{f}[\boldsymbol{Z}(t)]$ *be a constant quantity of the motion governed by* (5.5). *Then*

$$\boldsymbol{\epsilon} \in \mathcal{O}(\Delta t^{p+1}) \implies \boldsymbol{f}(\boldsymbol{Z}_{n+1}) - \boldsymbol{f}(\boldsymbol{Z}_n) \in \mathcal{O}(\Delta t^{p+q}),$$

where $q \geq 1$, *assuming* $\boldsymbol{Z}_n = \boldsymbol{Z}(t_n)$. *That is to say, any* p^{th}*-order algorithm will conserve a constant of motion up to order* p *or higher.*

Table 1: *Cumulative conditions for p^{th}-order accuracy.*

p	conditions needed
0	$\boldsymbol{\mathcal{G}}_0 = \boldsymbol{0}_{3N}$
1	$\boldsymbol{\mathcal{G}}_1 = \boldsymbol{0}_{3N},\ \boldsymbol{\mathcal{X}}_0 = \boldsymbol{\mathcal{F}}_n,\ \boldsymbol{\mathcal{M}}_0 = \boldsymbol{M},\ \boldsymbol{F}_{a,0} = \boldsymbol{F}_n,\ \boldsymbol{V}_{a,0} = \boldsymbol{0}$
2	$\boldsymbol{\mathcal{G}}_2 = \boldsymbol{0}_{3N},\ \boldsymbol{\mathcal{X}}_1 = \frac{1}{2}\dot{\boldsymbol{\mathcal{F}}}_n,\ \boldsymbol{\mathcal{M}}_1 = \boldsymbol{0}_{3N},\ \boldsymbol{F}_{a,1} = \frac{1}{2}\dot{\boldsymbol{F}}_n,\ \boldsymbol{V}_{a,1} = \boldsymbol{0}$
3	$\boldsymbol{\mathcal{G}}_3 = \frac{1}{12}\dot{\boldsymbol{\mathcal{F}}}_n\boldsymbol{M}^{-1},\ \boldsymbol{\mathcal{X}}_2 = \frac{1}{6}\ddot{\boldsymbol{\mathcal{F}}}_n + \frac{1}{12}\boldsymbol{\mathcal{F}}_n\boldsymbol{M}^{-1}\boldsymbol{\mathcal{F}}_n,\ \boldsymbol{\mathcal{M}}_2 = -\frac{1}{12}\boldsymbol{\mathcal{F}}_n,$ $\boldsymbol{F}_{a,2} = \frac{1}{6}\ddot{\boldsymbol{F}}_n + \frac{1}{12}\boldsymbol{\mathcal{F}}_n\boldsymbol{M}^{-1}\boldsymbol{F}_n,\ \boldsymbol{V}_{a,2} = \frac{1}{12}\boldsymbol{M}^{-1}\dot{\boldsymbol{F}}_n$
4	$\boldsymbol{\mathcal{G}}_4 = \frac{1}{24}\ddot{\boldsymbol{\mathcal{F}}}_n\boldsymbol{M}^{-1},\ \boldsymbol{\mathcal{X}}_3 = \frac{1}{24}\Big(\boldsymbol{\mathcal{F}}_n^{(3)} + \boldsymbol{\mathcal{F}}_n\boldsymbol{M}^{-1}\dot{\boldsymbol{\mathcal{F}}}_n + \dot{\boldsymbol{\mathcal{F}}}_n\boldsymbol{M}^{-1}\boldsymbol{\mathcal{F}}_n\Big),$ $\boldsymbol{\mathcal{M}}_3 = -\frac{1}{24}\dot{\boldsymbol{\mathcal{F}}}_n,\ \boldsymbol{F}_{a,3} = \frac{1}{24}\Big(\boldsymbol{F}_n^{(3)} + \boldsymbol{\mathcal{F}}_n\boldsymbol{M}^{-1}\dot{\boldsymbol{F}}_n + \dot{\boldsymbol{\mathcal{F}}}_n\boldsymbol{M}^{-1}\boldsymbol{F}_n\Big),$ $\boldsymbol{V}_{a,3} = \frac{1}{24}\boldsymbol{M}^{-1}\ddot{\boldsymbol{F}}_n$
5	$\boldsymbol{\mathcal{G}}_5 = \frac{1}{80}\boldsymbol{\mathcal{F}}_n^{(3)}\boldsymbol{M}^{-1} + \frac{1}{120}\dot{\boldsymbol{\mathcal{F}}}_n\boldsymbol{M}^{-1}\boldsymbol{\mathcal{F}}_n\boldsymbol{M}^{-1} + \frac{1}{240}\boldsymbol{\mathcal{F}}_n\boldsymbol{M}^{-1}\dot{\boldsymbol{\mathcal{F}}}_n\boldsymbol{M}^{-1},$ $\boldsymbol{\mathcal{X}}_4 = \frac{7}{240}\dot{\boldsymbol{\mathcal{F}}}_n\boldsymbol{M}^{-1}\dot{\boldsymbol{\mathcal{F}}}_n + \frac{1}{80}\Big(\ddot{\boldsymbol{\mathcal{F}}}_n\boldsymbol{M}^{-1}\boldsymbol{\mathcal{F}}_n + \boldsymbol{\mathcal{F}}_n\boldsymbol{M}^{-1}\ddot{\boldsymbol{\mathcal{F}}}_n\Big)$ $+ \frac{1}{120}\Big(\boldsymbol{\mathcal{F}}_n^{(4)} + \boldsymbol{\mathcal{F}}_n\boldsymbol{M}^{-1}\boldsymbol{\mathcal{F}}_n\boldsymbol{M}^{-1}\boldsymbol{\mathcal{F}}_n\Big),$ $\boldsymbol{\mathcal{M}}_4 = -\frac{1}{60}\ddot{\boldsymbol{\mathcal{F}}}_n - \frac{1}{720}\boldsymbol{\mathcal{F}}_n\boldsymbol{M}^{-1}\boldsymbol{\mathcal{F}}_n,$ $\boldsymbol{F}_{a,4} = \frac{7}{240}\dot{\boldsymbol{\mathcal{F}}}_n\boldsymbol{M}^{-1}\dot{\boldsymbol{F}}_n + \frac{1}{80}\Big(\boldsymbol{\mathcal{F}}_n\boldsymbol{M}^{-1}\ddot{\boldsymbol{F}}_n + \ddot{\boldsymbol{\mathcal{F}}}_n\boldsymbol{M}^{-1}\boldsymbol{F}_n\Big)$ $+ \frac{1}{120}\Big(\boldsymbol{F}_n^{(4)} + \boldsymbol{\mathcal{F}}_n\boldsymbol{M}^{-1}\boldsymbol{\mathcal{F}}_n\boldsymbol{M}^{-1}\boldsymbol{F}_n\Big),$ $\boldsymbol{V}_{a,4} = \frac{1}{80}\boldsymbol{M}^{-1}\boldsymbol{F}_n^{(3)} + \frac{1}{120}\boldsymbol{M}^{-1}\boldsymbol{\mathcal{F}}_n\boldsymbol{M}^{-1}\dot{\boldsymbol{F}}_n + \frac{1}{240}\boldsymbol{M}^{-1}\dot{\boldsymbol{\mathcal{F}}}_n\boldsymbol{M}^{-1}\boldsymbol{F}_n$

Hence *any algorithm that does not conserve one or more of the constants of motion must be limited in its order of accuracy*.

6 Example momentum-conserving algorithm

To illustrate the theory developed in Sections 4 and 5, we now present a sample fourth-order accurate, momentum-conserving algorithm that is time-symmetric and preserves relative equilibria; we acknowledge that there are many other possibilities. Due to the inherent inability of higher-order energy-conserving algorithms from this family to preserve the paths of relative equilibria, we prefer this choice to the conservation of energy. An energy–momentum higher-order algorithm (which does not preserve the paths of relative equilibria) is, of course, possible, as is the higher-order energy-conserving and angular-momentum non-conserving (and thus relative equilibrium non-preserving) algorithm due to Labudde and Greenspan[20]. For

fourth-order accuracy we require, from Table 1,

$$\begin{aligned}
\boldsymbol{\mathcal{G}} &= \frac{1}{12}\dot{\boldsymbol{\mathcal{F}}}_n\boldsymbol{M}^{-1}\Delta t^3 + \frac{1}{24}\ddot{\boldsymbol{\mathcal{F}}}_n\boldsymbol{M}^{-1}\Delta t^4 + \mathcal{O}(\Delta t^5),\\
\boldsymbol{\mathcal{X}} &= \boldsymbol{\mathcal{F}}_n + \frac{1}{2}\dot{\boldsymbol{\mathcal{F}}}_n\Delta t + \Big(\frac{1}{6}\ddot{\boldsymbol{\mathcal{F}}}_n + \frac{1}{12}\boldsymbol{\mathcal{F}}_n\boldsymbol{M}^{-1}\boldsymbol{\mathcal{F}}_n\Big)\Delta t^2\\
&\quad + \frac{1}{24}\Big(\boldsymbol{\mathcal{F}}_n^{(3)} + \boldsymbol{\mathcal{F}}_n\boldsymbol{M}^{-1}\dot{\boldsymbol{\mathcal{F}}}_n + \dot{\boldsymbol{\mathcal{F}}}_n\boldsymbol{M}^{-1}\boldsymbol{\mathcal{F}}_n\Big)\Delta t^3 + \mathcal{O}(\Delta t^4),\\
\boldsymbol{\mathcal{M}} &= \boldsymbol{M} - \frac{1}{12}\boldsymbol{\mathcal{F}}_n\Delta t^2 - \frac{1}{24}\dot{\boldsymbol{\mathcal{F}}}_n\Delta t^3 + \mathcal{O}(\Delta t^4),\\
\boldsymbol{F}_a &= \boldsymbol{F}_n + \frac{1}{2}\dot{\boldsymbol{F}}_n\Delta t + \Big(\frac{1}{6}\ddot{\boldsymbol{F}}_n + \frac{1}{12}\boldsymbol{\mathcal{F}}_n\boldsymbol{M}^{-1}\boldsymbol{F}_n\Big)\Delta t^2\\
&\quad + \frac{1}{24}\Big(\boldsymbol{F}_n^{(3)} + \boldsymbol{\mathcal{F}}_n\boldsymbol{M}^{-1}\dot{\boldsymbol{F}}_n + \dot{\boldsymbol{\mathcal{F}}}_n\boldsymbol{M}^{-1}\boldsymbol{F}_n\Big)\Delta t^3 + \mathcal{O}(\Delta t^4) \quad \text{and}\\
\boldsymbol{V}_a &= \frac{1}{12}\boldsymbol{M}^{-1}\dot{\boldsymbol{F}}_n\Delta t^2 + \frac{1}{24}\boldsymbol{M}^{-1}\ddot{\boldsymbol{F}}_n\Delta t^3 + \mathcal{O}(\Delta t^4).
\end{aligned} \tag{6.1}$$

We can therefore define

$$\begin{aligned}
\boldsymbol{\mathcal{G}} &:= \frac{\Delta t^2}{12}\boldsymbol{\mathcal{F}}_\Delta\boldsymbol{M}^{-1}, \quad \boldsymbol{\mathcal{M}} := \boldsymbol{M}\Big(\boldsymbol{M} + \tfrac{\Delta t^2}{12}\boldsymbol{\mathcal{F}}_{1/2}\Big)^{-1}\boldsymbol{M},\\
\boldsymbol{\mathcal{X}} &:= \widetilde{\boldsymbol{\mathcal{X}}} + \frac{\Delta t^2}{12}\widetilde{\boldsymbol{\mathcal{X}}}\boldsymbol{M}^{-1}\widetilde{\boldsymbol{\mathcal{X}}}, \quad \text{where} \quad \widetilde{\boldsymbol{\mathcal{X}}} := \boldsymbol{\mathcal{F}}_{1/2} - \frac{\Delta t}{12}\dot{\boldsymbol{\mathcal{F}}}_\Delta,\\
\boldsymbol{F}_a &:= \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}}\boldsymbol{F}(t)\mathrm{d}t + \frac{\Delta t^2}{12}\boldsymbol{\mathcal{F}}_{1/2}\boldsymbol{M}^{-1}\boldsymbol{F}_{1/2} \quad \text{and} \quad \boldsymbol{V}_a := \frac{\Delta t}{12}\boldsymbol{M}^{-1}\boldsymbol{F}_\Delta
\end{aligned} \tag{6.2}$$

in order to fulfil the respective accuracy criteria, which are verified by taking a Taylor series expansion of each expression. (The definition of $\boldsymbol{F}_a$ obviously presumes the function $\boldsymbol{F}(t)$ to be integrable.) The properties of conservation of energy and momenta and also the preservation of relative equilibria can all be easily verified for this scheme. This algorithm has been numerically tested in [12], where it has been confirmed that appending global energy conservation to it as described in Proposition 3 indeed deteriorates the performance. The algorithm itself, however, still lacks robustness needed for its practical recommendation. Presumably, this is due to the term $\dot{\boldsymbol{\mathcal{F}}}_\Delta = \dot{\boldsymbol{\mathcal{F}}}_{n+1} - \dot{\boldsymbol{\mathcal{F}}}_n$ in the definition of $\boldsymbol{\mathcal{X}}$, which involves dot-product calculations when evaluating the entries $\dot{f}_{ij\,n}$ and $\dot{f}_{ij\,n+1}$, which are known to cause numerical difficulties when dealing with stiff problems [13].

7 Conclusions

In this work, the design of conservative algorithms with higher-order accuracy was investigated for multi-element truss structures (or, equivalently, multi-particle dynamics), and conditions under which they can be developed were given. A general framework for algorithms that conserve linear and angular momenta was presented, and conditions for global energy conservation were also given. Criteria for the preservation of relative equilibrium states, time symmetry, and also for higher-order accuracy were elaborated in detail, and the exact solution to a general non-linear problem was derived (in power-series form).

Time-symmetric algorithms that conserve linear and angular momenta can be designed to have arbitrarily high orders of accuracy. This can be achieved without recourse to extra stages or

calculation or additional degrees of freedom in a manner proposed by Argyris *et al.* [1, 2] and LaBudde and Greenspan [19, 20], although the resulting systems of equations will no longer be as sparse for higher-order schemes. These algorithms can also be designed to conserve energy or preserve relative equilibrium states, although they cannot do both. It looks as though in the process of developing efficient time-stepping schemes for structural dynamics (where the problems are inherently stiff), the order of accuracy should not be given too prominent a role. Instead, techniques designed to improve global accuracy should be investigated and, as an example, we have indicated a method to capture the exact relative equilibrium solutions. In future, we intend to explore such algorithms in more detail.

REFERENCES

[1] J.H. Argyris, P.C. Dunne, and T. Angelopoulos. Dynamic response by large step integration. *Earthquake Engineering and Structural Dynamics*, 2:185–203, 1973.

[2] J.H. Argyris, P.C. Dunne, and T. Angelopoulos. Non-linear oscillations using the finite element technique. *Computer Methods in Applied Mechanics and Engineering*, 2:203–250, 1973.

[3] F. Armero and E. Petőcz. Formulation and analysis of conserving algorithms for frictionless dynamic contact/impact problems. *Computer Methods in Applied Mechanics and Engineering*, 158(3–4):269–300, 1998.

[4] F. Armero and I. Romero. On the formulation of high-frequency dissipative time-stepping algorithms for nonlinear dynamics. Part I: low order methods for two model problems and nonlinear elastodynamics. *Computer Methods in Applied Mechanics and Engineering*, 190:2603–2649, 2001.

[5] F. Armero and I. Romero. On the formulation of high-frequency dissipative time-stepping algorithms for nonlinear dynamics. Part II: second order methods. *Computer Methods in Applied Mechanics and Engineering*, 190:6783–6824, 2001.

[6] U. M. Ascher and S. Reich. On some difficulties in integrating highly oscillatory Hamiltonian systems. *Lecture Notes in Computational Science and Engineering*, 4:281–296, 1998.

[7] P. Betsch and P. Steinmann. Conservation properties of a time FE method. Part I: time-stepping schemes for N-body problems. *International Journal for Numerical Methods in Engineering*, 49:599–638, 2000.

[8] P. Betsch and P. Steinmann. Inherently energy conserving time finite elements for classical mechanics. *Journal of Computational Physics*, 160:88–116, 2000.

[9] P. Betsch and P. Steinmann. Conservation properties of a time FE method. Part II: time-stepping schemes for non-linear elastodynamics. *International Journal for Numerical Methods in Engineering*, 50:1931–1955, 2001.

[10] E. Forest. Sixth-order Lie group integrators. *Journal of Computational Physics*, 99:209–213, 1992.

[11] O. Gonzalez and J.C. Simo. On the stability of symplectic and energy-momentum algorithms for non-linear Hamiltonian systems with symmetry. *Computer Methods in Applied Mechanics and Engineering*, 134:197–222, 1996.

[12] E. Graham. *Higher-order accuracy in implicit, conservative, single-step time-integration schemes for non-linear structural dynamics*. PhD thesis, Imperial College of Science, Technology and Medicine, London, England, October 2003.

[13] E. Graham and G. Jelenić. A general framework for conservative single-step time-integration schemes with higher-order accuracy for a central-force system. *Computer Methods in Applied Mechanics and Engineering*, 192:3585–3618, 2003.

[14] E. Graham and G. Jelenić. Conservative single-step time-integration schemes with higher-order accuracy for particle dynamics with local two-point potentials. *submitted*, 2005.

[15] D. Greenspan. Completely conservative, covariant numerical methodology. *Computers and Mathematics with Applications*, 29(4):37–43, 1995.

[16] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II*. Springer-Verlag, Berlin, Germany, second edition, 1996.

[17] T.J.R. Hughes. *The Finite Element method: Linear Static and Dynamic Finite Element Analysis*. Prentice-Hall, Englewood Cliffs, New Jersey, 1987.

[18] D. Kuhl and E. Ramm. Constraint energy momentum algorithm and its application to non-linear dynamics of shells. *Computer Methods in Applied Mechanics and Engineering*, 136:293–315, 1996.

[19] R.A. LaBudde and D. Greenspan. Energy and momentum conserving methods of arbitrary order for the numerical integration of equations of motion I. Motion of a single particle. *Numer. Math.*, 25:323–346, 1976.

[20] R.A. LaBudde and D. Greenspan. Energy and momentum conserving methods of arbitrary order for the numerical integration of equations of motion II. Motion of a system of particles. *Numer. Math.*, 26:1–16, 1976.

[21] S. Reich. Enhancing energy conserving methods. *BIT*, 36:122–134, 1996.

[22] J.M. Sanz-Serna and M.P. Calvo. *Numerical Hamiltonian Problems*. Chapman and Hall, London, 1994.

[23] J.C. Simo and O. Gonzalez. Assessment of energy-momentum and symplectic schemes for stiff dynamical systems. *Papers - American Society of Mechanical Engineers - All Series*, 93(4), 1993. Presented at the ASME Winter Annual Meeting, New Orleans, Louisiana, November 28 - December 3, 1993.

[24] J.C. Simo, D. Lewis, and J.E. Marsden. Stability of relative equilibria. Part I: the reduced energy-momentum method. *Archive for Rational Mechanics and Analysis*, 115:15–59, 1991.

[25] J.C. Simo, T.A. Posbergh, and J.E. Marsden. Stability of relative equilibria. Part II: application to nonlinear elasticity. *Archive for Rational Mechanics and Analysis*, 115:61–100, 1991.

[26] J.C. Simo and N. Tarnow. The discrete energy-momentum method. Conserving algorithms for non-linear elastodynamics. *Journal of Applied Mathematics and Physics (ZAMP)*, 43:757–792, 1992.

[27] J.C. Simo, N. Tarnow, and K.K. Wong. Exact energy-momentum conserving algorithms and symplectic schemes for non-linear dynamics. *Computer Methods in Applied Mechanics and Engineering*, 100:63–116, 1992.

[28] N. Tarnow and J.C. Simo. How to render second order accurate time-stepping algorithms fourth order accurate while retaining the stability and conservation properties. *Computer Methods in Applied Mechanics and Engineering*, 115:233–252, 1994.

[29] H. Yoshida. Construction of higher order symplectic integrators. *Physics Letters A*, 150(5):262–268, 1990.