Preprocessed Discrete Moser–Veselov algorithm for the full dynamics of the rigid body

Ernst Hairer[*] and Gilles Vilmart[†]

August 29, 2006

Abstract

The Discrete Moser–Veselov algorithm is an integrable discretisation of the equations of motion for the free rigid body. It is symplectic and time-reversible, and it conserves all first integrals of the system. The only drawback is its low order. We present a modification of this algorithm to arbitrarily high order which has negligible overhead but considerably improves the accuracy.

Keywords: rigid body, modifying vector field integrator, geometric numerical integration, Discrete Moser–Veselov algorithm.

1 Introduction

The motion of a rigid body, relative to a fixed coordinate system, is described by an orthogonal matrix $Q(t)$. Its dynamics is determined by a Hamiltonian system constrained to the Lie group $SO(3)$. In the absence of an external potential, the Hamiltonian is given by $T = \frac{1}{2}\big(I_1\omega_1^2 + I_2\omega_2^2 + I_3\omega_3^2\big)$, where $(\omega_1, \omega_2, \omega_3)^T$ is the angular velocity in the body frame and the constants I_1, I_2, I_3 are the three moments of inertia of the rigid body. The equations of motion can be written in terms of the angular momentum $y = (y_1, y_2, y_3)^T$, $y_j = I_j\omega_j$, as follows:

$$\dot{y} = \widehat{y}\, I^{-1}y, \qquad \dot{Q} = Q\, \widehat{I^{-1}y}, \tag{1}$$

where $I = \operatorname{diag}(I_1, I_2, I_3)$ (see [5, Sect. VII.5]). We use the standard *hatmap* notation for the correspondence between vectors and skew-symmetric matrices,

$$y = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}, \qquad \widehat{y} = \begin{pmatrix} 0 & -y_3 & y_2 \\ y_3 & 0 & -y_1 \\ -y_2 & y_1 & 0 \end{pmatrix}.$$

[*]University of Geneva, Switzerland.

[†]INRIA Rennes, France, and University of Geneva, Switzerland.

We notice that the flow of (1) exactly conserves the energy and the angular momentum relative to the fixed frame. In formulae, this means that Qy and

$$C(y) = \frac{1}{2}\Big(y_1^2 + y_2^2 + y_3^2\Big) \qquad \text{and} \qquad H(y) = \frac{1}{2}\Big(\frac{y_1^2}{I_1} + \frac{y_2^2}{I_2} + \frac{y_3^2}{I_3}\Big) \tag{2}$$

(Casimir and Hamiltonian) are first integrals of the system.

As numerical integrator we consider the Discrete Moser–Veselov (DMV) algorithm [9] with update for Q_n proposed by [7]. It can be written as

$$\widehat{y}_{n+1} = \omega_n \, \widehat{y}_n \, \omega_n^T, \qquad Q_{n+1} = Q_n \, \omega_n^T, \tag{3}$$

where the orthogonal matrix ω_n is computed from

$$\omega_n^T D - D \, \omega_n = h \, \widehat{y}_n. \tag{4}$$

Here, $y_n \approx y(t_n)$, $Q_n \approx Q(t_n)$, and h is the step size. The entries of the diagonal matrix $D = \operatorname{diag}(d_1, d_2, d_3)$ are determined by

$$d_1 + d_2 = I_3, \quad d_2 + d_3 = I_1, \quad d_3 + d_1 = I_2, \tag{5}$$

so that $\omega^T I \, \omega = \operatorname{trace}(\widehat{\omega} D \, \widehat{\omega}^T)$. It is shown in [8] that this discretisation is equivalent to the RATTLE algorithm which is designed to solve general constrained Hamiltonian systems (see also [6, Chap. 8] and [5, Sect. VII.5.3]).

This Discrete Moser–Veselov algorithm (3)-(4) is an excellent geometric integrator. It exactly conserves (up to round-off) the Hamiltonian $H(y)$, the angular momentum Qy (in the fixed frame) and, since Q is orthogonal, also the Casimir $C(y)$. It is a symmetric (time-reversible) and symplectic discretisation of (1) and therefore well suited for long-time integrations.

The DMV algorithm gives a second order approximation to the solution of (1), and this low order is its only drawback. Based on the ideas of [2] we propose here a modification that allows us to increase the order arbitrarily high, so that a significantly improved accuracy can be obtained. The modification simply consists in replacing the moments of inertia I_j by expressions that depend in a suitable way on $H(y)$ and $C(y)$ (Sect. 2). Numerical experiments and a theoretical justification are presented in Sects. 3 and 4, respectively. An important suggestion for the implementation of the algorithm using quaternions (Sect. 5) and a MAPLE script for the computation of the modified moments of inertia (Sect. 6) conclude this article.

Let us mention that a time transformation has been proposed recently in [8] which improves the order of the DMV algorithm for the angular momentum y but not for the rotation matrix Q. Our modification for the y variables is closely related to but different from this time transformation.

2 Preprocessed DMV algorithm

A technique for increasing the order of numerical methods has recently been proposed in [2] (modifying vector field integrators). It consists in applying the same numerical scheme to a modified differential equation. In the context of the equations of motion for the free rigid body, we consider a modified equation which consists in replacing the moments of inertia I_j by $\widetilde{I}_j = \widetilde{I}_j(y)$ of the form ($j = 1, 2, 3$)

$$\frac{1}{\widetilde{I}_j} = \frac{1}{I_j}\Big(1 + h^2 s_3(y) + h^4 s_5(y) + \ldots\Big) + h^2 d_3(y) + h^4 d_5(y) + \ldots \, . \tag{6}$$

In the DMV algorithm we only have to use $\widetilde{D} = \operatorname{diag}(\widetilde{d}_1, \widetilde{d}_2, \widetilde{d}_3)$ instead of D, where the $\widetilde{d}_j$ are computed from $\widetilde{I}_j = \widetilde{I}_j(y_n)$ via the relations (5).

Theorem 2.1 *There exist two formal series,*

$$\begin{aligned} 1 + h^2 s_3(y) + h^4 s_5(y) + \ldots &= s\big(H(y), C(y)\big), \\ h^2 d_3(y) + h^4 d_5(y) + \ldots &= d\big(H(y), C(y)\big), \end{aligned}$$

depending on y only via $H(y)$ and $C(y)$, such that the DMV algorithm (3)-(4) applied with $\widetilde{I}_j(y_n)$ from (6) yields the exact solution of (1) in the sense of formal power series in h. The first terms of these series are given in Table 1 (see also the appendix of Sect. 6).

Table 1: Scalar functions for the preprocessed DMV algorithm

$$\delta = I_1 I_2 I_3, \qquad \sigma_a = I_1^a + I_2^a + I_3^a, \qquad \tau_{b,c} = \frac{I_2^b + I_3^b}{I_1^c} + \frac{I_3^b + I_1^b}{I_2^c} + \frac{I_1^b + I_2^b}{I_3^c},$$

$$s_3(y) = -\frac{\sigma_{-1}}{3}\, H(y) + \frac{\sigma_1}{6\delta}\, C(y), \qquad d_3(y) = \frac{\sigma_1}{6\delta}\, H(y) - \frac{1}{3\delta}\, C(y),$$

$$s_5(y) = \frac{3\sigma_1 + 2\delta\sigma_{-2}}{60\delta}\, H(y)^2 + \frac{1 - \tau_{1,1}}{30\delta}\, C(y)H(y) + \frac{\sigma_2 - \delta\sigma_{-1}}{30\delta^2}\, C(y)^2,$$

$$d_5(y) = -\frac{9 + \tau_{1,1}}{60\delta}\, H(y)^2 + \frac{6\delta\sigma_{-1} - \sigma_2}{60\delta^2}\, C(y)H(y) - \frac{\sigma_1}{60\delta^2}\, C(y)^2,$$

$$\begin{aligned} s_7(y) &= \frac{15 - \delta\sigma_{-3} - 2\tau_{1,1}}{630\delta}\, H(y)^3 + \frac{6\delta\tau_{1,2} - 100\delta\sigma_{-1} + 53\sigma_2}{2520\delta^2}\, C(y)H(y)^2 \\ &+ \frac{9\sigma_1 + 10\delta\sigma_{-2} - 6\tau_{2,1}}{420\delta^2}\, C(y)^2 H(y) + \frac{4\delta + 17\sigma_3 - 15\delta\tau_{1,1}}{2520\delta^3}\, C(y)^3, \end{aligned}$$

$$\begin{aligned} d_7(y) &= \frac{9\delta\sigma_{-1} + \delta\tau_{1,2} - 11\sigma_2}{1260\delta^2} H(y)^3 + \frac{47\sigma_1 + 13\tau_{2,1} - 38\delta\sigma_{-2}}{2520\delta^2} C(y)H(y)^2 \\ &+ \frac{\sigma_3 + 2\delta\tau_{1,1} - 85\delta}{1260\delta^3}\, C(y)^2 H(y) + \frac{34\delta\sigma_{-1} - 19\sigma_2}{2520\delta^3}\, C(y)^3. \end{aligned}$$

The proof of this theorem is postponed to Sect. 4. We notice that the modified differential equation

$$\dot{y} = \widehat{y}\, \widetilde{I}(y)^{-1} y, \qquad \dot{Q} = Q\, \widehat{\widetilde{I}(y)^{-1} y}, \tag{7}$$

with $\widetilde{I}(y)$ from (6) shares most of the geometric properties with that of (1). It still has Qy, the Casimir $C(y)$, and the Hamiltonian $H(y)$ as first integrals. For the angular momentum this is true for general $s_j(y)$ and $d_j(y)$; for the Hamiltonian only if they depend exclusively on $H(y)$ and $C(y)$. However, the Hamiltonian structure is inherited only if $\widetilde{I}(y)^{-1}y$ is the gradient of a scalar function. This is the case when the series in (6) are truncated after the h^2 term, but not in general.

Theorem 2.1 suggests the following modification of the DMV algorithm.

Algorithm 2.2 (Preprocessed DMV of order $2r$)

1. *Compute the modified moments of inertia $\widetilde{I}_1, \widetilde{I}_2, \widetilde{I}_3$ from (6) truncated after the h^{2r-2} terms and evaluated at y_n.*
2. *Apply the DMV algorithm (3)-(4) to a rigid body with the moments of inertia $\widetilde{I}_1, \widetilde{I}_2, \widetilde{I}_3$ instead of I_1, I_2, I_3.*

For instance, the preprocessed version of order 4 reads

$$\begin{aligned}
\frac{1}{\widetilde{I}_j} &= \frac{1}{I_j}\Big(1 + h^2 s_3(y_n)\Big) + h^2 d_3(y_n), \quad j = 1,2,3,\\
s_3(y_n) &= -\frac{1}{3}\Big(\frac{1}{I_1} + \frac{1}{I_2} + \frac{1}{I_3}\Big) H(y_n) + \frac{I_1 + I_2 + I_3}{6\, I_1\, I_2\, I_3}\, C(y_n),\\
d_3(y_n) &= \frac{I_1 + I_2 + I_3}{6\, I_1\, I_2\, I_3}\, H(y_n) - \frac{1}{3\, I_1\, I_2\, I_3}\, C(y_n).
\end{aligned}$$

Proposition 2.3 *The numerical solution obtained with Algorithm 2.2 satisfies the following properties:*

- *it has order $2r$;*
- *it exactly preserves Qy, $C(y)$, and $H(y)$;*
- *it is symmetric (time-reversible);*
- *restricted to the angular momentum y, it is a Poisson integrator.*

Proof. By Theorem 2.1 the error after one step is a $\mathcal{O}(h^{2r+1})$ perturbation of the exact flow. This implies that the method is of order $2r$.

One step of Algorithm 2.2 is precisely the DMV method with I_j replaced by the constant value $\widetilde{I}_j(y_n)$. Hence, it exactly conserves Qy, $C(y)$ and $\widetilde{H}(y)$, where $\widetilde{H}(y) = \frac{1}{2}\sum_j \widetilde{I}_j(y_n)^{-1} y_j^2$. Due to the particular structure in (6) we have

$$\widetilde{H}(y) = \big(1 + h^2 s_3(y_n) + \ldots\big) H(y) + \big(1 + h^2 d_3(y_n) + \ldots\big) C(y),$$

and the conservation of $C(y)$ and $\widetilde{H}(y)$ implies that of $H(y)$.

The statement on the symmetry follows from the exact conservation of $H(y)$ and $C(y)$, so that $\widetilde{I}_j(y_{n+1}) = \widetilde{I}_j(y_n)$. In Sect. 4.3 we shall show that this algorithm is a Poisson integrator for the angular momentum. □

Remark 2.4 The time transformation of the DMV algorithm (3)-(4) proposed in [8] is equivalent to replace the step size h by a modified step size $\widetilde{h}$ of the form

$$\widetilde{h} = h\big(1 + h^2 s_3(y_n) + h^4 s_5(y_n) + \ldots\big). \tag{8}$$

It is possible to complement this time transformation to obtain high order also for the rotation matrix Q. Since the matrix ω_n^T is orthogonal, it can be represented by a Cayley transform

$$\omega_n^T = \Big(Id + \frac{h}{2}\widehat{I^{-1}Y_n}\Big)\Big(Id - \frac{h}{2}\widehat{I^{-1}Y_n}\Big)^{-1}, \tag{9}$$

where Id stands for the identity matrix, and Y_n is a vector close to y_n. Now, one can use the new update

$$Q_{n+1} = Q_n\, \widetilde{\omega}_n^T,$$

where the matrix $\widetilde{\omega}_n^T$ is defined as in (9), but with modified moments of inertia $\widetilde{I}(y_n) = \operatorname{diag}(\widetilde{I}_1, \widetilde{I}_2, \widetilde{I}_3)$ of the form (6), instead of the diagonal matrix I.

We notice that this modification of the DMV algorithm is not equivalent to the preprocessed DMV Algorithm 2.2 (for y and also for Q). The scalar functions $s_k(y), d_k(y)$ in (8) and in $\widetilde{I}(y_n)$ are the same as in Table 1 for $k = 3$ but not for $k > 3$. Our numerical tests revealed that this modification of DMV is inferior to that of Algorithm 2.2.

3 Comparison with other rigid body integrators

In this section, we compare the preprocessed Discrete Moser–Veselov Algorithm 2.2 (denoted DMV$2r$), with several free rigid body integrators[1]:

- DMV, the *Discrete Moser–Veselov algorithm* (3)-(4),
- IMR$2r$, the *implicit midpoint rule* for $r = 1$, and the *modifying implicit midpoint rule* for $r > 1$, introduced in [2],
- JEM$2r$ [1] where the Euler equations are integrated exactly using Jacobi elliptic functions, and the rotation matrix is approximated using a truncated Magnus series,

[1]The FORTRAN codes used in this section are available from the authors upon request.

- SR$2r$, the so-called *Symmetric+Rotation Splitting* algorithm based on the Strang splitting $H(y) = \frac{1}{2}R(y) + S(y) + \frac{1}{2}R(y)$ where

$$R(y) = \Big(\frac{1}{I_1} - \frac{1}{I_2}\Big)\frac{y_1^2}{2}, \quad S(y) = \frac{1}{2}\Big(\frac{y_1^2+y_2^2}{I_2} + \frac{y_3^2}{I_3}\Big),$$

  combined with a *composition method* of order $2r$ (see for instance [5]). For the numerical experiments, SR4 and SR6 are chosen as compositions of respectively 5 and 9 times the basic method SR2.

Geometric properties In Table 2, we compare the geometric properties of the above integrators. Column "symplectic" indicates whether the method is a symplectic integrator. In the context of backward error analysis (see Sect. 4.3) this means that the modified differential equation is of the form

$$\dot{y} = \widehat{y}\,\nabla H_h^{[2r]}(y), \qquad \dot{Q} = Q\,\widehat{\nabla H_h^{[2r]}}(y).$$

If the modified equation has this form only for the y component, the method is still a Poisson integrator. This is indicated in column "Poisson".

Table 2: Geometric properties

integrator	order of accuracy		exact preservation of quadratic invariants			Poisson	symplectic
	y	Q	Qy	$C(y)$	$H(y)$		
DMV$2r$	$2r$	$2r$	$\surd$	$\surd$	$\surd$	$\surd$	no
DMV	2	2	$\surd$	$\surd$	$\surd$	$\surd$	$\surd$
IMR$2r$	$2r$	$2r$	$\surd$	$\surd$	$\surd$	no[1]	no
JEM$2r$	exact	$2r$	no	$\surd$	$\surd$	$\surd$	no
SR$2r$	$2r$	$2r$	$\surd$	$\surd$	no	$\surd$	$\surd$

Numerical experiments We consider the system (1) for the free rigid body on the interval $[0, 10]$ with two different sets of moments of inertia: an asymmetric body with $I_1 = 0.6$, $I_2 = 0.8$, $I_3 = 1.0$ (as in [1]) and a flat body with $I_1 = 0.345$, $I_2 = 0.653$, $I_3 = 1.0$, which corresponds to the water molecule as considered in [3]. Initial values are $y(0) = (1.8,\, 0.4,\, -0.9)^T$ and $Q(0)$ is the identity matrix.

We have carefully implemented the above integrators in FORTRAN, using quaternions for the rotation matrices in all codes. Since there is no external potential, the invariants $H(y)$, $C(y)$ and the modified moments of inertia are constant along the numerical solution, so they need to be computed

[1]It can be shown that IMR$2r$ ($r \geq 1$) is Poisson with respect to a different bracket generated by $\gamma_h^{[2r]}(y)\,\widehat{y}$, where $\gamma_h^{[2r]}(y)$ is a scalar function (see [4]). We thank an anonymous referee for drawing our attention to this fact.

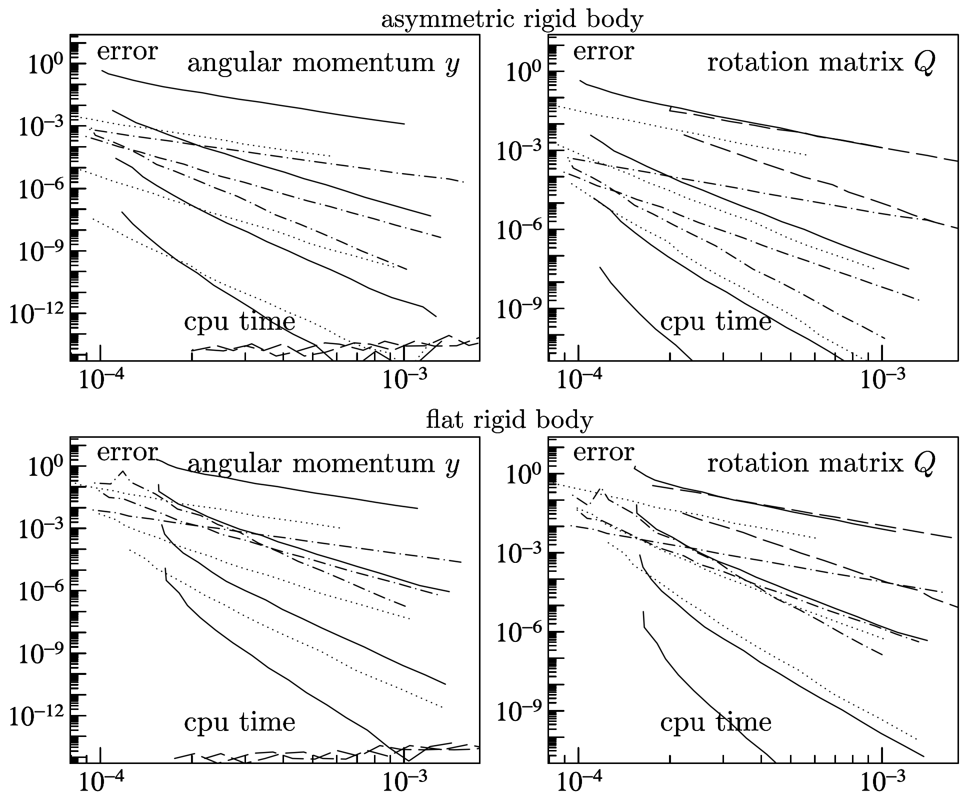

Figure 1: Work-precision diagrams for rigid body integrators: DMV, DMV4, DMV6, DMV8 (solid lines), IMR2, IMR4, IMR6 (dotted lines), JEM2, JEM4 (dashed lines), SR2, SR4, SR6 (dashed-dotted lines).

only once. However, to simulate the presence of an external potential, in our implementation we recalculate them in every step. All codes permit to include an external potential.

For each method and many different step sizes, we plot in Figure 1 the global errors at the endpoint for the angular momentum (left pictures) and the quaternion representation of the rotation matrix (right pictures), as a function of the cpu times (on a SUN Blade 1500 work station). The execution times are taken as the average of 1000 experiments. For symmetric bodies similar results are obtained with the exception that the splitting method used yields the exact solution.

We observe that all methods show the correct order (lines of slope 2, 4, 6, and 8). It is remarkable that the modifying (preprocessed) vector field integrators IMR$2r$ and DMV$2r$ significantly improve the accuracy with increasing order, even if we use very large step sizes. This is due to the fact that the higher order versions have only very little overhead with respect to the basic methods. For example, one step of DMV8 costs only about 50% more cpu time than DMV.

Remark 3.1 In the situation of a small and costly external potential, one would like to use a large step size for DMV$2r$. If it is so large that the iterations for the nonlinear equations do not converge, one can replace the step of DMV$2r$ by m steps with step size h/m. It can be efficiently implemented, because the $\widetilde{I}_k(y_n)$ do not change within these m steps and need to be computed only once. As illustration, 10 steps of DMV8 is only about 4 times more expensive than one step.

4 Proof of the main theorem

The proof of Theorem 2.1 relies heavily on backward error analysis [5, Chapter IX] and on the theory of modified differential equations presented in [2].

4.1 Backward error analysis for DMV

The DMV algorithm is equivalent to RATTLE [8] which is a symplectic discretisation for constrained Hamiltonian systems. Backward error analysis allows one to interpret formally the numerical solution of this method as the exact solution of a modified constrained Hamiltonian system (Theorem IX.5.6 of [5]). For the special case of the rigid body problem it follows from Sect. VII.5.5 of [5] that the modified differential equation is of the form

$$\dot{y} = \widehat{y}\,\nabla H_h(y), \qquad \dot{Q} = Q\,\widehat{\nabla H_h(y)}. \tag{10}$$

where $H_h(y)$ is the modified Hamiltonian,

$$H_h(y) = H(y) + h^2 H_3(y) + h^4 H_5(y) + \dots\ ,$$

so that $y_n = y(nh)$ and $Q_n = Q(nh)$ in the sense of formal power series. It is in even powers of h because the numerical method is symmetric.

Lemma 4.1 *The numerical solution of the DMV algorithm is formally equal to the exact solution of (10) where the modified Hamiltonian $H_h(y)$ depends on y only via the conserved quantities $H(y)$ and $C(y)$,*

$$\begin{aligned} H_h(y) &= K(H(y), C(y)) && (11)\\ K(H,C) &= H + h^2 K_3(H,C) + h^4 K_5(H,C) + \dots\ . \end{aligned}$$

Proof. The Hamiltonian $H(y)$ is a first integral of (10), because it is exactly preserved by the DMV agorithm (see [5, Sect. IX.5.1]), i.e., for all k

$$\nabla H(y)^T\,\widehat{y}\,\nabla H_k(y) = 0.$$

Since $\nabla H(y)$ and $\nabla C(y)$ are also orthogonal to $\widehat{y}\,\nabla H(y)$, the vector $\nabla H_k(y)$ lies in the span of the two vectors $\nabla H(y), \nabla C(y)$, as long as they are linearly independent (which is the case when y is not a stationary point of (1)).

We choose local coordinates $z = \chi(y)$, where $z_1 = H(y)$ and $z_2 = C(y)$, and we define $K_k(z)$ via $K_k(\chi(y)) = H_k(y)$. Since $\nabla H_k(y)$ is a linear combination of $\nabla H(y)$ and $\nabla C(y)$, the function $K_k(z)$ does not depend on the variable z_3. □

The scalar functions $K_j(H,C)$ for the modified Hamiltonian can be computed recursively as

$$\begin{aligned}
K_3(H,C) &= \frac{\sigma_{-1}}{6}\,H^2 - \frac{\sigma_1}{6\delta}\,CH + \frac{1}{6\delta}\,C^2,\\
K_5(H,C) &= \frac{3\sigma_1 + 2\delta\sigma_{-2}}{20\delta}\,H^3 - \frac{7+3\tau_{1,1}}{20\delta}\,CH^2\\
&+ \frac{\sigma_2 + 4\delta\sigma_{-1}}{20\delta^2}\,C^2H - \frac{\sigma_1}{20\delta^2}\,C^3,\\
\ldots
\end{aligned}$$

where the constants $\delta, \sigma_a, \tau_{b,c}$ are those of Table 1.

4.2 The modified moments of inertia

We shall show that the modified equation (10) is of the form (1) with modified moments if inertia.

Lemma 4.2 *The numerical solution of DMV applied to the rigid body problem (1) can be interpreted (formally) as the exact solution of a rigid body problem (7) with modified moments of inertia* $\overline{I}_1, \overline{I}_2, \overline{I}_3$, *given by*

$$\frac{1}{\overline{I}_j} = \frac{1}{I_j}\,\frac{\partial K}{\partial H}\big(H(y),C(y)\big) + \frac{\partial K}{\partial C}\big(H(y),C(y)\big), \quad j=1,2,3, \tag{12}$$

where $K(H,C)$ is the function of Lemma 4.1.

Proof. The special form of the modified Hamiltonian $H_h(y)$ in (11) implies that

$$\nabla H_h(y) = \frac{\partial K}{\partial H}\big(H(y),C(y)\big)\,\nabla H(y) + \frac{\partial K}{\partial C}\big(H(y),C(y)\big)\,\nabla C(y) = \overline{I}^{-1}y,$$

where $\overline{I} = \operatorname{diag}(\overline{I}_1, \overline{I}_2, \overline{I}_3)$ with $\overline{I}_j$ from (12). □

Proof of Theorem 2.1. For fixed y, formula (12) defines a mapping

$$\Psi : (I_1, I_2, I_3) \longmapsto (\overline{I}_1, \overline{I}_2, \overline{I}_3)$$

which is $\mathcal{O}(h^2)$-close to the identity. Notice that in (12) the moments of inertia I_j also appear in $K(H,C)$ and in $H(y)$.

Letting $(\widetilde{I}_1, \widetilde{I}_2, \widetilde{I}_3) = \Psi^{-1}(I_1, I_2, I_3)$, it follows from Lemma 4.2 that the DMV algorithm applied with $\widetilde{I}_j(y_n)$ yields the exact solution of (1). This relation can be reformulated as

$$\frac{1}{\widetilde{I}_j} = \frac{1}{I_j} - h^2\Big(\frac{1}{\widetilde{I}_j}\,\frac{\partial K_3}{\partial H}\big(\widetilde{H}(y),C(y)\big) + \frac{\partial K_3}{\partial C}\big(\widetilde{H}(y),C(y)\big)\Big) - \ldots\ ,$$

where $\widetilde{H}(y) = \frac{1}{2}\sum_j \widetilde{I}_j^{-1} y_j^2$. Formal fixed point iteration shows that the $\widetilde{I}_j$ are of the form (6). □

4.3 Backward error analysis for the preprocessed DMV

We study here symplecticity properties of the preprocessed DMV algorithm. This will be done with help of backward error analysis.

Theorem 4.3 *The numerical solution of the preprocessed DMV Algorithm 2.2 applied to (1) is (formally) the exact solution of*

$$\dot{y} = \widehat{y}\ \overline{I}(y)^{-1}y, \qquad \dot{Q} = Q\ \widehat{\overline{I}(y)^{-1}y}, \tag{13}$$

where $\overline{I}(y)$ is obtained from (12), with I_j replaced by $\widetilde{I}_j^{[2r]}(y)$ given by

$$\frac{1}{\widetilde{I}_j^{[2r]}} = \frac{1}{I_j}\Big(1+h^2 s_3(y)+\ldots+h^{2r-2}s_{2r-1}(y)\Big)+h^2 d_3(y)+\ldots+h^{2r-2}d_{2r-1}(y)\ .$$

Furthermore, there exists a modified Hamiltonian

$$H_h^{[2r]}(y) = H(y) + h^{2r}H_{2r+1}^{[2r]}(y) + h^{2r+2}H_{2r+3}^{[2r]}(y) + \ldots\ ,$$

such that the modified equation for the angular momentum y in (13) has the Poisson structure

$$\dot{y} = \widehat{y}\nabla H_h^{[2r]}(y). \tag{14}$$

Proof. The first statement is an immediate consequence of Lemma 4.2, where the I_j are replaced by $\widetilde{I}_j^{[2r]}$.

The fixed point argument in the proof of Theorem 2.1 implies that

$$\frac{1}{\overline{I}_j} = \frac{1}{I_j}\Big(1 + h^2\sigma_3\big(H(y),C(y)\big) + \ldots\Big) + h^2\delta_3\big(H(y),C(y)\big) + \ldots\ ,$$

for some scalar functions $\sigma_k(H,C), \delta_k(H,C)$, $k = 3,5,\ldots$ Since $\widehat{y}\,y = 0$, the modified equation (13) for y has the form

$$\dot{y} = \widehat{y}\,I^{-1}y\,\big(1 + h^2\sigma_3\big(H(y),C(y)\big) + \ldots\big) = \widehat{y}\,\nabla H_h^{[2r]}(y), \tag{15}$$

where $H_h^{[2r]}(y) = K_h^{[2r]}\big(H(y),C(y)\big)$ and $K_h^{[2r]}(H,C)$ is chosen as an integral with respect to H of the scalar factor $1+h^2\sigma_3(H,C)+\ldots.$ The derivative of $K_h^{[2r]}(H,C)$ with respect to C is not involved in (15), because $\widehat{y}\,\nabla C(y) = 0$. □

Theorem 4.3 implies that the preprocessed DMV Algorithm 2.2 is a Poisson integrator for all orders $2r$. However, in the modified equation (13) for the rotation matrix Q we cannot replace $\overline{I}(y)^{-1}y$ by $\nabla H_h^{[2r]}(y)$. This means that the preprocessed DMV Algorithm 2.2 is not symplectic for the complete system for $r > 1$.

5 Quaternion implementation of DMV

For an efficient implementation, it is a standard approach to use quaternions to represent orthogonal matrices (see [5] in the context of RATTLE and splitting implementations). Let Y_n be the vector defined from ω_n^T through the Cayley transform mentioned in (9). The orthogonal matrix ω_n^T can then be represented by the quaternion ρ_n of norm 1 given by

$$\begin{aligned}\rho_n &= \frac{1}{\sqrt{\alpha_n}}\Big(1+\frac{h}{2}\Big(i\,\frac{Y_{n,1}}{I_1}+j\,\frac{Y_{n,2}}{I_2}+k\,\frac{Y_{n,3}}{I_3}\Big)\Big), \qquad (16)\\ \alpha_n &= 1+\frac{h^2}{4}\Big(\frac{Y_{n,1}^2}{I_1^2}+\frac{Y_{n,2}^2}{I_2^2}+\frac{Y_{n,3}^2}{I_3^2}\Big).\end{aligned}$$

In a similar way, we represent the rotation matrix Q_n by a quaternion q_n. Some algebraic manipulations show that the DMV algorithm (3)-(4) reduces to the following computation, with a simple multiplication of quaternions for the update of the rotation matrix,

$$y_{n+1} = y_n + \alpha_n^{-1} h\, f(Y_n), \qquad q_{n+1} = q_n \cdot \rho_n, \qquad (17)$$

where $f(y) = \widehat{y}\ I^{-1}y$, and α_n, ρ_n are defined in (16). Here, the internal stage Y_n can be computed from the implicit relation

$$Y_n = \alpha_n y_n + \frac{h}{2}\, f(Y_n). \qquad (18)$$

A simple way for solving the nonlinear (quadratic) system (18) is by fixed-point iteration. To improve efficiency, one may calculate the vector $e_n = \frac{h}{2} I^{-1} Y_n$ instead of Y_n, so that the computation of α_n reduces to

$$\alpha_n = 1 + e_{n,1}^2 + e_{n,2}^2 + e_{n,3}^2.$$

Formulae (17) for y_{n+1} and q_{n+1} are explicit. Other approaches for the solution of (4) are discussed in [8]. Suppressing the factor α_n in (17) and in (18), but not in the definition (16) of ρ_n, yields the implicit midpoint rule for problem (1) which is discussed in [2].

6 Appendix

Using a symbolic manipulation package like MAPLE, the functions in Table 1 can be computed formally by comparing the Taylor series of the exact solution of (1), recursively with the series expansion of the DMV algorithm applied with modified moments of inertia. A MAPLE script for this computation is the following:

```
# MAPLE SCRIPT
> with(linalg): Order := 8:
# Modified moments of inertia
> s := 1+h^2*s3+h^4*s5+h^6*s7:
> d := h^2*d3+h^4*d5+h^6*d7:
> i1mod := 1/(s/i1+d): i2mod := 1/(s/i2+d): i3mod := 1/(s/i3+d):
# SERIES EXPANSION OF THE NUMERICAL SOLUTION (DMV)
> e1 := 0: e2 := 0: e3 := 0:
> for i from 1 to 13 do
>   alpha := series(1+e1^2+e2^2+e3^2,h);
>   e1 := series(alpha*h/2*y1/i1mod+(i2mod-i3mod)/i1mod*e2*e3,h);
>   e2 := series(alpha*h/2*y2/i2mod+(i3mod-i1mod)/i2mod*e3*e1,h);
>   e3 := series(alpha*h/2*y3/i3mod+(i1mod-i2mod)/i3mod*e1*e2,h);
> od:
> y1dmv := series(y1+4/h/alpha*(i2mod-i3mod)*e2*e3,h):
# Cayley transform
> Id := matrix(3,3,[1,0,0,0,1,0,0,0,1]):
> ehat := matrix(3,3,[0,-e3,e2,e3,0,-e1,-e2,e1,0]):
> Qdmv := evalm((Id+ehat)&*inverse((Id-ehat))):
# SERIES EXPANSION OF THE EXACT SOLUTION
> fncy := y1(t),y2(t),y3(t):
> fncQ := q11(t),q12(t),q13(t),q21(t),q22(t),q23(t),q31(t),q32(t),q33(t):
> Qexact := matrix(3,3,[fncQ]):
# Equations of motion
> eqy := diff(y1(t),t)=y2(t)*y3(t)*(i2-i3)/i2/i3,
>        diff(y2(t),t)=y1(t)*y3(t)*(i3-i1)/i1/i3,
>        diff(y3(t),t)=y1(t)*y2(t)*(i1-i2)/i2/i1:
> W := matrix(3,3,[0, -y3(t)/i3, y2(t)/i2,
>                  y3(t)/i3, 0, -y1(t)/i1,
>                  -y2(t)/i2, y1(t)/i1, 0]):
> QW := evalm(Qexact&*W):
> eqQ := seq(seq(diff(Qexact[i,j],t)=QW[i,j],j=1..3),i=1..3):
# Initial condition
> init := q11(0)=1,q12(0)=0,q13(0)=0,
>         q21(0)=0,q22(0)=1,q23(0)=0,
>         q31(0)=0,q32(0)=0,q33(0)=1,
>         y1(0)=y1,y2(0)=y2,y3(0)=y3:
# Exact solution
> assign(dsolve({eqy,eqQ,init},{fncy,fncQ},type=series)):
> y1exact := subs(t=h,y1(t)):
> Qexact := simplify(subs(t=h,matrix(3,3,[fncQ]))):
# LOCAL ERROR
> erry := simplify(series(y1exact-y1dmv,h)):
> errQ := simplify(series(Qexact[3,2]-Qdmv[3,2],h)):
# COMPUTATION OF s3,d3,s5,d5,...
> sol := proc(coeff,err,n)
> solve(convert(series(err,h,n),polynom),coeff):
> end:
> s3 := sol(s3,erry,4): d3 := sol(d3,errQ,4):
> s5 := sol(s5,erry,6): d5 := sol(d5,errQ,6):
> s7 := sol(s7,erry,8): d7 := sol(d7,errQ,8):
# DECOMPOSITION as polynomials in H(y) and C(y)
> C:=(y1^2+y2^2+y3^2)/2: H:=(y1^2/i1+y2^2/i2+y3^2/i3)/2:
> decomp := proc(expr,vars)
> solve({subs({y1=1,y2=0,y3=0},expr),subs({y1=0,y2=1,y3=0},expr),
>        subs({y1=0,y2=0,y3=1},expr),subs({y1=1,y2=1,y3=1},expr)},vars);
> end:
> decomp(s3=a1*C+a2*H,{a1,a2});
      /       i2 i3 + i1 i3 + i1 i2       i2 + i1 + i3\
     { a2 = - ---------------------, a1 = ------------ }
      \            3 i1 i2 i3              6 i1 i2 i3 /
> decomp(s5=a1*C^2+a2*C*H+a3*H^2,{a1,a2,a3});
```

Acknowledgement. This work was partially supported by the Fonds National Suisse, project No. 200020-109158.

References

[1] E. Celledoni and N. Säfström. Efficient time-symmetric simulation of torqued rigid bodies using Jacobi elliptic functions. *J. Phys. A*, 39:5463–5478, 2006.

[2] P. Chartier, E. Hairer, and G. Vilmart. Numerical integrators based on modified differential equations. *To appear in Math. Comput.*, 2006.

[3] F. Fassò. Comparison of splitting algorithm for the rigid body. *J. Comput. Phys.*, 189:527–538, 2003.

[4] F. Fassò, A. Giacobbe, and N. Sansonetto. Periodic flows, rank-two Poisson structures, and nonholonomic mechanics. *Regular and Chaotic Dynamics*, 10(3):267–284, 2005.

[5] E. Hairer, C. Lubich, and G. Wanner. *Geometric Numerical Integration. Structure-Preserving Algorithms for Ordinary Differential Equations.* Springer Series in Computational Mathematics 31. Springer-Verlag, Berlin, second edition, 2006.

[6] B. Leimkuhler and S. Reich. *Simulating Hamiltonian Dynamics.* Cambridge Monographs on Applied and Computational Mathematics 14. Cambridge University Press, Cambridge, 2004.

[7] D. Lewis and J. C. Simo. Conserving algorithms for the n-dimensional rigid body. *Fields Inst. Commun.*, 10:121–139, 1996.

[8] R. I. McLachlan and A. Zanna. The discrete Moser–Veselov algorithm for the free rigid body, revisited. *Found. Comput. Math.*, 5:87–123, 2005.

[9] J. Moser and A. P. Veselov. Discrete versions of some classical integrable systems and factorization of matrix polynomials. *Comm. Math. Phys.*, 139:217–243, 1991.