Found. Comput. Math. OF1–OF37 (2004)
© 2004 SFoCM
DOI: 10.1007/s10208-004-0118-6

FOUNDATIONS OF
COMPUTATIONAL
MATHEMATICS
The Journal of the Society for the Foundations of Computational Mathematics

The Discrete Moser–Veselov Algorithm for the Free Rigid Body, Revisited

Robert I. McLachlan[1] and Antonella Zanna[2]

[1]Institute of Fundamental Sciences
Massey University
Private Bag 11-222
Palmerston North, New Zealand.
r.mclachlan@massey.ac.nz

[2]Institutt for Informatik
Universiteit Bergen
Høyteknologisenteret
N-5020 Bergen, Norway
anto@ii.uib.no

Abstract. In this paper we revisit the Moser–Veselov description of the free rigid body in body coordinates, which, in the 3×3 case, can be implemented as an *explicit*, *second-order*, *integrable* approximation of the continuous solution. By backward error analysis, we study the modified vector field which is integrated exactly by the discrete algorithm. We deduce that the discrete Moser–Veselov (DMV) is well approximated to higher order by time reparametrizations of the continuous equations (modified vector field). We use the modified vector field to scale the initial data of the DMV to improve the order of the approximation and show the equivalence of the DMV and the RATTLE algorithm. Numerical integration with these preprocessed initial data is several orders of magnitude more accurate than the original DMV and RATTLE approach.

Date received: January 7, 2004. Final version received: July 8, 2004. Date accepted: July 15, 2004. Communicated by Peter Olver. Online publication: December 23, 2004.
AMS classification: 70E15, 65L05.
Key words and phrases: Rigid body dynamics, Integrable systems, Backward error analysis.

1. Introduction

In 1991 Moser and Veselov published a memorable paper [**12**] in which they described discrete versions of several classical integrable systems, among which, the rigid body (RB). The integrability of these discrete maps was shown with the help of a Lax pair representation corresponding to a factorization of certain matrix polynomials.

There is a large interest in the computational community toward good methods for the integration of the RB equations, since they arise naturally in a number of applications, for instance, celestial mechanics and molecular dynamics. RB equations are used here to follow particles (planets, atoms, molecules, etc.) in between other body–body interactions. It is of fundamental importance that some qualitative features of the system under consideration are preserved under integration, for instance, energy. However, in recent years it has become clear that numerical preservation of energy alone is not enough to obtain ”good” qualitative descriptions of the system. Other properties like symplecticity and time reversibility have been shown to produce a favorable propagation of errors, hence better numerical methods, especially when interested in long-term time behavior.

Another interesting aspect of the work of Moser and Veselov is that their approach is based on the theory of *discrete Lagrangians*, which has become apparent in the computational mechanics community (see, for instance, Marsden and West [**10**] and references therein) because the methods derived by this approach naturally inherit the symplectic structure of the system under consideration, they preserve the discrete Lagrangian, preserve momentum, and preserve energy. However, this approach also has some major drawbacks:

(i) it is difficult to obtain numerical methods of an order higher than 2,
(ii) it is usually difficult to estimate the numerical error; and
(iii) the derived methods (with a few exceptions) are highly implicit, hence computationally expensive.

In fact, the DMV algorithm for the 3×3 RB is an exception, since, as we shall see in the sequel, it can be implemented as an *explicit* numerical method for the reduced dynamics (in body coordinates). In passing, we mention that a second-order accurate equivariant exact energy-momentum conserving symplectic algorithm for the RB was proposed by Lewis and Simo [**8**]. Although this algorithm corresponds to DMV for the reduced dynamics and has the same reduced discrete variational interpretation, it has a different configuration update, which renders the two algorithms, for the full dynamics, different.

Motivated by this, in this paper we revisit the DMV algorithm for the reduced RB equations (body coordinates) and analyze it as a numerical algorithm for their time integration.

The original DMV is a second-order integrable method but, unfortunately, it has a large error constant, so that it is not competitive with other numerical algorithms (like the splitting method of McLachlan [**11**], which is second-order, explicit, reversible, and norm-preserving, or the implicit midpoint rule (IMR), which is second-order, norm- and energy-preserving for this problem but implicit, though remarkably accurate) which are nowadays widely used for the integration of the RB.

Using tools dear to numerical analysis, like *backward error analysis*, we are able to construct a modified vector field which is integrated by the DMV up to order 6 and describe a general procedure for obtaining higher-order approximations. The study of the modified vector field for the 3×3 RB reveals that the DMV algorithm solves a time reparametrization of the original RB equations. This observation is hence used to "preprocess" and "postprocess" the initial condition of the original DMV algorithm so that a fourth- and sixth-order method are obtained, adding only a tiny percentage to the total expense of the original DMV algorithm! These observations also hold for the RATTLE methods for the RB, which is shown to be equivalent to the DMV algorithm. Remarkably, for the 3×3 RB, the DMV algorithm can be implemented as an *explicit* method, without losing any of its original integrability.

The present paper is organized as follows. In the rest of this Section 1 we describe briefly the connection between the DMV algorithm and the discrete Lagrangian approach. In Section 2 we study the modified vector field of the DMV algorithm deriving the components f_3 and f_5 explicitly and sketching the procedure for the higher-order components. In Section 3 we focus on the case of the 3×3 RB equations and obtain explicit expressions for f_3 and f_5 as a constant times $[M, \Omega]$. These constants are then used in Section 4 to rescale the initial condition in order to obtain numerical approximation of order 4 and 6, respectively (DMV4, DMV6). In Section 5 we discuss the behavior of the methods DMV, DMV4, and DMV6 as function of the step size h of integration. In Section 6 we establish the equivalence of the RATTLE method for the RB with the DMV algorithm and conclude that the scalings for order 4 and 6 for DMV can be used to improve RATTLE in an analogous fashion. In Section 7 we present some numerical experiments illustrating the results presented in this paper. In Section 8 we discuss the full dynamics of the RB, including the configuration update. The algorithmic update of the configuration does not correspond to a simple time reparametrization, therefore the whole algorithm (momentum in body coordinates plus configuration) is an overall order 2 method. However, for certain values of the initial condition and the inertia matrix, it is possible to expect substantial error reduction by using a higher-order DMV on the reduced dynamics. Finally, in Section 9, we describe the explicit algorithm used to implement the DMV in this paper.

1.1. *The DMV via the Discrete Lagrangian Approach*

Moser and Veselov [**12**] consider the functional $S(X)$ determined by

$$S = \sum_k \operatorname{tr}(X_k J X_{k+1}^\top), \tag{1.1}$$

where $X = \{X_k\}$ with $X_k \in \mathrm{O}(N)$ and J is a symmetric matrix. To obtain the stationary points of S, they take constrained variations

$$\sum_k \mathrm{tr}(X_k J X_{k+1}^\top) - \frac{1}{2}\sum_k \mathrm{tr}(\Lambda_k(X_k X_k^\top - I)),$$

(where $\Lambda_k = \Lambda_k^\top$ is a Lagrange multiplier), so that $\delta S = 0$ becomes

$$X_{k+1}J + X_{k-1}J = \Lambda_k X_k,$$

from which, multiplying by $X_k^\top$ on the right and taking into consideration the symmetry of Λ_k,

$$X_{k+1}JX_k^\top + X_{k-1}JX_k^\top = \Lambda_k = \Lambda_k^\top = X_kJX_{k+1}^\top + X_kJX_{k-1}^\top, \tag{1.2}$$

hence, the *discrete analog of the angular momentum in space*,

$$m_k = X_kJX_{k-1}^\top - X_{k-1}JX_k^\top,$$

is conserved.

In the body variables, setting $\omega_k = X_k^\top X_{k-1} \in \mathrm{O}(N)$ and $M_k = X_{k-1}^{-1} m_k X_{k-1} = \omega_k^\top J - J\omega_k \in \mathfrak{so}(N)^*$ (angular momentum with respect to the body), (1.2) becomes

$$\begin{aligned} M_{k+1} &= \omega_k M_k \omega_k^\top, \\ M_k &= \omega_k^\top J - J\omega_k. \end{aligned} \tag{1.3}$$

the **discrete Euler–Arnold** equation.

What is the relation between the discrete Lagrangian (1.1) and the Lagrangian of the continuous RB equations?

Let us consider the continuous RB equations in body coordinates [**2**],

$$M' = [M, \Omega], \qquad M = \Omega J + J\Omega, \tag{1.4}$$

where M is the angular momentum and Ω the angular velocity. M and Ω are skew-symmetric matrices and J is a diagonal matrix with positive entries, the inertia operator. The Lagrangian of the continuous RB equations is the kinetic energy,

$$L = \tfrac{1}{2}\mathrm{tr}(\Omega^\top M) = \tfrac{1}{2}\mathrm{tr}(-\Omega^2 J - \Omega J\Omega) = \mathrm{tr}(\Omega^\top J\Omega), \tag{1.5}$$

where we take into account that $\Omega^\top = -\Omega$ and that the trace is invariant under cyclic permutations. Following Marsden et al. [**9**], discretize $\Omega = g^{-1}\dot{g}$, where $g \in \mathrm{SO}(N)$ is the configuration of the body, using a finite difference approximation of the derivative,

$$\Omega = g^{-1}\dot{g} \approx \frac{1}{h} g_{k+1}^\top (g_{k+1} - g_k), \qquad g_k, g_{k+1} \in \mathrm{SO}(N),$$

which gives

$$L \approx \frac{1}{h^2}\mathrm{tr}(J - g_k^\top g_{k+1} J - J g_{k+1}^\top g_k - g_k^\top g_{k+1} J g_{k+1}^\top g_k).$$

Due to the orthogonality of the g_k's and the cyclicity of the trace, the first and last terms cancel and, moreover, we can write

$$L \approx -\frac{2}{h^2}\mathrm{tr}(g_k J g_{k+1}^\top).$$

Up to a scaling factor, this is precisely the discrete Lagrangian (1.1) whereas X_k is replaced by g_k.

1.2. *The DMV Algorithm for the RB Equations*

Assume that we wish to solve the RB equations (1.4) in the interval $[t_0, t_n]$, where $t_k = t_0 + kh$, h is the step size of integration, and M_0 is the initial condition. Without loss of generality, the matrix J is assumed to be diagonal with positive entries.

The DMV algorithm:

```
1. Set $M_0 := M(t_0)h$.
2. For $k = 0, 1, \ldots, n-1$,
       find the unique $\omega_k$ with the properties below such that $M_k = \omega_k^\top J - J\omega_k$
       set $M_{k+1} = \omega_k M_k \omega_k^\top$
   end
3. Reconstruct $M_n := M_n/h \approx M(t_n)$.
```

In Step 2 one has to solve, time and again, the Moser–Veselov equation

$$M = \omega^\top J - J\omega, \tag{1.6}$$

where M is a skew-symmetric matrix and J is diagonal with entries $J_1, J_2, \ldots, J_N$, for the unknown orthogonal matrix ω. The Moser–Veselov equation (1.6) does not have a unique solution; however, if the set S of eigenvalues λ_i of $W = \omega^\top J$ admits a splitting $S = S_+ \cup S_-$, with

$$\bar{S}_+ = S_+, \qquad \bar{S}_- = S_-, \qquad S_- = -S_+, \qquad S_+ \cap S_- = \emptyset, \tag{1.7}$$

then there exists a unique $\omega = J^{-1}W^\top$ that satisfies (1.6), with spec $W = S_+$ [**12**]. We recall that the eigenvalues λ are all the solutions of the characteristic equation

$$P(\lambda) = \det(\lambda^2 I - \lambda M - J^2) = 0. \tag{1.8}$$

For the 3×3 RB,

$$\begin{aligned}-P(\lambda) &= \lambda^6 - \lambda^4 \left(J_1^2 + J_2^2 + J_3^2 - m_{1,2}^2 - m_{1,3}^2 - m_{2,3}^2\right) \\ &\quad + \lambda^2 \left(J_1^2 J_2^2 + J_1^2 J_3^2 + J_2^2 J_3^2 - m_{1,2}^2 J_3^2 - m_{1,3}^2 J_2^2 - m_{2,3}^2 J_1^2\right) - J_1^2 J_2^2 J_3^2.\end{aligned}$$

Due to the skew-symmetry of M, $P(\lambda) = P(-\lambda)$, which implies that, introducing the auxiliary variable $\mu = \lambda^2$, $P(\mu)$ is a polynomial of degree 3,

$$\begin{aligned}-P(\mu) &= \mu^3 - \mu^2 \left(J_1^2 + J_2^2 + J_3^2 - m_{1,2}^2 - m_{1,3}^2 - m_{2,3}^2\right) \\ &\quad + \mu^1 \left(J_1^2 J_2^2 + J_1^2 J_3^2 + J_2^2 J_3^2 - m_{1,2}^2 J_3^2 - m_{1,3}^2 J_2^2 - m_{2,3}^2 J_1^2\right) - J_1^2 J_2^2 J_3^2 \\ &= \mu^3 + a_2 \mu^2 + a_1 \mu + a_0,\end{aligned}$$

whose roots can be explicitly found (see Abramowitz and Stegun [**1,** p.17]). Once μ_1, μ_2, and μ_3 are determined, we recover the λ_i's by

$$\lambda_{\pm i} = \pm\sqrt{\mu_i}, \qquad i = 1, 2, 3.$$

It must be noted that the coefficients of $P(\lambda)$ are invariants, since the matrix J is constant, $\|\mathbf{m}\|_2^2 = m_{1,2}^2 + m_{1,3}^2 + m_{2,3}^2$ is a Casimir of the system and $H_2 = m_{1,2}^2 J_3^2 + m_{1,3}^2 J_2^2 + m_{2,3}^2 J_1^2$ is a combination of the Hamiltonian H and $\|\mathbf{m}\|_2$ (see also Sections 3 and 5). Therefore the computation of the roots λ_i can be done once and for all (before Step 2), even if the matrix M_k changes at each step. This is also true for a general dimension N of the system and is a consequence of the isospectrality of the flow [**12**].

Having computed the λ_i's, one can then proceed to the computation of the matrix W by solving for eigenvectors corresponding to the λ_i's having positive real part, hence $\omega = J^{-1} W^\top$ is constructed for each iteration k. We refer to this method as the *direct method* and it is essentially described in [**12**].

The matrix ω can also be computed indirectly, via Schur real forms, as proposed by Cardoso and Leite [**3**]. This approach, and our explicit approach, is described in greater detail in Section 9.

2. Modified Vector Field of the DMV

In this section, we consider the discrete Moser–Veselov (DMV) algorithm for the dynamics of the rigid body (RB),

$$M_k = \omega_k^\top J - J\omega_k, \quad M_{k+1} = \omega_k M_k \omega_k^\top, \tag{2.1}$$

and assume that $\omega_k \approx I - h\Omega(t_k)$, $t_k = t_0 + hk$, where $\Omega(t_k)$ is the angular momentum of the continuous RB equation at time $t = t_k$.

Setting

$$M_{k+1} = \Phi_h(M_k) = M_k + h[M_k, \Omega_k] + h^2 d_2 + h^3 d_3 + h^4 d_4 + \cdots, \tag{2.2}$$

we look for a modified vector field (see, for instance, Hairer et al. [**6**]),

$$\tilde{M}' = [\tilde{M}, \tilde{\Omega}] + hf_2(\tilde{M}, \tilde{\Omega}) + h^2 f_3(\tilde{M}, \tilde{\Omega}) + h^3 f_4(\tilde{M}, \tilde{\Omega}) + \cdots, \tag{2.3}$$

such that $\Phi_h(M_k)$ equals the solution $\tilde{M}(t_{k+1})$ at time $t_{k+1} = t_0 + (k+1)h$ of the modified vector field (2.3).

2.1. *Deriving the Map* $\Phi_h(M_k)$

To arrive at formulation (2.2), we write

$$\omega_k = \exp(-h\Omega_0 - h^2\Omega_1 - h^3\Omega_2 - h^4\Omega_3 - h^5\Omega_4 + \cdots), \tag{2.4}$$

where $\Omega_0, \Omega_1, \Omega_2, \ldots,$ are skew-symmetric matrices computed so that

$$\omega_k^\top J - J\omega_k = h(\Omega(t_k)J + J\Omega(t_k)). \tag{2.5}$$

Next, we expand $M_{k+1} = \Phi_h(M_k)$ in (2.1) in terms of powers of h using the series expansion for ω_k. Finally, comparison of the Taylor expansion of the solution of (2.3) and (2.2) allows us to derive the expressions for the f_i's, $i = 3, 4, 5, \ldots$, in (2.3).

We commence by expanding ω_k in powers of h,

$$\begin{aligned}
\omega_k = I &- h\Omega_0 + h^2(-\Omega_1 + \tfrac{1}{2}\Omega^2) \\
&+ h^3(-\Omega_2 + \tfrac{1}{2}(\Omega_0\Omega_1 + \Omega_1\Omega_0) - \tfrac{1}{6}\Omega^3) \\
&+ h^4(-\Omega_3 + \frac{1}{2!}(\Omega_1^2 + \Omega_0\Omega_2 + \Omega_2\Omega_0) \\
&- \frac{1}{3!}\left(\Omega_0^2\Omega_1 + \Omega_0\Omega_1\Omega_0 + \Omega_1\Omega_0^2) + \frac{1}{4!}\Omega_0^4\right) \\
&+ h^5\left(-\Omega_4 + \frac{1}{2!}(\Omega_3\Omega_0 + \Omega_1\Omega_2 + \Omega_2\Omega_1 + \Omega_0\Omega_3)\right. \\
&\qquad - \frac{1}{3!}(\Omega_0^2\Omega_2 + \Omega_0\Omega_1^2 + \Omega_0\Omega_2\Omega_0 + \Omega_1\Omega_0\Omega_1 + \Omega_1^2\Omega_0 + \Omega_2\Omega_0^2) \\
&\qquad \left.+\frac{1}{4!}(\Omega_0^3\Omega_1 + \Omega_0^2\Omega_1\Omega_0 + \Omega_0\Omega_1\Omega_0^2 + \Omega_1\Omega_0^3) - \frac{1}{5!}\Omega_0^5\right) \\
&+ \mathcal{O}(h^6).
\end{aligned} \tag{2.6}$$

Substituting in (2.5), we obtain

$$\begin{aligned}
h(\Omega(t_k)J + J\Omega(t_k)) = h\ &(\Omega_0 J + J\Omega_0) + h^2(\Omega_1 J + J\Omega_1 + \tfrac{1}{2}(\Omega_0^2 J - J\Omega_0^2)) \\
&+ h^3(\Omega_2 J + J\Omega_2 + \tfrac{1}{2}[(\Omega_0\Omega_1 + \Omega_1\Omega_0), J] \\
&+ \tfrac{1}{6}(\Omega_0^3 J + J\Omega_0^3)) + \cdots.
\end{aligned}$$

Table 1. M and its first derivatives with respect to time as functions of Ω (the commutator is the usual matrix commutator).

$M = \Omega J + J\Omega$	
$M' = -[\Omega^2, J]$	$= \Omega' J + J\Omega'$
$M'' = -[\Omega'\Omega + \Omega\Omega', J]$	$= \Omega'' J + J\Omega''$
$M''' = -[\Omega''\Omega + 2\Omega'^2 + \Omega\Omega'', J]$	$= \Omega''' J + J\Omega'''$
$M^{(\text{iv})} = -[\Omega'''\Omega + 3(\Omega''\Omega' + \Omega'\Omega'') + \Omega\Omega''', J]$	$= \Omega^{(\text{iv})} J + J\Omega^{(\text{iv})}$

Comparing the left- and right-hand sides, it is trivially observed that the order h term disappears if $(\Omega_0 - \Omega)J + J(\Omega_0 - \Omega) = 0$ (to simplify notation, we omit the dependence of Ω on t_k).

This is an equation of the type $XJ + JX = 0$. Since we are looking for a skew-symmetric matrix X and, moreover, the matrix J is diagonal, we have $n(n-1)/2$ equations of the type $x_{i,j}(J_j + J_i) = 0$. Because of the positiveness of the J_i's, we can deduce that the equations have the unique solution $x_{i,j} = 0$.

Coming back to the order h term, we can conclude that

$$\Omega_0 = \Omega.$$

In order to annihilate the h^2 term, we require that

$$\Omega_1 J + J\Omega_1 + \tfrac{1}{2}(\Omega_0^2 J - J\Omega_0^2) = 0.$$

Recall that $M = \Omega J + J\Omega$ and hence $M' = \Omega' J + J\Omega'$. On the other hand, $M' = [M, \Omega] = -(\Omega^2 J - J\Omega^2)$. Hence we can write

$$0 = \Omega_1 J + J\Omega_1 - \tfrac{1}{2}M' = \Omega_1 J + J\Omega_1 - \tfrac{1}{2}(\Omega' J + J\Omega').$$

Proceeding as above, we deduce that the identity is satisfied if and only if,

$$\Omega_1 = \tfrac{1}{2}\Omega'. \tag{2.7}$$

By a similar token, we require that

$$\begin{aligned}\Omega_2 J + J\Omega_2 &= -\frac{1}{3!}(\Omega_0^3 J + J\Omega_0^3) - \frac{1}{2!}[(\Omega_0\Omega_1 + \Omega_1\Omega_0), J]\\ &= -\tfrac{1}{6}(\Omega^3 J + J\Omega^3) + \tfrac{1}{4}M''\\ &= -\tfrac{1}{6}(\Omega^3 J + J\Omega^3) + \tfrac{1}{4}(\Omega'' J + J\Omega''),\end{aligned}$$

Table 2. M and its first derivatives with respect to time in terms of a single occurrence of M.

$M = \Omega J + J\Omega$
$M' = [M, \Omega]$
$M'' = [[M, \Omega], \Omega] + [M, \Omega']$
$M''' = [[[M, \Omega], \Omega], \Omega] + [[M, \Omega'], \Omega] + 2[[M, \Omega], \Omega'] + [M, \Omega'']$
$M^{(iv)} = [[[[M, \Omega], \Omega], \Omega], \Omega] + [[[M, \Omega'], \Omega], \Omega] + 2[[[M, \Omega], \Omega'], \Omega]$
$+ 3[[[M, \Omega], \Omega], \Omega'] + [[M, \Omega''], \Omega] + 3[[M, \Omega'], \Omega'] + 3[[M, \Omega], \Omega'']] + [M, \Omega''']$

from which we deduce that

$$\Omega_2 = \tfrac{1}{4}\Omega'' - \tfrac{1}{6}\Omega^3. \tag{2.8}$$

The equation for Ω_3 is

$$\begin{aligned}0 = {} & \Omega_3 J + J\Omega_3 + \frac{1}{2!}[\Omega_1^2 + \Omega_0\Omega_2 + \Omega_2\Omega_0, J] \\ & + \frac{1}{3!}\big((\Omega_0^2\Omega_1 + \Omega_0\Omega_1\Omega_0 + \Omega_1\Omega_0^2)J + J(\Omega_0^2\Omega_1 + \Omega_0\Omega_1\Omega_0 + \Omega_1\Omega_0^2)\big) \\ & + \frac{1}{4!}[\Omega_0^4, J],\end{aligned}$$

from which we deduce that

$$\Omega_3 = \tfrac{1}{8}\Omega''' - \tfrac{1}{12}(\Omega^2\Omega' + \Omega\Omega'\Omega + \Omega'\Omega^2) + \tfrac{1}{8}\mathcal{J}^{-1}[\Omega^4, J] + \tfrac{1}{8}\mathcal{J}^{-1}[\Omega'^2, J],$$

where the operator $\mathcal{J}$ is the linear operator such that $\mathcal{J}S = SJ + JS$. Since J has diagonal entries of the same sign (they are all positive), the operator $\mathcal{J}$ is invertible, as seen before.

Using the relations between M, Ω, and J and their derivatives, after some tedious computations we find

$$[\Omega'^2, J] = -\big((\Omega'\Omega^2 + \Omega^2\Omega')J + J(\Omega'\Omega^2 + \Omega^2\Omega')\big) - [\Omega^4, J],$$

Hence, Ω_3 simply reduces to

$$\Omega_3 = \tfrac{1}{8}\Omega''' - \tfrac{1}{24}(5\Omega^2\Omega' + 2\Omega\Omega'\Omega + 5\Omega'\Omega^2). \tag{2.9}$$

Next, we consider the equation for Ω_4. Setting the coefficient of h^5 to zero, we obtain

$$\begin{aligned}0 = {} & \Omega_4 J + J\Omega_4 + \frac{1}{2!}[\Omega_3\Omega_0 + \Omega_2\Omega_1 + \Omega_1\Omega_2 + \Omega_0\Omega_3, J] \\ & + \frac{1}{3!}\big((\Omega_0^2\Omega_2 + \Omega_0\Omega_2\Omega_0 + \Omega_2\Omega_0^2 + \Omega_0\Omega_1^2 + \Omega_1\Omega_0\Omega_1 + \Omega_1^2\Omega_0)J \\ & \qquad + J(\Omega_0^2\Omega_2 + \Omega_0\Omega_2\Omega_0 + \Omega_2\Omega_0^2 + \Omega_0\Omega_1^2 + \Omega_1\Omega_0\Omega_1 + \Omega_1^2\Omega_0)\big) \\ & + \frac{1}{4!}[\Omega_0^3\Omega_1 + \Omega_0^2\Omega_1\Omega_0 + \Omega_0\Omega_1\Omega_0^2 + \Omega_1\Omega_0^3, J] \\ & + \frac{1}{5!}(\Omega_0^5 J + J\Omega_0^5).\end{aligned}$$

At this point, we repeat the tedious work of substituting in the above expression the known values of $\Omega_0, \Omega_1, \ldots$, and of expressing the whole as $\Omega_4 J + J\Omega_4 = CJ + JC$, from which we will deduce $\Omega_4 = C$.

First, we have

$$\begin{aligned}0=\Omega_4 J + J\Omega_4 &- \tfrac{1}{16}(\Omega^{(\mathrm{iv})}J + J\Omega^{(\mathrm{iv})})\\ &- \tfrac{1}{8}[\Omega''\Omega' + \Omega'\Omega'' + \Omega^2\Omega'\Omega + \Omega\Omega'\Omega^2 + \Omega^3\Omega' + \Omega'\Omega^3, J]\\ &+ \tfrac{1}{24}\big((\Omega^2\Omega'' + \Omega\Omega''\Omega + \Omega''\Omega^2 + \Omega\Omega'^2 + \Omega'\Omega\Omega' + \Omega'^2\Omega)J\\ &\qquad + J(\Omega^2\Omega'' + \Omega\Omega''\Omega + \Omega''\Omega^2 + \Omega\Omega'^2 + \Omega'\Omega\Omega' + \Omega'^2\Omega)\big)\\ &- \tfrac{3}{40}(\Omega^5 J + J\Omega^5).\end{aligned}$$

Second, using the equalities

$$\begin{aligned}&[\Omega''\Omega' + \Omega'\Omega'', J]\\ &\qquad = -[M', \Omega''] - [M'', \Omega']\\ &\qquad = [[\Omega^2, J], \Omega''] + [[\Omega'\Omega + \Omega\Omega', J], \Omega']\\ &\qquad = -((\Omega''\Omega^2 + \Omega^2\Omega'')J + J(\Omega''\Omega^2 + \Omega^2\Omega''))\\ &\qquad\quad +\Omega^2 M'' + M''\Omega^2 + [[\Omega'\Omega + \Omega\Omega', J], \Omega']\\ &\qquad = -((\Omega''\Omega^2 + \Omega^2\Omega'')J + J(\Omega''\Omega^2 + \Omega^2\Omega''))\\ &\qquad\quad -(\Omega^2[\Omega'\Omega + \Omega\Omega', J] + [\Omega'\Omega + \Omega\Omega', J]\Omega^2)\\ &\qquad\quad +[[\Omega'\Omega + \Omega\Omega', J], \Omega'],\end{aligned}$$

$$\begin{aligned}&[\Omega^2\Omega'\Omega + \Omega\Omega'\Omega^2 + \Omega^3\Omega' + \Omega'\Omega^3, J]\\ &\qquad = [\Omega^2(\Omega'\Omega + \Omega\Omega') + (\Omega'\Omega + \Omega\Omega')\Omega^2, J]\\ &\qquad = \Omega^2[\Omega'\Omega + \Omega\Omega', J] + [\Omega'\Omega + \Omega\Omega', J]\Omega^2\\ &\qquad\quad -(M'(\Omega'\Omega + \Omega\Omega') + (\Omega'\Omega + \Omega\Omega')M'),\end{aligned}$$

we deduce that

$$\begin{aligned}&[\Omega''\Omega' + \Omega'\Omega'' + \Omega^2\Omega'\Omega + \Omega\Omega'\Omega^2 + \Omega^3\Omega' + \Omega'\Omega^3, J]\\ &\qquad = -\big((\Omega''\Omega^2 + \Omega^2\Omega'' + \Omega'^2\Omega + 2\Omega'\Omega\Omega' + \Omega\Omega'^2)J\\ &\qquad\qquad + J(\Omega''\Omega^2 + \Omega^2\Omega'' + \Omega'^2\Omega + 2\Omega'\Omega\Omega' + \Omega\Omega'^2)\big),\end{aligned}$$

from which we obtain

$$\begin{aligned}\Omega_4 \;=\; &\tfrac{1}{16}\Omega^{(\mathrm{iv})} - \tfrac{1}{24}(4\Omega''\Omega^2 + \Omega\Omega''\Omega + 4\Omega^2\Omega'' + 4\Omega'^2\Omega + 7\Omega'\Omega\Omega' + 4\Omega\Omega'^2)\\ &+\tfrac{3}{40}\Omega^5.\end{aligned} \tag{2.10}$$

Table 3. The functions Ω_i.

$\Omega_0 = \Omega$
$\Omega_1 = \frac{1}{2}\Omega'$
$\Omega_2 = \frac{1}{4}\Omega'' - \frac{1}{6}\Omega^3$
$\Omega_3 = \frac{1}{8}\Omega''' - \frac{1}{24}(5\Omega^2\Omega' + 2\Omega\Omega'\Omega + 5\Omega'\Omega^2)$

In general, the algorithm to derive Ω_i, $i = 1, 2, \ldots$, is:

(1) Find the coefficient of h^{i+1} in (2.5) and set it equal to zero. This will give an equation of the type $\Omega_i J + J\Omega_i = C_i J + JC_i + [D_i, J]$. Note that the terms $C_i J + JC_i$ have an odd occurrence of the Ω_j,s, while the terms of type $[D_i, J]$ have an even occurrence of the Ω_j,s (this statement can be rigorously proved by writing $\omega = \exp(-f(h))$, where $f(h) = \sum_{j=0}^{\infty} h^{j+1}\Omega_j$, and then using the relations between exp and sinh, cosh).
(2) Use the derivatives of M and Ω to express the term $[D_i, J]$ as $\tilde{C}_i J + J\tilde{C}_i$.
(3) Deduce $\Omega_i = C_i + \tilde{C}_i$.

Once the Ω_i,s are known, substituting back in (2.1) and using the well-known identity

$$\exp(X)Y\exp(-X) = \exp_{\mathrm{ad}_X} Y = \sum_{k=0}^{\infty} \frac{1}{k!}\mathrm{ad}_X^k(Y),$$

where $\mathrm{ad}_X(Y) = [X, Y]$ and, recursively, $\mathrm{ad}_X^k(Y) = [X, \mathrm{ad}_X^{k-1}(Y)]$ (see, for instance, Iserles et al. [**7**]), we obtain the Taylor expansion for the DMV algorithm,

$$\begin{aligned} M_{k+1} = {} & M_k - h[\Omega_0, M_k] + h^2(-[\Omega_1, M_k] + \frac{1}{2!}[\Omega_0, [\Omega_0, M_k]]) \\ & + h^3\Big(-[\Omega_2, M_k] + \frac{1}{2!}([\Omega_0, [\Omega_1, M_k]] + [\Omega_1, [\Omega_0, M_k]]) \\ & \quad -\frac{1}{3!}[\Omega_0, [\Omega_0, [\Omega_0, M_k]]]\Big) \\ & + \mathcal{O}(h^4), \end{aligned}$$

hence,

$$\begin{aligned} M_{k+1} = {} & M_k - h[\Omega, M_k] + \tfrac{1}{2}h^2(-[\Omega', M_k] + [\Omega, [\Omega, M_k]]) \\ & + h^3(-\tfrac{1}{4}[\Omega'', M_k] + \tfrac{1}{4}[\Omega, [\Omega', M_k]] + \tfrac{1}{4}[\Omega', [\Omega, M_k]] \\ & - \tfrac{1}{6}[\Omega, [\Omega, [\Omega, M_k]]] + \tfrac{1}{6}[\Omega^3, M]) \\ & + \mathcal{O}(h^4). \end{aligned}$$

By the skew-symmetry of the commutator and by comparison with (2.2), we obtain the explicit expressions for the functions $d_2, d_3, \ldots$,

$$d_2 = \tfrac{1}{2}([M, \Omega'] + [[M, \Omega], \Omega]),$$

$$\begin{aligned} d_3 &= \tfrac{1}{4}[M, \Omega''] + \tfrac{1}{4}[[M, \Omega'], \Omega] + \tfrac{1}{4}[[M, \Omega], \Omega'] \\ &\quad + \tfrac{1}{6}[[[M, \Omega], \Omega], \Omega] - \tfrac{1}{6}[M, \Omega^3], \\ d_4 &= \ldots, \end{aligned} \tag{2.11}$$

where $M \equiv M_k \approx M(t_k)$. The general expression for d_i as a function of $\Omega_0, \ldots, \Omega_{i-1}$ is

$$d_i = \sum_{j=1}^{i} \frac{(-1)^j}{j!} \sum_{k_1+k_2+\cdots+k_j=i-j} \mathrm{ad}_{\Omega_{k_1}} \mathrm{ad}_{\Omega_{k_2}} \cdots \mathrm{ad}_{\Omega_{k_j}} M, \qquad k_1, \ldots k_j \in \{0, 1, \ldots, i-1\}. \tag{2.12}$$

2.2. *Taylor Expansion of the Solution of the Modified Equation*

The derivation of the Taylor expansion of the solution of the modified vector field can be done according to Hairer et al.[**6**]. Consider

$$\frac{d}{dt}\tilde{M} = f(\tilde{M}) + hf_2(\tilde{M}) + h^2 f_3(\tilde{M}) + \cdots,$$

where $f(M) = [M, \Omega] = [M, \mathcal{J}^{-1}M]$ is the original vector field of the RB equations where, as above, $\mathcal{J}$ is the linear operator such that $\mathcal{J}\Omega = \Omega J + J\Omega = M$. We assume that $\tilde{M}(t) = M(t)$, and expand $\tilde{M}(t+h)$, the solution of the above differential equation, in a Taylor series at $h = 0$. Collecting the corresponding powers of h,

$$\begin{aligned} \tilde{M}(t+h) &= M(t) + hf(M) + h^2 \left(f_2(M) + \frac{1}{2!} f' f(M) \right) \\ &\quad + h^3 \left(f_3(M) + \frac{1}{2!}(f' f_2(M) + f_2' f(M)) + \frac{1}{3!}(f''(f, f)(M) \right. \\ &\quad \left. + f' f' f(M)) \right) + \cdots, \end{aligned}$$

where f' is considered as a linear operator, f'' as a bilinear operator, and so on. In our case,

$$\begin{aligned} f'(z)(M) &= [z, \mathcal{J}^{-1}M] + [M, \mathcal{J}^{-1}z] \\ &= [z, \Omega] + [M, \mathcal{J}^{-1}z], \\ f''(z_1, z_2)(M) &= 2[z_1, \mathcal{J}^{-1}z_2], \end{aligned}$$

and, since f is quadratic, f''' and all the other higher derivatives equal zero.

Taking into account the equalities displayed in Tables 1 and 2, we have

$$f' f(M) = [[M, \Omega], \Omega] + [M, [\mathcal{J}^{-1}[M, \Omega]]], = [[M, \Omega], \Omega] + [M, \Omega'],$$

$$\begin{aligned} f''(f,f)(M) &= 2[[M,\Omega],\mathcal{J}^{-1}[M,\Omega]] = 2[[M,\Omega],\Omega'],\\ f'f'f(M) &= [[[M,\Omega],\Omega],\Omega] + [M,\mathcal{J}^{-1}[[M,\Omega],\Omega]] + [[M,\Omega'],\Omega]\\ &\quad + [M,\mathcal{J}^{-1}[M,\Omega']]\\ &= [[[M,\Omega],\Omega],\Omega] + [M,\Omega''] + [[M,\Omega'],\Omega], \end{aligned}$$

from which we immediately deduce that

$$f_2 = d_2 - \frac{1}{2!} f'f(M) = 0,$$

as we would expect from a second-order method and, moreover,

$$f_3 = d_3 - \frac{1}{3!}(f''(f,f)(M) + f'f'f(M)) = \tfrac{1}{12}[M, \Omega'' - [\Omega,\Omega'] - 2\Omega^3].$$

For our computations, it will be convenient to write f_3 as

$$f_3 = \tfrac{1}{12}\left([M,\Omega''] - [[M,\Omega],\Omega'] + [[M,\Omega'],\Omega] - 2[M,\Omega^3]\right). \tag{2.13}$$

Note that f_3 is a quartic polynomial in M. This confirms what has already been proved in Deift et al. [**4**] by a completely different argument: Deift et al. used the theory of loop groups and rank 2 extension, and their expression for $K(M) = f_3(M)$ is given as a limit of a double integral. In the sequel, we shall see that, due to its simplicity, our formula (2.13) is much more amenable and easy to use in numerical computations, to dramatically improve the performance of the DMV algorithm.

At this point it is important to stress an important difference between the expressions for the modified vector field of Hairer et al. [**6**] and ours. While the vector field discussed in [**6**] is in $\mathbb{R}^n$, hence the f'' is a symmetric quadratic operator, this is not the case for our vector field which is on matrices, thus in general $f''(f'f, f) \neq f''(f, f'f)$. This noncommutative case is discussed with more generality in Munthe-Kaas and Krogstad [**13**]. However, we observe that *all* the terms containing combinations of f'', f', and f correspond simply to higher derivatives of f. The mixed terms are treated instead specifically. Therefore, we have

$$\begin{aligned} f_4 &= d_4 - \frac{1}{4!}M^{(iv)} - \frac{1}{2!}(f'f_3 + f_3'f),\\ f_5 &= d_5 - \frac{1}{5!}M^{(v)} - \frac{1}{2!}\left(f'f_4 + f_4'f + \frac{1}{2!}\frac{d}{dt}(f_3'f + f'f_3)\right), \end{aligned} \tag{2.14}$$

whereby all the vector fields are computed at M.

By direct computation,

$$\begin{aligned}
f' f_3 &= \tfrac{1}{12}\big([[M, \Omega'' - [\Omega, \Omega'] - 2\Omega^3], \Omega] + [M, \mathcal{J}^{-1}([M, \Omega''] - [[M, \Omega], \Omega'] \\
&\qquad + [[M, \Omega'], \Omega] - 2[M, \Omega^3])]\big) \\
&= \tfrac{1}{12}\big([[M, \Omega'' - [\Omega, \Omega'] - 2\Omega^3], \Omega] + [M, \Omega''' - 3(\Omega^2\Omega' + \Omega'\Omega^2)]\big),
\end{aligned}$$

$$\begin{aligned}
f_3' f &= \tfrac{1}{12}(-2([[M, \Omega], \Omega^3] + [M, \Omega'\Omega^2 + \Omega\Omega'\Omega + \Omega^2\Omega'])\\
&\quad - ([[M, \Omega], [\Omega, \Omega']] + [M, [\Omega, \Omega'']])\\
&\quad + [[M, \Omega], \Omega''] + [M, \mathcal{J}^{-1}([[[M, \Omega], \Omega], \Omega])]\\
&\quad + [M, \mathcal{J}^{-1}([[M, \Omega'], \Omega])]\\
&\quad + [M, \mathcal{J}^{-1}([[M, \Omega], \Omega'])] + [M, \mathcal{J}^{-1}([[M, \Omega], \Omega'])]\\
&\quad + [M, \mathcal{J}^{-1}([M, \mathcal{J}^{-1}[[M, \Omega], \Omega]])]\\
&\quad + [M, \mathcal{J}^{-1}([M, \mathcal{J}^{-1}[M, \Omega']])])\\
&= \tfrac{1}{12}(-2([[M, \Omega], \Omega^3] + [M, \Omega'\Omega^2 + \Omega\Omega'\Omega + \Omega^2\Omega'])\\
&\quad - ([[M, \Omega], [\Omega, \Omega']] + [M, [\Omega, \Omega'']])\\
&\quad + [[M, \Omega], \Omega''] + [M, \Omega''']).
\end{aligned}$$

To reduce the second term in $f' f_3$ we have used the equation for M''' in Table 2, hence the identities $[M, \Omega] = M' = \Omega' J + J\Omega' = -[\Omega^2, J]$.

We have

$$\begin{aligned}
f' f_3 + f_3' f \;=\; \tfrac{1}{12}\big(&2[M, \Omega'''] + 2[[M, \Omega''], \Omega] - 4[[M, \Omega], \Omega^3]\\
&-[M, 5\Omega^2\Omega' + 2\Omega\Omega'\Omega + 5\Omega'\Omega^2]\\
&+ [[[M, \Omega'], \Omega], \Omega] - [[[M, \Omega], \Omega], \Omega']\big).
\end{aligned}$$

By a long and tedious computation, it is verified that, indeed,

$$f_4 = d_4 - \frac{1}{4!}M^{(\mathrm{iv})} - \frac{1}{2!}(f_3' f + f' f_3) = 0.$$

Similarly,

$$\begin{aligned}
f_5 = {} & \tfrac{1}{80}\ [M, \Omega^{(\mathrm{iv})}] - \tfrac{1}{80}[M, [\Omega, \Omega''']] + \tfrac{3}{40}[M, \Omega^5 - \Omega'\Omega\Omega']\\
&+ \tfrac{1}{80}[M, [\Omega', \Omega'']] - \tfrac{1}{40}[M, \Omega\Omega''\Omega] - \tfrac{1}{20}[M, \Omega^2\Omega'' + \Omega''\Omega^2]\\
&+ \tfrac{1}{20}[M, [\Omega^3, \Omega']] - \tfrac{1}{40}[M, \Omega'^2\Omega + \Omega\Omega'^2 + \Omega[\Omega, \Omega']\Omega].
\end{aligned} \tag{2.15}$$

The fact that $f_2 = f_4 = 0$ is not unexpected. Indeed, it can be proved that all the functions $f_{2i} = 0$ for $i = 1, 2, \ldots$, as shown in the result below.

Theorem 1. *The discrete Moser–Veselov algorithm* (2.1) *is time reversible*, *hence* $f_{2i} = 0$ *for* $i = 1, 2, \ldots$.

Proof. It is sufficient to show that if $M_{k+1} = \Phi_h(M_k)$ and $M_{k+2} = \Phi_{-h}(M_{k+1})$, then $M_{k+2} = M_k$.

By construction,

$$\begin{aligned} M_k &= \omega_k^\top J - J\omega_k, \\ M_{k+1} &= \Phi_h(M_k) = -\omega_k J + J\omega_k^\top. \end{aligned}$$

Recall that, for a positive time step, the orthogonal solution ω of the matrix equation (1.6),

$$M = \omega^\top J - J\omega,$$

is computed as the unique matrix such that $W = \omega^\top J$ has eigenvalues with positive real part (see Section 1.2).

Next, we compute $M_{k+2} = \Phi_{-h}(M_{k+1})$. For a negative step size, either:

(i) we compute the unique solution corresponding to eigenvalues λ solving (1.8) having negative real parts; or
(ii) we take the solution with positive real parts and exchange ω by $\omega^\top$; both procedures are equivalent.

Considering the first case, we have

$$M_{k+1} = \omega_{k+1}^\top J - J\omega_{k+1},$$

which, by comparison with $M_{k+1} = -\omega_k J + J\omega_k^\top$, immediately gives

$$\omega_{k+1}^\top = -\omega_k$$

for sufficiently small step size, because of the uniqueness of the orthogonal solution with the required eigenvalue properties and the $\mathcal{O}(h^2)$ consistency. Hence,

$$\begin{aligned} M_{k+2} &= \omega_{k+1} M_{k+1} \omega_{k+1}^\top = \omega_k^\top(-\omega_k J + J\omega_k^\top)\omega_k \\ &= \omega_k^\top J - J\omega_k = M_k, \end{aligned}$$

which implies that $\Phi_{-h} = \Phi_h^{-1}$. This condition is equivalent to $f_{2i} = 0$, for $i = 1, 2, \ldots$, see [**6**]. □

Moser and Veselov [**12**] prove that, for general N, the algorithm (2.1) is a time reparametrization of the flow of the original vector field of the RB: since the mapping preserves the underlying Poisson structure and all the integrals $F_i = c_i$ of the system, it commutes with all commuting Hamiltonian flows generated by the F_i,s, $M' = \{M, \nabla F_i\}$. The nonsingular compact level sets $T_c = \bigcap_i (F_i = c_i)$ consists of a finite union of tori and on each torus the DMV mapping is a shift along the trajectory depending on an integral quantity H_2 which will be introduced later.

This, in combination with Theorem 1, means that (2.1) solves the modified equation

$$M' = (1 + h^2\tau_3 + h^4\tau_5 + \cdots + h^{2i}\tau_{2i+1} + \cdots)[M, \Omega],$$

where h is the step size of integration and the τ_{2i+1}, $i = 1, 2, \ldots$, are constants that depend only on the function H_2, the matrix J, and the Casimirs of the system.

In what follows, we use our results on the modified vector field of the DMV algorithm to find an explicit expression for some of the τ_i,s in the 3×3 case.

3. Modified Vector Fields and Time Reparametrization of the Discrete Moser–Veselov Algorithm for the 3×3 Rigid Body

The previous results were general and independent of the dimensions of the RB under consideration. In what follows, unless otherwise specified, we focus on the 3×3 case. In this case, the equations of motion for the components $m_3 = -m_{1,2}$, $m_1 = -m_{2,3}$, $m_2 = m_{1,3}$ are

$$\begin{aligned} m_1' &= -\frac{(J_2 - J_3)m_2m_3}{(J_1 + J_2)(J_1 + J_3)}, \\ m_2' &= \frac{(J_1 - J_3)m_1m_3}{(J_1 + J_2)(J_2 + J_3)}, \\ m_3' &= -\frac{(J_1 - J_2)m_1m_2}{(J_2 + J_3)(J_1 + J_3)}. \end{aligned}$$

The correspondence between the matrix and vector notation is well known and is given by the *hatmap* function,

$$\hat{\mathbf{a}} = A = \begin{pmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{pmatrix}, \qquad \mathbf{a} = (a_1, a_2, a_3)^\top,$$

while commutation $[A, B]$ corresponds to computing the cross product $\mathbf{a} \times \mathbf{b}$ of the corresponding vectors ($A = \hat{\mathbf{a}}$, $B = \hat{\mathbf{b}}$). The Hamiltonian is

$$H = \frac{m_1^2}{J_2 + J_3} + \frac{m_2^2}{J_1 + J_3} + \frac{m_3^2}{J_1 + J_2}. \tag{3.1}$$

In Figure 3.1 we display the components m_1, m_2, m_3 of the solution of the DMV algorithm (dotted lines) compared to the exact solutions of the RB equations (1.4) (solid lines) and the exact solutions of the modified vector fields $f + h^2 f_3$ (dashed

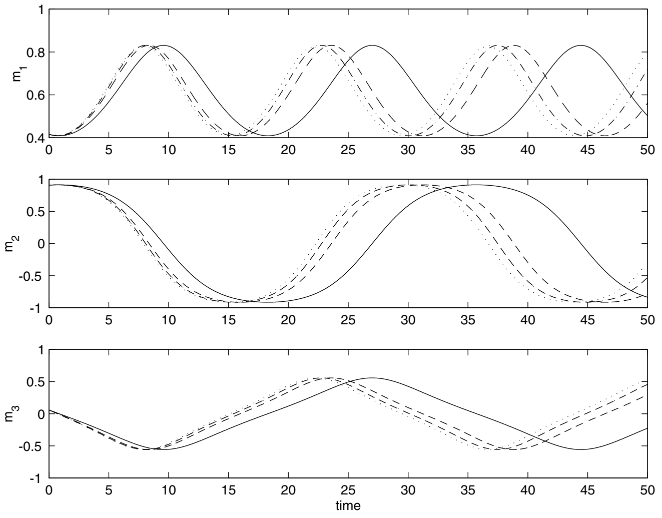

Fig. 3.1. The DMV solution of the RB equations (dotted lines), the exact solution (solid lines) and the trajectories corresponding to the modified vector fields $f + h^2 f_3$ (dashed lines) and $f + h^2 f_3 + h^4 f_5$ (dash–dotted lines) in the interval [0, 50] with $h = \frac{8}{10}$.

lines) and $f + h^2 f_3 + h^4 f_5$ (dash–dotted lines) in the interval [0, 50]. The initial condition $\mathbf{m}_0$ and the matrix J are reported in Section 7. The step size of integration h for the DMV is chosen as $h = \frac{8}{10}$. The "exact solutions" are computed using the MATLAB routine `ode45` with absolute and relative errors set to machine precision. The numerics confirm that the order of approximations are 2, for the RB; 4 for the modified vector field $f + h^2 f_3$ and 6 for $f + h^2 f_3 + h^4 f_5$. Figures 3.1 and 3.2 indicate that DMV approximation, though preserving very well the "qualitative" features of the solution of (1.4), oscillates too rapidly compared to exact solution.

In what follows, we focus on the 3 × 3 case and we set

$$H_2 = m_1^2 J_1^2 + m_2^2 J_2^2 + m_3^2 J_3^2, \tag{3.2}$$

and

$$\Delta = (J_1 + J_2)(J_1 + J_3)(J_2 + J_3). \tag{3.3}$$

Proposition 3.1. *The function* f_3 *is of the form*

$$f_3 = \tau_3[M, \Omega],$$

where

$$\tau_3 = \frac{1}{6\Delta^2}\big((3\det(J)\,\mathrm{tr}(J) + J_1^2 J_2^2 + J_1^2 J_3^2 + J_2^2 J_3^2)\|\mathbf{m}\|_2^2 + (3(J_1 J_2 + J_1 J_3 + J_2 J_3) + \mathrm{tr}(J^2))H_2\big)$$

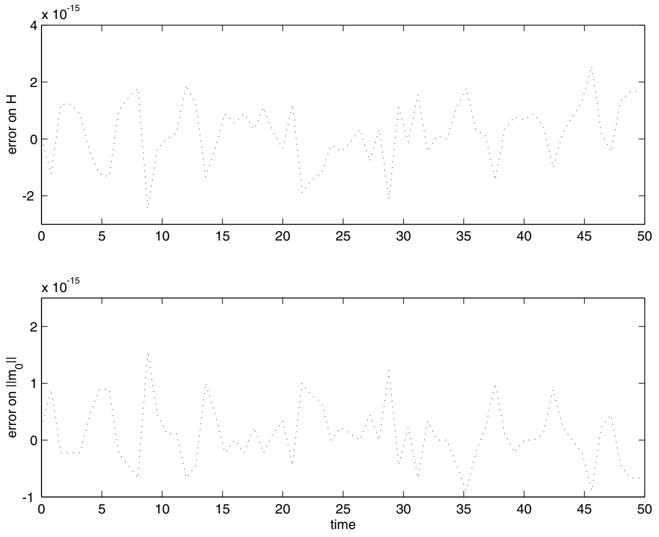

Fig. 3.2. Error on the Hamiltonian (top) and Casimir, 2 norm of the solution (bottom), for the DMV with $h = \frac{8}{10}$.

is a constant which depends the integral function H_2, *on the Casimir* $\|\mathbf{m}\|_2 = \|\mathbf{m}(t_0)\|_2$, *and on the entries of the matrix* J.

Proof. We start recalling that H_2 in (3.2) is an integral of the system,[1] though it is not independent of the Hamiltonian H. Reducing the right-hand side of (3.1) to a common denominator and eliminating it, we obtain

$$(J_1+J_2)(J_1+J_3)(J_2+J_3)H = (J_1J_2+J_1J_3+J_2J_3)(m_{1,2}^2+m_{1,3}^2+m_{2,3}^2)+H_2.$$

Note that $m_{1,2}^2+m_{1,3}^2+m_{2,3}^2 = \|\mathbf{m}\|_2^2$ is constant, therefore H_2 cannot depend on time and must be a constant too.

Second, let us denote by $\mathbf{f}_3 = (f_{3;1}, f_{3;2}, f_{3;3})^\top$ the three-vector such that $\hat{\mathbf{f}}_3 = f_3$. We show that the first component of this vector is of the form

$$f_{3;1} = \tau_3 m_1'.$$

To this scope, we compute $f_{3;1}$ explicitly using (2.13). Collecting terms carefully, we have

$$f_{3;1} = \frac{-(J_2-J_3)m_2m_3}{6(J_1+J_2)^3(J_1+J_3)^3(J_2+J_3)^2}\big((3J_1^2(J_1J_2+J_1J_3+J_2J_3) \\ + J_1^2(J_1^2+J_2^2+J_3^2)m_1^2\big)$$

[1] Moser and Veselov call this quantity H, while we reserve H for the Hamiltonian function (3.1).

$$
\begin{aligned}
&+ (3J_2^2(J_1J_2 + J_1J_3 + J_2J_3) + J_2^2(J_1^2 + J_2^2 + J_3^2))m_2^2 \\
&+ (3J_3^2(J_1J_2 + J_1J_3 + J_2J_3) + J_3^2(J_1^2 + J_2^2 + J_3^2))m_3^2 \\
&+ (3J_1J_2J_3(J_1 + J_2 + J_3) + J_1^2J_2^2 + J_1^2J_3^2 + J_2^2J_3^2)(m_1^2 + m_2^2 + m_3^2)\big),
\end{aligned}
$$

from which, using the above expression for H_2, the identity $m_1^2 + m_2^2 + m_3^2 = \|\mathbf{m}\|_2^2 = \|\mathbf{m}(t_0)\|_2^2$, and the equation for m_1', the result follows.

The procedure is analogous for the other components of the vector $\mathbf{f}_3$. □

Before proceeding, we introduce the following simplifying notation: for $i, j = 1, 2, \ldots$, let

$$
\begin{aligned}
C_{J,i,j} &= J_1^iJ_2^j + J_1^iJ_3^j + J_2^iJ_3^j, \\
C_{J,i} &= C_{J,i,i}, \\
C_J &= C_{J,1}.
\end{aligned}
$$

Proposition 3.2. *For the* 3×3 *RB, the function* f_5 *is of the form*

$$f_5 = \tau_5[M, \Omega],$$

where

$$
\begin{aligned}
\tau_5 = \frac{1}{40\Delta^4}\big(&(3\,\mathrm{tr}(J^4) + 27C_{J,2} + 15\,\mathrm{tr}(J^2)C_J + 45\det(J)\,\mathrm{tr}(J))H_2^2 \\
&+ (10C_{J,3} + 50\det(J)\,\mathrm{tr}(J)C_J + 10\det(J)\,\mathrm{tr}(J)\,\mathrm{tr}(J^2) \\
&+ 2C_{J,2}\,\mathrm{tr}(J^2) - 28\det(J^2))\|\mathbf{m}\|_2^2H_2 \\
&+ (60\det(J^2)C_J + 3C_{J,4} + 27\det(J^2)\,\mathrm{tr}(J^2) \\
&+ 15\det(J)(C_{J,2,3} + C_{J,3,2}))\|\mathbf{m}\|_2^4\big)
\end{aligned}
$$

is a positive constant which depends on the constant function H_2*, the Casimir* $\|\mathbf{m}\|_2$*, and on the entries of the matrix* J.

Proof. By direct computation, as in the case of τ_3. The positiveness of τ_5 is not immediate, because of the presence of the term $-28\|\mathbf{m}\|_2^2\det(J^2)H_2$, but can also be verified by expanding the above expression (τ_5 is ultimately a combination of positive terms with positive coefficients). □

4. Constructing Higher-Order Integrable Approximations

In this section, we wish to use the results derived in the previous section to obtain higher-order integrable approximations to the RB equations. To preserve the properties of the integrable discretizations, we propose simply to scale the initial condition (since, for the continuous equations, this amounts to a time reparametrization). The scaling must, of course, depend on the step size h of integration.

Our ansatz is

$$\frac{h(\Omega(t_k)J + J\Omega(t_k))}{1 + \tilde{\tau}_3 h^2 + \tilde{\tau}_5 h^4 + \cdots}. \tag{4.1}$$

We perform again the backward error analysis with this new initial condition. We now set $\tilde{\omega} = \exp(-h\tilde{\Omega}_0 - h^2\tilde{\Omega}_1 + \cdots)$ and solve for the $\tilde{\Omega}_i$,s as the skew-symmetric matrices that obey

$$h(1 - \tilde{\tau}_3 h^2 + (\tilde{\tau}_3^2 - \tilde{\tau}_5)h^4 + \cdots)(\Omega J + J\Omega) = \tilde{\omega}^\top J - J\tilde{\omega}. \tag{4.2}$$

It is immediately verified that

$$\tilde{\Omega}_0 = \Omega_0 = \Omega, \qquad \tilde{\Omega}_1 = \Omega_1 = \tfrac{1}{2}\Omega',$$

hence it is clear that $\tilde{f}_1 = f_1 = 0$ and $\tilde{f}_2 = f_2 = 0$ for the new modified vector field. The first $\tilde{\Omega}_i$ to differ is $\tilde{\Omega}_2$. Equating the h^3 term and taking into account (2.8), we have

$$\tilde{\Omega}_2 J + J\tilde{\Omega}_2 = -\tilde{\tau}_3(\Omega J + J\Omega) - \frac{1}{3!}(\Omega_0^3 J + J\Omega_0^3) - \frac{1}{2!}[(\Omega_0\Omega_1 + \Omega_1\Omega_0), J],$$

hence

$$\tilde{\Omega}_2 = -\tilde{\tau}_3\Omega + \Omega_2.$$

Thus,

$$\tilde{d}_3 = -[\tilde{\Omega}_2, M] + \frac{1}{2!}[\tilde{\Omega}_1, [\tilde{\Omega}_1, M]] = \tilde{\tau}_3[\Omega, M] + d_3 = -\tilde{\tau}_3[M, \Omega] + d_3.$$

Moreover,

$$\begin{aligned} \tilde{f}_3 &= \tilde{d}_3 - \frac{1}{3!}M''' = -\tilde{\tau}_3[M, \Omega] + d_3 - \frac{1}{3!}M''' \\ &= -\tilde{\tau}_3[M, \Omega] + f_3 = (-\tilde{\tau}_3 + \tau_3)[M, \Omega], \end{aligned}$$

hence, in order to have an order 4 scheme, we must set $\tilde{f}_3 = 0$ which corresponds to the choice

$$\tilde{\tau}_3 = \tau_3.$$

One could be tempted to extrapolate that $\tilde{\tau}_5 = \tau_5$ does the job of rendering the methods of order 6, but, unfortunately, this is not the case, since there are more complicated nonlinear effects arising from the scaling of the initial condition. Therefore we need to proceed with our backward error analysis.

The equation for $\tilde{\Omega}_3$ is identical to (2.9), whereby Ω_2 is replaced by $\tilde{\Omega}_2$, hence,

$$\tilde{\Omega}_3 = \Omega_3 - \tilde{\tau}_3\Omega'.$$

For what $\tilde{\Omega}_4$ is concerned, we also need to take into account the h^5 term on the left-hand side of (4.2), hence

$$\begin{aligned}(\tilde{\tau}_3^2 - \tilde{\tau}_5)(\Omega J + J\Omega) = {} & \tilde{\Omega}_4 J + J\tilde{\Omega}_4 + \frac{1}{2!}[\tilde{\Omega}_3\Omega_0 + \tilde{\Omega}_2\Omega_1 + \Omega_1\tilde{\Omega}_2 + \Omega_0\tilde{\Omega}_3, J] \\ & + \frac{1}{3!}\Big((\Omega_0^2\tilde{\Omega}_2 + \Omega_0\tilde{\Omega}_2\Omega_0 + \tilde{\Omega}_2\Omega_0^2 + \Omega_0\Omega_1^2 + \Omega_1\Omega_0\Omega_1 \\ & \qquad + \Omega_1^2\Omega_0)J + J(\Omega_0^2\tilde{\Omega}_2 + \Omega_0\tilde{\Omega}_2\Omega_0 + \tilde{\Omega}_2\Omega_0^2 \\ & \qquad + \Omega_0\Omega_1^2 + \Omega_1\Omega_0\Omega_1 + \Omega_1^2\Omega_0)\Big) \\ & + \frac{1}{4!}[\Omega_0^3\Omega_1 + \Omega_0^2\Omega_1\Omega_0 + \Omega_0\Omega_1\Omega_0^2 + \Omega_1\Omega_0^3, J] \\ & + \frac{1}{5!}(\Omega_0^5 J + J\Omega_0^5),\end{aligned}$$

from which we deduce

$$\tilde{\Omega}_4 = (\tilde{\tau}_3^2 - \tilde{\tau}_5)\Omega + \tfrac{1}{2}\tilde{\tau}_3\Omega^3 - \tfrac{3}{4}\tilde{\tau}_3\Omega'' + \Omega_4.$$

We now compute $\tilde{d}_5$ according to (2.12),

$$\begin{aligned}\tilde{d}_5 = & - (\tilde{\tau}_3^2 - \tilde{\tau}_5)[\Omega, M] + \tfrac{3}{4}\tilde{\tau}_3[\Omega'', M] - \tfrac{1}{2}\tilde{\tau}_3[\Omega^3, M] \\ & - \tfrac{3}{4}\tilde{\tau}_3([\Omega, [\Omega', M]] + [\Omega', [\Omega, M]]) + \tfrac{1}{2}\tilde{\tau}_3[\Omega, [\Omega, [\Omega, M]]] + d_5.\end{aligned}$$

Because of the nature of the scaling, it is evident that $\tilde{f}_4 = f_4 = 0$, therefore

$$\tilde{f}_5 = \tilde{d}_5 - \frac{1}{5!}M^{(\mathrm{v})},$$

having chosen $\tilde{\tau}_3$ so that $\tilde{f}_3 = 0$. We have

$$\begin{aligned}\tilde{f}_5 = & - (\tilde{\tau}_3^2 - \tilde{\tau}_5)[\Omega, M] + \tfrac{3}{4}\tilde{\tau}_3[\Omega'', M] - \tfrac{1}{2}\tilde{\tau}_3[\Omega^3, M] - \tfrac{3}{4}\tilde{\tau}_3([\Omega, [\Omega', M]] \\ & + [\Omega', [\Omega, M]]) + \tfrac{1}{2}\tilde{\tau}_3[\Omega, [\Omega, [\Omega, M]]] + d_5 - \tfrac{1}{5!}M^{(\mathrm{v})} \\ = & - (\tilde{\tau}_3^2 - \tilde{\tau}_5)[\Omega, M] + \tfrac{3}{4}\tilde{\tau}_3[\Omega'', M] - \tfrac{1}{2}\tilde{\tau}_3[\Omega^3, M] - \tfrac{3}{4}\tilde{\tau}_3([\Omega, [\Omega', M]] \\ & + [\Omega', [\Omega, M]]) + \tfrac{1}{2}\tilde{\tau}_3[\Omega, [\Omega, [\Omega, M]]] \\ & + f_5 + \frac{1}{5!}M^{(\mathrm{v})} + \tfrac{1}{4}\tfrac{d}{dt}(f' f_3 + f_3' f) - \tfrac{1}{5!}M^{(\mathrm{v})} \\ = & \ (\tilde{\tau}_3^2 - \tilde{\tau}_5 + \tau_5)[M, \Omega] + \tfrac{3}{4}\tilde{\tau}_3[\Omega'', M] - \tfrac{1}{2}\tilde{\tau}_3[\Omega^3, M] - \tfrac{3}{4}\tilde{\tau}_3([\Omega, [\Omega', M]] \\ & + [\Omega', [\Omega, M]]) + \tfrac{1}{2}\tilde{\tau}_3[\Omega, [\Omega, [\Omega, M]]] + \tfrac{1}{4}\tfrac{d}{dt}(f' f_3 + f_3' f).\end{aligned}$$

Since $f_3 = \tau_3[M, \Omega]$, we have $f' f_3 + f_3' f = 2\tau_3 f' f$, hence $d/dt(f' f_3 + f_3' f) = 2\tau_3(f''(f, f) + f' f' f) = 2\tau_3 M'''$. Moreover, recall that $f_5 = \tau_5[M, \Omega]$. It follows that

$$\begin{aligned}\tilde{f}_5 &= (\tilde{\tau}_3^2 - \tilde{\tau}_5 + \tau_5)[M, \Omega] - \tfrac{3}{4}\tilde{\tau}_3[M, \Omega''] + \tfrac{1}{2}\tilde{\tau}_3[M, \Omega^3] - \tfrac{3}{4}\tilde{\tau}_3([[M, \Omega'], \Omega] \\ &\quad + [[M, \Omega], \Omega']) - \tfrac{1}{2}\tilde{\tau}_3[[[M, \Omega], \Omega], \Omega] \\ &\quad + \tfrac{1}{2}\tau_3([[[M, \Omega], \Omega], \Omega] + [[M, \Omega'], \Omega] + 2[[M, \Omega], \Omega'] + [M, \Omega'']) \\ &= (\tilde{\tau}_3^2 - \tilde{\tau}_5 + \tau_5)[M, \Omega] - \tfrac{1}{4}\tilde{\tau}_3([M, \Omega''] \\ &\quad + [[M, \Omega'], \Omega] - [[M, \Omega], \Omega'] - 2[M, \Omega^3]).\end{aligned}$$

We recognize that the last term is proportional to $f_3 = \tau_3 f$. Hence,

$$\tilde{f}_5 = (\tilde{\tau}_3^2 - \tilde{\tau}_5 + \tau_5 - 3\tilde{\tau}_3^2)[M, \Omega] = (-2\tilde{\tau}_3^2 - \tilde{\tau}_5 + \tau_5)[M, \Omega],$$

and setting $\tilde{f}_5 = 0$ we find

$$\tilde{\tau}_5 = \tau_5 - 2\tau_3^2.$$

This value of $\tilde{\tau}_5$ gives indeed a method of order 6.

The new proposed algorithms of order 4 and 6 are described below.

The DMV4 algorithm:

```
1. Compute $\tau_3$ and set $M_0 := M(t_0)h/(1 + h^2\tau_3)$.
2. Compute the roots of (1.8) having positive real parts.
3. For $k = 0, 1, \dots, n-1$,
       find the unique $\omega_k$ as in Section 1.2 such that $M_k = \omega_k^\top J - J\omega_k$
       set $M_{k+1} = \omega_k M_k \omega_k^\top$
   end
4. Reconstruct $M_n := M_n(1 + h^2\tau_3)/h \approx M(t_n)$.
```

The DMV6 algorithm:

```
1. Compute $\tau_3$, $\tau_5$ and set $\tilde{\tau}_5 = \tau_5 - 2\tau_3^2$ and $M_0 := M(t_0)h/(1 + h^2\tau_3 + h^4\tilde{\tau}_5)$.
2. Compute the roots of (1.8) having positive real parts.
3. For $k = 0, 1, \dots, n-1$,
       find the unique $\omega_k$ as in Section 1.2 such that $M_k = \omega_k^\top J - J\omega_k$
       set $M_{k+1} = \omega_k M_k \omega_k^\top$
   end
4. Reconstruct $M_n := M_n(1 + h^2\tau_3 + h^4\tilde{\tau}_5)/h \approx M(t_n)$.
```

5. How Large Can the Step Size h Be?

The choice of the step size is intimately related to the accuracy of the method. Other limitations to the choice of the step size for a method can be loss of stability or

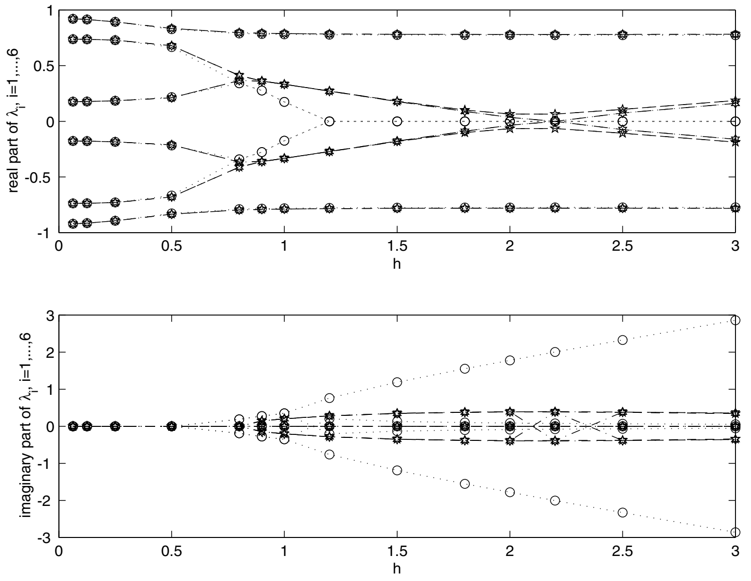

Fig. 5.1. Real (top) and imaginary (bottom) part of the eigenvalues λ_i as a function of the step size h. Legend: DMV corresponds to circles joined by the dotted lines; DMV4 to pentagrams joined by the dash–dotted lines; and DMV6 to hexagons joined by the dashed lines.

convergence (for instance, for an implicit scheme). For the DMV-type algorithms described in this paper, there is a further limitation, which comes from the spectrum S of the polynomial $P(\lambda)$ in (1.8): when two opposite sign roots become purely imaginary, then the choice of ω in the Moser–Veselov equation (1.6) is no longer unique; in other words, relation (1.6) cannot give rise to a one-to-one map.

In this section we intend to investigate this limit on the step size h for the original DMV algorithm and the new proposed algorithms DMV4 and DMV6.

In Figure 5.1 we plot real and imaginary parts of the eigenvalues λ_i of the characteristic equation (1.8) versus step size h for our running test problem (see Section 7). The roots are computed using MATLABs routine `roots` for finding zeros of polynomials. When h is very small, all the roots are real (they correspond to plus/minus the squares of the moments of inertia in J). For larger steps, the typical behavior is that some roots become complex.

For the DMV algorithm (circles joined by a dotted line) the maximum allowed step size is $h \approx 1.2$; for this value, two pairs of opposite sign complex and conjugate roots hit the imaginary axis. This also occurs for the DMV6 algorithm, though at a larger step size, $h \approx 2.2$; afterward the spectrum recovers its original splitting (1.7). Remarkably, the roots of the DMV4 algorithm do not appear to hit the imaginary axis at all for this test problem and preserve the desired spectrum splitting (1.7) for all values of h. Does this mean that the limitations on the step size are dictated only by the accuracy required in the computations? The answer is no: let us look more carefully at the polynomial $-P(\lambda)$ in (1.8). Assume that $\mathbf{m}_0$ is the given initial condition, that we scale through the algorithms DMV4 and DMV6 to $\mathbf{m}_0/(1+h^2\tau_3)$ for DMV4 and $\mathbf{m}_0/(1+h^2\tau_3+h^4\tilde{\tau}_5)$ for DMV6. Let us denote by $-P_{4,h}(\lambda)$ and $-P_{6,h}(\lambda)$ the corresponding characteristic polynomials in which we factor out the dependence on the scaling from the initial condition,

$$\begin{aligned} -P_{4,h}(\lambda) &= \lambda^6 - \left(J_1^2+J_2^2+J_3^2 - \frac{h^2}{(1+h^2\tau_3)^2}\|\mathbf{m}_0\|^2\right)\lambda^4 \\ &\quad + \left(J_1^2J_2^2+J_1^2J_3^2+J_2^2J_3^2 - \frac{h^2}{(1+h^2\tau_3)^2}H_2\right)\lambda^2 - J_1^2J_2^2J_3^2, \quad (5.1) \\ -P_{6,h}(\lambda) &= \lambda^6 - \left(J_1^2+J_2^2+J_3^2 - \frac{h^2}{(1+h^2\tau_3+h^4\tilde{\tau}_5)^2}\|\mathbf{m}_0\|^2\right)\lambda^4 \\ &\quad + \left(J_1^2J_2^2+J_1^2J_3^2+J_2^2J_3^2 - \frac{h^2}{(1+h^2\tau_3+h^4\tilde{\tau}_5)^2}H_2\right)\lambda^2 \\ &\quad -J_1^2J_2^2J_3^2, \quad (5.2) \end{aligned}$$

where $H_2 = H\Delta/(\|\mathbf{m}_0\|^2(J_1J_2+J_1J_3+J_2J_3))\|$ is independent of h. By the same token,

$$\begin{aligned} -P_h(\lambda) &= \lambda^6 - (J_1^2+J_2^2+J_3^2 - h^2\|\mathbf{m}_0\|^2)\lambda^4 \\ &\quad +(J_1^2J_2^2+J_1^2J_3^2+J_2^2J_3^2 - h^2H_2)\lambda^2 - J_1^2J_2^2J_3^2. \end{aligned}$$

We consider the polynomial as parametrized by the h and we analyze their roots letting $h\to\infty$. Starting with P_h, we have

$$-P_h(\lambda) \approx h^2(\|\mathbf{m}_0\|^2\lambda^4 - H_2\lambda^2)$$

for h sufficiently large. Two pure imaginary roots go to $\pm\infty$, two pure imaginary roots go to zero, and two stay real. In particular, this means that (1.6) cannot be made into a map which is symplectic.

The situation is different for the scaled methods DMV4 and DMV6. Since

$$\lim_{h\to\infty}\frac{h^2}{(1+h^2\tau_3)^2} = \lim_{h\to\infty}\frac{h^2}{(1+h^2\tau_3+h^4\tilde{\tau}_5)^2} = 0,$$

we have

$$\begin{aligned} -P_{4,\infty} &= -P_{6,\infty} = -P_{4,0} = -P_{6,0} \\ &= \lambda^6 - (J_1^2+J_2^2+J_3^2)\lambda^4 + (J_1^2J_2^2+J_1^2J_3^2+J_2^2J_3^2)\lambda^2 - J_1^2J_2^2J_3^2, \end{aligned}$$

which is the same as having $h=0$, in other words, no motion! Hence, the map is still integrable, symplectic, and momentum preserving, but the motion becomes slower and slower until it stops at infinity. This seems to indicate that there exist

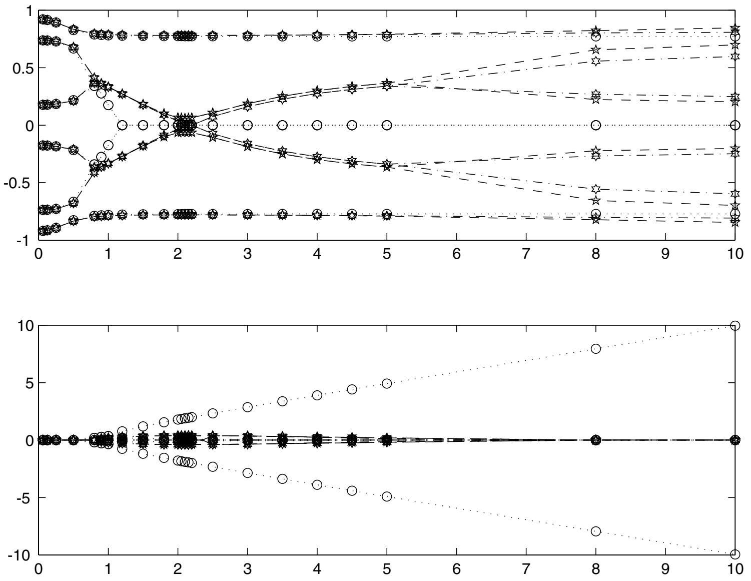

Fig. 5.2. Real (top) and imaginary (bottom) part of the eigenvalues λ_i as a function of the step size h letting $h \to \infty$. Legend: DMV corresponds to circles joined by the dotted lines; DMV4 to pentagrams joined by the dash–dotted lines; and DMV6 to hexagons joined by the dashed lines.

optimal values of the step size for moderate h. This is interesting in some molecular dynamics applications, where the cost of long-range forces often dominates so that the work to implement the RB algorithm becomes unimportant and the maximum step size one can use becomes potentially more relevant.

The behavior of the roots of $-P_h$, $-P_{4,h}$, $-P_{6,h}$ for large h is displayed in Figure 5.2.

6. RATTLE $\equiv$ DMV

In this section we consider the RATTLE algorithm for the integration of the RB and prove that RATTLE is the same as DMV.

In brief we mention that the RATTLE algorithm can be derived as a Lie–Poisson integrator and is described in greater detail in [**6**]. A reformulation of RATTLE for the RB equations is in [**6**, p. 247], which we report here for convenience. Note that we replace h with $-h$ to obtain the same equations. The notation is as in this paper.

The RATTLE algorithm for the RB equations:

With the same notation of this paper, $M(t_0) = J\Omega(t_0) + \Omega(t_0)J$ is the initial condition.

1. Set $Y_0 = J\Omega(t_0)$.

2. For $k = 0, 1, 2, \dots, n-1$,
   - determine a symmetric matrix S such that $Q = I - hJ^{-1}Y_{1/2}$ is orthogonal, $Y_{1/2} = Y_0 - hS$;
   - determine a symmetric matrix T such that $J^{-1}Y_1$ is skew-symmetric, where $Y_1 = Y_0 - hY_{1/2}Y_{1/2}^\top J^{-1} - hT$;
   - set $Y_0 = Y_1$;

   end
3. Reconstruct $\Omega_n = J^{-1}Y_1 \approx \Omega(t_n)$, $M_n = J\Omega_n + \Omega_n J \approx M(t_n)$.

The first step in 2 is equivalent to solving the Riccati equation

$$J^{-1}S + SJ^{-1} = -(Y_0 - hS)^\top J^{-2}(Y_0 - hS).$$

This method is of order 2, it preserves the Casimir $\|\mathbf{m}\|_2^2$ and the Hamiltonian H in (3.1).

Theorem 1. *The RATTLE method for the RB equation is equivalent to the DMV algorithm.*

Proof. It is sufficient to demonstrate that the two algorithms are equivalent for the first step.

By construction, we have $M(t_0) = J\Omega(t_0) + \Omega(t_0)J = Y_0 - Y_0^\top$, and $Y_0 = J(I-Q)/h + hS$, where S is a symmetric matrix determined so that Q is orthogonal. Hence, by direct computation,

$$hM(t_0) = h(Y_0 - Y_0^\top) = Q^\top J - JQ,$$

namely, Q is a solution of the Moser–Veselov equation (1.6). Similarly, since T is a symmetric matrix and $Y_0 - hS = J(I-Q)/h$,

$$\begin{aligned} h(Y_1 - Y_1^\top) &= h(Y_0 - Y_0^\top) - h^2 \frac{J(I-Q)}{h}\frac{(I-Q^\top)J}{h}J^{-1} \\ &\quad + h^2 J^{-1}\frac{J(I-Q)}{h}\frac{(I-Q^\top)J}{h} \\ &= Q^\top J - JQ - J(2I - Q - Q^\top) + (2I - Q - Q^\top)J \\ &= JQ^\top - QJ, \end{aligned}$$

which, compared with (2.1), reveals that $Y_1 - Y_1^\top$ is precisely the DMV approximation for $M(t_1) = M(t_0 + h)$ since, for sufficiently small h, the DMV gives the only order 2 solution of (2.1) consistent with the dynamics of the flow. Hence RATTLE coincides with DMV. □

As a consequence, our results on the DMV can be immediately be extended to RATTLE for the RB, and it is of no surprise that the method preserves both the Hamiltonian and the Casimirs of the system, since the method is integrable.

Moreover, our analysis allows us to improve the orders of the RATTLE approximation to orders 4 and 6 simply by scaling the initial condition and scaling back the final approximation!

The RATTLE4, RATTLE6 algorithms:

1. Compute τ_3 for order 4 and $\tau_3, \tau_5, \tilde{\tau}_5 = \tau_5 - 2\tau_3^2$ for order 6. Set $Y_0 = J\Omega(t_0)/(1+h^2\tau_3)$ for order 4 and $Y_0 = J\Omega(t_0)/(1+h^2\tau_3+h^4\tilde{\tau}_5)$ for order 6.
2. For $k = 0, 1, 2, \ldots, n-1$,
   - determine a symmetric matrix S such that $Q = I - hJ^{-1}Y_{1/2}$ is orthogonal, $Y_{1/2} = Y_0 - hS$;
   - determine a symmetric matrix T such that $J^{-1}Y_1$ is skew-symmetric, where $Y_1 = Y_0 - hY_{1/2}Y_{1/2}^{\top}J^{-1} - hT$;
   - set $Y_0 = Y_1$;

   end
3. Reconstruct $\Omega_n = (1+h^2\tau_3)J^{-1}Y_1 \approx \Omega(t_n)$, $M_n = J\Omega_n + \Omega_n J \approx M(t_n)$ for order 4 and $\Omega_n = (1+h^2\tau_3+h^4\tilde{\tau}_5)J^{-1}Y_1 \approx \Omega(t_n)$, $M_n = J\Omega_n+\Omega_n J \approx M(t_n)$ for order 6.

7. Comparison with Other Methods

In this section we compare the numerical results of the DMV algorithm with the new proposed schemes DMV4 and DMV6 over the integration interval $[0, T]$. Moreover, we also compare with:

- the explicit second-order Lie–Poisson integrator for the RB of McLachlan (LP2) [**11**], which is obtained by splitting the Hamiltonian H into three pieces, $H = \mathcal{H}_1 + \mathcal{H}_2 + \mathcal{H}_3$, $\mathcal{H}_i = m_i^2/(J_{i-1} + J_{i+1})$, whose corresponding vector fields can be integrated exactly, generating the flows $\varphi_{i,h}$. Then, at each step, the numerical approximation is computed as

$$\mathbf{m}_{k+1} = \varphi_{3,h/2} \circ \varphi_{2,h/2} \circ \varphi_{1,h} \circ \varphi_{2,h/2} \circ \varphi_{3,h/2}(\mathbf{m}_k), \qquad k = 0, 1, 2, \ldots;$$

  and
- the well-known implicit midpoint rule (IMR), reading

$$y_{k+1} = y_k + hf\left(\frac{y_k + y_{k+1}}{2}\right), \qquad k = 0, 1, 2\ldots,$$

  for the problem $y' = f(y)$, which is second-order, energy and momentum preserving–but implicit.

Table 4. Error for the various methods and selected step sizes.

Method	$h = \frac{1}{16}$	$h = \frac{1}{2}$	$h = 1.2$	$h = 2.2$	$h = 2.5$	$h = 4$
LP2	6.7903e−03	5.1043e−01	1.6055e+00	1.7002e+00	1.9489e+00	1.3902e+00
IMR	1.5494e−04	9.9329e−03	1.3119e−01	3.9514e−01	2.5276e−01	6.1905e−01
DMV	1.5014e−02	5.9899e−01	NaN	NaN	NaN	NaN
DMV4	1.757e−07	7.6167e−04	1.0785e−01	1.6094e+00	5.0245e−01	5.1624e−01
DMV6	1.962e−10	1.6440e−06	7.0269e−02	NaN	7.0407e−01	1.0519e−01

In Figure 7.1 we display the error of the methods versus step size $h = \frac{1}{16}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}$, and then 0.8, 1, 1.2, 1.5, 2.2, 2.5, 4. The error is computed at $T = 100$ as the 2 norm of the difference between the numerical approximation and the exact solution for our running example,

$$\mathbf{m}_0 = \begin{pmatrix} 0.4165 \\ 0.9072 \\ 0.0577 \end{pmatrix}, \qquad J = \begin{pmatrix} 0.9218 & 0 & 0 \\ 0 & 0.7382 & 0 \\ 0 & 0 & 0.1762 \end{pmatrix}.$$

(Without loss of generality, $\mathbf{m}_0$ is normalized so that $\|\mathbf{m}_0\|_2 = 1$.)

The slope of the lines agrees with the order of the methods under consideration. The original DMV method is the one that has lowest accuracy and the most limitations on the step size: it fails at around $h = 1.2$ (see also Table 4) never to recover. Also DMV6 fails, but at around $h = 2.2$ to recover afterward.

In Figure 7.2 we plot instead the cost of the methods (in floating point operations) versus their accuracy. We have not taken into consideration the computation of τ_3 and $\tilde{\tau}_5$, but these are reasonably small (about 40 operations for τ_3 and 70 for $\tilde{\tau}_5$) and done only once for all the computations, hence of little relevance to the overall estimate of the methods.[2] The computation of the coefficients of the polynomial P_λ and the computation of the roots is instead taken into account and performed at each step. Although this can be avoided and done once and for all, it appears to give more stable approximations especially for larger step sizes and long intervals of integration (probably this is due to a different accumulation of rounding errors, since the two algorithms are in theory equivalent). Roughly speaking, the computation of the coefficients of P_λ requires about 30 operations, while the computation of its roots requires about 50 operations. The roots are computed by slightly modifying the explicit formula in [**1**] in order to avoid cancellations and other type of errors (see, for instance, Press et al. [**14**]).

As one can see, the methods DMV, DMV4, and DMV6 have the same floating point operations—they differ only on the scaling of the initial condition. The gain in accuracy for the latter two is however so significant that DMV6 is the method which is the most efficient, especially when good accuracy is required.

[2] It could be observed that the entries in τ_3 and $\tilde{\tau}_5$ give the coefficients of the polynomial P_λ almost for free.

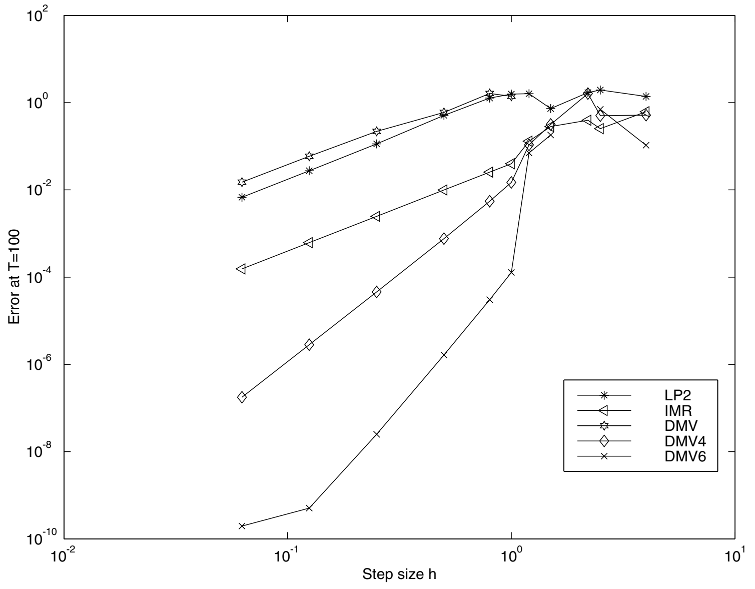

Fig. 7.1. Error versus step size computed at $T = 100$ for the methods LP2, IMR, DMV, DMV4, and DMV6.

The core of the algorithms DMV, DMV4, and DMV6 (solution of the Moser–Veselov equation) is implemented using the algorithm described in Section 9, and it requires about 1250 operations per step, when all the roots are real, and 30% more (about 2000 operations) for complex roots. When the roots are real, the cost of a single time stepping with the DMV (and its higher-order implementations) is about 1330 operations, including the computation of roots, roughly 22 times more expensive that a step of the LP2 algorithm, which amounts to about 60 operations per step. This agrees well with the numerical results in Figure 7.2.

Figure 7.3 is similar to Figure 7.2, but now the roots of the polynomial P_λ are computed only at the beginning of the computations, which leads to some overall savings (approximately 1.5%). However, the quality of the solution is somehow deteriorated, for instance, for the smallest step size h= 1/16, there appears to be more accumulation of round-off error. For comparison, we have also presented an implementation of RATTLE with the order 6 improvement. The error is of the same magnitude as DMV6, however the cost is higher due to the fact that we need to solve a nonlinear equation. In this numerical experiment, this is done by simply applying a fixed point iteration to the Riccati equation

$$J^{-1}S + SJ^{-1} = -(Y_0 - hS)^{\top} J^{-2}(Y_0 - hS).$$

Finally, in Figure 7.4, we plot the solutions obtained with the methods LP2, IMR, DMV4, and DMV6 for a large value of h, $h = 5$. The method LP2 performs very poorly in this case, the solution completely loses its shape, DMV is not plotted

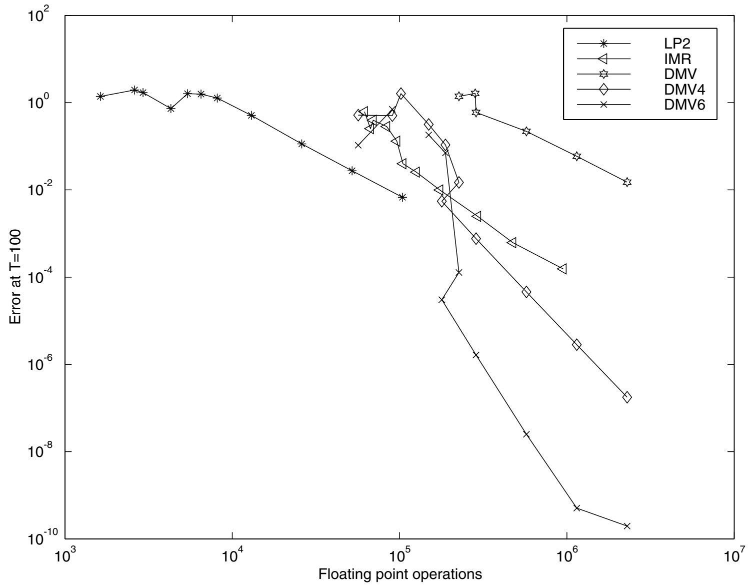

Fig. 7.2. Floating point operations versus accuracy ($T = 100$) for the methods LP2, IMR, DMV, DMV4, and DMV6.

because it does not generate a solution for this choice of step size. As our analysis predicted, DMV4 and DMV6 still produce a solution, which has precisely the same integrals as the original problem, but is much too slow compared with the true solution. Remarkably, the IMR still preserves some form and the integrals as the original problem. For this test example the IMR fails to converge at $h \approx 6.5$.

For a comparison, when the step size is large but not too large (for instance, $h = 1$), a plot would show that both the IMR, DMV4 and DMV6, give very good agreement with the exact solution and are indistinguishable from it to the naked eye. The methods LP2 and DMV are instead much less accurate.

8. The Full Dynamics: The Configuration Update

To obtain the full dynamics of the RB, the output of the DMV algorithm (M_k, the discrete body momentum, and Ω_k, the discrete angular velocity, both in body coordinates) has to be combined with the space configuration. If $X(t)$ is the orthogonal matrix in SO(3) that gives the position of the body in the fixed coordinate system at time t, then it obeys the differential equation

$$X' = X\Omega, \qquad X(0) = X_0. \tag{8.1}$$

In the Moser–Veselov derivation of the discrete Lagrangian described in Section 1.1, the configuration is associated with the orthogonal matrix X_k, so that the

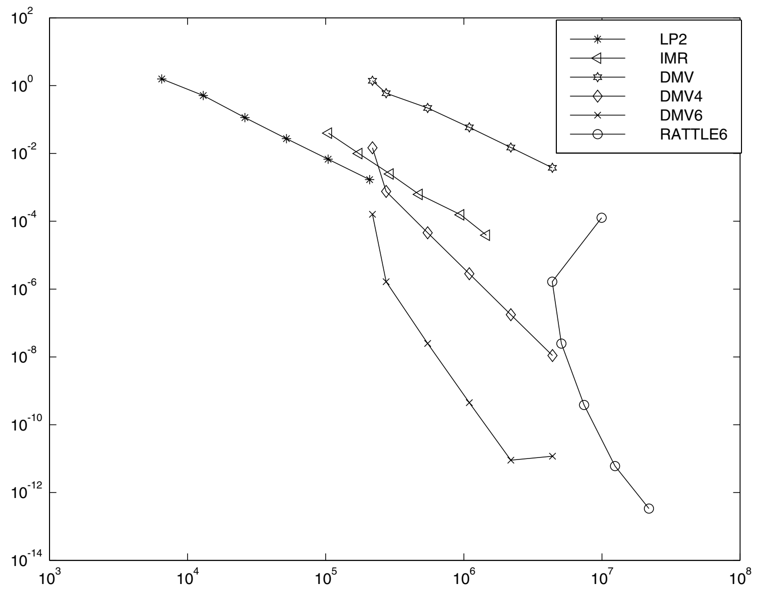

Fig. 7.3. Floating point operations versus accuracy ($T = 100$) for the methods LP2, IMR, DMV, DMV4, DMV6, and RATTLE6

time $k+1$ update is given by

$$X_{k+1} = X_k \omega_{k+1}^{\top}. \tag{8.2}$$

As pointed out by Lewis and Simo [**8**], the resulting pair (X_k, M_k) does not preserve spatial momentum but a staggered momentum conservation law. To obtain a spatial momentum conservation, Lewis and Simo [**8**] suggest replacing (8.2) with the shifted configuration-momentum pair (X_{k+1}, M_k) that, up to relabeling variables, corresponds to solving (1.3) and then obtaining the configuration update as

$$X_{k+1} = X_k \omega_k^{\top}. \tag{8.3}$$

Using the series expansion of ω_k as in (2.4) and performing a backward error analysis of the configuration update (8.3), as was done earlier in this paper, we obtain that X_{k+1} is the solution to order 4 of the modified differential equation

$$X' = X\Omega + h^2 g_3, \tag{8.4}$$

where

$$g_3 = X \begin{pmatrix} 0 & -\sigma_{3,3}w_3 & \sigma_{3,2}w_2 \\ \sigma_{3,3}w_3 & 0 & -\sigma_{3,1}w_1 \\ -\sigma_{3,2}w_2 & \sigma_{3,1}w_1 & 0 \end{pmatrix}, \quad \Omega = \hat{\mathbf{w}}, \quad \mathbf{w} = \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix}^{\top}, \tag{8.5}$$

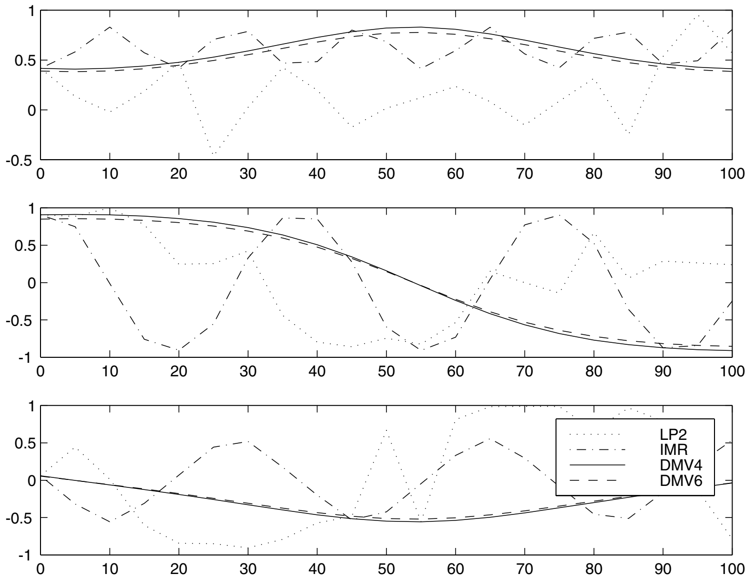

Fig. 7.4. Numerical approximations of the RB equations for h large ($h = 5$).

and

$$\sigma_{3,i} = \frac{1}{6\Delta^2}\big((2\det(J)\,\operatorname{tr}(J) + C_{J,2} + J_{i-1}J_i^2J_{i+1})\|\mathbf{m}\|^2 + (2J_i(J_{i-1}+J_{i+1}) - (J_i^2 + J_{i+1}J_{i-1}))H_2\big), \qquad (8.6)$$

$i = 1, 2, 3$, with periodicity over the indices (for instance, it should be intended that $J_0 = J_3$ and so on). The computation of the higher-order terms of the modified vector field follows in a similar fashion.

It is immediately observed that:

- The modified equation (8.4) is not a time reparametrization of (8.1) since, in general, $\sigma_{3,i} \neq \sigma_{3,j}$ for $i \neq j$.
- Some of the terms appearing in the $\sigma_{3,i}$ are in common with the scaling parameter τ_3.

As a consequence, our proposed scaling of the initial condition M_0, leading to DMV4 and DMV6, *does not generally increase* the order of the configuration update, which remains 2. However, for some values of the initial condition, and of the inertia moments J_i, the configuration error can be substantially reduced, due to cancellation of common terms.

Finally, we conclude this section observing that higher-order approximation of the configuration can be obtained by solving

$$X' = X\tilde{\Omega},$$

where $\tilde{\Omega}$ is a function which is obtained interpolating to suitable order the values of Ω_k (computed by DMV4 and DMV6) and their derivatives.

9. On the Explicit Solution of the Moser–Veselov Equation for the 3×3 Rigid Body

9.1. *Reduction to a Matrix Riccati Equation*

Consider the matrix equation

$$M = X^\top J - JX, \tag{9.1}$$

where M is a skew-symmetric matrix and J is diagonal. Cardoso and Leite [**3**] have shown that every solution of (9.1) (not necessarily orthogonal) can be written as

$$X = J^{-1}(-\tfrac{1}{2}M + S),$$

for some symmetric matrix S. The proof is immediate. Clearly, if S is symmetric, $X = J^{-1}(-M/2 + S)$ is a solution of (9.1). Next, assume that X solves (9.1) and set $\tilde{X} = JX$. Then (9.1) becomes $\tilde{X}^\top - \tilde{X} = M$. Recall that any matrix $\tilde{X}$ can be uniquely decomposed in its symmetric and skew-symmetric part,

$$\tilde{X} = \tfrac{1}{2}(\tilde{X} + \tilde{X}^\top) + \tfrac{1}{2}(\tilde{X} - \tilde{X}^\top),$$

from which it follows that the skew-symmetric part of $\tilde{X}$ is precisely $-M/2$, while the symmetric part is an arbitrary symmetric matrix S. From this it follows immediately that $X = J^{-1}(-M/2 + S)$.

Furthermore, they have shown that X is an orthogonal solution of (9.1) if and only if S is a symmetric solution of the Riccati equation

$$SS^\top + S(M/2) + (M/2)^\top S^\top - (\tfrac{1}{4}M^2 + J^2) = 0. \tag{9.2}$$

Thus, the problem is reduced to the well-known problem of solving Riccati equations.

Riccati equations can be associated to symplectic matrices. For this problem, the corresponding symplectic matrix is

$$H_{\text{sympl}} = \begin{bmatrix} M/2 & I \\ M^2/4 + J^2 & M/2 \end{bmatrix}. \tag{9.3}$$

If $M^2/4 + J^2$ is positive definite, it has been shown in [**3**] that (9.2) has a unique solution S which is symmetric, positive definite, and such that the eigenvalues of

$$W = (-M/2 + S)^\top$$

have positive real parts. This matrix W is precisely the same matrix in [**12**] such that $WW^\top = J^2$ and

$$\omega = J^{-1}W^\top.$$

In this case, Cardoso and Leite [**3**] propose the following algorithm for the computation of X, as the unique solution of (9.1) in the special orthogonal group $\mathrm{SO}(N)$:

(1) Find a real Schur form of H_{sympl},

$$\tilde{Q}^\top H_{\text{sympl}} \tilde{Q} = \begin{bmatrix} T_{11} & T_{12} \\ O & T_{22} \end{bmatrix}, \tag{9.4}$$

where T_{11} and T_{22} are block upper-triangular matrices such that the real parts of the spectrum of T_{11} are positive and the real parts of the spectrum of T_{22} are negative.

(2) Partition $\tilde{Q}$ accordingly,

$$\tilde{Q} = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{bmatrix}.$$

Then, compute

$$S = Q_{21}Q_{11}^{-1}.$$

(3) Compute

$$X = J^{-1}\left(-\frac{M}{2} + S\right).$$

A real Schur form of H_{sympl} can be computed using the QR iteration for eigenvalues (Golub and van Loan [**5**]), which is an $\mathcal{O}N^3$ operation for an $N \times N$ matrix M, a cost that is comparable with that of using implicit ODE methods for the solution of the RB equations.

In what follows, we focus on the particular case when $N = 3$. In this case, the eigenvalues of H_{sympl} are explicitly known (they are the roots of the polynomial $P(\lambda)$), hence it is possible to find an explicit spectral decomposition of H_{sympl} (without using the QR eigenvalue method). Only the eigenvectors corresponding to the roots λ_1, λ_2, and λ_3 of (1.8) with positive real part need be computed. We construct the partial real Schur decomposition (9.4) of this eigenspace and hence X. This yields an *explicit* numerical method for the reduced RB equations.

To this end, assume that $\boldsymbol{x}_\lambda = \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{bmatrix}$ is an eigenvector of H_{sympl} with eigenvalue λ. From

$$\begin{bmatrix} M/2 & I \\ M^2/4 + J^2 & M/2 \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{bmatrix} = \lambda \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \end{bmatrix}$$

we deduce that $\boldsymbol{x}_2 = \lambda \boldsymbol{x}_1 - \frac{1}{2}M\boldsymbol{x}_1$, hence $\boldsymbol{x}_1$ is an eigenvector for the quadratic eigenvalue problem

$$J^2\boldsymbol{x}_1 + \lambda M \boldsymbol{x}_1 = \lambda^2 \boldsymbol{x}_1. \tag{9.5}$$

9.2. *Computation of the Eigenvectors*

For the irreducible case (all the roots are real $\lambda_1, \lambda_2, \lambda_3 > 0$), we solve the quadratic eigenvalue problem (9.5),

$$A_{\lambda_i}\boldsymbol{x}_{1,\lambda_i} = (J^2 - \lambda_i^2 I + \lambda_i M)\boldsymbol{x}_{1,\lambda_i} = 0, \qquad i = 1, 2, 3,$$

by computing the QR factorization of $A_{\lambda_i} = Q_{\lambda_i} R_{\lambda_i}$, where R_{λ_i} is now an upper-triangular matrix with the last diagonal element equal to zero (A_{λ_i} is singular, rank 2). We choose the third component of $\boldsymbol{x}_{1,\lambda_i} \neq 0$ arbitrary and determine the remaining components by solving the upper triangular system

$$R_{\lambda_i}\boldsymbol{x}_{1,\lambda_i} = 0.$$

Then we construct $\boldsymbol{x}_{2,\lambda_i} = \lambda_i \boldsymbol{x}_{1,\lambda_i} - (M/2)\boldsymbol{x}_{1,\lambda_i}$, and $\boldsymbol{x}_{\lambda_i} = \begin{bmatrix} \boldsymbol{x}_{1,\lambda_i} \\ \boldsymbol{x}_{2,\lambda_i} \end{bmatrix}$.

The procedure is analogous in the case of one real and two complex and conjugate roots, except for the fact that the complex eigenvalues require complex arithmetic. In that case, complex arithmetic is avoided in the usual way, computing the real and imaginary parts of the corresponding eigenvectors. Specifically, assume that

$$\lambda = \mu + \mathrm{i}\nu$$

is a complex eigenvalue with eigenvector $\boldsymbol{y}_1 = \boldsymbol{u}_1 + \mathrm{i}\boldsymbol{v}_1$. From the eigenvector equation $J^2\boldsymbol{y}_1 + \lambda M\boldsymbol{y}_1 - \lambda^2\boldsymbol{y}_1 = 0$, separating the real and imaginary parts we obtain

$$\begin{aligned} J^2\boldsymbol{u}_1 + \mu M\boldsymbol{u}_1 - \nu M\boldsymbol{v}_1 + (\nu^2 - \mu^2)\boldsymbol{u}_1 + 2\mu\nu\boldsymbol{v}_1 &= 0, \\ J^2\boldsymbol{v}_1 + \nu M\boldsymbol{u}_1 + \mu M\boldsymbol{v}_1 - 2\mu\nu\boldsymbol{u}_1 + (\nu^2 - \mu^2)\boldsymbol{v}_1 &= 0, \end{aligned}$$

which can be written as

$$\begin{bmatrix} J^2 + \mu M + (\nu^2 - \mu^2)I & -\nu + 2\mu\nu I \\ \nu M - 2\mu\nu I & J^2 + \mu M + (\nu^2 - \mu^2)I \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_1 \\ \boldsymbol{v}_1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

If $\mu \neq 0$, the 6×6 coefficient matrix on the left-hand side of the above linear system has rank 4. The vectors $\boldsymbol{u}_1$ and $\boldsymbol{v}_1$ can be computed by performing a QR factorization of the coefficient matrix (eventually, with column pivoting) and then by solving an upper-triangular system by backward substitution.

Once $\boldsymbol{u}_1$ and $\boldsymbol{v}_1$ are computed, we also compute

$$\boldsymbol{u}_2 = \mu\boldsymbol{u}_1 - \nu\boldsymbol{v}_1 - \frac{M}{2}\boldsymbol{u}_1, \qquad \boldsymbol{v}_2 = \nu\boldsymbol{u}_1 + \mu\boldsymbol{v}_1 - \frac{M}{2}\boldsymbol{v}_1,$$

hence, we set $\mathbf{u} = \begin{bmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \end{bmatrix}$ and $\mathbf{v} = \begin{bmatrix} \boldsymbol{v}_1 \\ \boldsymbol{v}_2 \end{bmatrix}$.

Note that it is not necessary to compute the real Schur form of H_{sympl} explicitly. What is needed is only an orthogonal basis that spans the eigenvectors space. This basis can be obtained either by QR or simply by a Gram–Schmit orthogonalization.

- Construct the 6×3 matrix $V = [\boldsymbol{x}_{\lambda_1}, \boldsymbol{x}_{\lambda_2}, \boldsymbol{x}_{\lambda_3}]$ if λ_i, $i = 1, 2, 3$, are all real and positive; or $V = [\boldsymbol{x}_{\lambda_1}, \boldsymbol{u}, \boldsymbol{v}]$ if λ_1 is real and positive and $\lambda_2 = \bar{\lambda}_3$ are the two complex conjugate roots with real part $\mu > 0$.
- Compute the QR decomposition (or the Gram–Schmit orthogonalization) $\tilde{V} R = V$, and decompose $\tilde{V} = \begin{bmatrix} \tilde{V}_1 \\ \tilde{V}_2 \end{bmatrix}$, where $\tilde{V}_1$, $\tilde{V}_2$ are 3×3 blocks.
- Compute $S = \tilde{V}_2 \tilde{V}_1^{-1}$.
- Compute $W = -\frac{1}{2} M + S$.
- Compute $\omega = J^{-1} W^{\top}$.

This is the core of the explicit algorithm used to implement Step 2 of DMV and Step 3 of DMV4, DMV6 for the 3×3 RB.

The method can also be implemented in a cheaper way by using LU factorizations instead of QR, and even cheaper by computing the explicit spectral decomposition as described originally in [**12**]. This usually reduces the cost of the method by about 50% per iteration, that is, approximately 600 operations per step, thereby resulting in a method that is roughly 10 times more expensive than LP2, and retains all the integrals to machine accuracy. However, these algorithms appear to be less stable over long-term integrations and for large step sizes than the one used for our implementation due to the inversion of the eigenvectors matrix, which can be poorly conditioned when the eigenvectors are close to linear dependence.

Acknowledgments

This work was initiated while the authors were guests of the program "Special Year in Geometric Integration," hosted at the Center for Advanced Study, Oslo. We wish to thank the staff for making our stay so pleasant and productive. Thanks to Ben Leimkhuler and Sebastian Reich for indicating possible connections with the RATTLE algorithm.

References

[1] M. Abramowitz,and I. A. Stegun, *Handbook of Mathematical Functions*, Dover, New York, 1965.

[2] V. I. Arnold, *Mathematical Methods of Classical Mechanics*, 2nd ed. Graduate Texts in Mathematics, Vol. 60, Springer-Verlag, New York, 1989.

[3] J. R. Cardoso, and F. S. Leite, The Moser–Veselov equation, *Linear Algebra Appl.* **360** (2003), 237–248.

[4] P. Deift, L.-C. Li, and C. Tomei, Loop groups, discrete versions of some classical integrable systems and rank-2 extensions, *Mem. Amer. Math. Soc.* **100**(479) (1992), 1–101.

[5] G. H. Golub, and C. F. van Loan, *Matrix Computations*, 2nd ed., Johns Hopkins University Press, Baltimore, 1989.

[6] E. Hairer, C. Lubich, and G. Wanner, *Geometric Numerical Integration*, Springer Series in Computational Mathematics, No. 31, Springer-Verlag, Berlin, 2002.

[7] A. Iserles, H. Munthe-Kaas, S. P. Nørsett, and A. Zanna, Lie-group methods, *Acta Numer.* **9** (2000), 215–365.

[8] D. Lewis, and J. C. Simo, *Conserving Algorithms for the n-Dimensional Rigid Body*, Fields Inst. Commum., American Mathematical Society, Providence, RI, 1995.

[9] J. E. Marsden, and M. West, Discrete mechanics and variational integrators, *Acta Numer.* **10** (2001), 357–514.

[10] J. E. Marsden, S. Pekarsky, and S. Shkoller, Discrete Euler–Poincaré and Lie–Poisson equations, *Nonlinearity* **12** (1999), 1647–1662.

[11] R. I. McLachlan, Explicit Lie–Poisson integration and the Euler equations, *Phys. Rev. Lett.* **71** (1993), 3043–3046.

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

[13] H. Z. Munthe-Kaas, and S. Krogstad, On enumeration problems in Lie–Butcher theory, *Future Generation Computer Systems*, Special issue **19**(7) (2003), 1197–1205.

[14] W. H., Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes*: *The Art of Scientific Computing*, Cambridge University Press, Cambridge, 1986.