To appear in *Modeling, Identification and Control*
This is an extended version of a paper printed in *Applied Modelling and Simulation — Proceedings of the 38th SIMS Simulation Conference*, T. Iversen (editor), pp. 231–242, Tapir trykkeri, Norway, 1996

Simulation of Ordinary Differential Equations on Manifolds

—

Some Numerical Experiments and Verifications*

Arne Marthinsen† Hans Munthe-Kaas‡ and Brynjulf Owren§

Department of Mathematical Sciences, NTNU, N-7034 Trondheim, Norway

Abstract

During the last few years, different approaches for integrating ordinary differential equations on manifolds have been published. In this work, we consider two of these approaches. We present some numerical experiments showing benefits and some pitfalls when using the new methods. To demonstrate how they work, we compare with well known classical methods, e.g. Newmark and Runge-Kutta methods.

AMS Subject Classification: 65L05, 34A50

Key Words: ordinary differential equations, manifolds, numerical analysis, initial value problems.

1 Introduction

In recent years, many authors have argued that certain problems in mechanics cannot be solved in a satisfactory way by classical numerical methods. For instance, it might be desirable that problems which possess certain invariants are solved with a numerical method that preserves the same invariants. Typical examples are the symplectic structure in a Hamiltonian system, the energy in a conservative mechanical system and the angular momentum of a rotating rigid body in space. Classical numerical methods normally fail to preserve these quantities.

Special energy preserving methods as well as symplectic methods are old, but it is within the last decade that symplectic Runge-Kutta methods have become popular (see e.g. [13]). In computational mechanics, numerical integration of ordinary differential equations on Lie groups is of interest and has been investigated by e.g. Simo and Wong ([14]). An example of such a Lie group is the configuration space of a rigid body (also called SO(3)).

During the last few years, different approaches for integrating ordinary differential equations on manifolds have been published. Isospectral flows are considered in [2] and orthogonal flows in [5]. In this work, we consider two of these approaches (see [4] and [10]).

The two approaches are essentially different. Crouch and Grossman ([4]) formulate their methods in terms of flows of simpler systems of vector fields. The Munthe-Kaas methods ([10]) are developed

*This work was in part sponsored by The Norwegian Research Council under contract no. 111038/410, through the SYNODE project. WWW: `http://www.imf.unit.no/num/synode`

†Email: `Arne.Marthinsen@imf.unit.no`, WWW: `http://www.imf.unit.no/~arnema`
‡Email: `Hans.Munthe-Kaas@imf.unit.no`, WWW: `http://www.ii.uib.no/~hans`
§Email: `Brynjulf.Owren@imf.unit.no`, WWW: `http://www.imf.unit.no/~bryn`

for Lie groups, but they may be generalized to include more general manifolds. In Euclidean space, both types of methods coincide with the traditional Runge-Kutta formulation.

Classical numerical methods make use of vector space operations, that are not generally defined on manifolds, where the new methods operate. These methods only perform operations that are defined on manifolds, for instance computing flows and pullbacks. The methods are developed using elementary theory from differential geometry.

Submanifolds embedded in some Euclidean space often arise from constraints being imposed or invariants the solution is known to possess. The numerical methods are defined in such a way that the computed solution is restricted to the submanifold. The numerical error produced by the method will only result in error *on* the manifold, and will not let the numerical solution drift away from the constraint manifold. Due to physical understanding of many problems, this submanifold on which the solution evolves is often known in advance. The integration methods incorporate this knowledge to integrate exactly with respect to the constraints.

The configuration space of a rigid body may be viewed as a 3-dimensional manifold embedded in $\mathbb{R}^9$. The new methods integrate by computing flows of vector fields on this manifold, and the numerical solution will, hence, never leave the three dimensional manifold.

2 How the Methods Work

In this section, we explain briefly how the numerical methods work.

2.1 Explicit Runge-Kutta Methods

Classical, s-stage, explicit Runge-Kutta methods for solving the first order initial value problem

$$y' = f(t, y), \qquad y(t_0) = y_0, \tag{1}$$

on the interval $[t_n, t_n + h]$, are traditionally written as

$$\begin{aligned} k_i &= f(t_n + c_i h, y_n + h \sum_{j=1}^{i-1} a_{ij} k_j) \\ y_{n+1} &= y_n + h \sum_{i=1}^{s} b_i k_i, \end{aligned}$$

where y_n and y_{n+1} are approximations to $y(t_n)$ and $y(t_{n+1})$, respectively. The coefficients of the method are normally specified through a Butcher tableau,

$$\begin{array}{c|ccccc} 0 & 0 & & & \\ c_2 & a_{21} & 0 & & \\ \vdots & \vdots & \ddots & 0 & \\ c_s & a_{s1} & \cdots & a_{s,s-1} & 0 \\ \hline & b_1 & \cdots & b_{s-1} & b_s \end{array} \quad = \quad \begin{array}{c|c} c & A \\ \hline & b^T \end{array}$$

and, as usual it is assumed that $c_i = \sum_{j=1}^{i-1} a_{ij}$, $i = 1, \ldots, s$.

Runge-Kutta methods work in vector spaces, and a geometric interpretation of a step is shown in Figure 1. At the starting point, $p_1 = y_n$, we evaluate the right hand side of the differential equation to find k_1. p_2 is computed by following the direction k_1 a distance ha_{21}. At p_2, the differential equation is again evaluated, and we find k_2. We continue like this, and compute the final step by taking a linear combination, $h\sum_{j=1}^{s} b_j k_j$, of $k_1, \ldots, k_s$. At each point p_i we compute a direction, and a linear combination of these directions gives the final direction to follow a distance h. Note that the operations which constitute the primitives of the schemes are not generally defined on manifolds.

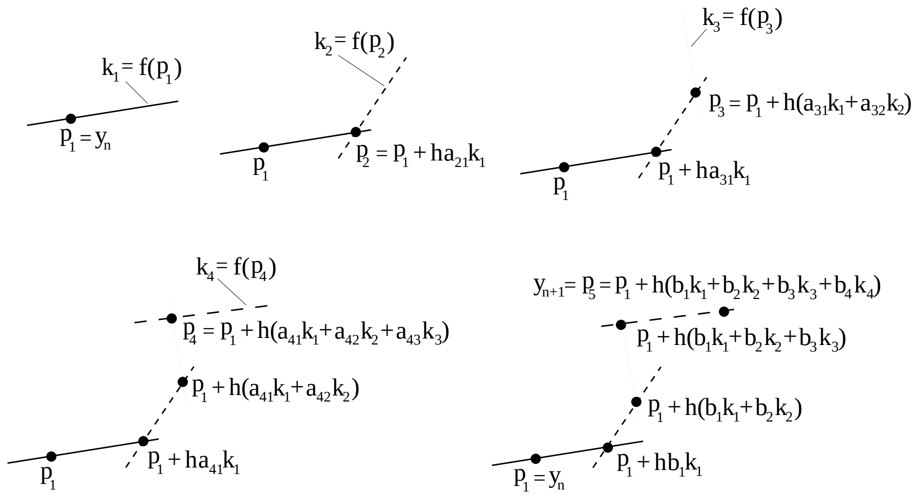

Figure 1: Geometric interpretation of how a Runge-Kutta method with four stages integrates from point y_n to y_{n+1}.

2.2 Crouch-Grossman Methods

The Crouch-Grossman method ([4]) is formulated for differential equations on the form

$$\dot{y}(t) = F(t,y) = \sum_{i=1}^{n} f_i(t,y)E_i(y), \qquad y(0) = y_0 \in \mathcal{M},$$

whose solution evolves on an n dimensional manifold, $\mathcal{M}$, of $\mathbb{R}^N$. $y \in \mathcal{M}$, $E_1, \ldots, E_n$ are real analytic vector fields on $\mathbb{R}^N$ and $f_i : \mathbb{R} \times \mathbb{R}^N \to \mathbb{R}$. We denote by $F_p(y)$ a vector field with coefficients frozen at (τ, p) relative to a given frame,

$$F_p(y) = F_{(\tau,p)}(y) = \sum_{i=1}^{n} f_i(\tau,p)E_i(y).$$

The explicit Crouch-Grossman method may now be written as follows:

$$\begin{array}{rcll}
p &=& y_k, & \\
\nu_1(p) &=& p, & \\
F_p^1(h)(y) &=& F_p(y), & \\
\nu_r(h,p) &=& e^{ha_{r,r-1}F_p^{r-1}} e^{ha_{r,r-2}F_p^{r-2}} \cdots e^{ha_{r,1}F_p^1} p, & \multirow{2}{*}{$\left.\right\} \quad r = 2, \ldots, s,$} \\
F_p^r(h)(y) &=& F_{\nu_r(h,p)}(y), & \\
y_{k+1} &=& e^{hb_sF_p^s} e^{hb_{s-1}F_p^{s-1}} \cdots e^{hb_1F_p^1} p. &
\end{array}$$

where the exponentiation of a vector field as usual denotes its flow.[1] A geometric interpretation of a step is shown in Figure 2. At point $p_1 = z_n$ we compute the vector field, F_p^1, defined by

[1] The flow of a vector field $F \in \mathfrak{X}(\mathcal{M})$ is an $\mathbb{R}$-indexed family of mappings $\phi_t : \mathcal{M} \to \mathcal{M}$ such that $\phi_{t'}(y(t)) = y(t+t')$ for all t and t' and all integral curves y of F.

the differential equation. Point p_2 is found by following the integral curve of F_p^1 through p_1 a distance ha_{21}, and F_p^2 is the vector field defined by the equations at point p_2. Point p_3 is found by following the integral curve of F_p^1 through p_1 a distance ha_{31} and the integral curve of F_p^2 through this point a distance ha_{32}. When all the vector fields F_p^i, $i = 1 \dots, s$, are computed, a step with the Crouch-Grossman method is taken by following in turn the integral curves of $F_p^1, \dots, F_p^s$ a distance $hb_1, \dots, hb_s$, respectively.

In Euclidean space, this method concides with the traditional Runge-Kutta formulation, since following integral curves just becomes following straight lines.

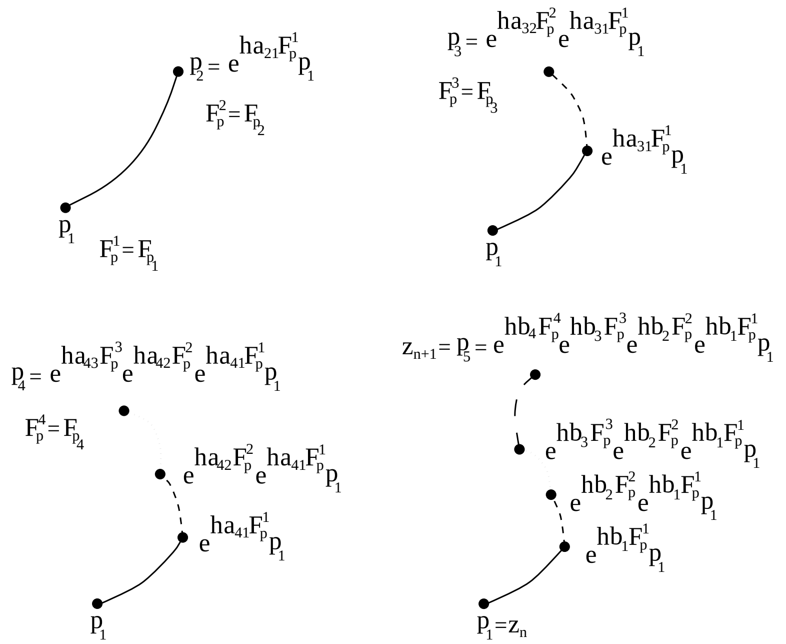

Figure 2: Geometric interpretation of how a Crouch-Grossman method with four stages integrates from point z_n to z_{n+1}. The lines denote flow of certain vectorfields.

2.3 Munthe-Kaas Methods

Let $\mathcal{M}$ be a Lie group (which is also a manifold), $F \in \mathfrak{X}(\mathcal{M})$ a vector field on $\mathcal{M}$ and $\mathfrak{g}$ the Lie algebra of $\mathcal{M}$. Let further $y_0 = y(t_0) \in \mathcal{M}$, and denote by $\overline{F}$ the element in $\mathfrak{g}$ which is tangent to F in the point y_0. We will also denote by $\Psi_{t,Y} : \mathcal{M} \to \mathcal{M}$ the flow of a given vector field Y, and by $\Psi_{t,Y}^* F$ the pullback of F along Y. An explicit Runge-Kutta step in the Munthe-Kaas setting may then be written as follows.

$$
\begin{aligned}
K_i &= \overline{\Psi_{t=1,Y_i}^* F}, \qquad i = 1, \dots, s, \\
Y_i &= h \sum_{j=1}^{i-1} a_{ij} K_j, \qquad i = 1, \dots, s, \\
Y_h &= h \sum_{j=1}^{s} b_j K_j, \\
y_1 &= \Psi_{t=1,Y_h}(y_0),
\end{aligned}
$$

where K_i, Y_i, $Y_h \in \mathfrak{g}$ and y_0, $y_1 \in \mathcal{M}$. A three stage method in this setting may thus be written in detail as

$$
\begin{aligned}
p_1 &= y_0, \\
K_1 &= F(p_1) \\
\widetilde{p}_2 &= \exp(ha_{21}K_1) \\
p_2 &= \widetilde{p}_2 \cdot p_1 \\
K_2 &= \widetilde{p}_2^{-1}F(p_2)\widetilde{p}_2 \\
\widetilde{p}_3 &= \exp\left(h(a_{31}K_1 + a_{32}K_2)\right) \\
p_3 &= \widetilde{p}_3 \cdot p_1 \\
K_3 &= \widetilde{p}_3^{-1}F(p_3)\widetilde{p}_3 \\
y_1 &= \exp\left(h(b_1K_1 + b_2K_2 + b_3K_3)\right) p_1.
\end{aligned}
$$

A geometric interpretation of a step with the Munthe-Kaas algorithms is shown in Figure 3.

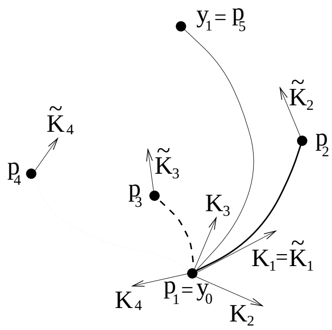

Figure 3: Geometric interpretation of how a Munthe-Kaas method with four stages works. The lines denote flow of certain vector fields.

On manifolds, we can not add vectors as we do in Euclidean spaces. The base point of the vectors is important, and when evaluating the equation at a point, the resulting vector is not, in general, compatible with a vector computed in another point. The pullback operation, which transforms the vectors in different fibers of the tangent bundle to a common fiber, makes use of the inverse tangent mapping. When all vectors are in the same fiber, we do have a vector space, and it makes sense to compute linear combinations of the vectors. This is what we have tried to picture in Figure 3. Let $\widetilde{K}_i$ be the value of a vector field at point p_i, and let K_i be the value at e of the pulled back vector field. Then $K_1 = \widetilde{K}_1$, since we compute K_1 at the base point p_1. $\widetilde{K}_2$ is computed at p_2, which has been found by following the integral curve of K_1 a distance ha_{21}. $\widetilde{K}_3$ is computed at point p_3, which has been found by following the integral curve of the linear combination $h(a_{31}K_1 + a_{32}K_2)$ of K_1 and K_2. $\widetilde{K}_3$ and $\widetilde{K}_4$ are computed in the same way. The final step is taken by following the integral curve of the linear combination $h\sum_{j=1}^{s} b_jK_j$ of all the previously computed vector fields $K_1, \ldots, K_s$.

The above algorithm is the original Munthe-Kaas algorithm that was presented in [10]. That paper explains the ideas and developes the tools needed. One drawback, however, is that it turns out that methods of the above mentioned type are at most of order two. Various variants of the algorithm has therefore been suggested. The main idea is to use so called *correction functions* to increase the order of the schemes (see [11]).

A general initial value problem on a matrix Lie group, G, can be written as

$$y_t' = f(y_t) \cdot y_t, \qquad y_0 = p,$$

where $f : G \to \mathfrak{g}$. Here, $y_t : \mathbb{R} \to G$ is a curve on G and $y_t' : \mathbb{R} \to \mathrm{T}G$ is a curve on the tangent space to G. The following setting approximates the solution to order three on matrix Lie groups when using the coefficients of an arbitrary third order classical Runge-Kutta method ([11]).

```
y_0 = p
for n = 0, 1, 2, ...
    I_1 = k_1 = f(y_n)
    for i = 2, ..., s
        u_i = h Σ_{j=1}^{i-1} a_{ij} k_j
        k_i = f(exp(u_i) · y_n)
    end
    v = h Σ_{j=1}^{s} b_j k_j
    ṽ = v - (h/6)[I_1, v]
    y_{n+1} = exp(ṽ) · y_n
end
```

where I_1, k_i, u_i, v, $\widetilde{v} \in \mathfrak{g}$ and $y_i \in G$.

The following algorithm approximates the solution to order four when using the coefficients of an arbitrary fourth order classical Runge-Kutta method ([11]).

Let $c_i = \sum_{j=1}^{s} a_{ij}$ and $d_i = \sum_{j=1}^{s} a_{ij} c_j$. Compute the coefficients (m_1, m_2, m_3) by solving the linear system

$$(m_1\ m_2\ m_3) \begin{pmatrix} c_2 & c_2^2 & 2d_2 \\ c_3 & c_3^2 & 2d_3 \\ c_4 & c_4^2 & 2d_4 \end{pmatrix} = (1\ 0\ 0).$$

The fourth order scheme is then given by

```
y_0 = p
for n = 0, 1, 2, ...
    I_1 = k_1 = f(y_n)
    for i = 2, ..., s
        u_i = h Σ_{j=1}^{i-1} a_{ij} k_j
        ũ_i = u_i - (c_i h/6)[I_1, u_i]
        k_i = f(exp(ũ_i) · y_n)
    end
    I_2 = (m_1(k_2 - I_1) + m_2(k_3 - I_1) + m_3(k_4 - I_1))/h
    v = h Σ_{j=1}^{s} b_j k_j
    ṽ = v - (h/4)[I_1, v] - (h^2/24)[I_2, v]
    y_{n+1} = exp(ṽ) · y_n
end
```

where I_i, k_i, u_i, $\widetilde{u}_i$, v, $\widetilde{v} \in \mathfrak{g}$ and $y_i \in G$.

Note that Crouch-Grossman methods are constructed for general differentiable manifolds, which in particular include the Lie groups, while Munthe-Kaas methods are constructed for Lie groups only. There should, however, be no major problems with generalizing the Munthe-Kaas methods.

2.4 Newmark Methods

The Hilber-Hughes-Taylor method ([7]) for solving the linearized version of the initial value problem

$$y'' = g(t, y, y'), \qquad y(t_0) = y_0, \qquad y'(t_0) = y_0', \tag{2}$$

i.e.

$$My'' + Cy' + Ky = F(t), \qquad y(t_0) = y_0, \qquad y'(t_0) = y_0',$$

where M, C and K are $m \times m$ matrices and M is non-singular, may be written as

$$\begin{aligned} d_{n+1} &= d_n + hv_n + h^2 \left(\left(\frac{1}{2} - \beta \right) a_n + \beta a_{n+1} \right) \\ v_{n+1} &= v_n + h \left((1-\gamma)\, a_n + \gamma a_{n+1} \right) \end{aligned}$$

where a_{n+1} is computed from

$$M a_{n+1} + C\left((1+\alpha) v_{n+1} - \alpha v_n \right) + K\left((1+\alpha) d_{n+1} - \alpha d_n \right) = F(t_{n+1} + \alpha h).$$

Here, d_n and v_n are approximations to $y(t_n)$ and $y'(t_n)$, respectively. The classical Newmark method is obtained with $\alpha = 0$.

Note that (2) may be written as a first order system (1), and the explicit Runge-Kutta method is applicable also for second order initial value problems. Solution of (2) by Runge-Kutta-Nyström methods is discussed in [6].

3 Numerical Experiments

In this section we present numerical experiments showing some properties of the new methods. Let the methods be defined as follows.

RK4: the classical fourth order Runge-Kutta method given by the Butcher tableau

$$\begin{array}{c|cccc} 0 & & & & \\ 1/2 & 1/2 & & & \\ 1/2 & 0 & 1/2 & & \\ 1 & 0 & 0 & 1 & \\ \hline & 1/6 & 2/6 & 2/6 & 1/6 \end{array}$$

CG3: a third order Crouch-Grossman method given by the Butcher tableau

$$\begin{array}{c|ccc} 0 & & & \\ -1/24 & -1/24 & & \\ 17/24 & 161/24 & -6 & \\ \hline & 1 & -2/3 & 2/3 \end{array}$$

MK3: a third order Munthe-Kaas method given by the same Butcher tableau as method **CG3**.

MK4: a fourth order Munthe-Kaas method given by the same Butcher tableau as method **RK4**.

N: a Newmark method with $\alpha = 0$, $\beta = 1/4$ and $\gamma = 1/2$.

Test Case 1:

The first test problem we solve is a first order system of differential equations (see [4]),

$$\dot{R} = S(f(R))R, \qquad R \in \mathrm{GL}(3) \subset \mathbb{R}^{3\times 3}, \tag{3}$$

where $f(R) = (f_1(R), f_2(R), f_3(R))^T$ and f_i are real-valued functions on GL(3), the group of nonsingular 3×3 matrices. $S(a)$ is the skew symmetric matrix,

$$S(a) = \begin{pmatrix} 0 & a_3 & -a_2 \\ -a_3 & 0 & a_1 \\ a_2 & -a_1 & 0 \end{pmatrix}$$

where $a = (a_1, a_2, a_3)^T \in \mathbb{R}^3$. It is not difficult to show that when S is skew-symmetric, every solution of (3) satisfies

$$R^T(t)R(t) = R_0^T R_0, \qquad \text{with} \qquad R(0) = R_0.$$

In the case $R_0 \in \mathrm{SO}(3)$, the solution of (3) evolves on SO(3). Note that by defining the vector fields

$$E_1(R) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{pmatrix} R, \qquad E_2(R) = \begin{pmatrix} 0 & 0 & -1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} R, \qquad E_3(R) = \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} R,$$

problem (3) may be rewritten in the form

$$\dot{R} = \sum_{i=1}^{3} f_i(R) E_i(R).$$

In this test case, we have chosen f_1 to be the Frobenius norm of R, f_2 to be the 3-norm of R and f_3 to be the 4-norm of R.

We are not primarily interested in the solution to this problem, but rather in how well the methods stay on the constraint manifold and the methods' order when solving the problem.

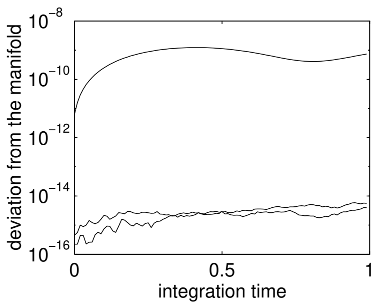

Figure 4: The picture shows how well the solution stays on the manifold. The two lower lines are produced by the **CG3** and **MK3** methods, while the upper line is produced by the **RK4** method. The stepsize used in the simulation was 0.01.

Figure 4 shows how well the solution stays on the manifold. The two lower lines show the results from the **CG3** and **MK3** methods, while the upper line shows the result from the **RK4** method. Note that **CG3** and **MK3** stay on the manifold within the order of accuracy of the computer, while **RK4** drifts off with an error of order 10^{-9}.

In order to compute the order of the methods, we first compute the "correct" solution with a high order numerical method with a strict error tolerance (10^{-12}). We then run the other methods with stepsizes $h_1 = h$ (the whole interval), $h_2 = h/2, \ldots, h_n = h/2^{n-1}$, and compute the error, e_i, $i = 1, \ldots, n$, relative to the "correct" solution. A sequence of approximations to the order of the method is found by computing

$$p_i = \frac{\log(e_{i-1}/e_i)}{\log(h_{i-1}/h_i)}, \qquad i = 2, \ldots, n. \tag{4}$$

In these test cases, we have put $n = 10$.

Figure 5 shows that the approximations from the **CG3** and **MK3** methods asymptotically approach order three.

Figure 6 shows the result after applying the coefficients of the **RK4** method in the Crouch-Grossman setting to the same test problem. By construction, all coefficients that satisfy the

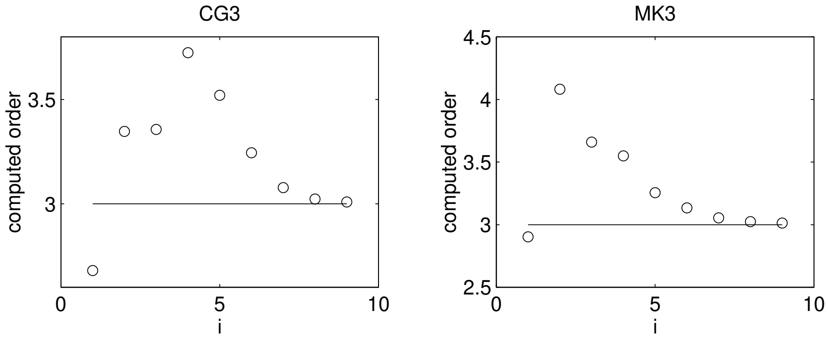

Figure 5: The pictures show the computed order of the **CG3** method (left) and **MK3** method (right). The order is computed with (4). The x-axis denotes i (describing the stepsize) and the y-axis denotes the computed order, p_i.

classical order conditions of order two will generate numerical schemes of order two in the Crouch-Grossman and the Munthe-Kaas settings. This is due to the fact that for first and second order methods the order conditions for classical and the new methods coincide. For third and higher order methods the classical order conditions are only a subset of the new methods' order conditions (see [9]).

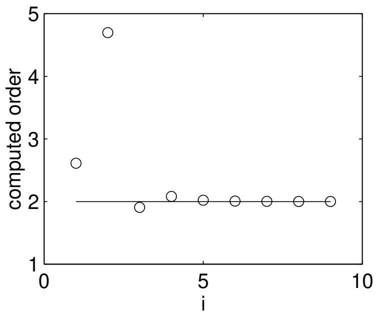

Figure 6: The picture shows the computed order of the Crouch-Grossman method with the coefficients from the **RK4** method. The order is computed with (4). The x-axis denotes i (describing the stepsize) and the y-axis denotes the computed order, p_i.

Test Case 2:

The second test problem we solve is a second order problem modelling a humming top. A schematic picture is shown in Figure 7.

The position of the humming top can be described by a matrix $B(t) \in \mathrm{SO}(3)$. This matrix transforms local to global coordinates, i.e. a point with local coordinates $\vec{Q}$ ($\vec{Q}$ constant) on the humming top has global coordinates $\vec{q}(t) = B(t)\vec{Q}$. The angular velocity $\omega \in \mathfrak{so}(3)$ is identified with a vector $\widetilde{\omega} = (\omega_1, \omega_2, \omega_3)^T \in \mathbb{R}^3$ through

$$\begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{pmatrix} \leftrightarrow \begin{pmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{pmatrix}.$$

The configuration space of the problem is thus the Lie group $\mathcal{G} = \mathrm{TSO}(3) \simeq \mathrm{SO}(3) \times \mathfrak{so}(3)$ and its Lie algebra $\mathfrak{g}$ is $\mathrm{TTSO}(3) \simeq \mathfrak{so}(3) \times \mathfrak{so}(3)$. The operations on the Lie group and its Lie algebra are:

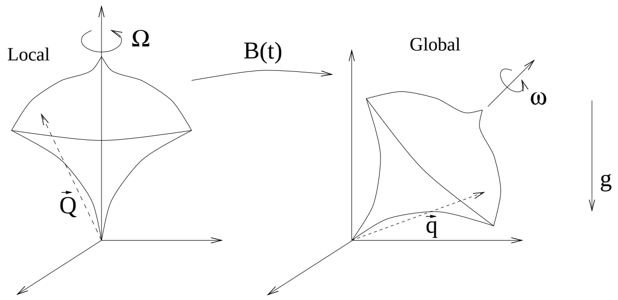

Figure 7: A schematic picture of the humming top in local and global coordinates.

Product in $\mathcal{G}$:	$(a, v) \cdot (b, w) = (a \cdot b, v + w)$.
Addition in $\mathfrak{g}$:	$(u, v) + (\widetilde{u}, \widetilde{v}) = (u + \widetilde{u}, v + \widetilde{v})$.
Multiplication by scalar in $\mathfrak{g}$:	$\alpha(u, v) = (\alpha u, \alpha v)$.
Lie bracket in $\mathfrak{g}$:	$[(u, v), (\widetilde{u}, \widetilde{v})] = ([u, \widetilde{u}], 0)$.
Exponential map in $\mathfrak{g}$:	$\exp(u, v) = (\exp(u), v)$.

Here, on the right hand side of the definitions, $\cdot$ is the multiplication in SO(3), $+$ is the addition in $\mathfrak{so}(3)$), $[\ ,\ ]$ is the bracket (commutator) on $\mathfrak{so}(3)$ and exp is the exponential map on $\mathfrak{so}(3)$.

The differential equation describing the motion of the humming top is given by

$$(B(t), \omega(t))' = F\left(B(t), \omega(t)\right) \cdot \left(B(t), \omega(t)\right),$$

where $B(t) \in \mathrm{SO}(3)$, $\omega(t) \in \mathfrak{g}$ and $(B(t), \omega(t)) \in \mathcal{G}$, and $F : \mathcal{G} \to \mathfrak{g}$ is defined by

$$F : (B, \omega) \mapsto (\omega, \omega'),$$

and ω' is given by $\omega' = \alpha[\omega, f]$ (or, alternatively, $\widetilde{\omega}' = \alpha(\widetilde{\omega} \times \widetilde{f})$). Here, α is a scalar and f is the external force. In this test case, we have assumed that $\alpha = 1$ and that the gravitational force ($mg := 1$, where $m = 1$, is the mass of the humming top) is the only external force:

$$\widetilde{f} = \begin{pmatrix} 0 \\ 0 \\ -1 \end{pmatrix} \leftrightarrow f = \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

The initial conditions used in the computations are

$$B(0) = B_0 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1/2\sqrt{3} & 1/2 \\ 0 & -1/2 & 1/2\sqrt{3} \end{pmatrix} \quad \text{and} \quad \omega(0) = \omega_0 = \begin{pmatrix} 0.0 & 0.8 & -1.0 \\ -0.8 & 0.0 & 1.1 \\ 1.0 & -1.1 & 0.0 \end{pmatrix}.$$

The model described above is only valid if the spin of the humming top is very large.

When solving the problem with the Runge-Kutta and the Newmark method, we formulate the problem in the following way:

$$\begin{pmatrix} B(t) \\ \omega(t) \end{pmatrix}' = \begin{pmatrix} \omega(t)B(t) \\ \alpha[\omega(t), f]) \end{pmatrix} \quad \text{with} \quad B(0) = B_0 \quad \text{and} \quad B'(0) = \omega(0) = \omega_0.$$

The pictures in Figure 8 show that the approximations from the **CG3** and **MK3** methods asymptotically approach order three. The pictures in Figure 9 show the results for the **N** and **RK4** methods and Figure 10 shows the results for the **MK4** method.

Note that **RK4** is supposed to be of order four (it is a classical fourth order method), but as we see in Figure 11, the method does not generate approximations that stay on the manifold. If we, as in Test Case 1, had used the **RK4** coefficients in the Crouch-Grossman setting, the order of the method would have been only two. The **N** method generates a solution of order two, but this

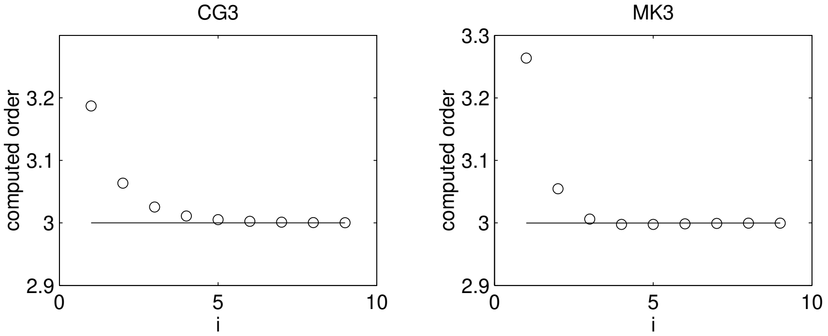

Figure 8: The pictures show the computed order of the **CG3** method (left) and **MK3** method (right) (first component, $B(t)$). The order is computed with (4). The x-axis denotes i (describing the stepsize) and the y-axis denotes the computed order, p_i.

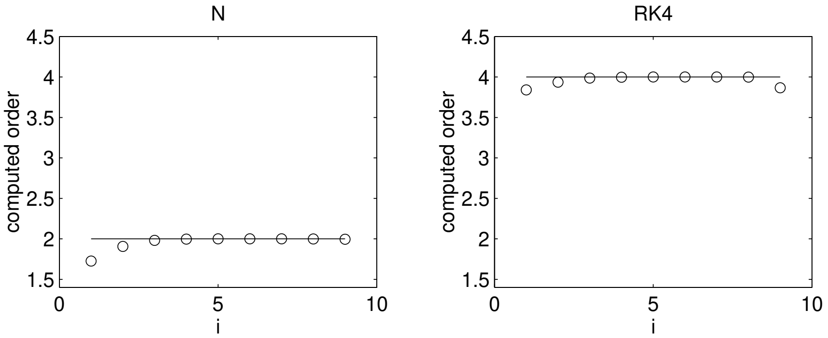

Figure 9: The pictures show the computed order of the **N** method (left) and **RK4** method (right) (first component, $B(t)$). The order is computed with (4). The x-axis denotes i (describing the stepsize) and the y-axis denotes the computed order, p_i.

method also fails to produce solutions that stay on the manifold. On the other hand, we see that the **CG3** and **MK3** methods behave as anticipated.

Only vector space operations are performed in the second component in the formulation on $\mathcal{G}$, and hence the Crouch and Grossman or Munthe-Kaas methods with coefficients of a classical method will be of the same order as the classical method. This is, however, not true for the first component, B.

4 Conclusion

We have described two quite new approaches for integrating ordinary differential equations on manifolds and presented numerical comparisons with the classical Runge-Kutta and Newmark methods. These new methods are especially useful in problems where certain quantities should be conserved. If we view SO(3) as embedded in $\mathbb{R}^9$, we may, as we did in Test Case 1, apply classical numerical methods to a problem with SO(3) as configuration space. In this case the new methods are in general more computationally expensive than classical methods, but the numerical experiments show a qualitative improvement of the solution.

The Crouch-Grossman and the Munthe-Kaas methods differ in cost per step from the classical Runge-Kutta methods. Classical methods with s stages roughly use s function evaluations in addition to a number of linear combinations. s-stage Crouch-Grossman methods use s function

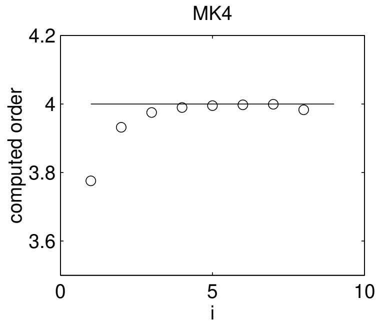

Figure 10: The picture shows the computed order of the **MK4** method (first component, $B(t)$). The order is computed with (4). The x-axis denotes i (describing the stepsize) and the y-axis denotes the computed order, p_i.

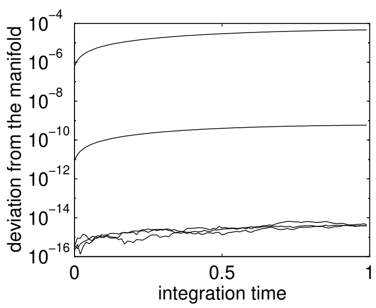

Figure 11: The picture shows how well the solution stays on the manifold. The three lower lines are produced by the **CG3**, **MK3** and **MK4** methods, while the two upper lines are produced by the **RK4** and **N** methods (the upper line is **N**). The stepsize used in the simulation was 0.01.

evaluations, $s(s+1)/2$ exponentiations and a number of linear combinations. The s-stage Munthe-Kaas methods of order three involve $s-1$ function evaluations, s exponentiations, 1 commutator and a number of linear combinations. The s-stage Munthe-Kaas methods of order four involve $s-1$ function evaluations, s exponentiations, $s+1$ commutators and a number of linear combinations.

Computing commutators between matrices and matrix exponentials are very time consuming operations. In the case of general 3×3 matrices, the commutator may be computed with a total of 117 floating point operations. In the 3×3 skew-symmetric case, the number of operations may be reduced to 9. For general 3×3 matrices one needs 823 floating point operations (counted by Matlab) to compute the exponential, but if we consider $v \in \mathfrak{so}(3)$, the Euler and Rodrigues formula (see e.g. [14]) yields

$$e^v = I + \frac{\sin\alpha}{\alpha} v + \frac{1}{2}\left[\frac{\sin\frac{\alpha}{2}}{\frac{\alpha}{2}}\right]^2 v^2, \quad \text{where } \alpha = \|v\|_F/\sqrt{2},$$

which computes the exponential by 41 floating point operations (we have counted the evaluation of sine and square root as one operation).

An alternative to the exact computation of the exponential is to use rational approximations, e.g. Padé approximants. The diagonal Padè approximants maps $\mathfrak{so}(n)$ analytically to $\mathrm{SO}(n)$ for all n, but there may not always exist functions, analytical in a neighbourhood of 0, other than exp, from the Lie algebra to the Lie group that in addition satisfies $f(0) = 1$ and $f'(0) = 1$. An example of such a case is that exp is the only analytical function from $\mathfrak{sl}(n)$ to $\mathrm{SL}(n)$ satisfying the requirements when $n > 2$.

For special problems a significant qualitative improvement of the solution has been demonstrated. One future challenge is, however, to find a clever way to choose frames. It should be noted that the new methods are generalizations of the classical methods, and a trivial choice of frames in $\mathbb{R}^n$ reduces the new methods to the classical ones.

References

[1] V. I. Arnold: *Mathematical Methods of Classical Mechanics*, Springer-Verlag, GTM 60, Second edition, 1989

[2] M. P. Calvo, A. Iserles and A. Zanna: *Numerical Solution of Isospectral Flows*, Tech. Rep. DAMTP 1995/NA03, Department of Applied Mathematics and Theoretical Physics, University of Cambridge, 1995

[3] M. P. Calvo, A. Iserles and A. Zanna: *Runge-Kutta Methods on Manifolds*, Preprint, 1995

[4] P. E. Crouch and R. Grossman: *Numerical Integration of Ordinary Differential Equations on Manifolds*, J. Nonlinear Sci., Vol. 3, pp. 1–33 (1993)

[5] L. Dieci, R. D. Russel and E. S. van Vleck: *Unitary Integrators and Applications to Continuous Orthonormalization Techniques*, SIAM J. Numer. Anal., Vol. 31, No. 1, pp. 261–281 (1994)

[6] E. Hairer, S. P. Nørsett and G. Wanner: *Solving Ordinary Differential Equations I*, SCM 8, Springer-Verlag, Second Edition, 1993

[7] H. M. Hilber, T. J. R. Hughes and R. L. Taylor: *Improved Numerical Dissipation for Time Integration Algorithms in Structural Dynamics*, Earthquake Eng. and Struct. Dynamics, Vol. 5, pp. 283–292, 1977

[8] A. Iserles and A. Zanna: *Qualitative Numerical Analysis of Ordinary Differential Equations*, Tech. Rep. DAMTP 1995/NA05, Department of Applied Mathematics and Theoretical Physics, University of Cambridge, 1995

[9] A. Marthinsen and B. Owren: *Order Conditions for Integration Methods Based on Rigid Frames*, manuscript, 1996

[10] H. Munthe-Kaas: *Lie-Butcher Theory for Runge-Kutta Methods*, BIT 35, pp. 572–587 (1995)

[11] H. Munthe-Kaas: *Runge-Kutta Methods on Lie Groups*, submitted to BIT, 1996

[12] H. Munthe-Kaas and A. Zanna: *Numerical Integration of Differential Equations on Homogeneous Manifolds*, in *Foundation of Computational Mathematics*, F. Cucker (ed.), Springer Verlag, 1997, to appear

[13] J. M. Sanz-Serna and M. P. Calvo: *Numerical Hamiltonian Problems*, Chapman & Hall, 1994

[14] J. C. Simo and K. K. Wong: *Unconditionally Stable Algorithms for Rigid Body Dynamics that Exactly Preserve Energy and Momentum*, Intl. J. for Num. Methods in Engineering, Vol. 31, 1991

[15] A. Zanna and H. Munthe-Kaas: *Iterated Commutators, Lie's Reduction Method and Ordinary Differential Equations on Matrix Lie Groups*, in *Foundation of Computational Mathematics*, F. Cucker (ed.), Springer Verlag, 1997, to appear