UNIVERSITY OF CAMBRIDGE

Numerical Analysis Reports

Projected Runge–Kutta methods for differential equations on matrix Lie groups

Yunkang Liu

DAMTP 1998/NA01
January 1998

Department of Applied Mathematics and Theoretical Physics
Silver Street
Cambridge
England CB3 9EW

Projected Runge–Kutta methods for differential equations on matrix Lie groups

Yunkang Liu

Gonville and Caius College, Cambridge University, Cambridge CB2 1TA, U.K.

Abstract For differential equations on a matrix Lie group it is known that most traditional methods do not keep the numerical solutions in the Lie group. To obtain numerical solutions with the correct qualitative behaviour we solve overdetermined systems of differential-algebraic equations, including differential equations on the corresponding Lie algebra, by standard Runge–Kutta methods with projection in the Lie algebra. Theoretical analysis and numerical experiment demonstrate that this new class of Lie-group invariant methods is more efficient than previous ones.

Key words. Differential-algebraic equation, matrix Lie group, matrix Lie algebra, projected Runge-Kutta method, local error estimation

AMS subject classifications. 65L05, 65L06, 22-04, 22E60

1 Introduction

Roughly speaking, a differential manifold $\mathcal{G}$ equipped with a group product that maps $\mathcal{G} \times \mathcal{G}$ to $\mathcal{G}$ is called a Lie group. The Lie algebra $\mathbf{g}$ of a Lie group $\mathcal{G}$ is the tangent space to $\mathcal{G}$ at the identity. In the case of matrix Lie group, the product is the usual matrix multiplication. In general, initial value problems for differential equations on a matrix Lie group $\mathcal{G}$ can be written in the form

$$y' = a(t, y)y, \qquad t > t_0, \qquad y(t_0) = y_0, \tag{1.1}$$

where $a : \mathbb{R} \times \mathcal{G} \mapsto \mathbf{g}$ is a smooth function and $y_0 \in \mathcal{G}$. It is well-known that the solution of (1.1), subject to existence, stays in the Lie group $\mathcal{G}$.

This paper is concerned with the numerical solution of (1.1). It has been shown by Calvo, Iserles and Zanna [3] and Iserles [11] that naive use of traditional numerical methods such as multistep methods and Runge–Kutta methods does not guarantee that the numerical solution stays in the Lie group. One simple, though not quite rigourous, reason for the failure of traditional methods is that matrix Lie groups are nonlinear manifold, linear combinations of elements in a group need not stay in that group. Many recent papers such as [3, 13, 16, 20, 21] have been devoted to the formulation of novel numerical methods that preserve the Lie-group structure of the exact solution. To understand why and how those methods work and what this paper is going to offer it is essential to know some more connections between a Lie group and its Lie algebra.

According to a classic result of Hausdorff [9] it is true that

$$y(t) = \mathrm{e}^{\sigma(t)} y_0, \qquad t \geq t_0, \tag{1.2}$$

where $\sigma : \mathbb{R} \mapsto \mathbf{g}$ is the solution of the initial value problem

$$\sigma' = \sum_{m=0}^{\infty} f_m \, \mathrm{ad}^m(a(t, y(t)), \sigma), \qquad t > t_0, \qquad \sigma(t_0) = O. \tag{1.3}$$

The coefficients $f_0, f_1, \ldots$ in the preceding equation are

$$f_0 = 1, \quad f_1 = B_1 + 1, \quad f_m = \frac{B_m}{m!}, \;\; m \geq 2$$

with $\{B_m\}_{m=0}^{\infty}$ being the *Bernoulli numbers*, while the *iterated commutator* ad^m for a pair of matrices p and q is defined recursively as

$$\mathrm{ad}^m(p, q) = \begin{cases} p, & m = 0 \\ [\,\mathrm{ad}^{m-1}(p, q), q], & m \in \mathbb{N} \end{cases}$$

with $[\cdot, \cdot]$ being the *commutator* (or *Lie bracket*) defined by

$$[p, q] = pq - qp.$$

If a is independent of the unknown function y then (1.3) can be reduced into an integral equation of the form

$$\sigma(t) = \sum_{m=0}^{\infty} f_m \int_0^t \mathrm{ad}^m(a(\tau), \sigma(\tau)) \, \mathrm{d}\tau, \qquad t \geq t_0.$$

The so called *Magnus expansion* [15]

$$\begin{aligned} \sigma(t) = & \int_0^t a(\kappa) \, \mathrm{d}\kappa + \frac{1}{2} \int_0^t \left[a(\kappa), \int_0^\kappa a(\xi) \, \mathrm{d}\xi \right] \mathrm{d}\kappa \\ & + \frac{1}{4} \int_0^t \left[a(\kappa), \int_0^\kappa \left[a(\xi), \int_0^\xi a(\eta) \, \mathrm{d}\eta \right] \mathrm{d}\xi \right] \mathrm{d}\kappa \\ & + \frac{1}{12} \int_0^t \left[\left[a(\kappa), \int_0^\kappa a(\xi) \, \mathrm{d}\xi \right], \int_0^\xi a(\eta) \, \mathrm{d}\eta \right] \mathrm{d}\kappa + \cdots \end{aligned} \tag{1.4}$$

can be obtained by iteration. The general form of the expansion (1.4) is obtained and its convergence is proved by Iserles and Nørsett [13] recently. Using the Magnus expansion and quadrature scheme in an innovative way they formulated some very efficient numerical methods.

One important reason for the success of Iserles and Nørsett's method [13] is that the initial value problem (1.3) instead of (1.1) is solved, noting that numerical solutions of (1.3) obtained from traditional numerical methods stay in the Lie algebra $\mathbf{g}$ since

1. Lie algebras are linear space; and

2. the result of an iterated commutation of two matrices in a Lie algebra stays in the Lie algebra.

Therefore, Lie-group invariant numerical solution of (1.1) can be obtained via (1.2).

In the nonlinear case, it is noted by Munthe-Kaas [16] that

$$\sigma' = \sum_{m=0}^{\infty} f_m \, \mathrm{ad}^m(a(t, \exp(\sigma(t))y_0), \sigma), \qquad t > t_0, \qquad \sigma(t_0) = O \tag{1.5}$$

after the substitution of (1.2) into (1.3). The method proposed by Munthe-Kaas consists of the application of a Runge–Kutta method to (1.3) (or a finite term truncation of (1.3)) and the use of (1.2). It should be pointed out that the paper of Munthe-Kaas [16] is on differential equations on homogeneous manifolds. Other Lie-group invariant numerical methods for solving (1.1) are formulated by Zanna [20, 21], based on Fer expansion [8] (or the numerical oriented method of iterated commutators [10]) and Magnus expansion [15]. The methods formulated by Munthe-Kaas [15] and Zanna [20, 21] keep the numerical solution of (1.1) in the Lie group. However, a shortcoming of their methods is that to advance one step (from the initial point t_0 to the next point $t_1 > t_0$) several matrix exponentiations are needed. For instance, order-three methods (see Section 4 of this paper) usually involve 3 or 4 matrix exponentiations per step. Higher order methods need more matrix exponentiations. The computations are usually dominated by these matrix exponentiations.

This paper is based on the observation that (1.1) is equivalent to the overdetermined system of differential-algebraic equations (1.1,1.2,1.3). The method to be proposed in the next section can be described as a prediction-projection-correction method. It consists of three stages per step. First, a standard Runge–Kutta method is applied to (1.1,1.3) to obtain an approximation $(\tilde{y}_1, \tilde{\sigma}_1)$ of the exact solution (y, σ) at $t = t_1$. Next, $\tilde{\sigma}_1$ is projected into a "nearby" matrix $\sigma_1 \in \mathbf{g}$, noting that in general $\tilde{\sigma}_1 \notin \mathbf{g}$ (nor is $\tilde{y}_1 \in \mathcal{G}$). Finally, a correction of $\tilde{y}_1$ is obtained by letting $y_1 = \exp(\sigma_1)y_0$. This method needs one matrix exponentiation per step regardless of its order of approximation. It is important to note that the projection is done in the Lie algebra instead of in the Lie group because Lie algebras are linear space but Lie groups are not in general. Since this prediction-projection-correction method is a straightforward adaptation of the standard Runge–Kutta method, local error estimation and stepsize control can be done by using embedded Runge–Kutta pairs. Two types of local error estimates are given in Section 3. In the final section, numerical examples are presented to demonstrate the advantage of the method proposed in this paper. At the end of this paper, we present numerical methods for isospectral flows, based on the idea of solving overdetermined systems of differential-algebraic equations.

2 Projected Runge–Kutta methods

Let $t_0 < t_1 < t_2 < \ldots$ be a partition of the integration interval $[t_0, T]$ for some $T > 0$. Assuming that we have $y_n \approx y(t_n)$, the usual practice is to use the standard s-stage

Runge–Kutta method defined by the Butcher tableau

$$\begin{array}{c|c} c & A \\ \hline & b^{\mathrm{T}} \end{array} \tag{2.1}$$

to (1.1) to obtain the following approximation of $y(t_{n+1})$:

$$\tilde{y}_{n+1} = y_n + h \sum_{i=1}^{s} b_i M_i, \tag{2.2}$$

where $h = t_{n+1} - t_n$ is the stepsize (for simplicity, we assume constant stepsize) and $M_1, \ldots, M_s$ are determined by the algebraic system

$$\begin{array}{l} M_i = a(t_n + c_i h, \Phi_i)\Phi_i, \\ \Phi_i = y_n + h \sum_{j=1}^{s} a_{ij} M_j, \end{array} \qquad i = 1, 2, \ldots, s. \tag{2.3}$$

The problem with the standard Runge–Kutta method is that $y_n \in \mathcal{G}$ does not guarantee $\tilde{y}_{n+1} \in \mathcal{G}$, despite the fact that $\tilde{y}_{n+1} \approx y(t_{n+1}) \in \mathcal{G}$ when $h << 1$. In order to obtain an approximation $y_{n+1} \in \mathcal{G}$ we recall from the previous section that

$$y(t) = \mathrm{e}^{\sigma(t)} y(t_n) \tag{2.4}$$

where $\sigma \in \mathbf{g}$ is the solution of the initial value problem

$$\sigma' = F(a(t, y(t)), \sigma), \qquad t > t_n, \qquad \sigma(t_n) = O, \tag{2.5}$$

where

$$F(a, \sigma) = \sum_{m=0}^{\infty} f_m \, \mathrm{ad}^m(a, \sigma).$$

Applying the Runge–Kutta method (2.1) to (2.5) yields the following approximation of $\sigma(t_{n+1})$:

$$\tilde{\sigma}_{n+1} = h \sum_{i=1}^{s} b_i K_i, \tag{2.6}$$

where $K_1, \ldots, K_s$ are determined by the algebraic system

$$\begin{array}{l} K_i = F(a(t_n + c_i h, \Phi_i), \Psi_i), \\ \Psi_i = h \sum_{j=1}^{s} a_{ij} K_j, \end{array} \qquad i = 1, 2, \ldots, s, \tag{2.7}$$

with $\Phi_1, \Phi_2, \ldots, \Phi_s$ being given by (2.3). Let

$$\sigma_{n+1} = \mathrm{P}(\tilde{\sigma}_{n+1}), \tag{2.8}$$

where P is a projection operator that maps $\tilde{\sigma}_{n+1}$ into a matrix which is in $\mathbf{g}$ as well as close to $\tilde{\sigma}_{n+1}$ up to the order of the Runge–Kutta method (2.1). Finally, the discrete analogue of (2.4)

$$y_{n+1} = \mathrm{e}^{\sigma_{n+1}} y_n. \tag{2.9}$$

is used to give an approximation to $y(t_{n+1})$. It is evident that $y_{n+1} \in \mathcal{G}$ as long as $y_n \in \mathcal{G}$. To summarize, we have the following result:

Theorem 1 *The method defined by (2.1,2.3,2.6–2.9) is*

1. *Lie-group invariant if the range of the projector* P *is* $\mathbf{g}$ *or a subset of* $\mathbf{g}$*; and*

2. *it is of order* $p \in \mathbb{N}$ *if the Runge–Kutta method* (2.1) *is of order* p *and the projector* P *obeys*

$$\tilde{\sigma} = \sigma + \mathcal{O}\big(h^{p+1}\big) \textit{ for some } \sigma \in \mathbf{g} \Longrightarrow P(\tilde{\sigma}) = \sigma + \mathcal{O}\big(h^{p+1}\big). \tag{2.10}$$

The projection operator P in the formula (2.8) needs further elaboration. If the function a is defined in such a way that it maps $\mathbb{R} \times \{$the set of $d \times d$ matrices$\}$ into $\mathbf{g}$ then it is always true that $\tilde{\sigma}_{n+1} \in \mathbf{g}$ and no projection is needed. Otherwise, we need to project $\tilde{\sigma}_{n+1}$ into a nearby matrix in the Lie group $\mathbf{g}$. Fortunately, this is not a difficult task since Lie algebras are linear space. Let L be a linear mapping such that

$$\mathrm{L}(\sigma) = 0 \text{ if and only if } \sigma \in \mathbf{g}.$$

For a matrix $\tilde{\sigma} \notin \mathbf{g}$, the projector P can be defined as

$$\mathrm{P}(\tilde{\sigma}) = \tilde{\sigma} + \delta,$$

where δ obeys

$$\mathrm{L}(\delta) = -\mathrm{L}(\tilde{\sigma}) \tag{2.11}$$

$$\|\delta\| \le M \|\mathrm{L}(\tilde{\sigma})\| \tag{2.12}$$

for some matrix norm $\|\cdot\|$ and constant M that is independent of $\tilde{\sigma}$. While (2.11) guarantees that $\tilde{\sigma} + \delta \in \mathbf{g}$, (2.12) ensures that (2.10) is satisfied, noting that

$$\|\mathrm{L}(\tilde{\sigma})\| \le \|\mathrm{L}(\tilde{\sigma}) - \mathrm{L}(\sigma)\| \le \|\mathrm{L}\| \cdot \|\tilde{\sigma} - \sigma\|, \quad \forall \sigma \in \mathbf{g}.$$

It should be noted that δ need not be unique since L is necessarily a singular mapping unless $\mathbf{g}$ contains zero matrix only. For some special cases, the projector P is much easier to define than (2.11,2.12) may suggest. Consider the following well-known examples of matrix Lie groups

1. the *special orthogonal group* $\mathrm{SO}(d, \mathbb{R})$ of $d \times d$ orthogonal matrices with unit determinant;

2. the *special linear group* $\mathrm{SL}(d, \mathbb{R})$ of $d \times d$ nonsingular matrices with unit determinant; and

3. the *symplectic group* $\mathrm{Sp}(2d, \mathbb{R})$ of $2d \times 2d$ matrices S obeying

$$S^{\mathrm{T}} J S = J,$$

where

$$J = \begin{bmatrix} O_d & I_d \\ -I_d & O_d \end{bmatrix};$$

The corresponding Lie algebras are

1. $\mathrm{so}(d,\mathbb{R})$, the set of $d \times d$ skew-symmetric matrices;
2. $\mathrm{sl}(d,\mathbb{R})$, the set of $d \times d$ matrices with zero trace; and
3. $\mathrm{sp}(2d,\mathbb{R})$, the set of $2d \times 2d$ matrices σ satisfying

$$\sigma^{\mathrm{T}} J + J\sigma = O.$$

In the case of $\mathrm{so}(d,\mathbb{R})$ the projector P can be defined as

$$\mathrm{P}(\sigma) = \frac{1}{2}(\sigma - \sigma^{\mathrm{T}}).$$

In the case of $\mathrm{sl}(d,\mathbb{R})$, P can be defined as

$$\mathrm{P}(\sigma) = \sigma - \mathrm{diag}\,(\delta, \ldots, \delta)\,,$$

where

$$\delta = \frac{1}{d}\sum_{i=1}^{d} \sigma_{ii}.$$

In the case of $\mathrm{sp}(2d,\mathbb{R})$ we recall from [19] that a matrix $\tilde{\sigma}$ is in the Lie algebra $\mathrm{sp}(2d,\mathbb{R})$ if and only if $\tilde{\sigma} = JS$ for some symmetric $2d \times 2d$ matrix S. Therefore, the corresponding projector P can be defined as

$$\mathrm{P}(\sigma) = \frac{1}{2}(\sigma + J\sigma^{\mathrm{T}} J),$$

noting that

$$J\frac{J^{-1}\sigma + (J^{-1}\sigma)^{\mathrm{T}}}{2} = \frac{1}{2}(\sigma + J\sigma^{\mathrm{T}} J).$$

It is obvious that (2.10) and (2.12) are satisfied for these three cases.

The first few coefficients of the infinite series in (2.5) are

$$\begin{aligned} &f_0 = 1, \quad f_1 = -\tfrac{1}{2}, \quad f_2 = \tfrac{1}{12}, \quad f_4 = -\tfrac{1}{720}, \quad f_6 = -\tfrac{1}{30240}, \\ &f_8 = -\tfrac{1}{1209600}, \quad f_{10} = \tfrac{1}{47900160}, \quad f_{12} = -\tfrac{1}{1307674368000}. \end{aligned}$$

In fact, the infinite series in (2.5) converges very fast if $\|a\| \cdot \|\sigma\| < \pi$, because

$$f_{2m} \approx (-1)^{m-1}\frac{2}{(2\pi)^{2m}} \qquad \text{as } m \to \infty$$

and $f_{2m+1} = 0$ for $m \in \mathbb{N}$. Therefore, it is all right to replace (2.5) by its truncated version

$$\sigma' = F_k(a(t, y(t)), \sigma), \qquad t > t_n, \qquad \sigma(t_n) = O, \tag{2.13}$$

where $k \in \mathbb{N}$ and

$$F_k(a, \sigma) = \sum_{m=0}^{k} f_m \,\mathrm{ad}^m(a, \sigma).$$

A reasonable choice of k is $p-2$ with p being the order of the Runge–Kutta method (2.1), because the solution of (2.5) obeys (see, e.g., [10, 13])

$$\left\{\begin{array}{l}\sigma(t)=a(t,y(t))(t-t_n)+\mathcal{O}\big((t-t_n)^2\big)\\ \mathrm{ad}^m[a(t,y(t)),\sigma(t)]=\mathcal{O}\big((t-t_n)^{m+1}\big)\end{array}\right. \quad \text{as } t\to t_n.$$

As long as $k\geq p-2$, replacing (2.5) by (2.13) does not affect the order of the approximation, nor does it change the group invariance of the numerical solution. Furthermore, there is no need to do the projection (2.8) if the function a maps $\mathbb{R}\times\{\text{the set of } d\times d \text{ matrices}\}$ into $\mathbf{g}$. This is because the iterated commutator $\mathrm{ad}^m(a(t,y(t)),\sigma)$ stays in g as long as $a,\sigma\in\mathbf{g}$.

3 Local error estimation

Since the standard Runge–Kutta methods are used in the previous section, the standard local error estimation and stepsize control strategy can be used as well. Let us consider the embedded Runge–Kutta pair:

$$\begin{array}{c|c} c & A \\ \hline & b^{\mathrm{T}} \\ & \hat{b}^{\mathrm{T}} \end{array} \tag{3.1}$$

where the method defined by c, A and b^{T} is of order p and the one defined by c, A and $\hat{b}^{\mathrm{T}}$ is of order $p+1$. Applying these two methods to the initial value problem

$$y'(t)=f(t,y), \qquad t>t_0, \qquad y(t_0)=y_0,$$

yield two approximations of $y(t_n)$:

$$y_{n+1}=y_n+h\sum_{i=1}^{s}b_iM_i, \tag{3.2}$$

$$\hat{y}_{n+1}=y_n+h\sum_{i=1}^{s}\hat{b}_iM_i, \tag{3.3}$$

where $M_1,\ldots,M_s$ are determined by the algebraic system

$$M_i=f(t_n+c_ih,y_n+h\sum_{j=1}^{s}a_{ij}M_j), \qquad i=1,2,\ldots,s.$$

The local error can be defined as

$$E_n=h\sum_{i=1}^{s}(b_i-\hat{b}_i)M_i.$$

Both (3.2) and (3.3) can be used as the numerical solution. The choice is often depend on which one is *error turned* [14]. One popular embedded Runge–Kutta method is

Fehlberg's **RKF45**:

$$
\begin{array}{c|cccccc}
0 & & & & & & \\
\frac{1}{4} & \frac{1}{4} & & & & & \\
\frac{3}{8} & \frac{3}{32} & \frac{9}{32} & & & & \\
\frac{12}{13} & \frac{1932}{2197} & -\frac{7200}{2197} & \frac{7296}{2197} & & & \\
1 & \frac{439}{216} & -8 & \frac{3680}{513} & -\frac{845}{4104} & & \\
\frac{1}{2} & -\frac{8}{27} & 2 & -\frac{3544}{2565} & \frac{1859}{4104} & -\frac{11}{40} & \\
\hline
 & \frac{25}{216} & 0 & \frac{1408}{2565} & \frac{2197}{4104} & -\frac{1}{5} & 0 \\
 & \frac{16}{135} & 0 & \frac{6656}{12825} & \frac{28561}{56430} & -\frac{9}{50} & \frac{2}{55}
\end{array}
\tag{3.4}
$$

which uses (3.2) as the numerical solution. Another (probably more popular) one is Dormand and Prince's **DOPRI(5,4)**:

$$
\begin{array}{c|ccccccc}
0 & & & & & & & \\
\frac{1}{5} & \frac{1}{5} & & & & & & \\
\frac{3}{10} & \frac{3}{40} & \frac{9}{40} & & & & & \\
\frac{4}{5} & \frac{44}{45} & -\frac{56}{15} & \frac{32}{9} & & & & \\
\frac{8}{9} & \frac{19372}{6561} & -\frac{25360}{2187} & \frac{64448}{6561} & -\frac{212}{729} & & & \\
1 & \frac{9017}{3168} & -\frac{355}{33} & \frac{46732}{5247} & \frac{49}{176} & -\frac{5103}{18656} & & \\
1 & \frac{35}{384} & 0 & \frac{500}{1113} & \frac{125}{192} & -\frac{2187}{6787} & \frac{11}{84} & \\
\hline
 & \frac{5179}{57600} & 0 & \frac{7571}{16695} & \frac{393}{640} & -\frac{92097}{339200} & \frac{187}{2100} & \frac{1}{40} \\
 & \frac{35}{384} & 0 & \frac{500}{1113} & \frac{125}{192} & -\frac{2187}{6784} & \frac{11}{84} & 0
\end{array}
\tag{3.5}
$$

which uses (3.3) as the numerical solution.

The application of the Runge–Kutta pair (3.1) to (1.1,1.3) provides us with $\tilde{y}_{n+1}$ and $\hat{\tilde{y}}_{n+1}$, σ_{n+1} and $\hat{\sigma}_{n+1}$. Depending on the stepsize control strategy we are going use, only two of them are needed. In the case of (2.13) we must choose $k \geq p-1$, since the method defined by c, A and $\hat{b}^{\mathrm{T}}$ is of order $p+1$. Local error can be estimated by the formula

$$
E_n = \begin{cases} \|y_{n+1} - \hat{\tilde{y}}_{n+1}\|, & \text{if } y_{n+1} = \mathrm{e}^{\sigma_{n+1}} y_n, \\ \|y_{n+1} - \tilde{y}_{n+1}\|, & \text{if } y_{n+1} = \mathrm{e}^{\hat{\sigma}_{n+1}} y_n. \end{cases} \tag{3.6}
$$

Alternatively, the following can be used as the local error estimate

$$
E_n = \|\sigma_{n+1} - \hat{\sigma}_{n+1}\| \cdot \|y_n\|, \tag{3.7}
$$

since

$$
\left\|\mathrm{e}^{\sigma_{n+1}} y_n - \mathrm{e}^{\hat{\sigma}_{n+1}} y_n\right\| \leq \left\|\mathrm{e}^{\sigma_{n+1}} - \mathrm{e}^{\hat{\sigma}_{n+1}}\right\| \|y_n\| \approx \|\sigma_{n+1} - \hat{\sigma}_{n+1}\| \cdot \|y_n\|.
$$

However, it tends to underestimate the error, possible due to the fact that $\sigma(t_n) = O$. This is demonstrated by numerical examples in the next section. In principle, one can use $\left\|\mathrm{e}^{\sigma_{n+1}} y_n - \mathrm{e}^{\hat{\sigma}_{n+1}} y_n\right\|$ to estimate the locale error. The problem is that it needs an extra matrix exponentiation per step.

4 Numerical examples

In this section some numerical examples are presented to demonstrate the superiority of the Lie-group invariant method (2.1,2.3,2.6–2.9).

First, we show by an example that methods that preserve the Lie group structure are better than the conversional methods as far as qualitative behaviour of the solution is concerned. The Hamiltonian system (nonlinear pendulum) corresponding to the Hamiltonian

$$H(y_1, y_2) = y_1^2/2 + y_2^2(1/2 - 10^{-2}\cos y_2)$$

can be written as

$$y'(t) = a(t, y(t))y(t), \tag{4.1}$$

where $y(t) = (y_1(t), y_2(t))^{\mathrm{T}}$ and

$$a(t,y) = \begin{pmatrix} 0 & 10^{-2}(2\cos y_2 - y_2 \sin y_2) - 1 \\ 1 & 0 \end{pmatrix} : \ \mathbb{R}\times\mathbb{R}^2 \mapsto \mathrm{so}(2,\mathbb{R}) \cap \mathrm{sp}(2,\mathbb{R}).$$

Corresponding to the classical Runge-Kutta method (**RK4**)

$$\begin{array}{c|cccc} 0 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ 1 & 0 & 0 & 1 & \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array} \tag{4.2}$$

we have the Lie-group invariant method method (4.2,2.3,2.6–2.9) which we call **RK4P** ("P" for projection). Let the initial values be the points on the unit circle centred at 6. In Figures 1 and 2 the numerical solutions obtained by using **RK4** and **RK4P**, respectively, with constant stepsize $h = \pi/2$ are plotted on the phase space (y_1, y_2) for $t = 0$ and $t \approx 4, 8, 12, 16, 20$. Neither **RK4** nor **PK4P** is symplectic for Hamiltonian systems. However, it can be seen from Figure 2 that the area of the initial unit disc is perfectly preserved as the solutions by **PK4P** evolves, while Figure 1 shows that the area shrinks in the case of **RK4**. Note that in the case of **RK4P** the numerical solution y_{n+1} is y_n multiplied on the left by a symplectic matrix. Further research on solving Hamiltonian systems by symplectic group invariant method is currently under investigation.

We now compare our new method (2.1,2.3,2.6–2.9) with some recently developed Lie-group methods that are based on the order-three explicit Runge–Kutta method due to **Kutta**

$$\begin{array}{c|ccc} 0 & & & \\ \frac{1}{2} & \frac{1}{2} & & \\ 1 & -1 & 2 & \\ \hline & \frac{1}{6} & \frac{2}{3} & \frac{1}{6} \end{array} \tag{4.3}$$

The methods

$$\begin{aligned} Y_1 &= y_n, & A_1 &= a(t_n, Y_1),\\ Y_2 &= \exp(\tfrac{h}{2}A_1)y_n, & A_2 &= a(t_n + \tfrac{h}{2}, Y_2),\\ Y_3 &= \exp(h(-A_1 + 2A_2))y_n, & A_3 &= a(t_n + h, Y_3),\end{aligned}$$

$$y_{n+1} = \exp\left(h\left(\frac{1}{6}A_1 + \frac{2}{3}A_2 + \frac{1}{6}A_3\right)\right)\exp\left(-\frac{h^2}{2}[A_1 - A_3, \frac{2}{15}A_2 + \frac{1}{30}A_3]\right)y_n \tag{4.4}$$

and

$$\begin{aligned} Y_1 &= y_n, & A_1 &= a(t_n, Y_1),\\ Y_2 &= \exp(\tfrac{h}{2}A_1)y_n, & A_2 &= a(t_n + \tfrac{h}{2}, Y_2),\\ Y_3 &= \exp(h(-A_1 + 2A_2))y_n, & A_3 &= a(t_n + h, Y_3),\end{aligned}$$

$$y_{n+1} = \exp\left(h\left(\frac{1}{6}A_1 + \frac{2}{3}A_2 + \frac{1}{6}A_3\right) - \frac{h^2}{2}[A_1 - A_3, \frac{2}{15}A_2 + \frac{1}{30}A_3]\right)y_n \tag{4.5}$$

are formulated by Zanna [21] based on the Fer expansion and the Magnus expansion, respectively. The method proposed by Munthe-Kaas [16] is of the form

$$\begin{aligned} \Psi_1 &= O, & K_1 &= a(t_n, y_n),\\ \Psi_2 &= \tfrac{h}{2}K_1, & K_2 &= F_1(a(t_n + \tfrac{h}{2}, \exp(\Psi_2)y_n), \Psi_2),\\ \Psi_3 &= h(-K_1 + 2K_2), & K_3 &= F_1(a(t_n + h, \exp(\Psi_3)y_n), \Psi_3),\end{aligned}$$

$$y_{n+1} = \exp\left(h\left(\frac{1}{6}A_1 + \frac{2}{3}A_2 + \frac{1}{6}A_3\right)\right)y_n. \tag{4.6}$$

We call the method (4.3,2.3,2.6–2.9) **KuttaP**. These four Lie-group invariant methods are of order 3. To test their efficiency and accuracy, we use the following two examples

$$a(t,y) = \begin{bmatrix} 0 & y_1 & & & & \\ -y_1 & 0 & y_2 & & & \\ & -y_2 & 0 & y_3 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -y_{d-2} & 0 & y_{d-1}\\ & & & & -y_{d-1} & 0 \end{bmatrix}, \tag{4.7}$$

$$a(t,y) = 2(y - y^{\mathrm{T}}). \tag{4.8}$$

The symbols in Figures 3–6 are as follows:

x-mark ×	Zanna's (4.4)
plus +	Zanna's (4.5)
circle ∘	Munthe-Kaas' (4.6)
star ∗	**KuttaP**
point ·	**Kutta**

Equation (1.1) is solved 10 times (the initial values are random orthogonal matrices generated by QR factorization) by each of the five order-three methods from $t = 0$ to 1 with stepsize $h = 0.2$. The results are plotted in Figures 3 (for the case (4.7)) and 4 (for the case (4.8)), where the x-axis and y-axis represent the matrix size d and $\frac{T}{d}$, respectively, with T being the total CPU time. It is evident that the cost of exponentiation of matrices dominates the four Lie-group invariant methods. **KuttaP** needs one matrix exponentiation per step, while Zanna's (4.5) needs four, Zanna's (4.6) and Munthe-Kaas' (4.6) needs 3 exponentiation per step. However, this does not mean **KuttaP** is 3 or 4 times cheaper. In fact, **KattaP** is about 2 times cheaper only. To account for this, let us consider Zanna's (4.4). The four matrix exponentiations are

$$\begin{aligned}
&\exp(\frac{h}{2}A_1) \approx \exp(\frac{h}{2}a),\\
&\exp(h(-A_1 + 2A_2)) \approx \exp(ha),\\
&\exp\left(h\left(\frac{1}{6}A_1 + \frac{2}{3}A_2 + \frac{1}{6}A_3\right)\right) \approx \exp(ha),\\
&\exp\left(-\frac{h^2}{2}[A_1 - A_3, \frac{2}{15}A_2 + \frac{1}{30}A_3]\right) \approx \exp(-\frac{1}{12}h^2[A_1 - A_3, a]),
\end{aligned}$$

where $a = a(t_n, y_n)$. The first and the last matrix exponentiations are relatively cheap to computer. Therefore, the cost of (4.4) is dominated by the computation of the second and third matrix exponentiations. This partly explains why Zanna's (4.4) is about twice as expensive as **KuttaP**.

Accuracy of those five order-three methods varies. In general, Zanna's (4.4) and (4.5), Munthe-Kaas' (4.6) and **KuttaP** are more accurate than **Kutta**. In Figures 5 and 6, we plot the errors of the numerical solutions of (4.7) by those five methods with constant stepsize $h = 0.1$ and the initial orthogonal matrices

$$y_0 = \begin{bmatrix} -0.8513 & 0.4089 & 0.1883 & 0.2582 & -0.0765 \\ -0.4054 & -0.6465 & -0.5412 & -0.0226 & -0.3525 \\ -0.2977 & 0.0326 & -0.1182 & -0.7945 & 0.5149 \\ -0.0070 & -0.0629 & 0.5365 & -0.5124 & -0.6676 \\ -0.1488 & -0.6402 & 0.6081 & 0.1976 & 0.3989 \end{bmatrix} \tag{4.9}$$

and

$$y_0 = \begin{bmatrix} -0.6164 & 0.1720 & -0.1993 & 0.1921 & 0.7168 \\ -0.0153 & -0.7925 & -0.0236 & 0.6091 & 0.0072 \\ -0.6484 & 0.0685 & -0.3235 & 0.0684 & -0.6823 \\ -0.4440 & -0.2564 & 0.7991 & -0.3136 & -0.0141 \\ -0.0470 & -0.5214 & -0.4654 & -0.6993 & 0.1427 \end{bmatrix} \tag{4.10}$$

respectively. The errors are obtained by comparing the numerical solutions with the "exact solution" given by **RK4** with stepsize $h = 0.01$.

Finally, we use the initial value problem [13]

$$y'' + ty = 0, \qquad t > 0, \qquad y(0) = 1,\ y'(0) = 0 \tag{4.11}$$

to test the local error estimation (3.6) and (3.7). Letting $x = y'$, we rewrite (4.11) as

$$\begin{bmatrix} x \\ y \end{bmatrix}' = a(t) \begin{bmatrix} x \\ y \end{bmatrix} \qquad \begin{bmatrix} x(0) \\ y(0) \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix},$$

where

$$a(t) = \begin{bmatrix} 0 & -t \\ 1 & 0 \end{bmatrix} : \mathbb{R} \mapsto \mathrm{so}(2, \mathbb{R}) \cap \mathrm{sp}(2, \mathbb{R}).$$

In the case of **DOPRI(5,4)P**, i.e., the method (3.5,2.3,2.6–2.9), the following simple strategy is used to control the stepsize $h_n = t_{n+1} - t_n$:

$$h_{n+1} = \begin{cases} \frac{2}{3} h_n, & \text{if } E_n > \mathrm{Tol}, \\ \frac{3}{2} h_n, & \text{if } E_n < \mathrm{Tol}/10, \\ h_n, & \text{otherwise} \end{cases}$$

where Tol denotes the tolerance. In Figures 7–10, the subplots on the upper-left, upper-right, lower-left and lower-right are for (t, y), (t, y'), (y, y') and (t_n, h_n), respectively. It can be seen from Figures 7 and 9 that (3.7) underestimate the error, a phenomena that is confirmed by a similar experiment with **RKF45P**, namely, the method (3.4,2.3,2.6–2.9). We remark in passing that **RKF45P** is not as accurate as **DOPRI(5,4)P** in this case.

The numerical experiment of this section are conducted on a Sun Sparc-10 using **MATLAB** Version 5.0. The exponentiation of matrices are obtained using the **expm** function of **MATLAB**. Celledoni and Iserles [6] have recently developed some efficient methods for approximating the exponential from a Lie algebra to a Lie group.

Before concluding this paper, we point out that the idea of solving the overdetermined system of differential-algebraic equations (1.1,1.2,1.3) can be used in other situations. For instance, the isospectral flow

$$L' = [B(t, L), L], \;\; t \geq t_0, \qquad L(0) = L_0 \tag{4.12}$$

studied extensively by Calvo, Iserles and Zanna [1, 2, 3, 4, 5] can be treated as the overdetermined system

$$L' = [B(t, L), L], \;\; t \geq t_0, \qquad L(0) = L_0, \tag{4.13}$$

$$U' = B(t, L)U, \;\; t \geq t_0, \qquad U(0) = I, \tag{4.14}$$

$$L(t) = U(t) L_0 U(t)^{-1}.$$

Subsequently, isospectral numerical solution of (4.12) can be obtained in the following way:

$$M_i = [B(t_n + c_i h, \Phi_i), \Phi_i], \;\; \Phi_i = L_n + h \sum_{j=1}^{s} a_{ij} M_j, \quad i = 1, \ldots, s, \tag{4.15}$$

$$K_i = B(t_n + c_i h, \Phi_i)(I + h \sum_{j=1}^{s} a_{ij} K_j), \quad i = 1, \ldots, s, \tag{4.16}$$

$$U_{n+1} = I + h\sum_{i=1}^{s} b_i K_i, \tag{4.17}$$

$$L_{n+1} = U_{n+1} L_n U_{n+1}^{-1},$$

where $h = t_{n+1} - t_n$ and $n = 0, 1, \ldots$.

In the case $B \in \mathrm{so}(d, \mathbb{R})$, the solution U stays in the Lie group $\mathrm{SO}(d, \mathbb{R})$. Solving the overdetermined system

$$L' = [B(t, L), L], \quad t \geq t_0, \qquad L(0) = L_0$$

$$\sigma' = \sum_{m=0}^{\infty} f_m \operatorname{ad}^m(B(t, L), \sigma), \quad t \geq t_0, \qquad \sigma(t_0) = O,$$

$$U(t) = \mathrm{e}^{\sigma(t)}$$

$$L(t) = U(t) L_0 U(t)^{\mathrm{T}}$$

can keep the numerical solution of U in $\mathrm{SO}(d, \mathbb{R})$, in addition to reproducing the isospectrality of the solution L. More specifically, we have

$$M_i = [B(t_n + c_i h, \Phi_i), \Phi_i], \qquad \Phi_i = L_n + h\textstyle\sum_{j=1}^{s} a_{ij} M_j, \qquad i = 1, \ldots, s,$$
$$\Psi_i = \textstyle\sum_{m=0}^{\infty} f_m \operatorname{ad}^m(B(t_n + c_i, \Phi_i), h\sum_{j=1}^{s} a_{ij}\Psi_j), \qquad i = 1, \ldots, s,$$
$$U_{n+1} = \exp\left(h\textstyle\sum_{i=1}^{s} b_i \Psi_i\right),$$
$$L_{n+1} = U_{n+1} L_n U_{n+1}^{\mathrm{T}},$$

for $n = 0, 1, \ldots$. Alternatively, if the corresponding Runge-Kutta method (2.1) satisfies

$$b_i a_{ij} - b_j a_{ji} = b_i b_j, \qquad i, j = 1, 2, \ldots, s, \tag{4.18}$$

then according to a result of Dieci, Russell and van Vleck [7] the numerical solution U_{n+1} given by (4.17) stays in $\mathrm{SO}(d, \mathbb{R})$. A Runge-Kutta method satisfying (4.18) is necessarily implicit. However, in the case of the isospectral flow, it is the linear system (4.16), not the nonlinear system (4.15), that has to be solved exactly (or upto machine epsilon) in order to have $U_{n+1} \in SO(d, \mathbb{R})$.

Acknowledgements

The author thanks Arieh Iserles, Per Christian Moan and Brynjulf Owren for many useful discussions. It is Arieh who characterized the essence of the method presented in this paper as solving an overdetermined system.

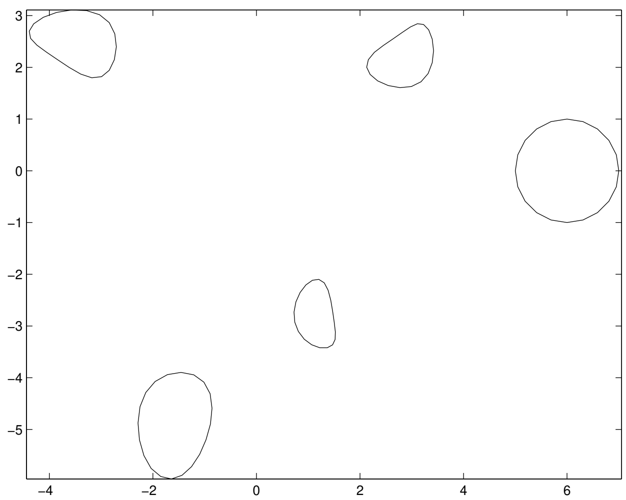

Figure 1. Numerical solutions of (4.1) by **RK4** on the phase space

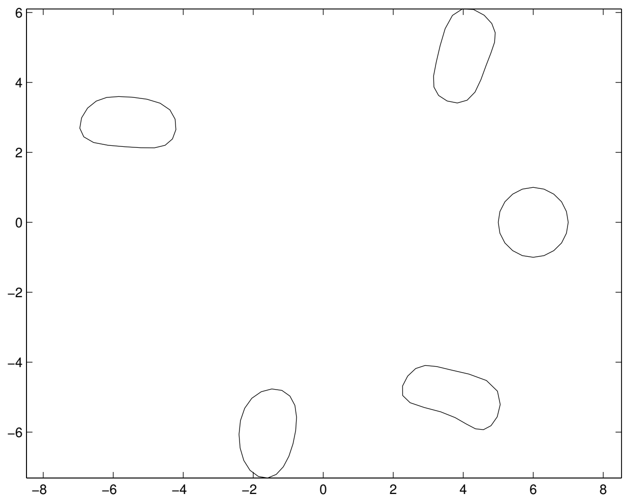

Figure 2. Numerical solutions of (4.1) by **RK4P** on the phase space

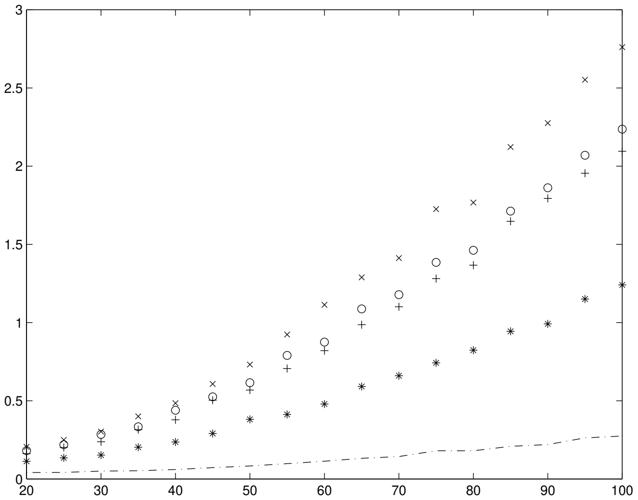

Figure 3. Efficiency in the case (4.7)

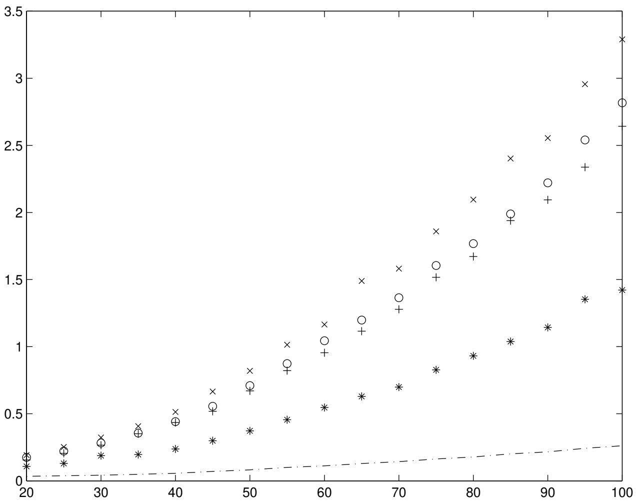

Figure 4. Efficiency in the case (4.8)

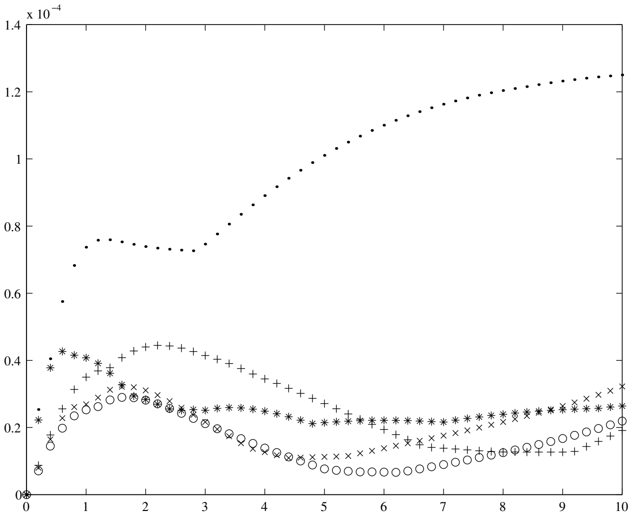

Figure 5. Errors in the case (4.7) with the initial matrix (4.9)

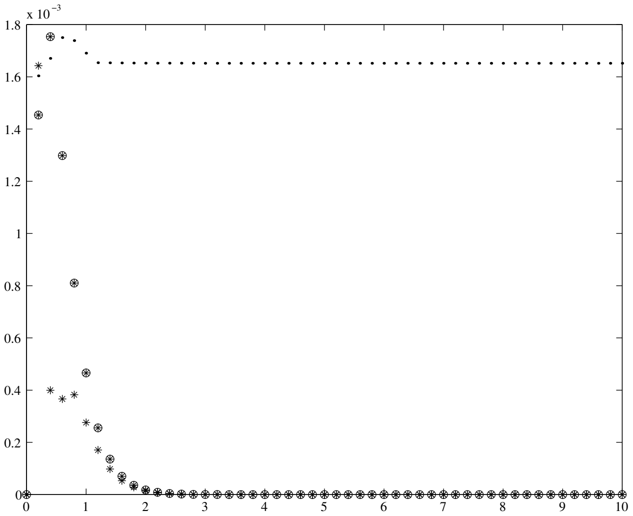

Figure 6. Errors in the case (4.7) with the initial matrix (4.10)

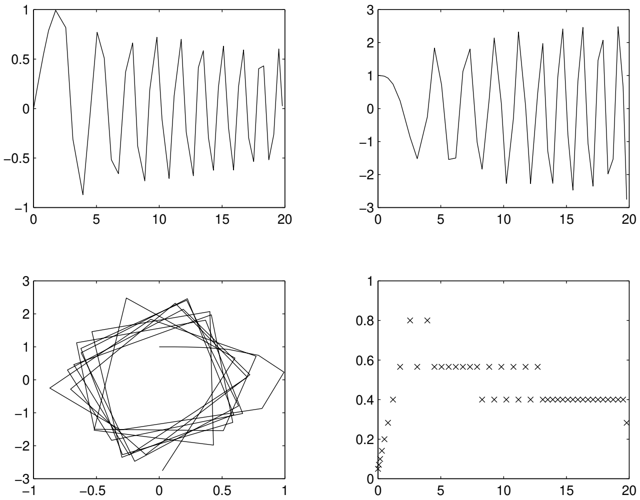

Figure 7. Stepsize control for **DOPRI(5,4)P** using (3.7) and Tol= 5×10^{-4}

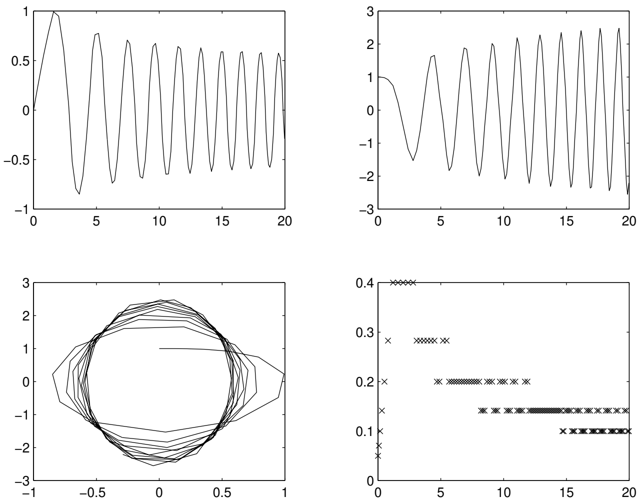

Figure 8. Stepsize control for **DOPRI(5,4)P** using (3.6), Tol= 5×10^{-4}

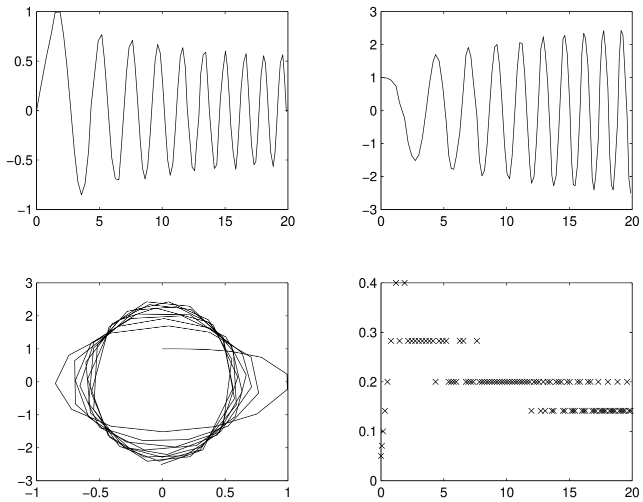

Figure 9. Stepsize control for **DOPRI(5,4)P** using (3.7), Tol= 5×10^{-5}

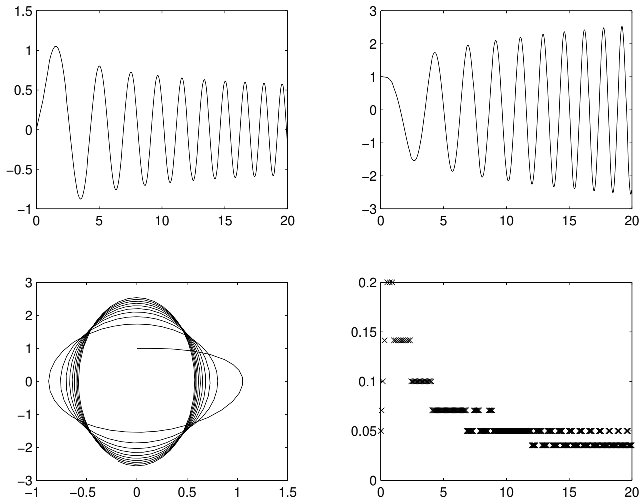

Figure 10. Stepsize control for **DOPRI(5,4)P** using (3.6), Tol= 5×10^{-5}

References

[1] M.P. Calvo, A. Iserles and A. Zanna, Runge-Kutta methods on manifolds, in *Numerical Analysis: A.R. Mitchell's 75th Birthday Volume* (G.A. Watson & D.F. Griffiths, eds), World Scientific, Singapore (1996), 57–70.

[2] M.P. Calvo, A. Iserles and A. Zanna, Conservative methods for the Toda lattice equations, Cambridge University Tech. Rep. DAMTP, 1996/NA18.

[3] M.P. Calvo, A. Iserles and A. Zanna, Runge–Kutta methods for orthogonal and isospectral flows, *Applied Numer. Maths* **22** (1996), 153–163.

[4] M.P. Calvo, A. Iserles and A. Zanna, Semi-explicit methods for isospectral flows, Cambridge University Tech. Rep. DAMTP, 1997/NA16.

[5] M.P. Calvo, A. Iserles and A. Zanna, Numerical solution of isospectral flows, *Math. Comp.* **66** (1997), 1461–1486.

[6] E. Celledoni and A. Iserles, Approximating the exponential from a Lie algebra to a Lie group, Cambridge University Tech. Rep. DAMTP, 1998/NA03.

[7] L. Dieci, D.R. Russell and E.S. van Vleck, Unitary integrator and applications to continuous orthonormalization techniques, *SIAM J. Numer. Anal.* **31** (1994), 261–281.

[8] F. Fer, Résolution del l'equation matricielle $\dot{U} = pU$ par produit infini d'exponentielles matricielles', *Bull. Classe des Sci. Acad. Royal Belg.* **44** (1958), 818–829.

[9] F. Hausdorff, Die symbolische exponentialformel in der gruppentheorie, *Berichte der Sšchsischen Akademie der Wissenschaften (Math. Phys. Klasse* **58** (1906), 19–48.

[10] A. Iserles, Solving linear ordinary differential equations by exponentials of iterated commutators, *Numer. Math.* **45** (1984), 183–199.

[11] A. Iserles, Numerical methods on (and off) manifolds, in *Foundations of Computational Mathematics* (F. Cucker & M. Shub, eds), Springer-Verlag, New York (1997), 180–189.

[12] A. Iserles, Multistep methods on manifolds, Cambridge University Tech. Rep. DAMTP, 1997/NA13.

[13] A. Iserles and S.P. Nørsett, On the solution of linear differential equations in Lie groups, Cambridge University Tech. Rep. DAMTP, 1997/NA3.

[14] J.D. Lambert, *Numerical Methods for Ordinary Differential Systems*, John Wiley & Sons, 1991.

[15] W. Magnus, On the exponential solution of differential equations for a linear operator, *Comm. Pure & Appl. Maths* **7** (1954), 649–673.

[16] H. Munthe-Kaas, High order Runge–Kutta methods on manifolds, Cambridge University Tech. Rep. DAMTP, 1997/NA14.

[17] P.J. Olver, *Applications of Lie Groups to Differential Equations*, **GTM 107**, Springer-Verlag, 1993.

[18] D.H. Sattinger and O.L. Weaver, *Lie Groups and Algebras with Applications to Physics, Geometry, and Mechanics*, **AMS 61**, Springer-Verlag, 1986.

[19] A.M. Stuart and A.R. Humphries, *Dynamical Systems and Numerical Analysis*, Cambridge University Press, 1996.

[20] A. Zanna, The method of iterated commutators for ordinary differential equations on Lie groups, Cambridge University Tech. Rep. DAMTP, 1996/NA12.

[21] A. Zanna, Collocation and relaxed collocation for the Fer and the Magnus expansions, Cambridge University Tech. Rep. DAMTP, 1997/NA17.

Technique reports of the numerical analysis group, Cambridge University, are available at **http://www.damtp.cam.ac.uk/user/na/reports.html**