On the implementation of the method of Magnus series for linear differential equations

A. Iserles* A. Marthinsen† S. P. Nørsett†

Abstract

The method of Magnus series has recently been analysed by Iserles & Nørsett (1997). It approximates the solution of linear differential equations $y' = a(t)y$ in the form $y(t) = e^{\sigma(t)}$, solving a nonlinear differential equation for σ by means of an expansion in iterated integrals of commutators. An appealing feature of the method is that, whenever the exact solution evolves in a Lie group, so does the numerical solution.

The subject matter of the present paper is practical implementation of the method of Magnus series. We commence by briefly reviewing the method and highlighting its connection with graph theory. This is followed by the derivation of error estimates, a task greatly assisted by the graph-theoretical connection. These error estimates have been incorporated into a variable-step fourth-order code. The concluding section of the paper is devoted to a number of computer experiments that highlight the promise of the proposed approach even in the absence of a Lie-group structure.

1 Introduction

The subject matter of this paper is the solution of linear equations by the method of Magnus series and, in particular, the practical issues of error control and step-size selection in the implementation of the aforementioned technique.

It has been known since at least the turn of the century that the solution of $y' = a(t)y$, $y(0) = y_0$ (where $a(t)$ might be a matrix or a linear operator) can locally be represented in the form $y(t) = e^{\sigma(t)}y_0$, where σ obeys a certain nonlinear differential equation. An important advantage of this representation is apparent when $y_0 \in G$, where G is a Lie group, while $a(t) : \mathbb{R}^+ \to \mathfrak{g}$, where $\mathfrak{g}$ is the Lie algebra of G. In that case $\sigma(t) \in \mathfrak{g}$ for all $t \geq 0$, therefore $y(t) \in G$, $t \geq 0$. This is an important qualitative feature of the solution and its retention under discretization is sometimes essential and often desirable.

*Department of Applied Mathematics and Theoretical Physics, University of Cambridge, Cambridge CB3 9EW, England.

†Department of Mathematical Sciences, Norwegian University of Science and Technology, N-7034 Trondheim, Norway.

Unlike G, the set $\mathfrak{g}$ is a linear space, therefore it is a more suitable framework for discretization. Solving for $\sigma \in \mathfrak{g}$ provides a promising avenue toward the recovery of a Lie-group structure.

The solution of σ has already been considered by a number of authors. In particular, Magnus (1954) demonstrated that σ can be represented as a specific expansion in integrals of commutators, but he neither proved convergence nor elucidated a general recursive tool for the derivation of this expansion. Both tasks have recently been accomplished by Iserles & Nørsett (1997).

The purpose of the present paper is to pursue a number of computational issues that arise from (Iserles & Nørsett 1997) and, in particular, to shed light on error control in a practical implementation of Magnus series. There are three sources of numerical error in the method and each requires separate treatment:

1. Truncation of the infinite expansion;

2. Discretization of multivariate integrals; and

3. The approximation of a matrix exponential.

This paper addresses itself to the first two sources of error and our underlying assumption is that the exponential is evaluated correctly to machine accuracy. The reason for this assumption is that the familiar rational approximants to the exponential in general fail to map $\mathfrak{g}$ into G. New approximation methods are under intensive investigation and it is our hope that, in fullness of time, they will prove themselves effective in approximating the exponential without straying off the Lie group G. Yet, for the time being, we concentrate on numerical errors committed in the Lie algebra $\mathfrak{g}$.

In Section 2 we briefly review the method of *Magnus series,* in particular paying attention to its algorithmic aspects. Section 3 is concerned with the estimation of numerical errors in the truncation of the infinite expansion and in the replacement of multivariate integrals by quadrature. Finally, in Section 4 we present a number of computer experiments. Our numerical results affirm that the error control mechanism is effective. However, perhaps their most striking consequence is that, in line with (Iserles & Nørsett 1997), they indicate that the method of Magnus series is competitive with classical schemes even when the conservation of Lie structure is not at issue.

2 The method of Magnus series

2.1 The Magnus expansion

Iserles & Nørsett (1997) have recently considered the solution of the matrix differential equation

$$y'(t) = a(t)y(t), \quad t \geq 0, \tag{2.1}$$

with initial condition $y(0) = y_0 \in G$, where G is a Lie group with Lie algebra $\mathfrak{g}$ and $a : \mathbb{R}^+ \to \mathfrak{g}$. In that case it is well known that $y(t) \in G$ for all $t \geq 0$, a qualitative feature that, arguably, should be retained under discretization or approximation. They developed a general technique, called the *Magnus series method*, which can be employed as either a numerical or a perturbative solution scheme of (2.1). As Magnus (1954) noticed, the analytical solution of (2.1) can be written in the form

$$y(t) = \mathrm{e}^{\sigma(t)} y_0, \qquad t \geq 0,$$

where $\sigma : \mathbb{R}^+ \to \mathfrak{g}$, is an infinite sum of elements in $\mathfrak{g}$,

$$\begin{aligned} \sigma(t) &= \int_0^t a(\kappa)\,\mathrm{d}\kappa + \frac{1}{2}\int_0^t \left[a(\kappa), \int_0^\kappa a(\xi)\,\mathrm{d}\xi\right]\mathrm{d}\kappa \\ &+\frac{1}{4}\int_0^t \left[a(\kappa), \int_0^\kappa \left[a(\xi), \int_0^\xi a(\eta)\,\mathrm{d}\eta\right]\mathrm{d}\xi\right]\mathrm{d}\kappa \\ &+\frac{1}{12}\int_0^t \left[\left[a(\kappa), \int_0^\kappa a(\xi)\,\mathrm{d}\xi\right], \int_0^\kappa a(\eta)\,\mathrm{d}\eta\right]\mathrm{d}\kappa + \cdots . \end{aligned} \tag{2.2}$$

This function is, as shown by Hausdorff (1906), the solution of the implicit differential equation

$$\sum_{k=0}^{\infty} \frac{(-1)^k}{(k+1)!}\mathrm{ad}^k(\sigma', \sigma) = a, \qquad t \geq 0, \tag{2.3}$$

with $\sigma(0) = 0$ and

$$\mathrm{ad}^k(p,q) = \begin{cases} p, & k = 0, \\ [\mathrm{ad}^{k-1}(p,q), q], & k \geq 1. \end{cases}$$

To arrive at (2.2), Magnus recognized that (2.3) can be transformed to the explicit form

$$\sigma' = a + \tfrac{1}{2}\mathrm{ad}(a,\sigma) + \sum_{k=1}^{\infty} \frac{B_{2k}}{(2k)!}\mathrm{ad}^{2k}(a,\sigma),$$

where B_m, $m = 0, 1, \ldots$, are Bernoulli numbers. The idea behind Magnus series methods is to truncate (2.2) and evaluate the terms in a clever and computationally efficient manner.

Iserles & Nørsett (1997) presented a general device that identifies expansion terms in (2.2) with rooted trees, thereby allowing for their recursive evaluation and analysis. Before embarking on a brief survey of the theory in (Iserles & Nørsett 1997), we note for future reference that the first few terms of the Magnus expansion of σ' can be written in the form

$$
\begin{aligned}
\sigma'(t) = {} & f_0 a(t) + f_0 f_1 \int_0^t [a(t), a(\xi)] \,\mathrm{d}\xi \\
& + f_0^2 f_2 \int_0^t \int_0^t \left[[a(t), a(\xi)], a(\eta) \right] \mathrm{d}\eta \,\mathrm{d}\xi \\
& + f_0 f_1^2 \int_0^t \int_0^\xi \left[a(t), [a(\xi), a(\eta)] \right] \mathrm{d}\eta \,\mathrm{d}\xi \\
& + f_0^2 f_1 f_2 \int_0^t \int_0^\xi \int_0^\xi \Big[a(t), \left[[a(\xi), a(\eta)], a(\rho) \right] \Big] \mathrm{d}\rho \,\mathrm{d}\eta \,\mathrm{d}\xi \\
& + f_0 f_1^3 \int_0^t \int_0^\xi \int_0^\eta \Big[a(t), \left[a(\xi), [a(\eta), a(\rho)] \right] \Big] \mathrm{d}\rho \,\mathrm{d}\eta \,\mathrm{d}\xi \\
& + f_0^2 f_1 f_2 \int_0^t \int_0^\xi \int_0^t \Big[\left[a(t), [a(\xi), a(\eta)] \right], a(\rho) \Big] \mathrm{d}\rho \,\mathrm{d}\eta \,\mathrm{d}\xi \\
& + f_0^2 f_1 f_2 \int_0^t \int_0^t \int_0^\eta \left[[a(t), a(\xi)], [a(\eta), a(\rho)] \right] \mathrm{d}\rho \,\mathrm{d}\eta \,\mathrm{d}\xi \\
& + f_0^3 f_3 \int_0^t \int_0^t \int_0^t \Big[\left[[a(t), a(\xi)], a(\eta) \right], a(\rho) \Big] \mathrm{d}\rho \,\mathrm{d}\eta \,\mathrm{d}\xi + \cdots,
\end{aligned}
\tag{2.4}
$$

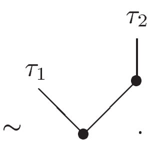
where f_m, $m = 0, 1, \ldots$, are the coefficients in the expansion

$$
\begin{aligned}
f(z) = \sum_{m=0}^{\infty} f_m z^m &= \sum_{m=0}^{\infty} \frac{B_m}{m!} z^m + z \\
&= 1 + \tfrac{1}{2} z + \tfrac{1}{12} z^2 - \tfrac{1}{720} z^4 + \tfrac{1}{30420} z^6 - \tfrac{1}{1209600} z^8 + \cdots .
\end{aligned}
$$

The association between expansion terms in (2.4) and rooted trees is as follows. The function $a(t)$ is associated with the trivial order-one tree, a relationship which is denoted by

$$a \sim \bullet .$$

Given two expansion terms, $q_1(t) \sim \tau_1$ and $q_2(t) \sim \tau_2$, we associate

$$\left[q_1(t), \textstyle\int_0^t q_2(\kappa)\,\mathrm{d}\kappa \right] \sim$$ 

.

It has been shown in (Iserles & Nørsett 1997) that all the expansion terms in (2.4) can be obtained in this fashion. Association between the first terms in (2.4) and trees is displayed in Table 1.

A term with m integrals corresponds to a tree with $3m+1$ vertices (cf. Table 1) and it is possible to prove that, for $m \geq 1$, it is of the order of magnitude $\mathcal{O}(t^{m+1})$ for every sufficiently smooth function a. Recall that we are expanding σ'. To recover σ requires another integration and the order of magnitude increases to $\mathcal{O}(t^{m+2})$. Let

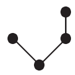
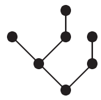
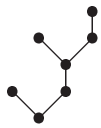
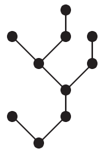
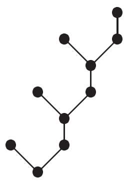
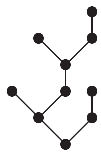
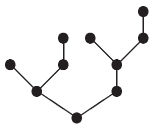
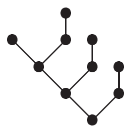
Expansion term	Order of magnitude	Associated tree	Compact notation
a	$\mathcal{O}(1)$		τ
$[a(t), \int_0^t a(\xi)]$	$\mathcal{O}(t^2)$		$[\tau, [\tau]]$
$[[a(t), \int_0^t a(\xi)], \int_0^t a(\eta)]$	$\mathcal{O}(t^3)$		$[[\tau, [\tau]], [\tau]]$
$[a(t), [\int_0^t a(\xi), \int_0^\xi a(\eta)]]$	$\mathcal{O}(t^3)$		$[\tau, [[\tau, [\tau]]]]$
$[a(t), [[\int_0^t a(\xi), \int_0^\xi a(\eta)], \int_0^\xi a(\rho)]]$	$\mathcal{O}(t^4)$		$[\tau, [[[\tau, [\tau]], [\tau]]]]$
$[a(t), \int_0^t [a(\xi), \int_0^\xi [a(\eta), \int_0^\eta a(\rho)]]]$	$\mathcal{O}(t^4)$		$[\tau, [[\tau, [[\tau, [\tau]]]]]]$
$[[a(t), \int_0^t [a(\xi), \int_0^\xi a(\eta)]], \int_0^t a(\rho)]$	$\mathcal{O}(t^4)$		$[[\tau, [[\tau, [\tau]]]], [\tau]]$
$[[a(t), \int_0^t a(\xi)], \int_0^t [a(\eta), \int_0^\eta a(\rho)]]$	$\mathcal{O}(t^6)$		$[[\tau, [\tau]], [[\tau, [\tau]]]]$
$[[[a(t), \int_0^t a(\xi)], \int_0^t a(\eta)], \int_0^t a(\rho)]$	$\mathcal{O}(t^4)$		$[[[\tau, [\tau]], [\tau]], [\tau]]$

Table 1: Expansion terms, their orders of magnitude and the associated trees.

σ_m denote a truncated Magnus expansion, that excludes terms that correspond to trees with number of vertices exceeding $3m+1$. It follows that

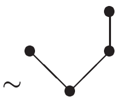
$$\mathrm{e}^{\sigma_m(t)} y_0 = y(t) + \mathcal{O}\big(t^{m+2}\big)$$

and we obtain an order-$(m+1)$ numerical scheme (Iserles & Nørsett 1997).

As a matter of fact, terms corresponding to trees with $3m+1$ vertices might be of higher order of magnitude, the eighth term in Table 1 being a case in point. It is possible to use graph theory in a more subtle manner, to identify exactly the order of magnitude of each term and this will be a subject of a forthcoming paper.

When implementing tree-manipulation routines in a symbolic computer algebra program it is convenient to use a variant of the Butcher tree notation (Butcher 1987). We again associate $a(t)$ with the single-vertex tree, denoted by τ. A bracket with a single argument denotes integration, while a bracket with two arguments denotes commutation of the arguments. Thus, for example,

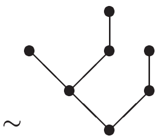
$$a(t) \sim \bullet \sim \tau, \qquad \left[a(t), \textstyle\int_0^t a(\kappa)\,\mathrm{d}\kappa\right] \sim$$



$$\sim [\tau, [\tau]]$$

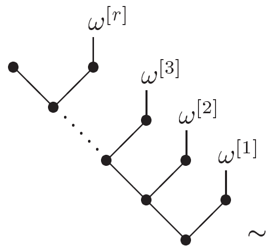
and

$$\left[\left[a(t), \textstyle\int_0^t a(\kappa)\,\mathrm{d}\kappa\right], \textstyle\int_0^t a(\eta)\,\mathrm{d}\eta\right] \sim$$



$$\sim [[\tau, [\tau]], [\tau]].$$

The coefficients $f_0, f_0 f_1, f_0^2 f_2, f_0 f_1^2 \ldots$ in (2.4) can be computed recursively, once the association between expansion terms and trees has been established. To this end, and also to assist in symbolic manipulation, we say that a *standard form* of a tree is the unique representation



$$\sim \left[\left[\left[\cdots\left[\tau, \left[\omega^{[r]}\right]\right], \cdots, \left[\omega^{[3]}\right]\right], \left[\omega^{[2]}\right]\right], \left[\omega^{[1]}\right]\right], \tag{2.5}$$

where $\omega^{[1]}, \omega^{[2]}, \ldots, \omega^{[r]}$ are lower-order trees.

The Magnus expansion of σ is

$$\sigma(t) = \sum_{k=0}^{\infty} \sum_{\omega \in \mathcal{T}_{3k+1}} \alpha(\omega) \int_0^t H_\omega(\kappa)\,\mathrm{d}\kappa, \tag{2.6}$$

where $\mathcal{T}_m$ is the set of all trees with m vertices, $H_\omega \sim \omega$ and the coefficients α_τ have been obtained by means of the standard form (2.5),

$$\alpha(\tau) = 1,$$
$$\alpha(\omega) = f_r \prod_{i=1}^{r} \alpha(\omega^{[i]}).$$

2.2 Multivariate quadrature

Occasionally, e.g. when the components of $a(t)$ are polynomials, it is possible to evaluate all the necessary integrals in (2.6) explicitly, preferably by a symbolic algebra programme. However, in most realistic cases practical implementation of Magnus series requires the replacement of integrals by quadrature.

In general, performing multivariate quadrature is costly and requires an excessive number of function evaluations. This task is further aggravated by the need, evident in (2.4), to approximate a large number of integrals in different simplices of increasing dimension. It is fortunate, therefore, that the special nature of the integrands and the domains of integration allows for very considerable savings (Iserles & Nørsett 1997). As a matter of fact, an order-p method requires just $\lfloor (p+1)/2 \rfloor$ function evaluations, which can be reused in all the quadrature formulae!

All the integrals in (2.6) need be evaluated over simplices of the form $h\mathcal{S}$, where

$$\mathcal{S} = \left\{ (t_1, t_2, \ldots, t_m) \,:\, 0 \le t_j \le t_{i_j}, j = 1, 2, \ldots, m \right\},$$

where $t_0 = 1$ and $i_j \in \{1, 2, \ldots, j\}$, $j = 1, 2, \ldots, m$. For example,

$$\begin{aligned}
I_1 &= \int_0^h a(t_1)\,\mathrm{d}t_1 : & \mathcal{S}_1 &= \{0 \le t_1 \le 1\},\\
I_2 &= \int_0^h \int_0^{t_1} [a(t_1), a(t_2)]\,\mathrm{d}t_2\,\mathrm{d}t_1 : & \mathcal{S}_2 &= \{0 \le t_1 \le 1,\ 0 \le t_2 \le t_1\},\\
I_3 &= \int_0^h \int_0^{t_1} \int_0^{t_1} [[a(t_1), a(t_2)], a(t_3)]\,\mathrm{d}t_3\,\mathrm{d}t_2\,\mathrm{d}t_1 : & \mathcal{S}_3 &= \{0 \le t_1 \le 1,\ 0 \le t_2, t_3 \le t_1\},\\
I_4 &= \int_0^h \int_0^{t_1} \int_0^{t_2} [a(t_1), [a(t_2), a(t_3)]]\,\mathrm{d}t_3\,\mathrm{d}t_2\,\mathrm{d}t_1 : & \mathcal{S}_4 &= \{0 \le t_1 \le 1,\ 0 \le t_2 \le t_1,\\
& & & \qquad 0 \le t_3 \le t_2\}.
\end{aligned}$$

Moreover, the integrand is, in each case, a function of the form

$$\mathcal{L}\big(a(t_1), a(t_2), \ldots, a(t_m)\big),$$

where $\mathcal{L}$ is multilinear. It has been proposed in (Iserles & Nørsett 1997) to use the quadrature formula

$$\int_{h\mathcal{S}} \mathcal{L}\big(a(t_1), \ldots, a(t_m)\big)\,\mathrm{d}t_m \cdots \mathrm{d}t_1 \approx h^m \sum_{\boldsymbol{l} \in C_m^\nu} b_{\boldsymbol{l}} \mathcal{L}\big(a(hc_{l_1}), a(hc_{l_2}), \ldots, a(hc_{l_\nu})\big), \tag{2.7}$$

where $c_1, c_2, \ldots, c_\nu$ are distinct points in $[0,1]$ and C_m^ν is the set of all the combinations of m-tuples from the set $\{1, 2, \ldots, \nu\}$. The weights $b_{\boldsymbol{l}}$ can be evaluated explicitly by the formula

$$b_{\boldsymbol{l}} = \int_{\mathcal{S}} \prod_{i=1}^{m} \lambda_i(t_{l_i})\,\mathrm{d}t_m \cdots \mathrm{d}t_1, \qquad \boldsymbol{l} \in C_m^\nu,$$

where $\lambda_k \in \mathbb{P}_{\nu-1}[t]$ is the kth cardinal polynomial of Lagrange interpolation at the nodes $c_1, c_2, \ldots, c_\nu$.

The usefulness of the quadrature formula (2.7) is underscored by a theorem from (Iserles & Nørsett 1997). Suppose that

$$\int_0^1 t^{k-1} c(t)\,\mathrm{d}t = 0, \qquad k = 1, 2, \ldots, s,$$

where

$$c(t) = \prod_{i=1}^{\nu} (t - c_i).$$

Then the quadrature formula (2.7) is of order $\nu + s$. In other words, the order of the above multivariate quadrature is *exactly* the same as of the classical univariate quadrature with the same nodes. In other words, choosing $c_1, c_2, \ldots, c_\nu$ as Gauss–Legendre quadrature points in $[0,1]$ results in order 2ν in (2.7) for *all* integrals necessary for the evaluation of truncated Magnus series.

As an example, let

$$a_1 = a\left(\left(\tfrac{1}{2} - \tfrac{\sqrt{3}}{6}\right)h\right), \qquad a_2 = a\left(\left(\tfrac{1}{2} + \tfrac{\sqrt{3}}{6}\right)h\right)$$

be the nodes of the fourth-order Gauss-Legendre quadrature in $[0,1]$, whence we obtain

$$\begin{aligned}
I_1 &\approx \tfrac{1}{2} h (a_1 + a_2), \\
I_2 &\approx \tfrac{\sqrt{3}}{6} h^2 [a_2, a_1], \\
I_3 &\approx h^3 \left[[a_2, a_1], \left(\tfrac{3}{80} + \tfrac{\sqrt{3}}{16}\right) a_1 - \left(\tfrac{3}{80} - \tfrac{\sqrt{3}}{16}\right) a_2 \right] \\
I_4 &\approx -h^3 \left[\left(\tfrac{3}{80} - \tfrac{\sqrt{3}}{48}\right) a_1 - \left(\tfrac{3}{80} + \tfrac{\sqrt{3}}{48}\right) a_2, [a_2, a_1] \right].
\end{aligned} \tag{2.8}$$

Although the quadrature formula (2.7) leads to remarkable savings in the number of function evaluations, it might result in considerable cost of linear algebra, since the number of terms in the sum behaves like ν^m and the computation of each such term requires $m-1$ commutators. However, very considerable reduction in the expense of linear algebra takes place when the special structure of the Lie algebra $\mathfrak{g}$ is taken into account. We refer the reader to (Iserles & Nørsett 1997) and to the forthcoming paper of Munthe-Kaas & Owren (n.d.) for details.

To conclude this section, we combine the Magnus expansion (2.6) with the aforementioned multivariate quadrature to present a fourth-order method for the solution of the differential equation (2.1),

$$\begin{aligned} a_1 &= a\left(t_n + (\tfrac{1}{2} - \tfrac{\sqrt{3}}{6})h\right), \qquad a_2 = a\left(t_n + (\tfrac{1}{2} + \tfrac{\sqrt{3}}{6})h\right), \\ \sigma^n &= \tfrac{1}{2}h(a_1 + a_2) + \tfrac{\sqrt{3}}{12}h^2[a_2, a_1], \\ y_{n+1} &= \mathrm{e}^{\sigma^n} y_n. \end{aligned} \tag{2.9}$$

In Section 4 the method (2.9) is designated as **MG4,** to distinguish it from the other two methods therein.

The method (2.9) differs from the fourth-order method in (Iserles & Nørsett 1997), since the latter has an extra term,

$$\sigma^n = \tfrac{1}{2}h(a_1 + a_2) + \tfrac{\sqrt{3}}{12}h^2[a_2, a_1] + \tfrac{1}{80}h^3[a_2 - a_1, [a_2, a_1]].$$

On the face of it, the extra term is required, since it caters for the $\mathcal{O}(h^4)$ terms in the Magnus expansion. More careful analysis, however, reveals that it can be discarded with impunity. This is a special case of a result which is discussed in (Iserles, Nørsett & Rasmussen n.d.) in a more general framework. Here we justify (2.9) by two straightforward and alternative observations. Firstly, it is easy to ascertain directly that $h^3[a_2 - a_1, [a_2, a_1]] = \mathcal{O}(h^5)$ for every sufficiently smooth matrix a. Secondly, it will be evident by the techniques of Section 3 that although, according to Table 1, there are two trees which are $\mathcal{O}(h^4)$ (and which contribute to the extra term), their linear combination in the Magnus expansion is $\mathcal{O}(h^5)$.

3 Error control in Magnus series methods

Let us assume that both G and $\mathfrak{g}$ are embedded in an Euclidean space, hence affording meaning to mathematical operations that combine elements from both sets.

3.1 Truncating Magnus series

The first step in converting (2.6) into a numerical scheme is its truncation,

$$\sigma_p(t) = \sum_{k=0}^{p-1} \sum_{\omega \in \mathcal{T}_{3k+1}} \alpha(\omega) \int_0^t H_\omega(\kappa)\,\mathrm{d}\kappa, \tag{3.1}$$

say, where $p \in \mathbb{N}$. It follows from the discussion in Section 2 that

$$\sigma_p(t) = \sigma(t) + \varepsilon t^{p+1} + \mathcal{O}(t^{p+2}), \tag{3.2}$$

where $\varepsilon \in \mathfrak{g}$. Therefore

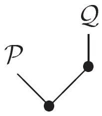
$$\mathrm{e}^{\sigma_p(h)} y_0 = \mathrm{e}^{\sigma(h)}(\mathrm{Id} + \varepsilon h^{p+1}) y_0 + \mathcal{O}(h^{p+2}) = y(h) + \varepsilon y_0 h^{p+1} + \mathcal{O}(h^{p+2}).$$

We deduce that $\varepsilon y_0 h^{p+1}$ represents the local contribution to the truncation error of the method.

In order to derive the error constant ε in (3.2), we associate each term $\int_0^t H(t)\,\mathrm{d}t$ with a rooted tree. The procedure is identical to that of Section 2, except that we are adding a root to denote extra integration – recall that the association between expansion terms and trees in (Iserles & Nørsett 1997) refers to σ' while at present we are dealing with σ. Therefore all the trees under consideration have $3m+2$ vertices for some $m \in \mathbb{Z}^+$. Let us suppose that the function a is sufficiently smooth and consider its Taylor expansion

$$a(t) = \alpha + \beta t + \cdots .$$

Let $H \sim \mathcal{H}$, where

$\mathcal{H} =$ 

and $P \sim \mathcal{P}$ and $Q \sim \mathcal{Q}$. If $P(t) = p_0 t^k + p_1 t^{k+1} + \cdots$ and $Q(t) = q_0 t^l + q_1 t^{l+1} + \cdots$, it follows at once that

$$\begin{aligned} H(t) &= \int_0^t [P(t), Q(\kappa)]\,\mathrm{d}\kappa \\ &= \frac{1}{l+1}[p_0, q_0] t^{k+l+1} + \left(\frac{1}{l+1}[p_1, q_0] + \frac{1}{l+2}[p_0, q_1] \right) t^{k+l+2} + \cdots . \end{aligned}$$

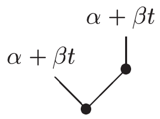
In other words, for $t \to 0$

$$H(t) \approx \frac{1}{l+1}[p_0, q_0] t^{k+l+1},$$

unless $[p_0, q_0] = 0$, in which case

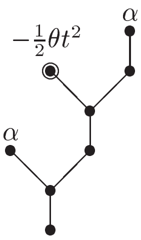
$$H(t) \approx \left(\frac{1}{l+1}[p_1, q_0] + \frac{1}{l+2}[p_0, q_1] \right) t^{k+l+2}.$$

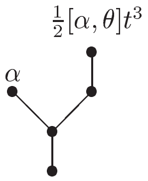
Given $H \sim \mathcal{H}$, we can thus obtain its Taylor expansion (and, more in line with the purpose of the present section, the leading term in its Taylor expansion) by enumerating each 'top' vertex (that is, a vertex with no children) by the expansion of the function a and pruning the tree in accordance with the above construction. Letting $\theta = [\alpha, \beta]$, we first observe that the leading term of $\left[a(t), \int_0^t a(\kappa)\right] \mathrm{d}\kappa$ is



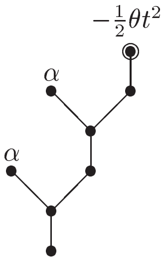
$$\Rightarrow \quad -\frac{1}{2}\theta t^2.$$

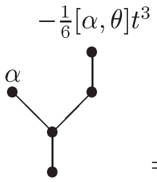
As a notational convention, we henceforth replace the subtree associated with $\left[a(t), \int_0^t a(\kappa)\right] \mathrm{d}\kappa$ by a fat vertex. It is apparent from Table 1 that there are five trees with eleven vertices (recall that we have added a root to each tree) and these are the

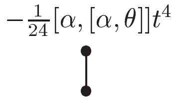


$\Rightarrow$


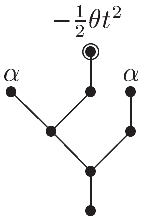
$\Rightarrow \quad \frac{1}{40}[\alpha,[\alpha,\theta]]t^5,$

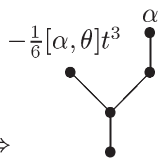


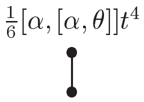
$\Rightarrow$


$\Rightarrow$


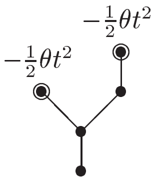
$\Rightarrow \quad -\frac{1}{120}[\alpha,[\alpha,\theta]]t^5,$

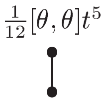


$\Rightarrow$


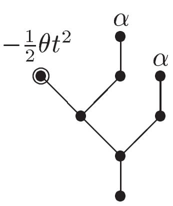
$\Rightarrow$


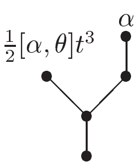
$\Rightarrow \quad \frac{1}{30}[\alpha,[\alpha,\theta]]t^5,$

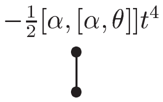


$\Rightarrow$


$\Rightarrow \quad \mathcal{O}(t^7),$



$\Rightarrow$


$\Rightarrow$


$\Rightarrow \quad -\frac{1}{10}[\alpha,[\alpha,\theta]]t^5.$

Table 2: Leading expansion terms of order-11 trees.

candidates for the error term in an order-4 method. Their leading expansion terms are displayed in Table 2.

Note that one of the five trees with eleven vertices results in an $\mathcal{O}(t^7)$ perturbation, hence need not be considered in our error estimate. This is in line with the estimate in Table 1. As a matter of fact, the phenomenon whereby the order of magnitude of a term is larger (often, much larger) than the order of the associated tree is increasingly prevalent as the order increases. Its analysis will be a subject of a forthcoming paper (Iserles et al. n.d.).

Another observation from Table 2 is that all the leading error coefficients are scalar multiples of the same element $[\alpha, [\alpha, \theta]] \in \mathfrak{g}$. This can be explained by the analysis in (Iserles & Nørsett 1997) – in fact, a similar argument can be extended to higher orders where the number of distinct leading error coefficients is considerably smaller than the number of expansion terms.

In a practical estimation of a truncation error we time-step from t_n to $t_{n+1} = t_n + h$, where $h > 0$, and wish to estimate ε at t_{n+1}. The expansion being about t_{n+1}, we may estimate $a'(t_{n+1})$ by a finite difference,

$$a(t) = a(t_{n+1}) + \frac{a(t_{n+1}) - a(t_n)}{h}(t - t_{n+1}) + \mathcal{O}\big((t - t_{n+1})^2\big) .$$

Therefore

$$\theta \approx \frac{1}{h}[a(t_{n+1}), a(t_{n+1}) - a(t_n)] = -\frac{1}{h}[a(t_{n+1}), a(t_n)]$$

and

$$[\alpha, [\alpha, \theta]]h^5 \approx [a(t_{n+1}), [a(t_{n+1}), [a(t_n), a(t_{n+1})]]]h^4.$$

According to (2.4), the coefficients in the linear combination of the terms from Table 2 are $\frac{1}{24}, \frac{1}{8}, \frac{1}{24}, \frac{1}{24}$ and 0 respectively. Note, thus, that the last two terms do not contribute to our error bound: the penultimate because of its own order of magnitude, $\mathcal{O}(t^7)$, and the last because $f_3 = 0$ means that it does not feature at all in the expansion. We deduce that the scalar coefficient is $\frac{1}{24} \times \frac{1}{40} - \frac{1}{8} \times \frac{1}{120} + \frac{1}{24} \times \frac{1}{30}$ and our estimate of the truncation error is

$$\frac{1}{720}[a(t_{n+1}), [a(t_{n+1}), [a(t_n), a(t_{n+1})]]]h^4. \tag{3.3}$$

We have already mentioned that, in general, (3.3) represents just one of the two components that need be taken into account in our error estimate. However, in two important instances the quadrature error is nil. Firstly, whenever the entries of $a(t)$ are simple enough (e.g., polynomial, exponential, trigonometric), it is possible to evaluate all the requisite integrals exactly, possibly with the help of a symbolic algebra package. Secondly, as long as the entries of $a(t)$ are cubic, all our fourth-order integrals are exact.

3.2 The quadrature error

We restrict the discussion in this subsection to the fourth-order method (2.9), designated in the next section as **MG4**. This restriction of generality somewhat simplifies the discussion but it is easy to extend our error bounds to methods of different orders.

Our interest lies in the leading error coefficient in

$$\tilde{\sigma}_4(t) - \sigma(t),$$

where $\tilde{\sigma}_p(t)$ is the outcome of replacing all the integrals in $\sigma(t)$ by the pth order quadrature (2.7). Alternatively, we may set in the step t_{n+1}

$$\sigma_4^\star(t) = \sigma_4(t) - \frac{1}{720}[a(t_{n+1}), [a(t_{n+1}), [a(t_n), a(t_{n+1})]]]t^4,$$

the *truncation correction* of the fourth-order method (2.9) (which itself approximates σ to order five), let $\tilde{\sigma}_4^\star(t)$ stand for $\sigma_4^\star(t)$ with all integrals replaced by the above quadratures and consider the leading error coefficient in

$$\tilde{\sigma}_4^\star(t) - \sigma_4^\star(t).$$

The advantage of the second formulation is that it demonstrates that the quadrature error depends just on the four integrals that are actually discretized in (2.9), while the contribution of all the other integrals is subsumed in the truncation error. Letting μ denote the leading error coefficient, $\tilde{\sigma}_4^\star(t) - \sigma_4^\star(t) = \mu t^5 + \mathcal{O}(t^6)$, we thus deduce that

$$\mu = \mu_1 + \tfrac{1}{2}\mu_2 + \tfrac{1}{12}\mu_3 + \tfrac{1}{4}\mu_4,$$

where each μ_k is the error constant incurred in the quadrature of the integral I_k in (2.8). The coefficients in the linear combination follow from (2.4).

Estimation of the error in quadrature formulae is a notoriously difficult problem (Cools 1997, Davis & Rabinowitz 1984). The obvious recourse, to use a small number of extra function evaluations to estimate the error, similarly to the technique of embedded Runge–Kutta, say, is impossible. We need ν function evaluations for a univariate Gaussian quadrature of order 2ν, but further $\nu+1$ evaluations are necessary to embed it in a higher-order scheme .... Instead, we have opted for a different method of error estimation, expressing the error in terms of derivatives and approximating the latter by finite differences.

Assuming again, without loss of generality, that $t_n = 0$, we let

$$a(t) = \alpha + \beta t + \gamma t^2 + \delta t^3 + \eta t^4 + \mathcal{O}(t^5).$$

Expanding integrals and quadrature formulae into Taylor series, we obtain

$$\begin{aligned}
\mu_1 &= -\tfrac{1}{180}\eta \\
\mu_2 &= \tfrac{1}{240}[\alpha, \delta] + \tfrac{1}{1080}[\beta, \gamma] \\
\mu_3 &= \tfrac{1}{1080}\left\{[\alpha, [\alpha, \gamma]] + [\beta, [\alpha, \beta]]\right\} \\
\mu_4 &= -\tfrac{1}{270}\left[\alpha, [\alpha, \gamma]\right] - \tfrac{1}{720}\left[\beta, [\alpha, \beta]\right].
\end{aligned}$$

Therefore,

$$\mu = -\tfrac{1}{180}\eta + \tfrac{1}{480}[\alpha, \delta] + \tfrac{1}{2160}[\beta, \gamma] - \tfrac{11}{12960}[\alpha, [\alpha, \gamma]] - \tfrac{7}{25920}[\beta, [\alpha, \beta]]. \qquad (3.4)$$

The coefficients β, γ, δ and η can be approximated as follows. We construct a polynomial of sufficiently large degree that interpolates $a(t)$ in $\mathfrak{g}$. Coefficients of the expansion can then be evaluated explicitly by differentiating the polynomial.

Based on five points in $\mathfrak{g}$, $a_1 = a(t_1), \ldots, a_5 = a(t_5)$, say, we may construct a polynomial $p(t) \in \mathfrak{g}$ of degree four. By differentiating this polynomial and evaluating at $t = \tilde{t} = t_1$, we obtain the coefficients:

$$\alpha = p(\tilde{t}), \quad \beta = p^{(1)}(\tilde{t}), \quad \gamma = \tfrac{1}{2!}p^{(2)}(\tilde{t}), \quad \delta = \tfrac{1}{3!}p^{(3)}(\tilde{t}), \quad \text{and} \quad \eta = \tfrac{1}{4!}p^{(4)}(\tilde{t}).$$

By employing standard divided differences, we compute

$$p_4(t) = a_1 + (t - t_1)f[t_1, t_2] + \cdots + \prod_{j=1}^{4}(t - t_j)f[t_1, t_2, \ldots, t_5] + \mathcal{O}(t^5), \tag{3.5}$$

which is the unique polynomial in $\mathbb{P}_4$ that satisfies the interpolation conditions

$$p(t_i) = a(t_i), \qquad i = 1, \ldots, 5,$$

where, as usual,

$$f[t_1, \ldots, t_n] = \sum_{k=1}^{n} \frac{a(t_k)}{\prod_{\substack{j=1 \\ j \neq k}} (t_k - t_j)}.$$

Now, by ignoring higher order terms, we obtain

$$\begin{aligned}
p_4^{(1)}(t) &= f[t_1, t_2] + f[t_1, t_2, t_3] \sum_{i=1}^{2}(t - t_i) + f[t_1, t_2, t_3, t_4] \sum_{i=2}^{3} \sum_{j=1}^{i-1} (t - t_i)(t - t_j) \\
&\quad + f[t_1, t_2, t_3, t_4, t_5] \sum_{i=3}^{4} \sum_{j=2}^{i-1} \sum_{k=1}^{j-1} (t - t_i)(t - t_j)(t - t_k), \\
\tfrac{1}{2!}p_4^{(2)}(t) &= f[t_1, t_2, t_3] + f[t_1, t_2, t_3, t_4] \sum_{i=1}^{3}(t - t_i) \\
&\quad + f[t_1, t_2, t_3, t_4, t_5] \sum_{i=2}^{4} \sum_{j=1}^{i-1} (t - t_i)(t - t_j), \\
\tfrac{1}{3!}p_4^{(3)}(t) &= f[t_1, t_2, t_3, t_4] + f[t_1, t_2, t_3, t_4, t_5] \sum_{i=1}^{4}(t - t_i), \\
\tfrac{1}{4!}p_4^{(4)}(t) &= f[t_1, t_2, t_3, t_4, t_5].
\end{aligned}$$

These expressions, together with (3.5), give $\alpha, \ldots, \eta$ when evaluated at $t = \tilde{t}$, and the correction (3.4) may be computed.

3.3 Stepsize selection strategy

Traditional stepsize control strategies rely on a local error estimate. As soon as it is available, a corrected stepsize may be computed. It is natural to employ similar techniques also in the case of integration on Lie groups or more general manifolds. Description of the strategy may be found in most standard texts on integration methods for ordinary differential equations, but for completeness we include a brief overview of the procedure.

In order to attain the local error estimate $r_{k+1} = \varepsilon$ at step time step $k+1$, the next stepsize h_{k+1} is chosen as a function of the previous stepsize, h_k, as follows (cf. e.g. (Stetter 1973) or (Gustafsson 1992)). Compute first

$$\widehat{h}_{k+1} = \alpha \left(\frac{\varepsilon}{r_k} \right)^{1/(p+1)} h_k,$$

where p is the order of the method (the order of the lower-order approximation scheme in the case of embedded schemes), $r_k = e_k$ (the error estimate) and α is a 'pessimist factor' (typically between 0.8 and 0.9, heuristically determined). In order to prevent rapid oscillations of the stepsize, it is common to restrict the extent of stepsize variation in any single step. A typical strategy is

$$h_{k+1} = \min\{h_{\max}, \max\{\alpha_{\text{small}} h_k, \min\{\alpha_{\text{large}} h_k, \widehat{h}_{k+1}\}\}\},$$

where $h_{\max}$ is the largest allowed stepsize, while α_{small} and α_{large} are two constants (typically 0.5 and 2.0, respectively).

If the local error exceeds the tolerance by a factor more than α_{accept}, which again is a constant chosen by the user (typically 1.2), then we reject the step and retry with a smaller stepsize h_{k+1}. This algorithm proceeds until the local error estimate satisfies

$$e_{k+1} \leq \alpha_{\text{accept}} \varepsilon.$$

In the next section we present a number of numerical results. In most of the simulations we have compared the above stepsize selection algorithm implemented for the Magnus series method (2.9) with the MATLAB routine `ode45`.

4 Numerical results

In this section we demonstrate some of the properties of the fourth-order Magnus method (2.9) which we designate **MG4**. For clarity we repeat the definition of the method from Section 2,

$$\begin{aligned} a_1 &= a\left(t_n + (\tfrac{1}{2} - \tfrac{\sqrt{3}}{6})h\right), \qquad a_2 = a\left(t_n + (\tfrac{1}{2} + \tfrac{\sqrt{3}}{6})h\right), \\ \sigma^n &= \tfrac{1}{2}h(a_1 + a_2) + \tfrac{\sqrt{3}}{12}h^2[a_2, a_1], \\ y_{n+1} &= \mathrm{e}^{\sigma^n} y_n. \end{aligned}$$

We compare the solutions generated by **MG4** with those obtained from the classical fourth order Runge-Kutta method (**RK4**) and the fourth order Gauss-Legendre method (**GL4**). The Butcher tableaux of these methods are displayed in Table 3.

Note that all the methods effectively use two function evaluations per step. In addition, **RK4** uses four matrix multiplications and seven linear combinations, **MG4** uses four matrix multiplications, four linear combinations and one matrix exponential,

$$
\begin{array}{c|cccc}
0 & & & & \\
\frac{1}{2} & \frac{1}{2} & & & \\
\frac{1}{2} & 0 & \frac{1}{2} & & \\
1 & 0 & 0 & 1 & \\
\hline
 & \frac{1}{6} & \frac{2}{6} & \frac{2}{6} & \frac{1}{6}
\end{array}
\qquad\qquad
\begin{array}{c|cc}
\frac{3-\sqrt{3}}{6} & \frac{1}{4} & \frac{3-2\sqrt{3}}{12} \\
\frac{3+\sqrt{3}}{6} & \frac{3+2\sqrt{3}}{12} & \frac{1}{4} \\
\hline
 & \frac{1}{2} & \frac{1}{2}
\end{array}
$$

Table 3: Butcher tableaux for the fourth-order Runge–Kutta (left) and Gauss–Legendre (right) methods.

and **GL4** uses two matrix multiplications, two linear combinations and it solves a system of linear equations at each time step. For each test problem we have plotted the global error obtained versus number of floating point operations (counted by MATLAB) used by each of the numerical methods. A striking result is that although the Magnus series method evaluates a matrix exponential at each time step, the accuracy obtained is significantly higher than that from **RK4**, so that it is actually more efficient on the test problems reported in this paper. This picture may possibly change when integrating very large systems of linear equations since then the evaluation of the matrix exponential may become very time consuming. Note that this increase in accuracy is an added bonus, since our original intention was to design methods that are assured to stay on a Lie group, a feat which is achieved by neither of the 'competitor' methods **RK4** and **GL4**.

To the end of this paper we let, unless explicitly redefined, solid lines denote numerical results from **MG4**, dashed lines denote numerical results from **RK4** and dotted lines denote numerical results from **GL4**.

Differential Riccati equations We wish to solve the *matrix differential Riccati equation*

$$y'(t) = a(t)y(t) + b(t) - y(t)c(t)y(t) - y(t)d(t), \tag{4.1}$$

whose coefficients are matrix functions,

$$a(t) \in \mathbb{R}^{n\times n}, \quad b(t) \in \mathbb{R}^{n\times m}, \quad c(t) \in \mathbb{R}^{m\times n} \quad \text{and} \quad d(t) \in \mathbb{R}^{m\times m}. \tag{4.2}$$

Consider first the scalar equation $y'(t) = a(t)y(t) + b(t) - c(t)y^2(t)$ that evolves via infinitesimal group transformations, as shown e.g. in (Schiff & Shnider 1996). Each element $\left[\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right]$ of $\mathrm{SL}(2,\mathbb{R})$ gives rise to a Möbius transformation $z \mapsto (az+b)/(cz+d)$ of the extended real line, $\mathbb{R}\cup\{\infty\}$, where we set $ad - bc = 1$ by rescaling. These transformations form the real Möbius group under composition of functions, and is nothing else but the *real projective group* $\mathbb{R}\mathrm{P}(2) \simeq \mathrm{S}^1/\mathbb{Z}_2 \simeq \mathrm{S}^1$. Consider next the generalized Möbius transformation

$$y(t) = [\alpha(t)y(t_0) + \beta(t)]\,[\gamma(t)y(t_0) + \delta(t)]^{-1} \tag{4.3}$$

that maps $y(t_0)$ to $y(t)$ (both as elements in $\mathbb{R}^{n\times m}$, the set of $n \times m$ real matrices) and the coefficients are matrix functions as in (4.2). We solve

$$\dot{\Gamma}(t) = A(t)\Gamma(t) \quad \text{with} \quad \Gamma(t_0) = \mathrm{I}_{n+m} \in \mathrm{GL}(n+m, \mathbb{R}) \tag{4.4}$$

where

$$\Gamma(t) = \begin{bmatrix} \alpha(t) & \beta(t) \\ \gamma(t) & \delta(t) \end{bmatrix} \in \mathrm{GL}(n+m, \mathbb{R})$$

and

$$A(t) = \begin{bmatrix} a(t) & b(t) \\ c(t) & d(t) \end{bmatrix} \in \mathfrak{gl}(n+m, \mathbb{R}).$$

Then (4.3) is the solution of (4.1) at time t with initial conditions $y(t_0)$.

Schiff & Shnider (1996) propose to use the method

$$\widetilde{\Gamma}_2(t_k, h) = \left[\mathrm{I}_{n+m} + hA\left(t_k + \frac{h}{2}\right) + \frac{h^2}{2}A\left(t_k + \frac{h}{2}\right)^2\right]\widetilde{\Gamma}_2(t_{k-1}, h)$$

with $\widetilde{\Gamma}_2(t_0, h) = \Gamma(t_0)$ to solve (4.4) (we will denote this method by **Schiff2**). This is nothing else but a truncated Magnus series method of order two. To see this, consider $p = 1$ in (2.4), restricting the attention to terms involving only a single integral,

$$\sigma_1(t) = \int_0^t a(\tau)\,\mathrm{d}\tau.$$

If integrated exactly, we obtain $\sigma_1(t) = \sigma(t) + \mathcal{O}(t^3)$, hence it suffices to use the second-order Gaussian quadrature given by the abscissa $c_1 = \frac{1}{2}$ and the weight $b = 1$. It follows that

$$\mathrm{e}^{\tilde{\sigma}_1(t)} = \Gamma(t) + \mathcal{O}(t^3),$$

where $\tilde{\sigma}_1(t)$ represents the approximation of $\sigma_1(t)$ by the above quadrature rule. Finally, we truncate the exponential so as to obtain a second-order approximation to the solution

$$\sum_{k=0}^{2} \frac{1}{k!}\tilde{\sigma}_1^k(t) = \Gamma(t) + \mathcal{O}\left(t^3\right).$$

Observe that, the exponential mapping having being truncated, the solution does not stay on the real projective group as time integration proceeds.

Note that the Magnus series methods integrate linear differential equations (2.1) with constant coefficients exactly. This implies that, subject to the above procedure of rendering Riccati equations in a projective space, these methods also integrate differential Riccati equations with constant coefficients exactly.

We consider the differential Riccati equations described in Problem 1 and Problem 3 in (Schiff & Shnider 1996) (equivalently, Example 1 and Example 4 in (Dieci 1992)). Firstly, let

$$A(t) = \begin{bmatrix} 0 & 0 & 0 & 1 \\ -10 & -1 & 10 & 0 \\ 0 & 1 & 0 & 0 \\ 100 & 0 & -100 & -1 \end{bmatrix} \tag{4.5}$$

in (4.4) and advance (4.3) with the initial condition $y(0) = \left[\begin{smallmatrix} 0 & 0 \\ -1 & 0 \end{smallmatrix}\right]$ in the interval $0 \leq t \leq 5$.

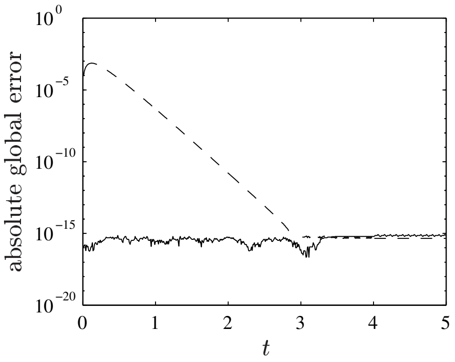


Figure 1: The error, $|y_{i,\text{exact}} - y_i|$, for the Riccati problem described by (4.5). The solid line denotes the solution as produced by **MG4** while the dashed line denotes the solution produced by **Schiff2**. The stepsize used in the simulation was 0.01.

Secondly, we choose

$$A(t) = \begin{bmatrix} 0 & \frac{t}{2\epsilon} & \frac{1}{2} & 1 \\ 0 & 0 & 0 & 1 \\ \frac{1}{\epsilon} & 0 & -\frac{t}{2\epsilon} & 0 \\ 0 & \frac{1}{\epsilon} & 0 & 0 \end{bmatrix} \tag{4.6}$$

in (4.4) and advance (4.3) with the initial condition $y(-1) = \left[\begin{smallmatrix} 0 & 0 \\ 0 & 0 \end{smallmatrix}\right]$ in the interval $-1 \leq t \leq 1$.

Problems with exponentially growing solutions It is generally difficult to obtain qualitatively correct solutions of differential equations with exponentially-growing components. Consider for example the problem (2.1) with

$$a(t) = \begin{bmatrix} 100t & 0 \\ 0 & -100 \end{bmatrix} \tag{4.7}$$

and initial conditions $y_0 = [1, 1]^{\mathrm{T}}$, integrated between $t_0 = 0.0$ and $t_{\text{end}} = 0.5$. The first solution component grows from 1 to 10^5 within this time window and, as shown in

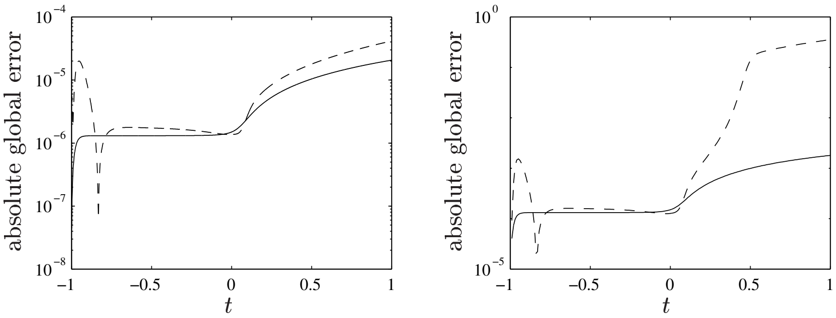


Figure 2: The error, $|y_{i,\text{exact}} - y_i|$, for the Riccati problem described by (4.6). The solid line denotes the solution as produced by **MG4** while the dashed line denotes the solution produced by **Schiff2**. In the simulation we used $\epsilon = 0.001$ and stepsizes 0.001 (left) and 0.01 (right).

Figure 3, classical methods have serious problems integrating with sufficient accuracy. **MG4**, on the other hand, generates a solution with an error that is many orders of magnitude smaller.

The lower right figure shows the stepsizes chosen by **MG4** and `ode45` when the required error at the endpoint of the intervals was 10^{-8}. The attainment of this error bound required tolerance of 10^{-6} in the **MG4** case and 10^{-14} in the `ode45` case, the latter dangerously near the machine epsilon in IEEE arithmetic.

Stiff problems We consider next the solution of (2.1) with

$$a(t) = \begin{bmatrix} -1000t & 1 \\ 0 & -t \end{bmatrix} \qquad \text{and} \qquad y(0) = \begin{bmatrix} -1 \\ 1 \end{bmatrix}. \tag{4.8}$$

Typical of stiff problems, (4.8) imposes severe restrictions on the stepsize used in the integration by classical Runge–Kutta methods, not because of the accuracy of the solution but because of the stability of the underlying method.

Since the Riccati equation (4.1) is invariant under transformations of the form $A \mapsto A + p(t)I_{n+m}$ in (4.4), i.e. transformations where $a(t) \mapsto a(t) + p(t)I_n$ and $d(t) \mapsto d(t) + p(t)I_m$, it was proposed by Schiff & Shnider (1996) to transform the spectrum of A so that the stiffness in the problem disappears. As can be seen from numerical experiments, a shift of the spectrum of A is not necessary when using Magnus series methods, since these methods exploit the underlying structure in the problem and avoid the degradation in performance associated with stiffness in classical methods.

The Magnus series methods are geometrically stable in the sense that if $\gamma_n \in \mathrm{G}$ is the approximation to the integral curve $t \mapsto \gamma(t)$ at time t_n and G is a Lie group, then, except for roundoff errors, the next iterate γ_{n+1} also lies in G. Since the solution remains bounded on all compact manifolds, this implies numerical boundedness whenever the Lie group G is compact. This is the case, for example, with the orthogonal group $\mathrm{O}(n)$, the unitary group $\mathrm{U}(n)$ and the projective spaces $\mathbb{RP}(n)$ and $\mathbb{CP}(n)$

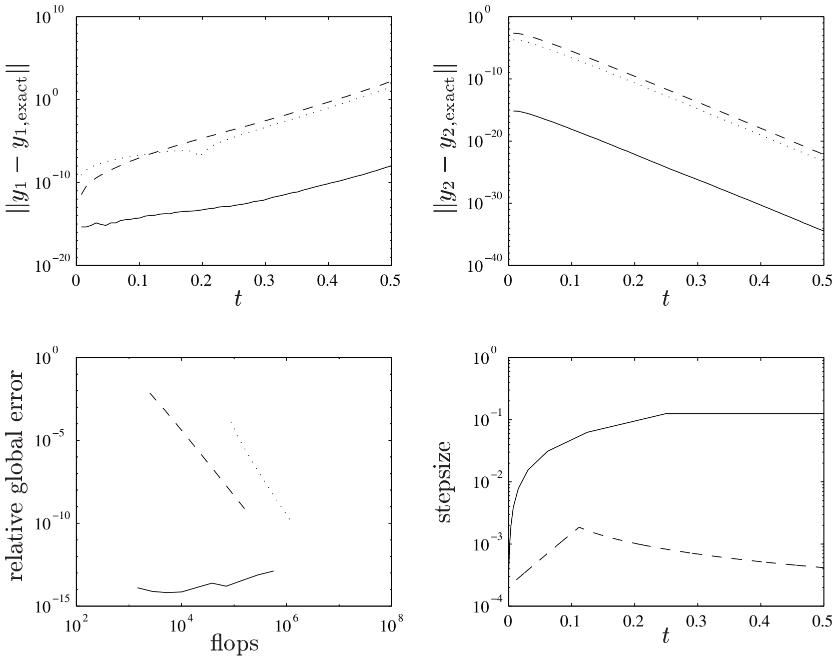


Figure 3: The error, $|y_{i,\mathrm{exact}} - y_i|$, for problem (4.7) with an exponentially growing solution is shown in the two upper figures. The lower left figure shows the relationship between the relative global error and number of floating points used by each routine and the lower right figure shows the stepsizes chosen by **MG4** (solid line) and `ode45` (dashed line) while integrating with a global error of 10^{-8} at the endpoint of the integration interval.

(which are in general not Lie groups), but not with the special linear group $\mathrm{SL}(n)$ and the symplectic group $\mathrm{Sp}(n)$. However, even if G is compact, small perturbations during the solution might well lead to integral curves that, although bounded, are far from exact at time $t > t_0$. This behaviour is familiar in computational dynamics, e.g. in the integration of Lorenz equations, but has not yet been investigated fully in the context of Lie-group solvers.

Figure 4 displays the error $|y_{i,\mathrm{exact}} - y_i|$ for the stiff problem (4.8). It is clear that only the Magnus series method **MG4** is stable. The stepsize used in the simulations was $h = \frac{1}{1000}$. The lower left figure shows the global error when using **MG4** with ten times larger stepsize, $h = \frac{1}{100}$. The lower right figure shows the stepsizes chosen by **MG4** (solid line) and `ode45` (dashed line) when the global error at the endpoint of the integration interval was required not to exceed 10^{-8}.

Orthogonal problems As an example of an equation (2.1) which evolves on the special orthogonal group $\mathrm{SO}(n)$, we consider the problem defined by the skew-symmetric

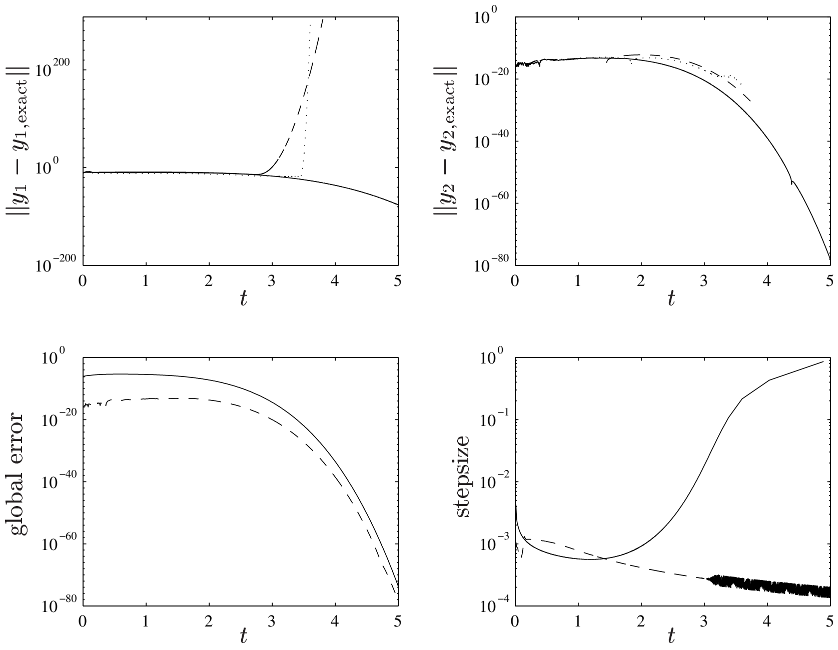


Figure 4: The two upper figures show the error, $|y_{i,\mathrm{exact}} - y_i|$, when integrating the stiff problem (4.8).

matrix, $a(t) \in \mathfrak{so}(n)$, whose upper triangular entries are

$$(-1)^{i+j} \frac{i}{j+1} t^{j-i}, \qquad 1 \le i < j \le n.$$

As the initial value we take the $n \times n$ identity matrix, integrating from $t_0 = 0$ to $t_{\mathrm{end}} = 3$. In the computation we have taken $n = 6$.

Figure 5 depicts the 2-norm of the error and the distance from the manifold (computed as $\|Y_i Y_i^T - Y_0 Y_0^T\|$) while integrating the above orthogonal problem. The constant stepsize used in the simulations was $h = \frac{1}{50}$. The lower left figure shows the stepsizes chosen by the variable stepsize **MG4** and the `ode45` methods. The global error at the endpoint of the integration interval was limited to 10^{-6} in the simulations. The lower right figure shows the efficiency of the codes when applied to the orthogonal problem. Although **MG4** exponentiates a 6×6 matrix at each step, the method performs much better than both **RK4** and **GL4** methods. Note that in this case we could have replaced exponentiation with the fourth-order [2/2] diagonal Padé approximant without any damage to orthogonality, whilst further emphasizing the advantage in using **MG4.**

Problems in the special unitary group The special unitary group, $\mathrm{SU}(n)$, consists of $n \times n$ unitary matrices with unit determinant. Its Lie algebra, $\mathfrak{su}(n)$, is the

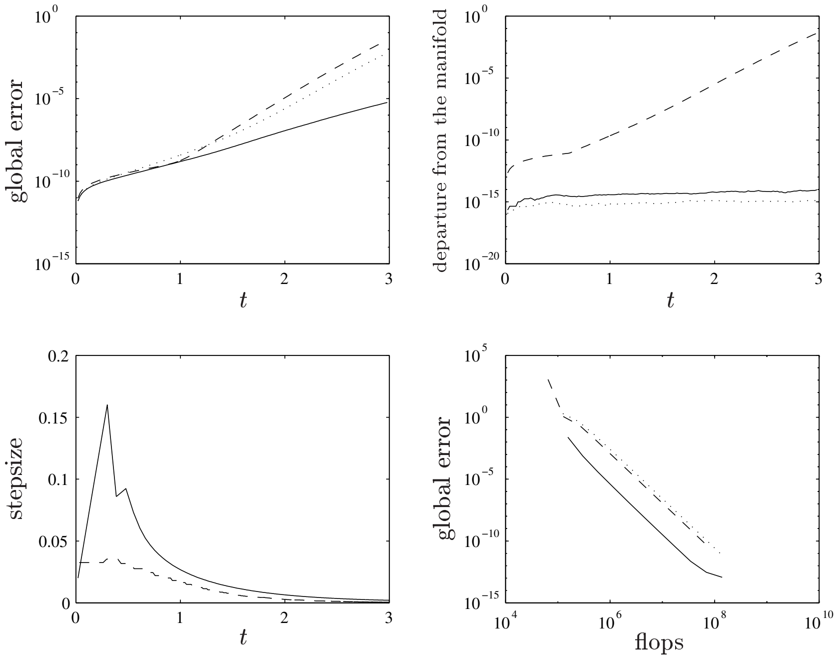


Figure 5: The 2-norm of the error, departure from orthogonality, stepsizes and the efficiency for the orthogonal problem.

set of all $n \times n$ skew-Hermitian complex matrices with zero trace. Such flows occur in a number of applications, not least in the calculation of Lyapunov exponents (Dieci, Russell & van Vleck 1994) and of isospectral flows (Zanna 1996). Integration of unitary flows has been studied by several authors, among them Dieci et al. (1994), Higham (1996), Iserles & Zanna (1995) and Zanna (1996). In particular, the familiar Runge–Kutta Gauss–Legendre methods are unitary.

Note that a unitary matrix has determinant with modulus equal to one, i.e. it is of the form $\mathrm{e}^{\mathrm{i}\theta}$. As shown in (Zanna 1996), theoretically the determinant of the numerical solution produced by **GL4** will not be preserved, but it will remain one in modulus. Although Gauss–Legendre methods are unitary, they are not $\mathrm{SL}(n, \mathbb{C})$-invariant, hence they depart off $\mathrm{SU}(n)$.

As a test problem we use (2.1) with

$$a(t) = \begin{bmatrix} 0 & 1 - \mathrm{i}t & \log(1+t) + 2\mathrm{i} \\ -1 - \mathrm{i}t & 0 & -t - \mathrm{i}\log(1+t) \\ -\log(1+t) + 2\mathrm{i} & t - \mathrm{i}\log(1+t) & 0 \end{bmatrix}. \tag{4.9}$$

Our initial condition is the 3×3 identity matrix and we integrate from $t_0 = 0$ to $t_{\mathrm{end}} = 5$ with the time step $\frac{1}{100}$.

Figure 6 shows that the **RK4** and the **GL4** methods generate drift in the determinant of the solution matrices of size 10^{-6} and 10^{-7}, respectively. The determinant

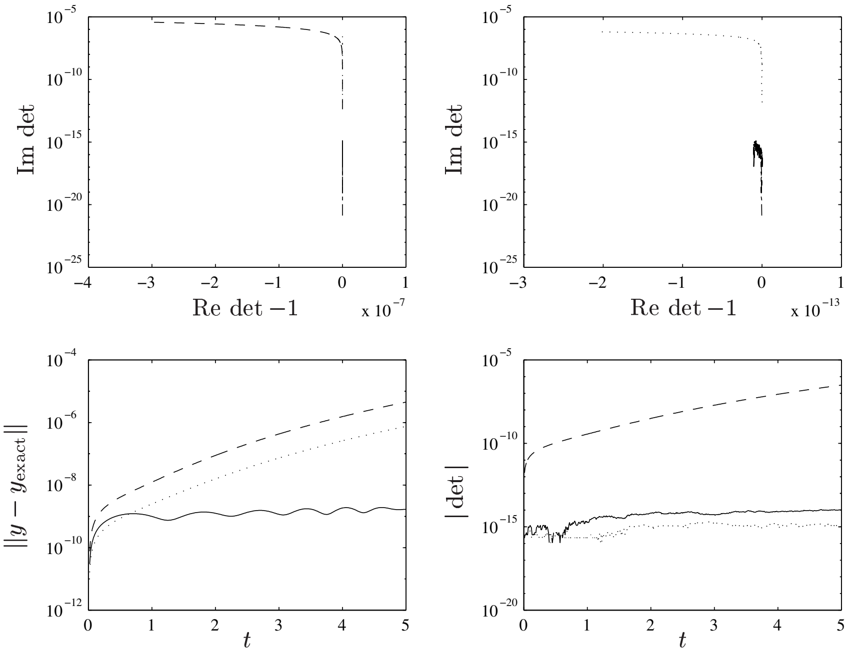


Figure 6: The solution of the unitary system (4.9) with $h = \frac{1}{100}$. Note the drift of the determinant for **RK4** and **GL4.**

of the result produced by **GL4** has modulus one and it thus stays on the unit circle in the complex plane. The numerical solution therefore lies in U(3), as expected.

The **MG4** method is the only of these three methods that generates, up to machine accuracy, solutions in the correct group, SU(3).

Symplectic problems Numerical solution of Hamiltonian equations has received a great deal of attention in the last decade (Sanz-Serna & Calvo 1994). An important qualitative attribute of a Hamiltonian system is that the Jacobian matrix of its flow evolves in the symplectic group $\mathrm{Sp}(n)$.

As a symplectic test problem we consider the harmonic oscillator with time-varying spring constant $k(t)$. This system has one degree of freedom and the Hamiltonian energy function

$$H(p,q) = T(p) + V(q), \quad \text{with} \quad T(p) = \frac{p^2}{2m} \quad \text{and} \quad V(q) = \frac{k(t)q^2}{2}, \tag{4.10}$$

with $k(t) = 1 + \varepsilon\cos(t)$. We have used $\varepsilon = 10^{-4}$ and $m = 2$ in the simulations. Figure 7 displays the energy error, $E_n = H(p_n, q_n) - H(p_0, q_0)$, as a function of time. Note that the Magnus series method has the same energy conservation properties as the symplectic Runge–Kutta Gauss–Legendre method.

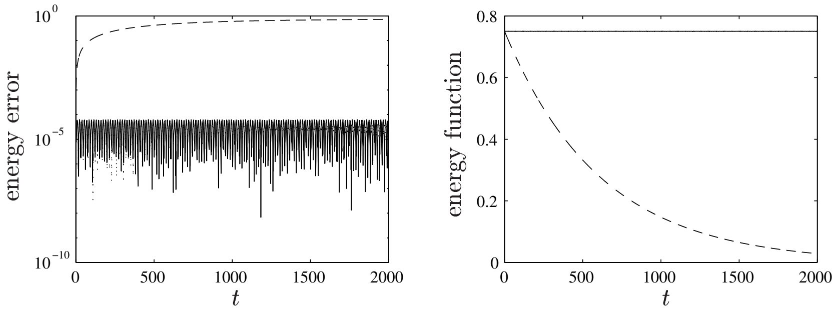


Figure 7: The energy error in the three codes when simulating the harmonic oscillator (4.10). The stepsize used in the simulations was $h = 1$.

Note that **MG4** is not a symplectic method in the usual sense: although the Jacobian matrix evolves on $\mathrm{Sp}(n)$, it is not necessarily true that $\mathrm{d}\boldsymbol{p} \wedge \mathrm{d}\boldsymbol{q}$ remains constant when the method is applied to the Hamiltonian problem

$$\dot{\boldsymbol{p}} = -\frac{\partial H(\boldsymbol{p},\boldsymbol{q})}{\partial \boldsymbol{q}}, \qquad \dot{\boldsymbol{q}} = \frac{\partial H(\boldsymbol{p},\boldsymbol{q})}{\partial \boldsymbol{p}}.$$

This issue is discussed more extensively by Zanna (1996). Yet, Figure 7 indicates that Hamiltonian energy is conserved by **MG4** just as well as by the symplectic method **GL4.**

In Figure 8 we compare the drift of the solutions when all three methods are applied to the slightly unstable problem (2.1) with

$$a(t) = \begin{bmatrix} 1 & -1 & t & 1 \\ 2 & 2 & 1 & -t \\ -2t & -1 & -1 & -2 \\ -1 & 1 & 1 & -2 \end{bmatrix} \in \mathfrak{sp}(4). \tag{4.11}$$

The initial condition was taken to be the 4×4 identity matrix.

Second-order equations We have solved the Mathieu equation and the Bessel equation, both second-order problems. The Mathieu equation is given as

$$\ddot{x} + (a + b\cos t)x = 0, \tag{4.12}$$

and in the simulation we have used $a = 1$ and $b = \frac{1}{10}$, integrating from $t_0 = 0$ to $t_{\text{end}} = 50$ with a stepsize of $h = \frac{1}{5}$. The initial conditions were $x(0) = 1$ and $\dot{x}(0) = 1$.

Figure 9 is concerned with the solution of (4.12). The upper two graphs depict the error $|y_{i,\text{exact}} - y_i|$, while the lower right graph displays the stepsizes chosen by **MG4** (solid line) and `ode45` (dashed line) when the error at the endpoint of the intervals was required to be 10^{-3}. The tolerance imposed on the stepsize selection routine was 10^{-3} in the **MG4** case and 10^{-14} in the `ode45` case. Finally, the lower left figure

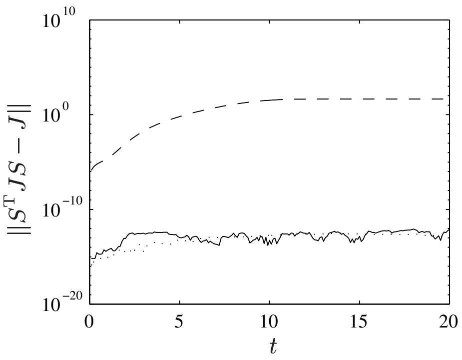


Figure 8: The drift from the symplectic manifold Sp(4) (represented by the quantity $\|S^T JS - J\|$) for the symplectic problem (4.11). The stepsize was constant and equal to $\frac{1}{10}$ for all three methods.

shows the relationship between the relative global error and number of floating point operations used by each routine.

The Bessel equation

$$\ddot{x} + \frac{1}{t}\dot{x} + \left(1 - \frac{\nu^2}{t^2}\right) x = 0. \tag{4.13}$$

is regular-singular at the origin. To avoid this singularity, we have integrated it from $t_0 = 1$ to $t_{\text{end}} = 50$. We have used $\nu = 1$ and the initial condition $x(1) = 1$, $\dot{x}(1) = 1$.

The error, $|y_{i,\text{exact}} - y_i|$, for the Bessel problem (4.13) is shown in the two upper graphs of Figure 10. The lower left graph therein displays the stepsizes chosen by **MG4** (solid line) and `ode45` (dashed line) when integrating with a global error of 10^{-3} at the endpoint of the integration interval, while the lower right graph shows the relationship between the relative global error and number of floating point operations used by each routine.

As shown in Figure 11, the correspondence between the measured global error and the imposed tolerance is much better for the variable stepsize **MG4** method than for the variable stepsize MATLAB routine, `ode45`.

5 Concluding Remarks

The original purpose of the research that has led to this paper was the practical implementation of the Magnus series approach from (Iserles & Nørsett 1997) to linear differential equations in Lie groups. This goal has been accomplished by establishing a framework for error control. This framework consists of two ingredients, the estimation of the truncation and of quadrature errors. While the truncation error is estimated by following the 'inner logic' of Magnus series and their association with graph theory, the derivation of quadrature error relies on polynomial interpolation in Lie algebras.

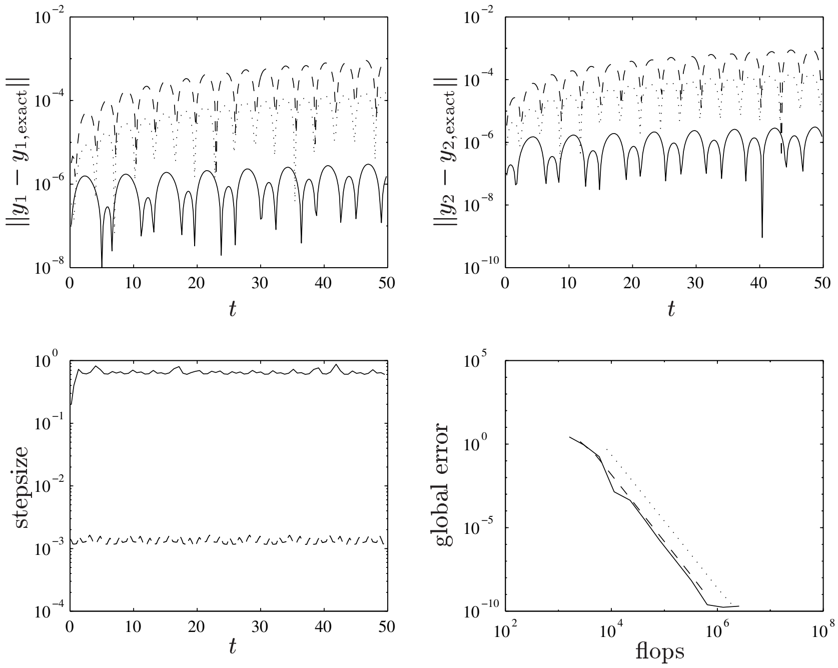


Figure 9: The solution of the Mathieu problem (4.12).

We have not addressed ourselves to the important issue of how to approximate the matrix exponential. This problem is common to a number of Lie-group solvers (Crouch & Grossman 1993, Munthe-Kaas 1998, Owren & Marthinsen 1997, Zanna 1996). It is known that the standard approach of replacing the exponential function by a rational approximant is unsuitable for general Lie groups. For example, it is possible to prove that the only analytic function mapping $\mathfrak{sl}(n)$ into $\mathrm{SL}(n)$ and consistent with e^z is the exponential function itself (Feng & Shang 1995). Although diagonal Padé approximants to e^z are sometimes appropriate, e.g. in the case of the orthogonal and the symplectic groups, the general computational problem of approximating the exponential inside a Lie group is still wide-open. Although it is currently the subject of active investigation (Celledoni & Iserles 1998), we have adopted in this paper a similar approach to other publications in the general area of Lie-group solvers, evaluating the exponential exactly.

It is hardly surprising that the method of Magnus series performs consistently better than classical Runge–Kutta methods, whether **RK4, GL4** or MATLAB's `ode45` insofar as the retention of Lie-group structure is concerned. After all, this is the whole purpose of Lie-group methods! We regard as considerably more surprising the phenomenon whereby 'general' equations, like the Mathieu equation (4.12) and the Bessel equation (4.13) are discretized considerably better (i.e., greater accuracy for less computational cost) by **MG4.** This, together with the remarkable behaviour of Magnus series in the discretization of the *Airy equation* $y'' + ty = 0$ in (Iserles & Nørsett 1997)

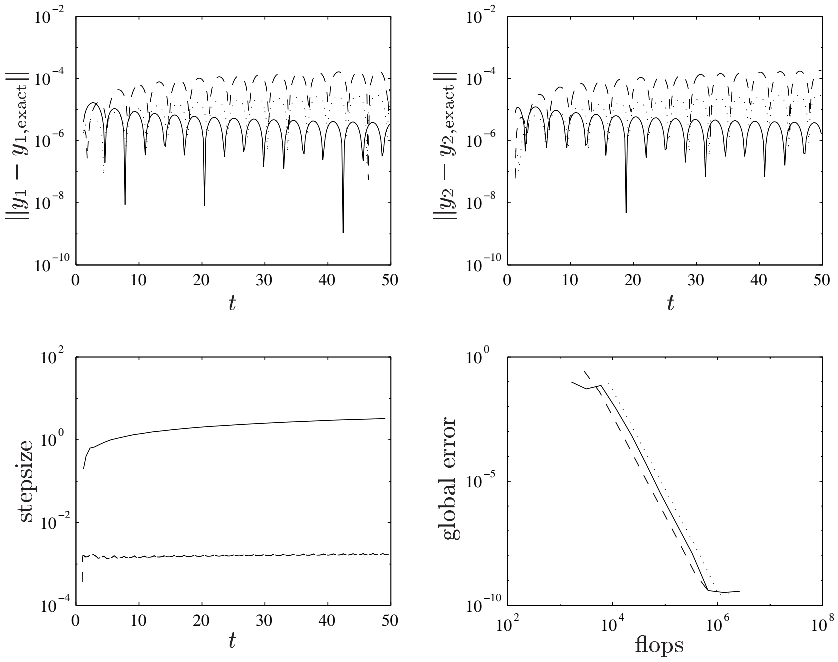


Figure 10: The solution of the Bessel equation (4.13).

implies that there is more to the method of Magnus series than meets the eye.

An intuitive explanation of the remarkable performance of Magnus series (and, by implication, of other Lie-group methods that first solve in the Lie algebra, subsequently mapping into the Lie group by means of the exponential function (Munthe-Kaas 1998, Zanna 1996)) is that they represent the solution as an exponential of a matrix function. Classical numerical methods are all underlied by the *ansatz* that the solution of a differential equation is of a polynomial character. Hence, for example, the entire classical concept of order. Yet, we all know that solutions of many differential equations are not polynomial in their behaviour – in particular, they may exhibit fast growth or decay, oscillations etc. Exponentials of polynomials model this range of behaviour much better and this, we believe, may account for the superior performance of Magnus series. Having said this, we are the first to acknowledge that the matter deserves further and more formal explanation.

Acknowledgements

The work of Arne Marthinsen was in part sponsored by British Petroleum under grant no. B94060.00 and by the Norwegian Research Council under contract no. 111038/410, through the SYNODE project. (WWW: `http://www.math.ntnu.no/num/synode/`.) The work of Syvert Nørsett was accomplished during his stay as an EPSRC Visiting

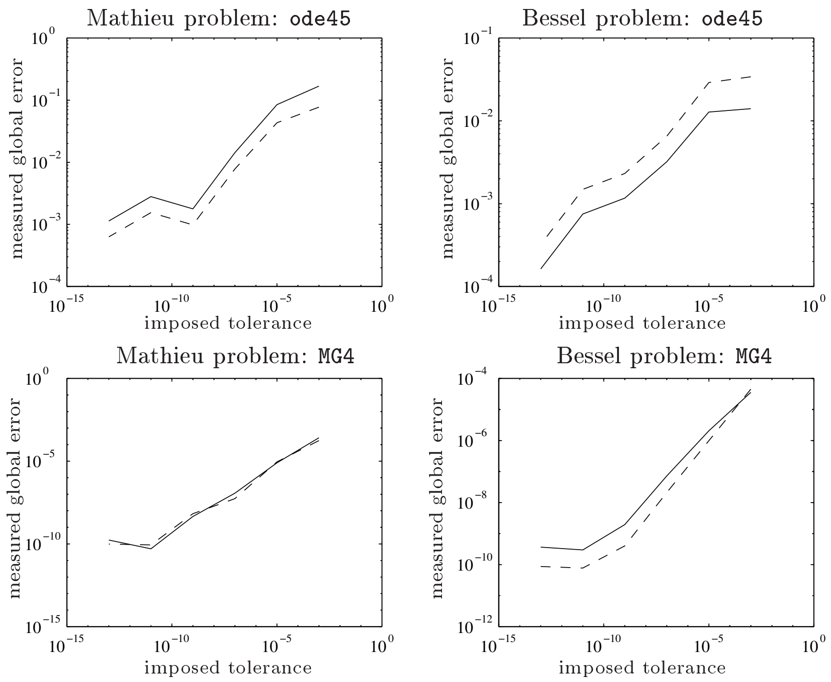


Figure 11: Measured global error versus requested tolerance for the MATLAB `ode45` routine and **MG4**. The first component of the solution is denoted by a solid line, while a dashed style is used for the second component.

Fellow at the Department of Applied Mathematics and Theoretical Physics, University of Cambridge.

References

Butcher, J. C. (1987), *The Numerical Analysis of Ordinary Differential Equations*, Wiley.

Celledoni, E. & Iserles, A. (1998), Approximating the exponential from a Lie algebra to a Lie group, DAMTP, University of Cambridge, to appear.

Cools, R. (1997), 'Constructing cubature formulae: the science behind the art', *Acta Numerica* **6**, 1–54.

Crouch, P. E. & Grossman, R. (1993), 'Numerical integration of ordinary differential equations on manifolds', *J. Nonlinear Sci.* **3**, 1–33.

Davis, P. & Rabinowitz, P. (1984), *Methods of Numerical Integration*, Academic Press, New York.

Dieci, L. (1992), 'Numerical integration of the differential Riccati equation and some related issues', *SIAM J. Numer. Anal.* **29**(3), 781–815.

Dieci, L., Russell, R. D. & van Vleck, E. S. (1994), 'Unitary integrators and applications to continuous orthonormalization techniques', *SIAM J. Numer. Anal.* **31**(1), 261–281.

Feng, K. & Shang, Z.-J. (1995), 'Volume-preserving algorithms for source-free dynamical systems', *Numer. Math.* **71**, 451–463.

Gustafsson, K. (1992), Control of Error and Convergence in ODE Solvers, PhD thesis, Department of Automatic Control, Lund Institute of Technology, Sweden.

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

Higham, D. J. (1996), Runge-Kutta type methods for orthogonal integration, Technical Report NA/168, Department of Mathematics and Computer Science, University of Dundee.

Iserles, A. & Nørsett, S. P. (1997), On the solution of linear differential equations in Lie groups, Technical Report 1997/NA3, Department of Applied Mathematics and Theoretical Physics, University of Cambridge, England.

Iserles, A. & Zanna, A. (1995), Qualitative numerical analysis of ordinary differential equations, *in* J. Renegar, M. Shub & S. Smale, eds, 'Lectures in Applied Mathematics', AMS, Providence RI. DAMTP Technical Report 1995/NA05.

Iserles, A., Nørsett, S. P. & Rasmussen, A. F. (n.d.), High-order discretization methods in a Lie group, to appear.

Magnus, W. (1954), 'On the exponential solution of differential equations for a linear operator', *Comm. Pure and Appl. Math.* **7**, 649–673.

Munthe-Kaas, H. (1998), 'Runge-Kutta methods on Lie groups', *BIT* **38**, 92–111.

Munthe-Kaas, H. & Owren, B. (n.d.), Computations in a free Lie algebra, to appear.

Owren, B. & Marthinsen, A. (1997), Integration methods based on rigid frames, Technical Report Numerics No. 1/1997, Department of Mathematical Sciences, The Norwegian University of Science and Technology.

Sanz-Serna, J. M. & Calvo, M. P. (1994), *Numerical Hamiltonian Problems*, Chapman & Hall, London.

Schiff, J. & Shnider, S. (1996), A natural approach to the numerical integration of Riccati differential equations, Technical report, Department of Mathematics and Computer Science, Bar Ilan University, Israel.

Stetter, H. J. (1973), *Analysis of Discretization Methods for Ordinary Differential Equations*, Springer-Verlag, Berlin.

Zanna, A. (1996), The method of iterated commutators for ordinary differential equations on Lie groups, Technical Report 1996/NA12, Department of Applied Mathematics and Theoretical Physics, University of Cambridge, England.