A brief introduction to Lie-group methods

Arieh Iserles [*]

1 Why Lie groups?

As mathematicians, we are excited by the profundity and æsthetic majesty of mathematical concepts. Yet, as numerical analysts, we cannot escape this elementary test: how much does a given construct, beautiful as it might be, help us to elucidate the nature of the computational phenomenon that we happen to investigate? Nothing can be taken for granted without justification and, by the same token, nothing that furthers our understanding can be discarded.

Lie groups have played for more than a century a decisive role in our understanding of the geometry of differential equations. Their theory is extensive, deep and beautiful. It is the contention of this paper that the concept of Lie groups, within the wider terminology and machinery of differential geometry, is very helpful indeed in devising and analysing superior methods to discretise continuous dynamical systems. By sharing geometric structure and invariants with the original differential system, these methods are more accurate, more stable and occasionally cheaper for a significant range of differential problems.

1.1 Example 1: An orthogonal flow

Consider the matrix-valued differential system

$$Y' = A(t, Y)Y, \quad t \geq 0, \qquad Y(0) = Y_0 \in \mathrm{O}(n), \tag{1}$$

where $A : \mathbb{R}_+ \times \mathrm{O}(n) \to \mathfrak{so}(n)$. Here $\mathrm{O}(n)$ is the set of $n \times n$ real orthogonal matrices and $\mathfrak{so}(n)$ is the linear space of $n \times n$ real skew-symmetric matrices. This system has a feature which, while trivial to prove, leads to surprising consequences and applications. Because of the skew-symmetry of A,

$$\frac{\mathrm{d}}{\mathrm{d}t} Y^\top Y = {Y'}^\top Y + Y^\top Y' = Y^\top (A^\top + A) Y = \mathbf{O}.$$

[*]Department of Applied Mathematics and Theoretical Physics, University of Cambridge, Silver Street, Cambridge CB3 9EW, England

Since $Y_0^\top Y_0 = \mathbf{I}$, we deduce that $Y^\top(t)Y(t) \equiv \mathbf{I}$ for all $t \geq 0$: the system (1) evolves in the set of orthogonal matrices! This leads to a number of important applications, not least in the computation of Lyapunov exponents [12]. In all such applications it is vital to respect orthogonality. Yet, once standard numerical solvers are applied to (1), it is highly likely that orthogonality will be lost [6]: the only notable exception are certain highly-implicit Runge–Kutta schemes, e.g. the Gauss–Legendre methods [13]. This is hardly surprising: Classical numerical methods evolve in a Euclidean, 'flat' space, while $\mathrm{O}(n)$ is a nonlinear object!

The set $\mathrm{O}(n)$ is an example of a *Lie group:* while from purely algebraic point of view it is a *group:* $P, Q \in \mathrm{O}(n) \Rightarrow PQ, P^{-1} \in \mathrm{O}$, from a topological standpoint $\mathrm{O}(n)$ is a *manifold.* Moreover, the group operations (left and right multiplication, inversion) are continuous with respect to the underlying topology of the manifold. Other familiar examples of Lie groups are the *special linear group* $\mathrm{SL}(n)$ of all real $n \times n$ matrices with unit determinant, the *unitary group* $\mathrm{U}(n)$ of all complex $n \times n$ unitary matrices, the *special orthogonal group* $\mathrm{SO}(n) = \mathrm{O}(n) \cap \mathrm{SL}(n)$ of $n \times n$ rotations and the *symplectic group* $\mathrm{Sp}(n)$ of real $(2n) \times (2n)$ matrices X such that

$$X \begin{bmatrix} \mathbf{O} & \mathbf{I} \\ -\mathbf{I} & \mathbf{O} \end{bmatrix} X^\top = \begin{bmatrix} \mathbf{O} & \mathbf{I} \\ -\mathbf{I} & \mathbf{O} \end{bmatrix}.$$

Lie groups often embody crucial information about the geometry, invariants and even variational principles of differential equations [31].

Challenge 1: Given a differential equation evolving in a Lie group $\mathcal{G}$, devise a discretization method that also evolves in $\mathcal{G}$.

1.2 Example 2: Rotation-driven flows

Consider the ordinary differential system

$$\boldsymbol{y}' = \boldsymbol{f}(t, \boldsymbol{y}), \quad t \geq 0, \qquad \boldsymbol{y}(0) = \boldsymbol{y}_0 \in \mathbb{R}^n, \tag{2}$$

where $\|\boldsymbol{y}_0\|_2 = 1$ and $\boldsymbol{x}^\top \boldsymbol{f}(t, \boldsymbol{x}) = 0$ for all $t \geq 0$ and $\boldsymbol{x} \in \mathbb{R}^n$. We deduce at once that the function $\boldsymbol{y}(t)$ evolves in the unit sphere $\mathbb{S}^{n-1} \subset \mathbb{R}^n$. Given the ubiquity of such flows (after all, our planet is approximately a sphere!), we consider

Challenge 2: Given a differential equation evolving in a sphere, devise a discretization method that respects this feature.

Next, we turn our attention to the *isospectral flow*

$$Y' = [A(t, Y), Y], \quad t \geq 0, \qquad Y(0) = Y_0 \in \mathrm{Sym}(n), \tag{3}$$

where $A : \mathbb{R}_+ \times \mathrm{Sym}(n) \to \mathfrak{so}(n)$ and $[\,\cdot\,,\,\cdot\,]$ is the matrix commutator. The main structural feature of (3) is already implicit in its name: as t varies, the *eigenvalues* of $Y(t)$ stay the same [21]. Isospectral flows occur in many parts of mathematics but perhaps their most important application (at least insofar as numerical analysts are concerned) is to numerical linear algebra [9].

Challenge 3: Given an isospectral flow, devise a discretization method that keeps the eigenvalues constant.

On the face of it, equations (2) and (3) have nothing in common. In particular, from computational point of view, Challenge 2 is relatively easy: symplectic Runge–Kutta methods raise to it and, at least in principle, it is easy to project (unfortunately, often with poor stability characteristics) the outcome of any numerical method on a sphere. On the other hand, Challenge 3 is insurmountable and no classical numerical methods (Runge–Kutta, multistep, Taylor expansions, ...) can be assured to keep the eigenvalues constant [6]. Yet, from the viewpoint of differential geometry, (2) and (3) are closely associated, since they are both examples of an *orthogonal action* and their evolution is, in a well-defined sense, driven by rotation.

Every point of the sphere $\mathbb{S}^{n-1}$ can be represented in the form $Q\boldsymbol{y}_0$, where $Q \in \mathrm{SO}(n)$. Therefore, instead of asking "what is the solution $\boldsymbol{y}(t)$ of (2)?", we might ask "what is the rotation $Q(t)$ that takes $\boldsymbol{y}_0$ to $\boldsymbol{y}(t)$?". By the same token, every symmetric matrix which is similar to Y_0 can be written in the form $PY_0P^\top$ where $P \in \mathrm{SO}(n)$. Thus, instead of determining the solution $Y(t)$ of (3), we might seek $P(t) \in \mathrm{SO}(n)$ that induces the similarity transformation $Y(t) = P(t)Y_0P^\top(t)$. Note that, once Q or P above are available, our challenges are fulfilled since, by design, $\|\boldsymbol{y}_N\|_2 = 1$ and $\sigma(Y_N) = \sigma(Y_0)$, where $\boldsymbol{y}_N \approx \boldsymbol{y}(Nh)$, $Y_N \approx Y(Nh)$.

(2) and (3) alike are elementary examples of a *group action.* With greater generality, suppose that we are given a differential equation (2) that evolves in a smooth manifold $\mathcal{M} \subset \mathbb{R}^n$. In addition, suppose that there exists a Lie group $\mathcal{G}$ and a map $\Lambda : \mathcal{G} \times \mathcal{M} \to \mathcal{M}$ such that

$$\Lambda(\mathbf{I}, \boldsymbol{y}) = \boldsymbol{y}, \qquad \Lambda(X_1, \Lambda(X_2, \boldsymbol{y})) = \Lambda(X_1 \cdot X_2, \boldsymbol{y}), \qquad X_1, X_2 \in \mathcal{G},\ \boldsymbol{y} \in \mathcal{M}.$$

We say that $\mathcal{G}$ *acts* upon $\mathcal{M}$ and that Λ is the *group action.* A group action is said to be *transitive* if, given any $\boldsymbol{y}_1, \boldsymbol{y}_2 \in \mathcal{M}$ there exists $X \in \mathcal{G}$ such that $\Lambda(X, \boldsymbol{y}_1) = \boldsymbol{y}_2$. A manifold subjected to a transitive group action is called a *homogeneous space* [31]: an elementary example is a Lie group acting on itself, less trivial examples include spheres, tori, isospectral orbits, Stiefel and Grassmann manifolds, semi-direct product spaces and much more. The terminology of group actions allows us to issue a considerably more general challenge,

Challenge 4: Given a flow over a homogeneous space $\mathcal{M}$, discretise it so that the numerical solution lies in $\mathcal{M}$.

It also suggests a powerful means to answering it: Instead of seeking the discretization directly in $\mathcal{M}$, find the element of $\mathcal{G}$ whose action induces the approximation.

1.3 Example 3: Highly-oscillatory equations

Suppose for the sake of simplicity that we are given the linear oscillator

$$y'' + g(t)y = 0, \tag{4}$$

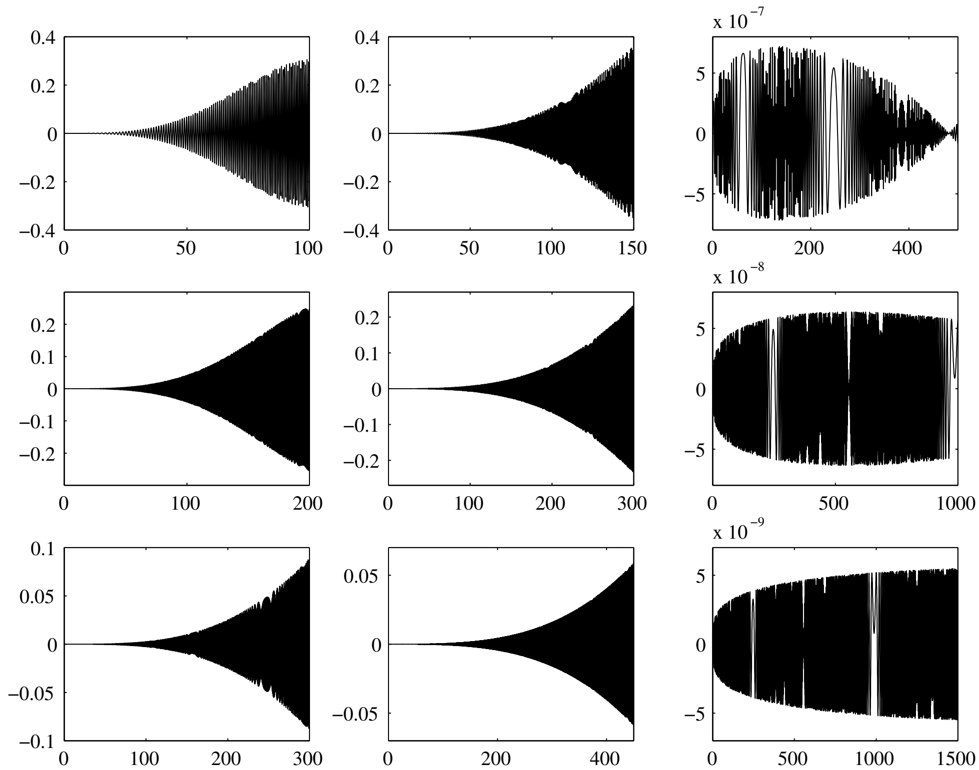


Figure 1. *The error in solution of the Airy equation by 4th-order explicit (left column) and implicit (middle column) Runge–Kutta methods and by 4th-order Magnus method (right column), with step sizes* $\frac{1}{10}, \frac{1}{20}$ *and* $\frac{1}{40}$.

where $g(t) > 0$, $\lim_{t\to\infty} g(t) = +\infty$ and $g'(t) = o(g(t))$, $t \gg 1$. According to standard WKB asymptotic analysis,

$$\begin{bmatrix} y(t) \\ y'(t) \end{bmatrix} \approx \begin{bmatrix} g^{-1/4}(t) & 0 \\ 0 & g^{1/4}(t) \end{bmatrix} \begin{bmatrix} \cos\theta(t) & \sin\theta(t) \\ -\sin\theta(t) & \cos\theta(t) \end{bmatrix} \boldsymbol{s}, \tag{5}$$

where $\theta(t) = \int_0^t g^{1/2}(x)\mathrm{d}x$ and $\boldsymbol{s} \in \mathbb{R}^2$ is constant. In other words, while the amplitude of y usually decays gently, the frequency of oscillation increases.

Equation (4) is typical of the many highly-oscillatory systems that occur in numerous applications, yet it is simple enough to be amenable to very detailed analysis. One can hardly expect classical numerical methods to do well for such equations, and this is confirmed in Figure 1. It displays the global error committed in the numerical solution of the *Airy equation*

$$y'' + ty = 0, \quad t \geq 0, \qquad y(0) = 1,\ y'(0) = 0,$$

by three fourth-order methods: the classical Runge–Kutta scheme (11), the Gauss–Legendre Runge–Kutta method and the Magnus method. As can be seen, except for the rightmost column, the error accumulates very fast indeed and quite soon the solution becomes virtually meaningless. Note that classical fourth-order, four-stage Runge–Kutta has poor stability features, yet the performance of the A-stable, symplectic Gauss–Legendre scheme is not much better! Of course, Figure 1 has been produced with constant step size and without error control. Yet, no standard error control can cope successfully with very high oscillation and, for example, common MATLAB ODE routines experience severe degradation in their efficiency and reliability [20].

Numerical analysis of classical and Lie-group methods for linear oscillators (4) has been presented in [17]. Thus, it is possible to show that for every classical (i.e., Runge–Kutta, multistep, multiderivative or general linear) method of even order $p = 2s$, implemented with sufficiently small constant step h, it is true that

$$y_{t/h} - y(t) \approx ch^4 g^{-1/4}(t) \int_0^t g^{s+\frac{1}{2}}(x) \mathrm{d}x \chi(t), \qquad t \gg 1,$$

where $c \neq 0$ is a constant and $\chi \in [-1, 1]$ is a fast-oscillating function. In particular, for the classical fourth-order methods in Figure 1 we obtain the power law $y_{t/h} - y(t) \approx ch^4 t^{13/4} \chi(t)$. This explains the poor behaviour observed in the figure and motivates our next challenge,

Challenge 5: Solve highly-oscillating equations with methods that experience no degradation in performance.

As evident from the rightmost column in Figure 1, we already have a worthy candidate, the Magnus method. This is one of the Lie-group methods that will be described in the sequel. At the moment we mention just two relevant items of information. Firstly, also the fourth-order Magnus method accumulates global error according to a power law, except that the latter is much more favourable, $y_{t/h} - y(t) \approx ch^4 t^{1/4} \chi(t)$ [17]. Secondly, reparametrization of the Magnus method with respect to a fast-rotating frame of reference has led in [17] to a method that attenuates the error (for the Airy equation and many similar equations) at exactly the same rate as the decay of the exact solution, uniformly for all $t \in [0, \infty)$.

Challenges 1–4 have had a common thread: the the desirability of respecting invariants and geometric structure under discretization. Challenge 5 is of a different kind, since no invariants are involved. The reason for suitability of Lie-group methods for highly-oscillatory equations is more subtle [17].

2 The main paradigm of Lie-group methods

Our point of departure is the differential equation (2), evolving in the *homogeneous space* $\mathcal{M}$. We denote the underlying group action by Λ. Our goal is to characterise (2) and subsequently reformulate it as an equation evolving in $\mathcal{G}$ but first we must introduce another concept, that of a *Lie algebra.* A Lie algebra is a linear space $\mathfrak{g}$, closed under *commutation* $[\,\cdot\,,\,\cdot\,] : \mathfrak{g} \times \mathfrak{g} \to \mathfrak{g}$, subject to the conditions

$$\left.\begin{array}{l} [X_1, X_2] + [X_2, X_1] = \mathbf{O} \\ [a_1 X_1 + a_2 X_2, X_3] = a_1 [X_1, X_3] + a_2 [X_2, X_3] \\ [X_1, [X_2, X_3]] + [X_3, [X_1, X_2]] + [X_2, [X_3, X_1]] = \mathbf{O} \end{array}\right\} \quad \begin{array}{l} X_1, X_2, X_3 \in \mathfrak{g}, \\ a_1, a_2 \in \mathbb{R}. \end{array}$$

It is well known that the linear space of all tangents to $\mathbf{I} \in \mathcal{G}$ can be endowed with Lie-algebraic structure, provided that we define

$$[X_1, X_2] = \frac{\partial^2}{\partial t_1 \partial t_2} \rho_1(t_1) \rho_2(t_2) \rho_1(-t_1) \bigg|_{t_1 = t_2 = 0}, \qquad X_1, X_2 \in \mathfrak{g},$$

where each ρ_k is a smooth curve on $\mathcal{G}$ such that $\rho_k(0) = \mathbf{I}$ and $\rho_k'(0) = X_k$. If $\mathfrak{g}$ is finite-dimensional then by the Ado theorem it is always isomorphic to a Lie algebra of matrices, whereby $[\,\cdot\,,\,\cdot\,]$ becomes the standard matrix commutator [31].

We follow in our exposition a general pattern, pioneered by Munthe-Kaas and Zanna in [28]. Given a group action Λ, we set

$$\lambda(X)(\boldsymbol{y}) = \frac{\mathrm{d}}{\mathrm{d}t} \Lambda(\rho(t), \boldsymbol{y}) \bigg|_{t=0}, \qquad X \in \mathfrak{g}, \quad \boldsymbol{y} \in \mathcal{M},$$

where ρ is a smooth curve on $\mathcal{G}$ such that $\rho(0) = \mathbf{I}$, $\rho'(0) = X$. The function λ maps $\mathfrak{g} \times \mathcal{M}$ into $\mathfrak{X}(\mathcal{M})$, the set of all *vector fields* on $\mathcal{M}$. Since vector fields describe all possible values of derivatives of flows evolving on a manifold, we deduce that a generic equation in the homogeneous space $\mathcal{M}$ can be always written in the form

$$\boldsymbol{y}' = \lambda(B(t, \boldsymbol{y}))(\boldsymbol{y}), \quad t \geq 0, \qquad \boldsymbol{y}(0) = \boldsymbol{y}_0 \in \mathcal{M}, \tag{6}$$

where $B : \mathbb{R}_+ \times \mathcal{M} \to \mathfrak{g}$ is sufficiently smooth. The solution of (6) can be written explicitly in terms of the group action

$$\boldsymbol{y}(t) = \Lambda(Q(t), \boldsymbol{y}_0), \qquad \text{where} \qquad Q' = B(t, \Lambda(Q, \boldsymbol{y}_0))Q, \;\; t \geq 0, \quad Q(0) = \mathbf{I}. \tag{7}$$

The main idea is to reduce (6) to the Lie-group equation (7): instead of approximating $\boldsymbol{y}(t)$, we seek an approximate group action $Q(t)$ that carries $\boldsymbol{y}_0$ to $\boldsymbol{y}(t)$. Thus, provided we can discretise in a Lie group, we can respect the structure of any homogeneous space acted upon by the group in question.

Revisiting the two examples of rotation-driven flows from Section 1, we note that in each case we have an $\mathrm{SO}(n)$ action and the corresponding Lie algebra is $\mathfrak{so}(n)$.

Reducing the problem of homogeneous-space discretization to the underlying Lie group is usually just the first step. In general, $\mathcal{G}$ is a nonlinear object and it is highly unlikely that classical numerical solvers can be applied efficiently in a manner that respects Lie-group structure: symplectic (hence expensive!) Runge–Kutta methods can be used in $\mathrm{O}(n), \mathrm{SO}(n), \mathrm{U}(n)$ and $\mathrm{Sp}(n)$, but no classical method can respect $\mathrm{SL}(n)$ structure for $n \geq 3$ [21]. However, *it is possible to transform the problem further so that it evolves in a linear space,* thereby rendering it into a form suitable for efficient numerical discretization. The linear space in question is none other but the Lie algebra $\mathfrak{g}$.

A fundamental feature of Lie groups and Lie algebras is that the *exponential map* takes $\mathfrak{g}$ to $\mathcal{G}$. In the case of matrix Lie algebras, this map is simply the familiar exponential of a matrix. To solve the Lie-group equation

$$Q' = A(t,Q)Q, \quad t \geq 0, \qquad Q(0) = Q_0 \in \mathcal{G}, \tag{8}$$

where $A : \mathbb{R}_+ \times \mathcal{G} \to \mathfrak{g}$ (note that (7) is a special case), we thus represent $Q(t) = e^{\Omega(t)}Q_0$ and, instead of Q, seek the Lie-algebra function Ω. According to a classical result of Hausdorff, the equation (8) can be transformed to the Lie-algebraic equation

$$\Omega' = \sum_{m=0}^{\infty} \frac{\mathrm{B}_m}{m!} \mathrm{ad}_\Omega^m A(t, \mathrm{e}^\Omega Q_0), \quad t \geq 0, \qquad \Omega(0) = \mathbf{O}, \tag{9}$$

where $\{\mathrm{B}_m\}_{m\in\mathbb{Z}_+}$ are *Bernoulli numbers,*

$$\sum_{m=0}^{\infty} \frac{\mathrm{B}_m}{m!} z^m = \frac{z}{\mathrm{e}^z - 1},$$

while ad_Ω, the *adjoint operator* in $\mathfrak{g}$, is an iterated commutator,

$$\mathrm{ad}_\Omega^0 B = B, \qquad \mathrm{ad}_\Omega^m B = [\Omega, \mathrm{ad}_\Omega^{m-1} B], \quad m \geq 1.$$

We call (9) the *dexpinv equation* [21]. Note that the computation of right-hand side of (9) requires just commutators and linear combinations, hence it resides in $\mathfrak{g}$. Being a linear space, there is no distinction between $\mathfrak{g}$ and its tangent space and we deduce that Ω indeed evolves in the Lie algebra.

The "main paradigm" of Lie-group methods can thus be expressed as in the following diagram, where $\boldsymbol{y}_M \approx \boldsymbol{y}(t_M)$,

$$\begin{tikzcd}
\boldsymbol{y}_N \arrow[d, "\text{eqn (8)}"] & \boldsymbol{y}' = \boldsymbol{f}(t,\boldsymbol{y}) & \boldsymbol{y}_{N+1} \\
Q_N \arrow[d, "\text{eqn (9)}"] & & Q_{N+1} \arrow[u, "\Lambda(Q_{N+1},\boldsymbol{y}_N)"'] \\
\Omega_N \arrow[rr, "\text{Lie-algebra solver}"] & & \Omega_{N+1}\,. \arrow[u, "\mathrm{e}^{\Omega_{N+1}}Q_N"']
\end{tikzcd} \tag{10}$$

Initial conditions in both (8) and (9) are given at t_N, rather than the origin, but this should cause no confusion.

The advantage of time-stepping in the Lie algebra $\mathfrak{g}$, rather than in either $\mathcal{M}$ or $\mathcal{G}$, is that the latter are *nonlinear* manifolds, while $\mathfrak{g}$ is a *linear space.* As long as the underlying time-stepping procedure requires just linear combinations and commutations, we are certain to remain in the right configuration manifold.

An important remark is that usually the diagram (10) must be traversed in every time step. It is dangerous simply to map (7) to (9) once, integrate a large number of steps in the algebra and only then return to the homogeneous space, since, like all good things in numerical analysis, (10) is valid only for sufficiently small time steps.

Not all Lie-group integrators follow the paradigm (10). Two important counterexamples are the method of *moving frames* of Crouch and Grossman [11] and the Owren–Marthinsen technique of *canonical coordinates of the second kind* [30]. Yet, the pattern (10) is very general and useful. Our exposition in the sequel is thus focussing exclusively on Lie-algebra methods.

3 Lie-algebra methods

3.1 Runge–Kutta–Munthe-Kaas methods

The most obvious means of solving the dexpinv equation (9) is by using Runge–Kutta methods. A general ν-stage explicit Runge–Kutta method, as applied to the ordinary differential system (2), reads

$$\left.\begin{aligned} \boldsymbol{\varphi}_l &= \boldsymbol{y}_N + \sum_{j=1}^{l-1} a_{l,j}\boldsymbol{k}_j \\ \boldsymbol{k}_l &= h\boldsymbol{f}(t_N + c_l h, \boldsymbol{\varphi}_l) \end{aligned}\right\} \qquad l = 1, 2, \ldots, \nu,$$

$$\boldsymbol{y}_{N+1} = \boldsymbol{y}_N + \sum_{l=1}^{\nu} b_l \boldsymbol{k}_l,$$

where $h = t_{N+1} - t_N$ is the step size. As we have already commented, it cannot be reasonably expected to respect the underlying homogeneous-space structure. Instead, we follow Munthe-Kaas [26] in applying the method to the (truncated) dexpinv equation,

$$\left.\begin{aligned} \Phi_l &= \sum_{j=1}^{l-1} a_{l,j} K_j, \qquad A_l = hA(t_N + c_l h, \mathrm{e}^{\Phi_l} Y_N) \\ K_l &= \sum_{m=0}^{p-2} \frac{\mathrm{B}_m}{m!} \mathrm{ad}_{\Phi_l}^m A_l \end{aligned}\right\} \qquad l = 1, 2, \ldots, \nu,$$

$$Y_{N+1} = \exp\left(\sum_{l=1}^{\nu} b_l K_l\right) Y_N,$$

where p is the order of the method. For example, the classical fourth-order Runge–Kutta method with the tableau

$$\begin{array}{c|cccc} 0 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ 1 & 0 & 0 & 1 & \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array} \tag{11}$$

becomes the Runge–Kutta–Munthe-Kaas scheme

$$\begin{aligned}
\Phi_1 &= \mathbf{O}, & A_1 &= hA(t_N, Y_N),\\
K_1 &= A_1, & &\\
\Phi_2 &= \tfrac{1}{2}K_1, & A_2 &= hA(t_N + \tfrac{1}{2}h, \mathrm{e}^{\Phi_2}Y_N),\\
K_2 &= A_2 - \tfrac{1}{2}[\Phi_2, A_2] + \tfrac{1}{12}[\Phi_2, [\Phi_2, A_2]], & &\\
\Phi_3 &= \tfrac{1}{2}K_2, & A_3 &= hA(t_N + \tfrac{1}{2}h, \mathrm{e}^{\Phi_3}Y_N),\\
K_3 &= A_3 - \tfrac{1}{2}[\Phi_3, A_3] + \tfrac{1}{12}[\Phi_3, [\Phi_3, A_3]], & &\\
\Phi_4 &= K_3, & A_4 &= hA(t_N + h, \mathrm{e}^{\Phi_4}Y_N),\\
K_4 &= A_4 - \tfrac{1}{2}[\Phi_4, A_4] + \tfrac{1}{12}[\Phi_4, [\Phi_4, A_4]], & &\\
Y_{N+1} &= \exp(\tfrac{1}{6}K_1 + \tfrac{1}{3}K_2 + \tfrac{1}{3}K_3 + \tfrac{1}{6}K_4)Y_N. & &
\end{aligned}$$

Each step requires four function evaluations, six commutators (this can be reduced to just two by using the graded-space techniques of Subsection 3.3) and the computation of four matrix exponentials.

Implicit Runge–Kutta methods can be subjected to similar treatment.

3.2 Magnus expansions for linear equations

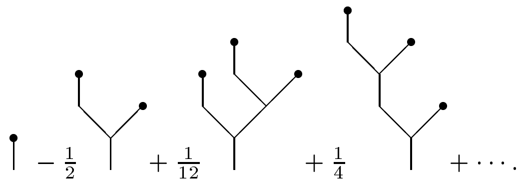
In the special, yet important case when the Lie-group equation is *linear,*

$$Y' = A(t)Y, \quad t \geq 0, \qquad Y(0) = Y_0 \in \mathcal{G},$$

where $A : \mathbb{R}_+ \to \mathfrak{g}$, we can expand the function Ω directly in a manner that has been pioneered by Magnus [23] and systematised as a numerical procedure by Iserles and Nørsett [22]. We commence from the expansion

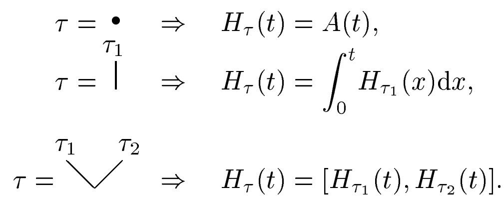
$$\begin{aligned}
\Omega(t) = {} & \int_0^t A(x)\mathrm{d}x - \tfrac{1}{2}\int_0^t\int_0^{x_1}[A(x_2), A(x_1)]\mathrm{d}x_2\mathrm{d}x_1\\
&+ \tfrac{1}{12}\int_0^t\int_0^{x_1}\int_0^{x_1}[A(x_3), [A(x_2), A(x_1)]]\mathrm{d}x_3\mathrm{d}x_2\mathrm{d}x_1\\
&+ \tfrac{1}{4}\int_0^t\int_0^{x_1}\int_0^{x_2}[[A(x_3), A(x_2)], A(x_1)]\mathrm{d}x_3\mathrm{d}x_2\mathrm{d}x_1 + \cdots,
\end{aligned} \tag{12}$$

which can be verified directly from (9). (For ease of notation we assume without loss of generality that $t_N = 0$.) This is the point of departure for the *Magnus expansion* of Ω, which can be most conveniently written in terms of *rooted binary trees.* For example, the expansion (12) corresponds to the trees



(13)

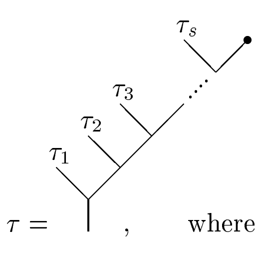
The connection between (12) and (13) is self-explanatory. Thus, place the matrix A at every leaf of a tree and prune, replacing 'vertical' edges by integration and 'diagonal' edges by commutation. More formally, we define a map H_τ from the set of all weakly-binary rooted trees to the Lie algebra $\mathfrak{g}$ by recursion:



Using this approach, pioneered in [22], we can write the Magnus expansion in the form

$$\Omega(t) = \sum_{m=0}^{\infty} \sum_{\tau \in \mathbb{T}_m} \alpha(\tau) H_\tau(t), \qquad t \geq 0. \tag{14}$$

Here $\mathbb{T}_0$ includes just the tree , while for $m \geq 1$ each $\mathbb{T}_m$ is the set of all trees of the form



$$\text{where} \qquad \tau_k \in \mathbb{F}_{r_k}, \quad k = 1, 2, \ldots, s, \quad \sum_{k=1}^{s} r_k = m - 1$$

and $\alpha(\tau) = 1$ for $\tau \in \mathbb{T}_0$,

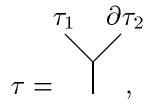
$$\alpha(\tau) = \frac{\mathrm{B}_s}{s!} \prod_{k=1}^{s} \alpha(\tau_k), \qquad \tau \in \mathbb{T}_m, \quad m \geq 1.$$

Note that each tree in $\mathbb{T}_m$ corresponds to a term with $m+1$ integrals [22].

Although (14) yields the explicit form of Ω, this manner of writing the infinite series can be improved. Thus, let $\mathbb{F}_m$ be the subset of all trees $\tau \in \cup_{k \geq 0} \mathbb{T}_k$ such that $H_\tau(t) = O(t^{m+1})$ for all sufficiently-smooth matrix functions A. We rewrite (14) in the form

$$\Omega(t) = \sum_{m=0}^{\infty} \sum_{\tau \in \mathbb{F}_m} \alpha(\tau) H_\tau(t), \qquad t \geq 0. \tag{15}$$

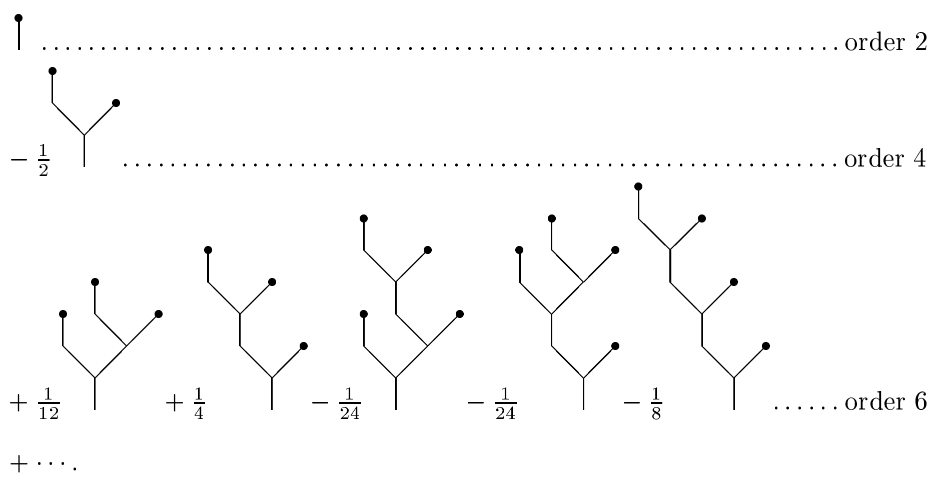
The only difference is in the composition rules for the sets $\mathbb{F}_m$. Clearly, $\mathbb{F}_0 = \mathbb{T}_0$. Otherwise, we note that every tree in our expansion is of the form



where τ_1 and τ_2 are simpler trees and $\partial\tau_2$ means that the root of τ_2 has been removed. In that case, given that $\tau_1 \in \mathbb{F}_{m_1}$ and $\tau_2 \in \mathbb{F}_{m_2}$, it is true that

$$\tau \in \begin{cases} \mathbb{F}_{m_1+m_2}, & \tau_1 \neq \tau_2, \\ \mathbb{F}_{m_1+m_2+1}, & \tau_1 = \tau_2. \end{cases}$$

The advantage of the 'expansion by power' (15) is not just that, once we truncate it so that the error is $O(t^{p+1})$, it contains smaller number of terms, but that such a truncated sum is an *odd* function. Thus, the error is always $O(h^{p+1})$ for *even* p, even if we impose $O(h^p)$ order conditions. Herewith, in the terminology of rooted trees, the 'expansion by power' (15),



3.3 Multivariate quadrature and graded algebras

Practical numerical implementation of Magnus expansions is, on the face of it, fraught with difficulties, since it requires the computation of multivariate integrals. Thus, to advance a single time step with a sixth-order method we need to approximate (to the same order of accuracy) one line integral, one double integral, two triple integrals and three quadruple integrals, each over a different polytope. Fortunately, all this can be accomplished with a surprisingly small number of function evaluations!

Let

$$\mathcal{I}(h) = \int_{h\mathcal{P}} \boldsymbol{L}(A(x_1), A(x_2), \cdots, A(x_r))\mathrm{d}\boldsymbol{x}$$

be a typical integral in the Magnus expansion (15). Here $\mathcal{P}$ is an r-dimensional polytope and $\boldsymbol{L}$ is a multilinear function composed of nested commutators. Following [22], we evaluate the matrix A at a set of ν distinct quadrature points, $A_k = hA(c_k h)$, $k = 1, 2, \ldots, \nu$, and consider the quadrature formula

$$\mathcal{Q}(h) = \sum_{\boldsymbol{i} \in \mathrm{C}_r^\nu} b_{\boldsymbol{i}} \boldsymbol{L}(A_{i_1}, A_{i_2}, \ldots, A_{i_r}), \tag{16}$$

where the sum ranges over the set C_r^ν of all length-r combinations from $\{1, 2, \ldots, \nu\}$ and

$$b_{\boldsymbol{i}} = \int_{\mathcal{P}} \prod_{k=1}^{r} \ell_{i_k}(x_i)\mathrm{d}\boldsymbol{x},$$

ℓ_j being the jth cardinal polynomial of Lagrange's interpolation at $c_1, c_2, \ldots, c_\nu$. Note that the very same function values are recycled time and again in *all* the multivariate integrals in the expansion (15): the overall cost is just ν function evaluations per step! However, the really surprising (and far less trivial) observation is that *the order of the quadrature (16) is precisely the same as the order of the univariate quadrature with the same nodes* $c_1, c_2, \ldots, c_\nu$ [22]. In particular, if we choose the c_ka as Gauss–Legendre quadrature points, we are assured of order 2ν.

The *quid pro quo* for the small number of function evaluations is the large volume of linear algebra. In principle, each combination $\boldsymbol{i}$ requires the evaluation of nested commutators and, needless to say, there are $\nu!/(\nu - r)!$ members of C_r^ν – the outcome might be a prohibitively large overall number of commutators. This is precisely the moment when the magic of Lie-algebra theory steps in to render the method more efficient by entire orders of magnitude. We follow here the analysis of Munthe-Kaas and Owren [27]. Assume that the c_ks are symmetric with respect to $\frac{1}{2}$, as is the case with the two most useful choices, Gauss–Legendre and Gauss–Lobatto, and replace the function values $A_1, A_2, \ldots, A_\nu$ by their linear combinations $B_1, B_2, \ldots, B_\nu$ that solve the Vandermonde system

$$\sum_{l=1}^{\nu} (c_k - \tfrac{1}{2})^{l-1} B_l = A_k, \qquad k = 1, 2, \ldots, \nu.$$

It is easy to confirm that $B_l = O(h^l)$, $l = 1, 2, \ldots, \nu$. The quadrature (16) can now be rewritten in the form

$$\mathcal{Q}(h) = \sum_{\boldsymbol{i} \in \mathrm{C}_r^\nu} \bar{b}_{\boldsymbol{i}} \boldsymbol{L}(B_{i_1}, B_{i_2}, \ldots, B_{i_r}), \quad \text{where} \quad \bar{b}_{\boldsymbol{i}} = \int_{\mathcal{P}} \prod_{k=1}^{r} (x_k - \tfrac{1}{2})^{i_k - 1}\mathrm{d}x. \tag{17}$$

The reformulation (17) has three important advantages: two mechanisms that allow us to throw away most of the terms and one that allows us to express the remaining terms using a surprisingly small basis. We commence by noting that each $\boldsymbol{L}(B_{i_1}, B_{i_2}, \ldots, B_{i_r})$ resides in the *free Lie algebra* $\mathcal{B}_\nu$ generated by the terms $\{B_1, B_2, \ldots, B_\nu\}$, since it can be obtained from the latter by commutation. Recalling that $B_l = O(h^l)$, we say that B_l is of *grade* l, denoting this by $\omega(B_l) = l$, and propagate the grades throughout $\mathcal{B}_\nu$ in a natural manner, $\omega([X_1, X_2]) = \omega(X_1) + \omega(X_2)$. Thus, intuitively, a grade tells us what is the size of each object in $\mathcal{B}_\nu$ and we are free to throw away in (17) all terms of grade exceeding p, the desired order of the method. Since

$$\omega(\boldsymbol{L}(B_{i_1}, B_{i_2}, \ldots, B_{i_r})) = \sum_{k=1}^{r} i_k = |\boldsymbol{i}|,$$

we thus replace (17) with

$$\mathcal{Q}^*(h) = \sum_{\substack{\boldsymbol{i} \in \mathrm{C}_r^\nu \\ |\boldsymbol{i}| \le p}} \bar{b}_{\boldsymbol{i}} \boldsymbol{L}(B_{i_1}, B_{i_2}, \ldots, B_{i_r}). \tag{18}$$

Next, the symmetry of the quadrature points with respect to $\frac{1}{2}$, in tandem with the fact that truncated expansion by power (15) is an odd faction, imply that linear combinations of terms with *even grades* are nil, hence can be expunged from (18) with impunity. This yields

$$\mathcal{Q}^{**}(h) = \sum_{\substack{\boldsymbol{i} \in \mathrm{C}_r^\nu \\ |\boldsymbol{i}| \le p,\ |\boldsymbol{i}| \text{ odd}}} \bar{b}_{\boldsymbol{i}} \boldsymbol{L}(B_{i_1}, B_{i_2}, \ldots, B_{i_r}). \tag{19}$$

The set $\mathcal{K}_\nu^r \subset \mathcal{B}_\nu$ of all elements of grade r is a linear space, whose (surprisingly small) dimension has been determined in [27], in tandem with a constructive means of deriving its basis. Thus, our ultimate step is to replace each $\boldsymbol{L}(B_{i_1}, B_{i_2}, \ldots, B_{i_r})$ in (19) by a linear combination from such a basis. For an order-6 method this reduces the number of commutators from 80 to 7, while for order 8 the saving is from 3304 down to 22.

Specifically, the sixth-order Magnus method is

$$\begin{aligned}\Omega(h) \approx B_1 &+ \tfrac{1}{12} B_3 - \tfrac{1}{12}[B_1, B_2] + \tfrac{1}{240}[B_2, B_3] + \tfrac{1}{360}[B_1, [B_1, B_3]] - \tfrac{1}{240}[B_1, [B_1, B_2]] \\ &+ \tfrac{1}{720}[B_1, [B_1, [B_1, B_2]]],\end{aligned}$$

where

$$A_1 = hA((\tfrac{1}{2} - \tfrac{\sqrt{15}}{10})h), \qquad A_2 = hA(\tfrac{1}{2}h), \qquad A_3 = hA((\tfrac{1}{2} + \tfrac{\sqrt{15}}{10})h)$$

and

$$B_1 = A2, \qquad B_2 = \tfrac{\sqrt{15}}{3}(A_3 - A_1), \qquad B_3 = \tfrac{10}{3}(A_3 - 2A_2 + A_1).$$

Yet, the extent of our savings is not complete! Aggregating commutators in a clever manner, Blanes, Casas and Ros [1] demonstrated that just four commutators are required for a sixth-order method,

$$\begin{aligned} V_1 &= -\tfrac{1}{12}[B_1 + \tfrac{1}{20}B_3, B_2], \\ V_2 &= [B_1, [B_1, \tfrac{1}{360}B_3 - \tfrac{1}{60}V_1]] - \tfrac{1}{20}[B_2, V_1], \\ \Omega(h) &\approx B_1 + \tfrac{1}{12}B_3 + V_1 + V_2. \end{aligned}$$

3.4 Magnus expansions for double-bracket flows

Magnus expansion can be generalised to dexpinv equations that correspond to *nonlinear* Lie-group equations (8), but the outcome is itself nonlinear [33]. They fall outside the limited scope of this brief review. Yet, the solution of certain important homogeneous-space equations can be expanded *explictly* into Magnus series, nonlinearity notwithstanding. In this subsection, based on the very recent paper [19], we describe such an expansion for the *double-bracket equation*

$$Y' = [[Y, N], Y], \quad t \ge 0, \qquad Y(0) = Y_0 \in \mathrm{Sym}(n), \tag{20}$$

where $N \in \mathrm{Sym}(n)$ is a given fixed matrix [3, 10]. We identify (20) at once as a special case of an isospectral flow (3). It is possible to prove that among all $X \in \mathrm{Sym}(n)$ which are orthogonally similar to Y_0, fixed points of (20) locally minimise the quantity $\|X - N\|_{\mathsf{F}}$, where $\|\cdot\|_{\mathsf{F}}$ is the Frobenius norm [3]. Such equations also feature in a wide range of problems in mechanics [2].

Translating (20) to the Lie algebra $\mathfrak{so}(n)$ in line with the diagram (10) results in the isospectral algebra action $Y(t) = \mathrm{e}^{\Omega(t)} Y_0 \mathrm{e}^{-\Omega(t)}$, where

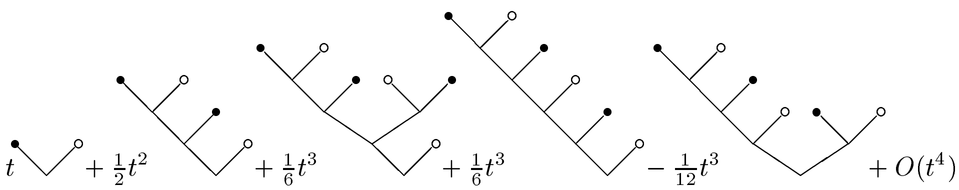
$$\Omega' = \sum_{m=0}^{\infty} \frac{\mathrm{B}_m}{m!} \mathrm{ad}_{\Omega}^m [\mathrm{e}^{\mathrm{ad}_\Omega} Y_0, N], \quad t \ge 0, \qquad \Omega(0) = \mathbf{O}. \tag{21}$$

Although this equation falls outside the scope of 'plain' Magnus expansions, we note that, regardless of what Ω might be, it necessarily depends on just two quantities, the matrices Y_0 and N. Moreover, the Taylor-expansion coefficients of Ω, say, must be obtainable from Y_0 and N by just two kinds of operations, commutation and taking linear combinations. Incidentally, we are *not* dealing here with a standard free Lie algebra, since $Y_0, N \notin \mathfrak{so}(n)$. Yet, it is possible to obtain skew-symmetric matrices from symmetric ones consistently with the rules of our game, e.g. $[Y_0, N] \in \mathfrak{so}(n)$.

A recursive algorithm for the generation of Taylor coefficients of Ω has been presented in [19]. The main idea is to index the expansion by *strictly-binary trees with bicolour leaves,* where

$$\tau = \bullet \quad \Rightarrow \quad H_\tau = Y_0, \qquad\qquad \tau = \circ \quad \Rightarrow \quad H_\tau = N.$$

The trees are pruned exactly like 'plain' Magnus trees, except that there is no integration, only commutation (hence the trees are strictly binary). The algorithm itself is considerably more complicated and here we present just the first few terms in the Taylor expansion of Ω,



It can be generalised to other nonlinear Lie-group and homogeneous-space equations.

3.5 The Cayley transformation

An unstated assumption in the diagram (10) is that the group action Λ and the exponential function are computed to machine accuracy. While exact calculation of Λ hardly ever represents difficulty, this is not the case with an exponential of a matrix. The obvious remedy is to approximate the exponential by, say, a rational function or a Krylov-subspace projection. Doing so, however, we should bear in mind the purpose of the exercise: to discretise differential equations in the right configuration space. Unless our approximation takes the underlying Lie algebra $\mathfrak{g}$ to 'its' Lie group $\mathcal{G}$, this goal cannot be attained. Standard methods for the computation of the matrix exponential, whether the 'nineteen dubious ways' of Moler and Van Loan [25] or numerous other dubious ways, cannot in general be guaranteed to fulfil this goal [7]. In Section 4 we discuss approximation methods that map Lie algebras into Lie groups. In the present subsection we briefly describe an alternative coordinatisation of $\mathcal{G}$ in terms of $\mathfrak{g}$ that does away with the exponential.

Many Lie groups are of the form

$$\mathcal{G} = \{X \in \mathrm{GL}(n) \,:\, XJ\mu(X^{-1}) = J\}, \tag{22}$$

where the *general linear group* $\mathrm{GL}(n)$ consists of all $n \times n$ nonsingular matrices, $J \in \mathrm{GL}(n)$ is given and $\mu : \mathrm{GL}(n) \to \mathrm{GL}(n)$ is an *involutive automorphism:* $\mu(X_1X_2) = \mu(X_1)\mu(X_2)$ and $\mu(\mu(X)) = X$. The most familiar nontrivial special case is $\mu(X) = (X^\top)^{-1}$, $J = \mathbf{I}$, whereby $\mathcal{G} = \mathrm{O}(n)$. Another well-known instances are symplectic and Lorenz groups, both with $\mu(X) = (X^\top)^{-1}$, but the reader can easily find more exotic examples. Whenever $\mathcal{G}$ is defined by (22), the corresponding Lie algebra is

$$\mathfrak{g} = \{X \in \mathfrak{gl}(n) \,:\, XJ - J\mathrm{d}\mu(X) = \mathbf{O}\},$$

where $\mathfrak{gl}(n)$ is the set of all $n \times n$ matrices and

$$\mathrm{d}\mu(X) = \lim_{\varepsilon \to 0} \frac{1}{\varepsilon}[\mu(\mathbf{I} + \varepsilon X) - 1]$$

is the differential of μ. Moreover, in that case it is easy to prove, generalising the approach of [7], that an analytic function f, $f(0) = 1$, takes $\mathfrak{g}$ to $\mathcal{G}$ if and only if $f(z)f(-z) \equiv 1$. $f(z) = e^z$ is an obvious special case, but so is the *Cayley transformation* $f(z) = (1 + \frac{1}{2}z)/(1 - \frac{1}{2}z)$. Replacing the exponential by Cayley results, as a matter of fact, in much simpler Lie-algebraic equation: in place of the dexpinv equation (9), we obtain the *dcayinv equation*

$$\Omega' = A - \tfrac{1}{2}[\Omega, A] - \tfrac{1}{4}\Omega A \Omega, \quad t \geq 0, \qquad \Omega(0) = \mathbf{O}, \tag{23}$$

where $A = A(t, (I - \frac{1}{2}\Omega)^{-1}(I + \frac{1}{2}\Omega)Q_0)$.

In line with the Runge–Kutta–Munthe-Kaas methods of Subsection 3.1, we may solve (23) with an arbitrary explicit Runge–Kutta method, an approach that has been developed by Engø in [14]. If the underlying Lie-group equation is linear, it is possible to introduce a Cayley equivalent of Magnus expansions [18]. We omit here the precise details of underlying methods, just observing that, in general, it is much cheaper to solve a linear system than to compute (or even to approximate, *a la* Section 4) a matrix exponential. Therefore, the Cayley coordinatisation, whenever applicable, offers considerably more economical route toward the computation of Lie-group equations for high-dimensional systems.

4 Computing the exponential of a matrix

We have already commented on the difficulty of approximating e^X for $X \in \mathfrak{g}$ so that the outcome resides in $\mathcal{G}$. For example, if $\mathcal{G} = \mathrm{SL}(n)$, $n \geq 3$, then it is easy to prove that the only function f, analytic near the origin, such that $f(0) = 1$, $f'(0) = 1$ and $f(X) \in \mathrm{SL}(n)$ for all $X \in \mathfrak{sl}(n)$ is the (exact) exponential [21]. Yet, if, like in (22), the Lie group is induced by involutive automorphism then any analytic function f such that $f(z)f(-z) \equiv 1$ is suitable for our purposes, in particular a *diagonal Padé approximation.*

It is, however, possible to devise approximations of the exponential that are assured to respect Lie-group structure. The underlying principle is to use a *splitting,* namely to approximate

$$\mathrm{e}^{tX} \approx \mathrm{e}^{tV_1}\mathrm{e}^{tV_2}\cdots\mathrm{e}^{tV_s} \tag{24}$$

for some $s \geq 1$, subject to the following conditions: $A = \sum_{l=1}^{s} V_l$, $V_l \in \mathfrak{g}$ for all $l = 1, 2, \ldots, s$, it is easy to compute *exactly* each e^{tV_l} and to multiply such exponentials and, finally, the approximation error is consistent with the error of the Lie-algebra method. Methods like this have been presented in [7]. The main idea is to choose matrices V_l of low rank. For example, for a first-order method in the special linear group, we commence from $X \in \mathfrak{sl}(n)$, the Lie algebra of $n \times n$ matrices with zero trace, set $s = n$ and

$$X^{[0]} = X - \operatorname{diag} X, \qquad \left.\begin{array}{l} V_i = \boldsymbol{b}_i\boldsymbol{e}_i^\top + \boldsymbol{e}_i\boldsymbol{a}_i^\top, \\ X^{[i]} = X^{[i-1]} - V_i, \end{array}\right\} \qquad i = 1, 2, \ldots, n,$$

where $\boldsymbol{a}_i^\top$ is the ith row and $\boldsymbol{b}_i$ the ith column of $X^{[i-1]}$ (the leading $i-1$ rows and columns of $X^{[i-1]}$ are all zero), while $\boldsymbol{e}_i \in \mathbb{R}^n$ is the ith unit vector. Each V_i is of rank at most two, hence its exponential is very easy to calculate exactly, and it resides in $\mathfrak{sl}(n)$, as required. Higher-order methods of this type have been developed in [7]. An important means to generate high-order methods of this kind is to use *time-symmetric splitting,* i.e. requiring that $V_{s+1-l} = V_l$, $l = 1, 2, \ldots, s$, followed by the *Yoshida device* [32]. Each application of the latter increases the order by two.

Low-rank methods have been generalised and greatly improved recently in [29] by associating them into a natural 'segmentation' of a Lie algebra.

An alternative point of view has been developed in [8] and it is closely allied with the canonical coordinates of the second kind [30], which have been already mentioned in Section 2. Let $s = \dim \mathfrak{g}$ and let $\boldsymbol{V} = \{V_1, V_2, \ldots, V_s\}$ be a basis of the Lie algebra. In place of (24) we seek a splitting of the form

$$\mathrm{e}^{tX} \approx \mathrm{e}^{\alpha_1(t)V_1} \mathrm{e}^{\alpha_2(t)V_2} \cdots \mathrm{e}^{\alpha_s(t)V_s}, \tag{25}$$

where $\alpha_1, \alpha_2, \ldots, \alpha_s$ are polynomials, chosen to satisfy suitable order conditions. Since $\boldsymbol{V}$ is a basis, there exist constants $c_{i,j}^k$, $i, j, k = 1, 2, \ldots, s$, such that

$$[V_i, V_j] = \sum_{k=1}^{s} c_{i,j}^k V_k, \qquad i, j = 1, 2, \ldots, s.$$

They are called the *structure constants* of $\boldsymbol{V}$ [31] and provide the right tool to impose order conditions on (25). Let

$$X = \sum_{k=1}^{s} a_k V_k.$$

Then for order one we need to set

$$\alpha_k(0) = 0, \quad \alpha_k'(0) = a_k, \qquad k = 1, 2, \ldots, s$$

while the order-two condition additionally requires

$$\alpha_k''(0) = \sum_{i=2}^{s} \sum_{j=1}^{i-1} a_i c_{i,j}^k a_j, \qquad k = 1, 2, \ldots, s. \tag{26}$$

For a general basis $\boldsymbol{V}$, this requires $\approx \frac{1}{2}s^3$ products, which is prohibitively large. However, this number can be reduced a very great deal by choosing a *Chevalley basis* of the Lie algebra. This ensures that structure constants are very sparse and most of the terms in (26) predictably vanish. For example, in $\mathfrak{sl}(n)$ the expense goes down from $O(n^6)$ to $O(n^3)$ products.

Higher-order approximations have been presented in [8], as well as time-symmetric splittings of this kind. The latter allow the use of the Yoshida device.

5 Conclusions

'Stability' is a nebulous and ill-defined term, it means different things to different people. Yet, most of these meanings boil down to 'structural stability', the robustness of a system subject to (sufficiently small) perturbations. It is, however, legitimate to ask *can we avoid certain perturbations altogether?* This is one of the underlying goals of *geometric integration,* the relatively new branch of computational mathematics aiming to develop, analyse and implement discretizations of differential equations that share important structural attributes of the underlying system [4, 5, 16, 24]. Alongside Lie symmetries, symplectic and Lie–Poisson structures etc., one of central themes of geometric integration is the recovery of invariants and strong integrals. An alternative phrasing of invariance is that a differential equation evolves on a manifold, and the latter is very often a homogeneous space. In this paper we have reviewed some of the ideas, methods and theory that have been recently developed in this context. This is the place to caution the reader that, like all other computational tools, Lie-group methods have their limitations and should be used only when justified by the imperative to respect underlying geometric structure or similar considerations.

The pace of developments in geometric integration is very fast indeed and any review is necessarily just a snapshot of a fast-changing and evolving picture. This is why we have attempted to address in this survey the underlying (and probably more enduring) principles and paradigms, rather than describing specific methods and their implementation in great detail. Having said this, Lie-group algorithms already in existence are powerful and they are being applied in a growing number of areas. `DiffMan,` an object-oriented suite of MATLAB programs [15], is freely available for those wishing to experiment with Lie-group methods and translate the material of this paper to concrete numbers and figures.

Bibliography

[1] S. Blanes, F. Casas, and J. Ros. Improved high order integrators based on the Magnus expansion. *BIT*, 40:434–450, 2000.

[2] A. Bloch, P. S. Krishnaprasad, J. E. Marsden, and T. S. Ratiu. The Euler–Poincaré equations and double bracket dissipation. *Comm Math. Phys.*, 175:1–42, 1996.

[3] R. W. Brockett. Dynamical systems that sort lists, diagonalize matrices and solve linear programming problems. *Lin. Alg. Applics*, 146:79–91, 1991.

[4] C. J. Budd and A. Iserles. Geometric integration: Numerical solution of differential equations on manifolds. *Philosophical Trans. Royal Soc. A*, 357:945–956, 1999.

[5] C. J. Budd and M. D. Piggott. Geometric integration and its applications. In F. Cucker, editor, *Handbook of Numerical Analysis.* North Holland, Amsterdam, 2001. To appear.

[6] M. P. Calvo, A. Iserles, and A. Zanna. Numerical solution of isospectral flows. *Maths Comput.*, 66:1461–1486, 1997.

[7] E. Celledoni and A. Iserles. Approximating the exponential from a Lie algebra to a Lie group. *Maths Comput.*, 69:1457–1480, 2000.

[8] E. Celledoni and A. Iserles. Methods for the approximation of the matrix exponential in a Lie-algebraic setting. *IMA J. Num. Anal.*, 21:463–488, 2001.

[9] M. T. Chu. A list of matrix flows with applications. In *Hamiltonian and Gradient Flows, Algorithms and Control*, volume 3 of *Fields Inst. Commun.*, pages 87–97. Amer. Math. Soc., Providence, RI, 1994.

[10] M. T. Chu and K. R. Driessel. The projected gradient method for least squares matrix approximations with spectral constraints. *SIAM J. Numer. Anal.*, 27:1050–1060, 1990.

[11] P. E. Crouch and R. Grossman. Numerical integration of ordinary differential equations on manifolds. *J. Nonlinear Sci.*, 3:1–33, 1993.

[12] L. Dieci and T. Eirola. Applications of smooth orthogonal factorizations of matrices. In *Numerical methods for bifurcation problems and large-scale dynamical systems (Minneapolis, MN, 1997)*, volume 119 of *IMA Vol. Math. Applcs*, pages 141–162. Springer Verlag, New York, 2000.

[13] L. Dieci, R. D. Russell, and E. S. van Vleck. Unitary integrators and applications to continuous orthonormalization techniques. *SIAM J. Num. Anal.*, 31:261–281, 1994.

[14] K. Engø. On the construction of geometric integrators in the RKMK clas. *BIT*, 40:41–61, 2000.

[15] K. Engø, A. Marthinsen, and H. Munthe-Kaas. *DiffMan — an object oriented MATLAB toolbox for solving differential equations on manifolds*, 1.6 edition, December 1998. Available on WWW at the URL `http://www.math.ntnu.no/num/diffman/`.

[16] E. Hairer. Geometric Integration. Unpublished notes, 1999.

[17] A. Iserles. On the global error of discretization methods for highly-oscillatory ordinary differential equations. Technical Report 11, DAMTP, University of Cambridge, 2000.

[18] A. Iserles. On Cayley-transform methods for the discretization of Lie-group equations. *Found. Comp. Maths*, 1:129–160, 2001.

[19] A. Iserles. On the discretization of double-bracket flows. Technical Report 8, DAMTP, University of Cambridge, 2001.

[20] A. Iserles. Think globally, act locally: solving highly-oscillatory ordinary differential equations. Technical Report 6, DAMTP, University of Cambridge, 2001.

[21] A. Iserles, H. Z. Munthe-Kaas, S. P. Nørsett, and A. Zanna. Lie-group methods. *Acta Numerica*, 9:215–365, 2000.

[22] A. Iserles and S. P. Nørsett. On the solution of linear differential equations in Lie groups. *Phil. Trans Royal Society A*, 357:983–1020, 1999.

[23] W. Magnus. On the exponential solution of differential equations for a linear operator. *Comm. Pure and Appl. Maths*, VII:649–673, 1954.

[24] R. I. McLachlan and G. R. W. Quispel. Six lectures on the geometric integration of ODEs. In R. A. DeVore, A. Iserles, and E. Sűli, editors, *Foundations of Computational Mathematics*, number 284 in London Mathematical Society Lecture Notes, pages 155–210, Cambridge, 2001. Cambridge University Press.

[25] C. Moler and C. F. Van Loan. Nineteen dubious ways to compute the exponential of a matrix. *SIAM Rev.*, 20:801–836, 1978.

[26] H. Munthe-Kaas. Runge–Kutta methods on Lie groups. *BIT*, 38:92–111, 1998.

[27] H. Munthe-Kaas and B. Owren. Computations in a free Lie algebra. *Phil. Trans Royal Society A*, 357:957–982, 1999.

[28] H. Munthe-Kaas and A. Zanna. Numerical integration of differential equations on homogeneous manifolds. In F. Cucker and M. Shub, editors, *Foundations of Computational Mathematics*, pages 305–315. Springer Verlag, 1997.

[29] H.Z. Munthe-Kaas, G.R.W. Quispel, and A. Zanna. Generalized polar decompositions on Lie groups with involutive automorphisms. *Found. Comp. Maths*, 2001.

[30] B. Owren and A. Marthinsen. Integration methods based on canonical coordinates of the second kind. *Muner. Math.*, 87:763–790, 2001.

[31] V. S. Varadarajan. *Lie Groups, Lie Algebras, and Their Representations.* Graduate Texts in Mathematics 102. Springer-Verlag, 1984.

[32] H. Yoshida. Construction of higher order symplectic integrators. *Physics Letters A*, 150:262–268, 1990.

[33] A. Zanna. Collocation and relaxed collocation for the Fer and the Magnus expansions. *SIAM J. Num. Anal.*, 36:1145–1182, 1999.