BIT 35 (1995), 572–587.

LIE-BUTCHER THEORY FOR RUNGE-KUTTA METHODS *

HANS MUNTHE–KAAS[1]

[1] *Department of Informatics, University of Bergen, Thormøhlensgt. 55 N-5020, Norway. email: Hans.Munthe-Kaas@ii.uib.no*

Abstract.

Runge–Kutta methods are formulated via coordinate independent operations on manifolds. It is shown that there is an intimate connection between Lie series and Lie groups on one hand and Butcher's celebrated theory of order conditions on the other. In Butcher's theory the elementary differentials are represented as trees. In the present formulation they appear as commutators between vector fields. This leads to a theory for the order conditions, which can be developed in a completely coordinate free manner. Although this theory is developed in a language that is not widely used in applied mathematics, it is structurally simple. The recursion for the order conditions rests mainly on three lemmas, each with very short proofs. The techniques used in the analysis are prepared for studying RK–like methods on general Lie groups and homogeneous manifolds, but these themes are not studied in detail within the present paper.

AMS subject classification: 65L06.

Key words: Butcher theory, Runge–Kutta methods, manifolds, Lie groups, Lie series, Lie algebras.

1 Introduction.

The theory of differential equations has diverged in two different directions in our century; the pure mathematical abstract presentation based on coordinate free formulations, and the applied mathematical presentation based on concrete coordinate representations. In previous papers, we have shown that coordinate free formulations can be very useful also in areas of applied mathematics; both as a tool for structuring numerical software (object oriented design) [10], and as a tool for developing new numerical algorithms [11].

In the present paper we use coordinate free techniques to analyze Runge–Kutta (RK) methods for solving ordinary differential equations. Our motivation is twofold:

1. We want to understand to what degree the RK process is depending on a particular coordinate formulation, and the fact that the domain in the classical formulation is a vector space.

*Received March 1995. Revised October 1995.

2. We are later interested in developing specialized RK methods for ODEs possessing various symmetries and conservation laws (e.g., symplectic systems, isospectral systems and systems whose solution is invariant under a Lie group of transformations). Such symmetries and conservation laws are most easily studied in a coordinate free setting. We believe that the analysis of numerical schemes would be simpler if the numerical scheme could be expressed and analyzed in a coordinate free form and the symmetries are introduced at this level, rather than by bringing the symmetries down to a particular coordinate representation and perform a classical coordinate dependent analysis.

In this paper we complete the task of developing the order conditions of classical RK methods in a coordinate free framework, and establish the connection between basic Lie group techniques and Butcher's order theory. The techniques we develop are also prepared for studying generalizations of classical RK schemes, such as RK methods based on non–commuting flows. It is a goal to keep the paper at a level where it can be read by people without previous knowledge of Lie group techniques. We have therefore deliberately avoided a detailed discussion of possible generalizations.

The Butcher theory of order conditions was originally developed in [4], see [7] for a thorough exposition of this theory. An alternative approach is given by Albrecht in [2]. Applications of Lie brackets in Runge–Kutta methods is discussed by several authors especially in the field of symplectic integration, see the book of Calvo and Sanz–Serna [13]. Numerical integration of ODEs on Lie groups is of considerable interest in computational mechanics, see [14] and the references therein. Crouch and [6] discuss RK methods on general manifolds in a formulation that is closely related to ours. Their techniques are based on formulations of the algorithms and analysis of order conditions on the Lie group, while our formulation and analysis is performed on the corresponding Lie algebra. The latter approach ties the bonds back to the Butcher theory in the case of abelian Lie groups, and seems to bring major simplifications also into the analysis of more general cases.

2 Basic Lie group techniques.

In this section we will introduce the following basic concepts:

- Differential manifolds, vector fields and flows.
- Pullbacks and Lie series.
- Lie algebras and Lie group actions on a manifold.

We will avoid some general mathematical definitions, and rather interpret these concepts through concrete examples. The examples are sufficient for understanding our development of the Butcher theory in the next section. The interested reader is referred to the texts [1, 12, 16, 3] for a rigorous treatment of basic topics and [9, 15] for advanced topics in Lie group theory.

A *differentiable n–manifold*, $\mathcal{M}$, is a domain supporting functions which everywhere can be differentiated in n different directions. For the purpose of developing the classical Butcher theory, we can think of a manifold as being the n–dimensional real vector space $\mathcal{M} = \mathbb{R}^n$ (or an open subset of this), and regard all translations of $\mathbb{R}^n$ as being a commutative Lie group acting on $\mathcal{M}$. This will henceforth be referred to as *the euclidean example.* Another useful mental image of $\mathcal{M}$ is a smooth n–dimensional hypersurface embedded in $\mathbb{R}^m$, $n < m$. It should, however, be noted that manifolds can be defined independently of a particular embedding in $\mathbb{R}^m$.

We let $\mathcal{F}(\mathcal{M})$ denote the set of all real valued functions:[1]

$$\mathcal{F}(\mathcal{M}) = \{f : \mathcal{M} \to \mathbb{R}\}$$

A *tangent vector* based on a point $p \in \mathcal{M}$ is $t = (p, \vec{v})$, where $\vec{v} \in \mathbb{R}^n$ is tangent to $\mathcal{M}$ in the point p. The set of all tangent vectors at all points is called the *tangent bundle* and is denoted $T\mathcal{M}$. $T\mathcal{M}$ is a $2n$–dimensional manifold. A *vector field* is a map $F : \mathcal{M} \to T\mathcal{M}$ such that $F(p) = (p, \vec{v})$. The set of all vector fields on $\mathcal{M}$ is denoted $\mathfrak{X}(\mathcal{M})$. Two tangent vectors $(p, \vec{u})$ and $(p, \vec{v})$, based on the same point p, can be added by adding their second components as vectors. There is generally no rule for adding two tangent vectors based on different points, unless we specify a process which transports the two vectors to a common basepoint. This process is called a *pullback* and is defined below.

EXAMPLE 2.1. In the euclidean example we have $\mathcal{M} = \mathbb{R}^n$ and $T\mathcal{M} = \mathbb{R}^n \times \mathbb{R}^n$, where the first component denotes the base point of the vector and the second its direction. In classical vector calculus on $\mathbb{R}^n$ it is tacitly assumed that tangent vectors can be brought to a common basepoint by parallel translations, thus one usually forgets about the first component of $T\mathcal{M}$, and the tangent bundle is identified with $\mathbb{R}^n$ itself. Hence classically a vector field is defined as a map $F : \mathbb{R}^n \to \mathbb{R}^n$. Such identifications are unfortunate from an analysis and software specification point of view, since it introduces 'type errors' by identifying objects (coordinate vectors and tangent vectors) which, because they possess different properties, should be treated as being of different type.

A *derivation* on $\mathcal{F}(\mathcal{M})$ is a mapping $D : \mathcal{F}(\mathcal{M}) \to \mathcal{F}(\mathcal{M})$ with the following properties:

- R–linear: $D(f + g) = D(f) + D(g)$ for all $f, g \in \mathcal{F}(\mathcal{M})$
- Leibniz rule: $D(fg) = D(f)g + fD(g)$ for all $f, g \in \mathcal{F}(\mathcal{M})$
- Local operator: $D(f)|_{\mathcal{U}} = D(f|_{\mathcal{U}})$ for all open sets $\mathcal{U} \subset \mathcal{M}$

The following fundamental result shows that we may define a vector field by specifying a derivation operator on $\mathcal{F}(\mathcal{M})$ and vice versa:

PROPOSITION 2.1. *[1](ch.4.2) There is a natural one–to–one correspondence between derivations on $\mathcal{F}(\mathcal{M})$ and vector fields on $\mathcal{M}$. The derivation corresponding to a given vector field $F \in \mathfrak{X}(\mathcal{M})$ is written $F[\_]$, and is called the* Lie derivative *on $\mathcal{F}(\mathcal{M})$ (w.r.t. F).*

[1] *All* functions in this paper are assumed to be infinitely smooth.

The *commutator* is a bilinear function $[\_,\_] : \mathfrak{X}(\mathcal{M}) \times \mathfrak{X}(\mathcal{M}) \to \mathfrak{X}(\mathcal{M})$ which can be defined by its action as an $\mathcal{F}(\mathcal{M})$ derivation. If $H = [F, G]$ then

$$H[f] = F[G[f]] - G[F[f]] \text{ for all } f \in \mathcal{F}(\mathcal{M}).$$

Given a fixed $F \in \mathfrak{X}(\mathcal{M})$, the map $[F,\_] : \mathfrak{X}(\mathcal{M}) \to \mathfrak{X}(\mathcal{M})$ defines a derivation on $\mathfrak{X}(\mathcal{M})$ which is called the Lie derivative on $\mathfrak{X}(\mathcal{M})$ (w.r.t. F).

EXAMPLE 2.2. In the euclidean example, a vector field $F(p) = (p, \vec{F}(p))$ corresponds to the (Lie–) derivation $F[\_]$ given as:

$$F[g](p) = (\nabla g)(p) \cdot \vec{F}(p) \text{ for all } g \in \mathcal{F}(\mathcal{M}),\ p \in \mathcal{M}.$$

Conversely, a derivation $F[\_] : \mathcal{F}(\mathcal{M}) \to \mathcal{F}(\mathcal{M})$ can always be written out in (local) coordinates as:

$$F[g] = \sum_i f^i \frac{\partial g}{\partial x^i} \text{ where } f^i, g \in \mathcal{F}(\mathcal{M}).$$

This corresponds to the Lie derivation on $\mathcal{F}(\mathcal{M})$ defined by the vector field $F(p) = (p, \vec{F}(p))$, where $\vec{F}(p) = \sum_i f^i \vec{e_i}$. We will henceforth use the symbol $\frac{\partial}{\partial x^i}$ both to denote a partial derivative operator, and to denote the constant vector field $\frac{\partial}{\partial x^i}(p) \equiv (p, \vec{e_i})$ for all p. Thus $F = \sum_i f^i \frac{\partial}{\partial x^i}$ is both a derivation and a vector field. If $F = \sum_i f^i \frac{\partial}{\partial x^i}$, $G = \sum_i g^i \frac{\partial}{\partial x^i}$ then

$$[F, G] = \sum_{i,j} \left(f^j \frac{\partial g^i}{\partial x^j} - g^j \frac{\partial f^i}{\partial x^j} \right) \frac{\partial}{\partial x^i} \ . \tag{2.1}$$

Proposition 2.1 is also the basis for relating tangent vectors based on different points. Let $\phi : \mathcal{M} \to \mathcal{M}$ be a diffeomorphism (i.e. a smooth function with a smooth inverse), and let $g \in \mathcal{F}(\mathcal{M})$. We define the *pullback* of g along ϕ, $\phi^* g \in \mathcal{F}(\mathcal{M})$, as:

$$\phi^* g = g \circ \phi$$

We define the *pullback of a vector field* $F \in \mathfrak{X}(\mathcal{M})$, written $\phi^* F \in \mathfrak{X}(\mathcal{M})$, such that derivations commute with pullbacks, i.e.:

$$(\phi^* F)[\phi^* g] = \phi^*(F[g]) \text{ for all } g \in \mathcal{F}(\mathcal{M}) \ . \tag{2.2}$$

Since this equation defines how $\phi^* F$ act as a derivation operator, it must also define a vector field.

EXAMPLE 2.3. In the euclidean example pullback of vector fields is expressed via the Jacobian matrix of ϕ. If $\phi : \mathbb{R}^n \to \mathbb{R}^n$ and $G = \phi^* F$, then we have:

$$J\ \vec{G}(p) = \vec{F}(\phi(p)) \ , \text{ where } J_{i,j} = \frac{\partial \phi_i}{\partial x_j}.$$

Given a vector field $F \in \mathfrak{X}(\mathcal{M})$. An *integral curve* of F is a curve $y : \mathbb{R} \to \mathcal{M}$, $t \mapsto y(t)$, which satisfy the ODE:

$$\mathbf{d}y(t) = F(y(t)) \text{ , where } \mathbf{d}y(t) = (y(t), \tfrac{dy(t)}{dt}) \in T\mathcal{M}.$$

The *flow* of a vector field $F \in \mathfrak{X}(\mathcal{M})$ is a $\mathbb{R}$–indexed family of mappings $\phi_t : \mathcal{M} \to \mathcal{M}$ such that

$$\phi_{t'}(y(t)) = y(t + t') \text{ for all } t, t' \text{ and all integral curves } y \text{ of } F.$$

We will henceforth use the notation $\phi_t = \Psi_{t,F} : \mathcal{M} \to \mathcal{M}$ to denote the flow of a given vector field F. The following fundamental formulas[1](p.271,p.278) relate Lie derivations and pullbacks:

$$\frac{d}{dt}(\Psi^*_{t,F} f) = \Psi^*_{t,F} F[f] \tag{2.3}$$

$$\frac{d}{dt}(\Psi^*_{t,F} G) = \Psi^*_{t,F}[F, G] \tag{2.4}$$

for $F, G \in \mathfrak{X}(\mathcal{M})$, $f \in \mathcal{F}(\mathcal{M})$. Successive application of these formulas yields Taylor series expansions of the pullback:

$$\begin{aligned}
\Psi^*_{t,F} f &= f + \frac{t}{1!}F[f] + \frac{t^2}{2!}F[F[f]] + \frac{t^3}{3!}F[F[F[f]]] + \cdots \\
\Psi^*_{t,F} G &= G + \frac{t}{1!}[F, G] + \frac{t^2}{2!}[F, [F, G]] + \frac{t^3}{3!}[F, [F, [F, G]]] + \cdots
\end{aligned}$$

These are the basic forms of *Lie series* and are the main tool for series developments on general manifolds.[2]

Many numerical algorithms can be viewed as a process where we are able to compute pullbacks along certain 'nice' flows exactly, and use this information to approximate certain operators, (e.g., finite difference approximations of differential operators can be viewed this way). By 'nice' we will mean that the flows act (transitively and effectively) as a Lie group on the manifold. This should be thought of as all flows generated by a set of vector fields which in each point of $\mathcal{M}$ spans all possible directions, and which act essentially in the same manner everywhere. Formally we define these vector fields as:[3]

DEFINITION 2.1. *Let $\mathcal{M}$ be an n–manifold. Given a set of vector fields $W_1, W_2, \ldots, W_n \in \mathfrak{X}(\mathcal{M})$ which satisfy:*

1. $\mathfrak{X}(\mathcal{M}) = \left\{ \sum_{i=1}^n f^i W_i \mid f^i \in \mathcal{F}(\mathcal{M}) \right\}$

[2]Due to these formulas, it is common to find the notation $\Psi^*_{t,F} \equiv \exp(tF)$ in the literature. We will avoid this notation, since we will later work with time dependent vector fields, and in that case the exponential notation may be misleading.

[3]Some readers will note that we, to keep the presentation simple, have avoided the usual abstract definition of Lie groups and Lie algebras. Our definition is close to Sophus Lie's original concept, see [12].

2. *There exist a set of constants* $c_{i,j}^k \in \mathbb{R}$ *such that*

$$[W_i, W_j] = \sum_{k=1}^{n} c_{i,j}^k W_k \text{ for all } i, j.$$

Then the set

$$\mathfrak{g} = \left\{ \sum_{i=1}^{n} c^i W_i \mid c^i \in \mathbb{R} \right\}$$

is called a Lie algebra *of* $\mathcal{M}$, *and the elements of* $\mathfrak{g}$ *are called* infinitesimal generators *(of the Lie group). The set of all flows generated by elements of* $\mathfrak{g}$ *is called a* Lie group action *on* $\mathcal{M}$. *If* $c_{i,j}^k = 0$, *we say that* $\mathfrak{g}$ *is* abelian *(=commutative).*

Note: If $\mathfrak{g}$ is abelian, then there exists a coordinate system on the manifold such that $W_i = \frac{\partial}{\partial x^i}$. Thus the abelian Lie group case is equivalent to the Euclidean case. However, by concentrating our emphasis on abelian Lie groups rather than $\mathbb{R}^n$, we are led to using geometric, coordinate independent, tools in the series developments rather than the coordinate dependent multivariate Taylor series of the classical analysis.

Let us fix an arbitrary point $e \in \mathcal{M}$. Any infinitesimal generator $Y \in \mathfrak{g}$ is completely determined by its value on e, and the commutator between different infinitesimal generators can also be computed in the single point e. Thus $\mathfrak{g}$ has the structure of the real vector space $\mathbb{R}^n$ equipped with the bilinear bracket $[\_,\_] : \mathfrak{g} \times \mathfrak{g} \to \mathfrak{g}$. The Lie algebra $\mathfrak{g}$ should be thought of as a set of special vector fields whose commutators and pullbacks are easily computed. We want to approximate general vector fields by elements in $\mathfrak{g}$:

DEFINITION 2.2. *Given a vector field* $F \in \mathfrak{X}(\mathcal{M})$ *and a fixed point* $e \in \mathcal{M}$, *we let* $\overline{F}$ *denote the unique element of* $\mathfrak{g}$ *whose flow is tangent to the flow of* F *in the point* e, *i.e.:*

$$\overline{F} \in \mathfrak{g} \text{ such that } \overline{F}(e) = F(e) \ .$$

EXAMPLE 2.4. In the euclidean example, the Lie algebra is spanned by the infinitesimal generators $W_i = \frac{\partial}{\partial x^i}$. These generate the set of all translations on $\mathbb{R}^n$. Evidently $\mathfrak{g}$ is abelian, since $[\frac{\partial}{\partial x^i}, \frac{\partial}{\partial x^j}] = 0$. *This fact is actually everything we need to know about the euclidean example to derive the classical Butcher theory!*

Given an arbitrary point $e \in \mathcal{M}$. The approximation $\overline{F}$ of $F = \sum_i f^i W_i \in \mathfrak{X}(\mathcal{M})$ is computed as

$$\overline{F} = \sum_i f^i(e) W_i \ .$$

If $Y \in \mathfrak{g}$ we have:

$$\begin{aligned} \Psi_{t,Y}(p) &= p + tY \\ (\Psi_{t,Y}^* f)(p) &= f(p + tY) \text{ for all } f \in \mathcal{F}(\mathcal{M}). \\ (\Psi_{t,Y}^* F)(p) &= F(p + tY) \text{ for all } F \in \mathfrak{X}(\mathcal{M}) \ , \end{aligned}$$

where we apply the usual identification of $\mathfrak{g}$ with $\mathbb{R}^n$ given as $W_i \mapsto \vec{e_i}$.

3 Lie–Butcher theory.

3.1 Runge–Kutta methods

Classically a single Runge–Kutta step can be formulated as: Let $\mathcal{M} = \mathbb{R}^n$, let $F : \mathbb{R}^n \to \mathbb{R}^n$ be a vector field and let $y(t) \in \mathbb{R}^n$ be an integral curve of F. Given the point $y_0 = y(t_0)$. We find $y_1 \approx y(t_0 + h)$ by the computation:

$$\begin{aligned} K_i &= F(y_0 + Y_i) \text{ , for } i = 1, 2, \ldots, s. \\ Y_i &= h \sum_{j=1}^{s} a_{i,j} K_j \text{ , for } i = 1, 2, \ldots, s. \\ y_1 &= y_0 + h \sum_{j=1}^{s} b_j K_j \text{ ,} \end{aligned}$$

where $K_i, Y_i \in \mathbb{R}^n$ and $a_{i,j}, b_j \in \mathbb{R}$ are constants defining a particular method. The method has order p if

$$\|y_1 - y(t_0 + h)\| = \mathcal{O}(h^{p+1}) \text{ ,}$$

where $\|\cdot\|$ is some norm on $\mathbb{R}^n$. The Butcher theory [4, 7] is a systematic way to write down the algebraic conditions that $a_{i,j}$ and b_j must satisfy for a method to have a given order.

We will restate RK methods in a coordinate free setting. Let $\mathcal{M}$ be a manifold, $F \in \mathfrak{X}(\mathcal{M})$ a vector field and $\mathfrak{g}$ a Lie algebra of $\mathcal{M}$. Let $e = y(t_0) \in \mathcal{M}$, and recall (Def. 2.2) that $\overline{F}$ denotes the approximation in $\mathfrak{g}$ which is tangent to F in the point y_0. A Runge–Kutta step is written as:

$$K_i = \overline{\Psi^*_{t=1,Y_i} F} \text{ , for } i = 1, 2, \ldots, s. \tag{3.1}$$

$$Y_i = h \sum_{j=1}^{s} a_{i,j} K_j \text{ , for } i = 1, 2, \ldots, s. \tag{3.2}$$

$$Y_h = h \sum_{j=1}^{s} b_j K_j \tag{3.3}$$

$$y_1 = \Psi_{t=1,Y_h}(e) \text{ ,} \tag{3.4}$$

where $K_i, Y_i, Y_h \in \mathfrak{g}$ and $y_0, y_1 \in \mathcal{M}$. The method has order p if

$$\left|(\Psi^*_{t=1,Y_h} f - \Psi^*_{h,F} f)(e)\right| = \mathcal{O}(h^{p+1}) \text{ for all } f \in \mathcal{F}(\mathcal{M}).$$

Notes:

1. This formulation coincides with the traditional formulation in the euclidean case.

2. The definition of the order of the method is only depending on the norm $|\cdot|$ on $\mathbb{R}$, and is independent of any metric on $\mathcal{M}$ (which in general may or may not be present).

3. We will only derive the order conditions under the assumption that $\mathfrak{g}$ is abelian, but the main tools are developed for general $\mathfrak{g}$.

4. On a general manifold $\mathcal{M}$ it may be impossible to find infinitesimal generators $W_1, \ldots, W_n$ which satisfies Definition 2.1 globally on the whole of $\mathcal{M}$. For our purpose it is, however, sufficient to find infinitesimal generators which satisfy the definition locally on an open subset $U \subset \mathcal{M}$, containing e, i.e. we are only interested in a *local* Lie group action on $\mathcal{M}$. Such a local Lie group action can always be found. It is even always possible to find a local Lie group action with an abelian Lie algebra $\mathfrak{g}$. There may, however, be situations where it is advantageous to work with a non–abelian Lie algebra $\mathfrak{g}$. (See [6] for a discussion of this.)

The analysis will proceed along the following lines:

1. Find a *time dependent* infinitesimal generator $Z_t \in \mathfrak{g}$ that satisfy
$$\Psi_{t,Z_t}(e) = \Psi_{t,F}(e) \text{ for all } t.$$

2. Given $Y_h \in \mathfrak{g}$ computed by the RK step, find a time dependent infinitesimal generator $\tilde{Z}_t \in \mathfrak{g}$ such that
$$\Psi_{t=1,Y_h}(e) = \Psi_{t=h,\tilde{Z}_t}(e) \text{ for all } h.$$

3. The order conditions arise by equating the first terms in a Lie series development of Z_t and $\tilde{Z}_t$ up to a given order.

3.2 Lie–Butcher series for the exact solution

We will find a special type Lie series expansion of a time dependent infinitesimal generator Z_t with the same integral curve as F through the point e.

LEMMA 3.1. *Let $F \in \mathfrak{X}(\mathcal{M})$ be a vector field and $Z_t \in \mathfrak{g}$ be a time dependent infinitesimal generator. If $\Psi_{t,Z_t}(e) = \Psi_{t,F}(e)$ for all t, then*
$$(\Psi^*_{t,Z_t} Z_t)(e) = (\Psi^*_{t,Z_t} F)(e) \text{ for all } t.$$
If $\mathfrak{g}$ is abelian, then
$$Z_t = \overline{\Psi^*_{t,Z_t} F} \text{ for all } t.$$

PROOF.
$$\Psi_{t,Z_t}(e) = \Psi_{t,F}(e) \Rightarrow \left(\Psi^*_{t,Z_t} f\right)(e) = \left(\Psi^*_{t,F} f\right)(e) \text{ for all } f \in \mathcal{F}(\mathcal{M}),\ t \in \mathbb{R}$$
By (2.2) and (2.3) we have:
$$\begin{aligned} \frac{d}{dt}(\Psi^*_{t,Z_t} f) &= \Psi^*_{t,Z_t}(Z_t[f]) = \left(\Psi^*_{t,Z_t} Z_t\right)[\Psi^*_{t,Z_t} f] \\ \frac{d}{dt}(\Psi^*_{t,F} f) &= \Psi^*_{t,F}(F[f]) = \Psi^*_{t,Z_t}(F[f]) = \left(\Psi^*_{t,Z_t} F\right)[\Psi^*_{t,Z_t} f] \end{aligned}$$

Thus $(\Psi^*_{t,Z_t} Z_t)(e) = (\Psi^*_{t,Z_t} F)(e)$. When $\mathfrak{g}$ is abelian it is a standard result that any element of $\mathfrak{g}$ is invariant under pullback by any flow in the Lie group, hence $Z_t(e) = (\Psi^*_{t,Z_t} F)(e)$. □

In the euclidean example, this lemma is just saying in a roundabout way the obvious fact that $Z_t(q) = F(q)$ in the point $q = \Psi_{t,Z_t}(e)$.

Let $Y \cdot Z$ define a second order derivation operator via:[4]

$$[Y \cdot Z, F] = [Y, Z, F] = [Y, [Z, F]] \quad \text{for } Y, Z \in \mathfrak{g}, \tag{3.5}$$

We then have

$$\frac{d}{dt}(Y \cdot Z) = (\frac{d}{dt}Y) \cdot Z + Y \cdot (\frac{d}{dt}Z).$$

LEMMA 3.2.

$$\frac{d^i}{dt^i}\left(\Psi^*_{t,Z_t} F\right) = \Psi^*_{t,Z_t}\left([B_i(Z_t), F]\right),$$

where $B_i(Z_t)$ are recursively given as:

$$B_1(Z_t) = Z_t \tag{3.6}$$

$$B_{i+1}(Z_t) = Z_t \cdot B_i(Z_t) + \frac{d}{dt} B_i(Z_t) \text{ for } i > 0. \tag{3.7}$$

PROOF. If $G_t \in \mathfrak{X}(\mathcal{M})$ is a time dependent vector field, we have [1](p.285)

$$\frac{d}{dt}(\Psi^*_{t,Z_t} G_t) = \Psi^*_{t,Z_t}\left([Z_t, G_t] + \frac{d}{dt} G_t\right).$$

The lemma follows by applying this formula recursively on $\Psi^*_{t,Z_t} F$. □

In the case where $\mathfrak{g}$ is abelian, the computation of $B_i(Z_t)$ can be done formally as if Z_t were just a scalar function. The first of these are:

$$B_1(Z_t) = Z_t \tag{3.8}$$

$$B_2(Z_t) = Z_t \cdot Z_t + \frac{d}{dt} Z_t = Z_t^2 + Z_t^{(1)} \tag{3.9}$$

$$B_3(Z_t) = Z_t \cdot (Z_t^2 + Z_t^{(1)}) + \frac{d}{dt}(Z_t^2 + Z_t^{(1)}) = Z_t^3 + 3Z_t \cdot Z_t^{(1)} + Z_t^{(2)} \tag{3.10}$$

$$B_4(Z_t) = Z_t^4 + 6Z_t^2 \cdot Z_t^{(1)} + 3Z_t^{(1)} \cdot Z_t^{(1)} + 4Z_t \cdot Z_t^{(2)} + Z_t^{(3)}, \tag{3.11}$$

where $Z_t^i = Z_t \cdot Z_t \cdots Z_t$ and $Z_t^{(i)} = \frac{d^i}{dt^i} Z_t$.

This recursion turns out to be the cornerstone in our explanation of the Butcher theory. It is essentially all we need to recursively generate all the order conditions. Before doing this, we will complete our Lie series development of Z_t and relate this to the classical B–series. From Lemma 3.1 and 3.2 we get:

[4]We may just think of $Y \cdot Z$ as being a formal product defined through the commutator. It is properly defined as belonging to the *enveloping algebra*, $\mathfrak{G}$, see [15, Ch.3.2].

THEOREM 3.3. *Let $F \in \mathfrak{X}(\mathcal{M})$ and let $\mathfrak{g}$ be an abelian Lie algebra for $\mathcal{M}$. If $Z_t \in \mathfrak{g}$ satisfy $\Psi_{t,Z_t}(e) = \Psi_{t,F}(e)$, then*

$$Z_t = \sum_{i=0}^{\infty} \frac{t^i}{i!} Z_0^{(i)} ,$$

where $Z_0^{(i)} = \frac{d^i}{dt^i} Z_t \Big|_{t=0}$ are recursively given as:

$$Z_0^{(0)} = \overline{F} \tag{3.12}$$

$$Z_0^{(i)} = \overline{[B_i(Z_t)|_{t=0}, F]} \quad \text{for } i > 0. \tag{3.13}$$

Note that we may compute all $Z_0^{(i)}$ simply by plugging the previous $Z_0^{(j)}$, $j < i$ into the expressions for $B_i(Z_t)$ given in (3.6),(3.7). This yields:

$$\begin{aligned}
Z_0^{(0)} &= Z_0 = \overline{F} \\
Z_0^{(1)} &= \overline{[B_1(Z_t)|_{t=0}, F]} = \overline{[Z_0, F]} = \overline{[\overline{F}, F]} \\
Z_0^{(2)} &= \overline{[B_2(Z_t)|_{t=0}, F]} = \overline{\left[Z_0^2 + Z_0^{(1)}, F\right]} = \overline{[\overline{F}, \overline{F}, F]} + \overline{\left[\overline{[\overline{F}, F]}, F\right]} \\
Z_0^{(3)} &= \overline{[\overline{F}, \overline{F}, \overline{F}, F]} + 3\overline{\left[\overline{F}, \overline{[\overline{F}, F]}, F\right]} + \overline{\left[\overline{[\overline{F}, \overline{F}, F]}, F\right]} + \overline{\left[\overline{\left[\overline{[\overline{F}, F]}, F\right]}, F\right]}
\end{aligned}$$

The connections to the Butcher theory appear when we use (2.1) to write out the coordinate expressions for each of these commutator brackets in local coordinates. Each bracket corresponds in a one–to–one fashion to an elementary differential represented by a tree in the Butcher theory. We have, e.g.:

$$\begin{aligned}
F &= \sum_j f^j \frac{\partial}{\partial x^j} \\
[\overline{F}, F] &= \sum_{j,k} f^k \, f_k^j \frac{\partial}{\partial x^j} \\
\left[\overline{F}, \overline{[\overline{F}, F]}, F\right] &= \sum_{j,k,l,m} f^m \, f^l f_l^k \, f_{k,m}^j \frac{\partial}{\partial x^j}
\end{aligned}$$

Where the notation is as in [7], i.e. upper indices are vector components, lower are partial derivations. The correspondence between elementary commutators and Butcher trees is shown in Figure 3.1. The recursive structure of the commutators is readily seen. The root of each tree corresponds to the un–barred F in the commutator, and the subtrees enter into the other slots. This shows that our Lie–series expansion of Z_t is really a B–series in disguise![5] We have not found this form of Lie series expansions in the literature on Lie group theory, and will henceforth call it a *Lie–Butcher type series.*

[5] Although we have now given the terms a geometric meaning, we have paved the road for future generalizations.

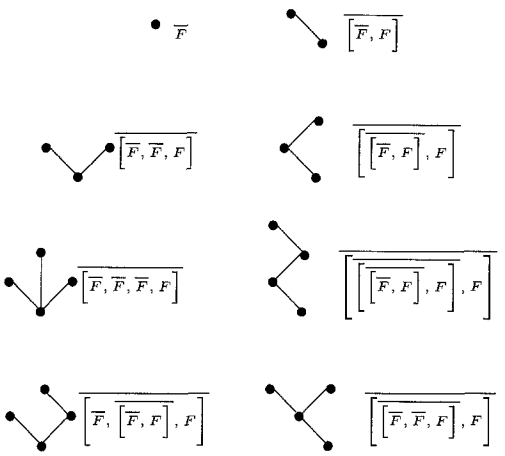

Figure 3.1: The correspondence between Butcher trees and commutators

3.3 Lie–Butcher series for the numerical solution

In this section we will derive the order conditions by computing the Lie–Butcher series for the Y_h computed by the RK method. To keep the exposition on an elementary level, we will be more explicit than necessary in our procedure. A mathematically more streamlined version of this process is given in the next section. The RK process computes a flow, generated by Y_h, which is close to the true solution in the two points $t = 0$ and $t = 1$ but not necessarily in between. For the analysis we must compute the flow which follows the solution for all times.

LEMMA 3.4. *Let* $\mathfrak{g}$ *be abelian and let* $Y_h, \tilde{Z}_h \in \mathfrak{g}$ *such that*

$$\Psi_{t=1,Y_h}(e) = \Psi_{h,\tilde{Z}_h}(e) \text{ for all } h.$$

Then

$$\tilde{Z}_h = \dot{Y}_h = \frac{d}{dh} Y_h.$$

PROOF. In the euclidean example this is an elementary result of vector calculus. We will compute $\tilde{Z}_h$ for *general* $\mathfrak{g}$. Define the *exponential map* $\exp : \mathfrak{g} \to \mathcal{M}$ as $\exp(Y) = \Psi_{t=1,Y}(e)$. Regard Y_h as a curve in $\mathfrak{g}$, which is mapped to a curve $\exp(Y_h)$ in $\mathcal{M}$. $\tilde{Z}_h$ is a tangent vector to this curve in the point $q = \exp(Y_h)$. Hence:

$$\tilde{Z}_h(q) = \mathbf{d}(\exp(Y_h))(h) = (\mathbf{d}\exp)_{Y_h}(\dot{Y}_h)(q),$$

where $\mathbf{d}\exp$ is the differential of the exponential map [15](p.108). If two infinitesimal generators are equal in a point they are equal everywhere, so in general

$$\tilde{Z}_h = (\mathbf{d}\exp)_{Y_h}(\dot{Y}_h).$$

In the abelian case $\mathbf{d}\exp$ is the identity map. □

THEOREM 3.5. *The Runge–Kutta algorithm compute $Y_h \in \mathfrak{g}$ such that*

$$\Psi_{t=1,Y_h}(e) = \Psi_{h,\tilde{Z}_h}(e),$$

where

$$\tilde{Z}_h = \sum_{i=0}^{\infty} \frac{h^i}{i!} \tilde{Z}_0^{(i)}.$$

If $\mathfrak{g}$ is abelian, then the terms $\tilde{Z}_0^{(i)}$ are given by the following recursion:

$$K_i^{(0)} = \overline{F} \tag{3.14}$$

$$K_i^{(n)} = \overline{\left[B_n(\dot{Y}_i), F\right]}\ ,\ n \geq 1 \tag{3.15}$$

$$\dot{Y}_i^{(n)} = (n+1)\sum_{j=1}^{s} a_{i,j} K_j^{(n)}\ ,\ n \geq 0 \tag{3.16}$$

$$\tilde{Z}_0^{(n)} = (n+1)\sum_{j=1}^{s} b_j K_j^{(n)}\ ,\ n \geq 0 \tag{3.17}$$

PROOF. Eqns. (3.14),(3.15) follows from Lemma 3.2. Eqns. (3.16),(3.17) follows from Lemma 3.4 by applying the Leibniz formula:

$$(h\phi(h))^{(q)}\Big|_{h=0} = q\,(\phi(h))^{(q-1)}\Big|_{h=0}$$

to (3.2), (3.3). □

Let us write out the first terms of this recursion:

$$\begin{aligned}
K_i^{(0)} &= \overline{F} \\
\dot{Y}_i^{(0)} &= \sum_j a_{i,j} K_j^{(0)} = \sum_j a_{i,j}\overline{F} \\
K_i^{(1)} &= \overline{\left[\dot{Y}_i^{(0)}, F\right]} = \sum_j a_{i,j}\overline{[\overline{F}, F]} \\
\dot{Y}_i^{(1)} &= 2\sum_j K_j^{(1)} = 2\sum_{j,k} a_{i,j}a_{j,k}\overline{[\overline{F}, F]} \\
K_i^{(2)} &= \overline{\left[\dot{Y}_i^2 + \dot{Y}_i^{(1)}, F\right]} = \sum_{j,k} a_{i,j}a_{i,k}\left[\overline{F}, \overline{F}, F\right] + 2\sum_{j,k} a_{i,j}a_{i,k}\overline{\left[\overline{[\overline{F}, F]}, F\right]}.
\end{aligned}$$

From these we obtain $\tilde{Z}_0^{(i)}$ and the order conditions:

$$\tilde{Z}_0^{(0)} = \sum_j b_j \overline{F} \Rightarrow \sum_j b_j = 1$$

$$\begin{aligned}
\tilde{Z}_0^{(1)} &= 2\sum_{i,j} b_i a_{i,j} \overline{[F,F]} \Rightarrow 2\sum_{i,j} b_i a_{i,j} = 1 \\
\tilde{Z}_0^{(2)} &= 3\sum_{i,j,k} b_i a_{i,j} a_{i,k} \overline{[F,\overline{F,F}]} + 6\sum_{i,j,k} b_i a_{i,j} a_{i,k} \overline{\left[\overline{[F,F]},F\right]} \\
&\Rightarrow 3\sum_{i,j,k} b_i a_{i,j} a_{i,k} = 1\ , \quad 6\sum_{i,j,k} b_i a_{i,j} a_{j,k} = 1.
\end{aligned}$$

3.4 Streamlining the theory

We will in this section introduce a more compact notation and simplify the recursion. Formally the theory could have been developed by defining the s–stage RK scheme on $\mathcal{M}$ as a projection of a 1–stage scheme on the s–fold product manifold $\mathcal{M}\times\cdots\times\mathcal{M}$. We will, however, rather do it directly by inspecting the formulas of the previous section.

The different internal stages K_i and Y_i are stacked together in $s\times n$ matrices $K, Y \in \mathfrak{g}_s$, where $\mathfrak{g}_s = \mathbb{R}^s \otimes \mathfrak{g}$. Commutators and pullbacks are extended to $\mathfrak{g}_s$ by parallelizing over the first component of $\mathfrak{g}_s$, if $Y = \sum_i u_i \otimes Y_i \in \mathfrak{g}_s$ then:

$$\overline{[Y,F]} = \sum_i u_i \otimes \overline{[Y_i,F]} \tag{3.18}$$

$$\overline{\Psi^*_{t,Y} F} = \sum_i u_i \otimes \overline{\Psi^*_{t,Y_i} F} \tag{3.19}$$

We equip $\mathfrak{g}_s$ with a product $\cdot$ by letting

$$(u_i \otimes Y_i)\cdot(v_j \otimes Z_j) = (u_i \cdot v_j) \otimes (Y_i \cdot Z_j) \tag{3.20}$$

where $u_i \cdot v_j \in \mathbb{R}^s$ is the Schur product i.e. the componentwise product of the two vectors, and $Y_i \cdot Z_j$ is the product in the sense of (3.5). The product is extended to all of $\mathfrak{g}_s$ by linearity, i.e.:

$$\left(\sum_i u_i \otimes Y_i\right) \cdot \left(\sum_j v_j \otimes Z_j\right) = \sum_{i,j} (u_i \cdot v_j) \otimes (Y_i \cdot Z_j)\,. \tag{3.21}$$

The operations on $\mathfrak{g}_s$ are now defined such that Lemma 3.2 holds in the same form even for $Z_t \in \mathfrak{g}_s$.

Let $A = [a_{i,j}] : \mathbb{R}^s \to \mathbb{R}^s$ and let $b^T = (b_1, b_2, \ldots, b_s) : \mathbb{R}^s \to \mathbb{R}$. The RK scheme can now be written:

$$K = \overline{\Psi^*_{t=1,Y} F} \tag{3.22}$$

$$Y = (hA \otimes I)\, K \tag{3.23}$$

$$Y_h = \left(hb^T \otimes I\right) K \tag{3.24}$$

$$y_i = \Psi_{t=1,Y_h}(e) \tag{3.25}$$

where $I : \mathfrak{g} \to \mathfrak{g}$ is the identity matrix, $Y, K \in \mathfrak{g}_s$ and $Y_h \in \mathfrak{g}$.

It is now a straightforward matter to verify that our recursion can be written as:

THEOREM 3.6. *The terms $\tilde{Z}_0^{(n)} \in \mathfrak{g}$ of Theorem 3.5 are given by the following recursion:*

$$K^{(0)} = 1 \otimes \overline{F} \tag{3.26}$$

$$\dot{Y}^{(n)} = ((n+1)A \otimes I)\, K^{(n)}\ , n \geq 0 \tag{3.27}$$

$$K^{(n)} = \overline{\left[B_n(\dot{Y}_i), F\right]}\ , n \geq 1 \tag{3.28}$$

$$\tilde{Z}_0^{(n)} = \left((n+1)b^T \otimes I\right) K^{(n)}\ , n \geq 0 \tag{3.29}$$

where $\mathbf{1} = (1, \ldots, 1)^T \in \mathbb{R}^s$ *and* $\dot{Y}^{(n)}, K^{(n)} \in \mathfrak{g}_s$.

Let A^n denote the n–fold matrix product, let $c = A\mathbf{1}$ and let $c^n = c \cdot c \cdots c$ denote the n–fold Schur product. We write out the recursion above, and let the coefficients of $B_n(\dot{Y})$ enter the recursion on the *right hand side* of each tensor product. Then the right hand parts of the terms equal the terms of $Z_0^{(n)}$ in Theorem 3.3, and the order conditions can be read directly out of the recursion:

$$\dot{Y}^{(0)} = (A \otimes I)K^{(0)} = (A \otimes I)\,(\mathbf{1} \otimes \overline{F}) \quad \Rightarrow \underline{b^T \mathbf{1} = 1}$$

$$\dot{Y}^{(1)} = (2A \otimes I)K^{(1)} = (2A \otimes I)\left(c \otimes \overline{[F, F]}\right) \quad \Rightarrow \underline{2b^T c = 1}$$

$$\dot{Y}^{(2)} = (3A \otimes I)\left(c^2 \otimes \overline{[F, F, F]} + 2Ac \otimes \overline{\left[[F, F], F\right]}\right) \quad \Rightarrow \underline{3b^T c^2 = 1,\ 6b^T Ac = 1}$$

$$\begin{aligned}\dot{Y}^{(3)} = {} & (4A \otimes I)\left(c^3 \otimes \overline{[F, F, F, F]} + c \cdot 2Ac \otimes 3\overline{\left[F, \overline{[F, F]}, F\right]} + \right. \\ & \left. 3Ac^2 \otimes \overline{\left[[F, F, F], F\right]} + 6A^2 c \otimes \overline{\left[\left[[F, F], F\right], F\right]}\right) \\ \Rightarrow {} & \underline{4b^T c^3 = 1,\ 8b^T (c \cdot Ac) = 1,\ 12b^T Ac^2 = 1,\ 24b^T A^2 c = 1}\end{aligned}$$

Note that to derive the order conditions, we only need the left part of each tensor product. It is neither necessary to know the coefficients of each term in $B_n(\dot{Y})$ nor the exact form of each commutator to do this, i.e. the recursion may be simply developed in compact form as:

(3.30)

n	Terms in $B_n(\dot{Y})$	$\dot{Y}^{(n)} = (n+1)A\left(K^{(n)}\right)$
0		$A(\mathbf{1}) = c$
1	$\left(\dot{Y}\right)$	$2A\left(B_1(\dot{Y})\right) = 2A\,(c)$
2	$\left(\dot{Y}^2, \dot{Y}^{(1)}\right)$	$3A\left(B_2(\dot{Y})\right) = 3A\left(c^2, 2Ac\right)$
3	$\left(\dot{Y}^3, \dot{Y} \cdot \dot{Y}^{(1)}, \dot{Y}^{(2)}\right)$	$4A\left(B_3(\dot{Y})\right) = 4A\left(c^3, c \cdot 2Ac, 3Ac^2, 6A^2 c\right)$

From the right column of (3.30) we get the order conditions:

$$
\begin{array}{ll}
& b^T\mathbf{1} = 1 \\
& 2b^Tc = 1 \\
(3.31) \quad & 3b^Tc^2 = 1\ , \quad 6b^TAc = 1 \\
& 4b^Tc^3 = 1\ , \quad 8b^T(c\cdot Ac) = 1\ , \quad 12b^TAc^2 = 1\ , \quad 24b^TA^2c = 1 \\
& \vdots
\end{array}
$$

Thus Theorem 3.6 as presented in (3.30, 3.31) provides a simple recursion for generating order conditions of arbitrary order.

4 Concluding remarks.

There are several applications and generalizations of the present theory that we want to be pursue in future work:

- **Non–abelian case:** From the theory of this paper it is possible to write down order conditions for the general Lie group case. This leads to several new order conditions, where the ordering of the subtrees in a given tree is of importance. We have not yet studied these conditions and the development of corresponding RK methods in detail.

- **Composition of methods:** There is a literature on the composition of RK methods [5, 8]. It seems feasible to study compositions of methods within the present framework by employing the Baker–Champbell–Hausdorff formula [15](p.114), but the details of how this is done is still open.

- **RK on homogeneous manifolds:** We believe that the present approach may be formulated on homogeneous manifolds rather than Lie groups. These are quotients of two Lie groups, and include several manifolds of major practical interest, such as (the surface of) n–spheres, projective spaces, Grassmann and Stiefel manifolds.

Acknowledgement

I would like to thank Brynjulf Owren for very valuable discussions and inputs.

REFERENCES

1. R. Abraham, J. E. Marsden and T. Ratiu, *Manifolds, Tensor Analysis, and Applications*, Springer-Verlag, 1980.
2. P. Albrecht, *The extension of the theory of A-methods to RK-methods*, in Numerical Treatment of Differential Equations, in Proceedings 4th Seminar NUMDIFF-4, ed. K. Strehmel, Teubner-Texte zur Mathematik, Leipzig, (1987), pp. 8–18.
3. R. L. Bryant, *An Introduction to Lie Groups and Symplectic Geometry*, in Geometry and Quantum field theory, Eds. Freed and Uhlenbeck, AMS IAS/Parc City math. series vol. 1, (1995), pp.1–182.

4. J. C. Butcher, *Coefficients for the study of Runge-Kutta integration processes*, J. Austral. Math. Soc., 3 (1963), pp. 185–201.
5. J. C. Butcher, *An algebraic theory of integration methods*, Math. Comput. 26 (1972), pp. 79–106.
6. P. E. Crouch and R. Grossman, *Numerical integration of ordinary differential equations on manifolds*, J. Nonlinear Sci.,3 (1993), pp. 1–33.
7. E. Hairer, S. P. Nørsett and G. Wanner, *Solving Ordinary Differential Equations I, Nonstiff Problems*, Springer-Verlag, Berlin, 1987.
8. E. Hairer and G. Wanner, *On the Butcher group and general multi–value methods*, Computing, 13 (1974), pp. 1–15.
9. S. Helgason, *Differential Geometry, Lie Groups and Symmetric Spaces*, Academic Press, New York, 1978.
10. M. Haveraaen, V. Madsen and H. Munthe–Kaas, *Algebraic programming technology for partial differential equations*, in Proceedings of Norsk Informatikk Konferanse (NIK), Ed. A. Maus, Tapir Trondheim Norway, (1992), pp. 55–68.
11. H. Munthe–Kaas and M. Haveraaen, *Coordinate free numerics I; How to avoid index wrestling in tensor computations*, Report no. 101, Department of Informatics, University of Bergen, Norway (1994).
12. P. J. Olver, *Applications of Lie Groups to Differential Equations*, Springer-Verlag GTM no. 107, New York, 1986.
13. J. M. Sanz–Serna and M. P. Calvo, *Numerical Hamiltonian Problems*, Chapman & Hall, London, 1994.
14. J. C. Simo and K. K. Wong, *Unconditionally stable algorithms for rigid body dynamics that exactly preserve energy and momentum*, Internat. J. Numer. Methods in Engrg., 31 (1989), pp. 19–52.
15. V. S. Varadarajan, *Lie Groups, Lie Algebras and Their Representations*, Springer-Verlag GTM no. 102, New York, 1984.
16. F. M. Warner, *Foundations of Differential Manifolds and Lie Groups*, Springer-Verlag GTM no. 94, 1983.