A Survey of the Explicit Runge-Kutta Method (Revised April, 1995)

Wayne H. Enright * Desmond J. Higham † Brynjulf Owren ‡ Philip W. Sharp §

Abstract

Research in explicit Runge-Kutta methods is producing continual improvements to the original algorithms, and the aim of this survey is to relate the current state-of-the-art. In drawing attention to recent advances, we hope to provide useful information for those who apply numerical methods.

We describe recent work in the derivation of Runge-Kutta coefficients: "classical" general-purpose formulas, "special" formulas for high order and Hamiltonian problems, and "continuous" formulas for dense output. We also give a thorough review of implementation details. Modern techniques are described for the tasks of controlling the local error in a step-by-step integration, computing reliable estimates of the global error, detecting stiffness, and detecting and handling discontinuities and singularities. We also discuss some important software issues.

1 Introduction

Explicit Runge-Kutta (ERK) formulas are among the oldest and best-understood schemes in the numerical analyst's toolkit. However, despite the evolution of a vast and comprehensive body of knowledge, ERK algorithms continue to be a source of active research [8]. The purpose of this survey paper is to acquaint the reader with recent developments in the field and to point out areas that are currently under investigation. Emphasis will be on practical aspects that are likely to have a direct influence on upcoming software. We deal exclusively with the design of *explicit* Runge-Kutta algorithms for solving *nonstiff* initial value problems in ordinary differential equations.

The history of ERK methods began almost a century ago. Classic references are [37, 51, 65]. For a thorough coverage of the conception and subsequent theoretical and practical development of the Runge-Kutta process the reader is referred to the texts [6, 34]. Because of their elegance and simplicity ERK methods are usually among the first to be taught in the ODE section of a numerical methods course. Thankfully, good quality introductory texts no longer dismiss "The Runge-Kutta Method" as a fixed stepsize implementation of the classic 4th order ERK formula. However, significant advancements in the state-of-the-art which post-date the work of Fehlberg [26], even in the fundamental area of deriving ERK formulas, tend to be ignored. Another side effect of the simple nature of the ERK formula is that a generation of non-experts have been tempted to write their own "quick and dirty" codes [50, page 328]. It is widely acknowledged that "squeaky-clean" codes require a great deal of expertise and programming effort. A high-level discussion of some of the issues involved in ERK code design is given in [81].

*Department of Computer Science, University of Toronto, Toronto, M5S 1A4, Canada The work of this author was supported by the Information Technology Research Centre of Ontario and the Natural Science and Engineering Council of Canada. (enright@cs.utoronto.ca).

†Department of Mathematics and Computer Science, University of Dundee, Dundee, DD1 4HN, Scotland. The work of this author was supported by the UK Science and Engineering Research Council and by a joint research grant from the Research Council of Norway and the British Council. (na.dhigham@na-net.ornl.gov).

‡Department of Mathematical Sciences, The University of Trondheim, N-7034 Trondheim-NTH, Norway. The work of this author was supported by a joint research grant from the Research Council of Norway and the British Council. (bryn@imf.unit.no).

§Department of Mathematics, University of Auckland, Private Bag 92019, Auckland, New Zealand. The work of this author was supported by the Natural Science and Engineering Council of Canada and the University of Auckland. (sharp@mat.aukland.ac.nz).

Perhaps the main appeal of ERK formulas lies in their self-contained, one-step nature. By retaining the minimum amount of information between steps, an ERK algorithm can adapt naturally to changes in the behavior of the solution. Furthermore, a rigorous analysis of the local error can be performed without reference to previously computed information. This in turn allows compact and (asymptotically) reliable error control algorithms to be developed with relatively few heuristics involved. Adaptive ERK codes were among the first to appear in numerical software libraries (see, for example, [29]) and they have undergone substantial refinement over the last twenty or so years. They continue to form an integral part of modern ODE packages, such as those in [7, 30, 48]. Despite the fact that good, reliable codes exist, there is still a significant amount of room for improvement. The underlying formulas found in existing codes are now known to be non-optimal (see section 2) and hence the basic efficiency of the codes can be increased. More importantly, techniques for improving the robustness, reliability, functionality and convenience of ERK codes have recently been developed. In the forthcoming sections, we emphasize those features which we believe will enhance general-purpose ERK software. The remainder of this section introduces the basic concepts, definitions and notation used throughout the paper.

Our aim is to compute an approximate solution of the system of ODEs

$$y'(x) = f(x, y(x)), \quad y(a) = y_a, \quad f : \mathbb{R} \times \mathbb{R}^N \to \mathbb{R}^N. \tag{1.1}$$

Here we are given the initial vector y_a and must follow the solution curves as x increases. We may want an approximation at a single point $b > a$, at a number of such points, or over a continuous interval $[a, b]$. The basic approach shared by all popular solution methods is to compute a sequence of discrete approximation vectors $y_i \approx y(x_i)$ in a step-wise fashion, starting with $x_0 = a$ and $y_0 = y_a$. On a general step, with y_{i-1} available, we choose a stepsize h_i and compute the next approximation at $x_i := x_{i-1} + h_i$. The ERK formula does this in the following way

$$\begin{aligned}
k_1 &= f(x_{i-1}, y_{i-1}), \\
k_r &= f(x_{i-1} + c_r h_i, y_{i-1} + h_i \sum_{j=1}^{r-1} a_{rj} k_j), \quad 2 \le r \le s, \\
y_i &= y_{i-1} + h_i \sum_{r=1}^{s} b_r k_r.
\end{aligned} \tag{1.2}$$

The formula is defined by the number of stages s, the nodes $\{c_r\}_{r=1}^s$, the internal weights $\{a_{rj}\}_{j=1,r=2}^{r-1,s}$, and the external weights $\{b_r\}_{r=1}^s$. These coefficients are normally displayed in a Butcher tableau of the form

$$\begin{array}{c|ccccc}
c_1 & & & & & \\
c_2 & a_{21} & & & & \\
\vdots & \vdots & \ddots & & & \\
\vdots & \vdots & & \ddots & & \\
c_s & a_{s1} & a_{s2} & \dots & a_{s,s-1} & \\
\hline
 & b_1 & b_2 & \dots & \dots & b_s
\end{array}$$

The process for computing y_i in (1.2) is explicit, involving s evaluations of the function f. The number of f evaluations required gives a reasonable measure of the cost of the step—if the function f is expensive then the f evaluations will dominate, if not then the overall step will be cheap anyway.

It is helpful to define the local solution $z_i(x)$ for the current step as the solution curve which matches y_{i-1} at x_{i-1}; that is

$$z_i'(x) = f(x, z_i(x)), \quad z_i(x_{i-1}) = y_{i-1}. \tag{1.3}$$

Since we begin with $y_{i-1} \approx y(x_{i-1})$, we may interpret our efforts on the current step as an attempt to follow the local solution. We define the local error for the step to be

$$le_i := y_i - z_i(x_i), \tag{1.4}$$

and the global error to be

$$ge_i := y_i - y(x_i). \tag{1.5}$$

We will always assume that the formula has order p; this means that p is the largest integer such that the local error satisfies $le_i = O(h_i^{p+1})$ as $h_i \to 0$. (Unless otherwise indicated, we assume that the function f is sufficiently differentiable.) The task of selecting the ERK coefficients to achieve a certain order will be considered in the next section. It can be shown that the global error in a pth order formula behaves like $O(h_{\max}^p)$ where $h_{\max}$ is the maximum stepsize used.

In order to control the errors in the numerical solution, a pair of formulas of different orders is normally used on each step. To save f evaluations, the two formulas can be "embedded" so that they share k_r values. The second formula will thus have the form

$$\widehat{y}_i = y_{i-1} + h_i \sum_{r=1}^{s} \widehat{b}_r k_r. \tag{1.6}$$

The difference between the two approximations y_i and $\widehat{y}_i$ gives an asymptotically correct estimate of the local error in the lower order approximation. We may then decide to accept this step if a suitably weighted norm of the local error estimate is kept within a pre-defined tolerance level. A pair of formulas of orders p and q is usually referred to as a (p, q) pair.

For the majority of embedded ERK formula pairs which have been derived, the formulas (1.2) and (1.6) differ in order by one. For reasons of efficiency, it is usually recommended that the pair be used with y_i as the higher order formula. In this way, an estimate of the local error in the lower order approximation is controlled, but the more accurate approximation is used to advance the integration. This mode of operation is called local extrapolation. The issues involved in reliably and efficiently controlling the error, via the stepsize, are discussed in section 3.

As we mentioned earlier in this section, there are times when a continuous approximation to the solution of (1.1) is required. A natural way to do this, which ties in with the one-step nature of the ERK formula, is to construct a local approximation $\widehat{z}_i(x) \approx y(x)$ on each step from x_{i-1} to x_i. By joining these local functions in a piecewise fashion, a global approximation is obtained. Several such classes of interpolants, or "continuous extensions", to the basic ERK formula have been developed and analyzed in recent years. They usually fit into the general form (with $x := x_{i-1} + \tau h_i$)

$$\widehat{z}_i(x_{i-1} + \tau h_i) = y_{i-1} + \tau h_i \sum_{r=1}^{\widehat{s}} \tilde{b}_r(\tau) k_r, \tag{1.7}$$

where $\{\tilde{b}_r(\tau)\}_{r=1}^{\widehat{s}}$ are polynomials in τ and $\widehat{s} \geq s$. If $\widehat{s} > s$, which is typically the case, then extra stages (and hence extra f evaluations) must be added to the basic step in order to form the continuous extension. Notice that $\widehat{z}_i(x_{i-1} + \tau h_i)$ can be regarded as the result of a step of length τh_i with an ERK formula whose coefficients are $\{c_r/\tau\}_{r=1}^{\widehat{s}}$, $\{a_{rj}/\tau\}_{j=1,r=1}^{r-1,\widehat{s}}$, and $\{\tilde{b}_r(\tau)\}_{r=1}^{\widehat{s}}$. The conditions $\tilde{b}_r(1) = b_r$ for $r = 1, \ldots, s$ and $\tilde{b}_r(1) = 0$ for $r = s+1, \ldots, \widehat{s}$ ensure that (1.7) reduces to the basic formula (1.2) at $\tau = 1$, and they also make the global approximation continuous. As an illustration, it is easily verified that the cubic Hermite polynomial interpolant to solution values $\{y_{i-1}, y_i\}$ and derivative values $\{f(x_{i-1}, y_{i-1}), f(x_i, y_i)\}$ has the form (1.7). In general, however, higher degree polynomials must be used in order to obtain sufficient accuracy in the approximation. A pair of formulas together with a continuous extension can be represented by the generalized tableau

$$
\begin{array}{c|cccccccc}
c_1 & & & & & & & & \\
c_2 & a_{21} & & & & & & & \\
\vdots & \vdots & \ddots & & & & & & \\
\vdots & \vdots & & \ddots & & & & & \\
c_s & a_{s1} & a_{s2} & \cdots & a_{s,s-1} & & & & \\
\hline
c_{s+1} & a_{s+1,1} & a_{s+1,2} & \cdots & \cdots & a_{s+1,s} & & & \\
\vdots & \vdots & \vdots & & & & \ddots & & \\
c_{\widehat{s}} & a_{\widehat{s}1} & a_{\widehat{s}2} & \cdots & \cdots & \cdots & \cdots & a_{\widehat{s},\widehat{s}-1} & \\
\hline
 & b_1 & b_2 & \cdots & \cdots & b_s & & & \\
 & \widehat{b}_1 & \widehat{b}_2 & \cdots & \cdots & \widehat{b}_s & & & \\
 & \tilde{b}_1(\tau) & \tilde{b}_2(\tau) & \cdots & \cdots & \cdots & \cdots & \cdots & \tilde{b}_{\widehat{s}}(\tau)
\end{array}
\tag{1.8}
$$

At this stage, it should be clear to the reader that a complete "ERK method" involves much more than a simple formula. Our main aim in this paper is to survey the state-of-the art and to point out some of the recent theoretical and practical developments that will enable existing methods to be improved. The next section deals with the derivation of formula pairs—an area where advances are still being made (partly due to the availability of sophisticated symbolic manipulation packages). Section 3 then discusses various options for controlling the local errors in the integration. The question of estimating global errors, which is of great practical importance, is considered in section 4. In section 5 we discuss techniques that determine when a problem is too stiff to be efficiently solved with an ERK method. Section 6 looks at the need for continuous extensions and summarizes recent results. Section 7 is concerned with the reliable detection and treatment of singularities and low order discontinuities in the solution. Some recent developments in higher order systems, Hamiltonian problems and dynamical systems theory are covered in section 8. Finally, in section 9, we address the issues involved in writing ERK software, and refer to a state-of-the-art code.

2 Derivation and selection of pairs

2.1 Derivation

From section 1, a formula is order p if the Taylor series expansions of y_i and $z_i(x_i)$ about the point (x_{i-1}, y_{i-1}) agree up to the h^p term. This definition implies that a formula can be derived by first finding the Taylor series expansions for y_i and $z_i(x_i)$. The coefficients of the partial derivatives in the expansion for y_i will involve the coefficients of the formula, while the coefficients in the expansion for $z_i(x_i)$ will be rational numbers. Equating terms in the two expansions gives a system of equations for the coefficients of the formula. These equations, which are called the order conditions, may then be solved. While techniques for generating the order conditions have been available for three decades, new solutions to the order conditions are still being found. We illustrate how the order conditions can be solved by discussing the derivation of one parametric family of seven-stage sixth order formulas and the related eight-stage (5,6) pairs.

We begin with the seven-stage sixth order formula. We assume that $c_1 = 0$, and as is invariably done except for very low order formulas, that

$$
c_i = \sum_{j=1}^{7} a_{ij}, \quad i = 2, \ldots, 7. \tag{2.1}
$$

With these assumptions there are thirty-seven order conditions to satisfy, which, along with (2.1), form a system of forty-three equations (mostly non-linear) in the thirty-four unknowns $\{c_i\}_{i=2}^{7}$, $\{b_i\}_{i=1}^{7}$ and $\{a_{ij}\}_{j=1,i=2}^{i-1,7}$.

Since there are more equations than unknowns it is tempting to conclude that the equations have no solution. However, this reasoning ignores the fact that many equations have a similar structure. This structure can be exploited to yield a smaller number of equations whose solutions are also solutions of the complete set of order conditions. For example, if in addition to (2.1), the coefficients satisfy

$$\sum_{i=j+1}^{7} b_i a_{ij} = b_j(1-c_j), \quad j=2,\ldots,7, \tag{2.2}$$

then sixteen of the thirty-seven order conditions beginning with $\sum_{i=j+1}^{s} b_i a_{ij}$ reduce to multiples of the remaining twenty-one. (Note that if the first two order conditions are satisfied then (2.2) is also valid for $j=1$.) We then have thirty-three equations to solve (six from each of (2.1) and (2.2) and the twenty-one order conditions), ten less than the original system. In addition, a larger proportion of the equations are linear in the $\{a_{ij}\}$. This has an advantage since feasible values of the nodes and external weights can be selected independently of these internal weights.

If we further assume that

$$\frac{c_i^2}{2} = \sum_{j=1}^{i-1} a_{ij}c_j, \quad i=3,\ldots,7, \tag{2.3}$$

and $b_2 = 0$, then (2.2) implies that $\sum_i b_i a_{i2} = 0$ also, and then seven of the twenty-one order conditions reduce to multiples of the remaining fourteen. We then have thirty-two equations to solve (six each from (2.1), (2.2) and (2.3) and fourteen order conditions), in the original thirty-four unknowns. (In fact, (2.2) and (2.3) imply that two of the fourteen order conditions are satisfied if and only if $\sum_i b_i c_i a_{i2} = 0$ so that there are only thirty-one equations to solve.) Since there are more unknowns than equations we may expect to solve this reduced set of equations. The use of such "simplifying assumptions" is a direct way of finding some general parametric solutions of the order equations, and complete solutions of this type have been developed in [5, 17]. In an alternative approach using a different reduced subset of the order equations, Verner [96, 97] uses homogeneous polynomials of the external and internal weights as an intermediate stage for finding parametric solutions of the order equations. Although indirect, this approach has been generalized successfully to find families of high-orders of several types of pairs and interpolants [99] and [86]. Since this careful approach illustrates that classes of solutions of the associated sets of nonlinear order equations can be characterized naturally by exploiting the underlying structure, the illustrative example for pairs of order six developed by Verner in [98] is included here with the author's permission. (The presentation in [99] extends this example by emphasizing the connection with integration, and leads to another type of method.)

The derivation is based on p-node interpolatory quadrature rules, and so initially it is assumed that only six stages are required. When the Runge–Kutta method is applied to the test problem

$$y' = g(x)$$

we obtain the quadrature conditions

$$\sum_{i=1}^{6} b_i c_i^k = \frac{1}{k+1}, \quad k=0,\ldots,5.$$

If the nodes c_i, $i=1,\ldots,6$, are known and distinct, these order conditions can be solved for $b_i, i=1,\ldots,6$, as in the direct approach.

We now apply the same method to the test problem

$$y'(x) = My(x) + g(x)$$

where M is an $n \times n$ matrix. One set of order conditions is

$$\sum_{j=1}^{5} \left[\sum_{i=j+1}^{6} b_i a_{ij} \right] c_j^k = \frac{1}{(k+1)(k+2)}, \quad k=0,\ldots,4.$$

The summation $\sum_{i=2}^{6} b_i a_{i1}$ ($= b_1$ by the implication of (2.2)) is usually non-zero even though c_1 is always zero, and appears explicitly only for $k=0$.

The above order conditions form a system of five linear equations for the five summations $\sum_{i=j+1}^{6} b_i a_{ij}$, $j = 1, \ldots, 5$. If c_i, $i = 1, \ldots, 5$ are distinct the coefficient matrix for the system is non-singular and there is a unique solution. Each of these summations is a homogeneous polynomial of degree 2 in the external and internal weights.

Another set of order conditions is

$$\sum_{l=1}^{4} \left[\sum_{i=j+1}^{6} \sum_{j=l+1}^{i-1} b_i a_{ij} a_{jl} \right] c_l^k = \frac{1}{(k+1)(k+2)(k+3)}, \quad k = 0, \ldots, 3,$$

which is a system of four linear equations in the homogeneous polynomials $\sum_{i=j+1}^{6} \sum_{j=l+1}^{i-1} b_i a_{ij} a_{jl}$, $l = 1, \ldots, 4$ of degree 3. The remaining order conditions for the second test equation form systems of linear equations for the homogeneous polynomials $\sum_{i,j,l} b_i a_{ij} a_{jl} a_{lm}$, $m = 1, 2, 3$, $\sum_{i,j,l,m} b_i a_{ij} a_{jl} a_{lm} a_{mn}$, $n = 1, 2$ and $b_6 a_{65} a_{54} a_{43} a_{32} a_{21}$ of increasing degrees.

Once the homogeneous polynomials are known we can arrange them formally in the tableau

$$\begin{array}{c|cccccc}
c_1 & & & & & & \\
c_2 & b_6 a_{65} a_{54} a_{43} a_{32} a_{21} & & & & & \\
c_3 & . & . & & & & \\
c_4 & . & . & . & & & \\
c_5 & \sum b_i a_{ij} a_{j1} & . & . & b_6 a_{65} a_{54} & & \\
c_6 & \sum b_i a_{i1} & . & . & \sum b_i a_{i4} & b_6 a_{65} & \\
\hline
 & b_1 & . & . & . & . & b_6
\end{array} \tag{2.4}$$

The $\{a_{ij}\}$ can be found from (2.4) by using a back substitution scheme (provided that $b_6 a_{65} a_{54} a_{43} a_{32} a_{21} \neq 0$). From the (5,5) element in the tableau we get a_{65} (b_6 is known from the quadrature conditions). Then from the (4,4) element we get a_{54}. Continuing up the main diagonal we get a_{43}, a_{32} and a_{21} from the (3,3), (2,2) and (1,1) elements respectively. Next, we move up the sub-diagonal, solving for a_{64}, a_{53}, a_{42}, a_{31}. We continue the back substitution until all the $\{a_{ij}\}$'s are found.

Because (2.1) holds, only ten of the above order conditions involving the homogeneous polynomials are linearly independent. This, along with the quadrature conditions, means sixteen order conditions are satisfied. If we set $c_6 = 1$, the simplifying assumptions (2.2) also hold, and these imply that six more order conditions are satisfied. More, but not all, order conditions can be satisfied by constraining the other nodes. Hence, to obtain the formula of order six we must add a stage.

We add the stage after the first stage. The exterior weight b_2 for the stage and the corresponding homogeneous polynomials are set to zero. (Recall in the earlier discussion that conditions $b_2 = \sum_i b_i a_{i2} = \sum_{i,j} b_i a_{ij} a_{j2} = 0$ were needed to complement (2.3) to reduce the set of order conditions.) The tableau of homogeneous polynomials becomes

$$\begin{array}{c|ccccccc}
c_1 & & & & & & & \\
c_2 & ? & & & & & & \\
c_3 & b_6 a_{65} a_{54} a_{43} a_{32} a_{21} & 0 & & & & & \\
c_4 & . & 0 & & & & & \\
c_5 & . & 0 & . & . & & & \\
c_6 & \sum b_i a_{ij} a_{j1} & 0 & . & . & b_7 a_{76} a_{65} & & \\
c_7 & \sum b_i a_{i1} & 0 & . & . & \sum b_i a_{i5} & b_7 a_{76} & \\
\hline
 & b_1 & 0 & . & . & . & . & b_7
\end{array}$$

This tableau differs from (2.4) in two important ways. There is a zero on the key diagonal and the first non-trivial row is incomplete. These differences cause the back substitution scheme to fail.

The failure can be avoided by using the back substitution scheme for just the fifth, sixth and seventh rows. For the third and fourth rows the simplifying assumptions

$$\frac{c_i^{k+1}}{k+1} = \sum_{j=1}^{i-1} a_{ij} c_j^k, \quad k = 0, 1, \quad i = 3, 4, \tag{2.5}$$

are solved for a_{31}, a_{32}, a_{41} and one of a_{42} and a_{43} (without loss of generality we take a_{43}).

Modifying the back substitution scheme in this way means twenty-five of the thirty-seven order conditions are satisfied. With the choice $c_7 = 1$ (c_6 is now a free parameter) six more order conditions are satisfied. These choices also reduce the remaining six conditions to only two. These remaining two are both valid together if both a_{42} satisfies a linear equation, and c_4 is determined by a rational expression involving only c_3.

To summarize:

- Set $c_1 = 0$, $c_7 = 1$. Then select values for c_2, c_3, c_5, c_6 such that c_1, c_3, c_4 (constrained in terms of c_3), c_5, c_6 and c_7 are distinct.
- Solve the quadrature conditions.
- Solve (2.5) and a linear equation in a_{42} for a_{31}, a_{32}, a_{41}, a_{42} and a_{43}.
- Calculate the homogeneous polynomials for the last three rows.
- Perform the back substitution for the last three rows.

Once the sixth order formula is obtained, a fifth order formula is found by first adding a stage with $c_8 = 1$, $\hat{b}_i = 0$, $i = 2, 6, 7$. The homogeneous polynomials for this stage are then calculated and one row of back substitution applied to get a_{8j}, $j = 1, \ldots, 6$ ($a_{87} = 0$).

Verner's approach forming homogeneous polynomials as an intermediate stage for solving order conditions can be generalized to higher orders. For example, to get an s-stage, pth order formula using homogeneous polynomials, we start with $b_i = 0$, $i = 2, \ldots, s-p+1$, $c_1 = 0$, $c_s = 1$ and c_{s-3} linearly constrained in terms of c_i, $i = s-p+2, \ldots, s-4$. Rows three through $s-3$ of the Butcher tableau are then found from a linear equation for the a_{ij} in the $(s-3)$rd row and simplifying assumptions of the form (2.5) (with larger values of i and k). The solution of the simplifying assumptions may place constraints on some of the c_i. Finally, the last three rows of the tableau are found by back substitution on the homogeneous polynomials. The extra stage for the order $p-1$ formula is found by setting $\hat{b}_i = 0$, $i = 2, \ldots s-p+1, s-1, s$ and $c_{s+1} = 1$. Then one row of back substitution on the homogeneous polynomials for the extra stage is performed. More detail on this derivation using homogeneous polynomials of the weights is to be found in [96, 98]. In this respect, the direct approach has also been generalized to find other methods in recent work by other authors [15, 61].

2.2 Selection

A class is usually searched over the free parameters for individual pairs that possess desirable properties. These properties include the efficiency of the pair, the accuracy of the local error estimate and the size of the stability region.

2.2.1 Efficiency

Suppose we are using a $(p-1, p)$ pair with local extrapolation. The local error in the order p formula on the step from x_{i-1} to x_i can be written as

$$h_i^{p+1} \sum_{j=1}^{N_{p+1}} t_{p+1,j} D_{p+1,j} + h_i^{p+2} \sum_{j=1}^{N_{p+2}} t_{p+2,j} D_{p+2,j} + O(h_i^{p+3}), \tag{2.6}$$

where $t_{q,j}$ is the jth truncation coefficient of order q ($q \geq p+1$), $D_{q,j}$ is the corresponding partial derivative (elementary differential) and N_q is the number of truncation coefficients of order q.

Asymptotically, the size of the global error is related to the size of the principal term in the local error in (2.6). This result suggests that if we want to achieve the smallest global error for a given stepsize sequence, the truncation coefficients should be as small as possible. While the requirement is easy to state, it is far from obvious how the size of the truncation coefficients should be measured. Two commonly used measures of the size are

$$T_{p+1} = \left[\sum_{j=1}^{N_{p+1}} t_{p+1,j}^2 \right]^{1/2} \tag{2.7}$$

and

$$T_{p+1} = \max_j \{ |t_{p+1,j}| \}. \tag{2.8}$$

These measures have been successfully used to select efficient pairs (see [17, 63, 79], for example).

If local extrapolation is not being used, the asymptotic result relating the size of the global error and principal truncation coefficients suggests the order p truncation coefficients of the lower order formula should be small. However there is one difficulty with this requirement. Let b_{p-1} and b_p be the vectors of the exterior weights for the $(p-1)$st and pth order formula respectively. Then the formula with exterior weights defined by

$$\alpha b_{p-1} + (1-\alpha) b_p, \quad 0 \neq \alpha \in \mathbb{R},$$

is order $p-1$ and can be used in place of the original order $p-1$ formula. As α tends to zero, the principal truncation coefficients in the new formula tend to zero which suggests a pair can be arbitrarily efficient. One way to overcome this difficulty is to make the truncation coefficients small subject to the local error estimate for the lower order formula being accurate.

2.2.2 Accurate error estimate

The true local error in the lower order formula of a pair is $\hat{y}_i - z_i(x_i)$ while the local error estimate is $\hat{y}_i - y_i$. This means the error in the local error estimate is $y_i - z_i(x_i)$, which is just the local error in the higher order formula. Hence, if we require a small absolute error in the error estimate, the local error in the higher order formula should be small. For pairs that use local extrapolation, this requirement is the same as for efficiency. For pairs that do not use local extrapolation the accuracy of the error estimate will depend on how the pair is made efficient.

Another requirement on the accuracy of the local error estimate is that all the principal truncation coefficients in the lower order formula are non-zero. This helps ensure all elementary differentials are controlled.

2.2.3 Stability

The stability regions of the two formulas in the pair, particularly the formula advancing the solution, should be sufficiently large that the stepsize is not unnecessarily limited by stability requirements. If a mildly stiff problem is being integrated then an enlarged stability region will be of benefit. Such an enlargement can be obtained to a limited extent, but often only by making the truncation coefficients larger, which makes the pair less efficient on non-stiff problems.

The related issue of stiffness is discussed in section 5.

2.2.4 Other properties

There are several other properties a pair could possess. For example, the $\{a_{ij}\}$, $\{b_i\}$ and $\{\hat{b}_i\}$ could be small in magnitude to help control the round-off error, and the truncation coefficients could be tuned to avoid rejected steps.

2.3 Two illustrative pairs

The tableaux in Figures 2.1 and 2.2 give the coefficients of two (4,5) pairs we selected using the general requirements of the previous subsection. The pair in Figure 2.1 is from the class of six-stage pairs derived by Fehlberg [25]. Here, the truncation coefficients of the order 5 formula are about 3.5 times smaller than for the more well-known pair RKF45 [52, page 185], and the error estimate is more reliable. This is roughly the optimal formula within the class. The pair in Figure 2.2 is from the class of seven-stage pairs derived by Dormand and Prince [17]. The pairs in this class have the "first-same-as-last" (FSAL) property—the

$$
\begin{array}{c|cccccc}
0 & & & & & & \\
\frac{1}{5} & \frac{1}{5} & & & & & \\
\frac{3}{10} & \frac{3}{40} & \frac{9}{40} & & & & \\
\frac{3}{5} & \frac{3}{10} & \frac{-9}{10} & \frac{6}{5} & & & \\
\frac{9}{10} & \frac{3}{40} & \frac{27}{40} & \frac{-3}{5} & \frac{3}{4} & & \\
1 & \frac{107}{162} & \frac{5}{2} & \frac{-140}{27} & \frac{35}{9} & \frac{-70}{81} & \\
\hline
\hat{b} & \frac{17}{162} & 0 & \frac{10}{27} & \frac{5}{18} & \frac{20}{81} & 0 \\
b & \frac{8}{81} & 0 & \frac{25}{63} & \frac{25}{108} & \frac{25}{81} & \frac{-1}{28}
\end{array}
$$

Figure 2.1: A pair from the Fehlberg (4,5) class

$$
\begin{array}{c|ccccccc}
0 & & & & & & & \\
\frac{1}{5} & \frac{1}{5} & & & & & & \\
\frac{3}{10} & \frac{3}{40} & \frac{9}{40} & & & & & \\
\frac{7}{10} & \frac{203}{360} & \frac{-49}{24} & \frac{98}{45} & & & & \\
\frac{5}{6} & \frac{-8285}{13608} & \frac{1925}{648} & \frac{-500}{243} & \frac{100}{189} & & & \\
1 & \frac{136}{315} & \frac{-5}{3} & \frac{575}{288} & \frac{-15}{56} & \frac{81}{160} & & \\
1 & \frac{59}{630} & 0 & \frac{125}{288} & \frac{125}{504} & \frac{27}{160} & \frac{1}{18} & \\
\hline
\hat{b} & \frac{59}{630}+\frac{4}{225}\hat{b}_7 & 0 & \frac{-5}{72}\hat{b}_7+\frac{125}{288} & \frac{5}{18}\hat{b}_7+\frac{125}{504} & \frac{-63}{200}\hat{b}_7+\frac{27}{160} & \frac{-41}{45}\hat{b}_7+\frac{1}{18} & \hat{b}_7 \\
b & \frac{59}{630} & 0 & \frac{125}{288} & \frac{125}{504} & \frac{27}{160} & \frac{1}{18} &
\end{array}
$$

Figure 2.2: A pair from the Dormand and Prince (4,5) FSAL class

seventh stage can be re-used as the first stage of the next step. Hence, the effective number of derivative evaluations per step is six, the same as for the Fehlberg class. We have included a free parameter $\hat{b}_7$ to emphasize that formulas are generally members of an infinite family. The choice $\hat{b}_7 = .1$ gives an effective pair.

3 Error Control and Stepsize Selection

Virtually all modern numerical methods vary the stepsize so that the accuracy of the resulting discrete solution, $(x_i, y_i)_{i=0}^{N_{\texttt{TOL}}}$ where $a = x_0 < x_1 \cdots < x_{N_{\texttt{TOL}}} = b$, is consistent with the error tolerance, TOL. The relationship between accuracy and TOL is very sensitive to the problem and can be sensitive to the method as well. At the very least it is desirable that the discrete solution has its maximum global error bounded by some multiple of TOL. The related concept of *tolerance proportionality* has been precisely defined and investigated by Stetter [90, 91]. It is not necessary for our purposes to be as precise. Essentially we assume only that the error control strategy chooses the stepsizes h_i in an attempt to ensure that

$$\max_{i=1,2,\ldots,N_{\texttt{TOL}}} \{\| ge_i \|\} \le \bar{K} \cdot \texttt{TOL}, \tag{3.1}$$

where $\bar{K}$ may depend on both the problem and the method, but is independent of the tolerance. The norm used in (3.1) and throughout this section is the infinity norm. In most cases it would be straightforward to extend the analysis to accommodate any standard norm, and to allow weighting with respect to the solution components.

Note that $\bar{K}$ in (3.1) is not uniquely defined. Generally, the asymptotic reliability and effectiveness of an error control strategy can be quantified by examining the smallest value of K such that

$$\max_{i=1,2,\ldots,N_{\mathtt{TOL}}} \{\| ge_i \|\} \leq K \cdot \mathtt{TOL} + O(h_{\max}\mathtt{TOL}), \tag{3.2}$$

where $h_{\max} := \max_i\{h_i\}$.

3.1 Error control strategies

In order to accomplish (3.2) a method will in general determine some measure, m_i, of the error introduced on the ith step and accept the step only if $m_i \leq \mathtt{TOL}$. If this criterion is not met the attempted step is rejected and the underlying formula is re-applied with a smaller stepsize.

It is clear that for a particular Runge-Kutta formula the reliability of an associated numerical method will be determined to a large extent by the choice of the particular measure m_i. As we shall see later, the error control strategies used in current software packages can be interpreted and justified in terms of the local error introduced on each step.

On the other hand, the efficiency of a method is affected by the strategy used to determine the next 'attempted step'. Clearly it would be ideal to choose on each step the largest stepsize that will lead to an (x_i, y_i) which satisfies $m_i \leq \mathtt{TOL}$ on the first attempt. Too large an attempted stepsize will lead to a rejected step (and consequently extra work) while too small an attempted stepsize will, if the situation persists, result in more steps than would otherwise be necessary.

3.2 Particular Error Control Strategies

We will discuss three particular measures of the error introduced on each step which give rise to effective error control strategies. These strategies can each be viewed as an attempt to ensure that (3.2) will be satisfied. For each alternative we present a theoretical justification for this interpretation.

In each case we introduce an associated continuous function $\hat{z}(x)$ which is defined on $[a, b]$; interpolates $(x_i, y_i)_{i=0}^{N_{\mathtt{TOL}}}$; and which satisfies, at all but a finite number of points in $[a, b]$, the perturbed ordinary differential equation

$$\hat{z}'(x) = f(x, \hat{z}(x)) + \hat{\delta}(x). \tag{3.3}$$

The perturbation term, $\hat{\delta}(x)$, is the *defect* (or residual) associated with $\hat{z}(x)$ and in the cases of interest will satisfy, as $h_{\max} \to 0$,

$$|||\hat{\delta}(x)||| \leq K_1 \cdot \mathtt{TOL} \ + O(h_{\max}\mathtt{TOL}), \tag{3.4}$$

with K_1 independent of $\mathtt{TOL}$. Here $|||\cdot|||$ denotes the maximum value of the norm of a function over the range of x for which it is defined. Once it has been established that $\hat{z}(x)$ satisfies (3.3) and (3.4) then from a standard application of differential inequalities or an application of Gronwall's Lemma (see for example [34, section I.10]) one can show

$$|||y(x) - \hat{z}(x)||| \leq K_2 \cdot \mathtt{TOL} + O(h_{\max}\mathtt{TOL}), \tag{3.5}$$

and therefore

$$\max_{i=1\cdots N_{\mathtt{TOL}}} \| ge_i \| \leq K_2 \cdot \mathtt{TOL} + O(h_{\max}\mathtt{TOL}).$$

With this approach the value of K_2 will depend linearly on K_1 and the analysis relating these two values will be independent of the method, with the constant of proportionality (relating K_2 to K_1) depending only on the mathematical conditioning of the differential equation. For this reason one can quantify and compare the expected accuracy of different error control strategies by considering the size of the respective values of K_1. For the sake of robustness and ease of use it is desirable to have K_1 independent of the method and the differential equation. In such cases it would also be reasonable to expect that the measure m_i is 'normalized' so that $K_1 \approx 1$.

The first error control we will consider is direct control of the local error per unit step (denoted LEPUS). If $z_i(x)$ is the local solution associated with step i and est_i is a reliable and asymptotically correct estimate of the local error in y_i, then the corresponding measure is

$$m_i = \| est_i \| / h_i.$$

An estimate of the local error could be considered 'reliable' if, for all tolerances,

$$\| z_i(x_i) - y_i \| / \| est_i \| \leq 5, \tag{3.6}$$

and is asymptotically correct if

$$est_i = z_i(x_i) - y_i + O(h^{p+1}).$$

In practice, to develop an effective error estimate one usually begins with an asymptotically correct candidate error estimate. An attempt to ensure reliability can be made by imposing a formula-dependent maximum stepsize, **HMAX**, determined to ensure that (3.6) is satisfied for some class of problems. If the resulting restriction on the stepsize is comparable to that imposed by the numerical stability of the underlying formula then it will automatically be satisfied. If this is not the case, a reliable numerical solution may only be available if the method explicitly forces $h < \mathtt{HMAX}$.

To justify an LEPUS strategy we introduce, for $x \in (x_{i-1}, x_i]$,

$$\tilde{z}_i(x) = z_i(x) + \frac{x - x_{i-1}}{h_i}(y_i - z_i(x_i)). \tag{3.7}$$

Clearly $\tilde{z}_i(x)$ interpolates (x_{i-1}, y_{i-1}) and (x_i, y_i). Therefore the corresponding piecewise interpolant, $\tilde{z}(x)$, provides a continuous approximation to $y(x)$ over $(a, b]$ which interpolates the underlying discrete solution $(x_i, y_i)_{i=0}^{N_{\mathtt{TOL}}}$. In addition we have

$$\begin{aligned}
\tilde{\delta}_i(x) &:= \tilde{z}'(x) - f(x, \tilde{z}(x)) \\
&= \tilde{z}_i'(x) - f(x, \tilde{z}_i(x)) \\
&= z_i'(x) + \frac{1}{h_i}(y_i - z_i(x_i)) - f(x, z_i(x) + \frac{(x - x_{i-1})}{h_i}(y_i - z_i(x_i))) \\
&= \frac{1}{h_i}(y_i - z_i(x_i)) + [f(x, z_i(x)) - f(x, z_i(x) + \frac{(x - x_i)}{h_i}(y_i - z_i(x_i)))].
\end{aligned}$$

Hence, letting L denote the Lipschitz constant for $f(x, y)$, we have

$$|||\tilde{\delta}_i(x)||| \leq (1 + h_i L)\frac{\| y_i - z_i(x_i) \|}{h_i}. \tag{3.8}$$

It follows from (3.6) that a reliable error estimate will give

$$|||\tilde{\delta}_i(x)||| \leq (1 + h_i L)5\mathtt{TOL}. \tag{3.9}$$

If the estimate is asymptotically correct then from (3.8) we see that (3.4) will be satisfied with $K_1 = 1$. Even for 'non-small' stepsizes, the error control strategy will maintain numerical stability, and hence $|h_i L|$ will never be greater than around 6. Thus, from (3.9) the bound

$$|||\tilde{\delta}(x)||| \leq 35\mathtt{TOL}.$$

can be expected to hold.

The natural way to implement a LEPUS strategy with an order $(p-1, p)$ Runge-Kutta formula pair is to advance with the lower order approximation (i.e. y_i defined by (1.2) is the lower order formula and $\hat{y}_i$ defined by (1.6) is the higher order formula). The difference $est_i = y_i - \hat{y}_i$ then gives an asymptotically correct local error estimate. It has been found that if the coefficients of the formula pair are carefully chosen, then this type of error estimate behaves reliably on a wide range of problems.

Note that if the method used a LEPUS strategy as discussed above but advanced with the higher order approximation $\hat{y}_i$ then the extra accuracy would be difficult to recognize or exploit. We would no longer have tolerance proportionality as K_1 of (3.4) would be $O(h)$ and very sensitive to both problem and method.

The second error control strategy we will consider is that which was referred to in the first section and which is used in most contemporary initial value methods. It can be viewed as an indirect control of the local error per unit step. Here, with a main formula of order p, the secondary formula has order $p-1$. With this strategy, denoted LEPSX, the local error per step of the lower order approximation is monitored but local extrapolation is used. In this case if $\widehat{est}_i$ is an asymptotically correct estimate for the local error in $\hat{y}_i$ (the lower order approximation) then

$$m_i = \| \widehat{est}_i \| .$$

This strategy can be justified by considering the $h_i \to 0$ expansion of the local errors in y_i and $\hat{y}_i$;

$$le_i := y_i - z_i(x_i) = \psi_{p+1}(x_i) h_i^{p+1} + O(h_i^{p+2}), \tag{3.10}$$

and

$$\widehat{le}_i := \hat{y}_i - z_i(x_i) = \hat{\psi}_p(x_i) h_i^p + O(h_i^{p+1}). \tag{3.11}$$

Clearly we also have

$$\widehat{est}_i := \hat{y}_i - y_i = \hat{\psi}_p(x_i) h_i^p + O(h_i^{p+1}). \tag{3.12}$$

The error coefficients $\psi_{p+1}(x)$ and $\hat{\psi}_p(x)$ are complicated vector-valued functions that depend on the problem and the coefficients defining the underlying formula pair (see section 2). It follows from (3.10), (3.11) and (3.12) that an asymptotically correct expression for $\| le_i \|$ is given by

$$h_i \frac{\| \widehat{\mathrm{est}}_i \| \| \psi_{p+1}(x_i) \|}{\| \hat{\psi}_p(x_i) \|}.$$

(We assume that $\hat{\psi}_p(x_i)$ does not vanish.) Requiring $m_i \leq \mathtt{TOL}$ is then asymptotically equivalent to requiring

$$\frac{\| le_i \|}{h_i} \leq \frac{\| \psi_{p+1} \|}{\| \hat{\psi}_p \|} \cdot \mathtt{TOL}. \tag{3.13}$$

From (3.13) we can show (using an argument analogous to that presented for the LEPUS strategy) that there exists $\hat{z}(x)$ interpolating $(x_i, y_i)_{i=0}^{N_{\mathtt{TOL}}}$ with an associated defect satisfying

$$|||\hat{\delta}(x)||| \leq K_1 \cdot \mathtt{TOL} + O(h_i \mathtt{TOL}), \tag{3.14}$$

with $K_1 = |||\psi_{p+1}(x)|||/|||\hat{\psi}_p(x)|||$. The value of K_1 thus depends strongly on the coefficients of the underlying formulas as well as the differential equation. As a result, the corresponding K_2 in (3.5) will be sensitive to the problem and the method and will not necessarily be largely determined by the mathematical conditioning of the underlying problem.

While it is possible to implement a local error per step strategy without local extrapolation this is not usually recommended, since in this case K_1 of (3.4) will be $O(h^{-1})$ and we will not observe tolerance proportionality.

The third error control strategy we consider is appropriate for use with explicit Runge-Kutta formulas that have continuous extensions, $\widehat{z}_i(x)$, characterised by (1.7). For such formulas, m_i can be taken as an estimate of the maximum magnitude over $[x_{i-1}, x_i]$ of the defect associated with $\widehat{z}_i(x)$. Such as strategy can be viewed as a direct attempt to ensure that the defect $\widehat{\delta}(x)$ satisfies (3.4) with K_1 not too large.

Defect control schemes (for various choices of $\widehat{z}_i(x)$ and m_i) have been derived and analyzed extensively by Enright [20, 21] and Higham [38, 41]. In particular these investigations have identified conditions on the interpolating scheme which result in the existence of a problem-independent $\tau^* \in (0,1)$ for which, with $m_i = \| \widehat{\delta}_i(x_{i-1} + \tau^* h_i) \|$, (3.4) is satisfied with $K_1 = 1$. Such an error control strategy will be denoted by SDEF (for strict defect control). The strong conditions on the interpolating scheme required to obtain this result are virtually equivalent to imposing the restriction that the stages defining the continuous extension,

Strategy	Order				
	4	5	6	7	8
LEPUS	6 (2)	8 (3)	10 (5)	13 (8)	16 (10)
LEPSX	4 (1)	6 (2)	8 (3)	10 (5)	13 (8)
SDEF	9	12	16	22	27
RDEF	6	9	11	15	21

Table 3.1: Costs (in derivative evaluations per step) of various error control strategies.

k_j, $j = 1, 2, \ldots, \widehat{s}$, also provide a $(p+1)$st order approximation to the solution. In this case an asymptotically correct estimate of the local error in the pth order solution is available at no extra cost (other than the $\widehat{s}$ derivative evaluations) and direct defect control of the pth order scheme is closely related to LEPUS control.

On the other hand we can relax the condition $K_1 = 1$ in (3.4) and ask that K_1 be close to unity for a large class of problems and, on most problems, only mildly sensitive to the method. (See [20, 21] for a detailed analysis and experiments with this approach.) The key idea is to derive a continuous extension $\widehat{z}_i(x)$ and choose τ^* with $m_i = \| \widehat{\delta}_i(x_{i-1} + \tau^* h_i) \|$ in such a way that the corresponding value of K_1 will be small (typically less than 10) for all problems for which the principal term in the expansion of the defect is dominated by the contribution of any single elementary differential. (In particular there can be no significant cancellation in the principal error term associated with the expansion of $\widehat{\delta}_i(x_{i-1} + \tau^* h)$.) This relaxed requirement allows the derivation of less expensive interpolating schemes which still perform well over a wide range of problems and tolerances. We will denote such an error control strategy by RDEF (for relaxed defect control). The observed values of K_1 in (3.4) and K_2 in (3.5) determined numerically for RDEF strategies are generally much less sensitive to the particular method than those associated with LEPSX strategies (see [21]).

Table 3.1 presents a comparison of the cost per step (measured in derivative evaluations) of the various error control strategies we have discussed for Runge-Kutta formulas of orders four through eight. We report the cost of the least expensive scheme known for a particular order. (See [100] for a justification of many of these entries.) For the LEPUS and LEPSX strategies we also report in parentheses the additional cost required to form an interpolant of the same local order on a step. Note that the costs for RDEF and SDEF include the cost of sampling the defect (computing $m_i = \| \widehat{\delta}_i(x_{i-1} + \tau^* h_i) \|$).

4 Global Error Estimation

We saw in section 3 that standard Runge-Kutta software controls a *local* quantity. This has the effect of indirectly controlling the global error—in particular, for many of the commonly used techniques it can be proved that the global error behaves like a linear multiple of the error tolerance. This quantifies the *rate* at which the global error tends to zero. It is perhaps more important, however, to know the actual *size* of the global error for a particular error tolerance. Hence, the availability of a global error estimate is a valuable addition to a Runge-Kutta code. Although several global error estimation techniques have been proposed and rigorously justified, surprisingly little of this work has filtered through to the available software. The code **GERK** [82] and the **RKSUITE** collection [4] are notable exceptions.

In this brief summary we concentrate on methods that have proved successful with explicit Runge-Kutta algorithms. (For a general review of global error estimation we recommend [62, 88].) We discuss four techniques:

- Re-integration with a smaller tolerance,
- Richardson extrapolation,
- Zadunaisky's technique, and
- Solving for the error estimate.

Re-integrating a problem with a smaller tolerance is a simple idea that can be used in conjunction with any code. If the error control satisfies certain natural conditions then it can be shown [39] that a continuous extension $\widehat{z}(x)$ with $q = p$ in (6.2) will satisfy

$$\widehat{z}(x) - y(x) = v(x)\mathrm{TOL} + o(\mathrm{TOL}), \tag{4.1}$$

where $v(x)$ is a C^1 function, independent of TOL. It follows that if we compute solutions $\widehat{z}^{[1]}(x)$ and $\widehat{z}^{[2]}(x)$ using tolerances $\mathrm{TOL}^{[1]}$ and $\mathrm{TOL}^{[2]}$, respectively, the ratio

$$\frac{\widehat{z}^{[1]}(x) - \widehat{z}^{[2]}(x)}{\mathrm{TOL}^{[1]} - \mathrm{TOL}^{[2]}} \tag{4.2}$$

provides an asymptotically correct (as $\mathrm{TOL} \to 0$) estimate of $v(x)$, and hence of the global error in either approximation. (Note that the introduction of an interpolant is necessary in general. We cannot compare the global errors 'at the meshpoints' in the two cases, because the location of the meshpoints varies with TOL.) This approach has the advantage of simplicity, but it is generally more expensive than the best alternative. (For other numerical methods, in particular those based on variable order linear multistep formulas, this technique is much less reliable since the relation (4.1) does not hold.)

Richardson extrapolation, or step halving, is used in [4, 82]. Here a solution $\{y_i\}$ is computed, using standard error control, from a stepsize sequence $\{h_i\}$. A second numerical solution $\{\bar{y}_i\}$ is also computed by applying the same Runge-Kutta formula over pairs of steps with stepsize $h_i/2$. An estimate for the global error in $\bar{y}_i$ is then given by

$$\frac{y_i - \bar{y}_i}{2^p - 1}, \tag{4.3}$$

where p is the order of the Runge-Kutta method. This technique has been rigorously justified by Henrici [36], under mild assumptions, for the case of smoothly varying stepsizes. We also mention that the approach is asymptotically equivalent to re-integrating with a second tolerance of $\mathrm{TOL}^{[2]} = \mathrm{TOL}^{[1]}/2^p$. In particular, the analysis in [39] can also be used to justify this global error estimate.

The next method that we discuss was proposed by the Argentinian astronomer Zadunaisky [104], and has been rigorously analyzed by several numerical analysts. (See, for example, the references in [88].) Here, in addition to solving the main problem $y'(x) = f(x, y(x))$, $y(x_0) = y_0$, we construct an interpolant $q(x)$ to the numerical solution and solve a neighboring problem, the most common form of which is

$$\widehat{y}'(x) = \widehat{f}(x, \widehat{y}(x)) := f(x, \widehat{y}(x)) + \delta(x), \quad \widehat{y}(x_0) = y_0, \tag{4.4}$$

where $\delta(x) := q'(x) - f(x, q(x))$ is the defect in $q(x)$. The neighboring problem (4.4) is solved with the same Runge-Kutta method and the same stepsize sequence as the main problem. By construction, the exact solution to (4.4) is $\widehat{y}(x) = q(x)$. Hence, we can compute the exact global error in the neighboring computation, and use this as our estimate for the global error in the main computation. Intuitively, this is reasonable if the neighboring problem is 'close' to the main problem; that is, if the two problems give rise to similar local errors and have similar error propagation properties.

In early references, the interpolant $q(x)$ in (4.4) was taken to be a polynomial that matched various combinations of $\{y_i, f_i\}$ data across a range of steps. However, this can give rise to oscillatory polynomials, and is susceptible to loss of accuracy when the interpolant straddles steps with widely differing stepsizes. More recently, Dormand et al. [15] avoided these difficulties by taking $q(x)$ to be a one-step continuous extension, of the form described in section 6. They showed that in order to make the global error estimate asymptotically valid, the Runge-Kutta formula must have certain truncation coefficients equal to zero. Hence, a special purpose formula must be used for both the main problem and the neighboring problem. Examples of suitable formulas can be found in [15].

Solving for the error estimate, or, in Skeel's terminology [88], solving for the correction, also involves the use of an interpolant $q(x)$. Here, we construct the secondary problem

$$\epsilon'(x) = \widehat{f}(x, \epsilon(x)) := q'(x) - f(x, q(x) - \epsilon(x)), \quad \epsilon(x_0) = 0. \tag{4.5}$$

This problem clearly has solution $\epsilon(x) = q(x) - y(x)$, which is the global error for the main problem. The secondary problem (4.5) is solved with the same stepsize sequence as that used for the main problem, but, generally, with a different Runge-Kutta formula. The numerical solution to the secondary problem is then used as the global error estimate. Clearly, in order for this estimate to be valid, the secondary problem must be solved more accurately than the main one. The key to developing an efficient method is to exploit the special structure of the secondary problem. (The expansion for the local error when a formula is applied to (4.5) is closely related to that for the difference between local errors for the main and neighboring problems in Zadunaisky's method.) Dormand et al. [15] showed that special formulas can be constructed which give the desired accuracy on (4.5) with fewer stages than standard formulas.

Solving for the error estimate has two main advantages over Zadunaisky's method. First, evaluating the right-hand side in the secondary problem (4.5) involves one f evaluation, contrasting with the two f evaluations that are needed for the right-hand side of (4.4). This leads to a more efficient process. Further, and equally importantly, Zadunaisky's method requires the same 'special' formula to be used for the main and neighboring problems, whereas solving for the error estimate can be combined with any Runge-Kutta formula. Hence, with the latter approach, the main integration can be done with a formula that has good overall properties. Based on theoretical and numerical measurements, Dormand et al. conclude that solving for the error estimate allows a reliable global error estimate to be computed at about twice the cost of a standard integration.

Overall, if a global error estimate is desired, we recommend two options. Re-integrating with a smaller tolerance is a general purpose approach that can be used with any good quality software. Solving for the error estimate with formulas such as those developed in [15] is a more complicated procedure that requires extra coding, but increases efficiency and reliability.

5 Stiffness detection

ERK methods can be effective tools for solving nonstiff problems; however, they are unsuitable for problems with more than a modest degree of stiffness. With a reliable error control mechanism an ERK code would produce accurate results on a stiff problem, but it would be forced to take extremely small stepsizes, rendering the code much less efficient than a good stiff solver. It is therefore useful to be able to determine those cases where an ERK method is being unduly constrained by stiffness. Armed with this information a user could try switching to a different method. A more sophisticated approach is to attempt to determine whether the problem could be completed more efficiently with a stiff method, and, if so, to switch automatically. We outline below some of the techniques that have been developed for stiffness detection in ERK methods, paying particular attention to the work of Shampine [76, 77]. A thorough overview of the area is given by Robertson [64]. We make no attempt here to give a precise definition of "stiffness"; for a clear discussion the reader is referred to [80].

To gain some insight into the expected behavior of an ERK method in the presence of stiffness, it is customary to rely on the linear test problem

$$y'(x) = Ay(x), \tag{5.1}$$

where the eigenvalues of $A \in \mathbb{R}^{N\times N}$ have negative real parts. We can tentatively interpret (5.1) as modeling the local behavior of a nonlinear system of ODEs with A representing the local Jacobian $\partial f/\partial y$. Applying an ERK formula to (5.1) we find

$$y_{i+1} = S(h_i A)y_i, \tag{5.2}$$

where the stability polynomial, S, depends on the coefficients of the formula. The true solution consists of decaying exponential terms and hence in order to mimic the qualitative behavior of $y(x)$ we must use stepsizes for which $\rho(S(h_i A)) < 1$, where $\rho(\cdot)$ denotes the spectral radius. We will refer to the eigenvalue(s), λ, of A for which this condition is most restrictive on the stepsize as the dominant eigenvalue(s). Shampine [75] shows that on (5.1), with standard error control techniques, a code will use a stepsize sequence such that the average stepsize h_{av} corresponds to being close to the absolute stability boundary; that is, $h_{\text{av}} \approx h_S$, where $|S(h_S\lambda)| = 1$. If $|\lambda|$ is large then this represents an extremely severe restriction on the stepsizes. (This is why ERK methods are unsuitable for very stiff problems.) Note that stepsizes will not necessarily remain constant. A detailed analysis by Hall [35] shows that two distinct situations are possible—$h_i\lambda$ either remains almost exactly on the stability boundary or fluctuates around it. In the latter case frequent step rejections add to the inefficiency of a method. For the case of absolute error control, Hall gave algebraic conditions on the coefficients of a formula which determine which of the two conditions will arise. Extensions of this result to more general weighting schemes are given in [42].

Shampine [76] proposes a simple, but effective, way to detect stiffness when reasonably high order ERK formulas (say order ≥ 4) are being used. He notes that with any such ERK formula the first two function evaluations on a step can be combined to give

$$y_{i+1}^{E} = y_i + c_2 h_i k_1, \quad y_{i+1}^{ME} = y_i + \frac{c_2 h_i}{2}\{k_1 + k_2\}, \tag{5.3}$$

which are the results of steps of length $c_2 h_i$ with the first order Euler method, and the second order Modified Euler method, respectively. (Note that in the above we assume that $c_2 = a_{2,1}$ which is true for all practical ERK formulas.) Differencing the two results gives an approximation to the local error in the Euler result. Hence, along with the usual local error estimate for the step, we may also examine the size of $y_{i+1}^{E} - y_{i+1}^{ME}$ and compare it with the local error tolerance. Shampine regards a problem as stiff if this "comparison formula" passes the local error test at least twenty five times in a sequence of fifty steps. (Clearly, by using more stages from the underlying ERK formula, more general comparison formulas can be derived.)

The rationale behind this idea is that on a typical nonstiff problem the difference in the order of y_{i+1} and y_{i+1}^{E} will manifest itself clearly. Hence the lower order formula will be unlikely to pass the local error test with the kind of stepsize used by the higher order formula. However, if stiffness is affecting the methods, then stability properties will come into effect and a sufficiently stable lower order formula should succeed. Robertson [64] gives further insight. Noting that the local error estimates for the underlying formula and comparison formula are differences between Runge-Kutta solutions, we see from (5.2) that they may be written in the form

$$e_{i+1} = E(h_i A) y_i, \quad \widehat{e}_{i+1} = \widehat{E}(h_i A) y_i, \tag{5.4}$$

respectively, where E and $\widehat{E}$ are polynomials. We can also argue from (5.2) that the numerical solution y_i should be rich in the direction of the dominant eigenvector(s). It then follows that the ratio $|\widehat{E}(h_S \lambda)|/|E(h_S \lambda)|$ will roughly determine the relative sizes of the two local error estimates. Hence the comparison formula should be chosen so that this ratio is sufficiently small for all $h_S \lambda$ on the stability boundary.

Note that the comparison formula test can be performed on every step without a significant increase in overhead. In the numerical tests of Robertson the technique worked well, never wrongly diagnosing a nonstiff problem as stiff, and usually recognizing a stiff problem. It was found that stiffness was detected least successfully at the more stringent tolerance levels. The detection rate can be improved by raising the order of the comparison formula, but clearly if the order is too high then the chance of the comparison formula succeeding on a nonstiff problem is increased. Finally we point out that result of the comparison tests is a black-or-white answer, "yes" or "no". As we implied at the start of this section, it may be useful to generate more information so that a decision about the relative merits of continuing with the ERK formula and switching to an alternative method can be made. In particular it would be useful to investigate the size of the dominant eigenvalue(s) of the local Jacobian. We describe below an approach put forward by Shampine [77] which provides a lower bound on the local Lipschitz constant, and can be used to obtain an estimate of the spectral radius of the local Jacobian.

The first result we must recall is that, for sufficiently smooth f, the Mean Value Theorem gives

$$f(x, u + \epsilon) - f(x, u) = J\epsilon, \tag{5.5}$$

where J represents a Jacobian of f whose rows are evaluated at points on the line between u and $u+\epsilon$. Hence by choosing sufficiently small perturbations, ϵ, we may apply a power method type algorithm to this local Jacobian by computing only f values. (Since we are essentially forming a finite difference approximation to a derivative, care must be taken to choose ϵ small enough to give useful information, but large enough to avoid unnecessary cancellation.) If the ERK formula has nodes which satisfy $c_i = c_j$ for some $i \neq j$, then the difference $k_j - k_i$ has the form of (5.5), and is available at little cost. A more expensive alternative is discussed in [77]. Ignoring the details introduced to avoid undue rounding errors, Shampine's algorithm is as follows:

Let $y^{(0)} = y_i$ and choose a vector $y^{(1)}$.
For $m = 1, 2, \ldots, l$
Let $y^{(m+1)} = y^{(0)} + \frac{1}{\rho_m}[f(x_i, y^{(m)}) - f(x_i, y^{(0)})]$,
where $\rho_m = ||f(x_i, y^{(m)}) - f(x_i, y^{(0)})||/||y^{(m)} - y^{(0)}||$.

Using (5.5) we see that $y^{(m+1)} - y^{(0)}$ has the form

$$y^{(m+1)} - y^{(0)} = \frac{1}{\rho_m} J[y^{(m)} - y^{(0)}],$$

and hence

$$y^{(m+1)} - y^{(0)} \approx \frac{1}{\rho_m \rho_{m-1} \cdots \rho_1} J^m [y^{(1)} - y^{(0)}],$$

where J now denotes the "local Jacobian". We see that $y^{(m+1)} - y^{(0)}$ resembles m steps of the power method applied to $y^{(1)} - y^{(0)}$ and hence each ρ_m is an approximation to the spectral radius of J. If J has a complex conjugate pair of dominant eigenvalues, then the sequence ρ_m can oscillate wildly. In this case a least-squares technique due to Wilkinson [102] may be used, see [64] for more details. It is clear that approximations to the dominant eigenvalue(s), rather than the spectral radius, could also be generated. Also note that, by definition, we may regard $\max_{1 \leq m \leq l}\{\rho_m\}$ as a lower bound on the Lipschitz constant of f. (Shampine uses $l = 3$.)

Robertson argues that the local error estimate for the step is likely to be rich in components of the dominant eigenvector(s) and hence a suitably normalized version is a good choice for the starting vector, $y^{(1)} - y^{(0)}$. The resulting algorithm was found to perform reliably in practice, and can be used to complement the cheaper, but less informative, comparison formula technique.

It is also pointed out in [64] that once stiffness is suspected, other tests can be performed in an attempt to confirm that a problem is stiff and to analyze the problem more carefully. The theory developed by Hall [35] gives a great deal of information about the expected behavior of the stepsize, error estimate, and numerical solution on stiff problems. Using Hall's results, by monitoring the effect of changes in the ERK formula, the error tolerance, or the stepsize selection process, it may be possible to verify stiffness, compute the approximate size and argument of the dominant eigenvalue(s), and decide whether it would be advantageous to switch to an implicit method. This appears to be a promising line of investigation for improving the robustness of existing stiffness detection techniques.

To conclude this section, we mention that even with the 'simplified' linear models (5.1) and (5.2), eigenvalues do not always give accurate quantifications of stiffness. Eigenvalues determine the long term behaviour ($x \to \infty$ and $x_i \to \infty$) but this may be misleading for two reasons. First, error control algorithms work on a local, step-by-step basis, and hence the *intermediate* behaviour (after a finite number of steps) is significant. Second, in general, the system (5.1) is a much simplified model that is only likely to be relevant over a small interval. Higham and Trefethen [44] address these issues and give examples where the classical eigenvalue viewpoint underestimates the stiffness and fails to explain the behaviour of an adaptive algorithm. They show that a better understanding arises from the use of *pseudo-eigenvalues*.

6 Continuous extensions

Conventional explicit Runge-Kutta formulas are designed to approximate the solution of (1.1) on a discrete mesh. The meshpoints are usually determined by a stepsize selection strategy as described in section 3. Such strategies will typically select the largest possible stepsizes under the constraint that some measure of the local error is bounded by a predefined tolerance. But sometimes an approximation to the solution is required at points which do not coincide with the meshpoints. The obvious remedy is to modify the mesh to include these additional points. But apart from the fact that this might severely affect the efficiency of the implementation, it might also happen that the location of the additional points is not known a priori. For instance, after the numerical integration process reaches the point x_n it may become clear that the solution is required at some x where $x < x_n$. We mention five applications where a continuous extension of the discrete solution obtained by a conventional Runge-Kutta method is required, or at least useful. Three of these applications are discussed in sections 3,4 and 7: monitoring the defect $\widehat{\delta}(x)$ as a means of error control, estimating the global error, and handling discontinuities/singularities. A fourth example arises when the numerical solution is required at a dense set of output points, for instance for the purpose of producing a graphical representation of the solution. Finally, consider a version of (1.1) where one or more *delay arguments* are present in the function f; e.g.

$$y'(x) = f(x, y(x), y(x-\tau)), \tag{6.1}$$

where $\tau > 0$ is a *delay* that may depend on x, or even on the solution $y(x)$. In many cases this type of system can be solved numerically by a method for standard ODE's, provided an approximation to the delay argument $y(x-\tau)$ can be obtained; for instance by a continuous extension of a Runge-Kutta method.

Having established the need for a continuous approximation to the solution, there are several ways to proceed. The first interpolants for explicit Runge-Kutta methods were based on interpolation over two or more meshpoints using Hermite or Hermite-Birkhoff interpolation. However, with this approach the desirable one-step nature possessed by the Runge-Kutta method is lost if more than two meshpoints are included in the continuous approximation. Thus, in this paper we will restrict our attention to one-step continuous extensions of the form (1.7). Using the notation from the introduction, it is natural to define the *uniform order* of the continuous extension as the greatest integer q for which

$$\max_{\tau \in [0,1]} \| \widehat{z}(x_{i-1} + \tau h_i) - z_i(x_{i-1} + \tau h_i) \| = O(h_i^{q+1}). \tag{6.2}$$

Since we are assuming that the continuous extension evaluated at the endpoint, $\tau = 1$, reduces to the discrete formula, it follows that $q \le p$ where p is the order of the discrete formula and that the piecewise polynomial approximation is globally continuous. It has been argued (see, for example, [18]) that for the purpose of treating dense output, it is sufficient that $q = p-1$. To see this, recall that the global error at the meshpoints is of order p while the continuous formula contributes an error term of order $q+1$. However, there are other applications where it necessary to have $q = p$ so that the local error in the continuous extension is asymptotically negligible compared with the global error at the meshpoints (see, for example, [20, 21, 39]).

Surveying the literature of the past decade, we find that a large variety of techniques have been applied for the construction of continuous explicit Runge-Kutta formulas. These techniques fall naturally into two classes: those which are based exclusively on solving the continuous version of the order conditions [58, 59, 100] and those which apply classical interpolation theory, possibly combined with solving order conditions, see, among others, [9, 18, 19, 22, 45, 60, 73, 74, 78, 79, 94].

The first significant contributions to the theory of one-step or local interpolants were made by Horn [45] and Shampine [78, 79]. Given a Runge-Kutta formula of order p, we wish to construct an interpolant of uniform order $q(\le p)$. First, we see if we can form approximations to $y(x)$ of local order $q+1$ somewhere within the step, possibly by adding some stages $k_{s+1}, \ldots, k_{s^*}$. That is, we seek $\tau \notin \{0,1\}$ (fixed) and a set of weights b_r^τ such that

$$y_{i-1+\tau} = y_i + h_i \sum_{r=1}^{s^*} b_r^\tau k_r \tag{6.3}$$

and such that the local error, $le_{i-1+\tau} := y_{i-1+\tau} - z_i(x_{i-1} + \tau h_i) = O(h_i^{q+1})$. The procedure for constructing approximations of the type (6.3) is based on solving order conditions where the step size h_i is replaced by τh_i. We will say more about this later. For the moment, assume that the procedure has been carried out, say $m-1 \le q-1$ times for distinct values $\tau = \tau_1, \ldots, \tau_{m-1}$ to obtain local approximations $y_{i,1}, \ldots, y_{i,m-1}$ together with the end point approximations $y_{i,0} := y_{i-1}$ and $y_{i,m} := y_i$. To be consistent we define $\tau_0 := 0$ and $\tau_m := 1$. Now we may find approximations to $z_i'(x)$ at some, say $l+1 \le m+1$, of these points, simply by forming $f_{i,j} := f(x_{i-1} + h_i\tau_j, y_{i,j})$ for all $j \in S$, where $S \subseteq \{0, \ldots, m\}$ is a set of $l+1$ indices. Notice that $f_{i,0} := z_i'(x_{i-1}) = k_1$ is already known and that $f_{i,m}$ can be obtained at practically no additional cost as it will usually serve as k_1 in the subsequent step. Thus, with $m+1$ approximations to $z_i(x)$ and $l+1$ approximations to $z_i'(x)$ one may construct a Hermite interpolating polynomial $\widehat{z}_i(x)$ of degree $l+m+1$ such that

$$\begin{aligned} \widehat{z}_i(x_{i-1} + \tau_j h_i) &= y_{i,j}, \quad j = 0, \ldots, m+1, \\ \widehat{z}_i{}'(x_{i-1} + \tau_j h_i) &= f_{i,j}, \quad \forall j \in S. \end{aligned}$$

If $m+l+1 \geq q$ then it follows from classical interpolation theory that the continuous approximation $\widehat{z}_i(x)$ is of uniform order q.

This approach has been used with some success by the authors mentioned above, by Dormand and Prince [18, 19], and more recently by Calvo et al. [9] for some of the most popular Runge-Kutta formulas. The key task is to find the qth order approximations within the step. This has frequently been done in a somewhat ad hoc manner. For instance, Horn pointed out that for the Fehlberg pair RKF45, a one-parameter family of 4th order approximations could be found at the point $\tau = 3/5$ without adding stages. Shampine adds one stage to the (5,4) pair by Dormand and Prince [17] to obtain a 5th order approximation at an arbitrary point. A similar approach is used in [9]. Enright et al. [22] obtain a one-parameter family of 5th order approximations at $\tau = 1/2$ for a pair of order (5,6) constructed by Verner. This approximation involves the 8 original stages of the method including $f(x_i, y_i)$.

Next we consider a general approach suggested in [22]. Given a RK formula of order p and some C^1 interpolant of uniform order $q_0 < p$ this technique provides a general algorithm for the construction of a C^1 interpolant of order $q = p$ or $q = p-1$. The algorithm is based on a "boot-strapping" procedure such that, starting with the interpolant of uniform order q_0 each step produces a new interpolant of one order higher. Notice that it is always possible to obtain $q_0 \geq \min\{3, p\}$ simply by computing the cubic Hermite interpolant from the function and derivative values at the end points. We describe here one such step. Assume that an interpolant of uniform order $q \geq q_0$ is available through a polynomial $u_q(x)$ of degree q. Assume also that $q < p$. Choose points $\tau_j \in (0,1), j = 1, \ldots, q-2$ and compute the polynomial $u_{q+1}(x)$ of degree $q+1$ such that

$$\begin{aligned} u_{q+1}(x_{i-1}) &= y_{i-1}, \quad u_{q+1}(x_i) = y_i, \\ u'_{q+1}(x_{i-1}) &= k_1, \qquad u'_{q+1}(x_i) = k_{s+1} := f(x_i, y_i), \\ u'_{q+1}(x_{i-1} + \tau_j h_i) &= k_{s+j+1} := f(x_{i-1} + \tau_j h_i, u_q(x_{i-1} + \tau_j h_i)). \end{aligned} \tag{6.4}$$

It is always possible to find points τ_j such that $u_{q+1}(x)$ is uniquely determined, but since this is a Hermite-Birkhoff interpolation problem, the points must be chosen with some care to ensure that a solution of (6.4) exists. Now, again assuming that f is sufficiently smooth, it can be shown that the new polynomial $u_{q+1}(x)$ approximates the solution with uniform order $q+1$. This procedure can be applied repeatedly until $q = p-1$ or $q = p$ depending on which is desired. We remark that this approach is general, and does not require information obtained from the order conditions. But naturally, to minimize the cost of the interpolant, it is recommended to use as much information as possible from the stages of the underlying discrete method.

It can be easily verified that all the interpolants above can be written in the form (1.7). It is of interest to classify all continuous methods which satisfy (6.2). The material we present on this topic is mainly due to Owren and Zennaro [58, 59] and Verner [100]. Standard theory (see, for example, [6]) shows that a RK method is of order p iff the method satisfies the order conditions up to order p; that is, $\Phi(t) = 1/\gamma(t)$ for all rooted trees t with order $r(t) \leq p$. Here $\gamma(t)$ is the *density* of the rooted tree while $\Phi(t)$ depends on the coefficients that define the RK method. For all t, $\Phi(t)$ depends linearly on the weights b_r of the method, hence $\Phi(t) = \sum_r b_r \phi_r(t)$ where $\phi_r(t)$ depends only on the coupling coefficients a_{ij}. Now, observe that for a fixed τ a continuous method of the type (1.7) reduces to a conventional RK method with weights $\tilde{b}_r(\tau)$ using step size τh_i. It then follows from the discrete theory that the continuous method (1.7) satisfies (6.2) if and only if for every $\tau \in [0,1]$

$$\sum_{r=1}^{\widehat{s}} \tilde{b}_r(\tau)\phi_r(t) = \frac{\tau^{r(t)}}{\gamma(t)}, \quad \text{for all } t : r(t) \leq q. \tag{6.5}$$

By assigning a unique index to each tree we may define the matrix $\Phi := ((\phi_{ij}))$ where $\phi_{ij} := \phi_j(t_i)$. The matrix Φ is $N_q \times \widehat{s}$, where N_q is the number of trees in (6.5). Defining the two vectors of polynomials $\tilde{b}(\tau) = (\tilde{b}_1(\tau), \ldots, \tilde{b}_{\widehat{s}}(\tau))^T$ and $p(\tau) = (p_i(\tau))$ with $p_i(\tau) = \tau^{r(t_i)}/\gamma(t_i)$ we can write (6.5) simply as

$$\Phi\tilde{b}(\tau) = p(\tau).$$

Notice that the right hand side is independent of the coefficients of the method. Observe that if Φ has full rank then all the polynomial weights must be of degree $\leq q$ and satisfy $b_r(0) = 0$. It is customary (see, for

Order q	EN(q)	CEN(q)
1	1	1
2	2	2
3	3	4
4	4	6
5	6	8
6	7	11

Table 6.1: Order barriers for discrete and continuous explicit RK methods.

example, [58, 100]) to assume that $b_r(\tau) = \sum_{k=1}^{q} b_{rk}\tau^k$. Thus, by defining the $\widehat{s} \times q$ matrix $B := ((b_{rk}))$ and the $N_q \times q$ matrix $P := ((p_{jk}))$ where $p_{ik} := 1/\gamma(t_i)$ if $k = r(t_i)$ and $p_{ik} = 0$ otherwise, we arrive at the matrix equation

$$\Phi B = P. \tag{6.6}$$

This equation characterizes all continuous Runge-Kutta methods of order q. The least number of stages required to obtain an explicit method of uniform order q is known for $q \leq 6$ and is given in Table 6.1 along with the corresponding numbers for conventional (discrete) RK methods. These numbers are denoted EN(q) and CEN(q), respectively. For a general theory of such order barriers, see [58]. The result for CEN(6) in Table 6.1 was obtained by Santo [66]. It is shown in [59] that for $q = 3, 4, 5$ the value of CEN(q) can be effectively reduced by one using a FSAL (first-same-as-last) stage. Studying Table 6.1, one sees that for $2 \leq p \leq 5$ the formula CEN(p) $= 2p - 2$ holds. Carnicer [13] has shown that for general p one has the lower bound CEN(p) $\geq 2p - 2$ under the condition that either $c_i \neq 0,\ i = 2, \ldots, \hat{s}$ or $a_{i,i-1} \neq 0,\ i = 2, \ldots, p$.

Sometimes it may also be useful to consider the matrix obtained by attaching the rows of Φ to the rows of P, i.e. we consider the $N_q \times (\widehat{s} + q)$ matrix $\Psi = [\Phi, P]$. Observe that a solution to (6.6) exists if and only if rank(Ψ)=rank(Φ). Moreover, given a RK method of order p with a corresponding matrix Φ of elementary weights, it is easy to verify that at least rank(Ψ)-rank(Φ) additional stages are required to construct a continuous extension of uniform order p.

As for higher order discrete Runge-Kutta methods, the analysis of continuous methods becomes exceedingly more difficult as the order increases. It also seems that the discrepancy between the number of stages needed for discrete and continuous methods increases rapidly. To our knowledge, the cheapest known continuous method of order 8 has 20 stages. Verner [100] has recently investigated high-order continuous extensions by considering the continuous order conditions. Starting with a RK pair of some order $(p, p-1)$ he constructs continuous extensions of order p or $p-1$ simply by adding new stages successively until a solution to (6.6) can be found. This procedure, like the one in [22], is general in the sense that it will always lead to a continuous extension of the desired uniform order. However, it is not guaranteed that the total number of stages needed to obtain uniform order p equals CEN(p) as defined above. We mention that some authors argue that the order of the interpolant is not necessarily its most important property. For example, [3] and [31] investigate the use of low order interpolants that guarantee to reproduce monotonicity or convexity in the discrete data.

A third way to obtain continuous explicit RK methods of arbitrary order is by means of extrapolation with an explicit basis method. Simonsen [87] shows that extrapolation based on the Euler method with harmonic step sequence yields an extrapolation table similar to the discrete one where all the entries are polynomial approximations with uniform order equal to the (discrete) order of the corresponding entries in the discrete extrapolation table. A similar result holds for the smoothed midpoint rule, but in that case there is the restriction on the step sequence, n_k, that it cannot grow more slowly than $n_k = 4k + 2$.

Finally, we address the question of global smoothness of continuous extensions. This question is particularly easy to answer for the case where classical interpolation techniques are used. In that case it is clear that the global polynomial approximation is C^1 if Hermite interpolation conditions are imposed at the end points. In the approach with continuous order conditions, it can be shown [59], that the uniform approximation is C^1 if and only if the matrix Φ of (6.6) has full rank and that one of the stages is the so called FSAL evaluation, i.e. $f_{i,j} = f(x_i, y_i)$ for some j. We remark that this stage cannot be involved in the discrete approximation, since, if so, the method would be implicit. Higham [40] considers continuous extensions with a higher degree of smoothness. He gives a procedure for constructing piecewise polynomial approximations such that the kth derivative is continuous at the meshpoints for arbitrary non-negative k. In this case, since the degree of the polynomial exceeds the uniform order when $k > 1$, it follows that some higher derivatives of the local error will become unbounded as the stepsize tends to zero.

7 Discontinuities and Singularities

In many practical applications there are discontinuities in the exact solution, or in one of its higher derivatives. In such cases standard numerical methods will encounter difficulties since the order of the underlying formula will effectively drop on steps that attempt to span the point of discontinuity. A similar difficulty arises if the solution passes through (or near) a point where the derivative is singular. Techniques to detect these special difficulties using a continuous extension have been introduced and analyzed in [23] and [24] for discontinuities and singularities, respectively. In this section we present a combined approach which covers both situations.

In the case of a discontinuity, a reliable numerical method will generally change the stepsize erratically as it attempts to cross the point of discontinuity. This is because the mathematical problem changes character at such a point and the accuracy of the underlying formula deteriorates abruptly. A reduction in stepsize (or a sequence of such reductions) will result in a step that does not attempt to span the point of discontinuity and the corresponding approximate solution will be more accurate than necessary (and the associated error estimate will reflect this). With standard error control this behavior (a sequence of rejected steps followed by an accurate successful step) will be repeated until either the stepsize associated with an attempt to span the discontinuity is so small that the error control criterion is satisfied or the stepsize falls below the minimum allowable stepsize and the method fails. Early detection of discontinuous points, followed by a restart, is the most efficient way to handle this difficulty.

In the case of a point of singularity, a reliable numerical method will exhibit a less abrupt sequence of stepsize reductions as the point of singularity is approached. This is due to the fact that the mathematical problem changes rapidly, but generally smoothly, near a singular point. In contrast to the case of a discontinuity there is no obvious general purpose technique to allow the integration to proceed in a reliable way. Halting the integration with a message indicating the location of the suspected singular point is an appropriate response.

The following approach is designed to recognize when either of the above difficulties is affecting the progress of an integration and to take the appropriate action automatically. After a successful step, the method checks whether a significant stepsize reduction has occurred, either suddenly over a single step or gradually over a sequence of several steps. If so, it is suspected that a point of discontinuity or singularity lies close ahead, and the method forms a continuous extension, $\hat{z}_i(x)$ (see section 6). If the leading coefficients of $\hat{z}_i(x)$ exhibit the characteristics expected of a series approximation to a singular function, then a nearby singular point is signaled and the integration is abandoned. Several techniques to identify the existence and location of a nearby singularity were investigated and found to be effective in [24].

If monitoring $\hat{z}_i(x)$ does not indicate the existence of a nearby point of singularity, a check for the existence of a point of discontinuity is performed by sampling the associated defect, $\hat{z}_i'(x) - f(x, \hat{z}_i(x))$, in the extended interval $[x_i + h,\ x_i + 2h]$. If the magnitude of the defect grows more rapidly than expected in this interval then a point of discontinuity is signaled. The location of this point is accurately determined by a bisection technique and $\hat{z}_i(x)$ is used to obtain 'initial values' to restart the integration at a point just beyond the point of discontinuity. Details concerning how accurately this point must be determined and an analysis of the resulting errors are presented in [23].

Overall, this automatic approach has proved effective on a wide variety of discontinuous and singular problems without requiring any additional information about the problem. In some cases where a discontinuous problem is specified using *switches* (or a *status vector*) it may be more efficient for a user to explicitly restart the integration after a switching point is located. In this case the user will make explicit use of $\hat{z}_i(x)$ to locate the switching point on those steps where a change in one (or more) switches is detected. This latter approach can be more efficient, as it avoids the rejected steps required to initiate the automatic technique, but the difference is not usually significant relative to the total cost of the integration.

8 Other topics

8.1 Runge-Kutta methods for higher order equations

In applications, particularly in connection with mechanical systems, one often sees differential equations of the form

$$q''(x) = f(x, q(x), q'(x)), \tag{8.1}$$

where $q(x)$ and $q'(x)$ typically represent coordinates and momenta. The obvious way to deal with such equations is to define $p(x) = q'(x)$ and consider the first order system of doubled dimension $(q(x), p(x))' = (p(x), f(x, q(x), p(x)))$. But as discussed in [34], by applying a Runge-Kutta method directly to (8.1) one saves storage, and there might also be a gain in computational efficiency. Nyström [54] was the first to consider this approach which can be formulated as follows

$$\begin{aligned} k_i &= f(x_n + \gamma_i h, q_n + \gamma_i h p_n + h^2 \sum \alpha_{ij} k_j, p_n + h \sum a_{ij} k_j), \quad i = 1, \ldots, s, \\ q_{n+1} &= q_n + h p_n + h^2 \sum \beta_i k_i, \\ p_{n+1} &= p_n + h \sum b_i k_i. \end{aligned}$$

Fehlberg [27] derived the first such high order pairs, but as has been pointed out in several subsequent papers, his methods only control the error in q and not in p. Dormand et al. [16], who considered methods of this type applied to a variant of (8.1) where q' is absent in the right hand side, commented that the error control strategy as proposed by Fehlberg could be unreliable. In more recent papers, it has therefore been prevalent to design error control strategies which also control the error in p. Fine [28] considered low order Runge-Kutta-Nyström methods, and was perhaps the first to observe an efficiency gain over conventional Runge-Kutta methods applied to the transformed problem. This effect was confirmed by Sharp and Fine in [83, 84]. In particular, they constructed a (6,5) pair with 8 stages that seems superior both to Fehlberg's methods and to the best of the traditional methods applied to the transformed problem.

Apparently, there are few papers on Runge-Kutta methods applied directly to equations with third or higher order derivatives. Cooper [14] derived the order conditions for equations of the form

$$y^{(n)}(x) = f(y(x), y'(x), \ldots, y^{(n-1)}(x)).$$

Recently Sharp et al.[85] derived methods for the problem

$$y'''(x) = f(x, y(x)).$$

Such equations have been used to model the draining of a fluid down a wall and the coating of a surface by a fluid.

8.2 Symplectic Runge-Kutta methods

In recent years, much research has been devoted to finding numerical methods for solving Hamiltonian systems. These are even-dimensional systems defined by means of a Hamiltonian, $H(\mathbf{p}, \mathbf{q})$, which frequently arises naturally as the energy function of a mechanical system. The equations

$$\frac{dp_i}{dx} = -\frac{\partial H}{\partial q_i}, \quad \frac{dq_i}{dx} = \frac{\partial H}{\partial p_i}, \quad i = 1, \ldots, d, \tag{8.2}$$

define a $2d$-dimensional system with respect to the variables $(p_1, \ldots, p_d, q_1, \ldots, q_d)$. In this brief presentation we shall follow the overview by Sanz-Serna in [68]. An authoritative reference for this material is [72]. In applications to mechanics (see Arnold [1]) the variables $q_1, \ldots, q_d$ are generalized coordinates and $p_1, \ldots, p_d$ are generalized momenta. Often, the Hamiltonian has the form

$$H(\mathbf{p}, \mathbf{q}) = T(\mathbf{p}) + V(\mathbf{q}), \tag{8.3}$$

where $T(\mathbf{p})$ and $V(\mathbf{q})$ typically represent the kinetic and potential energy of a mechanical system, respectively. Hamiltonians having the form (8.3) are referred to as *separable*. In particular, $T(\mathbf{p})$ may have the form $T(\mathbf{p}) = \frac{1}{2}\mathbf{p}^T\mathbf{p}$, in which case the variables $p_1, \ldots, p_d$ can be eliminated from (8.2) and one is left with a second order system for $q_1, \ldots, q_d$,

$$\frac{d^2 q_i}{dx^2} = -\frac{\partial V}{\partial q_i}, \quad i = 1, \ldots, d.$$

Following [68], we let the function $\phi_{x,H}$ be the phase flow of the system (8.2), i.e. for a fixed x, $\phi_{x,H} : \mathbf{R}^{2d} \to \mathbf{R}^{2d}$ such that $(\mathbf{p}(x), \mathbf{q}(x)) = \phi_{x,H}(\mathbf{p}_0, \mathbf{q}_0)$, where $(\mathbf{p}_0, \mathbf{q}_0)$ are the initial values. Many of the integration methods recently developed for (8.2) are based on the fact that the flow $\phi_{x,H}$ of a Hamiltonian system is a *symplectic* transformation. See [1] for details. In the above framework, $\phi_{x,H}$ being symplectic means that it preserves the differential 2-form

$$\omega_2(d\mathbf{p}, d\mathbf{q}) = d\mathbf{p} \wedge d\mathbf{q} = \sum_{i=1}^{d} dp_i \wedge dq_i.$$

The symbol $\wedge$ signifies the exterior product or wedge product between two differential 1-forms. More precisely, for symplectic transformations g_x one has

$$\omega_2(d\mathbf{p}_0, d\mathbf{q}_0) = \omega_2(dg_x(\mathbf{p}_0), dg_x(\mathbf{q}_0)) = \omega_2(d\mathbf{p}(x), d\mathbf{q}(x)).$$

If $d = 1$, the 2-form ω_2 can be interpreted as an area in $\mathbf{R}^2$ such that a transformation being symplectic can be viewed as preservation of area. This preservation property has many consequences for the long-time behavior of the solutions to (8.2). For instance, it rules out the existence of asymptotically stable equilibria and limit cycles [1].

For Runge-Kutta methods applied to (8.2), one step can be written

$$(\mathbf{p}_{n+1}, \mathbf{q}_{n+1}) = \psi_{h,H}(\mathbf{p}_n, \mathbf{q}_n).$$

To ensure that the long-time behavior of the Runge-Kutta method mimics that of (8.2), it is natural to look for methods such that $\psi_{h,H}$ is a symplectic transformation. Such a method is called a symplectic or canonical Runge-Kutta method. This should not be confused with the property of energy conservation. In fact, $\psi_{h,H}$ being symplectic is a far stronger condition than conservation of energy. Moreover Zhong and Marsden [105] have proved that energy conserving methods cannot be canonical, unless they solve the underlying continuous problem exactly.

To establish the existence of RK methods such that $\psi_{h,H}$ is symplectic, we follow Sanz-Serna [67]. Simply by writing out $\psi_{h,H}$ in terms of the Runge-Kutta coefficients a_{ij} and b_i and by using the bilinearity and skew-symmetry of ω_2, one arrives at the following characterization of canonical Runge-Kutta methods: An irreducible Runge-Kutta method is canonical if and only if

$$b_i a_{ij} + b_j a_{ji} - b_i b_j = 0, \quad 1 \le i, j \le s.$$

Unfortunately, no explicit Runge-Kutta method can satisfy these conditions, since if $j = i$ we get $b_i = 0$, $\forall i$. To obtain explicit canonical Runge-Kutta methods we must resort to partitioned methods, see [34]. Assume that we apply a Runge-Kutta method with coefficients a_{ij} and b_i to the variables $p_1, \ldots, p_d$, and another method with coefficients A_{ij} and B_i to $q_1, \ldots, q_d$. Calvo and Sanz-Serna [69] give the conditions for canonicity

$$b_i A_{ij} + B_j a_{ji} - b_i B_j = 0, \quad 1 \le i, j \le s,$$

if the partitioned method is applied to the separable Hamiltonian system (8.3). There are solutions to these conditions such that $A_{ij} = a_{ij} = 0$ for $j \ge i$. In particular such methods can be found for second order systems

$$\frac{d\mathbf{p}}{dx} = \mathbf{f}(\mathbf{q}), \quad \frac{d\mathbf{q}}{dx} = \mathbf{p}, \quad \text{where } \mathbf{f} = -\nabla V \text{ for some } V(\mathbf{q}),$$

among the class of Runge-Kutta-Nyström methods. They have the form

$$\begin{aligned}
\mathbf{Q}_i &= \mathbf{q}_n + h\gamma_i \mathbf{p}_n + h^2 \sum_{j=1}^{s} \alpha_{ij} \mathbf{f}(\mathbf{Q}_j), \\
\mathbf{p}_{n+1} &= \mathbf{p}_n + h \sum_{i=1}^{s} b_i \mathbf{f}(\mathbf{Q}_i), \\
\mathbf{q}_{n+1} &= \mathbf{q}_n + h\mathbf{p}_n + h^2 \sum_{i=1}^{s} \beta_i \mathbf{f}(\mathbf{Q}_i).
\end{aligned}$$

Such methods are canonical if the coefficients satisfy

$$\beta_i = b_i(1-\gamma_i),\ 1 \le i \le s,\ \ b_i(\beta_j - \alpha_{ij}) = b_j(\beta_j - \alpha_{ji}),\ 1 \le i, j \le s.$$

A simple proof is given by Okunbor and Skeel [56]. The special structure of Hamiltonian systems does not cause elementary differentials in their B-series [34] to vanish. But the conditions for canonicity are simplifying assumptions that lower the number of independent order conditions. A general theory of order conditions for canonical Runge-Kutta methods was first developed by Sanz-Serna and Abia [71]. Similar results for Runge-Kutta-Nyström methods were found by Calvo and Sanz-Serna [11]. Okunbor and Skeel [56] proved that an explicit method is canonical if and only if its adjoint is explicit, and they used this result in [55] to derive families of explicit canonical Runge-Kutta-Nyström methods. For instance, they present the following two-parameter family of canonical explicit methods of order two

$$\gamma_1 = \frac{1}{2} + \nu,\ \gamma_2 = \frac{1}{2} + \mu,\ b_1 = \frac{\mu}{\mu - \nu},\ b_2 = \frac{-\nu}{\mu - \nu},\ \mu_i = b_i(1-\gamma_i),\ i = 1, 2,\ \ \nu \ne \mu.$$

An important practical question is whether it is worthwhile to develop variable stepsize algorithms based on symplectic formulas. Calvo and Sanz-Serna [10] construct a symplectic variable stepsize Runge-Kutta-Nyström method and perform several numerical experiments where they compare this method (SV) with a fixed stepsize version of the same method (SF), a variable stepsize non-symplectic reference method (NSV) and a fixed stepsize version of this reference method (NSF). For a two-body test problem, they find that (SF) seems to work better than the others, the second best is the (NSV) which is about 4/3 times more efficient than the (SV) method. They present efficiency by plotting the global accuracy obtained versus the number of evaluations of the right hand side of (8.2). In [12], they conclude that the reason for this discouraging result is that with variable stepsizes the composition of the numerical phase flows $\psi_{h_1,H_1}\psi_{h_2,H_2},\ldots$ is not a symplectic transformation. This conclusion is supported in a paper by Skeel and Gear [89].

8.3 Dynamical systems

The long-term behavior of ERK methods on nonlinear ODEs can be studied using ideas from the field of *dynamical systems* (fixed points, periodic solutions, bifurcations, etc.). This has recently become an extremely active area of research. Most work has concentrated on fixed stepsize implementations, and has been concerned with the existence of fixed points and periodic solutions of the ERK formula. Key references in the area are [33, 47, 49, 103], and the article [92] summarizes some results for a wider audience.

In this discussion we restrict ourselves to autonomous systems

$$y'(x) = f(y(x)). \tag{8.4}$$

The solution is sometimes referred to as a *flow*. A fixed stepsize numerical method generates a discrete *map* of the form

$$y_{n+1} = F(y_n), \tag{8.5}$$

where the function F is parametrised by the stepsize. (Euler's Method, for example, has $F(y) = y + hf(y)$.) Our aim is now to compare the long term behavior of the map and flow.

Clearly, the constant function $y(x) \equiv y^*$ solves the ODE (8.4) if $f(y^*) = 0$. Such a solution is called a *fixed point* (steady state, equilibrium point, critical point, rest state) of the ODE. To investigate the attractivity of the fixed point, we may write $z(x) = y^* + \epsilon(x)$ and linearize about the fixed point to obtain

$$\epsilon'(x) = \frac{\partial f}{\partial y}(y^*)\epsilon(x). \tag{8.6}$$

We say that the fixed point y^* of (8.4) is *locally attractive* if there exists a neighborhood around y^* such that any solution $y(x)$ entering the neighborhood satisfies $y(x) \to y^*$ as $x \to \infty$. A result attributed to Poincaré and Liapunov [95, Theorem 7.1] shows that the linearized problem (8.6) generally determines local attractivity: under mild assumptions about f, a sufficient condition for a fixed point y^* of (8.4) to be locally attractive is

$$\Re\{\lambda\} < 0, \qquad \text{for every eigenvalue } \lambda \text{ of } \tfrac{\partial f}{\partial y}(y^*). \tag{8.7}$$

Further, replacing "$<$" by "$\leq$" in (8.7) gives a necessary condition for local attractivity.

For the corresponding map (8.5), y^* is a fixed point when $y^* = F(y^*)$. Perturbing to $y_n = y^* + \epsilon_n$, and linearizing, gives

$$\epsilon_{n+1} = \frac{\partial F}{\partial y}(y^*)\epsilon_n. \tag{8.8}$$

We say that the fixed point y^* of (8.5) is *locally attractive* if there exists a neighborhood around y^* such that any solution y_n entering the neighborhood satisfies $y_n \to y^*$ as $n \to \infty$. Ostrowski's Theorem [57, Theorem 10.1.3] shows that a sufficient condition for a fixed point y^* of (8.5) to be locally attractive is

$$\rho(\frac{\partial F}{\partial y}(y^*)) < 1, \tag{8.9}$$

where $\rho(\cdot)$ denotes the spectral radius. Replacing "$<$" by "$\leq$" in (8.9) gives a necessary condition for local attractivity.

To illustrate the relevance of these concepts we consider the scalar ODE

$$y'(x) = y(x)(1 - y(x)), \quad y(0) = y_0. \tag{8.10}$$

(This ODE, the logistic equation, has been proposed as a simple model for population growth where there is potential for overcrowding [53].) The ODE has two fixed points: $y^* \equiv 1$ is locally attractive and $y^* \equiv 0$ is not locally attractive. In fact, due to the simple nature of the ODE, we can solve analytically to give

$$y(t) = \frac{y_0 e^x}{1 + y_0(e^x - 1)}.$$

It follows that $y(x) \to 1$ as $x \to \infty$ for all positive initial data, and $y(x)$ blows up in finite time for all negative initial data.

Applying the two-stage, explicit Modified Euler formula [52, page 155] to this ODE leads to the map

$$y_{n+1} = y_n + h(y_n + .5hy_n(1 - y_n))(1 - y_n - .5hy_n(1 - y_n)). \tag{8.11}$$

This map shares the fixed points $y^* = 0, 1$ of the underlying ODE. (Indeed, it is easy to show that $f(y^*) = 0 \Rightarrow y^* = F(y^*)$ for any Runge-Kutta formula.) However, it can be verified that the map has the additional fixed points $y^* = 2/h$ and $y^* = 1 + 2/h$. Griffiths et al. [33, Table 2.1] examine the Jacobian of the map at the fixed points and deduce the following results.

1. $y^* = 0$ is not locally attractive for any $h > 0$.
2. $y^* = 1$ is locally attractive only for $0 < h < 2$.
3. $y^* = 2/h$ is locally attractive only for $2 < h < 1 + \sqrt{5} \approx 3.236$.
4. $y^* = 1 + 2/h$ is locally attractive only for $0 < h < -1 + \sqrt{5} \approx 1.236$.

Here, 1 and 2 are special cases of a general result: Iserles [49, Theorem 3] shows that if y^* is a fixed point of the ODE (8.4) then examining the local attractivity of y^* for the map (8.5) reduces to examining the absolute stability of the ERK method on the linear problem $y'(x) = \frac{\partial f}{\partial y}(y^*)y(x)$.

In Figure 8.1 we present an associated bifurcation diagram (a similar figure appears in [33, Figure 2]). This diagram was generated by applying the iteration (8.11) for 500 steps and then plotting the last 20 values of y_n. The iteration was performed for the initial values .5, 1.5 and 2.5 and was repeated for a range of h values between 1 and 4. After every iteration, the last 20 values are plotted as dots in the figure. We see that each of the locally attractive fixed points succeeds in attracting the iterates, at least for certain initial values. Further analysis is possible. For the fixed point $y^* = 1 + 2/h$, the Jacobian $\frac{\partial F}{\partial y}(y^*)$ passes through -1 at $h = -1 + \sqrt{5}$ and a 'flip' bifurcation leads to a stable period two solution continuing from this point. With $y^* = 1$ we find that $\frac{\partial F}{\partial y}(y^*)$ passes through $+1$ at $h = 2$. A 'transcritical' bifurcation produces the fixed point $y^* = 2/h$ which then 'flips' into a stable period two solution at $h = 1 + \sqrt{5}$. Analysis for general Runge-Kutta methods can be found in [33].

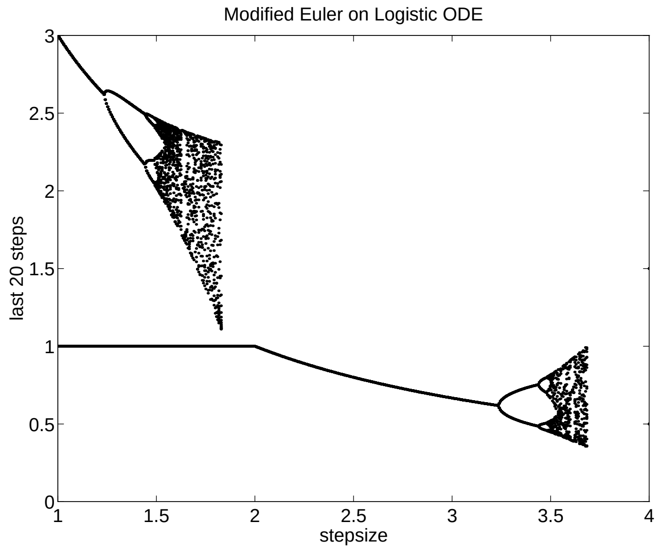

Figure 8.1: Modified Euler Method on Logistic ODE.

From a numerical analysis perspective, it is worrying that a Runge-Kutta formula can generate locally attractive fixed points that are *spurious*; that is, fixed points that are not shared by the underlying ODE. Hence, the iterates can converge to a deceptively smooth but completely incorrect limit. Further, as the example shows, these spurious solutions can exist for arbitrarily small stepsizes. However, we note that in the example above, the spurious fixed point $y^* = 1+2/h$ becomes arbitrarily large as h tends to zero. Hence, the 'conservative' approach of re-applying the formula with a smaller stepsize would quickly alert the user to the fact that this fixed point is spurious. Humphries [47] has proved a general result to this effect. If f is continuously differentiable and the ERK method admits spurious fixed points or period two solutions that exist for arbitrarily small h, then these solutions cannot remain bounded as $h \to 0$.

It should be emphasized that the results mentioned above apply when a fixed stepsize is used. Most modern software selects stepsizes adaptively in order to control an error estimate on every step (see section 3). Sanz-Serna [70, page 101] opines that judicious use of variable stepsizes greatly reduces the possibility of spurious dynamics, and some positive theoretical results in this area have recently been established in [2, 43, 93].

9 Software

In this concluding section we discuss some issues that must be addressed when an explicit Runge-Kutta pair is implemented in a general purpose integrator. We also present a comparison between a popular integrator from 1976 and a state-of-the-art integrator from 1992.

9.1 Simple integrator

Figure 9.1 is an overview of a simple hypothetical integrator called SIMP that uses an embedded pair to control the local error. The user specifies the initial conditions, the output point x_{out}, the tolerance `TOL` and possibly the initial stepsize. If x_{out} is reached, the integration can be continued by changing x_{out} and re-calling SIMP.

> **loop**
>   **if** $x_{i-1} = x_{\text{out}}$ **then**
>   set $ind = 0$, $x = x_i$, $y = y_i$ and **exit**
>   **end if**
>   calculate h_i
>   calculate the stages
>   calculate the local error estimate, est
>   **if** $||\text{est}|| \leq$ `TOL` **then**
>   set x_{i-1} and y_{i-1} to x_i and y_i
>   **end if**
> **end loop**

Figure 9.1: Overview of a simple integrator (SIMP).

The most important part of SIMP is the calculation of the stepsize h; a poorly designed scheme for the calculation can lead to a big reduction in the efficiency and reliability. Even for this simple integrator, there are four parts to the scheme.

9.1.1 Initial stepsize

The aim in choosing the initial stepsize is to select one that is on-scale, so that the local error test is just satisfied. This stepsize will depend on at least the order of the pair, the coefficients of the pair, the form of the differential equation, the tolerance, and the initial conditions. This dependence can make it difficult for the user to supply an on-scale stepsize. To illustrate the dependence, consider the two-body orbital problem

$$y_1'' = -\frac{y_1}{r^3}, \quad y_2'' = -\frac{y_2}{r^3}, \tag{9.1}$$

$$y_1(0) = 1 - e, \quad y_1'(0) = 0, \quad y_2(0) = 0, \quad y_2'(0) = \left(\frac{1+e}{1-e}\right)^{1/2}, \tag{9.2}$$

where $r = [y_1^2 + y_2^2]^{1/2}$, and $0 < e < 1$. Table 9.1 contains an estimate of the on-scale stepsize for DVERK [46] for a range of e and `TOL`.

Several schemes have been proposed for calculating the stepsize automatically. One scheme is to take the stepsize as $C(\texttt{TOL})^{1/p}$, where $p - 1$ is the order of the lower order formula in the pair, and C is a constant which depends on the coefficients of the pair, and possibly the scale of the problem. Another scheme is to use $f(x, y)$ to estimate the derivative terms in the local error estimate. This has the advantage over the first scheme of using more information about the problem. A more elaborate, three phase scheme is proposed in [32].

9.1.2 After an accepted step

A step is accepted if $\| est \| \leq \mathtt{TOL}$. In this case, we have the possibility of increasing h for the next step in an attempt to improve efficiency. The new stepsize is often calculated using the formula

$$\beta \left(\frac{\mathtt{TOL}}{est} \right)^{1/p} h_{\text{old}} \tag{9.3}$$

where h_{old} is the stepsize of the step just accepted, and $0 < \beta < 1$ is a safety factor.

For (9.3) to work well, the principal error term must be slowly varying from one step to the next. If this is not so, the new step will often be rejected. This increases the number of function evaluations needed to complete the integration. One approach which can reduce this increase is to use the ratio formula of Watts [101] in place of (9.3).

9.1.3 After a rejected step

A step is rejected if $\| est \| > \mathtt{TOL}$. On the first or second rejection at the point, formula (9.3) is often used to select the new step. If more rejected steps have occurred, the solution is probably badly behaved at the point (for example, a low order derivative of the solution may have a discontinuity). A new stepsize may then be selected by using a special scheme, or the stepsize can be reduced by a more substantial factor than that given by (9.3).

9.1.4 When near x_{out}

Suppose $x_{\text{out}} = 100$, and SIMP has just stepped from $x = 99$ to $x = 99.9$ with a step of size 0.9. Unless the solution changes significantly, an acceptable stepsize for the next step will be approximately 0.9. However, to hit x_{out}, a stepsize of 0.1 must be used, which would mean the stepsize is being restricted by something other than accuracy requirements. This restriction can complicate the performance of SIMP, particularly if there are many output points.

The effect of the restriction can often be reduced by anticipating the output point. For example, if the stepsize selected after an accepted step would take the integration more than halfway to x_{out} from the current point, the stepsize can be restricted to half of the distance to x_{out}.

9.2 Integrator with interpolants

If the number of output points is large relative to the typical stepsize, the stepsize taken by SIMP will often be restricted by the nearness of an output point, making SIMP inefficient and insensitive to changes in the error tolerance. These effects can be eliminated by using interpolants, as described in section 6; SIMP steps past x_{out} and then uses an interpolant to estimate the solution at x_{out}.

The addition of interpolants permits other extensions to SIMP. The interpolants can be used to estimate the defect and as part of a scheme to cross discontinuities efficiently. Interpolants can also be used to solve g-stops. A g-stop is an equation or equations of the form

$$g(x, y) = 0. \tag{9.4}$$

In many instances, a g-stop can be viewed as the problem of finding x such that y has a prescribed value (the g-stop is then also called implicit output).

One scheme for solving g-stops is the following. After each accepted step (from x_{i-1} to x_i), (9.4) is checked to see if it has a solution on $[x_{i-1}, x_i]$. If it does, (9.4) is solved iteratively, with the required values of y calculated from an interpolant over $[x_{i-1}, x_i]$.

9.3 Other extensions to SIMP

As well as adding interpolants, SIMP can be extended in a number of other ways. These extensions include

- providing more than one ERK pair, with the user being able to specify the pair at the start of the integration. For example, a (3,4), a (5,6) and a (7,8) could be provided; the (3,4) pair for use at lax tolerances or when the solution is badly behaved, the (5,6) for most other integrations, and the (7,8) pair for severe tolerances,
- global error estimation; section 4,
- stiffness detection; section 5.

9.4 Interfaces

There are two interfaces: between the calling program and the integrator, and between the integrator and the routine defining the derivative $f(x, y)$. An ideal interface should be easy to use, while guarding against incorrect usage.

9.4.1 Calling program-integrator interface

The calling program must specify the problem, the accuracy requirement, and possibly the value of options such as the initial stepsize. This information can be passed to the integrator in at least three ways; most integrators use more than one way. One approach is to specify each piece of information as an argument to the integrator. An alternative is to pack the information into an array and pass the array to the integrator. A third way is to use a set-up routine as shown in Figure 9.2. After the user's input has been obtained, a routine (the set-up routine) is invoked. The routine checks that the user's input is valid. If it is not, the routine exits with an error flag set. If the input is valid, the routine saves the input and initializes the integration. The integrator is then called after exiting the set-up routine.

Get the user input (initial conditions, accuracy requirements ...)
Invoke the set-up routine
Stop if the set-up failed. Otherwise invoke the integrator.

Figure 9.2: Use of a set-up routine.

9.4.2 Integrator-derivative interface

One way to supply the derivative is to have a routine that takes N, x and y as input and returns the derivative, with the routine invoked by the integrator. To give the user more flexibility, one or more arguments can be added to the routine to enable the user to pass information from the calling program to the derivative. For example, the derivative may contain a parameter whose value the user would like to specify at run-time instead of at compile-time.

The above way is an example of *forward* communication. Another way to supply the derivative is *reverse* communication. Whenever a derivative evaluation is required, the integrator exits to the calling program, and the derivative routine is invoked from here. The integrator is then invoked with the derivative evaluation passed to it. Reverse communication has the advantage of increasing the functionality of the integrator, but it complicates the structure of the calling program and makes it more difficult to verify the correctness of the integrator.

9.5 Comparison of DVERK and RKSUITE

DVERK [46] was developed at the University of Toronto by Hull, Enright, and Jackson. It has been widely used in its own right, and as part of the IMSL library [48]. A new version of DVERK incorporating many of the features discussed in this paper is being developed at the University of Toronto. RKSUITE [4] was developed at NAG and Southern Methodist University by Brankin, Gladwell and Shampine. It is a state-of-the-art explicit integrator.

Table 9.2 compares some important features of DVERK and RKSUITE. Explicit Runge-Kutta pairs have long been known to provide an efficient and reliable way to integrate non-stiff problems. This has fueled much research into improving their performance, an effort which is reflected in the significant differences between DVERK and RKSUITE.

References

[1] V. Arnold, *Mathematical Methods of Classical Mechanics*, Springer, second ed., 1989.

[2] M. Aves, D. Griffiths, and D. Higham, *Does error control suppress spuriosity?*, tech. report, University of Dundee, in preparation, 1994.

[3] R. Brankin and I. Gladwell, *Shape preserving interpolation for plotting solutions of ODEs*, IMA J. Numer. Anal., 9 (1989), pp. 555–566.

[4] R. Brankin, I. Gladwell, and L. Shampine, *RKSUITE: A suite of explicit Runge-Kutta codes*, in Contributions in Numerical Mathematics, R. Agarwal, ed., World Scientific, Series in Applicable Analysis, vol II, 1993, pp. 41–53.

[5] J. Butcher, *On Runge-Kutta processes of high order*, J. Austral. Math. Soc., 4 (1964), pp. 179–193.

[6] ——, *The Numerical Analysis of Ordinary Differential Equations*, J. Wiley & Sons, 1987.

[7] B. Buzbee, *The SLATEC common mathematical library*, in Sources and Development of Mathematical Software, W. Cowell, ed., Prentice-Hall, 1984.

[8] G. Byrne and A. Hindmarsh, *RK methods prove popular at IMA conference on numerical ODEs*, SIAM News, 23/2 (1990), pp. 14–15.

[9] M. Calvo, J. Montijano, and L. Rández, *A fifth order interpolant for the Dormand and Prince Runge-Kutta method*, J. Comput. Appl. Math., (1990), pp. 91–100.

[10] M. Calvo and J. Sanz-Serna, *Variable steps for symplectic integrators*, in Numerical Analysis 1991, D. Griffiths and G. Watson, eds., 1991, pp. 34–48.

[11] ——, *Order conditions for canonical Runge-Kutta-Nyström methods*, BIT, 32 (1992), pp. 131–142.

[12] ——, *Reasons for failure. The integration of the two-body problem with a symplectic Runge-Kutta-Nyström code with stepchanging facilities*, in Equadiff 91, C. Perelló, C. Simó, and J. de Sola-Morales, eds., World Scientific, Singapore, 1993.

[13] J. Carnicer, *A lower bound for the number of stages of an explicit continuous Runge-Kutta method to obtain convergence of given order*, BIT, 31 (1991), pp. 364–368.

[14] G. Cooper, *A class of single-step methods for systems of nonlinear differential equations*, Math. Comp., 21 (1967), pp. 597–610.

[15] J. Dormand, M. Lockyer, N. McGorrigan, and P. Prince, *Global error estimation with Runge-Kutta triples*, Comp. and Maths. with Appls., 18 (1989), pp. 835–846.

[16] J. Dormand, M. Mikkwy, and P. Prince, *Higher order embedded Runge-Kutta-Nyström formulae*, IMA J. Numer. Anal., 8 (1987), pp. 423–430.

[17] J. Dormand and P. Prince, *A family of embedded Runge-Kutta formulae*, J. of Computational and Applied Maths., 6 (1980), pp. 19–26.

[18] ——, *Runge-Kutta triples*, Comp. and Maths. with Appls., 12 (1986), pp. 1007–1017.

[19] ——, *Practical Runge-Kutta Processes*, SIAM J. Sci. Stat. Comput., 10 (1989), pp. 977–989.

[20] W. Enright, *Analysis of error control strategies for continuous Runge-Kutta methods*, SIAM J. Numer. Anal., 26 (1989), pp. 588–599.

[21] ——, *A new error-control for initial value solvers*, Applied Maths. and Computation, 31 (1989), pp. 288–301.

[22] W. Enright, K. Jackson, S. Nørsett, and P. Thomsen, *Interpolants for Runge-Kutta formulas*, ACM Transactions on Mathematical Software, 12 (1986), pp. 193–218.

[23] ——, *Effective solution of discontinuous IVPs using a Runge-Kutta formula pair with interpolants*, Applied Maths. and Computation, 27 (1988), pp. 313–335.

[24] W. Enright and H. Suhartanto, *Detecting and locating a singular point in the numerical solution of IVPs for ODEs*, Computing, 48 (1992), pp. 161–175.

[25] E. Fehlberg, *Low order classical Runge-Kutta formulae with stepsize control and their application to some heat transfer problems*, Tech. Report TR R-135, NASA, 1969.

[26] ——, *Klassiche Runge-Kutta-Formeln vierter und niedriger Ordnung mit Schrittweiten-Kontrolle und ihre Andwendung auf Wärmeleitungsprobleme*, Computing, 6 (1970), pp. 61–71.

[27] ——, *Classical seventh-,sixth- and fifth-order Runge-Kutta-Nyström formulas with step size control for general second order differential equations*, Tech. Report 432, NASA, 1974.

[28] J. Fine, *Low order practical Runge-Kutta-Nyström methods*, Computing, 38 (1987), pp. 281–297.

[29] I. Gladwell, *Initial value routines in the NAG library*, ACM Transactions on Mathematical Software, 5 (1979), pp. 386–400.

[30] ——, *The NAG library ordinary differential equations chapter and short term plans for its extension*, SIGNUM Newsletter, 20 (1985), pp. 35–40.

[31] I. Gladwell, L. Shampine, L. Baca, and R. Brankin, *Practical aspects of interpolation in Runge-Kutta codes*, SIAM J. Sci. Stat. Comput., 8 (1987), pp. 322–341.

[32] I. Gladwell, L. Shampine, and R. Brankin, *Automatic selection of the initial stepsize for an ODE solver*, J. of Computational and Applied Maths., 18 (1987), pp. 175–192.

[33] D. Griffiths, P. Sweby, and H. Yee, *On spurious asymptotic numerical solutions of explicit Runge-Kutta methods*, IMA J. Numer. Anal., 12 (1992), pp. 319–338.

[34] E. Hairer, S. Nørsett, and G. Wanner, *Solving Ordinary Differential Equations I, Nonstiff Problems*, Springer Verlag, second ed., 1993.

[35] G. Hall, *Equilibrium states of Runge-Kutta schemes*, ACM Transactions on Mathematical Software, 11 (1985), pp. 289–301.

[36] P. Henrici, *Discrete Variable Methods in Ordinary Differential Equations*, John Wiley and Sons, New York, 1962.

[37] K. Heun, *Neue Methode zur approximativen Integration der Differentialgleichungen einer unabhängigen Vëranderlichen*, Zeitschr. für Math. u. Phys., 45 (1900), pp. 23–38.

[38] D. Higham, *Robust defect control with Runge-Kutta schemes*, SIAM J. Numer. Anal., 26 (1989), pp. 1175–1183.

[39] ——, *Global error versus tolerance for explicit Runge-Kutta methods*, IMA J. Numer. Anal., 11 (1991), pp. 457–480.

[40] ——, *Highly Continuous Runge-Kutta Interpolants*, ACM Transactions on Mathematical Software, 17 (1991), pp. 368–386.

[41] ——, *Runge-Kutta defect control using Hermite-Birkhoff interpolation*, SIAM J. Sci. Stat. Comp., 12 (1991), pp. 991–999.

[42] D. Higham and G. Hall, *Runge-Kutta equilibrium theory for a mixed relative/absolute error measure*, in Proc. IMA Conf. on Computational Ordinary Differential Equations, J. Cash and I. Gladwell, eds., Oxford University Press, 1992, pp. 73–85.

[43] D. Higham and A. Stuart, *Local error control and global dynamics (working title)*, tech. report, Stanford University (in preparation), 1994.

[44] D. Higham and L. Trefethen, *Stiffness of ODEs*, BIT, 33 (1993), pp. 285–303.

[45] M. Horn, *Fourth- and fifth-order, scaled Runge-Kutta algorithms for treating dense output*, SIAM J. Numer. Anal., 20 (1983), pp. 558–568.

[46] T. Hull, W. Enright, and K. Jackson, *User's guide for DVERK—a subroutine for solving non-stiff ODEs*, Tech. Report 100, University of Toronto, 1976.

[47] A. Humphries, *Spurious solutions of numerical methods for initial value problems*, IMA J. Numer. Anal., 13 (1993), pp. 263–290.

[48] IMSL, *Reference manual.*

[49] A. Iserles, *Stability and dynamics of numerical methods for nonlinear ordinary differential equations*, IMA J. Numer. Anal., 10 (1990), pp. 1–30.

[50] D. Kahaner, C. Moler, and S. Nash, *Numerical Methods and Software*, Prentice Hall, New Jersey, 1989.

[51] W. Kutta, *Beitrag zur näherungsweisen Integration totaler Differentialgleichungen*, Zeitschr. für Math. u. Phys., 46 (1901), pp. 435–453.

[52] J. Lambert, *Numerical Methods for Ordinary Differential Systems*, Wiley, 1991.

[53] J. Murray, *Mathematical Biology*, Springer-Verlag, second ed., 1993.

[54] E. Nyström, *Ueber die numerische Integration von Differentialgleighungen*, Acta Soc. Sci. Fenn., 50 (1925), pp. 1–54.

[55] D. Okunbor and R. Skeel, *Explicit canonical methods for Hamiltonian systems*, Math. Comp., 59 (1992), pp. 439–455.

[56] ——, *An explicit Runge-Kutta-Nyström method is canonical if and only if its adjoint is explicit*, SIAM J. Numer. Anal., 29 (1992), pp. 521–527.

[57] J. Ortega and W. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, 1970.

[58] B. Owren and M. Zennaro, *Order barriers for continuous explicit Runge-Kutta methods*, Math. Comp., 56 (1991), pp. 645–661.

[59] ——, *Derivation of efficient, continuous, explicit Runge-Kutta methods*, SIAM J. Sci. Stat. Comput., 13 (1992), pp. 1488–1501.

[60] G. Papageorgiou, T. Simos, and C. Tsitouras, *Some new Runge-Kutta methods with interpolation properties and their application to the magnetic binary problem*, Celestial Mechanics, 44 (1988), pp. 167–177.

[61] S.N. Papakostas, C. Tsitouras, and G. Papageorgiou, *A general family explicit Runge-Kutta pairs of order 6(5)*, Report, Technical University of Athens, 26 pages, 1994.

[62] P. Peterson, *Global error estimation using defect correction techniques for explicit Runge-Kutta methods*, Tech. Report 192/86, University of Toronto, 1986.

[63] P. Prince and J. Dormand, *High order embedded Runge-Kutta formulae*, J. Comput. Appl. Math., 7 (1981), pp. 67–75.

[64] B. Robertson, *Detecting stiffness with explicit Runge-Kutta formulas*, Tech. Report 193/87, University of Toronto, 1987.

[65] C. Runge, *Ueber die numerische Auflösung von Differentialgleichungen*, Math. Ann., 46 (1895), pp. 167–178.

[66] M. Santo, *Metodi continui ad un passo per la risoluzione numerica di equazioni differenziali ordinarie*, PhD thesis, University of Udine, Italy, March 1990.

[67] J. Sanz-Serna, *Runge-Kutta schemes for Hamiltonian systems*, BIT, 28 (1988), pp. 877–883.

[68] ——, *Symplectic integrators for Hamiltonian problems: an overview*, Acta Numerica, (1991), pp. 243–286.

[69] ——, *The Numerical Integration of Hamiltonian Systems*, in Computational Ordinary Differential Equations, J. Cash and I. Gladwell, eds., The Institute of Mathematics & its Applications, 1992, pp. 437–449.

[70] ——, *Numerical ordinary differential equations vs. dynamical systems*, in The Dynamics of Numerics and the Numerics of Dynamics, D. Broomhead and A. Iserles, eds., The Institute of Mathematics & its Applications, 1992, pp. 81–106.

[71] J. Sanz-Serna and L. Abia, *Order conditions for canonical Runge-Kutta schemes*, SIAM J. Numer. Anal., 28 (1991), pp. 1081–1096.

[72] J. Sanz-Serna and M. Calvo, *Numerical Hamiltonian Problems*, Chapman and Hall, 1994.

[73] D. Sarafyan, *Continuous approximate solution of ordinary differential equations and their systems*, Comput. Math. Appl, 10 (1984), pp. 139–159.

[74] ——, *Families of continuous approximate processes for the solution of ordinary differential equations*, Comput. Math. Appl, 16 (1988), pp. 887–894.

[75] L. Shampine, *Stiffness and non-stiff differential equation solvers*, in Numerische Behandlung von Differentialgleichungen, L. Collatz, ed., Birkhauser, Basel, Switzerland, 1975, pp. 287–301. Int. Series Numer. Math. 27.

[76] ——, *Stiffness and nonstiff differential equation solvers, II: Detecting stiffness with Runge-Kutta methods*, ACM Transactions on Mathematical Software, 3 (1977), pp. 44–53.

[77] ——, *Lipschitz constants and robust ODE codes.*, in Computational Methods in Nonlinear Mechanics, J. Oden, ed., North-Holland, Amsterdam, 1980, pp. 427–449.

[78] ——, *Interpolation for Runge-Kutta methods*, SIAM J. Numer. Anal., 22 (1985), pp. 1014–1027.

[79] ——, *Some practical Runge-Kutta formulas*, Math. Comp., 46 (1986), pp. 135–150.

[80] L. Shampine and C. Gear, *A user's view of solving stiff ordinary differential equations*, SIAM Review, 21 (1979), pp. 1–17.

[81] L. Shampine and I. Gladwell, *The next generation of Runge-Kutta codes*, in Computational Ordinary Differential Equations, J. Cash and I. Gladwell, eds., The Institute of Mathematics & its Applications, 1992, pp. 145–164.

[82] L. Shampine and H. Watts, *Global error estimation for ordinary differential equations*, ACM Transactions on Mathematical Software, 2 (1976), pp. 172–186.

[83] P. Sharp and J. Fine, *A contrast of direct and transformed Nyström pairs*, J. Comput. Appl. Math., 42 (1992), pp. 293–308.

[84] ——, *Some Nyström pairs for the general second order initial value problem*, J. Comput. Appl. Math., 42 (1992), pp. 279–291.

[85] P. Sharp, A. Koishkin, and R. Vaillancourt, *Explicit Runge-Kutta pairs for third-order ordinary differential equations*. Working manuscript, July 1993.

[86] P. Sharp and J. Verner, *Completely imbedded Runge-Kutta pairs*. to appear in SIAM J. Numer. Anal.

[87] H. Simonsen, *Extrapolation Methods for ODE's: Continuous Approximations, a Parallel Approach.*, PhD thesis, University of Trondheim, Norway, December 1990.

[88] R. Skeel, *Thirteen ways to estimate global error*, Num. Math., 48 (1986), pp. 1–20.

[89] R. Skeel and C. Gear, *Does variable step size ruin a symplectic integrator?*, Physica D, 60 (1992), pp. 311–313.

[90] H. Stetter, *Considerations concerning a theory for ODE-solvers*, in Numerical Treatment of Differential Equations, R.Burlisch, R. Grigorieff, and J. Schröder, eds., Springer, Berlin, 1978, pp. 188–200. Lecture Notes in Mathematics 631, conference 1976.

[91] ——, *Tolerance proportionality in ODE-codes*, in Proc. Second Conf. on Numerical Treatment of Ordinary Differential Equations, R. März, ed., Humboldt University, Berlin, 1980, pp. 109–123. Also in Working Papers for the 1979 SIGNUM Meeting on Numerical Ordinary Differential Equations, Ed. R.D. Skeel, Department of Computer Science, University of Illinois at Urbana-Champagne.

[92] I. Stewart, *News and Views: Numerical methods, warning—handle with care!*, Nature, 355 (1992), pp. 16–17.

[93] A. Stuart and A. Humphries, *An analysis of local error control for dissipative, contractive and gradient dynamical systems*, Tech. Report NA-92-18, Computer Science Department, University of Stanford, California 94305, USA, 1992.

[94] C. Tsitouras and G. Papgeorgiou, *New interpolants for Runge-Kutta algorithms using second derivatives*, Intern. J. Computer Math., 31 (1989), pp. 105–113.

[95] F. Verhulst, *Nonlinear Differential Equations and Dynamical Systems*, Springer-Verlag, 1990.

[96] J. Verner, *Explicit Runge-Kutta methods with estimates of the local truncation error*, SIAM J. Numer. Anal., 15 (1978), pp. 772–790.

[97] ——, *Families of imbedded Runge-Kutta methods*, SIAM J. Numer. Anal., 16 (1979), pp. 857–875.

[98] ——, *Deriving explicit methods*, Report, Department of Mathematics and Statistics, Queen's University, Kingston, 56 pages, 1982.

[99] ——, *Deriving explicit methods or how to solve hard jigsaw puzzles*, (private communication), Invited Presentation at the CMS Seminar on Numerical Analysis, Toronto, July 1982.

[100] ——, *Differentiable interpolants for high-order Runge-Kutta methods*, SIAM J. Numer. Anal., 30 (1993), pp. 1446–1466.

[101] H. Watts, *HSTART—an improved initial step size routine for ODE codes*, Tech. Report SAND86-2633, Sandia National Laboratories, Albuquerque, New Mexico, 1986.

[102] J. Wilkinson, *The Algebraic Eigenvalue Problem*, Oxford University Press, 1965.

[103] H. Yee, P. Sweby, and D. Griffiths, *Dynamical systems approach study of spurious steady state numerical solutions of nonlinear differential equations. 1. The ODE connection and its implications for algorithm developments in computational fluid dynamics*, J. Comp. Phys., 97 (1991), pp. 249–310.

[104] P. Zadunaisky, *A method for the estimation of errors propagated in the numerical solution of a system of ordinary differential equations*, in Proc. of Intern. Astron. Union, Symposium No. 25, Thessaloniki, 1964, Academic Press, 1966, pp. 281–287.

[105] G. Zhong and J. Marsden, *Lie-Poisson Hamilton-Jacobi theory and Lie-Poisson integrators*, Phys. Letters A, 133 (1988), pp. 134–139.

Table 9.1: On-scale initial stepsize for DVERK on an orbital problem

	`TOL`				
e	10^{-2}	10^{-4}	10^{-6}	10^{-8}	10^{-10}
0.1	1.68	0.65	0.29	0.13	0.062
0.3	1.64	0.67	0.29	0.14	0.063
0.5	1.39	0.65	0.29	0.14	0.064
0.7	1.13	0.50	0.24	0.12	0.059
0.9	1.73	0.35	0.20	0.08	0.037

Table 9.2: Comparison of DVERK and RKSUITE

Feature	DVERK	RKSUITE
Pair(s)	(5,6)	(2,3), (4,5), (7,8)
Number of options	9	6
Initial stepsize	Simple formula	3 phase
Dense output	Hit x_{out} exactly	(2,3): interpolate (4,5): interpolate (7,8): hit x_{out} exactly
Global error	No assessment	Assessed
Stiffness	No detection	Detection
Integrator-derivative interface	forward comm.	forward comm.