

The Approximate Runge–Kutta Computational Process

HANS OLSSON

Dept. of Computer Science

Lund University, Box 118

S-221 00 Lund, Sweden.

e-mail: Hans.Olsson@dna.lth.se

GUSTAF SÖDERLIND

Dept. of Computer Science

Lund University, Box 118

S-221 00 Lund, Sweden.

e-mail: Gustaf.Soderlind@dna.lth.se

Abstract.

Implicit Runge–Kutta methods are efficient for solving stiff ODEs and DAEs. To bridge the gap between their theoretical analysis and practical implementation, we introduce the notion of the (Δ, K) –approximate Runge–Kutta process to account for inevitable iteration errors. We prove iteration error bounds uniform with respect to stiffness, and investigate stage derivative reuse for methods having a first explicit stage. The latter technique may result in significant performance gains, also when such methods are used as error estimators. Previous computational heuristics can therefore be replaced by a consistent approach supported by theoretical analysis. The approximate but well-defined computational process is evaluated using approved test problems.

AMS subject classifications: 65L05.

Key words: Runge–Kutta, computational process, stage derivative reuse, error bound.

1 Introduction

We shall consider Runge–Kutta methods for the numerical solution of dynamical systems of the form

$$(1.1) \quad B\dot{y} = f(t, y),$$

where $B \in \mathbb{R}^{d \times d}$ and $f : \mathbb{R} \times \mathbb{R}^d \rightarrow \mathbb{R}^d$. The problems to be considered are ODEs with $B = I_d$ and DAEs. For simplicity we shall drop t from (1.1) as the generalization of the techniques developed in this paper to the nonautonomous case is straightforward.

An s –stage Runge–Kutta method with coefficients (A, b) applied to (1.1) advances the numerical solution a step of size h from (t_n, y_n) by applying three sets of equations,

$$(1.2a) \quad Y = \mathbb{1} \otimes y_n + (A \otimes I_d)h\dot{Y}$$

$$(1.2b) \quad B\dot{Y}_i = f(Y_i); \quad i = 1 : s$$

$$(1.2c) \quad y_{n+1} = y_n + (b^T \otimes I_d)h\dot{Y}.$$

The first, (1.2a), defines the *stage manifold*, which gives a linear relation between stage values Y and stage derivatives¹ \dot{Y} . In tandem with the *differential equation* (1.2b), it uniquely determines the stage values and derivatives under reasonable conditions on method coefficients A and problem properties in terms of B and f , cf. [KS91]. Finally, the *quadrature formula* (1.2c) provides a numerical solution (t_{n+1}, y_{n+1}) .

The two equations (1.2a) and (1.2b) can in general not be solved exactly (exceptions are when (1.1) is linear or in the ODE case when the method (A, b) is explicit). Therefore some iterative procedure must be used, causing the equations to hold only approximately. The possibilities and consequences of a systematic approach to *approximate* the Runge–Kutta equations, however, have not been adequately analyzed in the literature. It is nevertheless known that different approaches may have a significant impact on performance. For example, for ODEs with $B = I_d$, (1.2c) is equivalent to

$$(1.3) \quad y_{n+1} = y_n + (b^T \otimes I_d)hF(Y),$$

with $(e_i^T \otimes I_d)F = f \circ (e_i^T \otimes I_d)$. If the system is stiff (i.e. if the Lipschitz constant $L[hf] \gg 1$) and the stage values Y are only approximate, then the evaluation of $hF(Y)$ “can be disastrously inaccurate”, cf. [Sha80], as this may strongly amplify approximation errors in Y . In contrast, if approximate stage derivatives have been computed it should be preferable to use the quadrature formula (1.2c) instead. Moreover, for the more general problem (1.1) this quadrature formula must be used as (1.3) has no counterpart in DAEs. Such considerations have led to the recommendation to solve for the scaled stage derivatives $h\dot{Y}$ instead of the stage values Y .

A related problem is found in algebraically accurate (i.e. $e_s^T A = b^T$) implicit RK methods having a first explicit stage, such as the Lobatto IIIa methods (e.g. the trapezoidal rule). In such “FSAL” (first [stage] same as last) methods, one has a choice of reusing the last stage from the previous step or recomputing the derivative at the starting point of the current step. The latter approach leads to problems similar to the one mentioned above; an error in y_n might be strongly amplified. *Stage derivative reuse* is sometimes advocated to save function evaluations, but more importantly it may overcome amplification of errors, reduce error transport and achieve better overall performance at the price of utilizing a method which strictly is not a true one-step method. A similar stage derivative reuse technique can also be employed to construct less sensitive error estimators for e.g. Radau IIa methods, cf. [HW96b, p. 123], where the difficulties can be attributed to a first explicit stage in the estimator.

The aim of this paper is to address these issues by defining an approximate Runge–Kutta process, in which the equations (1.2) are approximated in a very specific way. We show that in spite of approximation errors it is possible to achieve error bounds which are uniform with respect to stiffness, and we extend this approach to stage derivative reuse methods, which are analyzed in detail. The problem of constructing error estimators is also considered.

¹Although they are not functions of t , we denote stage derivatives by \dot{Y} .

2 The approximate Runge–Kutta process

We shall define an approximate RK process from the elements of (1.2) and note that in the exact RK process the stages are defined as the solution of (1.2a) and (1.2b). As the latter is generally some transcendental equation, numerically exact solutions cannot be obtained. We may nevertheless require (1.2a) to hold exactly (or down to roundoff level) and contain the remaining errors of the approximate stage in (1.2b).

DEFINITION 2.1. *The stage manifold at η is defined by*

$$(2.1) \quad \mathcal{M}(\eta) = \{(U, h\dot{U}) \in \mathbb{R}^{2sd} : U = \mathbb{1} \otimes \eta + (A \otimes I_d)h\dot{U}\}.$$

Assuming that the system consisting of (1.2a) and (1.2b) is uniquely solvable for any y_n in a sufficiently large domain, we make the following definition:

DEFINITION 2.2. *The exact stage pair $(U^*(\eta), h\dot{U}^*(\eta))$ at η is the unique solution of the system of equations*

$$(2.2a) \quad U = \mathbb{1} \otimes \eta + (A \otimes I_d)h\dot{U}$$

$$(2.2b) \quad B\dot{U}_i = f(U_i); \quad i = 1 : s.$$

We obviously have $(U^*(\eta), h\dot{U}^*(\eta)) \in \mathcal{M}(\eta)$. The computation of this point is, however, terminated after a finite number of iterations, so that only an approximation is obtained. As pointed out before, we require (2.2a) to hold, i.e. we consider computations which yield iterates on the stage manifold $\mathcal{M}(\eta)$.

DEFINITION 2.3. *Let $K \in \mathbb{R}^{s \times s}$ be nonsingular. $(U(\eta), h\dot{U}(\eta)) \in \mathcal{M}(\eta)$ is called a (Δ, K) -approximate stage pair at η if*

$$(2.3) \quad \|(K \otimes I_d)(h\dot{U}(\eta) - h\dot{U}^*(\eta))\| \leq \Delta.$$

The set of (Δ, K) -approximate stage pairs is a ball of radius Δ restricted to the stage manifold. Two particular cases are especially important. First, a (Δ, I_s) -approximation bounds *stage derivative errors* by $\|h\dot{U}(\eta) - h\dot{U}^*(\eta)\| \leq \Delta$. Second, choosing $K = A$ and using (2.2a), we find that a (Δ, A) -approximation correspondingly bounds *stage errors* by $\|U(\eta) - U^*(\eta)\| \leq \Delta$. In the theoretical analysis (Δ, A) -approximations have been discussed, [NT86], and several Runge–Kutta codes also aim for (Δ, A) -approximations with a suitable choice of Δ , cf. [HW96b, p. 120–121], [NT86]. In the following we let $(U(\eta), h\dot{U}(\eta))$ denote a (Δ, K) -approximate stage pair at η .

DEFINITION 2.4. *Let $\dot{U}(\eta; \Delta)$ denote a (Δ, K) -approximate stage derivative. A (Δ, K) -approximate Runge–Kutta step from η is a map \mathcal{R}_h^Δ defined by*

$$(2.4) \quad \mathcal{R}_h^\Delta : \eta \mapsto \eta + (b^T \otimes I_d)h\dot{U}(\eta; \Delta).$$

The exact Runge–Kutta step from η is the map \mathcal{R}_h^0 .

The exact RK process is obtained by taking $\eta = y_n$ and advancing the solution by the exact Runge–Kutta step; since $\dot{U}^*(y_n) = \dot{Y}$ we have $y_{n+1} = \mathcal{R}_h^0(y_n)$. An

approximate step on the other hand will generate a sequence $\{u_n\}$ deviating from $\{y_n\}$. Starting at $\eta = u_n$, an exact RK step yields $u_{n+1}^* = \mathcal{R}_h^0(u_n)$. By contrast, a (Δ, K) -approximate step yields $u_{n+1} = \mathcal{R}_h^\Delta(u_n)$, where u_{n+1} depends on the accuracy of the (Δ, K) -approximate stage pair, which is in principle controlled by the choice of K , the value of Δ and the iterative process used to obtain this approximate stage. As $\Delta \rightarrow 0$, however, a single approximate step tends to the exact RK step. Similarly, the approximate RK process tends to the exact RK process, since iteration errors are added to the truncation errors and therefore enter the error propagation in a similar way. This requires in practical computations that Δ be related to the truncation error tolerance.

To bound the errors we specify the norms as follows. We denote the norm on \mathbb{R}^d (“problem space”) by $|\cdot|_y$. The error bounds depend in a characteristic way on method parameters, which are measured in terms of $|\cdot|_{\text{RK}}$ on \mathbb{R}^s (“method parameter space”). Finally we construct a norm $\|\cdot\|$ on \mathbb{R}^{sd} (“stage space”) from the two first norms by

$$(2.5) \quad \|U\| = (|U_1|_y, \dots, |U_s|_y)^T|_{\text{RK}}.$$

For our conclusions to hold, we define

$$(2.6) \quad |b^T|_{\text{RK}} = |b^T|_p, \quad |u|_y = |Tu|_p,$$

where $T \in \mathbb{R}^{d \times d}$ is nonsingular and $|\cdot|_p$ is the usual l^p norm. Note that $|b^T|_{\text{RK}} = |b^T|_p = |b|_q$, where $1/p + 1/q = 1$. Further we have

$$(2.7) \quad \|v \otimes u\| = |v|_{\text{RK}} \cdot |u|_y, \quad \|A \otimes J\| = |A|_{\text{RK}} \cdot |J|_y,$$

where $v \in \mathbb{R}^s$, $u \in \mathbb{R}^d$, $A \in \mathbb{R}^{s \times s}$ and $J \in \mathbb{R}^{d \times d}$. Although this construction is sufficiently general it may be preferable to make problem and method norms entirely independent so that a correctly adapted algorithm, based on parameters measured by $|\cdot|_{\text{RK}}$, can be used without change for any user-selected problem norm. This can be achieved by choosing $|\cdot|_{\text{RK}}$ as either one of the l^1 or l^∞ norms, combining it with an *arbitrary* $|\cdot|_y$ through (2.5). The basic multiplicative properties (2.7) still hold in this case. Inner product norms are covered by (2.6), taking the l^2 norm on the method parameter space.

THEOREM 2.1. *Let K be nonsingular, let $(U, h\dot{U})$ be a (Δ, K) -approximate stage pair at u_n , and let u_{n+1} be the corresponding (Δ, K) -approximate Runge-Kutta step from u_n . We then have the following*

Stage error bounds:

$$(2.8a) \quad \|U - U^*\| \leq \Delta \cdot |AK^{-1}|_{\text{RK}}$$

$$(2.8b) \quad |U_i - U_i^*|_y \leq \Delta \cdot |e_i^T AK^{-1}|_{\text{RK}}$$

Stage derivative error bounds:

$$(2.9a) \quad \|h\dot{U} - h\dot{U}^*\| \leq \Delta \cdot |K^{-1}|_{\text{RK}}$$

$$(2.9b) \quad |h\dot{U}_i - h\dot{U}_i^*|_y \leq \Delta \cdot |e_i^T K^{-1}|_{\text{RK}}$$

$$(2.9c) \quad |hf(U_i) - hf(U_i^*)|_y \leq \Delta \cdot L_y[hf] \cdot |e_i^T AK^{-1}|_{\text{RK}}$$

Quadrature error bound:

$$(2.10) \quad |u_{n+1} - u_{n+1}^*|_y \leq \Delta \cdot |b^T K^{-1}|_{\text{RK}}.$$

Proof. By (2.2a) the difference between $U(\eta)$ and $U^*(\eta)$ satisfies

$$\begin{aligned} \|U - U^*\| &= \|(A \otimes I_d)(h\dot{U} - h\dot{U}^*)\| \\ &= \|(AK^{-1} \otimes I_d)(K \otimes I_d)(h\dot{U} - h\dot{U}^*)\| \\ &\leq |AK^{-1}|_{\text{RK}} \cdot \Delta. \end{aligned}$$

Bounds for individual stages are obtained by considering

$$\begin{aligned} |U_i - U_i^*|_y &= |(e_i^T \otimes I_d)(U - U^*)|_y \\ &= |(e_i^T AK^{-1} \otimes I_d)(K \otimes I_d)(h\dot{U} - h\dot{U}^*)|_y \\ &\leq |e_i^T AK^{-1}|_{\text{RK}} \|(K \otimes I_d)(h\dot{U} - h\dot{U}^*)\|. \end{aligned}$$

The two stage derivative bounds (2.9a) and (2.9b) follow by applying similar techniques to $\|h\dot{U} - h\dot{U}^*\| = \|(K^{-1} \otimes I_d)(K \otimes I_d)(h\dot{U} - h\dot{U}^*)\|$. The bound (2.9c) is a straightforward Lipschitz condition, where the *lub* Lipschitz constant with respect to $|\cdot|_y$ is defined by $L_y[f] = \sup_{u \neq v} |f(u) - f(v)|_y / |u - v|_y$. Finally, we obtain (2.10) by $|u_{n+1} - u_{n+1}^*|_y = |(b^T \otimes I_d)(h\dot{U} - h\dot{U}^*)|_y$. \square

The implications of the theorem are displayed in Table 2.1, where the bounds are given for the cases $K = I_s$ and $K = A$. The theorem assumes that K is invertible, but by minor extensions of the proof it can be shown that A need only be invertible in three of the bounds in the right column of Table 2.1. Moreover, for algebraically accurate methods ($b^T = e_s^T A$) the (Δ, A) -bound for the quadrature error simplifies to $\Delta \cdot |e_s^T|_{\text{RK}} = \Delta$, even if A should be singular (e.g. for the Lobatto IIIa methods) as b is then in the range of A^T . Note that some bounds are simplified by $|\cdot|_{\text{RK}}$ being an l^p norm, for which $|e_i^T|_{\text{RK}} = 1$.

The importance of a (Δ, K) -approximate step is that it bounds the influence of iteration errors on the end point of a step. The choice $K = A$ corresponds to a termination criterion for stage values. As A^{-1} appears in some of the (Δ, A) -bounds, however, special care may have to be taken for methods having a large $|A^{-1}|_{\text{RK}}$. In that case small stage errors do not imply a small error at the end point. The choice $K = I_s$ is then preferable, corresponding to a termination criterion based on stage derivatives. Other values of K may also be of interest. We note e.g. that the implementation of RADAU5 [HW96b, p. 121-122] uses a transformation of A to block diagonal form, and a corresponding accuracy criterion should then be used.

We further note that the bounds for $|hf(U_i) - hf(U_i^*)|_y$ are nonuniform with respect to stiffness, as measured by the Lipschitz constant $L_y[hf]$. This indicates that for stiff ODEs it is of importance not to approximate stage derivatives by

(Δ, K) -approximation error bounds

Quantity	(Δ, I_s) -bound	(Δ, A) -bound
$\ U - U^*\ $	$\Delta \cdot A _{\text{RK}}$	Δ
$ U_i - U_i^* _y$	$\Delta \cdot e_i^T A _{\text{RK}}$	Δ
$\ h\dot{U} - h\dot{U}^*\ $	Δ	$\Delta \cdot A^{-1} _{\text{RK}}$
$ h\dot{U}_i - h\dot{U}_i^* _y$	Δ	$\Delta \cdot e_i^T A^{-1} _{\text{RK}}$
$ hf(U_i) - hf(U_i^*) _y$	$\Delta \cdot L_y[hf] \cdot e_i^T A _{\text{RK}}$	$\Delta \cdot L_y[hf]$
$ u_{n+1} - u_{n+1}^* _y$	$\Delta \cdot b^T _{\text{RK}}$	$\Delta \cdot b^T A^{-1} _{\text{RK}}$

Table 2.1: Iteration error bounds implied by Theorem 2.1.

$f(U_i)$ when the stage value U_i is in error, cf. Figure 2. We shall next analyze this difference briefly.

Let $B = I_d$ and consider the ODE $\dot{y} = f(y)$. Further, assume that the method (A, b) has an invertible A . For a (Δ, K) -approximate stage pair at η and the corresponding exact stage pair we let $\delta U = U - U^*$ and $\delta \dot{U} = \dot{U} - \dot{U}^*$ denote the errors of the stage values and the stage derivatives, respectively. From this we obtain two different expressions for the relation between δU and $\delta \dot{U}$. Because $(U, h\dot{U})$ is on the stage manifold,

$$(2.11) \quad h\delta \dot{U} = (A^{-1} \otimes I_d)\delta U,$$

which shows that the stage derivative error is uniformly bounded in terms of the stage error and vice versa:

$$(2.12) \quad \|h\delta \dot{U}\| \leq |A^{-1}|_{\text{RK}} \cdot \|\delta U\|, \quad \|\delta U\| \leq |A|_{\text{RK}} \cdot \|h\delta \dot{U}\|.$$

The exact stage pair satisfies the ODE, i.e.

$$(2.13) \quad h\dot{U}^* = (I_s \otimes hf)(U^*),$$

but the approximation has a residual $h\rho$,

$$(2.14) \quad h\dot{U} = (I_s \otimes hf)(U) + h\rho,$$

which *may* be very large if $L_y[hf] \gg 1$, i.e. in the presence of stiffness. Since

$$(2.15) \quad h\delta \dot{U} = (I_s \otimes hf)(U) - (I_s \otimes hf)(U^*) + h\rho$$

must be uniformly bounded according to the left inequality of (2.12), there must be a considerable cancellation between $h\rho$ and $(I_s \otimes hf)(U) - (I_s \otimes hf)(U^*)$. We therefore conclude that *if $h\rho$ is large, then $\|(I_s \otimes hf)(U) - h\dot{U}^*\| \gg \|h\delta \dot{U}\|$*

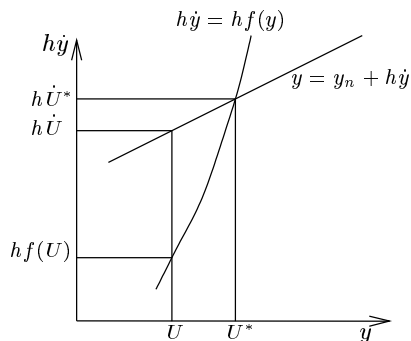


Figure 2.1: The exact stage pair $(y_{n+1}, h\dot{y}_{n+1})$ is the intersection of the stage manifold $y = y_n + h\dot{y}$ and the nonlinear manifold $h\dot{y} - hf(y) = 0$, here illustrated for the implicit Euler method. We require iterates to remain on the stage manifold as opposed to requiring $h\dot{y} = hf(y)$. When $L_y[hf]$ is large (“stiffness”), the latter approach may incur large errors in the stage derivative, but the former yields uniform error bounds.

and $(I_s \otimes hf)(U)$ is inferior to $h\dot{U}$ as an approximation to $h\dot{U}^*$. The reason for defining the (Δ, K) -approximation as restricted to the stage manifold is thus that it is possible to obtain error bounds which are uniform with respect to stiffness. The (Δ, K) -approximate RK step, $u_{n+1} = u_n + (b^T \otimes I_d)h\dot{U}$, admits such uniform error bounds, which cannot be obtained if the quadrature were to proceed by the formula $u_{n+1} = u_n + (b^T \otimes I_d)(I_s \otimes hf)(U)$, cf. (1.3). We remark that, while this is the theoretical motivation behind the definitions proposed in this paper, it may in some cases still be possible to achieve acceptable results with (1.3); even in the presence of stiffness it is not always the case that $\|h\rho\|$ is large—in many stiff problems the stiff subspace varies sufficiently slowly for the Newton iteration to generate accurate approximations to stage derivatives within a few iterations.

3 Stage derivative reuse

Some implicit RK methods have a first explicit stage, which—in the case of ODEs—formally requires the evaluation of $f(u_n)$. If the method in addition is algebraically accurate ($b^T = e_s^T A$), the stage pair from the previous step contains $U_s = u_n$ as well as the corresponding stage derivative, $\dot{U}_s = \dot{u}_n$, which is the accepted (Δ, K) -approximation to $\dot{U}_s^* = f(u_n^*)$. We can then reuse \dot{u}_n on the next step to avoid a reevaluation of $f(u_n)$ which may incur greater errors than those present in \dot{u}_n . Examples where this technique can be employed are the Lobatto IIIa methods, ESDIRK methods [AC90, Kvæ92], and embedded error estimators for algebraically accurate methods such as Radau IIa. The latter methods provide \dot{u}_n , and it is common to embed an error estimator with a first explicit stage, [HW96b, p. 123]. Such estimators may however overestimate stiff

error components unless special care is taken; here we suggest *stage derivative reuse* (SDR), i.e. reusing \dot{u}_n , to overcome these problems.

We partition an RK method (A, b) with an explicit first stage as

$$A = \begin{pmatrix} 0 & 0 \\ \tilde{a} & \tilde{A} \end{pmatrix}; \quad b = \begin{pmatrix} b_1 \\ \tilde{b} \end{pmatrix}$$

where we assume that \tilde{A} is invertible and that $b^T = e_s^T A$ (also written $\tilde{b}^T = \tilde{e}_s^T \tilde{A}$). The stages are partitioned conformally, with Y_1 denoting the first stage and \tilde{Y} the vector of the remaining stages. Moreover, if the method is algebraically accurate the s th stage pair satisfies $(Y_s, \dot{Y}_s) = (y_{n+1}, \dot{y}_{n+1})$.

The SDR computational process starts a step from a pair $(\eta, h\dot{\eta})$. To define this process we follow the same outline as in the true one-step case treated in the previous section.

DEFINITION 3.1. *The SDR manifold at $(\eta, h\dot{\eta})$ is defined by*

$$(3.1) \quad \mathcal{M}(\eta, h\dot{\eta}) = \{(V, h\dot{V}) \in \mathcal{M}(\eta) : (V_1, h\dot{V}_1) = (\eta, h\dot{\eta})\}.$$

We note that this implies a further restriction of the previously defined stage manifold $\mathcal{M}(\eta)$. Next we assume that, for any $(\eta, h\dot{\eta})$ in a sufficiently large domain, the system

$$(3.2a) \quad \tilde{V} = \tilde{\mathbf{l}} \otimes \eta + \tilde{a} \otimes h\dot{\eta} + (\tilde{A} \otimes I_d)h\dot{V}$$

$$(3.2b) \quad B\dot{V}_i = f(V_i); \quad i = 2 : s$$

has a unique solution, denoted by $(\tilde{V}^*(\eta, h\dot{\eta}), h\tilde{V}^*(\eta, h\dot{\eta}))$.

DEFINITION 3.2. *The exact SDR pair at $(\eta, h\dot{\eta})$ is defined by*

$$(3.3) \quad V^*(\eta, h\dot{\eta}) = \begin{pmatrix} \eta \\ \tilde{V}^*(\eta, h\dot{\eta}) \end{pmatrix}, \quad h\dot{V}^*(\eta, h\dot{\eta}) = \begin{pmatrix} h\dot{\eta} \\ h\tilde{V}^*(\eta, h\dot{\eta}) \end{pmatrix}.$$

The exact SDR pair is therefore in $\mathcal{M}(\eta, h\dot{\eta})$. We also note that for the exact Runge–Kutta process, the exact stage pair at y_n is identical to the exact SDR pair at $(y_n, h\dot{y}_n)$, where \dot{y}_n is the last stage derivative on the previous step, equal to $f(y_n)$ in the case of ODEs. Thus,

$$(3.4) \quad U^*(y_n) = V^*(y_n, h\dot{y}_n) = Y, \quad \dot{U}^*(y_n) = \dot{V}^*(y_n, h\dot{y}_n) = \dot{Y}.$$

In the SDR process, only $s-1$ stages are computed on each step, and we therefore define norms analogous to (2.5) and (2.6) by $\|\tilde{V}\| = (|\tilde{V}_2|_y, \dots, |\tilde{V}_s|_y)^T|_{\text{SDR}}$, where $|\tilde{b}^T|_{\text{SDR}} = |\tilde{b}^T|_p$ is defined on \mathbb{R}^{s-1} and $|\cdot|_y$ remains unchanged.

DEFINITION 3.3. *Let $\tilde{K} \in \mathbb{R}^{(s-1) \times (s-1)}$ be nonsingular. $(V(\eta, h\dot{\eta}), h\dot{V}(\eta, h\dot{\eta})) \in \mathcal{M}(\eta, h\dot{\eta})$ is called a (Δ, \tilde{K}) -approximate SDR pair at $(\eta, h\dot{\eta})$ if*

$$(3.5) \quad \|(\tilde{K} \otimes I_d)(h\tilde{V}(\eta, h\dot{\eta}) - h\tilde{V}^*(\eta, h\dot{\eta}))\| \leq \Delta.$$

(Δ, \tilde{K}) -approximation error bounds

Quantity	(Δ, I_{s-1}) -bound	(Δ, \tilde{A}) -bound
$\ \tilde{V} - \tilde{V}^*\ $	$\Delta \cdot \tilde{A} _{\text{SDR}}$	Δ
$ V_i - V_i^* _y$	$\Delta \cdot \tilde{e}_i^T \tilde{A} _{\text{SDR}}$	Δ
$\ h\tilde{V} - h\tilde{V}^*\ $	Δ	$\Delta \cdot \tilde{A}^{-1} _{\text{SDR}}$
$ h\dot{V}_i - h\dot{V}_i^* _y$	Δ	$\Delta \cdot \tilde{e}_i^T \tilde{A}^{-1} _{\text{SDR}}$

Table 3.1: Iteration error bounds implied by Theorem 3.1.

DEFINITION 3.4. Let $(V(\eta, h\dot{\eta}; \Delta), h\dot{V}(\eta, h\dot{\eta}; \Delta))$ denote a (Δ, \tilde{K}) -approximate SDR pair. A (Δ, \tilde{K}) -approximate SDR step from $(\eta, h\dot{\eta})$ is a map \mathcal{S}_h^Δ ,

$$(3.6) \quad \mathcal{S}_h^\Delta : (\eta, h\dot{\eta}) \mapsto (V_s(\eta, h\dot{\eta}; \Delta), h\dot{V}_s(\eta, h\dot{\eta}; \Delta)).$$

The exact SDR step from $(\eta, h\dot{\eta})$ is the map \mathcal{S}_h^0 .

Thus the approximate SDR computational process takes $(\eta, h\dot{\eta}) = (v_n, h\dot{v}_n)$ and advances one step by computing $(v_{n+1}, h\dot{v}_{n+1}) = \mathcal{S}_h^\Delta(v_n, h\dot{v}_n)$. The effects of iteration errors are then bounded in a way analogous to Theorem 2.1.

THEOREM 3.1. Let (A, b) be an algebraically accurate Runge–Kutta method with a first explicit stage. Further, let $(V, h\dot{V})$ be a (Δ, \tilde{K}) -approximate SDR pair at $(v_n, h\dot{v}_n)$, and let $(v_{n+1}, h\dot{v}_{n+1})$ be the corresponding (Δ, \tilde{K}) -approximate SDR step from $(v_n, h\dot{v}_n)$. Then the bounds given in Table 3.1 hold and

$$(3.7a) \quad |v_{n+1} - v_{n+1}^*|_y \leq \Delta \cdot |\tilde{e}_s^T \tilde{A} \tilde{K}^{-1}|_{\text{SDR}} = \Delta \cdot |\tilde{b}^T \tilde{K}^{-1}|_{\text{SDR}},$$

$$(3.7b) \quad |h\dot{v}_{n+1} - h\dot{v}_{n+1}^*|_y \leq \Delta \cdot |\tilde{e}_s^T \tilde{K}^{-1}|_{\text{SDR}}.$$

Proof. For $i = 2 : s$ the $s - 1$ vector \tilde{e}_i^T denotes e_i^T with its first element removed. The bounds of Theorem 2.1 then remain valid when K is replaced by \tilde{K} , A by \tilde{A} , e_i^T by \tilde{e}_i^T and $|\cdot|_{\text{RK}}$ by $|\cdot|_{\text{SDR}}$. Since (A, b) is algebraically accurate the bounds (3.7) correspond to $i = s$ in Table 3.1. \square

4 Stability

In order to investigate error propagation, we first need to establish the basic stability properties of SDR methods. To this end we consider the linear test equation, $\dot{y} = \lambda y$, for which the true one-step Runge–Kutta method yields $y_{n+1} = R(z)y_n$, where $\mathcal{R}_h^0 = R(z)$ is a rational function of $z = h\lambda$. In a similar

way an exact SDR step yields $S_h^0 = S(z)$, where $S(z)$ becomes a 2×2 matrix. For the linear test equation $h\dot{V}_s^*(\eta, h\dot{\eta}) = zV_s^*(\eta, h\dot{\eta})$, and (3.2a) yields $V_s^*(\eta, h\dot{\eta}) = \tilde{e}_s^T(I - z\tilde{A})^{-1}(\tilde{\mathbb{1}}\eta + \tilde{a}h\dot{\eta})$. Hence

$$(4.1) \quad S(z) = \begin{pmatrix} 1 \\ z \end{pmatrix} \tilde{e}_s^T(I - z\tilde{A})^{-1}(\tilde{\mathbb{1}} \quad \tilde{a}).$$

THEOREM 4.1. *$S(z)$ is a rank 1 matrix with eigenvalue $R(z)$ and*

$$(4.2) \quad S^n(z) = R^{n-1}(z)S(z),$$

i.e. $S(z)$ is power-bounded (stable) if and only if $|R(z)| \leq 1$.

Proof. The nonzero eigenvalue of $S(z)$ is obviously $\tilde{e}_s^T(I - z\tilde{A})^{-1}(\tilde{\mathbb{1}} + z\tilde{a}) = 1 + zb^T(I - zA)^{-1}\mathbb{1} = R(z)$. Consequently $S(z)/R(z)$ is an oblique projector, therefore idempotent, and (4.2) follows. \square

Thus the stability of the Runge–Kutta method is preserved in SDR mode. Moreover, a sequence of n exact SDR steps is equivalent to taking one exact SDR step followed by $n - 1$ exact Runge–Kutta steps. The propagation matrix $S(z)$ also has special structure in the nonstiff ($z \rightarrow 0$) and stiff ($z \rightarrow \infty$) limits, respectively. Thus, provided that \tilde{A} is invertible,

$$(4.3) \quad S(0) = \begin{pmatrix} 1 \\ 0 \end{pmatrix} (R(0) \quad b_1), \quad S(\infty) = \begin{pmatrix} 0 \\ 1 \end{pmatrix} (-\tilde{e}_s^T \tilde{A}^{-1} \tilde{\mathbb{1}} \quad R(\infty)).$$

Therefore, $S(0)$ projects errors in the data $(v_n, h\dot{v}_n)$ entirely onto v_{n+1}^* . On the other hand, $S(\infty)$ projects errors in stiff/algebraic components of $(v_n, h\dot{v}_n)$ entirely onto $h\dot{v}_{n+1}^*$ while v_{n+1}^* is left unaffected by previous errors; the behavior of an A -stable SDR method is in this regard similar to that of an L -stable Runge–Kutta method, irrespective of the value of $R(\infty)$.

Variable stepsize computations require special analysis. Let h_n denote the step from t_n to t_{n+1} , define $z_n = h_n\lambda$, and let $\omega_n = h_n/h_{n-1}$ denote the ratio between two consecutive steps. The data available at the start of the step are $(v_n, h_{n-1}\dot{v}_n)$. A single step can then be written

$$(4.4) \quad \begin{pmatrix} v_{n+1} \\ h_n\dot{v}_{n+1} \end{pmatrix} = S(z_n) \begin{pmatrix} v_n \\ h_{n-1}\dot{v}_n \end{pmatrix} = S(z_n) \begin{pmatrix} 1 & 0 \\ 0 & \omega_n \end{pmatrix} \begin{pmatrix} v_n \\ h_{n-1}\dot{v}_n \end{pmatrix}.$$

This implies that the error in the previous stage derivative is amplified by the stepsize ratio ω_n before being propagated by $S(z_n)$. Power boundedness therefore depends on the stepsize rescaling matrix $\Omega_n = \text{diag}(1 \quad \omega_n)$ in (4.4). Multiplying out $S(z_n)\Omega_n S(z_{n-1})$ we note that the range of $S(z_{n-1})$ is spanned by $(1 \quad z_{n-1})^T$, which is transformed into $(1 \quad z_n)^T$ by Ω_n . Since the latter vector is an eigenvector of $S(z_n)$ it follows that $S(z_n)\Omega_n S(z_{n-1}) = R(z_n)\Omega_n S(z_{n-1})$. Repetitive application of this formula yields the variable-step analogue of (4.2),

$$(4.5a) \quad \prod_{j=1}^n S(z_j)\Omega_j \cdot S(z_0) = \prod_{j=1}^n R(z_j)\Omega_j \cdot S(z_0)$$

$$(4.5b) \quad = \begin{pmatrix} 1 & 0 \\ 0 & h_n/h_0 \end{pmatrix} \prod_{j=1}^n R(z_j) \cdot S(z_0).$$

This appears to be a potentially unrestrained propagation of errors; in a typical stiff problem the stepsize may accumulate a growth of many orders of magnitude which is directly reflected by the size of h_n/h_0 . As long as this stepsize ratio is finite, however, the method may take any number of steps, showing that the error amplification is not a matter of instability—stability is again equivalent to that of the Runge–Kutta method via $\prod_j R(z_j)$. As for the error amplification due to variable steps, (4.5b) shows that this only affects the stage derivatives $\{h_{n-1}\dot{v}_n\}$ and not the solution $\{v_n\}$ itself. In order to examine this amplification, while making a fair comparison between the exact Runge–Kutta and SDR methods, we assume that the starting value η is in error by a quantity $\delta\eta$. The true one-step method will during n steps propagate this error by $R(z_j)$ to an error δu_n . If we compute the derivative at the n th step, the derivative will have an error $h\delta\dot{u}_n = z_n\delta u_n$. Therefore

$$(4.6) \quad \begin{pmatrix} \delta u_n \\ h\delta\dot{u}_n \end{pmatrix} = \begin{pmatrix} 1 \\ z_n \end{pmatrix} \delta\eta \prod_0^n R(z_j).$$

For the SDR method, the errors $(\delta\eta, h\delta\dot{\eta})$ in the starting point are correlated. Because the SDR procedure requires a plain Runge–Kutta step to start the process, we consider this first step to be a (Δ, K) -approximate step². As this solution is required to be in the manifold $\mathcal{M}(\eta, h\dot{\eta})$, the errors are related as in (2.11), i.e. $(\delta\eta, h\delta\dot{\eta}) = (\tilde{e}_s^T \delta V^0, \tilde{e}_s^T \tilde{A}^{-1} \delta V^0)$. After n exact SDR steps this error is propagated by (4.5b) into

$$(4.7a) \quad \begin{pmatrix} \delta v_n \\ h\delta\dot{v}_n \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & h_n/h_0 \end{pmatrix} \prod_{j=1}^n R(z_j) \cdot S(z_0) \begin{pmatrix} \tilde{e}_s^T I_{s-1} \\ \tilde{e}_s^T \tilde{A}^{-1} \end{pmatrix} \delta V^0$$

$$(4.7b) \quad = \begin{pmatrix} 1 \\ z_n \end{pmatrix} \tilde{e}_s^T (I - z_0 \tilde{A})^{-1} (\tilde{\mathbb{I}} \tilde{e}_s^T + \tilde{a} \tilde{e}_s^T \tilde{A}^{-1}) \delta V^0 \prod_{j=1}^n R(z_j)$$

$$(4.7c) \quad = \begin{pmatrix} 1 \\ z_n \end{pmatrix} \delta v_1 \prod_1^n R(z_j).$$

This demonstrates that the error propagations of the Runge–Kutta and the corresponding SDR method are very similar; the difference lies only in the first step, where the SDR method has the advantage over the Runge–Kutta method concerning projecting stiff errors entirely onto the derivatives, thus implying smaller errors in the solution. In other respects the two techniques can be expected to yield very similar results, even in the variable step case.

²If the first step were an *exact* Runge–Kutta step, then the following sequence of exact SDR steps is equivalent to taking only exact Runge–Kutta steps.

5 Propagation of iteration errors

The structure of the error propagation carries over also to the full nonlinear case. An exact SDR step yields a solution $(v_{n+1}^*, \dot{v}_{n+1}^*)$ which satisfies the differential equation, i.e. $B\dot{v}_{n+1}^* = f(v_{n+1}^*)$. A subsequent exact SDR step from this point is then identical to an exact Runge–Kutta step, in analogy with (4.2). A Runge–Kutta step from v_{n+1}^* computes the first stage derivative which must satisfy the differential equation at the starting point; this computation merely recreates information already available from the preceding exact SDR step.

By (2.4) the exact Runge–Kutta solution $\{y_n\}$ is given by the recursion

$$(5.1) \quad y_{n+1} = \mathcal{R}_h^0(y_n).$$

The computed approximate SDR solution is a sequence $\{(v_n, h\dot{v}_n)\}$ where

$$(5.2) \quad (v_{n+1}, h\dot{v}_{n+1}) = \mathcal{S}_h^\Delta(v_n, h\dot{v}_n).$$

An exact SDR step from any SDR solution point $(v_n, h\dot{v}_n)$ is a local solution

$$(5.3) \quad (v_{n+1}^*, h\dot{v}_{n+1}^*) = \mathcal{S}_h^0(v_n, h\dot{v}_n).$$

Similarly we define

$$(5.4) \quad (v_n^*, h\dot{v}_n^*) = \mathcal{S}_h^0(v_{n-1}, h\dot{v}_{n-1}); \quad (v_{n+1}^{**}, h\dot{v}_{n+1}^{**}) = \mathcal{S}_h^0(v_n^*, h\dot{v}_n^*),$$

where $v_{n+1}^{**} = \mathcal{R}_h^0(v_n^*)$. With this notation, we decompose the global iteration error $e_n = v_n - y_n$ from the (Δ, \tilde{K}) -approximate SDR steps into two parts,

$$(5.5) \quad e_n = (v_n - v_n^*) + (v_n^* - y_n).$$

The global iteration error $e_{n+1} = v_{n+1} - y_{n+1}$ on the next step can then be decomposed into three parts, cf. Figure 5.1,

$$(5.6) \quad e_{n+1} = (v_{n+1} - v_{n+1}^*) + (v_{n+1}^* - v_{n+1}^{**}) + (v_{n+1}^{**} - y_{n+1}),$$

where the *first* is the error incurred by \mathcal{S}_h^Δ on the present step due to terminating the iteration with a (Δ, \tilde{K}) -approximate solution; $|v_{n+1} - v_{n+1}^*|_y$ is bounded in accordance with (3.7a) of Theorem 3.1. The *second part* is the (Δ, \tilde{K}) -approximation error $v_n - v_n^*$ of the previous step (5.5) propagated by a single exact SDR step \mathcal{S}_h^0 ,

$$(5.7) \quad (v_{n+1}^*, h\dot{v}_{n+1}^*) - (v_{n+1}^{**}, h\dot{v}_{n+1}^{**}) = \mathcal{S}_h^0(v_n, h\dot{v}_n) - \mathcal{S}_h^0(v_n^*, h\dot{v}_n^*),$$

which has the desired property of projecting stiff/algebraic errors onto the derivative; the size of $|v_{n+1}^* - v_{n+1}^{**}|_y$ needs closer examination. Finally the *third part*,

$$(5.8) \quad v_{n+1}^{**} - y_{n+1} = \mathcal{R}_h^0(v_n^*) - \mathcal{R}_h^0(y_n),$$

is the error $v_n^* - y_n$ of (5.5) propagated by the exact Runge–Kutta method in a way which is well studied in the literature in connection with global truncation

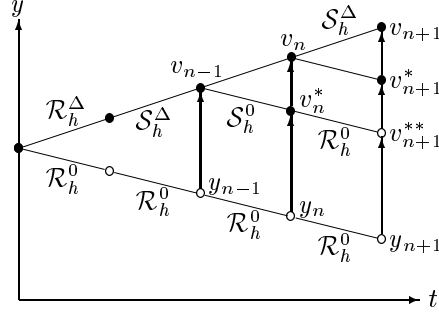


Figure 5.1: Propagation of iteration errors in SDR methods. Points generated by the SDR method (\mathcal{S}_h^Δ or \mathcal{S}_h^0) are indicated by ‘•’ while points generated by the exact RK method (\mathcal{R}_h^0) are indicated by ‘o’. Arrows indicate local and propagated iteration errors. For details, see text.

error propagation; $|v_{n+1}^{**} - y_{n+1}|_y$ can therefore in principle be considered to be a small addition to the usual global error propagation.

We shall estimate $|v_{n+1}^* - v_{n+1}^{**}|_y$. To this end we introduce the data vector for the SDR step, $g(\eta, h\dot{\eta}) = \tilde{\mathbf{l}} \otimes \eta + \tilde{\mathbf{a}} \otimes h\dot{\eta}$. Because it is linear in its arguments, $\delta g = g(\eta + \delta\eta, h\dot{\eta} + h\delta\dot{\eta}) - g(\eta, h\dot{\eta}) = g(\delta\eta, h\delta\dot{\eta})$. Rewriting (3.2), we obtain

$$(5.9a) \quad h\tilde{V} = (\tilde{A}^{-1} \otimes I_d)(\tilde{V} - g(\eta, h\dot{\eta}))$$

$$(5.9b) \quad (I_{s-1} \otimes B)h\tilde{V} = (I_{s-1} \otimes hf)(\tilde{V}).$$

Consequently, $(\tilde{A}^{-1} \otimes B)(\tilde{V} - g(\eta, h\dot{\eta})) = (I_{s-1} \otimes hf)(\tilde{V})$, with formal solution $\tilde{V}(\eta, h\dot{\eta}) = (I_{s-1} \otimes B - \tilde{A} \otimes hf)^{-1}((I_{s-1} \otimes B)g(\eta, h\dot{\eta}))$, which yields

$$(5.10a) \quad v_{n+1}^* = (\tilde{e}_s^T \otimes I_d)(I_{s-1} \otimes B - \tilde{A} \otimes hf)^{-1}((I_{s-1} \otimes B)g(v_n, h\dot{v}_n))$$

$$(5.10b) \quad v_{n+1}^{**} = (\tilde{e}_s^T \otimes I_d)(I_{s-1} \otimes B - \tilde{A} \otimes hf)^{-1}((I_{s-1} \otimes B)g(v_n^*, h\dot{v}_n^*)).$$

We therefore have the bound

$$(5.11) \quad |v_{n+1}^* - v_{n+1}^{**}|_y \leq |\tilde{e}_s^T|_{\text{SDR}} \hat{L} \cdot \|g(\delta v_n, h\delta \dot{v}_n)\|,$$

where $\delta v_n = v_n - v_n^*$ and $h\delta \dot{v}_n = h\dot{v}_n - h\dot{v}_n^*$, and where the Lipschitz constant $\hat{L} = L[(I_{s-1} \otimes B - \tilde{A} \otimes hf)^{-1}(I_{s-1} \otimes B)]$ is defined via $g(\eta, h\dot{\eta}) \mapsto \tilde{V}(\eta, h\dot{\eta})$ as

$$(5.12) \quad \hat{L} = \sup_{\delta g \neq 0} \frac{\|\delta \tilde{V}\|}{\|\delta g\|} = \sup_{\delta \eta, h\delta \dot{\eta} \neq 0} \frac{\|\tilde{V}(\eta + \delta\eta, h\dot{\eta} + h\delta\dot{\eta}) - \tilde{V}(\eta, h\dot{\eta})\|}{\|g(\delta\eta, h\delta\dot{\eta})\|}.$$

In the case of ODEs, when $B = I$, we have

$$(5.13) \quad \hat{L} = L[(I - \tilde{A} \otimes hf)^{-1}] \leq \frac{1}{1 - hM[\tilde{A} \otimes f]} = 1 + O(h),$$

provided that the logarithmic Lipschitz constant (cp. logarithmic norm) satisfies $hM[\tilde{A} \otimes f] < 1$. Since $|\tilde{e}_s^T|_{\text{SDR}} = 1$ the bound (5.11) can then be written

$$(5.14) \quad |v_{n+1}^* - v_{n+1}^{**}|_y \leq \frac{\|g(\delta v_n, h\delta \dot{v}_n)\|}{1 - hM[\tilde{A} \otimes f]},$$

showing that under a suitable monotonicity condition such as $M[\tilde{A} \otimes f] \leq 0$, there holds $|v_{n+1}^* - v_{n+1}^{**}|_y \leq \|\delta g\|$; more generally, the perturbation bound (5.11) is in principle independent of stiffness.

It remains to bound $\|g(\delta v_n, h\delta \dot{v}_n)\|$. A minor extension of Theorem 3.1 yields

$$(5.15a) \quad \|g(\delta v_n, h\delta \dot{v}_n)\| = \|(\tilde{\mathbb{1}} \otimes I_d)(v_n - v_n^*) + (\tilde{a} \otimes I_d)(h\dot{v}_n - h\dot{v}_n^*)\|$$

$$(5.15b) \quad \leq \Delta \cdot |\tilde{\mathbb{1}}\tilde{b}^T \tilde{K}^{-1}|_{\text{SDR}} + \Delta \cdot |\tilde{a}\tilde{e}_s^T \tilde{K}^{-1}|_{\text{SDR}}.$$

Thus, in conclusion we have

$$(5.16a) \quad |v_{n+1} - v_{n+1}^*|_y \leq \Delta \cdot |\tilde{e}_s^T \tilde{K}^{-1}|_{\text{SDR}}$$

$$(5.16b) \quad |v_{n+1}^* - v_{n+1}^{**}|_y \leq \Delta \cdot \hat{L} \cdot (|\tilde{\mathbb{1}}\tilde{b}^T \tilde{K}^{-1}|_{\text{SDR}} + |\tilde{a}\tilde{e}_s^T \tilde{K}^{-1}|_{\text{SDR}}),$$

where the first inequality is the error contribution from the present (Δ, \tilde{K}) -approximate step and the second is the corresponding error from the previous step, propagated by an exact SDR step. We summarize as follows.

THEOREM 5.1. *Let an algebraically accurate Runge–Kutta method with a first explicit stage be applied in SDR mode to (1.1). If each step is (Δ, \tilde{K}) -approximate, then*

(i) *the local iteration error is bounded by (5.16a);*

(ii) *the local iteration error from the previous step is propagated by the exact SDR map \mathcal{S}_h^0 and is bounded by (5.16b), where \hat{L} is defined by (5.12);*

(iii) *the global iteration error is propagated by the exact RK map \mathcal{R}_h^0 .*

We note in particular that $|l_{n+1}^{\text{iter}}|_y \rightarrow 0$ as $\Delta \rightarrow 0$, and that the bound (5.16b) is in principle uniform with respect to stiffness. The effect of the SDR time-stepping is local and comes from the exact SDR step producing $v_{n+1}^* - v_{n+1}^{**}$; this local contribution to the error is essentially devoid of errors in the stiff/algebraic components which are largely suppressed by \mathcal{S}_h^0 , implying that on subsequent steps no such errors will be propagated by the underlying algebraically accurate Runge–Kutta method. This implies that the error transport is identical to that of the exact Runge–Kutta method, only the iteration error increments are different.

To illustrate Theorem 5.1 we consider the trapezoidal rule ($\tilde{a} = \tilde{A} = \tilde{b} = 1/2$) applied to an ODE $\dot{y} = f(y)$. The monotonicity condition $M[f] \leq 0$ implies $\hat{L} = L[(I - (h/2)f)^{-1}] \leq 1$ for all $h > 0$, and for $\tilde{K} = \tilde{A}$ the bounds in (5.16) are both 2Δ . Similarly, for $\tilde{K} = 1$, the bounds are also the same, Δ . This indicates a restrained error propagation by \mathcal{S}_h^0 . The application of the trapezoidal rule in SDR mode is equivalent to using the implicit midpoint rule on a staggered grid while considering the *midpoints* as the solution $\{v_n\}$.

In DIRK methods, it is natural to solve for stage values sequentially, instead of through a simultaneous iteration on all stages, in order to obtain a (Δ, K) -approximate step. The bounded propagation of iteration errors shows that such an approach is feasible; a full analysis for such an iteration scheme is possible and yields results similar to those presented here.

6 SDR in error estimators

The advantage of SDR for methods with a first explicit stage is that the SDR step has a behavior similar to that of an L -stable method. This technique can be put to effective use in the error estimator for RADAU5, cf. [HW96b, p. 123]. As error estimators are only used locally they do not affect global error propagation except indirectly by inducing different stepsize sequences. SDR in error estimators will generally carry less error from the previous step than if the true one-step error estimator were used. In addition it is less expensive by saving one function evaluation per step.

The embedded error estimator for RADAU5 is described by the Butcher tableau

$$\begin{array}{c|cc} 0 & 0 & 0 \\ c & 0 & A \\ \hline y & 0 & b \\ \hat{y} & \hat{b}_0 & \hat{b} \end{array}$$

and has an explicit first stage. However, it is apparently not used in SDR mode and may consequently grossly overestimate stiff/algebraic errors. As a remedy, the estimate is “filtered” [HW96b, p. 123] to quench these errors. An alternative approach achieves a corresponding effect by instead using an error estimator of the form

$$\begin{array}{c|ccc} 0 & 0 & 0 & 0 \\ c & 0 & A & 0 \\ 1 & \hat{b}_0 & \hat{b} - \gamma e_s^T & \gamma \\ \hline y & 0 & b & 0 \\ \hat{y} & \hat{b}_0 & \hat{b} - \gamma e_s^T & \gamma \end{array}$$

where γ is chosen such that the available iteration matrix can be used for the computation of the extra stage and $\hat{b}_0 = 1/50$ is optimized with respect to error magnitude, [dSS97]. Both estimators gain significantly in performance from reusing the stage derivative from the last step. The cost is lower in both cases and the improved estimator also has better properties, supported by analysis. These techniques can easily be employed by designing the embedded error estimators properly for algebraically accurate Runge–Kutta methods.

To give a simplified explanation of why SDR performs better, we consider the embedded reference method of the first method above, which yields

$$\hat{y}_{n+1} = y_n + \hat{b}_0 h f(y_n) + (\hat{b}^T \otimes I) h \dot{Y}$$

in true one-step mode and

$$\hat{y}_{n+1} = y_n + \hat{b}_0 h \dot{y}_n + (\hat{b}^T \otimes I) h \dot{Y}$$

in SDR mode, where \dot{y}_n is the accepted last stage derivative from the previous step. For simplicity, consider the linear test equation $\dot{y} = \lambda y$ with $z = h\lambda$. For the true one-step estimator, we obtain

$$\hat{y}_{n+1} = y_n + \hat{b}_0 z y_n + z \hat{b}^T (I - zA)^{-1} \mathbb{1} y_n,$$

while the SDR estimator results in

$$\hat{y}_{n+1} = y_n + \hat{b}_0 h \dot{y}_n + z \hat{b}^T (I - zA)^{-1} \mathbb{1} y_n.$$

The disadvantage of the true one-step estimator is that if y_n deviates from its exact value, then this deviation is amplified by a factor $\hat{b}_0 z$, which is unbounded as $|z| \rightarrow \infty$. In contrast, if in the SDR estimator $h\dot{y}_n$ is inaccurate, then the latter deviation is only amplified by a constant factor \hat{b}_0 , and for varying step-sizes the influence is proportional to the step-size ratio. The SDR estimator is therefore less affected by the use of an approximate RK process, as Theorem 3.1 provides a stiffness independent bound for the error in $h\dot{y}$.

7 Iteration and truncation errors

A well-known technique for achieving global errors proportional to a given tolerance ε is to control the local error per unit step. While this approach is less efficient for stiff problems, one may as a remedy scale the “unit stepsize” locally and still achieve tolerance proportionality, according to [DB94]. We follow this approach and model truncation errors in terms of a *local time scale* $\tau(t)$. Thus, rather than assuming that the local error asymptotically is $|l(t)|_y \approx c(t)h(t)^{p+1}$, we make the modified asymptotic assumptions

$$(7.1) \quad |l(t)|_y \approx c \cdot (h(t)/\tau(t))^{p+1}; \quad |\hat{l}(t)|_y \approx \hat{c} \cdot (h(t)/\tau(t))^{\hat{p}+1},$$

where the left formula models the local error and the right the local error estimate. Further we shall assume that local extrapolation is used, i.e., $p \geq \hat{p} + 1$, and aim to control local errors per scaled unit step $h/\tau(t)$. We shall investigate how to select the local truncation error bound Δ_{trunc} , which will also tell us how to select the iteration error bound Δ in the approximate RK process.

By keeping the error estimate $|\hat{l}(t)|_y \approx \Delta_{trunc}$, the above asymptotic assumptions yield $\hat{c}(h/\tau)^{\hat{p}+1} \approx \Delta_{trunc}$, and it follows that $h/\tau \approx (\Delta_{trunc}/\hat{c})^{1/(\hat{p}+1)}$. For the local error $l(t)$ we therefore have

$$(7.2) \quad \frac{|l(t)|_y}{h/\tau} \approx c \cdot \left(\frac{\Delta_{trunc}}{\hat{c}} \right)^{p/(\hat{p}+1)}.$$

In order to obtain a globally ε -approximate solution we thus require

$$(7.3) \quad \Delta_{trunc} = \hat{c} \cdot \left(\frac{\varepsilon}{\hat{c}} \right)^{(\hat{p}+1)/p} = \mu_{trunc} \cdot \varepsilon^{(\hat{p}+1)/p}.$$

This indicates how to select the local error tolerance when the orders of the error and error estimator satisfy $p > \hat{p} + 1$. It should be compared to the settings in

Table 7.1: Estimated error coefficients for selected RK methods.

Method	c_{tall}	c_{bushy}	\hat{c}_{tall}	\hat{c}_{bushy}	$\tilde{\mu}_{\text{trunc}}$	$\tilde{\mu}_{\text{iter}}$
esdirk23a	2.6e-2	1.0e-2	7.9e-2	7.9e-2	3	3
esdirk34a	2.7e-2	7.8e-4	5.9e-2	7.5e-3	2	2
esdirk45a	4.9e-4	4.8e-5	2.8e-4	3.9e-5	0.6	5
sd34var	1.0e-3	6.0e-4	1.1e-3	1.4e-2	14	6
RadauIIa($p=5$)	1.4e-4	1.4e-5	3.3e-4	3.3e-4	0.4	6

the code RADAU5, [HW96a], where $p = 5$ and $\hat{p} = 3$, and Δ_{trunc} is chosen proportional to $\varepsilon^{4/6}$ instead of $\varepsilon^{4/5}$ as indicated by the analysis above. This difference is due to the fact that RADAU5 uses an error per step criterion, whereas our analysis is based on an error per unit time scale criterion. As a result, we may expect RADAU5 to allow too large errors for strict tolerances ε , and recent tests (see [dS95]) confirm that an order of magnitude stricter tolerance does not gain an entire magnitude in the achieved global error. The earlier version of RADAU5 used in [HW96b] had $\Delta_{\text{trunc}} = \varepsilon$, and thus decreased the global error more than an order of magnitude given a corresponding tolerance decrease.

In order to set Δ_{trunc} correctly, we need an estimate of μ_{trunc} , which depends on the constants c and \hat{c} , which in turn are problem as well as method dependent. Since the global error will always have an unknown proportionality factor, we may however model these constants in a way which accounts for method dependencies. As the constants depend on certain elementary differentials, we shall compare the coefficients in front of like differentials in the Taylor expansions of the method and its estimator. We choose to study two particular trees, the “tall” tree and the “bushy” tree, which correspond to two extreme cases, linear constant coefficient ODEs and pure quadrature. To this end, we estimate $c_{\text{tall}} = (b^T A^p \mathbf{1} - 1)/(p+1)!$ and $c_{\text{bushy}} = (b^T c^p - 1)/(p+1)!$, and put $c^* = \max(c_{\text{bushy}}, c_{\text{tall}})$, with corresponding definitions of \hat{c}_{tall} , \hat{c}_{bushy} and \hat{c}^* . We finally estimate μ_{trunc} by $\tilde{\mu}_{\text{trunc}} = \hat{c}^*/(c^*)^{(\hat{p}+1)/p}$. Table 7.1 contains approximate values of $\tilde{\mu}_{\text{trunc}}$ for some of the implicit RK methods which have been used to verify the resulting strategy in practical computations. The method coefficients (A, b, \hat{b}) of the ESDIRK-methods are available in [Kvæ92] and those of sd34var can be found in [Ols95]. We remark that the RadauIIa method here uses the scaled error estimator with $\hat{b}_0 = 1/50$, [dSS97].

Turning to the iteration error, we need an accurate solution for two reasons: (i) the error estimate should be accurate and not contaminated by iteration errors, (ii) iteration errors should only make a negligible contribution to the global error. The first item is treated in [NT86] and we will amend these results. In particular, we need to set the value of $\Delta/\Delta_{\text{trunc}}$ properly.

Let the iteration error tolerance be Δ_{iter} . For Runge-Kutta methods with invertible A , a $(\Delta_{\text{iter}}, K)$ -approximate step is then in error by $\Delta_{\text{iter}}|b^T K^{-1}|_{\text{RK}}$ at the end-point of the step. (We will not consider SDR methods here, as the results are completely analogous.) To have a global iteration error less than $\kappa\varepsilon$,

where a suitable fraction is $\kappa \leq 0.1$, we require

$$(7.4) \quad \frac{\Delta_{iter} |b^T K^{-1}|_{\text{RK}}}{h_n / \tau_n} \leq \kappa \varepsilon.$$

Relating this to the truncation error, note that $h/\tau \approx (\varepsilon/c)^{1/p}$. We thus take

$$(7.5) \quad \Delta_{iter} = \mu_{iter} \cdot \frac{\kappa}{|b^T K^{-1}|_{\text{RK}}} \cdot \varepsilon^{(p+1)/p},$$

where $\mu_{iter} = c^{-1/p}$ is estimated by $\tilde{\mu}_{iter}$ for selected RK methods as listed in Table 7.1. We note that Δ_{iter} is proportional to $\varepsilon^{(p+1)/p}$ as opposed to Δ_{trunc} which is proportional to $\varepsilon^{(\hat{p}+1)/p}$. For example, in the three-stage Radau IIa method this requires that Δ_{iter} be proportional to $\varepsilon^{2/5} \Delta_{trunc}$. In the present version of RADAU5, [HW96a] a similar rescaling, but with different exponents, is used, although no theoretical justification is provided. The tests in [HW96b] use an earlier version of RADAU5.

The chosen value for Δ_{iter} needs to be complemented with some safety nets. Thus, we make sure that the iteration error does not give a significant contribution to the actual local error estimate, and we require

$$(7.6) \quad \Delta_{iter} \leq \frac{\kappa_2 |\hat{l}_n|_y}{|(b - \hat{b})^T K^{-1}|_{\text{RK}}},$$

where κ_2 is sufficiently small, say $\kappa_2 = 0.1$. Here $|\hat{l}_n|_y = (h_n/h_{n-1})^{\hat{p}+1} |l_{n-1}|_y$ is an extrapolation of the local error estimate from the previous step, indicating how large an error is expected on the present step while the iteration has not yet been completed. Should this projected value be very small, we allow a threshold value based on Δ_{trunc} , and set

$$(7.7) \quad \Delta_n = \min \left(\Delta_{iter}, \frac{\kappa_2 \max(|\hat{l}_n|_y, \Delta_{trunc}/100)}{|(b - \hat{b})^T K^{-1}|_{\text{RK}}} \right),$$

where Δ_n is the value which is used in actual computation on the n th step.

8 Numerical tests

In this section on testing we will consistently use the strategies from [OS96] for refactorization with predictive stepsize selection taken from [Gus94]. We shall first use an artificial problem to verify the effects of SDR vs. RK as well as assess the effects of reevaluating the right-hand side of an ODE. We shall then test some representative problems from [dS95, dSvdVS95].

The possible disadvantage of reevaluating f will depend on the variations of the Jacobian in the stiff subspace. We therefore need a problem with the following properties: (i) stiffness, i.e., $L_y[hf]$ large, (ii) non-trivial solution in a slowly varying stiff subspace, (iii) significantly varying Jacobian along the solution, independently of stiffness. Note that it is not sufficient that the Jacobian varies

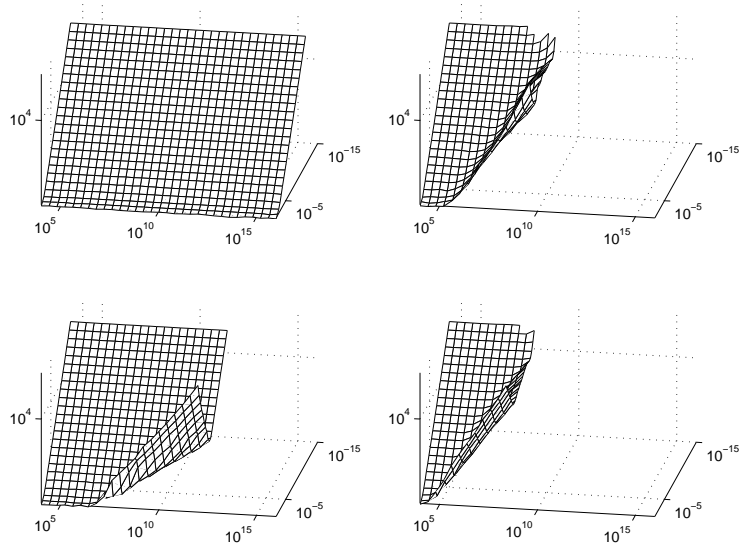


Figure 8.1: Work (number of steps) as a function of tolerance $\varepsilon \in [10^{-2}, 10^{-14}]$, and stiffness parameter $k \in [10^4, 10^{16}]$. The top graphs were computed with an SDR implementation of `esdirk23a`, while the graphs on the bottom were obtained with the same method in true RK mode. The graphs on the right correspond to an implementation where f is reevaluated after the iteration has been terminated. Only the recommended SDR approach (top left) shows a stiffness independent computational efficiency. (Tests requiring more than 10^5 steps are not included.)

considerably in a neighborhood of the solution, cf. Example 2 in [vDS94], but the variations must also occur along the solution.

We consider the quasi-linear problem

$$\frac{d}{dt}(z - \varphi(y(t))) = -\psi(t) \cdot (z - \varphi(y(t))),$$

where we seek $z(t)$. The initial value is taken as $z(0) = \varphi(y(0))$ so that the exact solution is $y(t)$. The equation is rewritten

$$\dot{z}(t) = \varphi'(y(t))\dot{y}(t) - \psi(t) \cdot (z - \varphi(y(t))),$$

where we have chosen $\varphi(y) = y^2$ and

$$\psi(t) = k(2 + \sin(0.4\sqrt{z(t)})); \quad y(t) = 100(1 + 0.8\sin(y(t))),$$

so that the parameter $k > 0$ can be used to control stiffness. The computational results are presented in Figure 8.1.

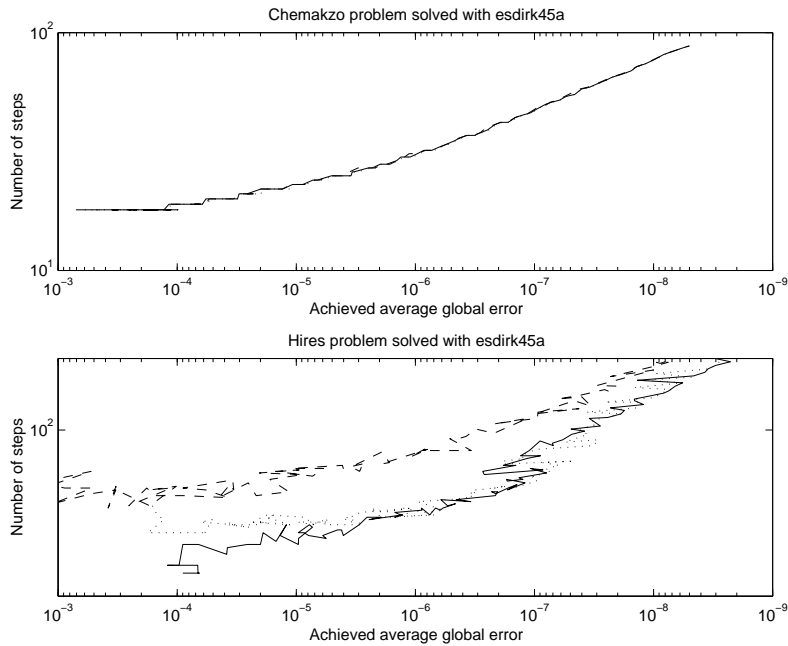


Figure 8.2: Test of reevaluations of f . Solid line: SDR, dotted line: RK, dashed line: RK with reevaluation of f after the iteration has been terminated. Reevaluation of f is inefficient for loose tolerances.

We then considered problems from [dSvdVS95]. The ODEs from this test set can be divided into three groups. Problems where it is harmless to reevaluate f for esdirk45a are: `chemakzo`, `medakzo`; problems where reevaluating f has significant drawbacks but there is little difference between RK and SDR (i.e., $f(Y_1)$ may be reevaluated) are: `pollu`, `hires`, `ringmod`; problems where SDR is essential are: `emep`. Using `RadauIIa` with SDR error estimator, however, yields approximately the same results as with the corresponding true one-step error estimator. We conclude that there are important applications for which SDR is instrumental. Moreover, for problems where true RK works well, SDR shows a similar performance; SDR performance is never inferior to that of true one-step RK methods. We give representative graphs from each of these types in Figures 8.2 and 8.3.

In order to obtain regular graphs the global error is measured in the discretized L^1 norm on the interval of integration by taking the mean global error over all steps, where the error pointwise is measured in an absolute 2-norm. In the `emep` problem we multiply all errors by a constant factor 10^{-10} . The work is measured by the total number of steps. The benefit of not reevaluating f is even more pronounced in terms of the number of function evaluations.

By using new algorithm for the rescaling of the error and the iteration er-

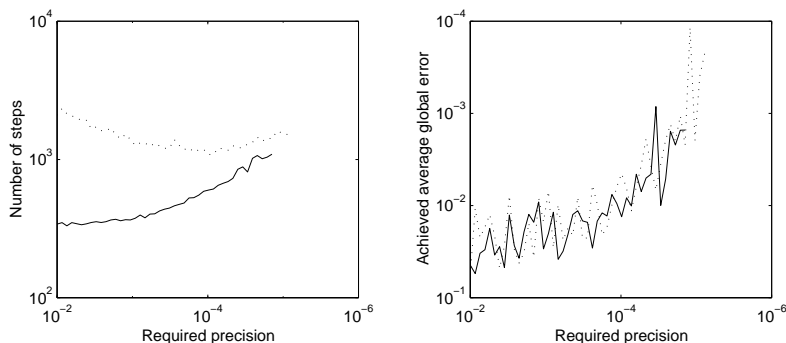


Figure 8.3: SDR vs. RK for `emep` problem solved with `esdirk45a`. Solid line: SDR, dotted line: true RK. SDR requires fewer steps for loose tolerances.

ror tolerance, the 3-stage RadauIIa-method with the third order error estimator from [dSS97] in SDR mode was used to solve representative problems from [dSvdVS95]. The results presented in Figure 8.4 were obtained. The tolerance adapted choice of Δ_{iter} given by (7.5) was compared to various *fixed* ratios for $\Delta_{iter}/\Delta_{trunc}$ to verify that the new strategy is consistently close to the optimum. Although not always optimal, this strategy may often obtain as much as a doubled efficiency, demonstrating that correctly adapted termination criteria have a significant influence on computational performance.

9 Conclusions.

This paper defines the approximate Runge–Kutta computational process by introducing (Δ, K) -approximate solutions to the Runge–Kutta equations. Iteration error bounds are derived in terms of Δ . For methods involving a first explicit stage, we also define and analyze the effects of stage derivative reuse. A tolerance adapted choice of Δ is presented, and the theoretical analysis is verified for several RK methods applied to a number of computational problems.

The goal has been to eliminate some of the heuristics found in most implementations and replace the strategies by techniques which, supported by theoretical analysis, are proven to work well in computational practice. All tests have been carried out *ceteris paribus*, i.e., they are carried out within the same implementation, while only changing the parameters or strategies in question and keeping all other details the same.

Theoretical analysis as well as computational experiments show that the new strategies have a significant influence on computational performance.

REFERENCES

- [AC90] R. K. Alexander and J. J. Coyle. Runge-Kutta methods and differential-algebraic systems. *SIAM Journal on Numerical Anal-*

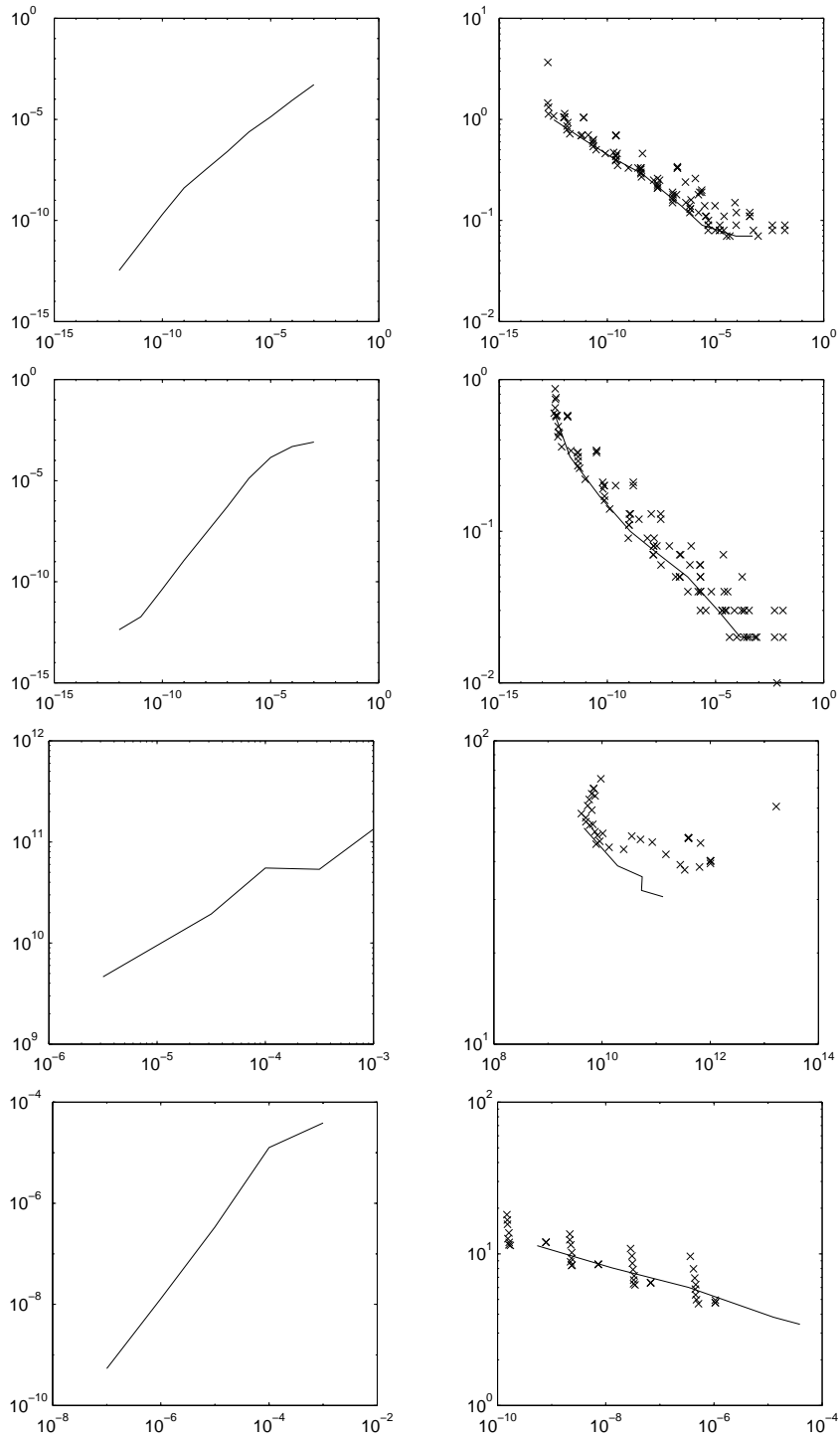


Figure 8.4: Test of error rescaling strategies. Left: global error vs. tolerance plots show a good tolerance proportionality. Right: CPU time vs. global error; tolerance adapted strategy (solid line) is compared to various fixed values of the $\Delta_{iter}/\Delta_{trunc}$ ('x'). The problems are from top to bottom: pollu, chemakzo, emep, ringmod.

- ysis*, 27(3):736–752, June 1990.
- [DB94] G. Dahlquist and Å. Björck. *Numerical Methods*. Prentice Hall, 2nd edition, 1994.
- [dS95] J. J. B. de Swart. Test set for IVP solver. <http://www.cwi.nl/cwi/projects/IVPtestset.html>, 1995.
- [dSS97] J. J. B. de Swart and G. Söderlind. On the construction of error estimators for implicit Runge-Kutta methods. *Journal of Computational and Applied Mathematics*, 86:347–358, 1997.
- [dSvdVS95] J. J. B. de Swart, W. A. van der Veen, and B. P. Sommeijer. Test set for IVPODE and IVPDAE solvers. Technical report, CWI, 1995.
- [Gus94] K. Gustafsson. Control theoretic techniques for stepsize selection in implicit Runge-Kutta methods. *ACM TOMS*, 21(4):496–517, December 1994.
- [HW96a] E. Hairer and G. Wanner. Radau5 code. <ftp://ftp.unige.ch/pub/doc/math/stiff/radau5.f>, July 1996. Accompanying [HW96b].
- [HW96b] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II*. Springer-Verlag, Berlin, 2nd edition, 1996.
- [KS91] J. F. B. M. Kraaijevanger and J. Schneid. On the unique solvability of the Runge-Kutta equations. *Numerische Mathematik*, 59:129–157, 1991.
- [Kvæ92] A. Kværnø. More, and to be hoped, better DIRK methods for the solution of stiff ODEs. Technical report, Mathematical Sciences Div., Norwegian Institute of Technology, Trondheim, 1992.
- [NT86] S. P. Nørsett and P. G. Thomsen. Local error control in SDIRK-methods. *BIT*, 26(1):100–113, 1986.
- [Ols95] H. Olsson. Practical implementation of Runge-Kutta methods for initial value problems. Licentiate Thesis, Lund Institute of Technology, December 1995.
- [OS96] H. Olsson and G. Söderlind. Stage value predictors and efficient Newton iterations in implicit Runge-Kutta methods. Technical Report LU-CS-TR:96-182, Department of Computer Science, Lund University, P.O. Box 118, S-221 00 Lund, Sweden, 1996. Will appear as [?].
- [Sha80] L. F. Shampine. Implementation of implicit formulas for the solution of ODEs. *SIAM Journal on Scientific and Statistical Computing*, 1:103–118, 1980.
- [vDS94] J. L. M. van Dorselaer and M. N. Spijker. The error committed by stopping the Newton iteration in the numerical solution of stiff initial value problems. *IMA Journal of Numerical Analysis*, 14(2):183–209, 1994.