ANZIAM J. 50(2009), 333–342
doi:10.1017/S1446181109000030

PRACTICAL RUNGE–KUTTA METHODS FOR SCIENTIFIC COMPUTATION

J. C. BUTCHER[1]

(Received 9 November, 2007; revised 18 November, 2008)

Abstract

Implicit Runge–Kutta methods have a special role in the numerical solution of stiff problems, such as those found by applying the method of lines to the partial differential equations arising in physical modelling. Of particular interest in this paper are the high-order methods based on Gaussian quadrature and the efficiently implementable singly implicit methods.

2000 *Mathematics subject classification*: primary 65L05.

Keywords and phrases: implicit Runge–Kutta methods, implicit Euler method, stiff problems, A-stability, L-stability, method of lines, Gauss–Legendre quadrature, DIRK methods, SIRK methods, Laguerre polynomials.

1. Introduction

The whole process of solving a scientific problem, including formulation, model construction and numerical approximation, requires an incredible range of skills and knowledge. Very few mathematical scientists can expect to be actively involved in all steps of the process, but Stephen White was such a scientist. Although his primary interests were in applied mathematical modelling, he took a serious interest in the computational algorithms that are now an integral part of the problem-solving process.

Crucial to many applied areas is the numerical solution of ordinary or partial differential equations. For partial differential equations in which diffusion plays a significant role, the method of lines is often used to replace continuous dependence on space variables by a discretized approximation by a high-dimensional system of ordinary differential equations. The initial value problems which result are typically highly stiff, and this is where implicit Runge–Kutta methods have a natural role.

Although Runge–Kutta methods were invented more than 100 years ago, *implicit* Runge–Kutta methods have been known for less than 50 years. Today implicit methods

[1]Department of Mathematics, University of Auckland, 38 Princes St, Science Centre, Building 303, Level 3, Auckland Central; e-mail: butcher@math.auckland.ac.nz.

have become much more important than explicit methods, even though their computational costs are far higher. The reason is that implicit methods are less hampered in their performance by stability restrictions. The aim of this paper is to give some of the flavour of what is possible with implicit Runge–Kutta methods.

Throughout the paper, we will consider the solution of an autonomous N-dimensional initial value problem written in the form

$$y'(t) = f(y(t)), \quad y(t_0) = y_0, \quad f : \mathbb{R}^N \to \mathbb{R}^N.$$

The simplest of all methods for the step-by-step solution of this problem is the (explicit) Euler method. This consists of forming approximations to the solutions at $t_n = t_0 + nh$, $n = 1, 2, \ldots$, where the time-step h is here assumed to be constant, for the sake of simplicity. The approximations are given by

$$y_n = y_{n-1} + hf(y_{n-1}), \quad n = 1, 2, \ldots. \tag{1.1}$$

In contrast to this process, in which each quantity is computed explicitly from known data, we have the *implicit* Euler method. In this method the term $hf(y_{n-1})$ in (1.1) is changed to $hf(y_n)$. We then obtain the sequence of approximations given by

$$y_n = y_{n-1} + hf(y_n). \tag{1.2}$$

The numerical scheme (1.2) requires the solution of a nonlinear algebraic equation and, for practical problems, this is an overwhelming cost. Problems in which this cost is worth paying are known as *stiff problems*.

To set the scene, stiffness is explained using a standard example problem in Section 2. This is followed in Section 3 by a brief introduction to Runge–Kutta methods and, in Section 4, to implicit Runge–Kutta methods in particular. The high-order methods based on Gaussian quadrature are discussed in Section 5 and, in contrast, the lower-order, but more efficient, DIRK and SIRK methods in Sections 6 and 7, respectively.

2. A classical example of a stiff problem

Consider the two-dimensional diffusion equation with Dirichlet boundary conditions on the unit square:

$$\frac{\partial u}{\partial t} = \nabla^2 u, \quad u(x, y) = 0, \text{ on the boundary of } [0, 1] \times [0, 1].$$

The spectrum of the Laplacian is the set

$$\{-\pi^2(m^2 + n^2) : m, n = 1, 2, \ldots\}. \tag{2.1}$$

To solve the problem numerically, discretise with N points in each space direction, and use the standard five-point approximation to the Laplacian.

https://doi.org/10.1017/S1446181109000030 Published online by Cambridge University Press

The discretized problem becomes

$$\frac{dU}{dt} = MU,$$

where M is a specific banded $N^2 \times N^2$ matrix. In replacing an unbounded operator by an approximation represented by a finite-dimensional matrix, we cannot expect a well-conditioned result. To make this comparison quantitative, we will find the eigenvalues of M. The spectrum of M is

$$\sigma(M) = \left\{ -4(N+1)^2 \left(\sin^2\left(\frac{m\pi}{2(N+1)} \right) + \sin^2\left(\frac{n\pi}{2(N+1)} \right) \right) : m, n = 1, 2, \ldots, N \right\}.$$

The members of $\sigma(M)$ run from about $-2\pi^2$ to about $-8(N+1)^2$, compared with the members of the set (2.1) which lie in $\{-\infty, -2\pi^2\}$. For accuracy, $2\pi^2 h$ should be small and, for stability, in the case of the Euler method, $8(N+1)^2 h$ should be small. However, for the implicit Euler method there is no such restriction due to stability.

To solve this problem by the Euler method, we need to compute approximations $y_1, y_2, \ldots, y_{n-1}, y_n, \ldots,$ using the formula $y_n = (I + hM)y_{n-1}$. However, for the implicit Euler method, we need to do the more costly computation $y_n = (I - hM)^{-1} y_{n-1}$. Costly though this is, it is simple compared with what is required for non-linear problems, where Newton iterations have to be carried out.

3. Introduction to Runge–Kutta methods

It will be convenient to consider only autonomous initial value problems

$$y'(t) = f(y(t)), \quad y(t_0) = y_0, \quad f : \mathbb{R}^N \to \mathbb{R}^N.$$

The simple Euler method,

$$y_n = y_{n-1} + hf(y_{n-1}), \quad h = t_n - t_{n-1},$$

can be made more accurate by using either the mid-point or the trapezoidal rule quadrature formula:

$$y_n = y_{n-1} + hf\left(y_{n-1} + \frac{1}{2} hf(y_{n-1}) \right),$$

$$y_n = y_{n-1} + \frac{1}{2} hf(y_{n-1}) + \frac{1}{2} hf(y_{n-1} + hf(y_{n-1})).$$

These methods, from Runge's 1895 paper [9], are "second-order" because the error in a single step behaves like $O(h^3)$. At a specific output point the error is $O(h^2)$. A few years later, Heun [7] gave a full explanation of third-order methods and Kutta [8] gave a detailed analysis of fourth-order methods.

https://doi.org/10.1017/S1446181109000030 Published online by Cambridge University Press

In the early days of Runge–Kutta methods, the aim was to find explicit methods of higher and higher order. However, more recently an important aim has been to find methods suitable for the solution of stiff problems.

In carrying out a step of a Runge–Kutta method, we evaluate s stage values $Y_1, Y_2, \ldots, Y_s$ and s stage derivatives $F_1, F_2, \ldots, F_s$, using the formula $F_i = f(Y_i)$.

Each Y_i is found as a linear combination of the F_j added on to y_0,

$$Y_i = y_0 + h \sum_{j=1}^{s} a_{ij} F_j \approx y(t_0 + c_i h),$$

and the approximation at $t_1 = t_0 + h$ is found from

$$y_1 = y_0 + h \sum_{i=1}^{s} b_i F_i \approx y(t_0 + h).$$

We represent the method by a tableau and introduce the matrix A and the vectors b^T and c:

$$\begin{array}{c|c} c & A \\ \hline & b^T \end{array} = \begin{array}{c|cccc} c_1 & a_{11} & a_{12} & \cdots & a_{1s} \\ c_2 & a_{21} & a_{22} & \cdots & a_{2s} \\ \vdots & \vdots & \vdots & & \vdots \\ c_s & a_{s1} & a_{s2} & \cdots & a_{ss} \\ \hline & b_1 & b_2 & \cdots & b_s \end{array}$$

where $c_i = \sum_{j=1}^{s} a_{ij}$, $i = 1, 2, \ldots, s$. If the method is explicit, so that $c_1 = 0$ and a_{ij} is zero unless $i > j$, a simplified tableau can be used in which the diagonal and upper triangular parts of A are omitted.

In the two examples of methods made famous by Runge [9], the corresponding tableaux are

$$\begin{array}{c|cc} 0 & & \\ \frac{1}{2} & \frac{1}{2} & \\ \hline & 0 & 1 \end{array} \quad \text{and} \quad \begin{array}{c|cc} 0 & & \\ 1 & 1 & \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

Third- and fourth-order methods require three and four stages, respectively. Examples of these can be found in the papers by Heun [7] and Kutta [8] and in [2] and [5].

The pattern that has emerged, that order s can be attained with an s-stage explicit method, is an illusion; order five requires six stages, order six requires seven stages and order seven requires nine stages. It is also known that for $p \geq 8$, $s \geq p + 3$ stages are needed. This bound is tight for $p = 8$ but, above this order, little more is known.

For details of the order conditions and the derivation of particular methods of various orders, see [2].

https://doi.org/10.1017/S1446181109000030 Published online by Cambridge University Press

4. Implicit Runge–Kutta methods

If A is a full matrix, one step of the method consists of the evaluation of $Y_1, Y_2, \ldots, Y_s$ which satisfy

$$Y_i = y_0 + h \sum_{j=1}^{s} a_{ij} f(Y_j).$$

Write this in the form

$$Y = \mathbf{1} \otimes y_0 + h(A \otimes I_N)F,$$

where Y and F are sN-dimensional vectors, for an N-dimensional problem and $\mathbf{1} \in \mathbb{R}^s$ has every component equal to 1.

For N large, this problem is very difficult to solve without excessive cost and usually a simplified form of Newton is used. Let J denote the Jacobian matrix for f computed "recently", so that the Newton corrections

$$Y_i \to Y_i - D_i, \quad i = 1, 2, \ldots, s,$$

satisfy

$$Y_i - D_i = y_0 + h \sum_{j=1}^{s} a_{ij}(F_j - J D_j). \tag{4.1}$$

Write (4.1) in the compact form

$$(I \otimes I_N - hA \otimes J)D = Y - \mathbf{1} \otimes y_0 - h(A \otimes I_N)F \tag{4.2}$$

and assess the cost of the numerical process. This is in two parts: first the costs associated with an occasional recomputation and factorization, and secondly the costs involved in an actual iteration.

The "occasional" costs are the evaluation of J followed by the factorization of the $(sN) \times (sN)$ matrix $I \otimes I_N - hA \otimes J$ at a cost of s^3N^3 multiplied by a small number.

The costs per iteration consist of the evaluation of the s values of f, the evaluation of $Y - \mathbf{1} \otimes y_0 - h(A \otimes I_N)F$, and finally the solution of a pre-factored $(sN) \times (sN)$ linear system (4.2) at a cost of s^2N^2 multiplied by a small constant. The factors N^3 and N^2 seem to be intrinsic requirements in implementing a multistage method. In contrast, the challenge in developing efficient methods is to lower the s^3 and s^2 coefficients.

In addition to the order and the implementation costs, a third vital question about implicit Runge–Kutta methods is their stability behaviour with stiff problems. Associated with each method is a stability function $R(z)$. This is defined in terms of a linear problem $y' = qy$, for which $R(hq)$ is the growth factor. That is $y_n = R(hq)y_{n-1}$. Write $z = hq$ and substitute $hF = zY$ so that $(I - zA)Y = y_0\mathbf{1}$, $y_1 = zb^TY + y_0$. Eliminate Y and we find $R(z) = y_1/y_0 = 1 + zb^T(I - zA)^{-1}\mathbf{1}$.

https://doi.org/10.1017/S1446181109000030 Published online by Cambridge University Press

We want stable behaviour for the exact solution, which corresponds to $\mathrm{Re}(z) \leq 0$, to imply stable behaviour of the computed solution. This means that for z in the left half-plane, $|R(z)| \leq 1$. This property is referred to as A-stability. Some methods have the additional property that $R(\infty) = 0$. A-stable methods, which possess this additional requirement, are said to be L-stable, and for many problems this is a desirable property. For a discussion of the advantages of L-stability over simple A-stability, see [6].

Our task is now to explore various families of implicit methods and ask, for each family, to what extent we can achieve the three desirable properties of high order, good stability, and moderate implementation costs.

5. Methods based on Gaussian quadrature

It is remarkable that associated with each Gaussian quadrature formula on [0, 1] there exists a Runge–Kutta method the same order $2s$ as the quadrature formula itself. These methods are simply constructed. First, the c_i are chosen as the zeros of the polynomial $P_s(2x-1)$ to give orthogonality on [0, 1], where P_s is the Legendre polynomial of degree s. The b_i are then chosen so that the quadrature formula

$$\int_0^1 \phi(x)\,dx \approx \sum_{i=1}^{s} b_i \phi(c_i), \tag{5.1}$$

is exact whenever ϕ is a polynomial of degree not exceeding $s-1$. The orthogonality property then implies that (5.1) actually holds up to degree $2s-1$. By substituting

$$\ell_j(x) = \prod_{j \neq i} \frac{x - x_i}{x_j - x_i}, \qquad j = 1, 2, \ldots, s,$$

as in Lagrange interpolation, we find

$$b_j = \int_0^1 \ell_j(x)\,dx.$$

The elements of A are also related to quadrature formulae, but on intervals $[0, c_i]$. Specifically,

$$a_{ij} = \int_0^{c_i} \ell_j(x)\,dx.$$

Methods defined in this way are always A-stable. Although they are not also L-stable, closely related methods exist with this additional property, such as the Radau IIA methods; see [6]. Even though the order of Gauss methods is $2s$, in practice it is not usually possible to observe rapid convergence of approximations, as $h \to 0$, because the stage order is only s. Methods based on Gaussian quadrature have been discovered to have an important role in the solution of problems in Hamiltonian mechanics [4], but their effectiveness for stiff problems is limited by their high implementation costs.

https://doi.org/10.1017/S1446181109000030 Published online by Cambridge University Press

We present the single example, with $s = 2$:

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

6. Diagonally implicit methods

If A is lower triangular, with constant diagonals, we get the diagonally implicit Runge–Kutta (DIRK) methods of Alexander [1]. The following method illustrates what is possible for DIRK methods:

$$\begin{array}{c|ccc} \lambda & \lambda & & \\ \frac{1}{2}(1+\lambda) & \frac{1}{2}(1-\lambda) & \lambda & \\ 1 & \frac{1}{4}(-6\lambda^2 + 16\lambda - 1) & \frac{1}{4}(6\lambda^2 - 20\lambda + 5) & \lambda \\ \hline & \frac{1}{4}(-6\lambda^2 + 16\lambda - 1) & \frac{1}{4}(6\lambda^2 - 20\lambda + 5) & \lambda \end{array}$$

where $\lambda \approx 0.435\,866\,521\,5$ satisfies $1/6 - (3/2)\lambda + 3\lambda^2 - \lambda^3 = 0$.

This method has order 3 and the stability function is

$$R(z) = \frac{1 + (1 - 3\lambda)z + \left(\frac{1}{2} - 3\lambda + 3\lambda^2\right)z^2}{(1 - \lambda z)^3}.$$

Because the numerator has degree only 2, $R(\infty) = 0$. Because $|R(z)| \le 1$ when $\mathrm{Re}(z) \le 0$, it is A-stable, and therefore also L-stable.

For a general implicit method with $s = 3$, the two components of the cost would be $(27N^3, 9N^2)$. But in this case, because of the special structure they are now only $(N^3, 3N^2)$. This is a major step forward, but low stage order still bedevils us. We attempt to overcome this handicap using singly implicit methods.

7. Singly implicit Runge–Kutta methods

A singly implicit Runge–Kutta (SIRK) method is characterized by the equation $\sigma(A) = \{\lambda\}$. That is, A has a one-point spectrum. While DIRK methods are a special case, they seem to possess advantages that are not possessed by the whole family. That is, for DIRK methods the stages can be computed independently and sequentially from equations of the form $Y_i - h\lambda f(Y_i) =$ a known quantity. Each stage requires the same factorized matrix $I_N - h\lambda J$ to permit solution by a modified Newton iteration process (where $J \approx \partial f/\partial y$).

Our aim is to extend this advantage to SIRK methods in general. The secret lies in the inclusion of a transformation to Jordan canonical form into the computation.

https://doi.org/10.1017/S1446181109000030 Published online by Cambridge University Press

Suppose the matrix T transforms A to canonical form by the formula $T^{-1}AT = \overline{A}$, where

$$\overline{A} = \lambda \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ -1 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & \cdots & -1 & 1 \end{bmatrix}.$$

We will consider a single Newton iteration, simplified by the use of the same approximate Jacobian J for each stage.

Assume that the incoming approximation is y_0, and that we are attempting to evaluate

$$y_1 = y_0 + h(b^T \otimes I_N)F,$$

where we recall that Y and F are made up from the s subvectors Y_i and $F_i = f(Y_i)$, respectively.

The implicit equations to be solved are

$$Y = \mathbf{1} \otimes y_0 + h(A \otimes I_N)F,$$

where we recall that $\mathbf{1}$ is the vector in $\mathbb{R}^s$ with every component equal to 1. The Newton process consists of solving the linear system

$$(I \otimes I_N - hA \otimes J)D = Y - \mathbf{1} \otimes y_0 - h(A \otimes I_N)F,$$

and then updating

$$Y \to Y - D.$$

To benefit from the SIRK property, write

$$\overline{Y} = (T^{-1} \otimes I_N)Y, \quad \overline{F} = (T^{-1} \otimes I_N)F, \quad \overline{D} = (T^{-1} \otimes I_N)D,$$

so that

$$(I \otimes I_N - h\overline{A} \otimes J)\overline{D} = \overline{Y} - \overline{\mathbf{1}} \otimes y_0 - h(\overline{A} \otimes I_N)\overline{F}.$$

If we are doing the back-substitutions using the transformed matrix $\overline{A}$, the cost components are reduced from (s^3N^3, s^2N^2) to (N^3, sN^2), just as for a DIRK method. There are extra costs associated with the transformations, but these consist of a moderate number multiplied by s^2N. For large problems, these N terms are completely swamped by the N^2 and N^3 terms and they can be regarded as a small overhead.

To obtain practical methods in the SIRK family, we will seek methods with high stage order. In fact, we will aim to achieve order at least s together with stage order s. Stage order s means that

$$\sum_{j=1}^{s} a_{ij}\phi(c_i) = \int_0^{c_i} \phi(t)\,dt,$$

https://doi.org/10.1017/S1446181109000030 Published online by Cambridge University Press

for ϕ any polynomial of degree $s-1$. This implies that

$$Ac^{k-1} = \frac{1}{k}c^k, \quad k = 1, 2, \ldots, s,$$

where the vector powers are interpreted component by component. This is equivalent to

$$A^k c^0 = \frac{1}{k!}c^k, \quad k = 1, 2, \ldots, s.$$

From the Cayley–Hamilton theorem,

$$(A - \lambda I)^s c^0 = 0,$$

which can be expanded in the form

$$\sum_{i=0}^{s} \binom{s}{i} (-\lambda)^{s-i} A^i c^0 = 0.$$

Hence, each component of c satisfies

$$\sum_{i=0}^{s} \frac{1}{i!} \binom{s}{i} \left(-\frac{x}{\lambda}\right)^i = 0,$$

so that

$$L_s\left(\frac{x}{\lambda}\right) = 0,$$

where L_s denotes the Laguerre polynomial of degree s.

Let $\xi_1, \xi_2, \ldots, \xi_s$ denote the zeros of L_s, so that

$$c_i = \lambda \xi_i, \quad i = 1, 2, \ldots, s.$$

Before discussing the choice of λ, we remark that it is possible to give an explicit expression for the transformation matrix. It is in fact equal to the generalized Vandermonde matrix

$$T = \begin{bmatrix} L_0(\xi_1) & L_1(\xi_1) & \cdots & L_{s-1}(\xi_1) \\ L_0(\xi_2) & L_1(\xi_2) & \cdots & L_{s-1}(\xi_2) \\ \vdots & \vdots & & \vdots \\ L_0(\xi_s) & L_1(\xi_s) & \cdots & L_{s-1}(\xi_s) \end{bmatrix}.$$

The choice of λ should be made taking stability into account. A convenient option is $\lambda = \xi_k^{-1}$, for some k. This will mean that $c_k = 1$ and that the stability function is zero at infinity. For s as high as eight (with the exception of seven) this leads to an L-stable method.

However, for $s > 2$, k has to be chosen less than s. This means that $c_{k+1}, c_{k+2}, \ldots, c_s$ are all greater than 1. Furthermore, as s increases, the amount

https://doi.org/10.1017/S1446181109000030 Published online by Cambridge University Press

by which some abscissae exceed 1 steadily increases. This has to be reckoned as a severe disadvantage of the method.

Fortunately, this apparent barrier to accurate and stable numerical modelling using singly implicit methods is not insuperable. It is possible to pull the abscissae back into [0, 1], in fact to any set of distinct points that one chooses, without losing any numerical property that really matters; see [3]. The trick is to replace "order" by "effective order". For most applications of effective order, complicated starting, finishing and step-changing schemes have to be added to the method, but in this case there are no significant overheads arising from a variable stepsize implementation.

Assuming that singly implicit methods are implemented as efficiently as possible, using techniques such as those discussed in this section, it must be asked how competitive they are likely to be when faced with demanding and large-scale stiff problems. While tests with a range of problems have shown singly implicit methods to give accurate and reliable results, they need to be compared in rigorous tests against well-known successful codes such as those based on Radau quadrature [6], but there are reasons to expect that they will exhibit their own specific advantages. They have high stage order and no order reduction should be anticipated, but this is at the cost, compared with Radau methods, of lower order per stage. However, the most significant advantage of the new methods is that the singly implicit methods scale up in an excellent manner for high-dimensional problems, simply because a single factorized matrix can be used for all the transformed stages. Furthermore, the cost of the back-substitutions is the same per stage as for the BDF (backward difference formulae) methods and less than for Radau methods. The transformations which seem to be an additional overhead for high-order singly implicit methods, are relatively insignificant for very large problems.

References

[1] R. Alexander, "Diagonally implicit Runge–Kutta methods for stiff ODEs", *SIAM J. Numer. Anal.* **14** (1977) 1006–1021.

[2] J. C. Butcher, *Numerical methods for ordinary differential equations*, 2nd edn (Wiley, Chichester, 2008).

[3] J. C. Butcher and D. J. L. Chen, "ESIRK methods and variable stepsize", *Appl. Numer. Math.* **28** (1998) 193–207.

[4] E. Hairer, C. Lubich and G. Wanner, *Geometric numerical integration. Structure-preserving algorithms for ordinary differential equations* (Springer-Verlag, Berlin, 2002).

[5] E. Hairer, S. P. Nørsett and G. Wanner, *Solving ordinary differential equations I. Nonstiff problems*, Volume 8 of *Springer Series in Comput. Math.* (Springer, Berlin, 1993).

[6] E. Hairer and G. Wanner, *Solving ordinary differential equations II. Stiff and differential-algebraic problems*, Volume 14 of *Springer Series in Comput. Math.* (Springer, Berlin, 1996).

[7] K. Heun, "Neue Methoden zur approximativen Integration der Differentialgleichungen einer unabhängigen veränderlichen", *Z. Math. Phys.* **45** (1900) 23–38.

[8] W. Kutta, "Beitrag zur näherungsweisen Integration totaler Differentialgleichungen", *Z. Math. Phys.* **46** (1901) 435–453.

[9] C. Runge, "Über die numerische Auflösung von Differentialgleichungen", *Math. Ann.* **46** (1895) 167–178.

https://doi.org/10.1017/S1446181109000030 Published online by Cambridge University Press