JOURNAL OF COMPUTATIONAL PHYSICS 35, 48–56 (1980)

Low-Storage Runge–Kutta Schemes

J. H. WILLIAMSON

Department of Computer Science, University of Reading, Whiteknights Park, Reading RG6 2AX, England

Received October 17, 1978; revised April 23, 1979

All second-order, many third-order, and a few fourth-order Runge–Kutta schemes can be arranged to require only two storage locations per variable, compared with three needed by Gill's method.

1. Introduction

There are many criteria to bear in mind when choosing a scheme for integrating systems of differential equations, but the storage requirement becomes crucial whenever the systems are very large. For instance, in the simulation of plasmas, there may 10,000 or even 1,000,000 particles to deal with. Predictor–corrector schemes need the storage of a sizable history of the system, but Runge–Kutta schemes, if appropriately designed, are more attractive. The purpose of this paper is to describe a set of higher-order schemes which require no more storage than is needed by the simple Euler method, namely $2N$ locations, one for each of the N coordinates and N velocities.

The nth-order explicit Runge–Kutta scheme to advance a set of differential equations

$$\dot{x} = f(x)$$

over a step h is

$$\begin{aligned} x(h) &= x(0) + \sum_{j=1}^{n} w_j k_j , \\ k_j &= hf\left(x(0) + \sum_{i=1}^{j-1} \beta_{ji} k_i\right), \\ \alpha_j &= \sum_{i-1}^{j-1} \beta_{ji} , \\ \sum_{j=1}^{n} w_j &= 1. \end{aligned} \tag{1}$$

The vector x represents the N variables, which include the independent variable if f depends explicitly upon it. In order for the integration to be accurate to the appropriate order in h, the coefficients must obey several conditions, but Kopal [9] gives explicit

0021-9991/80/040048–09$02.00/0

formulas for the coefficients in terms of some, typically α_2 and α_3, which can be chosen arbitrarily.

Perhaps the most popular of all Runge–Kutta schemes is the classic fourth order: $\alpha_2 = \alpha_3 = \beta_{32} = \frac{1}{2}$, $\alpha_4 = \beta_{43} = 1$, $\beta_{42} = 0$, $w_1 = w_4 = \frac{1}{6}$, $w_2 = w_3 = \frac{1}{3}$, which needs $4N$ locations of storage. Another commonly used method is Gill's [3]: $\alpha_2 = \alpha_3 = \frac{1}{2}$, $\alpha_4 = 1$, $\beta_{32} = 1 - (\frac{1}{2})^{1/2}$, $\beta_{43} = 1 + (\frac{1}{2})^{1/2}$, $w_1 = w_4 = \frac{1}{6}$, $w_2 = (1 - (\frac{1}{2})^{1/2})/3$, $w_3 = (1 + (\frac{1}{2})^{1/2})/3$. Here only $3N$ locations are required because only certain combinations of the early k_j need be kept, but on the other hand the irrational coefficients are a nuisance. However, Blum [1] has resolved this difficulty by showing that the classic method can itself be cast into a $3N$ storage version, and Fyfe [2] buttoned this up by proving that every fourth-order Runge–Kutta scheme can be packed in this way.

2. The Method

The principle adopted by Gill, Blum, and Fyfe and in this paper is to leave useful information on the register which will receive the contribution $f(x_j)$ instead of starting with an empty register. Thus the algorithm is

$$\begin{aligned} q_j &= a_j q_{j-1} + hf(x_{j-1}) \\ x_j &= x_{j-1} + b_j q_j \end{aligned} \qquad j = 1, n \tag{2}$$

with $a_1 = 0$. Successive values of q_j and x_j overwrite the previous ones so at any stage only $2N$ storage locations q and x are required.

The parameters a_j and b_j can be expressed in terms of the coefficients in (1) only if these coefficients bear certain ratios to each other; this will be discussed in detail for schemes of various orders in Sections 4 to 6. The result is

$$\begin{aligned} b_j &= \beta_{j+1\,j} \qquad (j \neq n), \\ b_n &= w_n, \\ a_j &= \frac{w_{j-1} - b_{j-1}}{w_j} \qquad (j \neq 1, \quad w_j \neq 0), \\ a_j &= \frac{\beta_{j+1\,j-1} - \alpha_j}{b_j} \qquad (j \neq 1, \quad w_j = 0). \end{aligned} \tag{3}$$

No advantage can be gained by trying to generalize (2) by including a term in x_{j-1} in the expression for q_j with a corresponding modification of the expression for x_j. This is because the additional equations determining the new parameters turn out to be linearly dependent on those for a_j and b_j.

3. Round-Off Errors

The algorithm adds contributions arising from the earlier velocities into the position vector x, but does not contaminate the velocity registers with contributions from the positions. This feature can be exploited to reduce the effect of round-off errors [3]. The crucial point is that the changes in x in a single step are usually much smaller than the values of x themselves. Hence the quantities in q are held to greater precision than those in x and so we need only consider the round-off errors arising in the x registers.

The round-off error inevitably introduced at stage j is

$$e_j = (x_j - x_{j-1}) - b_j q_j \,. \tag{4}$$

This is scaled and added into the q register at the end of each stage so that the algorithm becomes

$$q_j = a_j q_{j-1} - e_{j-1}/w_j + hf(x_{j-1}). \tag{5}$$

This ensures that the positions at the end of the complete step x_n are independent of e_j , $j < n$, and are only affected by the current round-off e_n . However, this contribution can itself be canceled during the n stages of the next step by using

$$q_1 = -e_n/w_1 + hf(x_0); \tag{6}$$

hence there will be no accumulation of the dominant (proportional to $| \, x \, |$) round-off error during the calculation. Should any of the w_j be zero, then it is clear from (5), (6) that the corresponding e_j cannot be nullified but will have to be left to accumulate in the usual way.

It should be remarked that a further N storage locations are not required for dealing with the vector e_j , because each element can at once be added into the register q. Thompson [7] showed that the round-off cancellation property was destroyed in a well-known algorithm for Gill's method, and it is important that the scheme described here should not be "simplified" to a form which is algebraically, but not computationally, identical.

We have seen that every round-off error can be expunged not later than the end of the next step. However, during its temporary sojourn in the vector x, it makes slight distortions of the calculated derivatives $f(x_j)$. The resulting errors in the final position x_n depend on the exact form of $f(x)$, but we may obtain a reasonable estimate of the effect by assuming that $\partial f/\partial x$ is constant throughout a single step. The result, for a three-stage method, is

$$\begin{aligned}\delta x = \frac{\partial f}{\partial x} \Big\{ & w_2 e_2 + [w_1 + (w_2 - b_2)]\, e_1 \\ & + \left[w_3 + (w_1 - \alpha_2) + (w_2 - b_2) \left(1 - \frac{\alpha_2}{w_1}\right)\right] e_3 \Big\}.\end{aligned} \tag{7}$$

TABLE I
The Integration Schemes

Case	Order	Coefficients						indirect round-off error	truncation error	Tests			
no.	p	α_2	α_3	β_{32}	w_1	w_2	w_3	ϵ	η	A_8	A_h	B_{32}	Remarks
1	2	$(\frac{1}{2})^{1/2}$			$1-(\frac{1}{2})^{1/2}$	$(\frac{1}{2})^{1/2}$		0.414	0.374	$2E-1$	3	$-6E-5$	Least ϵ
2	2	2/3			1/4	3/4		0.417	0.333	$1E-1$	3	$-5E-5$	Least η
3	2	1/2			0	1		[a]	0.500	$2E-16$		$-1E-5$	Classic 2nd order
4	3	2/3	0	−3/4	7/12	3/4	−1/3	2.304	0.593	$1E0$	3	$-2E-6$	
5	3	1/4	5/12	2/9	1	−3	3	3.964	0.495	$3E-1$	3	$-3E-7$	
6	3	1/4	2/3	8/9	1/4	0	3/4	[a]	0.259	$7E-4$	5	$-1E-8$	
7	3	1/3	3/4	15/16	1/6	3/10	8/15	1.149	0.157	$3E-5$	7	$3E-7$	
8	3	$3/5-6^{1/2}/10$	$3/5+6^{1/2}/15$	$2/3+6^{1/2}/9$	1/6	1/3	1/2	1.138	0.149	$4E-3$	3	$3E-7$	
9	3	0.45736	0.79262	0.92530	0.19546	0.41071	0.39383	0.972	0.124	$6E-2$	3	$6E-7$	Least η
10	3	1/2	3/4	3/4	2/9	1/3	4/9	[b]	0.111	$2E-1$	3	$6E-7$	Ralston 3rd order
11	3	1/2	1	2	1/6	2/3	1/6	[b]	0.250	$-2E0$	3	$1E-6$	Classic 3rd order
12	3	2/3	2/3	3/4	1/4	5/12	1/3	0.635	0.176	$5E-1$	3	$7E-7$	
13	3	$3/5+6^{1/2}/10$	$3/5-6^{1/2}/15$	$2/3-6^{1/2}/9$	1/6	1/3	1/2	0.357	0.255	$1E0$	3	$-2E-9$	
14	3	1	1/3	2/9	0	1/4	3/4	[a]	0.343	$1E0$	3	$-5E-7$	
15	3	1.13541	0.28650	0.15232	−0.16087	0.19721	0.96366	0.231	0.426	$1E0$	3	$-7E-7$	Least ϵ
16	4	1/2	1	0	1/6	2/3	1/6	[b]		$3E-1$	3	$-5E-5$	Degenerate 4th order
17	4	1/2	1	1	1/6	2/3	1/6	0.850		$-8E-16$		$-2E-5$	Degenerate 4th order
18	4	1/2	1	3/2	1/6	2/3	1/6	[b]		$-4E-1$	3	$-1E-5$	Degenerate 4th order
19	4	1	1/2	1/8	1/6	1/6	2/3	0.460		$6E-1$	3	$-2E-5$	Degenerate 4th order
20	4	1/2	1/2	1/2		etc.		[b]	0.101	$5E-2$	5	$-2E-8$	Classic 4th order
21	4	1/2	1/2	$1-(\frac{1}{2})^{1/2}$		etc.		[b]	0.083	$4E-2$	5	$-2E-8$	Gill
22	4	2/5	$7/8-3\,5^{1/2}/16$			etc.		[b]	0.055	$3E-2$	5	$-8E-9$	Ralston 4th order

[a] Complete cancellation of the direct effect of round-off errors is not possible.

[b] Schemes, given for comparison, which cannot be expressed in $2N$ storage form.

The three e_j are independent random variables so δx will be least when ϵ^2, the sum of the squares of the coefficients in (7), is minimized. The quantity ϵ is taken as a figure of merit; it is given in Table I for each of the schemes discussed below.

4. Second-Order Schemes

Second-order schemes can be expressed in terms of a single coefficient α_2 [9], the other ones being $w_1 = 1 - \frac{1}{2}\alpha_2^{-1}$, $w_2 = \frac{1}{2}\alpha_2^{-1}$. All schemes can be accommodated by the low-storage algorithm and the only case which is at all unfavorable is the classic form, $\alpha_2 = \frac{1}{2}$, because there the zero coefficient w_1 means that the round-off errors e_2 cannot be nullified.

5. Third-Order Schemes

The low-storage algorithm can be used if

$$(1 - \alpha_2 - w_3)\, b_2 = (\alpha_3 - \alpha_2)\, w_2 . \tag{8}$$

Using Kopal's expressions for the coefficients this condition becomes, in terms of α_2 and α_3,

$$\alpha_3^2(1 - \alpha_2) + \alpha_3(\alpha_2^2 + \tfrac{1}{2}\alpha_2 - 1) + (\tfrac{1}{3} - \tfrac{1}{2}\alpha_2) = 0. \tag{9}$$

The locus of points in the α_2, α_3 plane are shown in Fig. 1, where the dots represent schemes which seem attractive for various reasons, e.g., rational coefficients, low

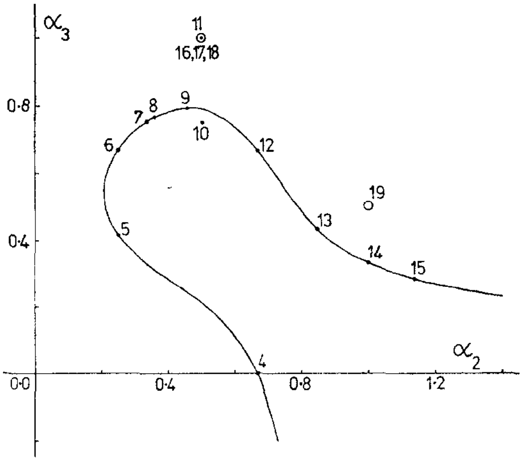

FIG. 1. The curve denotes values of the coefficients α_2, α_3 for which third-order low-storage schemes are possible. The numbered dots show the schemes which are considered in detail in Table I and the circles represent the degenerate fourth-order schemes.

truncation error, low ϵ, which are discussed further. There are in fact two further branches of the curve, which pass close to the points $(2\frac{1}{4}, 4)$ and $(-3, -1\frac{1}{4})$, but none of those schemes have favorable properties.

6. Fourth-Order Schemes

There are no values of α_2 and α_3 which simultaneously satisfy the three conditions necessary for $2N$ storage:

$$\begin{aligned}(w_2 - b_2)\, b_3 &= (\beta_{42} - b_2)\, w_3\,, \\ (1 - w_4 - \alpha_3)\, b_3 &= (1 - \alpha_3)\, w_3\,, \\ (\beta_{41} - \alpha_2)\, b_2 &= (\beta_{31} - \alpha_2)\, \beta_{42}\,.\end{aligned} \tag{10}$$

However, Kopal [9] has described four special schemes in which some coefficients can be allowed to approach zero, while others tend to infinity. A sufficient condition for the utility of these schemes is that $f(x)$ remains bounded as $x \to \infty$. If this is true, then, in the limit, one of the k_j makes no contribution at any further stage of the computation and so it might as well never be calculated in the first place. Thus fourth-order accuracy is achieved by a three-point scheme.

In Table I, these schemes have been written in quasi-third-order form by deleting all the uninteresting zeros and infinities and moving up the remaining coefficients into the appropriate columns. Table II gives the definitions of the limiting processes which led to the schemes. Being three-point schemes, these need only satisfy (8) instead of the impossible conditions (10), and two of the four schemes do indeed satisfy (10) which means that there do exist fourth-order $2N$ storage schemes.

TABLE II

Stability Properties of the Degenerate Fourth-Order Schemes

Case	Definition	$P(y)$	y_{limit}
16	$a_2 = \frac{1}{2}\ \ \alpha_3 \to \frac{3}{4}$	$1 + y + \frac{1}{2}y^2$	−2.000
17	$\alpha_3 = \alpha_2\ \ w_3 \to 0$	$1 + y + \frac{1}{2}y^2 + \frac{1}{12}y^3$	−4.520
18	$\alpha_3 = 0\ \ w_3 \to 0$	$1 + y + \frac{1}{2}y^2 + \frac{1}{8}y^3$	−3.087
19	$\alpha_2 = 1\ \ w_4 \to 0$	$1 + y + \frac{1}{2}y^2 + \frac{1}{12}y^3$	−4.520

7. Stability Analysis

A measure of the maximum allowable step h can be obtained by finding out what the integration scheme makes of the single differential equation

$$\dot{x} = \lambda x$$

as a function of the complex variable λ. It is readily shown [4] that, in all nondegenerate schemes of order $p \leqslant 4$, the stability polynomial (the growth of the single mode over the step h) is

$$P(h\lambda) = \sum_{j=0}^{p} h^j \lambda^j / j!$$

The polynomial is different for the four special cases described in the previous section and the results are given in Table II. Of particular interest is the value of $h\lambda$ when λ is real and as negative as possible: the table shows that three of the four schemes have enhanced stability properties over the conventional schemes for which $h\lambda$ is -2 for $p = 2$, -2.513 for $p = 3$, and -2.785 for $p = 4$.

8. Truncation Errors

Ralston [5] has set up criteria for assessing Runge–Kutta schemes based upon the error bound for the terms of next higher order in h. His results are expressed in the form $| E | < \eta \, ML^p$, where η is a numerical constant and M and L are norms of the function f. Values of η were computed using Ralston's formulas and are given in Table I. It should be noted that bounds cannot be given for the degenerate fourth-order schemes because Ralston's results cannot take cognizance of the fact that $f(x)$ is bounded at large x.

9. Conclusion

The results in Table I show that low-storage schemes are possible up to the fourth order, that nearly all schemes can be immunized against round-off error, and that their truncation error bounds are comparable to those of the more familiar Runge–Kutta methods.

Many authors have pointed out that no one scheme can be superior to all others for all sets of differential equations: by a judicious choice of example any scheme could be made to appear the best. So test A described below was selected to be useful in the application which motivated this research work, namely the simulation of guiding center plasma in two dimensions [6, 8], rather than to highlight any particular scheme.

Test A was to see how the schemes coped with simulating a single pair of particles for which the equation of motion is

$$\dot{r} = ir / | r |^2,$$

where the complex variable r represents the separation between the two particles. The orbit is circular but the frequency varies inversely as the square of the radius. The column "Test A_8" in Table I shows the discrepancy in $|r|^2$ after trying to follow the motion for one period with only eight time steps. The discrepancy falls off according to the power of h which is given in column "A_h." Cases 3 and 17 are particularly good and it can be shown that for any step h whatsoever they yield perfectly circular orbits although with a frequency which is bounded as $|r|^2 \to 0$. (In fact they both give the $m = 1$, $n = 1$ Padé approximant to the complex exponential, something which normally is only to be expected from an implicit Runge–Kutta scheme [4, pp. 131–135].) Case 7 ($\alpha_2 = \frac{1}{3}$, $\alpha_3 = \frac{3}{4}$) also performs this test well, much better, for instance, than any of the familiar third- or fourth-order schemes.

In test B, the result of integrating

$$\dot{x} = (1 - x^2)^{1/2}$$

in 32 steps from 0 to 1 is compared with the known solution $x = \sin t$. Here case 6 performs as well as, and case 13 better than, the standard fourth-order schemes.

If the lowest truncation error bound is desired, then case 9 should be chosen, as its value of η, $\sim\frac{1}{8}$, is almost as good as Ralston's overall minimum η, $\frac{1}{9}$. If the least possible indirect round-off error is considered important, then the choice is case 15 (which has a value of α_2 lying outside the usual range of values 0 to 1). On the other hand if stability is the criterion, then either case 17 or case 19 should be used. Thus no one scheme can be singled out as preferable to the others and it is suggested that users might try out a variety of schemes on the differential equations which they wish to solve. The author's choice is case 7, for which the algorithm is

$$\begin{aligned}
q_1 &= hf(x_0) - 6e_3\,,\\
x_1 &= x_0 + (1/3)\,q_1\,,\\
q_2 &= hf(x_1) - (10/3)\,e_1 - (5/9)\,q_1\,,\\
x_2 &= x_1 + (15/16)\,q_2\,,\\
q_3 &= hf(x_2) - (15/8)\,e_2 - (153/128)\,q_2\,,\\
x_3 &= x_2 + (8/15)\,q_3
\end{aligned} \tag{11}$$

In principle, these high-order low-storage schemes could be used either to increase the accuracy of the integration or to allow a larger time step to be used. For instance, in the author's simulation experiments, case 7 conserves energy 10 times better than the leapfrog code, or else reduces the computational effort to a quarter. However, the time step might well be limited by other considerations, for instance, the cell transit time in particle-mesh codes, so we would then have to settle for improved accuracy.

Acknowledgment

This work was supported by the Science Research Council under Grant GRA 13646.

References

1. E. K. Blum, *Math. Comput.* **16** (1962), 176.
2. D. J. Fyfe, *Math. Comput.* **20** (1966), 392.
3. S. Gill, *Proc. Cambridge Phil. Soc.* **47** (1951), 96.
4. L. Lapidus and J. H. Seinfeld, "Numerical Solution of Ordinary Differential Equations," Academic Press, New York, 1971.
5. A. Ralston, *Math. Comput.* **16** (1962), 431.
6. J. B. Taylor and B. McNamara, *Phys. Fluids* **14** (1971), 1492.
7. R. J. Thompson, *Commun. ACM* **13** (1970), 739.
8. J. H. Williamson, *Phys. Lett. A* **43** (1973), 321.
9. Z. Kopal, "Numerical Analysis," 2nd ed., Chapman & Hall, London, 1961.