Runge-Kutta Research at Toronto ⋆

T.E. Hull, W.H. Enright, K.R. Jackson

Department of Computer Science, University of Toronto, Toronto, Ontario, Canada, M5S 3G4.
{tehull,enright,krj}@cs.utoronto.ca

Abstract

The main purpose of this paper is to review the work on Runge-Kutta methods at the University of Toronto during the period 1963 to the present (1996). To provide some background, brief mention is also made of related work on the numerical solution of ordinary differential equations, but, with just a few exceptions, specific references are given only if the referenced material has a direct bearing on Runge-Kutta methods and their application to a variety of problem areas.

There are several main themes. New Runge-Kutta formulas and new error control strategies are developed, leading for example to continuous methods and their application to areas such as delay, differential-algebraic and boundary-value problems. Software design and implementation are also emphasized. And so is the importance of careful testing and comparing. Other topics, such as the notion of effectiveness, taking advantage of parallelism, and handling discontinuities, are also discussed.

1 A brief pre-history

Interest in Runge-Kutta methods was aroused for Hull at the University of British Columbia after the arrival in 1957 of UBC's first computer, an Alwac III E with an 8K word drum memory but with practically no software beyond a few elementary function routines and a program to play Tic-Tac-Toe. (Early on, a potentially worrisome event took place: the Dean with financial responsibility for the new Computing Centre suffered a humiliating defeat by the program on his first visit to the Centre.)

⋆ This research was supported by the Natural Sciences and Engineering Research Council of Canada, and also more recently by the Information Technology Research Centre of Ontario.

Programs for handling small matrix problems and solving ordinary differential equations were needed. At first it appeared from existing textbooks that, for the latter, one should try to decide between Milne's method, an Adams method, and the classic fourth-order Runge-Kutta method. Other possibilities soon arose however. Considerable inspiration was provided by Dahlquist through his famous 1956 paper [8], and by Peter Henrici whose books [36,37] were in preparation during the late 1950's.

This beginning culminated in several publications, one with Wim Luxemburg of CalTech and the others with students at UBC, namely Al Creemer, Laurie Johnston, Chris Newbery and Ray Zahar. Only the one with Johnston [50] was on Runge-Kutta methods. In this paper a two-parameter family of fourth-order Runge-Kutta formulas was tested to determine which parameter pair minimized the propagated error according to several rough measures of the size of that error. This was to be the first of a series of papers on testing and comparing. The other publications of this period were concerned primarily with existence, convergence, stability, efficiency and accuracy questions associated with linear multi-step methods.

2 The 1963–1971 period

When Hull moved to Toronto in 1963, his interest in ODE solvers broadened considerably. Several of the graduate students in Computer Science during this period worked on ODE solvers, considering such aspects as higher-order formulas, hybrid methods and special methods for restricted classes of problems. In most cases some relatively trivial comparisons were made between different methods.

The students were Jim Ash, Wayne Enright, Gerry Gabel, Wayne Jackson, Sylvia Jones, Dereck Meek, Art Sedgwick and Neil Stewart. Vel Kahan was on the faculty during most of this period and was always ready with helpful advice and encouragement.

In his MSc thesis Ash [1] introduced a hybrid program with control of the error per unit step. It used an Adams formula, but switched to the classic fourth-order Runge-Kutta formula for starting, changing step-size and stopping. His main objective was to have the program choose whichever method was the more efficient at each step of the integration. He also provided for interpolation, if required, and he incorporated a technique for handling jump discontinuities, which had been developed by Stewart [79] as indicated below.

Stewart's MSc thesis [79] introduced a program, with control of error per unit step, based on the same fourth-order Runge-Kutta formula. The error control used the well-known two-steps-one-double-step technique to estimate the local error. His calling sequence was carefully designed and motivated. He also took care to document the program well, mainly by presenting a flow chart beside the program so that it was easy to see the overall structure of the program at a glance. He also made provision for coping with jump discontinuities in the driving function. To accomplish this he allowed the program to creep up close to the discontinuity in the driving function, and then accept the result after taking a very small step over the discontinuity; thus the resulting contribution to the total accumulated error is small, even though the *error per unit step* is quite large over this one step. He studied the behaviour of his program on a variety of problems, and he compared its performance with that of other programs, including one based on a sixth-order Runge-Kutta formula.

Stewart's PhD thesis [80] begins with a discussion of how precise meanings can be given to the terms *problem*, *method* and *effectiveness*. A *problem* is defined by a five-tuple:

$$p = \langle f, x_0, y_0, x_e, a \rangle,$$

where

$$y' = f(x, y), \quad y(x_0) = y_0,$$

x_e is the end point of the integration, and a is an *acceptability criterion*. An example of an acceptability criterion is the following: a sequence of points (x_i, y_i), $i = 0, \ldots, n$, is an *acceptable solution* to the problem p if $x_n = x_e$ and there exists a differentiable interpolant $z(x)$ connecting all the (x_i, y_i) such that

$$z' = f(x, z) + \delta(x),$$

where the *defect* δ satisfies $\|\delta\| \leq \tau$ for a tolerance τ that characterizes the acceptability criterion in this example.

A *method* is simply a procedure for producing an approximate solution. A method is said to be *effective* for a particular problem if it generates an acceptable solution.

Stewart goes on to prove effectiveness theorems. For example, consider the class of problems associated with the linear system of equations $y' = Ay$, where A is a constant matrix, and with an acceptability criterion characterized by a constant τ. Consider a method, based on the classic fourth-order Runge-Kutta formula, which uses the two-steps-one-double-step technique for estimating the local error. Then we have the following:

> *Theorem. If we assume exact arithmetic, and the method restricts itself to a suitably determined maximum step-size, then the method is effective for the given class of problems, provided the result of each step is accepted by the method only when the norm of the error estimate per unit step is $\leq 3\tau/4$.*

The theorem can be modified to take roundoff into account, but the details are then quite messy and tend to obscure the essence of the theorem — which is that, if the method accepts an intermediate result only when the error estimate per unit step is bounded by 3/4 of the tolerance, then there is an interpolant which satisfies the differential equation to within the required tolerance. The notion of effectiveness has had a considerable influence on later developments.

Stewart goes on to consider comparisons. He clarifies notions about how a "best" method from a class of methods relative to a class of problems can be defined, both from an experimental as well as a theoretical point of view. Some of these ideas were introduced earlier by Hull [40], and related results are given in [41] as well as in a paper [42] presented at the IFIP Congress in 1968. (It was at this conference that Bill Gear presented his seminal paper on the use of backward differentiation formulas to solve stiff systems of ODEs.) Much more extensive comparisons were to become a major part of our efforts during the next period.

Stewart's final chapter considered error propagation further, showing a link with control theory, and also suggesting the possibility of automatically calculating an error bound along with an approximate solution. This idea was very much influenced by Kahan [63,64], and in turn led to Wayne Jackson's PhD thesis [62] on calculating ellipsoidal error bounds, rather than the rectangular parallelepipeds produced by interval arithmetic. Stewart extended some of his work further in a later paper [81].

3 The 1971–1983 period

During this period there was considerable interest in studying and experimenting with a variety of techniques for solving ODEs, such as higher-order Runge-Kutta formulas, Taylor series, extrapolation, second derivative methods and stiff solvers. Wayne Enright and Ken Jackson were graduate students early in this period, and later joined the faculty. The other faculty members interested in ODEs were Patrick Keast and Allan Jepson. The other students in ODEs were Cliff Addison, John Butcher (yes, we had one of our own!), Robert Duncan, Mohamed Kamel, Harry Lilleniit, Jim Molnar and Art Sedgwick.

There were some opportunities to apply what we knew to other areas, especially in using stiff solvers for chemical kinetics. One application that used Runge-Kutta formulas was done by our John Butcher [4]; he used them for parameter determination in a modelling experiment, under the supervision of Dave Shanno who was cross-appointed with the Faculty of Management Studies. Duncan [10] also used them to solve integral equations.

There were four main themes to our efforts during this period: *design*, *implementation*, *proving properties*, and *comparisons*. We were greatly helped by numerous visitors to the department, several of whom stayed for lengthy periods: namely Chris Baker, Victor Barwell, Julio Diaz, Graeme Fairweather, Len Freeman, Ian Gladwell, George Hall, Patricia Hanson, Wayne Jackson, Bengt Lindberg, John Pryce, Ron Sacks-Davis, Jack Williams and Ken Wright. Rob Bedet, Imre Farkas and Bryna Fellen assisted with much of the programming.

Hull and Enright proposed an overall *design* of a program in a 1974 technical report [46], and we have adhered quite closely to that design ever since. An outline, with some details omitted, is shown in Figure 1. The calling sequence was very much influenced by numerous discussions held in the "ODE Circus" which was meeting about once a year at the time. (Of course the Circus never did agree on what would be an ideal design! But the discussions were

```
procedure IVPSOLVER (fcn, x, y, xend, τ, scale, ind)
   initialize
   loop
      prepare to calculate (choose new step-size, etc.)
      calculate (determine corresponding values of x and y)
      estimate error
      if estimate acceptable then
         update x, y
         if x = xend then
            set ind accordingly
...............return
         endif
      else
         if it's time to give up then
            set ind accordingly
...............return
         endif
      endif
   endloop
end IVPSOLVER
```

Fig. 1. The outline of an overall design of a program for solving initial-value problems.

very stimulating.) The calling sequence corresponds to the specification of a *problem* mentioned earlier, except that *scale* and *ind* have been added. *Scale* is intended to be a measure of the scale of the problem, which is used by the program to ensure that it does not try to step over intervals in which the solution does not behave smoothly; for example, *scale* could be a Lipschitz constant, or the user could provide *hmax*, a maximum step-size (and *scale* would then be proportional to $1/hmax$). We consider *scale* to be a reliability control: increasing *scale* should make the approximate solution more reliable, just as decreasing τ should make it more accurate. In practice, however *scale* is usually omitted, in which case the corresponding constraint on the step-size is part of the *method*, as was the case in our earlier effectiveness theorem. *Ind* is simply an indicator, used primarily by the program to indicate the reason for returning, but could be used in other ways as well, to indicate the start of the calculations for example. Further aspects of *design* are discussed by Hull [43,44].

We have found that this overall design can be easily adapted to a variety of situations, including those, for example, which require the output of continuous solutions (which will be discussed in the next section). We have also found that it is relatively easy to understand a program, and therefore to modify it if necessary, if the main body of the **loop** construct is organized into four distinct stages (see Figure 1). The first stage involves preparing to calculate. This mostly involves determining the new trial step-size, which depends on several factors such as *scale*, and whether it is the first such step, or a new step after a previously accepted step, and so on. It could also involve choosing the order, if the method is a variable-order method. The second stage is a straightforward use of the formulas to determine the corresponding trial values of x and y. The third stage is to estimate a measure of the error associated with the step, and the fourth stage is to make decisions based on that estimate.

The *implementation* which followed was presented in another technical report [48] by Hull, Enright and Jackson. There the program is based on a Runge-Kutta formula pair of orders five and six provided to us by Jim Verner [83] of Queen's University in Kingston, Ontario. (Ivo Babuška had pointed out to us that Fehlberg's higher-order formula pairs would fail on quadrature problems, since their error estimates would be identically zero on such problems. We asked others for better alternatives, and chose the formula pair provided by Verner.) The program has been in the IMSL library [85] since that time. We originally called our double-precision program DVERK, in Verner's honour, but IMSL added a single-precision version and named the pair IVPRK/DIVPRK.

Considerable effort during this period was devoted to *proving properties* of programs. In his PhD thesis, Sedgwick [73] was the first to prove effectiveness theorems for a broad class of nonlinear problems. He concentrated on a variable-order variable-step Adams method, but his techniques are directly applicable to Runge-Kutta methods. In his MSc thesis, Ken Jackson [51] gave a very thorough discussion of the theoretical justifications and basic ideas for effectiveness theorems, followed by an application to Runge-Kutta programs based on formula pairs such as the pair used in DVERK. Roundoff and overflow/underflow are taken into consideration. In a 1978 paper by Jackson, Enright and Hull [56] a theoretical cost criterion is developed for comparing Runge-Kutta formulas. Based on this criterion, along with experimental comparisons, it was concluded that DVERK was as efficient as any other Runge-Kutta method available then, but it was conjectured that some higher-order formula pairs would eventually be found which are still more efficient. Further aspects regarding the proof of properties of programs are also discussed by Hull, Enright and Sedgwick [49] and Hull [45].

The third main theme during this period was the careful and extensive *comparisons* of different ODE solvers. The results of our first major effort in this area appeared in a 1972 paper by Hull, Enright, Fellen and Sedgwick [47]. A test program called DETEST, developed by Hall, Enright, Hull and Sedgwick [34], was used to collect statistics on how well each ODE solver performed. The original twenty-five test problems were selected from the literature and organized into problem classes based on the dimensions and sources of the differential equations. To ensure that the methods being compared provided consistent numerical results, we found it necessary to modify the error and stepsize selection strategies of several of the methods. Cost was determined by recording the total time, overhead time, and the number of function evaluations for each problem. Reliability was determined by recording the percent of steps taken for which the true local error per unit step exceeded the tolerance, and also the maximum observed ratio of the local error per unit step to the tolerance. To compare what we considered to be the key capabilities of different methods, we deliberately tried to suppress the effects of special features of the methods such as their ability to choose initial step-sizes, to handle discontinuities, or to cope with any significant stiffness in the differential equations.

DETEST was used in a later paper by Enright and Hull [23] where the results for twenty different solvers were presented. It became clear that an especially good method for general purposes had to be a variable-order method, mainly because such a method is able to adapt to different error tolerances, using low orders for modest tolerances but higher orders for more stringent tolerances. Two variable-order Adams methods developed by Krogh and by Sedgwick gave the best results. However, if the function evaluations are relatively inexpensive, extrapolation methods, such as those developed originally by Bulirsch and Stoer [3], are usually more efficient; Runge-Kutta methods also behave well in such situations, but their good behaviour in each case is restricted to somewhat narrower ranges of tolerance, the more stringent tolerances requiring high-order methods.

We were criticized, especially by Fred Krogh and Larry Shampine, for modifying existing methods to make them conform to our requirement that they control error per unit step, rather than error per step. We believe that our defense in [23] was a reasonable one, but DETEST was eventually revised by Enright and Pryce in 1983 [29], and this new version did not require any modification of the programs being assessed. A revision of the earlier version of a test program called STIFF DETEST for stiff solvers is also included in [29]. In these new versions the emphasis was shifted to judging solvers based on how efficiently and effectively they can maintain a requirement that the global error be kept proportional to the tolerance. Various options were provided for the user, particularly in allowing the user to request certain normalized statistics for making direct comparisons between different methods. Additional problem classes, including one with discontinuous driving functions, were included and the user was given the option of solving each problem in its original form or in a more appropriately scaled form. The entire package is a better example of good "software engineering" than the original DETEST, and of course it is available to anyone who wishes to use it. A more readily available description of the package did not appear until 1987 [30]. The package was to be used heavily during the next period.

4 The 1983–early 1996 period

A new theme, which has led to several new initiatives during this period, is the development of *continuous methods*. A major new *implementation* project is also being undertaken during this period, along with expanded testing facilities for *assessment and comparison* of new methods. Another theme is the development of *parallel* methods. Much of the work in this period is surveyed in a 1994 paper by Enright, Higham, Owren and Sharp [21].

Primary responsibility for the work on ODEs during this period was with Enright and Jackson. Tom Fairgrieve and Philip Sharp were also major contributors. Allan Jepson had an interest in the methods being developed because of their usefulness in solving problems that arise in his work on bifurcation problems. The students working on ODEs were Peter Balint, Vince Canale, Robert Enenkel, Tom Fairgrieve, Hiroshi Hayashi, Wayne Hayes, Martin MacDonell, Ned Nedialkov, Hung Nguyen, Richard Pancer, Jeff Peterson, Brian Robertson, Wendy Seward, Heru Suhartanto and Khalid Zuberi. Paul Muir was also a student and a frequent visitor later on. The other visitors working in this area were Kevin Burrage, Robert Chan, Jerry Fine, Des Higham, Min Hu, Mohamed Kamel, Ivar Lie, Paul Nørsett, Bryn Owren, Alastair Spence and Per Thomsen. Each of John Butcher (the elder), Ian Gladwell and Larry

Shampine paid us several short visits during this period, as well as during the previous one.

A major shortcoming of higher-order Runge-Kutta methods in use in the early 1980's was that they were inefficient for applications (such as graphics) that require frequent output and for problems with discontinuities. Several authors had proposed partial remedies for these deficiencies, but none were completely satisfactory. Enright, Jackson, Nørsett and Thomsen [24] developed a "bootstrapping" technique to associate with any s-stage, pth-order Runge-Kutta formula a "continuous extension" requiring $\bar{s} \geq s$ stages. This scheme yields an approximation to the local solution that differs from the true local solution by at most $O(h^p)$ or $O(h^{p+1})$. A method based on such a technique provides the user with a sequence of points (x_i, y_i), $i = 0, \ldots, n$, where $x_n = x_e$ in the specification of the problem, along with the coefficients for a sequence of piecewise interpolating polynomials between each adjacent pair of points. The general approach is developed in [24] and specific examples of methods of orders four, five and six are given there. Upper bounds on $\bar{s} - s$ were also obtained for both $O(h^p)$ and $O(h^{p+1})$ extensions. Classes of other continuous Runge-Kutta formulas were later investigated by Fine [33], Owren and Zennaro [69,70], Muir and Owren [66], and Verner [84].

Using such continuous extensions, an efficient scheme for detecting and stepping over discontinuities was developed and analyzed by Enright, Jackson, Nørsett and Thomsen [25]. New step-size and error control strategies have also been introduced and analyzed by Enright [15,16] and Higham [39]; they are based on the idea of trying to ensure that the defect associated with the $O(h^{p+1})$ continuous extension satisfies

$$\max_{x_{n-1} \leq x \leq x_n} \|\delta(x)\| \leq \tau$$

on each step. Effectiveness theory has been extended to this class of methods and continuous Runge-Kutta formulas of orders four to eight have been analyzed from this point of view by Enright [17]. Continuous extensions of Runge-Kutta formulas have also been applied to singular initial-value problems by Enright and Suhartanto [31] and Suhartanto [82], to boundary-value problems by Enright and Muir [28], to delay differential problems by Enright and Hayashi [18,19], Enright and Hu [22] and Hayashi [35], and to special classes of algebraic differential equations by Enright and MacDonald [26], MacDonald [65] and Nguyen [68].

An *implementation* of an explicit Runge-Kutta code for initial-value ODE problems, known as TERK [32], based on these new results has been under development for some time, and is now nearing completion. The work on TERK also motivated research on global error estimates by Peterson [71] and Higham [38] and on stiffness detection by Robertson [72]; it is planned to incorporate these capabilities in a future version of TERK. Some special Runge-Kutta formulas have been developed and implemented for boundary-value problems by Muir [67] and Enright and Muir [27]. Still others have been developed and implemented for first-order problems by Sharp [75], and second-order problems by Sharp [74] and Sharp and Fine [77,78] (but these last four do not involve continuous extensions).

Investigations related to the *assessment and comparison* of numerical methods have continued. Sharp [76] has developed new formats for viewing and reporting the results of comparisons and he has introduced an option into DETEST to report statistics in this format. For the comparison of different continuous methods, an extended version of DETEST, called X-DETEST, has been written which reports other measures of error such as the maximum defect and the maximum global error of the continuous approximation. DDE-TEST, another test program that measures performance based on maximum global error and the size of the defect, is being developed for the assessment of methods for delay differential equations. It uses a collection of twenty-one test problems, and it stores the "true solution" as piecewise polynomials, obtained using high-precision arithmetic.

Several aspects of *parallelism* have been considered during this period. The potential for parallelism in standard Runge-Kutta methods has been studied by Jackson and Nørsett [59]: both negative and positive results are presented. Many of the negative results are based on a theorem that bounds the order of a Runge-Kutta formula in terms of the minimal polynomial associated with its coefficient matrix. The positive results are largely examples of prototypical formulas which offer a potential for effective "coarse-grain" parallelism on machines with a few processors.

Runge-Kutta predictor-corrector methods have been discussed by Enenkel [11], and analyzed by Jackson, Kværnø and Nørsett [57,58]. The results in the last two papers are of value not only for the development of parallel ODE codes, but also for the efficient, reliable and robust implementation of implicit Runge-Kutta methods on standard sequential machines.

Some of the material discussed in the last two paragraphs is summarized in a survey paper by Jackson [53] on parallel methods for initial-value problems. In addition, the papers outlined above motivated three others by de Sterke, Jackson and Robert [9], Broderick, de Sterke and Jackson [2] and Jackson [54] in which Runge-Kutta predictor-corrector methods are applied to solve hyperbolic partial differential equations arising in optics. In addition, Enenkel's MSc research [11] has been extended by Enenkel and Jackson [13,14] and by Enenkel [12]. They derive a class of parallel general linear methods and provide numerical results that show the promise of this approach, although much more work must be done before an effective parallel implementation can be developed.

Parallelism has also been used by Enright and Higham [20] to improve the reliability of continuous Runge-Kutta methods. Jackson and Pancer [60] investigate the parallel solution of the "almost block diagonal" systems of linear equations that arise in the numerical solution of boundary-value problems by Runge-Kutta methods and many other schemes.

Some related work on solving large systems of ODEs has been carried out by Chan, Jackson and Zhu [7], Chan and Jackson [5,6], Jackson [52] and Jackson and Seward [61], and is summarized in a survey paper by Jackson [55]. Although the results in these papers have been applied mainly to codes based on linear multi-step methods, the techniques apply equally well to routines based on Runge-Kutta formulas.

5 Acknowledgements

We are very grateful to John Butcher, Tom Fairgrieve, Joe Flaherty, Ian Gladwell, Des Higham, Bryn Owren, Philip Sharp and Neil Stewart for helpful comments they made on an earlier draft of this paper.

References

[1] J.H. Ash. An Adams Runge-Kutta subroutine for systems of ordinary differential equations. Master's thesis, University of Toronto, 1965.

[2] N.G.R. Broderick, C.M. de Sterke, and K.R. Jackson. Coupled mode equations with free carrier effects: a numerical solution. *J. Optical and Quantum Electronics*, 26:S219–S234, 1994.

[3] R. Bulirsch and J. Stoer. Numerical treatment of ordinary differential equations by extrapolation methods. *Numer. Math.*, 8:1–13, 1966.

[4] J.W. Butcher. Numerical analysis in the construction of mathematical models. Master's thesis, University of Toronto, 1972.

[5] T.F. Chan and K.R. Jackson. Nonlinearly preconditioned Krylov subspace methods for discrete-Newton algorithms. *SIAM J. Sci. Stat. Comput.*, 5:533–542, 1984.

[6] T.F. Chan and K.R. Jackson. The use of iterative linear-equation solvers in codes for large systems of stiff IVPs for ODEs. *SIAM J. Sci. Stat. Comput.*, 7:378–417, 1986.

[7] T.F. Chan, K.R. Jackson, and B. Zhu. Alternating-direction incomplete factorizations. *SIAM J. Numer. Anal.*, 20:239–257, 1983.

[8] G. Dahlquist. Convergence and stability in the numerical integration of ordinary differential equations. *Math. Scand.*, 4:33–53, 1956.

[9] C.M. de Sterke, K.R. Jackson, and B.D. Robert. Nonlinear coupled mode equations on a finite interval: a numerical procedure. *J. of the Optical Society of America B*, 8:403–412, 1991. Special issue on *Numerical Simulation in Guided-Wave Optics and Electroptics*.

[10] R.P. Duncan. A Runge-Kutta method using variable stepsizes for Volterra integral equations of the 2nd kind. Master's thesis, University of Toronto, 1982.

[11] R.F. Enenkel. Implementation of parallel predictor-corrector Runge-Kutta methods. Master's thesis, University of Toronto, 1988.

[12] R.F. Enenkel. *DIMSEMs — Diagonally Implicit Single-Eigenvalue Methods for the Numerical Solution of Stiff ODEs on Parallel Computers.* PhD thesis, University of Toronto, 1996. In preparation.

[13] R.F. Enenkel and K.R. Jackson. DIMSEMs — diagonally implicit single-eigenvalue methods for the numerical solution of stiff ODEs on parallel computers: A preliminary report, 1995. 138 pages.

[14] R.F. Enenkel and K.R. Jackson. DIMSEMs — diagonally implicit single-eigenvalue methods for the numerical solution of stiff ODEs on parallel computers, 1996. To be submitted to *Advances in Computational Mathematics.* 31 pages.

[15] W.H. Enright. Analysis of error control strategies for continuous Runge-Kutta methods. *SIAM J. Numer. Anal.*, 26:588–599, 1989.

[16] W.H. Enright. A new error-control for initial value solvers. *Applied Mathematics and Computation*, 31:288–301, 1989.

[17] W.H. Enright. The relative efficiency of alternative defect control schemes for high order Runge-Kutta formulas. *SIAM J. Numer. Anal.*, 30:1419–1445, 1993.

[18] W.H. Enright and H. Hayashi. Convergence analysis for the numerical solution of retarded and neutral delay differential equations by continuous numerical methods, 1996. In preparation.

[19] W.H. Enright and H. Hayashi. A delay differential equation solver based on a continuous Runge-Kutta method, 1996. Submitted to *SIAM J. Sci. Comput.* 15 pages.

[20] W.H. Enright and D.J. Higham. Parallel defect control. *BIT*, 31:647–663, 1991.

[21] W.H. Enright, D.J. Higham, B. Owren, and P.W. Sharp. A survey of the explicit Runge-Kutta method. Technical Report No. 291, Department of Computer Science, University of Toronto, 1994.

[22] W.H. Enright and M. Hu. Interpolating Runge-Kutta methods for vanishing delay differential equations. *Computing*, 55:223–236, 1995.

[23] W.H. Enright and T.E. Hull. Test results on initial value methods for non-stiff ordinary differential equations. *SIAM J. Numer. Anal.*, 13:944–961, 1976.

[24] W.H. Enright, K.R. Jackson, S.P. Nørsett, and P.G. Thomsen. Interpolants for Runge-Kutta formulas. *ACM Trans. Math. Software*, 12:193–218, 1986.

[25] W.H. Enright, K.R. Jackson, S.P. Nørsett, and P.G. Thomsen. Effective solution of discontinuous IVPs using a Runge-Kutta formula pair with interpolants. *Applied Mathematics and Computation*, 27:313–335, 1988.

[26] W.H. Enright and C. MacDonald. The practical implications of order reduction for implicit Runge-Kutta methods. *Numerical Algorithms*, 2:351–370, 1992.

[27] W.H. Enright and P. Muir. Efficient classes of Runge-Kutta methods for two point boundary value problems. *Computing*, 37:315–334, 1986.

[28] W.H. Enright and P. Muir. A Runge-Kutta type boundary value ODE solver with defect control. *SIAM J. Sci. Comput.*, 17:479–497, 1996.

[29] W.H. Enright and J.D. Pryce. Two Fortran packages for assessing initial value methods. Technical Report No. 167, Department of Computer Science, University of Toronto, 1983.

[30] W.H. Enright and J.D. Pryce. Two FORTRAN packages for assessing initial value methods. *ACM Trans. Math. Software*, 13:1–27, 1987.

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

[32] T.F. Fairgrieve, W.H. Enright, and K.R. Jackson. A user's guide to TERK, 1996. In preparation.

[33] J.M. Fine. Interpolants for Runge-Kutta-Nyström methods. *Computing*, 39:27–42, 1987.

[34] G. Hall, W.H. Enright, T.E. Hull, and A.E. Sedgwick. DETEST: A program for comparing numerical methods for ordinary differential equations. Technical Report No. 60, Department of Computer Science, University of Toronto, 1973.

[35] H. Hayashi. *Numerical Solution of Retarded and Neutral Delay Differential Equations using Continuous Runge-Kutta Methods.* PhD thesis, University of Toronto, 1996.

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

[37] P. Henrici. *Error Propagation for Difference Methods.* Wiley, New York, 1963.

[38] D.J. Higham. Global error versus tolerance for explicit Runge-Kutta methods. *IMA J. Numer. Anal.*, 11:457–480, 1991.

[39] D.J. Higham. Runge-Kutta defect control using Hermite-Birkoff interpolation. *SIAM J. Sci. Stat. Comput.*, 12:991–999, 1991.

[40] T.E. Hull. A search for optimum methods for the numerical integration of ordinary differential equations. *SIAM Review*, 9:647–654, 1967.

[41] T.E. Hull. The effectiveness of numerical methods for ordinary differential equations. *SIAM Studies in Numer. Anal.*, 2:114–121, 1968.

[42] T.E. Hull. The numerical integration of ordinary differential equations. In *IFIP Congress 68 Proceedings*, pages 39–53. North-Holland, Amsterdam, 1969. Invited paper.

[43] T.E. Hull. The development of software for solving ordinary differential equations. In *Proceedings of the Conference on the Numerical Solution of Differential Equations*, pages 55–63, Dundee, Scotland, 1974. Springer-Verlag. Invited paper.

[44] T.E. Hull. Numerical solutions of initial value problems for ordinary differential equations. In A.K. Aziz, editor, *Proceedings of the NSF Symposium on Numerical Solutions of Boundary Problems for Ordinary Differential Equations*, pages 3–26, Baltimore, Maryland, 1975. Academic Press. Invited paper.

[45] T.E. Hull. Correctness of numerical software. In L.D. Fosdick, editor, *Performance of Evaluation of Numerical Software*, pages 3–15. North-Holland, 1979. Invited paper.

[46] T.E. Hull and W.H. Enright. A structure for programs that solve ordinary differential equations. Technical Report No. 66, Department of Computer Science, University of Toronto, 1974.

[47] T.E. Hull, W.H. Enright, B.M. Fellen, and A.E. Sedgwick. Comparing numerical methods for ordinary differential equations. *SIAM J. Numer. Anal.*, 9:603–637, 1972. See also Errata: *SIAM J. Numer. Anal.*, 11:681, 1974.

[48] T.E. Hull, W.H. Enright, and K.R. Jackson. User's guide for DVERK — a subroutine for solving non-stiff ODEs. Technical Report No. 100, Department of Computer Science, University of Toronto, 1976.

[49] T.E. Hull, W.H. Enright, and A.E. Sedgwick. The correctness of numerical algorithms. In *Proceedings of the SIGPLAN Symposium on Proofs of Assertions about Programs*, pages 66–73, Las Cruces, New Mexico, 1972.

[50] T.E. Hull and R.L. Johnston. Optimum Runge-Kutta methods. *Math. Comp.*, 18:306–310, 1964.

[51] K.R. Jackson. Proving properties about subroutines that solve ODEs numerically. Master's thesis, University of Toronto, 1974.

[52] K.R. Jackson. The use of Krylov subspace methods in codes for large stiff systems of ODEs. In R. Vichnevetsky and R.S. Stepleman, editors, *Advances in Computer Methods for Partial Differential Equations — VI: Proceedings of the Sixth IMACS International Symposium on Computer Methods for Partial Differential Equations*, pages 452–459, Lehigh University, Bethlehem PA, 1987.

[53] K.R. Jackson. A survey of parallel numerical methods for initial value problems for ordinary differential equations. *IEEE Transactions on Magnetics*, 27:3792–3797, 1991.

[54] K.R. Jackson. An application of Runge-Kutta predictor-corrector methods to two system of hyperbolic equations arising in optics. In W.F. Ames, editor, *Fourteenth IMACS World Congress*, pages 1329–1331, Atlanta GA, 1994.

[55] K.R. Jackson. The numerical solution of large systems of stiff IVPs for ODEs. *Applied Numerical Mathematics*, 20:1–16, 1996.

[56] K.R. Jackson, W.H. Enright, and T.E. Hull. A theoretical criterion for comparing Runge-Kutta formulas. *SIAM J. Numer. Anal.*, 15:618–641, 1978.

[57] K.R. Jackson, A. Kværnø, and S.P. Nørsett. An analysis of the order of Runge-Kutta methods that use an iterative scheme to compute their internal stage values. To appear in *BIT*.

[58] K.R. Jackson, A. Kværnø, and S.P. Nørsett. The use of Butcher series in the analysis of Newton-like iterations in Runge-Kutta formulas. *Applied Numerical Mathematics*, 15:341–356, 1994.

[59] K.R. Jackson and S.P. Nørsett. The potential for parallelism in Runge-Kutta methods. Part 1: RK formulas in standard form. *SIAM J. Numer. Anal.*, 32:49–82, 1995.

[60] K.R. Jackson and R.N. Pancer. The parallel solution of ABD systems arising in numerical methods for BVPs for ODEs. Technical Report No. 255, Department of Computer Science, University of Toronto, 1992.

[61] K.R. Jackson and W.L. Seward. Adaptive linear equation solvers in codes for large stiff systems of ODEs. *SIAM J. Sci. Comput.*, 14:800–823, 1993.

[62] L.W. Jackson. *Automatic Error Analysis for the Solution of Ordinary Differential Equations*. PhD thesis, University of Toronto, 1971.

[63] W.M. Kahan. A computable error-bound for systems of ordinary differential equations. Abstract in *SIAM Review*, 8:568–569, 1966.

[64] W.M. Kahan. An ellipsoidal error-bound for linear systems of ordinary differential equations. Unpublished manuscript, 1968.

[65] C. MacDonald. The practical implications of order reduction on the solution of stiff initial value problems using Runge-Kutta formulas. Master's thesis, University of Toronto, 1990.

[66] P. Muir and B. Owren. Order barriers and characterizations of continuous mono-implicit Runge-Kutta schemes. *Math. Comp.*, 61:675–699, 1993.

[67] P.H. Muir. *Implicit Runge-Kutta Methods for Two-point Boundary Value Problems*. PhD thesis, University of Toronto, 1984.

[68] H. Nguyen. *Interpolation and Error Control Schemes for Algebraic Differential Equations Using Continuous Implicit Runge-Kutta Methods.* PhD thesis, University of Toronto, 1995.

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

[70] B. Owren and M. Zennaro. Derivation of efficient continuous explicit Runge-Kutta methods. *SIAM J. Sci. Stat. Comput.*, 13:1488–1501, 1992.

[71] P.J. Peterson. Global error estimates using defect correction techniques for explicit Runge-Kutta methods. Master's thesis, University of Toronto, 1986.

[72] B.C. Robertson. Detecting stiffness with explicit Runge-Kutta formulas. Master's thesis, University of Toronto, 1986.

[73] A.E. Sedgwick. *An Effective Variable Order Variable Step Adams Method.* PhD thesis, University of Toronto, 1973.

[74] P.W. Sharp. ERNY — an explicit Runge-Kutta-Nyström integrator for second order initial value problems. Technical Report No. 199, Department of Computer Science, University of Toronto, 1987.

[75] P.W. Sharp. New low order explicit Runge-Kutta pairs. Technical Report No. 222, Department of Computer Science, University of Toronto, 1989.

[76] P.W. Sharp. Numerical comparison of explicit Runge-Kutta pairs of orders four through eight. *ACM Trans. Math. Software*, 17:387–409, 1991.

[77] P.W. Sharp and J.M. Fine. User guide for ERNY — an explicit Runge-Kutta-Nyström integrator for second order initial value problems. Technical Report No. 198, Department of Computer Science, University of Toronto, 1987.

[78] P.W. Sharp and J.M. Fine. Some Nyström pairs for the general second order initial value problem. *J. Comp. and Appl. Math.*, 42:279–291, 1992.

[79] N.F. Stewart. An integration subroutine using a Runge-Kutta method. Master's thesis, University of Toronto, 1965.

[80] N.F. Stewart. *The Comparison of Numerical Methods for Ordinary Differential Equations.* PhD thesis, University of Toronto, 1968.

[81] N.F. Stewart. Certain equivalent requirements of approximate solutions of $x' = f(t, x)$. *SIAM J. Numer. Anal.*, 7:256–270, 1970.

[82] H. Suhartanto. A new approach for detecting singular points in the numerical solution of initial value problems. Master's thesis, University of Toronto, 1990.

[83] J.H. Verner, 1976. Private communication.

[84] J.H. Verner. Differentiable interpolants for higher-order Runge-Kutta methods. *SIAM J. Numer. Anal.*, 39:1446–1466, 1993.

[85] Visual Numerics, Inc., Houston, Texas. *IMSL MATH/LIBRARY User's Manual: FORTRAN subroutines for mathematical applications.* Volume 2, 645–652, 1994.