The Potential for Parallelism in Runge-Kutta Methods.
Part 1: RK Formulas in Standard Form.

K. R. Jackson and S. P. Nørsett

Technical Report No. 239/90

November 1990.

Computer Science Department
University of Toronto
Toronto, Ontario, Canada M5S 1A4

Prof. K. R. Jackson, Computer Science Dept., University of Toronto, Toronto, Ontario, Canada M5S 1A4. E-mail: krj@na.toronto.edu

Prof. S. P. Nørsett, Division of Mathematical Sciences, Norwegian Institute of Technology, N–7034 Trondheim–NTH, Norway. E-mail: norsett@imf.unit.no

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada, the Information Technology Research Centre of Ontario, and the Norwegian Research Council for Science and Humanities, D.02.08.001.

The Potential for Parallelism in Runge-Kutta Methods. Part 1: RK Formulas in Standard Form.

K. R. Jackson and S. P. Nørsett

Abstract. We examine the potential for parallelism in Runge-Kutta (RK) methods based on formulas in standard one-step form. Both negative and positive results are presented. Many of the negative results are based on a theorem that bounds the order of a RK formula in terms of the minimum polynomial for 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.

1. Introduction. It is widely believed that the only feasible means of solving many important computationally intensive problems in science and engineering is to use parallel computers effectively. As a result, increasing numbers of researchers have begun investigating numerical methods for a wide variety of advanced machine architectures. In many application areas, though, this research is still in its infancy, as is the case, in particular, for the numerical solution of Initial Value Problems (IVPs) for Ordinary Differential Equations (ODEs).

Gear [**18**] [**19**] surveys the preliminary research in this area and discusses the need for parallel computation. He also outlines several open problems in this new field. In a more recent report, Burrage [**5**] provides a useful survey of parallel methods for nonstiff IVPs.

Gear [**18**] [**19**] classifies the means of achieving parallelism in IVP solvers into two main categories:

(1) *parallelism across the method* or equivalently *parallelism across time* and
(2) *parallelism across the system* or equivalently *parallelism across space*.

Included in class (1) are algorithms that exploit several concurrent function evaluations within a step, as well as techniques that solve for many steps simultaneously, as does the fast parallel linear recurrence solution algorithm described in [**20**, §3]. Class (2) includes *waveform relaxation* and *modular integration*, two currently active areas of investigation, as well as more obvious techniques such as exploiting parallelism in the evaluation of the function f associated with the IVP.

Most of the work to date on the parallel solution of IVPs can be considered preliminary research, in that it concentrates on developing potentially useful numerical schemes, rather than their effective implementation, comparisons of methods, or the develop of reliable, robust and (hopefully) portable mathematical software. This can be justified on the grounds that a wide variety of potentially useful approaches should be explored and a large collection of methods developed before tackling the problem of selecting the most promising schemes and working on their effective implementation on a broad class of parallel machines.

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada, the Information Technology Research Centre of Ontario, and the Norwegian Research Council for Science and Humanities, D.02.08.001.

This paper too can be classified as preliminary research on the parallel solution of IVPs, in the sense described above. We explore the potential for parallelism across Runge-Kutta (RK) methods, limiting our consideration to the exploitation of concurrent function evaluations within each step of a method based on a standard *one-step* RK formula and the associated parallel linear algebra in the case of fully-implicit RK methods. In a forthcoming paper [**31**], we explore the potential for parallelism in methods based on RK *predictor-corrector* formulas. We refer to both groups of parallel RK schemes as *PaRK* methods.

In this paper and its companion [**31**], we lay the ground-work for further study of PaRK methods. We do not address in any depth many important questions such as the efficient implementation of particular PaRK schemes on specific parallel computers. Rather, we explore general techniques applicable to a broad spectrum of parallel machines. Also, we concentrate on the well-studied class of RK formulas rather than the wider class of *General Linear Methods*, although much of our discussion can be extended to this wider class as well.

Although several of our results are negative, being of the form that a RK formula having certain desirable characteristics for parallel computation cannot exist, we do exhibit some examples of prototypical formulas having considerable promise for effective implementation on parallel machines. From theoretical considerations and some preliminary numerical results, it appears that for either stiff or nonstiff IVPs there exist s-stage PaRK formulas that may yield a speedup of almost s on a wide range of parallel computers. Since s is typically in the range 4 to 8 for nonstiff IVPs, and in the range 2 to 6 for stiff IVPs, such a speedup, although not overly dramatic, is significant and well-worth pursuing. Furthermore, several sources of parallelism can be exploited simultaneously in one IVP code. Therefore, instead of using s processors only for concurrent function evaluations, these techniques can be extended in the obvious way to make use of s groups of processors, with the processors within each group exploiting alternate sources of parallelism — for example, within the function evaluations themselves.

An outline of this paper follows. We introduce in §2 the notation and definitions used throughout the paper. Also, we prove a theorem that bounds the order of a RK formula in terms of the minimum polynomial of its coefficient matrix. This result is used in later sections to bound the order of various PaRK schemes.

In the remaining sections, we consider the exploitation of concurrent function evaluations in codes based on standard one-step RK formulas. In particular, we show in §3 that there is limited potential for parallelism in codes based on standard explicit RK formulas, a result noted by several other authors. Examples of some minor improvements that can be achieved through the exploitation of parallelism are discussed.

For *strictly-diagonal* implicit RK formulas — that is, RK formulas having nonzeros on the diagonal only of the coefficient matrix — the results presented in §4 are mixed. If the function $f(x, y)$ associated with the IVP is linear in both x and y, then a s-stage method of order $s + 1$ with real coefficients, or order $2s$ with complex coefficients, can be obtained for which all function evaluations can be performed simultaneously. In contrast, we prove the negative result that, if f is nonlinear, then the maximum order of a s-stage strictly-diagonal RK formula is 2, independent of s. We end §4 with a brief discussion of an extension of these results to *strictly-block-diagonal* RK formulas — that is, RK formulas having nonzeros in diagonal blocks only of the coefficient matrix. In particular, if 2×2

diagonal blocks are permitted, then our result for linear problems can be extended to show the existence of s-stage formulas having real coefficients only that attain order $2s$ for the restricted class of linear problems. We conclude with a brief summary of additional results of Lie [**38**] and Iserles and Nørsett [**29**] for strictly-block-diagonal RK formulas.

We begin §5 with a discussion of the structure of *diagonally-implicit* RK formulas that permits the exploitation of parallel function evaluations. The central result of this section is a bound on the order of these formulas in terms of the potential for parallelism inherent in the structure of the associated coefficient matrix and the multiplicity of the distinct diagonal coefficients of the formula. As examples, we present an $\mathrm{A}_\mathbf{0}$-stable 4-stage $4^{\mathbf{th}}$-order formula that requires two pairs of simultaneous function evaluations as well as a similar A-stable formula of Iserles and Nørsett [**29**]. We summarize some promising numerical results of Lie [**38**] for a fixed-stepsize implementation of formulas of this type obtained on a Cray XMP/2 using "macrotasking". The formulas of van der Houwen, Sommeijer and Couzy [**26**] also fall into this class. Finally, we briefly consider the extension of the results in this section to formulas having block-lower-triangular coefficient matrices.

We consider in §6 *fully-implicit* RK methods. The iterative methods used to solve the associated nonlinear equations for the internal stages allow for completely parallel function evaluations. For stiff equations, a Newton-like iteration is typically used to solve for the internal stages. The challenge in this case is to exploit parallelism in the solution of the associated linear systems. To this end, we review Butcher's strategy [**8**] of using similarity transformations to implement fully-implicit RK methods. A desirable characteristic for a formula implemented in this fashion is that its coefficient matrix has a few distinct eigenvalues only — preferably one. In addition, for parallel computation, it is advantageous that the associated Jordan blocks be small. We show that satisfying these two requirements places a severe restriction on the order of the formula. This leads us to consider s-stage formulas having coefficient matrices with s distinct real eigenvalues. Consequently, these matrices can be diagonalized. As others have observed, [**1**], [**33**], [**34**], [**36**], the latter property can be used to great advantage for parallel computation, since, not only can all s function evaluations required to evaluate the formula be computed concurrently, but also the system of linear equations associated with the Newton iteration for the solution of the *internal stage values* can be transformed into s decoupled equally-sized subsystems, all of which can be solved in parallel. We review results of Nørsett and Wanner [**43**], Bales, Karakashian and Serbin [**1**] and Keeling [**36**] that allow one to develop $\mathrm{A}_\mathbf{0}$-stable implicit RK collocation formulas with diagonalizable coefficient matrices that can be implemented effectively as outlined above on a wide range of parallel computers. We also review some exciting new results of Orel [**45**] on high-order L-stable methods with distinct real eigenvalues. Finally, we summarize some promising numerical results of Karakashian and Rust [**34**] for a fixed-stepsize implementation of a 2-stage $3^{\mathbf{rd}}$-order $\mathrm{A}_\mathbf{0}$-stable fully-implicit diagonalizable PaRK formula.

In a forthcoming paper [**31**], we use the well-established principle of converting a direct method to an iterative one to exploit the greater potential for parallelism in the latter formulation. This leads to the class of *predictor-corrector* (PC) PaRK methods, also studied by van der Houwen, Sommeijer and Couzy [**23**] [**25**] [**26**]. We prove that, after the $i^{\mathbf{th}}$ correction, the order of the approximation is $\min(\nu_\mathbf{0} + i + \delta, \nu_\mathbf{1})$, where $\nu_\mathbf{0}$ is the order of the predictor, $\nu_\mathbf{1}$ is the order of the underlying RK corrector formula, and δ is either

0 or 1 depending on the PC formulation. This result is somewhat surprising in the sense that it permits the predicted internal stage values to approximate the actual values to a much lower order than ν_0, as is often the case. If Newton's method is used in place of the simple PC iteration, then it appears that the order of the approximation is doubled on each iteration in most cases. We also discuss a *pipelined* variant of the PC PaRK methods that increases the potential for parallelism attainable in this approach. We again prove a result about the order of these methods and discuss the associated Newton variant. We give some preliminary numerical results for the four classes of methods discussed above.

Our preliminary results have been extended by Enenkel [**14**], Lie [**38**], Iserles and Nørsett [**29**], and Kalvenes [**32**]. As noted above, their results are summarized in the appropriate sections of this paper and [**31**]. Also, some of the results of these papers are summarized in [**42**].

Several other researchers have investigated the potential for concurrent function evaluations within each step of an otherwise standard *forward-step* method. In addition to the papers already cited, see [**2**], [**6**], [**12**], [**17**], [**24**], [**35**], [**39**], [**40**], [**49**] [**50**], [**53**], most of which consider predictor-corrector implementations of block methods similar to those discussed in [**31**] for RK methods. Finally, Karakashian [**33**], Karakashian and Rust [**34**] and Keeling [**36**] discuss implicit RK collocation formulas of the type presented in §6.

2. Notation and an Order Bound. In this section, we establish the definitions and notation used throughout this paper. Also, we prove a theorem that is used in several subsequent sections to bound the order of a PaRK formula in terms of the minimum polynomial of its coefficient matrix.

Consider the IVP

$$
\begin{aligned}
y'(x) &= f(x, y(x)) \quad \text{for} \quad x \in [x_s, x_e], \\
y(x_s) &= y_s,
\end{aligned}
\tag{2.1}
$$

where $y : \mathbb{R} \to \mathbb{R}^m$ and $f : \mathbb{R} \times \mathbb{R}^m \to \mathbb{R}^m$. A RK formula for the numerical solution of (2.1) is given by

$$
\begin{aligned}
Y_{n,i} &= y_n + h_n \sum_{j=1}^{s} a_{ij} F_{n,j}, \qquad \text{for } i = 1, \ldots, s, \\
y_{n+1} &= y_n + h_n \sum_{j=1}^{s} b_j F_{n,j}
\end{aligned}
\tag{2.2}
$$

which may be written more compactly using *tensor product* notation as

$$
\begin{aligned}
Y_n &= e \otimes y_n + h_n A \otimes F_n, \\
y_{n+1} &= y_n + h_n b^T \otimes F_n,
\end{aligned}
\tag{2.3}
$$

where $x_s = x_0 < x_1 < \cdots < x_N = x_e$ are the *gridpoints* of the *discretization*; $h_n = x_{n+1} - x_n$ is the *stepsize* at step n; s is the number of *stages* of the formula; $A = [a_{ij}] \in \mathbb{R}^{s \times s}$ is the *coefficient* matrix of the formula, while $b = (b_i) \in \mathbb{R}^s$ is its vector of *weights* and $c = (c_i) \in \mathbb{R}^s$ is its vector of *nodes*; for any vector v and matrix X, v^T and X^T, respectively, are their transposes; $Y_{n,i} \approx y(x_n + c_i h_n) \in \mathbb{R}^m$ for $i = 1, \ldots, s$ are the s *internal stage values* of the formula at step n and $F_{n,i} = f(x_n + c_i h_n, Y_{n,i}) \in \mathbb{R}^m$ for $i = 1, \ldots, s$ are the associated function values for the step; $y_n \approx y(x_n)$ are the *numerical-solution-values*; $e = (1, \ldots, 1)^T \in \mathbb{R}^s$; $Y_n = (Y_{n,i})_{i=1}^s \in \mathbb{R}^{sm}$; and $F_n = F(x_n, Y_n; h_n) = (F_{n,i})_{i=1}^s \in \mathbb{R}^{sm}$. The *tensor-product* of any two matrices $X = [x_{ij}] \in \mathbb{R}^{M_1 \times N_1}$ and $Z \in \mathbb{R}^{M_2 \times N_2}$ is

$$X \otimes Z = \begin{pmatrix} x_{11} Z & \ldots & x_{1N_1} Z \\ \vdots & \ldots & \vdots \\ x_{M_1 1} Z & \ldots & x_{M_1 N_1} Z \end{pmatrix} \in \mathbb{R}^{M_1 M_2 \times N_1 N_2}.$$

Also, we employ the frequently used "abuse of notation" $A \otimes F_n$ and $b^T \otimes F_n$ to stand for $(A \otimes I_m) F_n$ and $(b^T \otimes I_m) F_n$, respectively, where I_m is the identity matrix in $\mathbb{R}^{m \times m}$. The coefficients of a RK formula are often exhibited in tableau form as shown in Figure 2.1.

$$\begin{array}{c|ccc} c_1 & a_{11} & \ldots & a_{1s} \\ \vdots & \vdots & \ldots & \vdots \\ c_s & a_{s1} & \ldots & a_{ss} \\ \hline & b_1 & \ldots & b_s \end{array}$$

Figure 2.1. A RK Coefficient Tableau.

Formula (2.2) is an *Explicit* RK (ERK) formula iff $a_{ij} = 0$ for $i \leq j$ (assuming that the $\{Y_{n,i}\}$ are suitably ordered). Consequently, the $\{Y_{n,i}\}$ can be computed recursively without need to solve any implicit equations. A RK formula which is not explicit is an *Implicit* RK (IRK) formula. The *Diagonally-Implicit* RK (DIRK) formulas for which $a_{ij} = 0$ for $i < j$ (assuming again that the $\{Y_{n,i}\}$ are suitably ordered) are an important subclass of IRK formulas which, as is explained more fully in §5, enjoy the advantage that the implicit equations for the $\{Y_{n,i}\}$ of a DIRK formula can be solved one at a time: that is, they can be decoupled. We refer to those IRK formulas that are not DIRK schemes as *Fully-Implicit* RK (FIRK) formulas. Finally, if the coefficient matrix A of an IRK or DIRK formula has one distinct eigenvalue only, then it is a *Singly-Implicit* RK (SIRK) formula or a *Singly-Diagonally-Implicit* RK (SDIRK) formula, respectively.

The RK formula (2.2) is of order ν iff ν is the largest integer such that, for all *sufficiently smooth* functions f, the *local error* satisfies $y_{n+1} - y_n(x_{n+1}) = O(h_n^{\nu+1})$ as $h_n \to 0$, where $y_n(x)$ is the solution to the *local IVP*

$$y_n'(x) = f(x, y_n(x)), \qquad y_n(x_n) = y_n, \tag{2.4}$$

which satisfies the same differential equation as the original problem (2.1) but passes through the numerical solution y_n at x_n.

The *stability function* $R(z)$ of a RK formula is the rational function that satisfies $y_{n+1} = R(h\lambda) y_n$, where y_{n+1} is the approximation generated when the RK formula is applied to the simple test problem $y' = \lambda y$, $\lambda \in \mathbb{C}$, starting from y_n with stepsize h. As is well-known [**13**, §3.4],

$$R(z) = 1 + zb^T(I - zA)^{-1}e = \frac{\det(I - zA + zeb^T)}{\det(I - zA)}, \qquad z = h\lambda. \tag{2.5}$$

Thus, for a s-stage RK formula, $R(z) = P(z)/Q(z)$ for polynomials P and Q satisfying $\deg(P) \leq s$ and $Q(z) = \prod_{i=1}^{s}(1 - \gamma_i z)$ where $\{\gamma_i\}$ are the eigenvalues of A, since $\det(I - zA) = \det(I - zT^{-1}AT) = \det(I - zB)$ for the *Jordan Normal Form* B of A. Moreover, if the coefficients b and A of the RK formula are real, as we assume throughout most of this paper, then P and Q are both real polynomials. Furthermore, if all the eigenvalues $\{\gamma_i\}$ of A are zero, the $Q(z) = 1$ for all z and $R(z)$ reduces to the polynomial $P(z)$, which we refer to as the *stability polynomial* of the formula in this case.

A rational approximation $R(z)$ to e^z is said to be of order $\hat{\nu}$ iff $\hat{\nu}$ is the largest integer for which $R(z) = e^z + O(z^{\hat{\nu}+1})$ as $z \to 0$. Since the simple test equation $y' = \lambda y$ is a special case of the ODE in the general problem (2.1), the order $\hat{\nu}$ of the stability function of a RK formula always satisfies $\hat{\nu} \geq \nu$, where ν is the order of the RK formula.

As defined in [**16**], $z_n(x)$ is an interpolant of the numerical solution having *local error* of order ν on the interval $[x_n, x_{n+1}]$ iff $z_n(x_n) = y_n$, $z_n(x_{n+1}) = y_{n+1}$, and ν is the largest integer for which

$$\max_{x \in [x_n, x_{n+1}]} \|y_n(x) - z_n(x)\| = O(h_n^{\nu}) \qquad \text{as } h_n \to 0,$$

where $y_n(x)$ is the solution of the local problem (2.4).

Formula (2.2) is a *s-stage p-parallel q-processor* RK formula iff p is the smallest integer for which the s internal stage values $\{Y_{n,i}\}$ can be evaluated in p *time units* and q is the smallest number of processors for which this value of p can be attained. For an ERK formula, each *time unit* is equal to the time required for a function evaluation of the form $f(x + c_i h_n, Y)$ plus "a little" overhead, while, for an IRK formula, each *time unit* is equal to the time required to solve an equation of the form $Y = C_n + h_n \gamma f(x + c_i h_n, Y)$ plus "a little" overhead, where C_n and γ are constants that depend on previously computed values and the RK formula, respectively.

Some of our colleagues have found the terminology "*p-parallel q-processor* RK formula" misleading or confusing. We originally adopted this phrase as being short for "the RK formula can be evaluated in p time units on a *parallel* computer provided q processors are available". Another equivalent way of thinking of this is that the RK formula has p blocks of stages — or *super-stages* — with each super-stage consisting of at most q stages, and all the stages within each super-stage can be evaluated in *parallel*. Thus, on a parallel machine with at least q processors, the RK formula can be evaluated in p time units, although the number of stages, s, may be larger than p. Since we require p to be as small as possible, all stages within two blocks cannot be evaluated simultaneously — otherwise the two blocks could be merged into one and p would be reduced. The stages within each block are typically evaluated in parallel, but the blocks themselves are normally computed sequentially, although the definition does not exclude the possibility that some stages within one block can be evaluated simultaneously with some stages in another. The discussion of parallel ERK formulas at the beginning of §3 should help to clarify these concepts.

To complete the preliminaries, we remind the reader that the *minimum polynomial* of a square matrix X is the polynomial $m(x)$ of least degree for which $m(X) = \vec{0}$.

Using these definitions and notation, we now state and prove the main result of this section.

THEOREM 2.1. *If $m(x) = \prod_{i=1}^{k}(x-\gamma_i)^{r_i}$ is the minimum polynomial for the coefficient matrix A of a RK formula, then the order of the formula is at most $r + d + \delta$ where $r = \sum_{i=1}^{k} r_i$, d is the number of distinct complex eigenvalues of A, and $\delta = 1$ if A has at least one nonzero real eigenvalue, otherwise $\delta = 0$.*

PROOF: Since $m(x) = \prod_{i=1}^{k}(x-\gamma_i)^{r_i}$ is the minimum polynomial for A, there exists a nonsingular matrix S such that $S^{-1}AS = \mathrm{diag}(\gamma_i I + E_i)$, where $E_i^{r_i} = \vec{0}$ for $i = 1, \ldots, k$. Let $\tilde{b}^T = b^T S$, $\tilde{e} = S^{-1}e$ and partition both $\tilde{b} = (\tilde{b}_i)_{i=1}^{k}$ and $\tilde{e} = (\tilde{e}_i)_{i=1}^{k}$ to conform with the blocks of $S^{-1}AS$. Then the stability function $R(z)$ of the formula satisfies

$$\begin{aligned}
R(z) &= 1 + z b^T (I - zA)^{-1} e \\
&= 1 + z b^T S \left(S^{-1}(I - zA)S\right)^{-1} S^{-1} e \\
&= 1 + \sum_{i=1}^{k} z \tilde{b}_i^T \left((1-\gamma_i z)I - zE_i\right)^{-1} \tilde{e}_i \\
&= 1 + \sum_{i=1}^{k} \frac{z}{1-\gamma_i z} \tilde{b}_i^T \left(I - \frac{z}{1-\gamma_i z} E_i\right)^{-1} \tilde{e}_i \\
&= 1 + \sum_{i=1}^{k} \frac{z}{1-\gamma_i z} \tilde{b}_i^T \left(I + \frac{z}{1-\gamma_i z} E_i + \cdots + \left(\frac{z}{1-\gamma_i z} E_i\right)^{r_i - 1}\right) \tilde{e}_i \\
&= 1 + \sum_{i=1}^{k} \frac{P_i(z)}{(1-\gamma_i z)^{r_i}}
\end{aligned}$$

where, for $i = 1, \ldots, k$, $P_i(z)$ is a polynomial of degree at most r_i. Hence, $R(z) = P(z)/Q(z)$ for $P(z)$ a polynomial of degree at most $r = \sum_{i=1}^{k} r_i$ and $Q(z) = \prod_{i=1}^{k}(1 - \gamma_i z)^{r_i}$. It follows immediately from [**52**, Theorem 8] that $R(z)$ is a rational approximation to e^z of order $\hat{\nu} \le r + d + \delta$, whence the order of the RK formula is $\nu \le \hat{\nu} \le r + d + \delta$. ∎

3. Parallelism in "Classical" Explicit Runge-Kutta Methods. One of the first questions one is likely to ask when investigating the potential for parallelism in ERK formulas is: Can we retain the explicit nature of an ERK formula while computing blocks of internal stage values in parallel? Clearly, this requires that each $Y_{n,i}$ in the first block of internal stage values cannot depend on any $\{Y_{n,j}\}$. Thus, each must be of the form $Y_{n,i} = y_n$. (As these are all identical, there is no advantage from a mathematical point-of-view to computing more than one; consequently, we could identify them all as $Y_{n,1}$. However, in practice, there might be some benefit to duplicating the computation of $F_{n,1} = f(x_n, y_n)$ on different processors — for example, to avoid the need to "broadcast" this value.) Similarly, the second block of internal stage values to be computed in parallel can depend on the $\{Y_{n,i}\}$ in the first block only. Continuing this argument, it follows that the k^{th} block of internal stage values to be computed in parallel can depend on the $\{Y_{n,i}\}$ in blocks $1, \ldots, k-1$ only. If p is the number of blocks and q is the number of internal stage values in the largest block, then the formula with internal stage values grouped as described above is a s-stage p-parallel q-processor ERK formula. After a possible renumbering of stages, the RK tableau for such a formula can be written in block lower-triangular form as shown in Figure 3.1, where $A_{kl} = [a_{ij}]$ is a possibly full but not necessarily square submatrix of RK coefficients, while $C_k = (c_i)$ and $B_l = (b_j)$ are subvectors of RK nodes and weights, respectively. The number of rows in A_{kl} equals the number of elements in C_k which in turn equals the number of $Y_{n,i}$'s in the k^{th} block of internal stage values to be computed in parallel (which is at most q). Similarly, the number of columns in A_{kl} equals the number of elements in B_l which in turn equals the number of $Y_{n,i}$'s in the l^{th} block of internal stage values (which again is at most q). Using elementary graph theory, Iserles and Nørsett [**29**] give a more rigorous formulation of parallel ERK schemes.

$$
\begin{array}{c|cccc}
0 & 0 & & & \\
C_2 & A_{21} & 0 & & \\
\vdots & \vdots & \ddots & \ddots & \\
C_p & A_{p1} & \cdots & A_{p\,p-1} & 0 \\
\hline
 & B_1^T & \cdots & B_{p-1}^T & B_p^T
\end{array}
$$

Figure 3.1. A p-Parallel ERK Formula.

As a rule-of-thumb for good "load-balancing", the number of internal stage values in each block should be about q (except, as explained above, the first block typically consists of $Y_{n,1}$ only), whence the dimension of the blocks A_{kl} should be "close" to being $q \times q$ (except for the blocks A_{k1} which typically have just one column). Consequently, for good load-balancing, formulas with $s \approx pq$ are preferred.

Since the traditional "Butcher barriers" for the order of classical ERK formulas depend upon s while the time to evaluate a s-stage p-parallel q-processor ERK formula is proportional to p on many q-processor parallel computers, one might hope to attain a high-order explicit PaRK formula of the form described above having a significant speedup over traditional sequential ERK methods. However, as most authors who have considered parallel ERK formulas have observed, there is much less potential for parallelism in this class of schemes than first appears. As the next theorem shows, both the order and the stability are severely constrained.

THEOREM 3.1. *The stability polynomial $P(z)$ for a s-stage p-parallel q-processor ERK formula is of degree at most p, independent of s and q. Therefore, the order of such a scheme is at most p, and, if the order is $\nu \leq p$, then $P(z) = \sum_{i=0}^{\nu} z^i/i! + \sum_{i=\nu+1}^{p} p_i z^i$ where $p_i = b^T A^{i-1} e \in \mathbb{R}$ for $i = \nu+1, \ldots, p$.*

PROOF: The order bound follows from Theorem 2.1 after observing that the minimum polynomial for the coefficient matrix A of the formula is $m(x) = x^r$ for some $r \le p$, since $A^p = \vec{0}$, independent of s and q. However, it is both simple and instructive to prove the order bound together with the stability result directly in this simple case.

As noted above, $A^p = \vec{0}$. Therefore, the stability polynomial of the ERK formula is

$$P(z) = 1 + zb^T(I - zA)^{-1}e = 1 + \sum_{i=1}^{p}(b^T A^{i-1}e)\, z^i,$$

where $b^T A^{i-1}e \in \mathbb{R}$, whence $P(z)$ is a real polynomial of degree at most p, independent of s and q. Consequently, $P(z)$ is an approximation to e^z of order at most p, from which it follows that the order of the RK formula is at most p. Moreover, if the order of the RK formula is $\nu \le p$, then $P(z)$ is an approximation to e^z of order at least ν, whence $P(z) = \sum_{i=0}^{\nu} z^i/i! + \sum_{i=\nu+1}^{p} p_i z^i$ for $p_i = b^T A^{i-1}e \in \mathbb{R}$ for $i = \nu+1, \dots, p$. ∎

We proved several years ago that this order bound is attainable. A different proof of this simple result was found independently by van der Houwen and Sommeijer [**23**] [**25**], although they do not explicitly state the extension to interpolants nor stability polynomials.

THEOREM 3.2. *For any positive integer p, there exists a p-parallel family of embedded ERK formulas of orders 1 through p, with each formula of order ν, $1 \le \nu \le p$, being associated with a family of interpolants having local errors of orders $\hat{\nu}+1$ for $\hat{\nu} = 1, \dots, \nu$. Moreover, for any given set of real constants $\{p_i^{(\nu)} : i = \nu+1, \dots, p,\ \nu = 1, \dots, p-1\}$ for which $p_{\nu+1}^{(\nu)} \ne 1/(\nu+1)!$, we can construct each formula of order $\nu < p$ so that its stability polynomial is $P(z) = \sum_{i=0}^{\nu} z^i/i! + \sum_{i=\nu+1}^{p} p_i^{(\nu)} z^i$ for the specified constants $\{p_i^{(\nu)}\}$.*

REMARK. By a p-parallel family of embedded ERK formulas with interpolants we mean that all formulas and interpolants in the family can be calculated in p time units, where, as noted in §2, a time unit for an ERK formula is the time required for a function evaluation of the form $f(x + c_i h_n, Y)$ plus "a little" overhead.

PROOF: Assume p processors are available and, for $k = 1, \dots, p$, let the k^{th} processor use the forward Euler formula with k equally spaced internal steps to compute the approximation $y_{n+1}^{(k)}$. A p^{th}-order approximation y_{n+1} can be computed from the $\{y_{n+1}^{(k)}\}_{k=1}^{p}$ by polynomial extrapolation without any additional function evaluations. As is well-known [**21**, §II.9], this extrapolation process can be rewritten as an ERK formula, and, for our particular purposes, as a p-parallel p^{th}-order ERK formula.

Lower order approximations are given as a by-product of the extrapolation process without any additional function evaluations. For $\nu = 1, \dots, p$, the natural way of computing them, though, uses ν of the $\{y_{n+1}^{(k)}\}$ to compute an approximation of order ν by $\nu - 1$ extrapolation stages. The stability function for this approximation is $P(z) = \sum_{i=0}^{\nu} z^i/i!$. However, extrapolation can be viewed as a process of solving for the error coefficients in the error expansions of $\{y_{n+1}^{(k)}\}$ and of eliminating the associated error terms. Since the associated linear system is nonsingular [**21**, p. 220], we can add to the naturally arising approximation of order ν any linear combination of error coefficients of orders $\nu+1, \dots, p$. In particular, we can choose the linear combination of error terms in such a way that the stability polynomial becomes $P(z) = \sum_{i=0}^{\nu} z^i/i! + \sum_{i=\nu+1}^{p} p_i^{(\nu)} z^i$ for the specified real constants $\{p_i^{(\nu)}\}$ and the order of the formula remains ν since $p_{\nu+1}^{(\nu)} \neq 1/(\nu+1)!$. Note, this modification does not require any additional function evaluations. Furthermore, the required coefficients can all be precomputed and the formulas of orders $1, \ldots, p$ with the required stability polynomials can be represented as a p-parallel family of embedded ERK schemes.

To construct an interpolant for the formula of order p, use the same extrapolation process to compute $p-1$ auxiliary approximations each of order p at $p-1$ distinct points within (x_n, x_{n+1}). Then interpolate y_n, y_{n+1} and the $p-1$ auxiliary y-values with a polynomial of degree p, giving rise to an interpolant having local error $p+1$. For any or all $\nu = 1, \ldots, p$, an interpolant having local order $\nu+1$ can be constructed in a similar way by interpolating y_n, y_{n+1} and $\nu-1$ of the auxiliary y-values with a polynomial of degree ν. Since y_{n+1} and the $p-1$ auxiliary y-values can all be computed simultaneously, this remains a p-parallel ERK process.

For $\nu = 1, \ldots, p$ and $\hat{\nu} = 1, \ldots, \nu$, an interpolant having local error of order $\hat{\nu}+1$ can be constructed for the embedded formula of order ν in a similar manner, preserving the p-parallel ERK process. ∎

As noted by van der Houwen and Sommeijer [**23**] [**25**], an interesting question is: How many processors are required to compute a p-parallel $p^{\mathbf{th}}$-order ERK formula? A similar question arises for a p-parallel $p^{\mathbf{th}}$-order ERK formula with an interpolant of local order p or $p+1$ in particular or a family of interpolants having local errors 1 through $p+1$. In the proof of our result, we have not attempted to keep the number of stages or processors used small. We could easily halve the number employed by using the same processor to compute both $y_{n+1}^{(k)}$ and $y_{n+1}^{(p-k)}$ in p time units, and halve this number again by using the explicit midpoint rule rather than the forward Euler formula as the base method, but in this case it may not be possible to choose the stability polynomials at will. Even with this saving, we believe that our simple construction here leads to a gross over-estimate of both the stages and processors required for a p-parallel $p^{\mathbf{th}}$-order ERK formula with or without interpolants.

As noted above, the potential for parallelism in explicit PaRK methods is limited. Our enthusiasm for them is decreased further by the seemingly greater potential for parallelism in nonstandard predictor-corrector implementations of high-order RK formulas with interpolants [**31**] and other explicit block or general linear methods. Nevertheless, there are some limited advantages that can be attained by exploiting parallelism within ERK formulas and, because of our greater familiarity with them, parallel methods based on this class of formulas may be worth pursuing — in the short term at least.

As a specific example of a minor advantage, recall that 6 stages are required to obtain a $5^{\mathbf{th}}$-order ERK formula [**10**, §322]. However, each member of Kutta's [**37**] 3-parameter family of $5^{\mathbf{th}}$-order 6-stage ERK formulas has $a_{65} = 0$, whence the last two stages can be computed simultaneously. Therefore, each formula in Kutta's family is a 5-parallel 2-processor ERK scheme. Similarly, by taking $v = 0$ in Butcher's 2-parameter family of 6-stage $5^{\mathbf{th}}$-order ERK formulas [**10**, p. 199], we obtain a 1-parameter family each of which has $a_{43} = 0$ allowing the simultaneous evaluation of $F_{n,3}$ and $F_{n,4}$. This gives rise to another 5-parallel 2-processor family of ERK formulas. Theorem 3.2 ensures that a similar saving can be obtained for all higher order ERK formulas, although it does not guarantee that the number of processors (and the resulting inter-processor communication) required can be kept small, as is the case in these examples.

As a second example, it is tedious but straightforward to show that 5 stages are required for an ERK formula-pair of orders 3 and 4. However, it is easy to derive a 5-stage 4-parallel 2-processor formula-pair of orders 3 and 4. In fact, RKN(3,4) exhibited in [**16**, p. 205] is one such formula-pair, since a_{43} being 0 permits $F_{n,3}$ and $F_{n,4}$ to be computed in parallel.

Also note that all function evaluations required in each iteration of step (2) or (3) of the "boot-strapping" algorithm described in [**16**, p. 197] for the construction of interpolants for RK formulas can be evaluated in parallel. This could substantially reduce the time required on parallel machines for the evaluation of high-order ERK formulas with interpolants, sometimes referred to as *continuous* ERK formulas. In fact, there may be more than a minor advantage to be gained for high-order ERK formulas with interpolants, since the number of function evaluations required for continuous ERK schemes seems to grow even more rapidly with the order than it does for discrete ERK formulas. Owren and Zennaro [**46**] [**47**] [**48**] showed that 4, 6 and 8 stages are required for continuous ERK formulas of orders 3, 4, and 5, respectively. The least number of stages for continuous ERK formulas of orders 6, 7 and 8 that we know of are 11, 16 and 22, respectively. Hence, there is a potential for considerable savings in run time by exploiting parallelism in the computation of continuous ERK methods.

As a final example, Enright and Higham [**15**] have investigating the use of parallelism in *defect evaluation* and *control* to improve the reliability of ERK methods.

The preliminary numerical experiments of van der Houwen and Sommeijer [**23**] [**25**] demonstrate the potential of parallel ERK schemes. Their results show that a simple variable-stepsize implementation of their parallel ERK schemes, a subclass of their explicit *PIRK* methods, can yield a speedup of more than two at stringent tolerances compared to the efficient sequential ERK code DOPRI8 [**21**].

4. Parallelism in "Strictly-Diagonal" Implicit Runge-Kutta Methods. As we have seen, classical ERK formulas have limited potential for parallelism. So it is natural to consider next if IRK formulas may have more. In this section, we investigate the simplest class of implicit formulas — the *Strictly-Diagonal* IRK formulas for which $a_{ij} = 0$ for $i \neq j$. These formulas have the attractive property that all $Y_{n,i}$'s can be computed simultaneously, since each $Y_{n,i}$ depends on itself only. Thus, they are s-stage 1-parallel s-processor PaRK formulas.

A proof similar to the one in the previous section that limits the order of a s-stage p-parallel ERK formula to p fails in this case. In fact, we establish instead that, for any positive integer s, a s-stage 1-parallel s-processor Strictly-Diagonal IRK formula exists that is of order $2s$ for a restricted class of linear problems with constant coefficients. However, the formulas of order $2s$ have complex coefficients. If the coefficients are constrained to be real, then the maximal obtainable order of a s-stage 1-parallel s-processor Strictly-Diagonal IRK formula applied to this class of equations is $s + 1$.

The order results for the restricted class of linear problems, though, do not extend to nonlinear problems or even linear problems with variable coefficients. For these more general IVPs, we show that the maximal order of a Strictly-Diagonal IRK formula is two.

Finally, at the end of this section, we briefly consider a generalization of Strictly-Diagonal IRK formulas to *Strictly-Block-Diagonal* IRK formulas that allow blocks rather than single elements on the diagonal of the coefficient tableau. We generalize our result for the restricted class of linear problems by showing that, if 2×2 blocks are allowed, then formulas of order $2s$ with real coefficients only can be derived. We also quote some results of other authors about Strictly-Block-Diagonal IRK formulas for general smooth IVPs.

To prove the order result quoted above for Strictly-Diagonal IRK formulas applied to linear differential equations, we first establish the following.

LEMMA 4.1. *Let $R(z) = P(z)/Q(z)$, where P is a polynomial of degree at most s, $Q(z) = \prod_{i=1}^{s}(1 - \gamma_i z)$ for $\gamma_1, \ldots, \gamma_s \in \mathbb{C}$ distinct and nonzero, and $P(0) = Q(0) = 1$. Define an associated s-stage Strictly-Diagonal IRK formula by $c_i = a_{ii} = \gamma_i$ and*

$$b_i = \gamma_i \frac{P(1/\gamma_i)}{Q_i(1/\gamma_i)} \quad \text{where} \quad Q_i(z) = \prod_{\substack{j=1 \\ j \neq i}}^{s} (1 - \gamma_j z).$$

Then this formula applied to the simple test equation $y' = \lambda y$ for $\lambda \in \mathbb{C}$ yields $y_{n+1} = R(h_n \lambda) y_n$.

PROOF: Letting $z = h_n \lambda$ and applying the s-stage Strictly-Diagonal IRK formula defined above to the simple test equation $y' = \lambda y$, we get

$$y_{n+1} = \left(1 + z \sum_{i=1}^{s} \frac{b_i}{1 - \gamma_i z}\right) y_n = \frac{Q(z) + z \sum_{i=1}^{s} b_i Q_i(z)}{Q(z)} y_n.$$

Hence, all that remains is to show that

$$P(z) = Q(z) + z \sum_{i=1}^{s} b_i Q_i(z). \tag{4.1}$$

By the choice of b_i, the left and right sides of (4.1) are equal at $z = 1/\gamma_i$ for $i = 1, \ldots, s$, and, by the hypothesis, $P(0) = Q(0) = 1$. Therefore, since the left and right sides of (4.1) are polynomials of degree at most s that are equal at $s + 1$ distinct points, they are equal for all z. ∎

THEOREM 4.2. *Consider the restricted class of IVPs associated with linear differential equations of the form $y' = My + ux + v$ where the matrix M and the vectors u and v are constants. There exist s-stage 1-parallel s-processor Strictly-Diagonal IRK formulas that are of maximal obtainable order $2s$ when applied to any IVP from this restricted class of problems. If the coefficients of the formula are constrained to be real, then the maximal obtainable order is reduced to $s + 1$.*

PROOF: First note that, if a formula satisfies the *simplifying assumption* $c_i = \sum_{j=1}^{s} a_{ij}$ for $i = 1, \ldots, s$, then the order conditions for this restricted class of IVPs are the same as those that arise from the simple test equation $y' = \lambda y$ for $\lambda \in \mathbb{C}$ a constant. Therefore, to show that a RK formula which satisfies this simplifying assumption is of order ν for this restricted class of IVPs, it is sufficient to show that it is of order ν for the simple test equation $y' = \lambda y$.

Let $R(z) = P(z)/Q(z)$ be the $s^{\mathbf{th}}$ diagonal Padé approximation to e^z. It is well-known that the order of this approximation is $2s$ and that the polynomials P and Q, each of degree s, have no common factors, from which it follows that $P(0) = Q(0) \neq 0$, whence P and Q can be normalized so that $P(0) = Q(0) = 1$. In addition, Theorem 8 of [**52**] ensures that the roots of Q are distinct and nonzero. Hence, $Q(z)$ can be written as $Q(z) = \prod_{i=1}^{s}(1 - \gamma_i z)$ for $\gamma_1, \ldots, \gamma_s$ distinct and nonzero. Therefore, $R(z)$ satisfies the assumptions of Lemma 4.1 and the associated s-stage Strictly-Diagonal IRK formula defined in Lemma 4.1 is of order $2s$ for the simple test equation $y' = \lambda y$. Consequently, the formula is of order $2s$ for all problems in the restricted class of IVPs as well. Moreover, $2s$ is the maximal obtainable order for a s-stage Strictly-Diagonal IRK formula since this is the maximal obtainable order for any rational approximation $R(z) = P(z)/Q(z)$ to e^z with P and Q of degree at most s.

Since the denominator $Q(z)$ of the $s^{\mathbf{th}}$ diagonal Padé approximation to e^z has complex roots for $s > 1$ [**52**, Theorem 8], the associated Strictly-Diagonal IRK formula has complex coefficients. To ensure real coefficients only, we require that $Q(z)$ has real roots only, and it is well-known [**44**, Theorem 2.1] that $R(z)$ can be an approximation to e^z of order at most $s+1$ in this case. Moreover, Proposition 6 of [**43**] and the discussion following it indicate how to construct rational approximations $R(z) = P(z)/Q(z)$ to e^z of order $s+1$ for which P and Q satisfy the assumptions of Lemma 4.1 and Q has real roots only. Consequently, the associated Strictly-Diagonal IRK formulas are of maximal obtainable order $s+1$ when applied to any problem in the restricted class of IVPs. ∎

We turn now to the case of more general IVPs and establish the following negative result.

THEOREM 4.3. *The order of a Strictly-Diagonal IRK formula is* $\min(2, \hat{\nu})$, *where* $\hat{\nu}$ *is the order of the formula's stability function.*

PROOF: A RK formula must satisfy

(1) $\sum b_i = 1$ to be $1^{\mathbf{st}}$-order,
(2) $\sum b_i a_{ij} = 1/2$ to be $2^{\mathbf{nd}}$-order,
(3) and both $\sum b_i a_{ij} a_{ik} = 1/3$ and $\sum b_i a_{ij} a_{jk} = 1/6$ to be $3^{\mathbf{rd}}$-order.

For a Strictly-Diagonal IRK formula, $a_{ij} = 0$ for $i \neq j$, whence the two $3^{\mathbf{rd}}$-order conditions reduce to $\sum b_i a_{ii}^2 = 1/3$ and $\sum b_i a_{ii}^2 = 1/6$ which are clearly incompatible. Therefore, a Strictly-Diagonal IRK formula cannot be $3^{\mathbf{rd}}$-order in general. Conditions (1) and (2) are also the conditions that the formula's stability function is of order one or two, respectively. Thus, the order of the RK formula is $\min(2, \hat{\nu})$. ∎

A natural extension of the formulas considered above are Strictly-Block-Diagonal IRK formulas having coefficient tableaus of the form shown in Figure 4.1, where $A_{kk} \in \mathbb{R}^{s_k \times s_k}$, B_k, $C_k \in \mathbb{R}^{s_k}$ and $\sum_{k=1}^{\hat{s}} s_k = s$. Each block of internal stage values can be computed in parallel. Moreover, the techniques for IRK formulas discussed in §6 can be applied within each diagonal block to further increase the potential for parallelism within these formulas. Many can be implemented as s-stage 1-parallel s-processor PaRK methods.

One advantage of this extension is that it permits one to derive s-stage Strictly-Block-Diagonal IRK formulas of order $2s$ with 1×1 or 2×2 diagonal blocks having real coefficients

$$
\begin{array}{c|cccc}
C_1 & A_{11} & 0 & \ldots & 0 \\
C_2 & 0 & A_{22} & \ldots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
C_{\hat{s}} & 0 & 0 & \ldots & A_{\hat{s}\hat{s}} \\
\hline
 & B_1^T & B_2^T & \ldots & B_{\hat{s}}^T
\end{array}
$$

Figure 4.1. A Strictly-Block-Diagonal IRK Formula.

only. We state below without proof the lemma and theorem upon which this result is based. Their verification requires a modest extension only of the techniques used to prove Lemma 4.1 and Theorem 4.2, respectively.

LEMMA 4.4. *Let $R(z) = P(z)/Q(z)$, where P is a polynomial of degree at most s, $Q(z) = \prod_{i=1}^{s}(1 - \gamma_i z)$ for $\gamma_1, \ldots, \gamma_s \in \mathbb{C}$ distinct and nonzero, and $P(0) = Q(0) = 1$. For definiteness, assume that the $\{\gamma_i\}$ are ordered such that the first $2\bar{s}$ are complex conjugate pairs (i.e., $\gamma_{2i-1} = \bar{\gamma}_{2i}$ for $i = 1, \ldots, \bar{s}$) and the remaining $s - 2\bar{s}$ are real. Define an associated s-stage Strictly-Block-Diagonal IRK formula having 1×1 and 2×2 diagonal blocks and real coefficients only as follows.*

(1) *For $i = 1, \ldots, \bar{s}$, let $\alpha_i = \Re(\gamma_{2i-1})$, $\beta_i = \Im(\gamma_{2i-1})$,*

$$
A_{ii} = \begin{pmatrix} \alpha_i & \beta_i \\ -\beta_i & \alpha_i \end{pmatrix},
$$

$$
\begin{aligned}
b_{2i-1} = \sqrt{-1}\left(\frac{|\gamma_{2i-1}|}{2\beta_i}\right)^2 \Bigg(&(1 - \alpha_i/\bar{\gamma}_{2i-1} - \beta_i/\bar{\gamma}_{2i-1})\frac{\gamma_{2i-1}P(1/\gamma_{2i-1})}{Q_i(1/\gamma_{2i-1})} \\
&-(1 - \alpha_i/\gamma_{2i-1} - \beta_i/\gamma_{2i-1})\frac{\bar{\gamma}_{2i-1}P(1/\bar{\gamma}_{2i-1})}{Q_i(1/\bar{\gamma}_{2i-1})}\Bigg), \\
b_{2i} = \sqrt{-1}\left(\frac{|\gamma_{2i-1}|}{2\beta_i}\right)^2 \Bigg(&(1 - \alpha_i/\gamma_{2i-1} + \beta_i/\gamma_{2i-1})\frac{\bar{\gamma}_{2i-1}P(1/\bar{\gamma}_{2i-1})}{Q_i(1/\bar{\gamma}_{2i-1})} \\
&-(1 - \alpha_i/\bar{\gamma}_{2i-1} + \beta_i/\bar{\gamma}_{2i-1})\frac{\gamma_{2i-1}P(1/\gamma_{2i-1})}{Q_i(1/\gamma_{2i-1})}\Bigg).
\end{aligned}
$$

(2) *For $i = 2\bar{s}+1, \ldots, s$, let $A_{ii} = \gamma_i$ and $b_i = \gamma_i P(1/\gamma_i)/Q_i(1/\gamma_i)$.*
(3) *For $i = 1, \ldots, s$, let $c_i = \sum_{j=1}^{s} a_{ij}$.*
(4) *For $i = 1, \ldots, \bar{s}$ and $i = 2\bar{s}+1, \ldots, s$, let*

$$
Q_i(z) = \prod_{\substack{j=1 \\ j \neq i}}^{\bar{s}} \det(I - A_{jj}z) \prod_{\substack{j=2\bar{s}+1 \\ j \neq i}}^{s} (1 - \gamma_j z).
$$

Then this formula applied to the simple test equation $y' = \lambda y$ *for* $\lambda \in \mathbb{C}$ *yields* $y_{n+1} = R(h_n\lambda)y_n$.

REMARK. The b_{2i-1} defined in part (1) of the lemma above is real since

$$(1-\alpha_i/\bar{\gamma}_{2i-1}-\beta_i/\bar{\gamma}_{2i-1})\frac{\gamma_{2i-1}P(1/\gamma_{2i-1})}{Q_i(1/\gamma_{2i-1})}$$

is the complex conjugate of

$$(1-\alpha_i/\gamma_{2i-1}-\beta_i/\gamma_{2i-1})\frac{\bar{\gamma}_{2i-1}P(1/\bar{\gamma}_{2i-1})}{Q_i(1/\bar{\gamma}_{2i-1})}.$$

Similarly, it is easy to see b_{2i} is real.

THEOREM 4.5. *Consider the class of IVPs associated with linear differential equations of the form* $y' = My + ux + v$ *where the matrix* M *and the vectors* u *and* v *are constants. There exist* s*-stage Strictly-Block-Diagonal IRK formulas having* 1×1 *and* 2×2 *diagonal blocks and real coefficients only that are of maximal obtainable order* $2s$ *when applied to any IVP in this restricted class of problems.*

Others have begun to investigate Strictly-Block-Diagonal IRK formulas as well. Lie [**38**] discusses some preliminary results on the maximal obtainable order of such formulas while Iserles and Nørsett [**29**] provide some more concrete results. In particular, they derive a family of 4-stage 4$^{\text{th}}$-order L-stable (but not algebraically stable) Strictly-Block-Diagonal IRK formulas each having two 2×2 blocks. An example of one such formula is given in Figure 4.2. Thus, parallelism yields some advantage over classical sequential methods in this case since 4$^{\text{th}}$-order L-stable 2-stage IRK formulas do not exist.

$$\begin{array}{c|cccc}
\frac{1}{2}-\frac{\sqrt{3}}{6} & \frac{5}{12} & \frac{1}{12}-\frac{\sqrt{3}}{6} & 0 & 0 \\
\frac{1}{2}+\frac{\sqrt{3}}{6} & \frac{1}{12}+\frac{\sqrt{3}}{6} & \frac{5}{12} & 0 & 0 \\
\frac{1}{2}-\frac{\sqrt{3}}{6} & 0 & 0 & \frac{1}{2} & -\frac{\sqrt{3}}{6} \\
\frac{1}{2}+\frac{\sqrt{3}}{6} & 0 & 0 & \frac{\sqrt{3}}{6} & \frac{1}{2} \\
\hline
 & \frac{3}{2} & \frac{3}{2} & -1 & -1
\end{array}$$

Figure 4.2. A 4-Stage 4$^{\text{th}}$-Order L-Stable Strictly-Block-Diagonal IRK Formula.

Iserles and Nørsett [**29**] also obtain a bound on the maximal order of a family of Strictly-Block-Diagonal IRK formulas. This family consists of those formulas from this class having 2×2 blocks A_{kk} only for which each block A_{kk} together with the corresponding B_k normalized to sum to one is at least 3$^{\text{rd}}$-order when considered as a RK formula on its own. (The formula above is in this family.) They show that, independent of the number of blocks, the maximal obtainable order for a formula in this family is four. Thus, the advantage one can gain from exploiting parallelism is limited in this family of formulas as well.

Both Lie [**38**] and Iserles and Nørsett [**29**] also consider Strictly-Block-Diagonal IRK formulas for which each block A_{kk} is lower triangular. We summarize briefly some of their results for these formulas in the next section.

5. Parallelism in Diagonally-Implicit Runge-Kutta Methods. As noted in §2, the coefficients of a *Diagonally-Implicit* RK (DIRK) formula satisfy $a_{ij} = 0$ for $j > i$. Thus, $Y_{n,1} = y_n + h_n a_{11} f(x_n + c_1 h_n, Y_{n,1})$ is implicit in $Y_{n,1}$ only, and, consequently, can be solved for $Y_{n,1}$ without knowledge of any other $Y_{n,j}$ for $j > 1$. Similarly, given $Y_{n,j}$ for $j = 1, \ldots, i-1$, $Y_{n,i} = y_n + h_n \sum_{j=1}^{i} a_{ij} f(x_n + c_j h_n, Y_{n,j})$ is implicit in $Y_{n,i}$ only, and, consequently, can be solved for $Y_{n,i}$ without knowledge of any other $Y_{n,j}$ for $j > i$. This property constitutes the main advantage of DIRK formulas over Fully-Implicit RK (FIRK) formulas for which all the $Y_{n,i}$'s (may) depend on all the others and cannot (in general) be computed independently of one another. (However, some techniques that ameliorate this disadvantage for FIRK formulas have been developed and are discussed in §6.)

DIRK formulas are used primarily to solve stiff IVPs and consequently a Newton-like iteration is typically employed to solve $Y_{n,i} = y_n + h_n \sum_{j=1}^{i} a_{ij} f(x_n + c_j h_n, Y_{n,j})$. This requires solutions of linear systems with a coefficient matrix $I - h_n a_{ii} J$, where J is an approximation to the Jacobian $f_y(x_n + c_i h_n, Y_{n,i})$. *Singly-Diagonally-Implicit* RK (SDIRK) formulas, for which all diagonal coefficients $\{a_{ii}\}$ are equal, enjoy the advantage that only one matrix factorization is needed for $I - h_n a_{ii} J$ to solve for all internal stage values $\{Y_{n,i}\}$ (assuming the same J is used throughout).

Although advantageous for sequential machines, the serial nature of the solution process for the internal stage values $\{Y_{n,i}\}$ of DIRK formulas is not favourable for parallel computers. The objective of this section is to extend the favorable characteristic of DIRK formulas for sequential machines to parallel ones. More specifically, our aim is to derive a class of formulas for which blocks of $Y_{n,i}$'s can be computed simultaneously while retaining the property that each $Y_{n,i}$ in the block depends upon itself and previously computed $Y_{n,j}$'s only. We give the general structure of such formulas, examples of a few, and some preliminary results on order bounds and stability characteristics of these formulas.

Clearly, if all $Y_{n,i}$'s in the first block are to be computed simultaneously while retaining the simple form of the implicit equations described above, then the equation for each must be of the form $Y_{n,i} = y_n + h_n a_{ii} f(x_n + c_i h_n, Y_{n,i})$. Similarly, if all $Y_{n,i}$'s in the $k^{\mathbf{th}}$ block are to be computed simultaneously while again retaining the simple form of the implicit equations described above, then the equation for each must be of the form

$$Y_{n,i} = y_n + h_n \sum_j h_n a_{ij} f(x_n + c_j h_n, Y_{n,j}) + h_n a_{ii} f(x_n + c_i h_n, Y_{n,i}),$$

where $\sum_j$ is taken over previously computed $Y_{n,j}$'s in blocks $1, \ldots, k-1$ only. Hence, after possibly re-ordering the internal stage values of the formula, its coefficient tableau must be of the block lower-triangular form shown in Figure 5.1, where D_k is a square diagonal matrix while, as in §3, $A_{kl} = [a_{ij}]$ is a potentially full but not necessarily square matrix and both $C_k = (c_i)$ and $B_l = (b_j)$ are vectors. The number of rows in both A_{kl}

and D_k equals the number of elements in C_k which in turn equals the number of $Y_{n,i}$'s in the k^{th} block of internal stage values. Similarly, the number of columns in both A_{kl} and D_l equals the number of elements in B_l which in turn equals the number of $Y_{n,i}$'s in the l^{th} block of internal stage values. If we assume that the largest diagonal block is of dimension $q \times q$, then we call such a formula a s-stage p-parallel q-processor DIRK or SDIRK formula depending upon whether or not all the diagonal coefficients $\{a_{ii}\}$ are equal. Using elementary graph theory, Iserles and Nørsett [**29**] give a more rigorous formulation of parallel DIRK schemes.

$$
\begin{array}{c|cccc}
C_1 & D_1 & & & \\
C_2 & A_{21} & D_2 & & \\
\vdots & \vdots & \vdots & \ddots & \\
C_p & A_{p1} & A_{p2} & \dots & D_p \\
\hline
 & B_1^T & B_2^T & \dots & B_p^T
\end{array}
$$

Figure 5.1. A p-Parallel DIRK Formula.

We begin with a bound on the order of a p-parallel DIRK formula in terms of the number of distinct diagonal elements.

THEOREM 5.1. *Let $\gamma_1, \dots, \gamma_k$ be the distinct diagonal coefficients of a s-stage p-parallel q-processor DIRK formula and assume that γ_i occurs with multiplicity m_i. Then the maximal order of the formula is $1 + \sum_{i=1}^{k} \min(m_i, p)$, independent of s and q.*

PROOF: Let A be the coefficient matrix of the s-stage p-parallel q-processor DIRK formula:

$$
A = \begin{pmatrix} D_1 & 0 & \dots & 0 \\ A_{21} & D_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A_{p1} & A_{p2} & \dots & D_p \end{pmatrix}
$$

where the D_i are diagonal matrices with diagonal elements in $\{\gamma_1, \dots, \gamma_k\}$. Set $X = (A - \gamma_1 I) \cdots (A - \gamma_k I)$ and note that

$$
X = \begin{pmatrix} \tilde{D}_1 & 0 & \dots & 0 \\ \tilde{A}_{21} & \tilde{D}_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{A}_{p1} & \tilde{A}_{p2} & \dots & \tilde{D}_p \end{pmatrix}
$$

where $\tilde{D}_i = (D_i - \gamma_1 I) \cdots (D_i - \gamma_k I) = \vec{0}$ for $i = 1, \dots, p$. (The specific formula for each $\tilde{A}_{ij}$ is inconsequential in this proof.) Since X is a strictly-lower-triangular $p \times p$ block matrix, $X^p = \vec{0}$. Hence, the minimum polynomial for A divides $\prod_{i=1}^{k}(x-\gamma_i)^p$. Moreover, since the multiplicity of γ_i is m_i, the factor $(x-\gamma_i)$ occurs in the minimum polynomial for A with multiplicity at most m_i. Consequently, the minimum polynomial for A divides $\prod_{i=1}^{k}(x-\gamma_i)^{r_i}$ for $r_i = \min(m_i, p)$. Thus, it follows from Theorem 2.1 that the order of the DIRK formula is at most $r+1$ for $r = \sum_{i=1}^{k} r_i$ since all the $\{\gamma_i\}$ are real. ∎

An immediate consequence of this theorem is

COROLLARY 5.2. *The order of a s-stage p-parallel q-processor SDIRK formula is at most $p+1$, independent of s and q.*

The last result limits the potential effectiveness of SDIRK formulas in a parallel computing environment. Since algebraically stable s-stage SDIRK formulas of order $s+1$ exist for $s = 1, 2, 3$, there appears to be little advantage for low-order parallel SDIRK formulas. However, there is, for example, no 4-stage SDIRK formula of order 5. Therefore, there is potential for slight advantage for higher-order parallel SDIRK formulas. These schemes, though, are problematic in that their *stage-order* is at most two — and is one only unless $a_{ii} = 1/2$ for all $i = 1, \ldots, s$.

On the other hand, if a parallel DIRK formula has at least two distinct diagonal coefficients, then it is possible to obtain some advantage over standard serial DIRK formulas. For example, the 4-stage 2-parallel 2-processor DIRK formula shown in Figure 5.2 is 4^{th}-order and A_0-stable, but not A-stable. It is well-known [**44**] that there are no standard 4^{th}-order 2-stage DIRK formulas and Corollary 5.2 above precludes the existence of a 4^{th}-order 2-parallel SDIRK formula. However, it may be possible to find a 5^{th}-order 2-parallel DIRK formula, although we have not yet succeeded in doing so.

$$
\begin{array}{c|cccc}
1 & 1 & 0 & 0 & 0 \\
\frac{3}{5} & 0 & \frac{3}{5} & 0 & 0 \\
0 & \frac{171}{44} & -\frac{215}{44} & 1 & 0 \\
\frac{2}{5} & -\frac{43}{20} & \frac{39}{20} & 0 & \frac{3}{5} \\
\hline
 & \frac{11}{72} & \frac{25}{72} & \frac{11}{72} & \frac{25}{72}
\end{array}
$$

Figure 5.2. A 4-Stage 2-Parallel 2-Processor
4^{th}-Order A_0-Stable DIRK Formula.

Both Lie [**38**] and Iserles and Nørsett [**29**] consider parallel DIRK formulas having the restricted form shown in Figure 5.3, where each L_k is a lower-triangular matrix. These schemes form a subclass of the Strictly-Block-Diagonal RK formulas discussed in the previous section. They are also a subclass of the parallel DIRK formulas considered here: if the largest diagonal block L_k is of dimension $p \times p$, then the internal stage values of such a scheme can be reordered to yield a p-parallel q-processor DIRK formula with each A_{kl} (as well as each D_k) being a (possibly rectangular) diagonal block. An advantage of this subclass over the more general class of parallel DIRK formulas considered above is that the restricted form requires no communication between processors in the computation of the internal stage values $\{Y_{n,i}\}$.

$$
\begin{array}{c|cccc}
C_1 & L_1 & 0 & \dots & 0 \\
C_2 & 0 & L_2 & \dots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
C_q & 0 & 0 & \dots & L_q \\
\hline
 & B_1^T & B_2^T & \dots & B_q^T
\end{array}
$$

Figure 5.3. A Subclass of Strictly-Block-Diagonal RK Formulas.

Lie [**38**] derived a $3^{\mathbf{rd}}$-order 4-stage 2-parallel 2-processor SDIRK formula in this subclass that is A-stable but not algebraically stable. By Corollary 5.2, this is the maximal possible order for a 2-parallel SDIRK formula. He also derived a $4^{\mathbf{th}}$-order 4-stage 2-parallel 2-processor DIRK formula of this restricted form having a coefficient matrix with two distinct eigenvalues each of multiplicity two. This formula is not algebraically stable and its linear stability properties are not given.

Lie [**38**] implemented a fixed-stepsize code based on his $3^{\mathbf{rd}}$-order 4-stage 2-parallel 2-processor SDIRK formula. He reports some preliminary numerical experiments for this 2-processor code using "macrotasking" on a Cray XMP/2 computer and a similar code that uses one processor only on the same machine. For some simple test problems, he found a speedup of about 1.75 to 1.90 for the 2-processor implementation over the corresponding 1-processor version.

Iserles and Nørsett [**29**] derived the $4^{\mathbf{th}}$-order 4-stage 2-parallel 2-processor DIRK formula in this subclass shown in Figure 5.4. Its coefficient matrix has two distinct eigenvalues each of multiplicity two. This formula is A-stable but clearly not algebraically stable since b_3 and b_4 are negative. Furthermore, Iserles and Nørsett show that, independent of s, q and the number of distinct eigenvalues of the coefficient matrix A, there is no s-stage 2-parallel q-processor DIRK formula in this subclass of order greater than four. This is a stronger result than is possible to obtain from Theorem 5.1.

The *PIRK* formulation of van der Houwen, Sommeijer and Couzy [**26**] allows them to derive a large class of parallel DIRK schemes enjoying both high order and good stiff stability properties. Some preliminary numerical results in [**26**] demonstrate the potential of this approach.

Finally, before closing this section, we comment briefly on an obvious extension of DIRK formulas. The class of *Block DIRK (BDIRK)* formulas are those RK schemes having the same structure as the DIRK formulas discussed in this section except that each diagonal block D_k is allowed to be full. Similarly, *Block SDIRK (BSDIRK)* formulas are BDIRK formulas having a coefficient matrix with a single eigenvalue only.

$$
\begin{array}{c|cccc}
1/3 & 1/3 & 0 & 0 & 0 \\
2/3 & 1/3 & 1/3 & 0 & 0 \\
\frac{21+\sqrt{57}}{48} & 0 & 0 & \frac{21+\sqrt{57}}{48} & 0 \\
\frac{27-\sqrt{57}}{48} & 0 & 0 & \frac{3-\sqrt{57}}{24} & \frac{21+\sqrt{57}}{48} \\
\hline
 & \frac{9+3\sqrt{57}}{16} & \frac{9+3\sqrt{57}}{16} & -\frac{1+3\sqrt{57}}{16} & -\frac{1+3\sqrt{57}}{16}
\end{array}
$$

Figure 5.4. A 4-Stage 2-Parallel 2-Processor $4^{\mathbf{th}}$-Order A-Stable DIRK Formula.

Theorem 5.1 can be extended easily for BDIRK formulas. We state this result without proof as its verification is a straightforward extension of the proof of Theorem 5.1.

THEOREM 5.3. *Let $\gamma_1, \ldots, \gamma_k$ be the distinct eigenvalues of the coefficient matrix A of a s-stage p-parallel q-processor BDIRK formula. Assume that γ_i occurs with multiplicity m_i and that $\prod_{i=1}^{k}(x-\gamma_i)^{\hat{m}_i}$ is the minimum polynomial of diag(D_j). Then the order of the formula is at most $r+d+\delta$ where $r=\sum_{i=1}^{k}\min(m_i, \hat{m}_i p)$, d is the number of distinct complex eigenvalues of A, and $\delta = 1$ if A has at least one nonzero real eigenvalue, otherwise $\delta = 0$.*

An immediate consequence of this theorem is

COROLLARY 5.4. *The order of a s-stage p-parallel q-processor BSDIRK formula is at most $1+\min(s, \hat{m}p)$, where $\hat{m}$ is the degree of the minimum polynomial of diag(D_j).*

Sharper versions of Theorem 5.3 and Corollary 5.4 can be obtained from Theorem 2.1 by taking into account which diagonal block D_j has the minimum polynomial for which the factor $(x-\gamma_i)$ occurs with exponent $\hat{m}_i$. However, the statement of this sharper result seems to be too "messy" to be aesthetically pleasing: for a specific BDIRK formula, it is likely preferable to determine the appropriate order bound using Theorem 2.1 directly.

We have not yet seen examples of any potentially useful parallel BDIRK formulas.

6. Parallelism in Fully-Implicit Runge-Kutta Methods. In this section, we consider *Fully-Implicit* RK (FIRK) formulas. As noted in §2, a FIRK formula is an IRK scheme which is not a DIRK formula. That is, the stages of a FIRK formula cannot be re-ordered so that the coefficients satisfy $a_{ij} = 0$ for $i < j$.

For an IRK formula, the first equation in (2.3),

$$Y_n = e \otimes y_n + h_n A \otimes F_n, \tag{6.1}$$

is implicit in Y_n. Therefore, for general nonlinear f, some iterative scheme must be used to solve (6.1). The most common nonlinear equation solvers used in IVP codes are simple iteration

$$Y_n^{l+1} = e \otimes y_n + h_n A \otimes F_n^l \tag{6.2}$$

and Newton-like methods

$$(6.3) \qquad \begin{aligned} (I - h_n \mathcal{J}_n^l)\Delta Y_n^l &= e \otimes y_n + h_n A \otimes F_n^l - Y_n^l, \\ Y_n^{l+1} &= Y_n^l + \Delta Y_n^l, \end{aligned}$$

in which $\mathcal{J}_n^l \approx \partial(A \otimes F_n^l)/\partial Y$. In both cases, Y_n^l is the l^{th} approximation to Y_n and $F_n^l = F(x_n, Y_n^l; h_n)$.

Even if the RK formula is not implicit, both (6.2) and (6.3) permit the parallel evaluation of the s f evaluations associated with F_n^l. This observation is the basis for the parallel predictor-corrector methods discussed in [**5**], [**6**], [**23**], [**25**], [**26**] and [**31**]. We shall not discuss this novel approach here. Rather, in this section, we restrict ourselves to exploring the potential for parallelism in more standard IRK methods.

IRK formulas have been used primarily to solve stiff IVPs. It is well-known that simple iteration is ineffective in this context, since it does not converge unless the stepsize is severely restricted. As a result, virtually all IRK methods for stiff problems incorporate a Newton-like scheme to solve (6.1), although some codes also include simple iteration as an option or automatically switch between simple iteration and a Newton-like scheme depending on the local stiffness of the problem.

For large systems of ODEs, the dominant cost in the numerical integration is often the solution of the linear system in (6.3). Therefore, in the remainder of this section, we concentrate on the potential for parallelism in the solution of (6.3) and the consequences for the choice of FIRK formulas for parallel IVP solvers.

Butcher [**8**] proposed an effective scheme for solving (6.3); variants of it are discussed in [**11**] [**51**]. Central to this approach are the following two observations.

(1) If each $f_y(x_n + c_j h_n, Y_{n,j}^l)$, $j = 1, \ldots, s$, is approximated by a common $m \times m$ matrix J_n^l, then the resulting approximation $\mathcal{J}_n^l$ to $\partial(A \otimes F_n^l)/\partial Y$ is $\mathcal{J}_n^l = A \otimes J_n^l$. Using the latter approximation, we can rewrite the first equation in (6.3) in tensor-product notation as

$$(6.4) \qquad (I_s \otimes I_m - h_n A \otimes J_n^l)\Delta Y_n^l = e \otimes y_n + h_n A \otimes F_n^l - Y_n^l,$$

where I_s and I_m are $s \times s$ and $m \times m$ identity matrices, respectively.

(2) It follows from the definition of the tensor-product of two matrices (see §2) that

$$(X_1 \otimes Z_1)(X_2 \otimes Z_2) = (X_1 X_2) \otimes (Z_1 Z_2)$$

provided that the dimensions of X_1, X_2, Z_1 and Z_2 are such that the matrix products $X_1 X_2$ and $Z_1 Z_2$ are well-defined.

Thus, a similarity transformation for either A or A^{-1} can be used to simplify (6.4). The variants of Butcher's approach differ in their choice of this similarity transformation. Assuming A is nonsingular, we present below a simple variant of this class of schemes that captures the essence of the approach, although it may not be the most computationally effective version. This and other variants of Butcher's scheme can easily be modified to permit singular A.

Given any nonsingular matrix T, transform (6.4) to

$$\begin{aligned}(T^{-1} \otimes I_m)(I_s \otimes I_m - h_n A \otimes J_n^l)(T \otimes I_m)(T^{-1} \otimes I_m)\Delta Y_n^l \\ = (T^{-1} \otimes I_m)(e \otimes y_n + h_n A \otimes F_n^l - Y_n^l) \\ = (T^{-1} \otimes I_m)(e \otimes y_n + h_n(A \otimes I_m)(T \otimes I_m)(T^{-1} \otimes I_m)F_n^l - Y_n^l),\end{aligned}$$

which is equivalent to

$$(I_s \otimes I_m - h_n B \otimes J_n^l)\Delta \hat{Y}_n^l = \hat{e} \otimes y_n + h_n B \otimes \hat{F}_n^l - \hat{Y}_n^l, \tag{6.5}$$

where $B = T^{-1}AT$, $\hat{e} = T^{-1}e$, $\hat{F}_n^l = (T^{-1} \otimes I_m)F_n^l$, $\hat{Y}_n^l = (T^{-1} \otimes I_m)Y_n^l$ and $\Delta \hat{Y}_n^l = \hat{Y}_n^{l+1} - \hat{Y}_n^l$.

For our similarity transformation, we choose T to transforms A to Jordan Canonical Form:

$$B = T^{-1}AT = \begin{pmatrix} \gamma_1 & 0 & 0 & \dots & 0 \\ \mu_2 & \gamma_2 & 0 & \dots & 0 \\ 0 & \mu_3 & \gamma_3 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & \mu_s & \gamma_s \end{pmatrix}$$

where all equal eigenvalues are grouped together, $\mu_i = 0$ if $\gamma_{i-1} \neq \gamma_i$, and μ_i is either 0 or γ_i if $\gamma_{i-1} = \gamma_i$. (Note that this form of the Jordan Canonical Form with μ_i either 0 or γ_i is valid because the nonsingularity of A ensures that $\gamma_i \neq 0$.) Since T, B and $\hat{e}$ all depend on A only, they can be precomputed and thus regarded as alternative coefficients of the formula.

Because of the structure of B, (6.5) can be rewritten as s systems, each of m equations:

$$\begin{aligned}(I_m - h_n\gamma_i J_n^l)(\Delta \hat{Y}_{n,i}^l + \frac{\mu_i}{\gamma_i}\Delta \hat{Y}_{n,i-1}^l) \\ = \hat{e}_i y_n + h_n\gamma_i(F_{n,i}^l + \frac{\mu_i}{\gamma_i}F_{n,i-1}^l) - \hat{Y}_{n,i}^l + \frac{\mu_i}{\gamma_i}\Delta \hat{Y}_{n,i-1}^l\end{aligned} \tag{6.6}$$

where we have introduced $\mu_1 = 0$ to avoid the need to treat the first equation as a special case. Thus, the problem of solving one $ms \times ms$ system in (6.3) has been reduced to that of solving s $m \times m$ systems. The potential benefit from this reduction in a parallel computing environment is even greater than in a sequential computing setting, where it has long been recognized as significant.

The family of SIRK formulas introduced by Burrage [**3**] and Butcher [**9**] and implemented in STRIDE [**7**] has $\gamma_i = \gamma$ for $i = 1, \dots, s$ and $\mu_i = \gamma$ for $i = 2, \dots, s$, where $\gamma \in \mathbb{R}$ is nonzero. A major advantage of these schemes is that the matrix $I_m - h_n\gamma J_n^l$ on the left side of (6.6) is common to all s systems. Therefore, only one $m \times m$ matrix factorization is required to solve all s systems in (6.6), resulting in a significant computational saving on sequential machines. In a parallel computing environment, s distinct matrices $I_m - h_n\gamma_i J_n^l$ could all be factored simultaneously, so the advantage enjoyed by formulas for which all γ_i are equal is arguably not as great. However, even for parallel machines, having all γ_i equal remains a computationally attractive property, since one might for example

(1) use several processors to factor $I_m - h_n\gamma J_n^l$ in parallel,

(2) employ the processors not involved with factoring $I_m - h_n\gamma J_n^l$ to do some other useful work (possibly on another problem or even for another user), or

(3) as noted in [**26**], use one extra processor to continually refactor and updated $I_m - h_n\gamma J_n^l$ while the others proceed with the integration using the most recently factored $I_m - h_n\gamma J_n^l$ in (6.6).

The SIRK formulas in STRIDE have a serious disadvantage from a parallel computing perspective: because $\mu_i = \gamma \neq 0$ for $i = 2, \ldots, s$, the solution to the $i - 1^{\mathbf{st}}$ equation, $\Delta\hat{Y}_{n,i-1}^l$, appears on the right side of the $i^{\mathbf{th}}$ equation, for $i = 2, \ldots, s$. Thus, all s equations cannot be solved simultaneously, although a fast recurrence approach, such as that outlined in [**20**], might be used to solve the s systems more quickly than simply calculating the $\Delta\hat{Y}_{n,i}^l$ one after each other in the obvious sequential manner. A similar difficulty for parallel implementations arises whenever a sequence of one or more μ_i's are nonzero, with the disadvantage becoming more severe as the length of the sequence increases. (For either a parallel or sequential implementation, another minor disadvantage associated with having $\mu_i \neq 0$ is that more work is required to form the right side of (6.6).)

To avoid this sequential bottleneck in parallel FIRK implementations, several authors have considered FIRK formulas for which A has s distinct eigenvalues, thus ensuring that all $\mu_i = 0$ and consequently that the s systems in (6.6) are independent, allowing them to be solved completely in parallel. However, these methods have the disadvantage that the coefficient matrices, $I_m - h_n\gamma_i J_n^l$, on the left side of (6.6) are all distinct. Consequently, to solve these linear systems, either s $m \times m$ distinct matrices must be factored or a common approximate matrix might be used on the left side of some of the systems in (6.6), possibly with an iterative method to compensate for this simplification.

However, before abandoning SIRK formulas for parallel IVP codes, it seems natural to ask if we could not have the best of both worlds: a SIRK formula with all $\mu_i = 0$ — or at least no long sequences of nonzero μ_i's. Although this is possible, the next result shows that this condition puts such a severe restriction on the order of a SIRK formula that the scheme is unattractive from a computational point-of-view.

THEOREM 6.1. *If the largest Jordan Block associated with the coefficient matrix A of a SIRK formula is of size r, then the order of the formula is at most $r + 1$.*

REMARK. An equivalent hypothesis, more closely related to the discussion above, is that the longest sequence of nonzero μ_i's associated with the transformed coefficient matrix $B = T^{-1}AT$ of a SIRK formula is of length $r - 1$.

PROOF: The minimum polynomial associated with the coefficient matrix A of the SIRK formula is $m(x) = (x - \gamma)^r$, where $\gamma \in \mathbb{R}$ is the single eigenvalue of A. Therefore, it follows immediately from Theorem 2.1 that the maximum order of the formula is $r + 1$. ∎

This negative result leads us back to considering FIRK formulas for which the coefficient matrix A has several distinct eigenvalues. Theorem 2.1 can be used to relate various desirable properties for the parallel solution of (6.6) to the maximal order of such formulas. We state just one more result of this nature below. Since it, like the previous theorem, is an immediate consequence of Theorem 2.1, we omit the proof.

THEOREM 6.2. *If the coefficient matrix A of a RK formula has r_1 distinct real eigenvalues, r_2 distinct complex (nonreal) eigenvalues, and A is diagonalizable, then the order of the* *formula is at most $r_1 + 2r_2 + \delta$, where $\delta = 1$ if one of the real eigenvalues is nonzero and $\delta = 0$ otherwise.*

We refer below to an IRK scheme based on collocation as an IRKC formula. Also, we call a RK scheme having a coefficient matrix A with real eigenvalues only a *real* RK formula, or, more specifically, a *real* IRK, FIRK or IRKC formula, as the case may be.

If r_2 of the eigenvalues of A are complex, then the corresponding r_2 linear systems in (6.6) are complex also. There are several possible ways of coping with this, the two most obvious being the use of complex arithmetic in the r_2 complex linear systems, or a modification of the reduction scheme so that $r_2/2$ real 2×2 blocks replace the $r_2/2$ complex conjugate pairs of eigenvalues in B, giving rise to $r_2/2$ linear systems similar to (6.6), but each containing $2m$, rather than m, equations. All the adaptions known to us to accommodate complex eigenvalues in A lead to an increase in the computational work required to solve (6.3). Also, one of the main advantages to be obtained by allowing A to have complex eigenvalues is that the order of a s-stage formula can be increased beyond $s+1$, the maximum possible for a s-stage real RK formula [**44**]. However, the *stage order* of a s-stage RK formula is at most s, and this is obtained for IRKC formulas only. For a large class of stiff problems, the "observed order" of a scheme is at most one more than its stage order, due to the *order reduction phenomenon* (see [**13**, Ch. 7]). Consequently, this advantage of increased order may be lost. Due in part to these two observations, most preliminary research to date on the parallel implementation of FIRK formulas has focused on real FIRK schemes, and real IRKC schemes in particular.

As noted above, the maximal order of a real RK formula is $s+1$. If in addition, we require that A be diagonalizable, then Theorem 6.2 restricts the order further to $r+1$, where r is the number of distinct real eigenvalues of A. Thus, all eigenvalues must be distinct if a real RK formula with a diagonalizable A is it to attain its maximal order of $s+1$. As the discussion below indicates, formulas that attain this maximal order can be constructed easily.

To this end, we consider the stability function $R(z) = P(z)/Q(z)$ of a RK formula given in (2.5). Since we assume throughout this section that the coefficients b and A of the RK formula are real, P and Q are both real polynomials of degree at most s, and $Q(z) = \prod_{i=1}^{s}(1 - \gamma_i z)$ where $\{\gamma_i\}$ are the eigenvalues of A.

As noted in §2, it may happen that the order $\hat{\nu}$ of the stability function $R(z)$ when considered as a rational approximation to e^z is greater than the order ν of its associated RK formula. However, Hairer and Türke [**22**] show that, if $R(z) = P(z)/Q(z)$ is an irreducible A-acceptable approximation to e^z of order $\hat{\nu} \geq 1$ with $\deg(P) \leq \deg(Q) = s$, then there is a s-stage B-stable RK formula of order $\nu = \hat{\nu}$ having R as its stability function. Although not stated explicitly in [**22**], the argument presented therein can be extended easily to show also that, if $R(z) = P(z)/Q(z)$ is an irreducible approximation to e^z of order $\hat{\nu} \geq 1$ with $s = \max(\deg(P), \deg(Q))$, then there is a s-stage RK formula of order $\nu = \hat{\nu}$ having R as its stability function.

Because of the close connection between rational approximations to e^z and RK formulas, the former have been studied extensively. One useful tool for this analysis is the *C-polynomial theory* of Nørsett [**41**] and its extension by Nørsett and Wanner [**43**]. We formulate our results below in terms of the C-polynomial $N(x)$ of [**43**] rather than the $p(x)$ of the original paper [**41**]. As noted in [**43**], $N(x) = (-1)^s p(1-x)$, so it is simple to translate results from one formulation to the other. However, several important results are more naturally stated in terms of $N(x)$.

As the following result is a minor extension of Theorems 2.1 and 4.1 of [**41**] written in terms of $N(x)$ rather than $p(x)$, and it is closely related to Theorem 4 of [**43**], we state it without proof.

THEOREM 6.3. *$R(z) = P(z)/Q(z)$ is an approximation to e^z of order at least $s_P = \deg(P)$ and normalized so that $P(0) = Q(0) = 1$ iff there is a unique polynomial N of degree $s \geq \max(\deg(P), \deg(Q))$ normalized so that $N^{(s)}(x) = 1$ for which*

$$P(z) = \sum_{i=0}^{s_P} N^{(s-i)}(1) z^i \qquad \text{and} \qquad Q(z) = \sum_{i=0}^{s} N^{(s-i)}(0) z^i. \tag{6.7}$$

Moreover, the error associated with such a R is

$$\begin{aligned} e^z - R(z) &= \left(\sum_{i=s_P+1}^{s} N^{(s-i)}(1) z^i + z^{s+1} \int_0^1 N(x)\, e^{z(1-x)}\, dx \right) / Q(z) \\ &= \left(\sum_{i=s_P+1}^{\infty} N^{(s-i)}(1) z^i \right) / Q(z) \end{aligned}$$

where, for $i \geq 0$, $N^{(i)}(x) = d^i N(x)/dx^i$ and, for $i < 0$, $N^{(i)}(x)$ is defined recursively by

$$N^{(i)}(x) = \int_0^x N^{(i+1)}(t)\, dt.$$

Nørsett and Wanner [**43**] showed that $N(x)$ provides a connection between rational approximations to e^z and IRKC formulas. We restate the essence of their result without proof.

THEOREM 6.4. *Let $c_1, \ldots, c_s$ be s distinct real numbers. R is the stability function of the s-stage IRKC formula of order $\nu \geq s$ with nodes $c_1, \ldots, c_s$ iff $R(z) = P(z)/Q(z)$ is a rational approximation to e^z of order $\nu \geq s$ with P and Q given by (6.7) for $s_P = s$ and $N(x) = (x - c_1) \cdots (x - c_s)/s!$.*

It is a known, but possibly not well-known, result that all the principal truncation error terms of an IRKC formula of order ν have the form

$$e(t) = \frac{\omega(t)}{\nu!} \left(\frac{1}{\nu+1} - \sum_{i=1}^{s} b_i c_i^{\nu} \right) \tag{6.8}$$

where t is any tree of order $\rho(t) = \nu + 1$ and $\omega(t)$ is a nonzero rational constant that depends on the tree t, but not on the formula. This is noted by Burrage in [**6**, p. 9], but not proved there. As we know of no easily accessible proof of this result, we have included one in the Appendix of [**30**].

Because of this last result for IRKC formulas, it makes sense to say that the principal truncation error terms of one IRKC formula are smaller than those of another IRKC formula, since, if one principal truncation error term is smaller, all others are as well by the same constant of proportionality. Furthermore, let

(1) $R(z)$ be a rational approximation to e^z of order $\nu \geq s$,
(2) $N(x)$ be its associated C-polynomial,
(3) τ be the tree with one node only and $t_b = [_\nu\, \tau\,]_\nu$ be the tall branchless tree of order $\rho(t_b) = \nu + 1$, and
(4) $e(t_b)$ be the error term associated with the tall branchless tree t_b for the IRKC formula associated with $R(z)$.

Then the principal error term $N^{(s-\nu-1)}(1)$ for the rational approximation $R(z)$ is equal to the principal error term $e(t_b)$ for the IRKC formula. That is, $N^{(s-\nu-1)}(1) = e(t_b)$. Therefore, if $R_1(z)$ and $R_2(z)$ are two rational approximations to e^z of the same order $\nu \geq s$, $N_1(x)$ and $N_2(x)$ are their associated C-polynomials, and $e_1(t_b)$ and $e_2(t_b)$ are the error terms for the tall branchless tree t_b of order $\rho(t_b) = \nu + 1$ for their associated IRKC formulas, then $N_1^{(s-\nu-1)}(1) = e_1(t_b)$ and $N_2^{(s-\nu-1)}(1) = e_2(t_b)$. Consequently, for any tree t of order $\rho(t) = \nu + 1$, the corresponding principal error terms for the IRKC formulas satisfy $e_1(t)/e_2(t) = e_1(t_b)/e_2(t_b) = N_1^{(s-\nu-1)}(1)/N_2^{(s-\nu-1)}(1)$. Thus, if the principal error term $N_1^{(s-\nu-1)}(1)$ for $R_1(z)$ is smaller than the principal error term $N_2^{(s-\nu-1)}(1)$ for $R_2(z)$, then each principal error term $e_1(t)$ for the IRKC formula associated with $R_1(z)$ is smaller than the corresponding principal error term $e_2(t)$ for the IRKC formula associated with $R_2(z)$ by the same constant of proportionality.

Moreover, many of the $\omega(t)$ in (6.8) are the same. In particular, for $\nu = s$ and $\rho(t) = s+1$, all $\omega(t) = 1$, and, for $\nu = s + 1$ and $\rho(t) = s + 2$, either $\omega(t) = 1$ if $t = [t_1, \ldots, t_k]$ with $k \geq 2$ or $\omega(t) = -(s + 1)$ if $t = [t_1]$. As $\nu - s$ increases, the number of distinct $\omega(t)$ also increases, but it is always much less than the number of trees of order $\rho(t) = \nu + 1$.

An important application of Theorems 6.3 and 6.4 is the construction of s-stage IRKC formulas with either

(1) a coefficient matrix with predetermined eigenvalues, or equivalently
(2) a predetermined stability function,

subject to the restrictions that

(a) the stability function $R(z) = P(z)/Q(z)$ be of order at least s,
(b) $\max(\deg(P), \deg(Q)) \leq s$, and
(c) the roots $c_1, \ldots, c_s$ of the associated polynomial $N(x)$ are real and distinct.

In case (1), the order of the IRKC formula is at least s and, in case (2), its order is the same as that of the given stability function.

To see this, consider (1) first: that is, assume that we are given $\gamma_1, \ldots, \gamma_s$ and asked to construct a s-stage IRKC formula having a coefficient matrix A with eigenvalues $\gamma_1, \ldots, \gamma_s$. Form $Q(z) = \prod_{i=1}^{s}(1 - \gamma_i z) = \sum_{i=0}^{s} q_i z^i$ and let $N(x) = \sum_{i=0}^{s} q_i x^{s-i}/(s - i)!$. Determine $P(z)$ by the first equation in (6.7) with $s_P = s$. Since P, Q and N satisfy equations (6.7), it follows from Theorem 6.3 that $R(z) = P(z)/Q(z)$ is a rational approximation to e^z of order $\nu \geq s$. Therefore, we have reduced (1) to (2): in either case, we have a rational approximation $R(z) = P(z)/Q(z)$ to e^z of order $\nu \geq s$ for which $\max(\deg(P), \deg(Q)) \leq s$. Therefore, either by construction in case (1) or by Theorem 6.3 in case (2), we obtain an associated polynomial $N(x)$ of degree s normalized so that $N^{(s)}(x) = 1$ and satisfying

(6.7). If the roots $c_1, \dots, c_s$ of $N(x)$ are real and distinct, then, by Theorem 6.4, the s-stage IRKC formula with nodes $c_1, \dots, c_s$ is also of order $\nu \geq s$ and has $R(z) = P(z)/Q(z)$ as its stability function. This completes the argument for case (2). For case (1), note that $Q(z) = \prod_{i=1}^{s}(1 - \gamma_i z) = \det(I - zA)$. Therefore, $\gamma_1, \dots, \gamma_s$ are the eigenvalues of the coefficient matrix A of this IRKC formula.

Of restrictions (a)-(c) above, the only one that is problematic is (c): the roots $c_1, \dots, c_s$ of the associated polynomial $N(x)$ are real and distinct. It is easy to find examples of rational approximations to e^z for which the associated $N(x)$ does not have distinct real roots. For example, $R_1(z) = (1 + z + z^2)/(1 + z^2/2)$ and $R_2(z) = 1/(1 - z + z^2/2)$ are both $2^{\mathbf{nd}}$-order rational approximations to e^z. $N_1(x) = (x^2 + 1)/2$ and $N_2(x) = (x - 1)^2/2$ are the unique C-polynomials of degree two associated with $R_1(z)$ and $R_2(z)$, respectively. $N_1(x)$ has roots $\pm i$ and $N_2(x)$ has a double root at $x = 1$. Consequently, neither $R_1(z)$ nor $R_2(z)$ is associated with an IRKC formula with real coefficients.

However, if $Q(z) = \prod_{i=1}^{s}(1 - \gamma_i z)$ with all $\gamma_i \in \mathbb{R}$ and at most one $\gamma_i = 0$, then all roots of the associated $N(x)$ are real and distinct, and, consequently, there is a corresponding real IRKC formula. Moreover, the restriction that at most one $\gamma_i = 0$ cannot be relaxed, since, for an IRKC formula, the nodes $c_1, \dots, c_s$ must be distinct and the coefficient matrix A for the formula satisfies $A = CVSV^{-1}$, where $V = (c_i^{j-1})$ is the van der Monde matrix formed from $c_1, \dots, c_s$, $C = \operatorname{diag}(c_1, \dots, c_s)$ and $S = \operatorname{diag}(1, 1/2, \dots, 1/s)$. (See [**13**, p. 61], for example.) Since at most one $c_j = 0$, A can have at most one eigenvalue $\gamma_i = 0$. Also note that, if $\deg(Q) < s$, we can still write Q in the form $Q(z) = \prod_{i=1}^{s}(1 - \gamma_i z)$ by taking $s - \deg(Q)$ of the γ_i's to be zero. However, $\deg(Q)$ equals the number of nonzero γ_i's. Therefore, $\deg(Q)$ must be either s or $s - 1$ to satisfy the requirement that at most one $\gamma_i = 0$.

The result stated at the beginning of the last paragraph was developed in stages. Nørsett and Wanner [**43**] noted that all s roots of N are real if all s γ_i's are real. They did not, though, establish the distinctness of the roots. Bales, Karakashian and Serbin [**1**] extended the result of [**43**] by showing that all s roots of N are distinct if all s γ_i are real and nonzero. In addition, they showed that all s roots of N are positive if all s γ_i are positive. The following theorem, which augments these results, is a special case of a theorem from a forthcoming paper on extensions to the C-polynomial theory of Nørsett [**41**]. Its proof, which will be given there, uses an adaptation of the *bi-orthogonality theory* of Iserles and Nørsett [**27**] [**28**], and is quite different from that employed in [**1**] or [**43**].

THEOREM 6.5. *Let $Q(z) = \prod_{i=1}^{s}(1 - \gamma_i z)$, where all γ_i are real and can be divided into three disjoint groups: k_+ of the γ_i are positive, k_- of the γ_i are negative, and the remaining $k_0 = s - k_+ - k_-$ of the γ_i are 0. Then the roots of the associated C-polynomial $N(x)$ are all real and can be divided into three corresponding disjoint groups: k_+ distinct positive roots, k_- distinct negative roots, and a root of multiplicity k_0 at $x = 0$.*

REMARK. Although any of k_+, k_- or k_0 may be 0, each of them is ≥ 0 and $k_+ + k_- + k_0 = s$. Also, note the k_+ positive γ_i and k_- negative γ_i need not be distinct to ensure that the corresponding k_+ positive roots and k_- negative roots of N are distinct. In particular, by taking all the $\gamma_i = \gamma \in \mathbb{R} - \{0\}$, we get a singly-implicit real IRKC formula of order either s or $s + 1$, with order $s + 1$ occurring iff $N^{(-1)}(1) = 0$.

We summarize below several negative results for real IRKC formulas. First, as noted after Theorem 6.2, the order of a s-stage real RK formula can be at most $s+1$, whether or not it is an IRKC formula. Moreover, the real IRKC formulas of order $s+1$ with minimum principal truncation error terms are SIRK formulas, which follows from the corresponding result in [**44**] for rational approximations and the discussion surrounding (6.8) above. By Theorem 6.1, the Jordan Canonical Form of the coefficient matrix A associated with such a SIRK formula must contain one block only of size s, from which it follows that the associated transformed Newton equations (6.6) are completely coupled, making the formula less than ideal for parallel implementation. However, this last negative result is largely offset by the observation in [**34**] that, for the case $s = 2$ at least, there exist real IRKC formulas of order $s + 1$ having "quite distinct" γ_i's, but with principal truncation error terms very close to the minimum attained by a SIRK formula. That is, the principal truncation error terms when viewed as functions of the γ_i's seem to be quite flat near their minimum.

Second, Burrage [**4**] proved that a s-stage B-stable IRKC formula must be of order $2s-1$ or $2s$. This and the restriction quoted above that a real IRKC formula can be of order $s+1$ at most imply that these schemes can be B-stable for $s = 1$ or 2 only, and for $s = 2$ the order must be three while for $s = 1$ the order may be either one or two. It is easy to find examples of such formulas.

Third, Wanner, Hairer and Nørsett [**52**] showed that a s-stage real SIRK formula of order $s + 1$ can be A-stable for $s = 1, 2, 3, 5$ only. Keeling [**36**] generalized their result by showing that the same restriction applies to all real IRK formulas. It is well-known that A-stable s-stage SIRK formulas of order $s+1$ do exist for $s = 1, 2, 3, 5$, and that L-stable s-stage SIRK formulas of order s exist for $1 \leq s \leq 6$ and $s = 8$, but for no other $s \leq 15$. (See [**3**] for example.) For $s = 2$ and 3, Keeling [**36**] exhibits a family of s-stage B-stable IRK formulas of order $s + 1$, each having a coefficient matrix A with distinct real eigenvalues, and he claims that a similar family of formulas exists for $s = 5$. The 2-stage schemes are real IRKC formulas, but the others cannot be, as noted in the preceding paragraph. Nevertheless, Keeling's examples establish that s-stage A-stable IRKC formulas of order $s + 1$ for which A has distinct real eigenvalues do exist for $s = 3$ and 5 also, since, as outlined above, it is easy to construct IRKC formulas having the same stability functions as the formulas given in his examples.

There are, though, some positive results about A_0- and I-stability of real IRKC formulas. Bales, Karakashian and Serbin [**1**] showed that a s-stage real IRK formula of order s or $s + 1$ is A_0-stable — in fact *strongly* Ao-stable for $s \geq 3$ — if the eigenvalues $\{\gamma_i\}$ of the formula's coefficient matrix A satisfy $\gamma_i \geq 1/2$ for $i = 1, \ldots, s$. Keeling [**36**] provided an alternate proof of this theorem that also indicates, for any $s \geq 1$, how to construct a s-stage strongly A_0-stable IRKC formula of order $s + 1$ for which A has distinct real eigenvalues. The same paper shows in addition, for any even $s \geq 2$, how to construct a s-stage I-stable IRKC formula of order s for which A has distinct real eigenvalues.

Another set of positive results is due to Orel [**45**], who generalized the *real-pole sandwich* theory of Nørsett and Wanner [**43**] by relaxing the restriction that $\deg(P) = s$. As a result, he was able to find many more L-acceptable rational approximations of the form (6.7) with $\deg(P) = s_P < s$ having real poles only and order $s_P + 1$. This forms the basis of his conjecture that, for any s_P, there is a S such that for all $s \geq S$, there exist L-acceptable rational approximations with real poles only of the form (6.7) with

$\deg(P) = s_P$, $\deg(Q) = s$ and order $s_P + 1$. Such a rational approximation cannot be the stability function of an IRKC formula for $s_P < s - 1$, since this would require that $N^{(s-i)}(1) = 0$ for $i = s_P + 1, \ldots, s$, which contradicts Theorem 6.5. However, the more general result of Hairer and Türke [**22**] cited above ensures that we can associate a s-stage B-stable IRK formula with each such A-acceptable rational approximation.

Karakashian and Rust [**34**] present some numerical results for a 2-stage $3^{\mathbf{rd}}$-order $\mathrm{A}_{\mathbf{0}}$-stable IRK method based on the theory in [**1**]. The coefficient matrix A of this formula has two distinct real eigenvalues, so the Newton iteration (6.3) associated with the formula can be reduced to two completely independent systems of the form (6.6). Since their test problem is linear, the computation can be simplified further. They compared the CPU times of a simple fixed-stepsize implementation of this scheme running on one and two processor for two parallel machines, an IBM-3081D and a Cray XMP. Their results show that a speedup close to the optimal value of two is achievable with this simple implementation if the problem is sufficiently large.

ACKNOWLEDGEMENT. This paper has benefited from discussions with several people. We thank in particular Kevin Burrage, John Butcher and Bob Skeel as well as our students and colleagues.

References

1. L. Bales, O. Karakashian and S. Serbin, *On the A_0-acceptability of rational approximations to the exponential function with only real poles*, BIT **28** (1988), 70–79.
2. L. G. Birta and O. Abou-Rabia, *Parallel block predictor-corrector methods for ode's*, IEEE Trans. Comput. **C–36** (1987), 299–311.
3. K. Burrage, *A special family of Runge-Kutta methods for solving stiff differential equations*, BIT **18** (1978), 22–41.
4. ———, *High order algebraically stable Runge-Kutta methods*, BIT **18** (1978), 373–383.
5. ———, *Solving nonstiff IVPs in a transputer environment*, manuscript, CMSR, Univ. of Liverpool,, England.
6. ———, *The error behaviour of a general class of predictor-corrector methods*, manuscript, CMSR, Univ. of Liverpool,, England.
7. K. Burrage, J. C. Butcher and F. H. Chipman, *An implementation of singly-implicit Runge-Kutta methods*, BIT **20** (1980), 326–340.
8. J. C. Butcher, *On the implementation of implicit Runge-Kutta methods*, BIT **16** (1976), 237–240.
9. ———, *A transformed implicit Runge-Kutta method*, J. ACM **26** (1979), 731–738.
10. ———, "The Numerical Analysis of Ordinary Differential Equations," John Wiley & Sons, New York, 1987.
11. ———, *Towards efficient implementation of singly-implicit methods*, ACM Trans. Math. Softw. **14** (1988), 68–75.
12. M. T. Chu and H. Hamilton, *Parallel solution of ODEs by multi-block methods*, SIAM J. Sci. Stat. Comp. **8** (1987), 342–353.
13. K. Dekker and J. G. Verwer, "Stability of Runge-Kutta Methods for Stiff Nonlinear Differential Equation," CWI Monograph, North-Holland, Amsterdam, 1984.
14. R. Enenkel, "The implementation of parallel Runge-Kutta methods," M.Sc. Thesis, Computer Science Dept., University of Toronto, Toronto, Canada, 1988.
15. W. H. Enright and D. J. Higham, "Parallel defect control," Tech. Rep. 237/90, Computer Science Dept., Univ. of Toronto,, Toronto, Canada, 1990.
16. W. H. Enright, K. R. Jackson, S. P. Nørsett and P. G. Thomsen, *Interpolants for Runge-Kutta formulas*, ACM Trans. Math. Softw. **12** (1986), 193–218.

17. M. A. Franklin, *Parallel solution of ordinary differential equations*, IEEE Trans. Comp. **C-27** (1978), 413–420.
18. C. W. Gear, "The potential for parallelism in ordinary differential equations," Tech. Rep. UIUC-DCS–R–86–1246, Computer Science Dept., Univ. of Illinois at Urbana-Champaign, Urbana, IL, 1986.
19. ———, "Parallel methods in ordinary differential equations," Tech. Rep. UIUCDCS–R–87–1396, Computer Science Dept., Univ. of Illinois at Urbana-Champaign, Urbana, IL, 1987.
20. ———, "Massive parallelism across the method in ODEs," Tech. Rep. UIUCDCS–R–88–1442, Computer Science Dept., Univ. of Illinois at Urbana-Champaign, Urbana, IL, 1988.
21. E. Hairer, S. P. Nørsett and G. Wanner, "Solving Ordinary Differential Equations I, Nonstiff Problems," Springer-Verlag, Berlin, 1987.
22. E. Hairer and H. Türke, *The equivalence of B-stability and A-stability*, BIT **24** (1984), 520–528.
23. P. J. van der Houwen and B. P. Sommeijer, "Variable step iteration of high-order Runge-Kutta methods on parallel computers," Tech. Rep. NM–R8817, Dept. of Numerical Mathematics, Centre for Mathematics and Computer Science, Amsterdam, The Netherlands, 1988.
24. ———, "Block Runge-Kutta methods on parallel computers," Tech. Rep. NM–R8906, Dept. of Numerical Mathematics, Centre for Mathematics and Computer Science, Amsterdam, The Netherlands, 1989.
25. ———, *Parallel iteration of high-order Runge-Kutta methods with stepsize control*, J. Comput. Appl. Math. **29** (1990), 111–127.
26. P. J. van der Houwen, B. P. Sommeijer and W. Couzy, "Embedded diagonally implicit Runge-Kutta algorithms on parallel computers," Tech. Rep. NM–R8912, Dept. of Numerical Mathematics, Centre for Mathematics and Computer Science, Amsterdam, The Netherlands, 1989.
27. A. Iserles and S. P. Nørsett, *Bi-orthogonal polynomials*, in "Polynômes Orthogonaux et Applications," Lecture Notes in Mathematics #1171, Proceedings Bar-le-Duc, 1984, A. Dold and B. Eckmann eds., Springer-Verlag, Berlin, 1985, pp. 92–100.
28. ———, *On the theory of biorthogonal polynomials*, Trans. Amer. Math. Soc. **306** (1988), 455–474..
29. ———, "On the theory of parallel Runge-Kutta methods," Tech. Rep. DAMTP NA12 / 1888, Dept. of Applied Math and Theoretical Physics, Cambridge University, Cambridge, England, 1989.
30. K. R. Jackson and S. P. Nørsett, "The potential for parallelism in Runge-Kutta methods. Part 1: RK Formulas in Standard Form," Tech. Rep. No. 239/90, Computer Science Dept., University of Toronto, Toronto, Canada, 1990.
31. ———, *The potential for parallelism in Runge-Kutta methods. Part 2: RK predictor-corrector formulas*, in preparation.
32. J. Kalvenes, "Experimentation with parallel ODE-solvers," Tech. Rep. NM–R8912, Mathematical Sciences Div., Norwegian Institute of Technology, Trondheim, Norway, 1986.
33. O. A. Karakashian, *On Runge-Kutta methods for parabolic problems with time-dependent coefficients*, Math. Comp. **47** (1986), 77–101.
34. O. A. Karakashian and W. Rust, *On the parallel implementation of implicit Runge-Kutta methods*, SIAM J. Sci. Stat. Comput. **9** (1988), 1985–1090.
35. I. N. Katz, M. A. Franklin and A. Sen, *Optimally stable parallel predictors for Adams-Moulton correctors*, Comp. & Maths. with Appls. **3** (1977), 217–233.
36. S. L. Keeling, *On implicit Runge-Kutta methods with a stability function having distinct real poles*, BIT **29** (1989), 91–109.
37. W. Kutta,, *Beitrag zur näherungsweisen Integration totaler Differentialgleichungen*, Z. Math. Phys. **46** (1901), 435–453.
38. I. Lie, "Some aspects of parallel Runge-Kutta methods," Math. and Comp. Rep. 3/87, Numerical Mathematics Dept., Norwegian Institute of Technology, Trondheim, Norway, 1987.
39. W. L. Miranker, *A survey of parallelism in numerical analysis*, SIAM Review **13** (1971), 524–547.
40. W. L. Miranker and W. Liniger, *Parallel methods for the numerical integration of ordinary differential equations*, Math. Comp. **21** (1967), 303–320.
41. S. P. Nørsett, *C-polynomials for rational approximation to the exponential function*, Numer. Math. **25** (1975), 39–56.

42. S. P. Nørsett and H. H. Simonsen, *Aspects of parallel Runge-Kutta methods*, in "Numerical Methods for Ordinary Differential Equations," Lecture Notes in Mathematics #1386, Proceedings of the l'Aquila Symposium, 1987, A. Bellen, C. W. Gear and E. Russo eds., Springer-Verlag, Berlin, 1989, pp. 103–107.
43. S. P. Nørsett and G. Wanner, *The real-pole sandwich for rational approximations and oscillation equations*, BIT **19** (1979), 79–94.
44. S. P. Nørsett and A. Wolfbrandt, *Attainable order of rational approximations to the exponential function with only real poles*, BIT **17** (1977), 200–208.
45. B. Orel, "Real pole approximations to the exponential function," Tech. Rep. 1/90, Mathematical Sciences Div., Norwegian Institute of Technology, Trondheim, Norway, 1989.
46. B. Owren, "Continuous explicit Runge-Kutta methods with applications to ordinary and delay differential equations," Doktor Ingeniøravhandling 1989:58, Mathematical Sciences Div., Norwegian Institute of Technology, Trondheim, Norway, 1989.
47. B. Owren and M. Zennaro, *Continuous explicit Runge-Kutta methods*, in "to appear in Proceedings of the London 1989 Conference on Computational ODEs,," 1989.
48. ——————, "Order barriers for continuous explicit Runge-Kutta methods,," Tech. Rep. 2/89,, Mathematical Sciences Div., Norwegian Institute of Technology, Trondheim, Norway, 1989. submitted to Math. Comp.
49. R. D. Skeel and H.-W. Tam, *Potential for parallelism in explicit linear methods*, manuscript, Computer Science Dept., Univ. of Illinois at Urbana-Champaign, Urbana, IL.
50. H.-W. Tam, "Parallel methods for the numerical solution of ordinary differential equations," Ph.D. Thesis, Tech. Rep. UIUCDCS–R–86–1246, Computer Science Dept., Univ. of Illinois at Urbana--Champaign, Urbana, IL, 1989.
51. J. M. Varah, *On the efficient implementation of implicit Runge-Kutta methods*, Math. Comp. **33** (1979), 557–561.
52. G. Wanner, E. Hairer and S. P. Nørsett,, *Order stars and stability theorems,*, BIT **18** (1978), 475–489.
53. P. B. Worland, *Parallel methods for the numerical solution of ordinary differential equations*, IEEE Trans. Comp. **C-25** (1976), 1045–1048.

Appendix. Principal Error Terms for IRKC Formulas. In this appendix, we prove the result associated with equation (6.8) for the principal error terms of an Implicit Runge-Kutta Collocation (IRKC) formula. As noted in §6, this result is stated by Burrage in [**6**, p. 9], but not proved there. As we know of no easily accessible proof of this result, we have included one below. We also note at the end of this section that a few closely related results mentioned in §6 follow from our proof.

THEOREM A.1. *Each principal error term of an IRKC formula of order ν satisfies*

$$e(t) = \frac{\omega(t)}{\nu!}\left(\frac{1}{\nu+1} - \sum_{i=1}^{s} b_i c_i^{\nu}\right)$$

where t is any tree of order $\rho(t) = \nu + 1$ and $\omega(t)$ is a nonzero rational constant that depends on the tree t, but not on the coefficients of the formula.

PROOF: A s-stage IRKC formula of order $\nu = s + r$, $0 \le r \le s$, satisfies the Butcher conditions

$$\begin{aligned} B(s+r): &\qquad \sum_{i=1}^{s} b_i c_i^{k-1} = 1/k, &&\qquad k = 1, \ldots, s+r, \\ C(s): &\qquad \sum_{j=1}^{s} a_{ij} c_j^{k-1} = c_i^k/k, &&\qquad i = 1, \ldots, s, \quad k = 1, \ldots, s, \end{aligned}$$

which may be rewritten in matrix form as

$$\begin{aligned} B(s+r): &\qquad b^T c^{k-1} = 1/k, &&\qquad k = 1, \ldots, s+r, \\ C(s): &\qquad A c^{k-1} = c^k/k, &&\qquad k = 1, \ldots, s, \end{aligned}$$

where $c^k = (c_1^k, \ldots, c_s^k)^T$. Since all the c_i's are distinct, $B(s+r)$ and $C(s)$ imply that the formula also satisfies

$$D(r): \qquad \sum_{i=1}^{s} b_i c_i^{k-1} a_{ij} = \frac{1}{k} b_j (1 - c_j^k), \qquad j = 1, \ldots, s, \quad k = 1, \ldots, r.$$

(See for example [**13**, Theorem 3.2.4].)

For convenience, we augment A by adding to it a $s+1^{\text{st}}$ row consisting of b^T in positions $1, \ldots, s$ and a $s+1^{\text{st}}$ column consisting of $s+1$ zeros. We also augment b and c by adding to b a $s+1^{\text{st}}$ component $b_{s+1} = 0$ and adding to c a $s+1^{\text{st}}$ component $c_{s+1} = 1$. The vector ϕ for the Butcher series for the formula consists of $s+1$ components, with the error in the formula associated with any tree t being $e(t) = (1 - \phi_{s+1}(t))/\rho(t)!$.

We use below the standard bracket notation for trees. If $t_1, \ldots, t_m$ are trees, then $t = [t_1, \ldots, t_m]$ is the tree formed by adding a new root and joining it by a new edge to the root of each of $t_1, \ldots, t_m$. (See for example [**21**, p. 152]).

Although not explicitly stated in [**21**], it follows immediately from Corollary 11.7 therein that, for any tree $t = [t_1, \ldots, t_m]$,

$$\phi(t) = \rho(t) A \phi(t_1) \cdots \phi(t_m). \tag{A.1}$$

Using (A.1) together with $C(s)$, it is easy to show by induction on $\rho(t)$ that

$$\phi(t) = c^{\rho(t)} \quad \text{for} \quad \rho(t) \le s. \tag{A.2}$$

For the remainder of the proof, assume that $\rho(t) = \nu + 1 = s + r + 1$, $0 \le r \le s$. That is, $e(t)$ is a principal error term for the formula. Either

(I) $t = [t_1, \ldots, t_m]$ with all $\rho(t_i) \le s$, or

(II) $t = [t_1, \ldots, t_m]$ with one $\rho(t_i) > s$ and all other $\rho(t_j) \le s$,

since $1 + \rho(t_1) + \cdots + \rho(t_m) = \rho(t) \le 2s + 1$.

For case (I),

$$\phi(t) = \rho(t) A \phi(t_1) \ldots \phi(t_m) = \rho(t) A c^{\rho(t)-1},$$

whence

$$\phi_{s+1}(t) = \rho(t) b^T c^{\rho(t)-1} = (\nu + 1) b^T c^\nu$$

and

$$e(t) = \frac{1 - \phi_{s+1}(t)}{\rho(t)!} = \frac{1}{\nu!} \left(\frac{1}{\nu + 1} - b^T c^\nu \right)$$

as required with $\omega(t) = 1$.

For case (II), assume without loss of generality that the t_i for which $\rho(t_i) > s$ is t_1. Therefore, $\rho(t_i) \le s$ for $i = 2, \ldots, m$. Consequently, by (A.1) and (A.2),

$$\phi(t) = \rho(t) A \phi(t_1) c^{\rho(t)-\rho(t_1)-1}. \tag{A.3}$$

We need to apply rule (A.3) at most r times to arrive at a formula containing ρ's, A's and c's only. To see this, note that, each time we apply rule (A.3), the order of the tree taking the place of t_1 in the recurrence decreases by at least 1 and we don't need to apply rule (A.3) to determine $\phi(t_1)$ for $\rho(t_1) \le s + 1$, since, in this case, $\phi(t_1)$ must be $\rho(t_1) A c^{\rho(t_1)-1}$ by (I).

The remainder of the proof is an induction on the number of times that we need to apply rule (A.3) before we arrive at a formula containing ρ's, A's and c's only.

If we need to apply rule (A.3) just once, then

$$\phi(t_1) = \rho(t_1) A c^{\rho(t_1)-1}.$$

Therefore,

$$\phi(t) = \rho(t) \rho(t_1) A \left(A c^{\rho(t_1)-1} \right) c^{\rho(t)-\rho(t_1)-1},$$

whence

$$\begin{aligned}
\phi_{s+1}(t) &= \rho(t) \rho(t_1) \sum_{i=1}^{s} b_i \left(\sum_{j=1}^{s} a_{ij} c_j^{\rho(t_1)-1} \right) c_i^{\rho(t)-\rho(t_1)-1} \\
&= \rho(t) \rho(t_1) \sum_{j=1}^{s} \left(\sum_{i=1}^{s} b_i a_{ij} c_i^{\rho(t)-\rho(t_1)-1} \right) c_j^{\rho(t_1)-1}.
\end{aligned}$$

Since $\rho(t) - \rho(t_1) \le r$, we can apply $D(r)$ to the sum in brackets to get

$$\begin{aligned}
\phi_{s+1}(t) &= \frac{\rho(t)\rho(t_1)}{\rho(t)-\rho(t_1)} \sum_{j=1}^{s} b_j \left(1 - c_j^{\rho(t)-\rho(t_1)}\right) c_j^{\rho(t_1)-1} \\
&= \frac{\rho(t)\rho(t_1)}{\rho(t)-\rho(t_1)} \left(\sum_{i=1}^{s} b_i c_i^{\rho(t_1)-1} - \sum_{i=1}^{s} b_i c_i^{\rho(t)-1} \right)
\end{aligned}$$

and finally $B(s+r)$ to the first sum to get

$$\begin{aligned}
\phi_{s+1}(t) &= \frac{\rho(t)\rho(t_1)}{\rho(t)-\rho(t_1)} \left(\frac{1}{\rho(t_1)} - b^T c^{\rho(t)-1} \right) \\
&= \frac{\rho(t_1)}{\rho(t)-\rho(t_1)} \left(\frac{\rho(t)}{\rho(t_1)} - \rho(t) b^T c^{\rho(t)-1} \right) \\
&= \frac{\rho(t_1)}{\rho(t)-\rho(t_1)} \left(1 + \frac{\rho(t)-\rho(t_1)}{\rho(t_1)} - \rho(t) b^T c^{\rho(t)-1} \right) \\
&= 1 + \frac{\rho(t_1)}{\rho(t)-\rho(t_1)} (1 - \rho(t) b^T c^{\rho(t)-1}) \\
&= 1 + \frac{\rho(t_1)}{\rho(t)-\rho(t_1)} (1 - (\nu+1) b^T c^{\nu}).
\end{aligned}$$

Consequently,

$$e(t) = \frac{1-\phi_{s+1}(t)}{(\nu+1)!} = \frac{\omega(t)}{\nu!} \left(\frac{1}{\nu+1} - b^T c^{\nu} \right) \quad \text{where} \quad \omega(t) = \frac{-\rho(t_1)}{\rho(t)-\rho(t_1)}$$

as required.

For the induction step, assume that, if we need to apply rule (A.3) at most k times before we arrive at a formula containing ρ's, A's and c's only, then

$$e(t) = \frac{\omega(t)}{\nu!} \left(\frac{1}{\nu+1} - b^T c^{\nu} \right)$$

where $\omega(t)$ is a nonzero rational constant that depends on the tree t, but not on the coefficients of the formula. Now consider the case that, for tree t, we need to apply rule (A.3) $k+1 \ge 2$ times. Thus,

$$\begin{aligned}
\phi(t) &= \rho(t) A \phi(t_1) c^{\rho(t)-\rho(t_1)-1} \\
&= \rho(t)\rho(t_1) A \left(A\phi(t_2) c^{\rho(t_1)-\rho(t_2)-1} \right) c^{\rho(t)-\rho(t_1)-1},
\end{aligned}$$

whence

$$\begin{aligned}
\phi_{s+1}(t) &= \rho(t)\rho(t_1) \sum_{i=1}^{s} b_i \left(\sum_{j=1}^{s} a_{ij} \phi_j(t_2) c_j^{\rho(t_1)-\rho(t_2)-1} \right) c_i^{\rho(t)-\rho(t_1)-1} \\
&= \rho(t)\rho(t_1) \sum_{j=1}^{s} \left(\sum_{i=1}^{s} b_i a_{ij} c_i^{\rho(t)-\rho(t_1)-1} \right) \phi_j(t_2) c_j^{\rho(t_1)-\rho(t_2)-1}.
\end{aligned}$$

Since $\rho(t) - \rho(t_1) \le r$, we can apply $D(r)$ to the term in brackets to get

$$\phi_{s+1}(t) = \frac{\rho(t)\rho(t_1)}{\rho(t)-\rho(t_1)} \sum_{j=1}^{s} b_j \left(1 - c_j^{\rho(t)-\rho(t_1)}\right) \phi_j(t_2) c_j^{\rho(t_1)-\rho(t_2)-1}$$

$$= \frac{\rho(t)\rho(t_1)}{\rho(t)-\rho(t_1)} \left(\sum_{i=1}^{s} b_i \phi_i(t_2) c_i^{\rho(t_1)-\rho(t_2)-1} - \sum_{i=1}^{s} b_i \phi_i(t_2) c_i^{\rho(t)-\rho(t_2)-1} \right). \tag{A.4}$$

Consider $\hat{t} = [t_2, \tau^{\rho(t_1)-\rho(t_2)-1}]$, where τ^l represents l trees each with one node. Since, $\rho(\hat{t}) = \rho(t_1) < \rho(t) = \nu + 1$,

$$\phi_{s+1}(\hat{t}) = \rho(t_1) \sum_{i=1}^{s} b_i \phi_i(t_2) c_i^{\rho(t_1)-\rho(t_2)-1} = 1. \tag{A.5}$$

Similarly, for $\tilde{t} = [t_2, \tau^{\rho(t)-\rho(t_2)-1}]$, $\rho(\tilde{t}) = \rho(t) = \nu + 1$, whence

$$\phi_{s+1}(\tilde{t}) = \rho(t) \sum_{i=1}^{s} b_i \phi_i(t_2) c_i^{\rho(t)-\rho(t_2)-1}.$$

Because of the relationship between t and $\tilde{t}$ involving t_1 and t_2, k applications of rule (A.3) to $\tilde{t}$ are sufficient to yield a formula containing ρ's, A's and c's only. Therefore, by the induction hypothesis,

$$e(\tilde{t}) = \frac{1 - \phi_{s+1}(\tilde{t})}{(\nu+1)!} = \frac{\omega(\tilde{t})}{\nu!} \left(\frac{1}{\nu+1} - b^T c^\nu \right),$$

where $\omega(\tilde{t})$ is a nonzero rational constant that depends on the tree $\tilde{t}$, but not on the coefficients of the formula. Hence,

$$\phi_{s+1}(\tilde{t}) = \rho(t) \sum_{i=1}^{s} b_i \phi_i(t_2) c_i^{\rho(t)-\rho(t_2)-1} = 1 - \omega(\tilde{t})(1 - (\nu+1) b^T c^\nu). \tag{A.6}$$

Substituting (A.5) and (A.6) into (A.4), we get

$$\begin{aligned}
\phi_{s+1}(t) &= \frac{\rho(t)\rho(t_1)}{\rho(t)-\rho(t_1)} \left(\frac{1}{\rho(t_1)} - \frac{1}{\rho(t)} [1 - \omega(\tilde{t})(1 - (\nu+1) b^T c^\nu)] \right) \\
&= \frac{\rho(t)}{\rho(t)-\rho(t_1)} - \frac{\rho(t_1)}{\rho(t)-\rho(t_1)} + \frac{\rho(t_1)}{\rho(t)-\rho(t_1)} \omega(\tilde{t})(1 - (\nu+1) b^T c^\nu) \\
&= 1 + \frac{\rho(t_1)}{\rho(t)-\rho(t_1)} \omega(\tilde{t})(1 - (\nu+1) b^T c^\nu),
\end{aligned}$$

from which it follows that

$$e(t) = \frac{1 - \phi_{s+1}(t)}{(\nu+1)!} = \frac{\omega(t)}{\nu!} \left(\frac{1}{\nu+1} - b^T c^\nu \right) \quad \text{where} \quad \omega(t) = -\omega(\tilde{t}) \frac{\rho(t_1)}{\rho(t)-\rho(t_1)}.$$

Moreover, $\omega(t)$ inherits from $\omega(\tilde{t})$ the property of being a nonzero rational constant that depends on the tree t, but not on the coefficients of the formula. ∎

A few additional observations are worth noting. We see from the proof above that, for a s-stage IRKC formula of order $\nu = s$, $\omega(t) = 1$ for all trees t satisfying $\rho(t) = s + 1$, since we can apply rule (I) to determine all associated $\phi_{s+1}(t)$. Similarly, for $\nu = s+1$ and $\rho(t) = s + 2$, either

(1) $t = [t_1, \ldots, t_k]$ with $k \geq 2$ and all $\rho(t_i) \leq s$, from which it follows that $\omega(t) = 1$, or
(2) $t = [t_1]$ with $\rho(t_1) = s + 1$, from which it follows that $\omega(t) = -(s + 1)$.

As $\nu - s$ increases, the number of distinct $\omega(t)$'s also increases, but this number is always much less than the number of trees of order $\rho(t) = \nu + 1$.

It also follows from the relations in the proof above and a straightforward induction argument that, for a s-stage IRKC formula of order $\nu = s + r$ with $0 \leq r \leq s$, $\omega(t_b) = (-1)^r\,(s+r) \cdots (s+1)/r!$, where $t_b = [_\nu\, \tau\,]_\nu$ is the tall branchless tree of order $\rho(t_b) = \nu + 1$ and τ is the tree with one node only. Equivalently,

$$e(t_b) = \frac{(-1)^r}{s!\,r!}\left(\frac{1}{s+r+1} - b^T c^{s+r}\right).$$

Furthermore, as noted in §6, $e(t_b) = N^{(-r-1)}(1)$, where $N(x)$ is the C-polynomial associated with the IRKC formula. It is easy to verify directly that

$$N^{(-r-1)}(1) = \frac{(-1)^r}{s!\,r!}\left(\frac{1}{s+r+1} - b^T c^{s+r}\right)$$

as required.

Keywords. Numerical Solution, Initial Value Problem, IVP, Ordinary Differential Equation, ODE, Runge-Kutta Method, Parallel Computation.
1980 *Mathematics subject classifications*: 65L05

Professor K. R. Jackson, Computer Science Dept., University of Toronto, Toronto, Ontario, Canada M5S 1A4. E-mail: krj@na.toronto.edu
Professor S. P. Nørsett, Division of Mathematical Sciences, Norwegian Institute of Technology, N–7034 Trondheim–NTH, Norway. E-mail: norsett@imf.unit.no