The Centre for Mathematics and Computer Science is a research institute of the Stichting Mathematisch Centrum, which was founded on February 11, 1946, as a nonprofit institution aiming at the promotion of mathematics, computer science, and their applications. It is sponsored by the Dutch Government through the Netherlands Organization for the Advancement of Research (N.W.O.).

Copyright © Stichting Mathematisch Centrum, Amsterdam
Block Runge-Kutta Methods

P.J. van der Houwen

CWI: The Centre for Mathematics and Computer Science
Post box 4079, 1009 AB Amsterdam, The Netherlands

In this paper block methods for solving ODEs on parallel computers are constructed. Most block methods found in the literature produce approximations to the exact solution at equidistant points. Here, we allow that the approximations correspond to nonequidistant points like the intermediate approximations computed in Runge-Kutta methods. This approach enables us to improve the order of accuracy. We concentrate on explicit methods such that they are suitable for use on parallel computers.

1980 Mathematics Subject Classification: 65M10, 65M20
Key Words and Phrases: numerical analysis, stability, block Runge-Kutta methods, parallelism.

1. Introduction

The work surveyed in this paper is joint work with B.P. Sommeijer and W. Couzy from the Centre for Mathematics and Computer Science in Amsterdam. Full details may be found in the references [7]-[11].

Our starting point is the conventional s-stage RK method for the initial-value problem

\begin{equation}
\frac{dy(t)}{dt} = f(y(t)), \quad y(t_0) = y_0,
\end{equation}

defined by

\begin{align}
Y^{(i)} &= y_n + h \sum_{j=1}^{s} a_{ij} f(Y^{(j)}), \quad i = 1, \ldots, s; \\
Y_{n+1} &= y_n + h \sum_{j=1}^{s} b_j f(Y^{(j)}), \quad n = 0, 1, \ldots.
\end{align}

The general structure of the Runge-Kutta-type methods considered in this paper is a direct generalization of this conventional method. We introduce k-dimensional block vectors \( Y_n \), the components of which are numerical approximations to the exact solution values:

\[ Y_{n+1} := (y_{n,1}, y_{n,2}, \ldots, y_{n,k})^T, \]

where \( y_{n,j} \) denotes a numerical approximation to the exact solution value \( y(t_n+cjh) \). We shall assume that \( c_k=1 \), while the other values of \( c_j \) are allowed to be any real number. Thus, the last component of the block vector \( Y_{n+1} \) always provides an approximation to \( y(t_{n+1}) \). The vector \( c:=(c_i) \) will be called the block point vector. For scalar ODEs, we now define the \( s \)-stage block RK (BRK) method

Report NM-R89/3
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands
where the matrices $A_i$, $A_{ij}$, $B_0$ and $B_j$ respectively are r-by-k, r-by-r, k-by-k and k-by-r matrices, $e_k$ denotes the kth unit vector, and where we use the convention that for any given vector $v=(v_j)$, $f(v)$ denotes the vector with entries $f(v_j)$. The method (1.3) can be considered as the block analogue of (1.2). It is straightforwardly extended to systems of ODEs and therefore also to nonautonomous equations. In order to start the method, one needs the initial vector $Y_0$, which requires as many starting values as there are distinct values $c_j$ (j=1,...,k). If $c=e$, e being the vector with unit entries, then only one starting value is needed. In fact, in this case, the BRK method reduces to a one-step RK method providing k approximations to $y(t_{n+1})$. We define the order of BRK methods by the order of $y_{n+1}$.

The method is explicit if the matrices $A_{ij}$ vanish for $j \geq i$, diagonally-implicit if the matrices $A_{ij}$ vanish for $j > i$ and if the matrices $A_{ii}$ are diagonal, and implicit otherwise. In this survey, we restrict our considerations to explicit and diagonally implicit methods. For such methods, the $r$ components of the block vectors $Y^{(i)}$ can be computed in parallel. Hence, if $r$ processors are available, then the required computational time per integration step is at most the time needed for computing $s$ block components sequentially. We define the optimal number of processors as the number of processors for which the number of (sequential) block component evaluations per step is minimal. In this connection, we remark that often less than $r$ processors are needed for implementing the BRK method.

In the explicit and diagonally-implicit case, the representation (1.3) is very convenient for implementing the method on a computer, because the actual code is a direct translation of the scheme (1.3) and the instructions for the computer in order to exploit the built-in parallelism of the method are obvious. Conversely, by representing a given method in BRK format, it is readily seen whether the method is suitable for use on a parallel computer or not.

Below we present examples of methods from the literature which have been constructed for use on parallel computers. We shall use the BRK notation defined above and we give the order $p$ and the number of processors needed for implementation.

**Method of Miranker and Liniger [16].** Explicit, two-processor method which seems to be the first method constructed for parallel solution of initial-value problems:

\[
\begin{array}{cccccc}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{array}
\]

\[
\begin{array}{cccccc}
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & -1/3 & 4/3 & 8/3 & -5/3 \\
0 & 0 & 0 & 1 & 1/24 & -5/24 & 9/24 & 19/24 \\
\end{array}
\]
Method of Worland [21]. Explicit two-processor method based on the PECE mode of a predictor of Shampine & Watts [19] and the Clippinger-Dimsdale corrector [3]:

\[
\begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
0 & 1/3 & 1/3 & 1/3 \\
0 & 1/3 & 1/3 & 1/3 \\
\end{pmatrix}
\begin{pmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 1/4 & -1/3 & 13/12 \\
0 & 29/24 & -3 & 79/24 \\
\end{pmatrix}
\]

\[c = (-1/2, 0, 1/2, 1)^T, \quad p=k=r=4, \quad s=2.\]

Multi-block method of Chu and Hamilton [2]. Explicit two-processor method based on the PECE mode of a PC pair using full block methods:

\[
\begin{pmatrix}
1 & 0 \\
0 & 1 \\
5 & -4 \\
28 & -27 \\
\end{pmatrix}
\begin{pmatrix}
1 & 2 \\
6 & 9 \\
\end{pmatrix}
\]

\[c = (1/2, 1)^T, \quad p=4, \quad k=r=s=2.\]
BRK method not derived from PC pairs [7]. Explicit two-processor method requiring two starting values:

\[
\begin{array}{ccc}
1 & 0 & 0 \\
0 & 0 & 1 \\
0 & 1-a & a \\
\end{array}
\]

(1.7) \[ a = 4/3 \pm \sqrt{13/12}, \ c = (0, 1, 1)^T, \ p=3, \ k=r=3, \ s=1. \]

\[
\begin{array}{ccc}
0 & 0 & 1 \\
0 & 0 & 1 \\
0 & 0 & 1 \\
\end{array}
\]

\[
\begin{array}{ccc}
0 & 0 & 0 \\
0 & \frac{5}{6(a-1)} & 0 \\
\frac{7}{3} + a & \frac{7}{3} - a & 1 \\
\end{array}
\]

DIRK method of Iserles and Nørsett [13]. L-stable, diagonally-implicit two-processor method:

\[
\begin{array}{ccc}
1 & 1/2 & 0 \\
1 & 0 & 2/3 \\
1 & -5/2 & 5/2 \\
1 & -5/3 & 4/3 \\
\end{array}
\]

(1.8) \[ c = e, \ p=4, \ k=1, \ r=s=2. \]

\[
\begin{array}{ccc}
1 & -1 & 3/2 \\
\end{array}
\]

2. Construction of high-order methods

The order of the above-mentioned methods do not exceed \( p=4 \). In this section we outline the construction of higher-order methods. We distinguish methods requiring only one starting value (RK methods) and methods requiring several starting values (multistep methods).

2.1. RK methods

Let \( O_{ij} \) be the i-by-j null matrix, let \( A \) be a given r-by-r matrix, let \( B \) and \( D \) be r-by-r diagonal matrices, let \( b \) be a given vector of dimension \( r \), and let \( 0 \) and \( e \) respectively denote the null and unit vector whose dimension should be clear from the context in which it is used. Setting \( k=s \), we define the matrices

\[
A_j: = (0 \ldots 0 e), \ i = 1, \ldots, s; \ B_0: = (0 \ldots 0 e); \ O := O_{rr};
\]

(2.1a)

\[
B_1 := (b \ldots 0 0 0 0)^T, \ B_2 := (0 b 0 \ldots 0 0)^T, \ldots, \ B_s := (0 \ldots 0 0 0 b)^T,
\]

and we consider BRK methods generated by the Butcher-type array
with $y_{n+1}$ defined by (1.3). Evidently, this tableau generates a particular family of DIRK methods providing $k$ approximations to the true solution $y_{n+1}$. If $B=D=0$, then (2.1) reduces to an explicit RK (ERK) method. It is readily verified that (2.1) originates from iterating the r-stage RK method

$$y = y_0 + hAf(Y), \quad y_{n+1} = y_n + hB\tau(Y).$$

(for details of the iteration process we refer to [8] and [10]). The method (2.2) will be called the generating corrector formula. Usually, this corrector formula is an implicit RK method, but the considerations below also apply to the case where (2.2) is an ERK method.

In the following subsections, we construct matrices $A$, $B$, $C$ and $D$ and a vector $b$ such that the components $y_{n,i}$ of $Y_{n+1}$ provided by the BRK method (2.1) have orders $p-s+i$, $i=1,...,s$. Thus, we shall construct a pth-order DIRK method with embedded formulas of orders $p-1$, ..., $p-s+1$. We start with explicit methods, i.e., $B=D=0$.

### 2.1.1. ERK methods

We summarize the results we obtained for parallel ERK methods (cf. [8]). For that purpose, we need the notion of the number of sequential stages (or, as it was termed by Iserles and Nørsett [13], the number of effective stages).

**Definition 2.1.** An ERK method is said to require $s$ *sequential stages* if the computation time required for evaluating all right-hand sides in one step is $s$ times the computation time required by one right-hand side evaluation, assuming that sufficiently many processors are available.

Thus, if $B=D=0$, then (2.1) is an ERK method requiring $s$ sequential stages. In the paper of Iserles and Nørsett, the following theorem was proved:

**Theorem 2.1.** ERK methods of order $p$ necessarily require at least $p$ sequential stages.

This assertion led Nørsett and Simonsen [17] to pose the problem whether it is always possible to find ERK methods of order $p$ using not more than $p$ sequential stages, assuming that sufficiently many processors are available. Such methods were called optimal in [17].

**Definition 2.2.** An ERK method is called *optimal* if its number of sequential stages equals its order.

For $p \leq 4$, the above problem can (of course) be answered positively and for $p=5$ Nørsett and Simonsen mention a 6-stage, 5th-order method of Butcher possessing 5 sequential stages. For higher-order ERK methods, the following theorem [8] solves the problem posed by Nørsett and Simonsen:

**Theorem 2.2.** If $B=D=0$, $C=A$, and if the corrector formula (2.2) is of order $s$, then the method defined by (2.1) is optimal and the components $y_{n,i}$ of $Y_{n+1}$ have orders $i$, $i=1,...,s$. 

For any even \( p \) there exist RK methods of order \( p \) requiring \( p/2 \) stages (Gauss-Legendre methods) and for any odd \( p \) there exist RK methods of order \( p \) requiring \( (p+1)/2 \) stages (Radau methods). Using these methods as generating corrector formula (2.2), we have in (2.1) \( r=\lfloor(p+1)/2 \rfloor \), where \( \lfloor . \rfloor \) denotes the integer part function, and we are led to the corollary:

**Corollary 2.1.** There exist optimal ERK methods of any order \( p \) requiring \( [(p+1)/2] \) processors. \( \Box \)

In order to demonstrate that the use of parallel computers may save computing time, we compare the 'parallel, iterated' RK (PIRK) methods of this section with the so-called 8(7) method of Prince and Dormand [18]. According to [6] this method is nowadays generally considered as one of the most efficient methods with automatic stepsize control for TOL-values approximately in the range \( 10^{-7} \) to \( 10^{-13} \). We compare the DOPRI8 code, as given by Hairer, Nørsett and Wanner [6], with the PIRK method based on the Gauss-Legendre correctors of orders 8 and 10. To let the comparison of the DOPRI8 code and the PIRK codes not be influenced by a different stepsize strategy, we equipped the PIRK codes with the same strategy. These codes are respectively denoted by PIRK8 and PIRK10.

As test problem we take the equation of motion (cf. Problem B5 from [12]):

\[
\begin{align*}
    y_1' &= y_2y_3, & y_1(0) &= 0, \\
    y_2' &= -y_1y_3, & y_2(0) &= 1, \quad 0 \leq t \leq T. \\
    y_3' &= -0.51y_1y_2, & y_3(0) &= 1,
\end{align*}
\]

In Table 2.1, we have listed the values \( \Delta N \), where \( \Delta \) denotes the number of correct decimal digits at the endpoint (i.e., we write the maximum norm of the error at \( t=T \) in the form \( 10^{-\Delta} \)) and where \( N \) denotes the total number of sequential right-hand side evaluations performed during the integration process. For tolerances TOL running from \( 10^{-5} \) up to \( 10^{-12} \) we list the values of \( N \) which were found for a number of values of \( \Delta \).

<table>
<thead>
<tr>
<th>Method</th>
<th>( \Delta = 6 )</th>
<th>( \Delta = 7 )</th>
<th>( \Delta = 8 )</th>
<th>( \Delta = 9 )</th>
<th>( \Delta = 10 )</th>
<th>( \Delta = 11 )</th>
<th>( \Delta = 12 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>DOPRI8</td>
<td>415</td>
<td>576</td>
<td>728</td>
<td>898</td>
<td>1133</td>
<td>1422</td>
<td>1817</td>
</tr>
<tr>
<td>PIRK8</td>
<td>294</td>
<td>381</td>
<td>534</td>
<td>728</td>
<td>961</td>
<td>1172</td>
<td>1746</td>
</tr>
<tr>
<td>PIRK10</td>
<td>252</td>
<td>297</td>
<td>357</td>
<td>426</td>
<td>580</td>
<td>730</td>
<td>920</td>
</tr>
</tbody>
</table>

**2.1.2. DIRK methods**

Our main results for parallel DIRK methods of the form (2.1) with \( D \neq 0 \) obtained in [10] are summarized below.

Since the bulk of the computational effort required by these methods goes into the solution of the \( s-1 \) systems of equations, we define:

**Definition 2.2.** A DIRK method is said to require \( s \) sequential stages if the computation time required for solving all systems of equations in one step is \( s \) times the computation time required by solving one system of equations, assuming that sufficiently many processors are available. \( \Box \)

Thus, if both \( B \) and \( D \) do not vanish, then (2.1) is a DIRK method requiring \( s \)-sequential stages. If \( B=0 \) and \( D \neq 0 \), then \( s-1 \) sequential stages are required. The following theorem determines the order of the approximations \( y_{n+1} \):
Theorem 2.3. If the corrector formula (2.2) is of order $p^*$, then the components $y_{n,i}$ of $Y_{n+1}$ defined by the method (2.1) have orders $p_i$, $i=1, \ldots, s$ which are given by:

- Type IIA.1: $B=0$, $C=A-D$  
  $=> p_i = \min\{p^*, i\}$
- Type IIB.1: $B=D$, $C=A-D$  
  $=> p_i = \min\{p^*, i\}$
- Type IIB.2: $B=D=\text{diag} (Ae)$, $C=A-D$  
  $=> p_i = \min\{p^*, i+1\}$
- Type IIC.1: $B=O$  
  $=> p_i = \min\{p^*, i\}$, $i \geq 2$
- Type IIC.2: $B=O$, $D=\text{diag} (Ae-Ce)$  
  $=> p_i = \min\{p^*, i\}$
- Type IIC.3: $B=O$, $D=\text{diag} (Ae-Ce)$, $DAe=A^2e$  
  $=> p_i = \min\{p^*, i+1\}$. \(\square\)

Since $y_{n+1} = y_{n,8}$, it follows from this theorem that Type IC.3 methods are the most efficient ones because order $p^*$ requires only $p^*-1$ sequential stages. Unfortunately, in general Type IC.3 methods are not $A$-stable. The following corollary of Theorem 2.3 can be proved[10]:

Corollary 2.2. The order of $A$-stable Type IC methods cannot exceed their number of sequential stages plus 1, unless $b^TD^{-1}[I-AD^{-1}]^s Ce=0$. \(\square\)

Thus, in order to construct a DIRK method of Type IC of order 4, at least 3 sequential stages are required, whereas the L-stable, fourth-order DIRK method (1.8) of Iserles and Nørsett requires only 2 sequential stages. On the other hand, (2.1) also generates formulas of orders 2, 3, and 1. However, more important is the possibility to generate embedded methods of orders higher than $p^*$ possessing quite favourable stability properties. First we consider the case where $D$ has constant diagonal elements. This case allows a theoretical analysis. Analogous to an analysis by Wolfbrandt [20] of SDIRK methods (that is, RK methods with constant diagonal in the Butcher tableau), stating that for $1 \leq p \leq 6$ and $p=8$ the stability function of SDIRK methods of order $p$ and requiring $p$ stages is L-stable, and for $p=7$ and $9 \leq p \leq 15$ it does not (see [I, p. 248] for a summary of Wolfbrandt's result), we arrive at the theorem:

Theorem 2.4. Let the corrector formula (2.2) be of order $p^*=s$, then there exist values of $d$ such that the Type IIB.1 methods are $L$-stable for $1 \leq s \leq 6$ and $s=8$. \(\square\)

Within the class of methods with $D=dI$, it is possible to construct (at the cost of an additional sequential stage) still higher-order $L$-stable approximations. This can be achieved by choosing the corrector formula (2.2) such that $b=ATe$ (so-called stiffly accurate corrector formula) and by adding to $Y_{n+1}$ the component $(e_i)^TY(8)$. The resulting method is again of the BRK form (1.3), but has the property that $y_{n+1} = (e_i)^TY(8)$. Let us call these special methods Type II methods. First we state the analogue of Theorem 2.3 for Type II methods:

Theorem 2.5. If the corrector formula (2.2) is of order $p^*$ and satisfies $b=ATe$, then the order of $y_{n,i}$ is given by Theorem 2.3 and $y_{n+1} = (e_i)^TY(8)$ is also stiffly accurate with order $p$ given by:

- Type IIA.1: $B=0$, $C=A-D$  
  $=> p = \min\{p^*, s-1\}$
- Type IIB.1: $B=D$, $C=A-D$  
  $=> p = \min\{p^*, s-1\}$
- Type IIB.2: $B=D=\text{diag} (Ae)$, $C=A-D$  
  $=> p = \min\{p^*, s\}$
- Type IIC.1: $B=O$  
  $=> p = \min\{p^*, s-2\}$
- Type IIC.2: $B=0$, $D=\text{diag} (Ae-Ce)$  
  $=> p = \min\{p^*, s-1\}$
- Type IIC.3: $B=0$, $D=\text{diag} (Ae-Ce)$, $DAe=A^2e$  
  $=> p = \min\{p^*, s\}$. \(\square\)

The main stability results obtained for the Type II methods are given by

Theorem 2.6. If the corrector formula (2.2) is stiffly accurate ($b=ATe$) and has order $p^*=s-1$, then there exist values of $d$ such that:

(a) Type IIA.1 methods are $L$-stable for $1 \leq s \leq 7$ and $s=9$.
(b) Type IIB.1 methods are $L$-stable for $1 \leq s \leq 9$ and $s=11$. \(\square\)
We illustrate the performance of the methods by integrating a test problem proposed by Kaps [1981]:

\[
\begin{align*}
\frac{dy_1}{dt} &= -(2 + \frac{1}{\varepsilon})y_1 + \frac{1}{\varepsilon}y_2, \\
\frac{dy_2}{dt} &= y_1 - y_2(1 + y_2), \\
y_1(0) &= y_2(0) = 1, \quad 0 \leq t \leq 1,
\end{align*}
\]

with exact solution \(y_1 = \exp(-2t)\) and \(y_2 = \exp(-t)\) for all values of the parameter \(\varepsilon\). We tested several correctors and all types of methods which are L-stable. In Table 2.2 the values of \(\Delta\) are listed (cf. Table 2.1). Notice that the Type II methods require a stiffly accurate corrector (such as the Radau II formulas) and that L-stable, seventh-order methods are only possible within the family of Type IIB methods. In all experiments we observe the phenomenon of order reduction (if \(p\) is the order of the method, then, on halving the step size, the value of \(\Delta\) should increase by \(.3p\) if no order reduction is exhibited). Roughly speaking, we see a reduction by one order.

**Table 2.2. Values of \(\Delta\) for problem (2.4) at \(t=1\) with \(\varepsilon=10^{-2}\).**

<table>
<thead>
<tr>
<th>Type</th>
<th>Corrector</th>
<th>Order</th>
<th>(h=1/4)</th>
<th>(h=1/8)</th>
<th>(h=1/16)</th>
<th>(h=1/52)</th>
<th>(h=1/64)</th>
<th>Seq. Stages</th>
<th>Proc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>IIB.1</td>
<td>Radau II A</td>
<td>3</td>
<td>3.9</td>
<td>4.7</td>
<td>5.4</td>
<td>5.8</td>
<td>6.4</td>
<td>3/h</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Gauss-Legendre</td>
<td>4</td>
<td>3.0</td>
<td>3.7</td>
<td>4.4</td>
<td>5.2</td>
<td>6.0</td>
<td>4/h</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Explicit RK</td>
<td>4</td>
<td>3.1</td>
<td>3.8</td>
<td>4.5</td>
<td>5.3</td>
<td>6.1</td>
<td>4/h</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>Radau II A</td>
<td>5</td>
<td>3.5</td>
<td>4.3</td>
<td>5.2</td>
<td>6.2</td>
<td>7.4</td>
<td>5/h</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Gauss-Legendre</td>
<td>6</td>
<td>3.2</td>
<td>4.1</td>
<td>5.0</td>
<td>6.2</td>
<td>7.5</td>
<td>6/h</td>
<td>3</td>
</tr>
<tr>
<td>IIA.1</td>
<td>Radau II A</td>
<td>3</td>
<td>3.6</td>
<td>4.3</td>
<td>4.9</td>
<td>5.6</td>
<td>6.2</td>
<td>3/h</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Radau II A</td>
<td>5</td>
<td>3.8</td>
<td>4.5</td>
<td>5.3</td>
<td>6.3</td>
<td>7.5</td>
<td>5/h</td>
<td>3</td>
</tr>
<tr>
<td>IIB.1</td>
<td>Radau II A</td>
<td>3</td>
<td>4.2</td>
<td>4.6</td>
<td>5.2</td>
<td>5.9</td>
<td>6.7</td>
<td>4/h</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Radau II A</td>
<td>5</td>
<td>6.2</td>
<td>5.6</td>
<td>6.1</td>
<td>6.9</td>
<td>8.0</td>
<td>6/h</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Radau II A</td>
<td>7</td>
<td>4.3</td>
<td>5.2</td>
<td>6.2</td>
<td>7.7</td>
<td>9.4</td>
<td>8/h</td>
<td>4</td>
</tr>
</tbody>
</table>

We also considered Type II methods with matrix \(D\) possessing distinct diagonal entries. From Theorem 2.5 and by means of computer calculations found:

**Corollary 2.3.** Let the corrector formula (2.2) be defined by the \(r\)-stage Radau II A formula, and let \(s=2r-1\), then the Type IIC.3 methods have order \(p=2r-1\), they require \(p-1\) sequential stages, and they are \(A(\alpha_p)\)-stable where \(\alpha_3 = \pi/2, \alpha_5 = \pi/2 - 310^{-3}, \alpha_7 = \pi/2 - 410^{-2}\). ![image](image)

Thus, by using distinct diagonal values we can save a sequential stage.

We conclude this section with a comparison of the various DIRK methods available in the literature and those discussed in this paper. The methods designed for parallel computation are indicated by PDIRK. Effectively, all methods in this survey are SDIRK methods. The order range of the embedded formulas are listed in the column headed with \(\text{Pemb}\).
### 2.2. Multistep methods

We consider the special two-stage method

\[ Y_{n+1} = AY_n + hBf(Y_n) + hCf(Y_{n+1}) \]

where A, B and C are arbitrary k-by-k matrices (here, \( r = k \)). Unlike the RK methods of the preceding sections, not just one component of \( Y_n \) plays a role in this scheme, and therefore it belongs to the class of multistep methods.

The order conditions for methods of the above special two-stage form are extremely simple:

**Theorem 2.7.** Let the error vectors \( C_j \) be defined by

\[ C_j := A(c - e)^j + j[B(c - e)^{j-1} + Cc^{j-1}] - c, \quad j = 0, 1, \ldots \]

Then order \( p \) is obtained if the error vectors \( C_j \) vanish for \( j = 0, 1, \ldots, p \). \( \square \)

In this theorem powers of vectors are meant to be componentwise powers. The above order conditions are sufficient conditions but often they are not all necessary (for a discussion cf. [9]). The block point vector \( c \) plays an important role in the order conditions and by using this vector, we can achieve higher order or better stability than is possible within the class of block methods where the abscissas \( \tau_{i+1} \) are equally spaced.

In the following subsections we shall give examples of, respectively, explicit, diagonally-implicit, and fully implicit methods. Based on these methods, and by means of predictor-corrector (PC) iteration, we can construct high-order explicit and diagonally-implicit BRK methods (see Section 2.2.4).
2.2.1. Explicit BRK methods

Explicit methods arise for C=O reducing the method to one-stage form:

\[
\begin{array}{cc}
I & O \\
A & B
\end{array}
\]

This method needs k starting values and on k-processor computers it requires one sequential righthand side evaluation per step. Therefore, its computational complexity is comparable with that of explicit k-step linear multistep (LM) methods. For example, for k=2 we can construct the one-parameter family of third-order BRK methods (examples of higher-order methods up to order 7 can be found in [9])

\[
\begin{array}{ccc}
1 & 0 & c = (c, 1)^T, c \neq 1, p=3, k=2, s=1. \\
0 & 1 & \\
\end{array}
\]

\[
\begin{array}{ccc}
\frac{c^2(3c)}{(1-c)^3} & \frac{1-3c}{(1-c)^3} & \frac{c^2}{(1-c)^2} \\
\frac{5-3c}{(1-c)^3} & \frac{-c^3+3c^2-4}{(1-c)^3} & \frac{2-c}{(1-c)^2}
\end{array}
\]

For \(c \leq 1-\sqrt{3}\) and \(c \geq 1+\sqrt{3}\) this method is zero-stable and can be used as a method in its own right. For \(c=1\pm\sqrt{6}\), this method has zero parasitic roots if \(h=0\). Alternatively, the parameter \(c\) may be used for maximizing stability intervals (cf.[11]).

2.2.2. Diagonally-implicit BRK methods

Such methods arise when we set C=diag(d). As before, this method needs k starting values and on k-processor computers it requires the solution of one sequential, implicit relation per step. Therefore, its computational complexity is comparable with that of implicit k-step LM methods. In this case, we can construct for k=2 a three-parameter family of third-order methods. The length of the formulas prevents us from presenting them here (see [9]). In this family there are several strongly A-stable methods, of which one of them is given below:

\[
\begin{array}{ccc}
1 & 0 & 0 \\
0 & 1 & 0 \\
1/2 & 1/2 & -9/4 -1/4 5/2 0 \\
1 & 0 & -7/4 9/4 0 5/2
\end{array}
\]

\[
\begin{array}{ccc}
1/2 & 1/2 & -9/4 -1/4 5/2 0 \\
1 & 0 & -7/4 9/4 0 5/2
\end{array}
\]

2.2.3. Fully implicit BRK methods

We constructed fully implicit methods for use in PC-type methods. For k=2, we found the one-parameter family of fourth-order, zero-stable methods
\[
\begin{array}{c|ccc}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 1 & \frac{-c^3}{12(1-c)} & \frac{c(c^2-6c+6)}{12(1-c)} & \frac{c(c^2-6c+6)}{12(1-c)} & \frac{-c^3}{12(1-c)} \\
0 & 1 & \frac{6c^2-14c+7}{12c(1-c)} & \frac{3-2c}{12c(1-c)} & \frac{3-2c}{12c(1-c)} & \frac{6c^2-14c+7}{12c(1-c)} \\
\end{array}
\]

\[k=s=2, \quad c = (c, 1), p=4,\]

If \(c=1-\sqrt{1/5}\), then the method becomes fifth-order accurate.

### 2.2.4. PC-type methods

Let the predictor be of one of the two forms:

\[
\begin{array}{c|ccc}
I & O & O & O \\
D & E & F & D \\
\end{array}, \quad \begin{array}{c|c}
I & O \\
D & E \\
\end{array}
\]

with \(F = O\),

and let the corrector be of the form

\[
\begin{array}{c|ccc}
I & O & O & O \\
A & B & C & A \\
\end{array}, \quad \begin{array}{c|ccc}
A & B & C & O \\
\end{array}
\]

then we can construct higher-stage methods by PC iteration. In choosing a PC pair, the block point vactors \(c\) should be identical. For example, the PC pair \((2.5),(2.7)\) generates the 3-stage BRK method

\[
\begin{array}{c|ccc}
I & O & O & O \\
D & E & O & A \\
A & B & C & O \\
\end{array}, \quad \begin{array}{c|ccc}
A & B & C & O \\
\end{array}
\]

which is fourth-order accurate for all values of \(c\) and requires two sequential righthand side evaluations and two starting values. The same \((2.5),(2.7)\) pair generates the 4-stage, fifth-order BRK method:

\[
\begin{array}{c|ccc}
I & O & O & O \\
D & E & O & A \\
A & B & C & O \\
A & B & O & C & O \\
\end{array}, \quad \begin{array}{c|ccc}
A & B & O & C & O \\
\end{array}
\]

which requires 3 sequential righthand side evaluations.
In this way, we can construct high-order methods in a relatively straightforward manner. However, as for most block methods, the stability of the higher-order methods offers a considerable problem (see, e.g., Donelson & Hansen [4]). The highest order method we constructed so far is an explicit, stabilized, eighth-order, 3-stage method of the form (2.10) requiring four starting values and with real stability interval [-0.302, 0]. The matrices A, B, C, D and E in (2.10) are given by:

\[
A := \begin{pmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 \\
\frac{5 \times 3.73 \times 13.83}{30469 \times 2^{10}} & \frac{3 \times 5.2 \times 2.263}{30469} & 0 & -\frac{3 \times 5 \times 3.3.827}{30469 \times 2^{10}} \\
\frac{4 \times 549}{30469} & \frac{3 \times 1039}{30469} & 0 & -\frac{3 \times 7.9}{30469}
\end{pmatrix}, \quad
C := \begin{pmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & \frac{3 \times 5 \times 7 \times 809}{30469 \times 2^{5}} & \frac{3 \times 5 \times 3.73 \times 3.7}{30469 \times 2^{10}} \\
0 & 0 & \frac{2 \times 9.11}{30469 \times 3.5 \times 7} & \frac{14369}{30469}
\end{pmatrix}
\]

\[
B := \begin{pmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
\frac{3 \times 5 \times 6.7}{30469 \times 2^{9}} & \frac{3 \times 5 \times 2.7 \times 17.67}{30469 \times 2^{10}} & \frac{3 \times 5 \times 2.73}{30469 \times 2^{5}} & \frac{2 \times 5.7 \times 1.3}{30469 \times 2^{5}} \\
\frac{3 \times 2092}{30469 \times 3.7} & \frac{3 \times 13 \times 1709}{30469 \times 5.7} & \frac{2 \times 3 \times 2.31}{30469 \times 3.7} & \frac{3 \times 2 \times 6.3 \times 337}{30469 \times 5.7}
\end{pmatrix}, \quad
e = (1, 0, 5/2, 1)^T,
\]

\[
D := \begin{pmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 \\
\frac{5975}{224} & \frac{1539}{20} & -\frac{537}{35} & -\frac{2793}{32} \\
82 & 117 & 63232 & 2 \\
343 & 125 & 128625 & 3
\end{pmatrix}, \quad
E := \begin{pmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
\frac{225}{32} & \frac{567}{8} & 9 & \frac{2205}{32} \\
3 & 18 & 128 & \frac{49}{25} & \frac{1225}{1}
\end{pmatrix}
\]
2.2.5. Numerical examples

In Table 2.2 we compare the BRK methods of the preceding Section 2.2 with methods from the literature (see also Table 2.1). The methods are grouped according to their k-value which mainly determines the computational complexity.

<table>
<thead>
<tr>
<th>Sequential righthand sides N</th>
<th>120</th>
<th>240</th>
<th>480</th>
<th>960</th>
<th>1920</th>
<th>p</th>
<th>k</th>
</tr>
</thead>
<tbody>
<tr>
<td>Two-step Adams-Bashforth method</td>
<td>1.2</td>
<td>1.9</td>
<td>2.5</td>
<td>3.1</td>
<td>3.7</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Two-step Adams pair: PECE</td>
<td>1.2</td>
<td>2.0</td>
<td>2.9</td>
<td>3.8</td>
<td>4.7</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>Chu-Hamilton multi-block method (1.6)</td>
<td>*</td>
<td>3.3</td>
<td>4.7</td>
<td>6.0</td>
<td>7.3</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>BRK method (2.5): c=1 -\sqrt{6}</td>
<td>1.6</td>
<td>2.6</td>
<td>3.5</td>
<td>4.4</td>
<td>5.3</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>BRK method (2.11)</td>
<td>2.5</td>
<td>3.9</td>
<td>5.5</td>
<td>7.0</td>
<td>8.5</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>Four-step Adams-Bashforth method</td>
<td>3.3</td>
<td>3.8</td>
<td>4.8</td>
<td>6.0</td>
<td>7.1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Four-step Adams pair: PECE</td>
<td>2.5</td>
<td>3.4</td>
<td>4.8</td>
<td>6.2</td>
<td>7.7</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>Miranker-Liniger method (1.4)</td>
<td>3.1</td>
<td>5.0</td>
<td>6.3</td>
<td>7.2</td>
<td>8.3</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Shampine-Watts method (1.5)</td>
<td>1.9</td>
<td>3.3</td>
<td>4.6</td>
<td>5.9</td>
<td>7.2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>BRK method (2.12)</td>
<td>2.9</td>
<td>7.4</td>
<td>9.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

References


