Explicit Parallel Two-Step Runge-Kutta-Nyström Methods

N. H. CONG
CWI, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands
and
Faculty of Mathematics, Mechanics and Informatics
University of Hanoi, Thuong Dinh, Dong Da, Hanoi, Vietnam

(Received January 1994; accepted February 1994)

Abstract—The aim of this paper is to design a class of two-step Runge-Kutta-Nyström methods of arbitrarily high order for the special second-order equation \( y''(t) = f(y(t)) \), for use on parallel computers. Starting with an \( s \)-stage implicit two-step Runge-Kutta-Nyström method of order \( p \) with \( k = p/2 \) implicit stages, we apply the highly parallel predictor-corrector iteration process in \( P(EC)mE \) mode. In this way, we obtain an explicit two-step Runge-Kutta-Nyström method that has order \( p \) for all \( m \) and that requires \( k(m + 1) \) right-hand side evaluations per step of which each \( k \) evaluation can be computed in parallel. By a number of numerical experiments, we show the superiority of the parallel predictor-corrector methods proposed in this paper over both sequential and parallel methods available in the literature.

Keywords—Runge-Kutta-Nyström methods, Predictor-corrector methods, Parallelism.

1. INTRODUCTION

In the literature, several explicit Runge-Kutta-Nyström (RKN) methods have been proposed for the nonstiff second-order initial-value problem (IVP)

\[
\frac{d^2y(t)}{dt^2} = f(y(t)), \quad y(t_0) = y_0, \quad y'(t_0) = y'_0, \quad t_0 \leq t \leq T.
\]  

Methods up to order 10 can be found in [1-4]. In order to exploit the facilities of multiprocessor computers, a class of predictor-corrector (PC) methods based on (one-step) RKN correctors have recently been considered in [5,6]. In the present paper, we propose a class of parallel PC methods based on a new class of two-step RKN correctors. The new corrector method is designed by replacing in an \( s \)-stage, implicit, one-step RKN method \( s - k \) stage values by extrapolation formulas using information from the preceding step (see Section 2). In this way, we obtain a \( k \)-stage, implicit, two-step RKN corrector (TRKN corrector). A natural option chooses for the generating one-step RKN method a collocation method with optimal order of accuracy (see, e.g., [3,7]). Unfortunately, it turns out that the resulting TRKN correctors are often zero-unstable. However, by changing the location of the collocation points in the generating RKN method, we succeeded in finding zero-stable TRKN correctors of arbitrarily high stage and step point order.

These investigations were supported by the University of Amsterdam who provided the author with a research grant for spending a total of two years at the Centre for Mathematics and Computer Science, Amsterdam, The Netherlands. The author is grateful to P. J. van der Houwen and B. P. Sommeijer for their help during the preparation of this paper.
We do not claim that the collocation points obtained in this paper are the best possible. A further study of this topic will be subject of future research.

Having designed suitable TRKN correctors, we apply the highly parallel PC iteration scheme. The resulting method is analogous to the parallel iterated RKN (PIRKN) methods proposed in [5,6] and will therefore be termed parallel-iterated TRKN method (PITRKN method).

Although, for a given number of processors, the order of the PITRKN methods proposed in this paper equals that of the PIRKN method, their rate of convergence is much better, so that their efficiency is expected to be increased (see Section 4). The increased efficiency is demonstrated in Sections 4.1 and 4.2, where numerical results are presented by comparing the PITRKN methods with PIRKN methods and with sequential RKN methods available in the literature.

2. TWO-STEP RKN METHODS

In this section, we define the class of TRKN correctors that will be used in the parallel PC iteration scheme. For simplicity of notation, we assume that equation (1.1) is a scalar equation. However, all considerations below can be straightforwardly extended to a system of ODEs, and therefore, also to nonautonomous equations. We will start with a fully implicit s-stage collocation-based RKN method (see, e.g., [7]). For a scalar equation (1.1), this method assumes the form

\[
U_n = u_n + h u'_n e + h^2 A f(U_n),
\]

\[
u_{n+1} = u_n + h u'_n + h^2 b^T f(U_n),
\]

\[
u'_{n+1} = u'_n + h d^T f(U_n),
\]

where \(A\) is an \(s\)-by-\(s\) matrix, \(b\), \(c\), \(d\), and \(e\) are \(s\)-dimensional vectors, \(e\) is the vector with unit entries, \(c\) is the collocation vector, and \(U_n\) is the stage vector corresponding to the \(n\)th step. Furthermore, we use the convention that for any given vector \(v = (v_j)\), \(f(v)\) denotes the vector with entries \(f(v_j)\). In this paper, we confine the considerations to the case where (2.1) is based on a collocation vector \(c\) with all its components different from 1, i.e., the stage values differ from the step-point values. The method (2.1) will be referred to as the generating RKN method.

Now, let \(k\) be an arbitrarily given integer \((k < s)\) and let the parameters of the generating RKN method (2.1) be partitioned according to

\[
A = \begin{pmatrix} A_{s-k,s-k} & A_{s-k,k} \\ A_{k,s-k} & A_{kk} \end{pmatrix}, \quad b = \begin{pmatrix} b_{s-k} \\ b_k \end{pmatrix}, \quad c = \begin{pmatrix} c_{s-k} \\ c_k \end{pmatrix}, \quad d = \begin{pmatrix} d_{s-k} \\ d_k \end{pmatrix}, \quad e = \begin{pmatrix} e_{s-k} \\ e_k \end{pmatrix},
\]

where \(A_{ij}\) are \(i\)-by-\(j\) matrices, \(c_i, b_i, d_i, e_i\) are \(i\)-dimensional vectors. Defining the vector

\[
U_n = ((U_n^{(s-k)})^T, (U_n^{(k)})^T)^T,
\]

where \(U_n^{(s-k)}, U_n^{(k)}\) are \((s-k)\)-dimensional and \(k\)-dimensional stage subvectors, respectively, the generating RKN method (2.1) can be written in the form

\[
U_n^{(s-k)} = u_n e_{s-k} + h u'_n c_{s-k} + h^2 A_{s-k,s-k} f(U_n^{(s-k)}) + h^2 A_{s-k,k} f(U_n^{(k)}),
\]

\[
U_n^{(k)} = u_n e_k + h u'_n c_k + h^2 A_{k,s-k} f(U_n^{(s-k)}) + h^2 A_{kk} f(U_n^{(k)}),
\]

\[
u_{n+1} = u_n + h u'_n + h^2 b^T_{s-k} f(U_n^{(s-k)}) + h^2 b_k f(U_n^{(k)}),
\]

\[
u'_{n+1} = u'_n + h d^T_{s-k} f(U_n^{(s-k)}) + h d_k f(U_n^{(k)}),
\]

Suppose that we replace \(U_n^{(s-k)}\) by an extrapolation formula based on the stage vector \(U_{n-1}\). Then, we obtain the method

\[
V_n = y_n v + B_{s-k,s-k} V_{n-1} + B_{s-k,k} W_{n-1}, \quad n \geq 1,
\]

\[
W_n = y_n e_k + h y'_n c_k + h^2 A_{k,s-k} f(V_n) + h^2 A_{kk} f(W_n), \quad n \geq 0,
\]

(2.2a)
where the $B_{ij}$ are i-by-j extrapolation matrices and $v$ is an $(s - k)$-dimensional vector. The vector $(V_n^T, W_n^T)^T$ may be considered as the new stage vector for (2.2). Obviously, (2.2) can be considered as a two-step RKN method (TRKN method) with $s - k$ explicit and $k$ implicit stages, using the stage vectors $(V_{n-1}^T, W_{n-1}^T)^T$ and $(V_n^T, W_n^T)^T$. We shall call $V_n$ and $W_n$ the stage subvectors of the TRKN method. The parameters $v$ and $B_{ij}$ in (2.2a) are defined by order conditions which will be discussed in the next section. In addition to the initial values $y_0$ and $y'_0$, the TRKN method (2.2) requires $s - k$ starting values, that is, the $(s - k)$-dimensional starting vector $V_0$.

### 2.1. Order Conditions for the Explicit Stages

In this section, we describe the derivation of the parameter matrices $B_{s-k,s-k}$, $B_{s-k,k}$ and vector $v$ in (2.2a). In this derivation, we assume that $V_0$ is provided with the same order of accuracy as the stage order of the generating RKN method (2.1). We start with the following lemma.

**Lemma 2.1.** Let $U^{(s-k)}(t_n)$ denote the vector with components $y(t_n + c_i h)$, $i = 1, \ldots, s - k$, with $y$ the locally exact solution of (1.1). Moreover, let $u_n = y_n = y(t_n)$ and $u'_n = y'_n = y'(t_n)$. If (2.1) has stage order $r^* \geq s$ and if $U_n^{(s-k)}(t_n) - V_n = O(h^{q+1})$, then

$$U_n^{(s-k)} - V_n = O(h^{r^*+1}) + O(h^{q+1}), \quad U_n^{(k)} - W_n = O(h^{r^*+3}) + O(h^{q+3}).$$

**Proof.** Since the RKN method (2.1) is a collocation method, it has at least stage order $r^* = s$ and step point order $p^* = s$, for all sets of distinct collocation points $c_i$, $i = 1, \ldots, s$. The first relation is immediate from

$$U_n^{(s-k)} - V_n = U_n^{(s-k)} - U_n^{(s-k)}(t_n) + U_n^{(s-k)}(t_n) - V_n = O(h^{r^*+1}) + O(h^{q+1}).$$

Using this relation, we find

$$U_n^{(k)} - W_n = \left[ u_n e_k + h u'_n c_k + h^2 A_{k,s-k} f \left( U_n^{(s-k)} \right) + h^2 A_{kk} f \left( U_n^{(k)} \right) \right]$$

$$- \left[ y_n e_k + h y'_n c_k + h^2 A_{k,s-k} f(V_n) + h^2 A_{kk} f(W_n) \right]$$

$$= h^2 A_{k,s-k} \left[ f \left( U_n^{(s-k)} \right) - f(V_n) \right] + h^2 A_{kk} \left[ f \left( U_n^{(k)} \right) - f(W_n) \right]$$

$$= O(h^{r^*+3}) + O(h^{q+3}) + O(h^2) \left[ U_n^{(k)} - W_n \right],$$

which proves the second relation. \[ \Box \]

Now, we arrive at the following result for the TRKN method defined by (2.2).

**Theorem 2.1.** If (2.1) has stage order $r^* \geq s$ and step point order $p^* \geq s$, and if $U_n^{(s-k)}(t_n) - V_n = O(h^{q+1})$, then the TRKN method (2.2) has stage order $r = \min(r^*, q)$ and step point order $p = \min(p^*, r^* + 1, q + 1)$ for any set of collocation points.

**Proof.** For the local truncation error of the TRKN method (2.2), we may write

$$y(t_{n+1}) - y_{n+1} = y(t_{n+1}) - u_{n+1} + u_{n+1} - y_{n+1} = O(h^{p^*+1}) + u_{n+1} - y_{n+1},$$

$$y'(t_{n+1}) - y'_{n+1} = y'(t_{n+1}) - u'_{n+1} + u'_{n+1} - y'_{n+1} = O(h^{p^*+1}) + u'_{n+1} - y'_{n+1}.$$
By virtue of Lemma 2.1, we have
\[
\begin{align*}
 u_{n+1} - y_{n+1} &= h^2 b_{n-k}^T \left( f \left( U_n^{(s-k)} \right) - f(V_n) \right) + h^2 b_k^T \left( f \left( U_n^{(k)} \right) - f(W_n) \right) \\
 &= O \left( h^{r+3} + h^{q+3} \right) + O \left( h^{r+5} + h^{q+5} \right) = O \left( h^{r+3} + h^{q+3} \right),
\end{align*}
\]
\[
\begin{align*}
 u'_{n+1} - y'_{n+1} &= h d_{n-k}^T \left( f \left( U_n^{(s-k)} \right) - f(V_n) \right) + h d_k^T \left( f \left( U_n^{(k)} \right) - f(W_n) \right) \\
 &= O \left( h^{r+2} + h^{q+2} \right) + O \left( h^{r+4} + h^{q+4} \right) = O \left( h^{r+2} + h^{q+2} \right).
\end{align*}
\]
Hence, we obtain \( p = \min(p^*, r^* + 1, q + 1) \) and \( r = \min(r^*, q, p) = \min(r^*, q) \) (because \( r^* \leq p^* \)) which proves the assertion of the theorem.

The order conditions for the vector \( V_n \) ensuring that \( U_n^{(s-k)}(t_n) - V_n = O(h^{q+1}) \) are derived by replacing \( V_n, y_n, V_{n-1}, \) and \( W_{n-1} \) by the exact solution values \( y(t_n), y(t_n-e_{s-k} + c_{s-k} h), y(t_n-e_{s-k} + c_{s-k} h), y(t_n-1 e_k + c_k h), \) respectively. On substitution of these exact values into (2.2a) and by requiring that the residue is of order \( q + 1 \) in \( h, \) we are led to
\[
y(t_n e_{s-k} + c_{s-k} h) - y(t_n) = B_{s-k,s-k} y(t_n-1 e_{s-k} + c_{s-k} h) - B_{s-k,k} y(t_n-1 e_k + c_k h) = O \left( h^{q+1} \right).
\]
Using \((s + 1)\)-point Lagrange interpolation formulas with abscissa vector \( a = (c^T, 1)^T, \) we obtain (see, e.g., [8, p. 878])
\[
y(t_n + th) = \sum_{j=1}^{s+1} L_j(t+1) y(t_n-1 + a_j h) + C_{s+1}(t) \left( h \frac{d}{dt} \right)^{(s+1)} y(t^*),
\]
\[
L_j(x) := \prod_{i=1, i \neq j}^{s+1} \frac{x - a_i}{a_j - a_i}, \quad C_{s+1}(t) := \frac{1}{(s+1)!} \sum_{j=1}^{s+1} (t+1 - a_j),
\]
where \( t^* \) is a suitably chosen point in the interval containing the values \( t_n, t_{n-1} + c_i h, i = 1, \ldots, s + 1. \) Hence,
\[
y(t_n + c_m h) - \sum_{j=1}^{s-k} L_j(c_m + 1) y(t_n-1 + a_j h) - \sum_{j=s-k+1}^{s} L_j(c_m + 1) y(t_n-1 + a_j h) \\
- L_{s+1}(c_m + 1) y(t_n-1 + h) = C_{s+1}(c_m) \left( h \frac{d}{dt} \right)^{(s+1)} y(t^*),
\]
where \( t^*_m \) is a suitably chosen point in the interval containing the values \( t_n, t_{n-1} + c_i h, i = 1, \ldots, s + 1, m = 1, \ldots, s - k. \) Using componentwise notation, we obtain
\[
y(t_n e_{s-k} + c_{s-k} h) - (L_1(c_{s-k} + e_{s-k}), \ldots, L_{s-k}(c_{s-k} + e_{s-k})) y(t_n-1 e_{s-k} + c_{s-k} h) \\
- (L_{s-k+1}(c_{s-k} + e_{s-k}), \ldots, L_s(c_{s-k} + e_{s-k})) y(t_n-1 e_k + c_k h) - L_{s+1}(c_{s-k} + e_{s-k}) y(t_n) \\
= C_{s+1}(c_{s-k}) \left( h \frac{d}{dt} \right)^{s+1} y(t^*),
\]
where \( t^* = (t^*_{s-1-k}, \ldots, t^*_{s-k})^T. \) By defining
\[
B_{s-k,s-k} := (L_1(c_{s-k} + e_{s-k}), \ldots, L_{s-k}(c_{s-k} + e_{s-k})),
\]
\[
B_{s-k,k} := (L_{s-k+1}(c_{s-k} + e_{s-k}), \ldots, L_s(c_{s-k} + e_{s-k})),
\]
\[
v := L_{s+1}(c_{s-k} + e_{s-k}),
\]
a comparison with (2.4) reveals that we achieve \( q = s \) for any set of collocation points, and \( q = s + 1, \) if \( C_{s+1}(c_{s-k}) \) vanishes.
2.2. Zero-Stability

Since we have transformed the one-step RKN method (2.1) into the two-step method (2.2), we have to check the property of zero-stability. To that end, we rewrite (2.2) in the one-step form

$$Y_n = RY_{n-1} + hSY_{n-1} + hPf(Y_n) + h^2Qf(Y_n),$$

where $Y_n := (V_n, W_n, y_{n+1}^0, y_{n+1}^1)^T$, and $P, Q, R, S$ are all $(s + 2)$-by-$(s + 2)$ matrices given by

$$R = \begin{pmatrix} B_{s-k, s-k} & B_{s-k, k} & 0_{s-k} \\ O_{k,s-k} & O_{k,k} & e_k & 0_k \\ 0_{s-k}^T & 0_k^T & 0 & 1 \\ 0_{k}^T & 0_k^T & 0 & 1 \end{pmatrix}, \quad S = \begin{pmatrix} O_{s-k, s-k} & O_{s-k, k} & 0_{s-k} & 0_{s-k} \\ O_{k,s-k} & O_{k,k} & 0_k & c_k \\ 0_{s-k}^T & 0_k^T & 0 & 1 \\ 0_{k}^T & 0_k^T & 0 & 1 \end{pmatrix},$$

$$P = \begin{pmatrix} O_{s-k, s-k} & O_{s-k, k} & 0_{s-k} & 0_{s-k} \\ O_{k,s-k} & O_{k,k} & 0_k & 0_k \\ 0_{s-k}^T & 0_k^T & 0 & 1 \\ 0_{k}^T & 0_k^T & 0 & 1 \end{pmatrix}, \quad Q = \begin{pmatrix} O_{s-k, s-k} & O_{s-k, k} & 0_{s-k} & 0_{s-k} \\ A_{k,s-k} & A_{k,k} & 0_k & 0_k \\ b_{s-k}^T & b_k^T & 0 & 1 \\ 0_{s-k}^T & 0_k^T & 0 & 1 \end{pmatrix},$$

and where $O_{i,j}$ and $0_i$ are, respectively, $i$-by-$j$ matrices and $i$-dimensional vectors with zero entries. For zero-stability, we have to demand that no eigenvalue of the matrix $R$ has modulus greater than one, and that every eigenvalue of modulus one has multiplicity not greater than two. Hence, a sufficient condition for zero-stability of the TRKN method (2.2) is that the parameter matrix $B_{s-k, s-k}$ has its eigenvalues within the unit circle.

2.3. Choice of the Method Parameters

Suppose that the generating RKN method (2.1) is a collocation method. Then, the freedom in the choice of the collocation points $c_i$ of the TRKN method (2.2) can be used for obtaining some useful method properties. It seems natural to choose the abscissae such that the generating RKN method (2.1) has the highest possible order. For example, we may use the Gauss-Legendre points in each interval $[t_n, t_{n+1}]$. However, this choice can easily violate the condition of zero-stability. In Table 2.1, we have listed the spectral radius $\rho(B_{s-k, s-k})$ of $B_{s-k, s-k}$ for a few $(s, k)$-pairs.

<table>
<thead>
<tr>
<th>$(s, k)$</th>
<th>(3,2)</th>
<th>(4,3)</th>
<th>(4,2)</th>
<th>(5,4)</th>
<th>(5,3)</th>
<th>(5,2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\rho(B_{s-k,s-k})$</td>
<td>0.059</td>
<td>0.023</td>
<td>3.05</td>
<td>0.011</td>
<td>1.72</td>
<td>47.7</td>
</tr>
</tbody>
</table>

A second option minimizes the principal error vector associated with the extrapolation formula for the vector $V_n$, i.e., the vector

$$C_{s-k} := C_{s+1}(c_{s-k}) = (C_{s+1}(c_1), \ldots, C_{s+1}(c_{s-k}))^T,$$

where, according to (2.4),

$$C_{s+1}(c_\mu) = \frac{1}{(s+1)!} \prod_{i=1}^{s+1} (c_\mu + 1 - c_i) = \frac{1}{(s+1)!} c_\mu \prod_{i=1}^{s} (c_\mu + 1 - c_\mu), \quad \mu = 1, \ldots, s - k. \quad (2.8)$$

This vector vanishes if the set of components of the collocation vector $c$ contains the set of components of the vector $c_{s-k} + e_{s-k}$. By means of (2.6), it can be verified that the parameter matrix $B_{s-k, s-k}$ is strictly upper triangular so that it has zero eigenvalues, and consequently, the TRKN method is zero-stable. Thus, we have the following theorem.

**Theorem 2.2.** If the components of the collocation vector $c$ contain the components of the vector $c_{s-k} + e_{s-k}$, then the associated TRKN method is zero-stable.
3. PARALLEL ITERATED TRKN METHODS

Using (2.2) as corrector formula with predictor formula

\[ W_n^{(0)} = y_n w + C_{k,s-k} V_{n-1} + C_{kk} W_{n-1}^{(m)}, \]  

where the \( i \)-by-\( j \) matrices \( C_{ij} \) and the \( k \)-dimensional vector \( w \) are determined by order conditions, we arrive at the following PC iteration scheme (in \( P(EC)^mE \) mode)

\[ V_n = y_n v + B_{s-k,s-k} V_{n-1} + B_{s-k,k} W_{n-1}^{(m)}, \]
\[ W_n^{(j)} = y_n e_k + h c_k y_n + h^2 A_{k,s-k} f(V_n) + h^2 A_{kk} f(W_n^{(j-1)}), \quad j = 1, \ldots, m, \]
\[ y_{n+1} = y_n + h y_n + h^2 b_{s-k} f(V_n) + h^2 b_k f(W_n^{(m)}), \]
\[ y'_{n+1} = y'_{n} + h d_{s-k} f(V_n) + h d_k f(W_n^{(m)}). \]

The computational costs are measured by the number of sequential right-hand side evaluations (f-evaluations) per step (notice that the \((s-k)\) and \(k\) components of the vectors \( f(V_n) \) and \( f(W_n^{(j-1)}) \) can be computed in parallel, provided that \( \max(s-k, k) \) processors are available).

In general, we need \( m + 2 \) sequential f-evaluations. However, if \( c \) satisfies the condition of Theorem 2.2, then one f-evaluation can be saved, because \( f(V_n) \) can be copied from the preceding step and only \( k \) processors are needed. We shall call (3.1) a parallel-iterated TRKN method (PITRKN method).

3.1. Order Conditions for the Predictor

Along the lines of Section 2.1, we can prove that the conditions

\[ C_{k,s-k} = (L_1(c_k + e_k), \ldots, L_{s-k}(c_k + e_k)), \]
\[ C_{k,k} = (L_{s-k+1}(c_k + e_k), \ldots, L_s(c_k + e_k)), \quad w = L_{s+1}(c_k + e_k) \]  

imply that

\[ W(t_n) - W_n^{(0)} = O(h^{s+1}). \]  

Since each iteration raises the order of the iteration error by 2, the following order relations are obtained:

\[ W_n - W_n^{(m)} = O(h^{2m+s+1}), \]
\[ u_{n+1} - y_{n+1} = h^2 b_k \left[ f(W_n) - f(W_n^{(m)}) \right] = O(h^{2m+s+3}), \]
\[ u'_{n+1} - y'_{n+1} = h d_k \left[ f(W_n) - f(W_n^{(m)}) \right] = O(h^{2m+s+2}). \]

Thus, we have the following theorem.

**Theorem 2.3.** If (2.2) has step point order \( p \geq s \), and if (3.3) is satisfied, then the PITRKN method (3.1) has step point order \( \min(p, 2m + s + 1) \), for any set of collocation points.

3.2. The Rate of Convergence

The convergence boundary of a PITRKN method is defined in a similar way as for the PIRKN, BPIRK and PISRK methods proposed in [5,9,10]. Using the model test equation \( y''(t) = \lambda y(t) \), where \( \lambda \) runs through the eigenvalues of the Jacobian matrix \( \frac{\partial f}{\partial y} \), we obtain the iteration error equation

\[ W_n^{(j)} - W_n = z A_{kk} \left[ W_n^{(j-1)} - W_n \right], \quad z := \lambda h^2, \quad j = 1, \ldots, m. \]
Hence, with respect to the test equation, the rate of convergence is determined by the spectral radius \( \rho(A_{kk}) \) of the matrix \( A_{kk} \). We shall call \( \rho(A_{kk}) \) the convergence factor of the PITRKN method. Requiring that \( \rho(zA_{kk}) < 1 \) leads us to the convergence condition

\[
|z| < \frac{1}{\rho(A_{kk})} \quad \text{or} \quad h^2 < \frac{1}{\rho(A_{kk}) \rho(\frac{\partial}{\partial y})}.
\]

The freedom in the choice of the collocation points in the TRKN corrector can be used for obtaining a small convergence factor \( \rho(A_{kk}) \). Specification of convergence factors for a specified class of PITRKN methods is reported in Section 4.

### 3.3. Stability Regions

First, let us define the \((s + 2)\)-dimensional vectors

\[
E_{s+1} = (0, \ldots, 0, 1, 0)^T, \quad E_{s+2} = (0, \ldots, 0, 1)^T, \quad S_{s+2} = (0, \ldots, 0, 1, 1)^T
\]

and the matrices

\[
Q_{s-k,s+2} = v E_{s+1}^T + B_{s-k,s-k}(I_{s-k,s-k}, O_{s-k,k+2}) + B_{s-k,k}(O_{k,s-k}, I_{kk}, O_{k,2}),
\]

\[
P_{s+k+2} = w E_{s+1}^T + C_{s-k,s-k}(I_{s-k,s-k}, O_{s-k,k+2}) + C_{kk}(O_{k,s-k}, I_{kk}, O_{k,2}),
\]

\[
R_{s+k+2} = e_k E_{s+1}^T + c_k E_{s+2}^T,
\]

where \( I_{jj} \) is the \( j \)-by-\( j \) identity matrix. The linear stability of the method (3.1) is determined by again applying it to the model test equation \( y''(t) = \lambda y(t) \), where \( \lambda \) is assumed to be negative. Defining

\[
X_n^{(m)} = \begin{pmatrix} (V_{n-1})^T, (W_{n-1}^{(m)})^T, y_n, h y_n' \end{pmatrix}^T,
\]

and using (3.4), we obtain

\[
V_n = Q_{s-k,s+2} X_n^{(m)}.
\]

\[
W_n^{(m)} = (R_{s,k+2} + z A_{s-k,k} Q_{s-k,s+2}) X_n^{(m)} + z A_{kk} W_n^{(m-1)}
\]

\[
= (I + z A_{kk} + \cdots + (z A_{kk})^{m-1}) (R_{s,k+2} + z A_{s-k,k} Q_{s-k,s+2}) X_n^{(m)}
\]

\[
+ (z A_{kk})^m P_{k,s+2} X_n^{(m)}
\]

\[
= \left[(I - z A_{kk})^{-1} (I - (z A_{kk})^m) (R_{s,k+2} + z A_{s-k,k} Q_{s-k,s+2}) + (z A_{kk})^m P_{k,s+2}\right] X_n^{(m)},
\]

\[
y_{n+1} = y_n + h y_n' + z b_{s-k}^T V_n + z b_{k}^T W_n^{(m)}
\]

\[
= S_{s+2} X_n^{(m)} + z b_{s-k}^T Q_{s-k,s+2} X_n^{(m)} + z b_{k}^T (I - z A_{kk})^{-1} (I - (z A_{kk})^m)
\]

\[
\times (R_{s,k+2} + z A_{s-k,k} Q_{s-k,s+2}) + (z A_{kk})^m P_{k,s+2} X_n^{(m)}
\]

\[
= \left[S_{s+2} + z b_{s-k}^T Q_{s-k,s+2} + z b_{k}^T (I - z A_{kk})^{-1} (I - (z A_{kk})^m)
\]

\[
\times (R_{s,k+2} + z A_{s-k,k} Q_{s-k,s+2}) + (z A_{kk})^m P_{k,s+2}\right] X_n^{(m)},
\]

\[
h y_{n+1}' = h y_n' + z d_{s-k}^T V_n + z d_{k}^T W_n^{(m)}
\]

\[
= E_{s+2} X_n^{(m)} + z d_{s-k}^T Q_{s-k,s+2} X_n^{(m)} + z d_{k}^T (I - z A_{kk})^{-1} (I - (z A_{kk})^m)
\]

\[
\times (R_{s,k+2} + z A_{s-k,k} Q_{s-k,s+2}) + (z A_{kk})^m P_{k,s+2} X_n^{(m)}
\]

\[
= \left[E_{s+2} + z d_{s-k}^T Q_{s-k,s+2} + z d_{k}^T (I - z A_{kk})^{-1} (I - (z A_{kk})^m)
\]

\[
\times (R_{s,k+2} + z A_{s-k,k} Q_{s-k,s+2}) + (z A_{kk})^m P_{k,s+2}\right] X_n^{(m)},
\]

and the matrices

\[
Q_{s-k,s+2} = v E_{s+1}^T + B_{s-k,s-k}(I_{s-k,s-k}, O_{s-k,k+2}) + B_{s-k,k}(O_{k,s-k}, I_{kk}, O_{k,2}),
\]

\[
P_{s+k+2} = w E_{s+1}^T + C_{s-k,s-k}(I_{s-k,s-k}, O_{s-k,k+2}) + C_{kk}(O_{k,s-k}, I_{kk}, O_{k,2}),
\]

\[
R_{s+k+2} = e_k E_{s+1}^T + c_k E_{s+2}^T,
\]

where \( I_{jj} \) is the \( j \)-by-\( j \) identity matrix. The linear stability of the method (3.1) is determined by again applying it to the model test equation \( y''(t) = \lambda y(t) \), where \( \lambda \) is assumed to be negative. Defining
By introducing the matrices

\[ M_{k,s+2}(z) = (I - z A_{kk})^{-1} (I - (z A_{kk})^m) (R_{k,s+2} + z A_{k,s-k} Q_{s-k,s+2}) + (z A_{kk})^m P_{k,s+2}, \]

\[ M_{s+2}(z) = S_{s+2} + z b_{s-k}^T Q_{s-k,s+2} + z b_k^T \]
\[ \times ((I - z A_{kk})^{-1} (I - (z A_{kk})^m) (R_{k,s+2} + z A_{k,s-k} Q_{s-k,s+2}) + (z A_{kk})^m P_{k,s+2}), \]

\[ M_{s+2}^*(z) = E_{s+2} + z d_{s-k}^T Q_{s-k,s+2} + z d_k^T \]
\[ \times ((I - z A_{kk})^{-1} (I - (z A_{kk})^m) (R_{k,s+2} + z A_{k,s-k} Q_{s-k,s+2}) + (z A_{kk})^m P_{k,s+2}), \]

the relations (3.5) yield the recursion

\[ X_{n+1}^{(m)} = M_m(z) X_n^{(m)}, \quad M_m(z) = \left( \begin{array}{c} Q_{s-k,s+2} \\ M_{s+2}(z) \\ M_{s+2}^*(z) \end{array} \right). \]

The (s + 2)-by-(s + 2) matrix \( M_m(z) \) defined by (3.6), which determines the stability of the PITRKN methods, will be called the amplification matrix, its spectral radius \( \rho(M_m(z)) \) the stability function. For a given \( m \), the stability intervals of the PITRKN methods are defined by

\[ \{-\beta(m), 0\} := \{ z : \rho(M_m(z)) < 1, z \leq 0 \}. \]

The stability boundaries \( \beta(m) \) for the PITRKN methods used in our experiments can be found in Section 4.

**4. NUMERICAL EXPERIMENTS**

In this paper, we report numerical results for PITRKN methods with \( s = 2k \) and

\[ c = (c_{s-k}^T, c_k^T)^T, \quad c_{s-k} = (-c_k, \ldots, -c_1)^T, \quad c_k = (c_1, \ldots, c_k)^T, \quad k = 2, \ldots, 5, \]

where \( c_1, \ldots, c_k \) are the \( k \) components of the \( k \)-dimensional Gauss-Legendre collocation vector. By this choice, we have that \( p^* = s, r^* = s, \) and \( q = s + 1 \) (because the vector \( C_{s+1}(c_{s-k}) \) vanishes), so that the PITRKN methods defined by (3.1) have order \( s = 2k \) (see Theorems 2.1 and 2.3) and can be implemented on \( k = s/2 \) processors. These orders and number of processors are the same as used by the PIRKN methods proposed in [5,6]. However, a direct numerical computation reveals that the convergence factor as defined in Section 3.2 is much smaller than that of PIRKN methods (see Table 4.1).

<table>
<thead>
<tr>
<th>Parallel pth-order PC methods</th>
<th>( p = 4 )</th>
<th>( p = 6 )</th>
<th>( p = 8 )</th>
<th>( p = 10 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct PIRKN methods (cf. [5])</td>
<td>0.048</td>
<td>0.029</td>
<td>0.018</td>
<td>0.013</td>
</tr>
<tr>
<td>Indirect PIRKN methods (cf. [5])</td>
<td>0.083</td>
<td>0.046</td>
<td>0.027</td>
<td>0.019</td>
</tr>
<tr>
<td>PITRKN methods</td>
<td>0.026</td>
<td>0.015</td>
<td>0.009</td>
<td>0.006</td>
</tr>
</tbody>
</table>

As shown in Table 4.2, the stability boundaries of the PITRKN methods are sufficiently large for nonstiff problems.

<table>
<thead>
<tr>
<th>pth-order PITRKN methods</th>
<th>( p = 4 )</th>
<th>( p = 6 )</th>
<th>( p = 8 )</th>
<th>( p = 10 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( m = 1 )</td>
<td>0.42</td>
<td>0.09</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>( m = 2 )</td>
<td>4.15</td>
<td>1.37</td>
<td>0.51</td>
<td>0.10</td>
</tr>
<tr>
<td>( m = 3 )</td>
<td>7.93</td>
<td>7.07</td>
<td>2.54</td>
<td>1.13</td>
</tr>
<tr>
<td>( m = 4 )</td>
<td>8.50</td>
<td>16.20</td>
<td>7.48</td>
<td>3.74</td>
</tr>
</tbody>
</table>
In order to see the efficiency of the various PC methods, we applied a dynamical strategy for determining the number of iterations in the successive steps using the stopping criterion

\[ \left\| W_n^{(m)} - W_n^{(m-1)} \right\|_\infty \leq \text{TOL} = C h^{p-1}, \tag{4.2} \]

where \( p \) is the order of the corrector method, and \( C \) is a parameter depending on the method and on the problem. Notice that by this criterion, the iteration error is of the same order in \( h \) as the underlying corrector.

4.1. Comparison with Parallel Methods

In this section, we report numerical results obtained by the best parallel methods available in the literature, the (indirect) PIRKN methods proposed in [6] and the PITRKN methods considered in this paper. The absolute error obtained at the end of the integration interval is presented in the form \( 10^{-d} \) (\( d \) may be interpreted as the number of correct decimal digits (NCD)). Furthermore, in the tables of results, \( N_{\text{seq}} \) denotes the total number of sequential \( f \)-evaluations, and \( N_{\text{steps}} \) denotes the total number of integration steps. The following three problems possess exact solutions in closed form. Initial conditions are taken from the exact solutions.

4.1.1. Linear nonautonomous problem

As a first numerical test, we apply the various \( p \)th-order PC methods to the linear problem (cf. [5])

\[
\frac{d^2 y(t)}{dt^2} = \left( \begin{array}{cc}
-2\alpha(t) + 1 & -\alpha(t) + 1 \\
2(\alpha(t) - 1) & \alpha(t) - 2
\end{array} \right) y(t), \quad \alpha(t) = \max(2\cos^2(t), \sin^2(t)), \quad 0 \leq t \leq 20,
\]

with exact solution \( y(t) = (-\sin(t), 2\sin(t))^T \). The results listed in Table 4.3 clearly show that the PITRKN methods are by far superior to the PIRKN methods of the same order. The average number of sequential \( f \)-evaluations per step for PITRKN methods is about two for all methods.

<table>
<thead>
<tr>
<th>( p )th-order PC methods</th>
<th>( p )</th>
<th>( N_{\text{steps}} = 80 )</th>
<th>( N_{\text{steps}} = 160 )</th>
<th>( N_{\text{steps}} = 320 )</th>
<th>( N_{\text{steps}} = 640 )</th>
<th>( N_{\text{steps}} = 1280 )</th>
<th>( C )</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIRKN</td>
<td>4</td>
<td>4.0 / 237</td>
<td>5.3 / 477</td>
<td>6.5 / 958</td>
<td>7.7 / 1919</td>
<td>8.9 / 3836</td>
<td>( 10^{-1} )</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>4.8 / 161</td>
<td>6.2 / 321</td>
<td>7.5 / 641</td>
<td>8.7 / 1281</td>
<td>10.0 / 2561</td>
<td>( 10^{-1} )</td>
</tr>
<tr>
<td>PITRKN</td>
<td>6</td>
<td>7.4 / 320</td>
<td>9.2 / 640</td>
<td>11.0 / 1280</td>
<td>12.8 / 2559</td>
<td>14.6 / 5119</td>
<td>( 10^{-3} )</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>8.2 / 163</td>
<td>10.5 / 322</td>
<td>12.5 / 642</td>
<td>14.4 / 1282</td>
<td>16.2 / 2562</td>
<td>( 10^{-3} )</td>
</tr>
<tr>
<td>PIRKN</td>
<td>8</td>
<td>11.0 / 399</td>
<td>13.4 / 799</td>
<td>15.8 / 1600</td>
<td>18.2 / 3198</td>
<td>20.6 / 6398</td>
<td>( 10^{-4} )</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>12.1 / 211</td>
<td>14.2 / 380</td>
<td>17.9 / 683</td>
<td>20.2 / 1283</td>
<td>22.8 / 2563</td>
<td>( 10^{-4} )</td>
</tr>
<tr>
<td>PITRKN</td>
<td>10</td>
<td>13.3 / 438</td>
<td>18.0 / 921</td>
<td>20.9 / 1881</td>
<td>23.8 / 3803</td>
<td>24.1 / 1403</td>
<td>( 10^{-4} )</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>14.2 / 233</td>
<td>17.3 / 407</td>
<td>20.3 / 750</td>
<td>24.1 / 1403</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4.1.2. Nonlinear Fehlberg problem

For the second numerical example, we consider the often-used orbit equation (cf., e.g., [1,2, 11,12])

\[
\frac{d^2 y(t)}{dt^2} = \left( \begin{array}{cc}
-4t^2, & -2 \\
2 & -4t^2
\end{array} \right) y(t), \quad r(t) = \sqrt{y_1^2(t) + y_2^2(t)}, \quad \sqrt{\frac{\pi}{2}} \leq t \leq 10. \tag{4.4}
\]
Table 4.4. Values of \( N_{CD}/N_{seq} \) for problem (4.4) obtained by various \( p^{th} \)-order parallel PC methods.

<table>
<thead>
<tr>
<th>( p^{th} )-order PC methods</th>
<th>( p )</th>
<th>( N_{steps} = 200 )</th>
<th>( N_{steps} = 400 )</th>
<th>( N_{steps} = 800 )</th>
<th>( N_{steps} = 1600 )</th>
<th>( N_{steps} = 3200 )</th>
<th>( C )</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIRKN</td>
<td>4</td>
<td>1.6 / 591</td>
<td>2.8 / 1197</td>
<td>4.0 / 2400</td>
<td>5.2 / 4800</td>
<td>6.4 / 9600</td>
<td>10^2</td>
</tr>
<tr>
<td>PITRKN</td>
<td>4</td>
<td>2.7 / 441</td>
<td>3.8 / 802</td>
<td>5.1 / 1601</td>
<td>6.4 / 3201</td>
<td>7.6 / 6401</td>
<td>10^2</td>
</tr>
<tr>
<td>PIRKN</td>
<td>6</td>
<td>4.0 / 775</td>
<td>5.8 / 1532</td>
<td>7.6 / 3096</td>
<td>9.4 / 6257</td>
<td>11.2 / 12648</td>
<td>10^3</td>
</tr>
<tr>
<td>PITRKN</td>
<td>6</td>
<td>5.3 / 495</td>
<td>7.1 / 880</td>
<td>9.0 / 1601</td>
<td>11.0 / 3201</td>
<td>12.9 / 6401</td>
<td>10^3</td>
</tr>
<tr>
<td>PIRKN</td>
<td>8</td>
<td>6.6 / 1022</td>
<td>9.0 / 2032</td>
<td>11.5 / 4028</td>
<td>13.9 / 7966</td>
<td>16.3 / 15725</td>
<td>10^3</td>
</tr>
<tr>
<td>PITRKN</td>
<td>8</td>
<td>8.7 / 575</td>
<td>11.1 / 1051</td>
<td>13.5 / 1988</td>
<td>15.9 / 3672</td>
<td>18.3 / 6616</td>
<td>10^3</td>
</tr>
<tr>
<td>PIRKN</td>
<td>10</td>
<td>9.4 / 1234</td>
<td>12.4 / 2458</td>
<td>15.5 / 4893</td>
<td>18.5 / 9734</td>
<td>21.5 / 19332</td>
<td>10^3</td>
</tr>
</tbody>
</table>

The exact solution is given by \( y(t) = (\cos(t^2), \sin(t^2))^T \). The results are reported in Table 4.4. For this nonlinear problem, we observe a similar superiority of the PITRKN methods over the PIRKN methods as in the previous example.

4.1.3. Newton's equations of motion problem

The third example is the two-body gravitational problem for Newton's equation of motion (see [13, p. 245]):

\[
\frac{d^2 y_1(t)}{dt^2} = -\frac{y_1(t)}{(r(t))^3}, \quad \frac{d^2 y_2(t)}{dt^2} = -\frac{y_2(t)}{(r(t))^3}, \quad 0 \leq t \leq 20,
\]

\[
y_1(0) = 1 - \varepsilon, \quad y_2(0) = 0, \quad y'_1(0) = 0, \quad y'_2(0) = \sqrt{\frac{1 + \varepsilon}{1 - \varepsilon}},
\]

where \( r(t) = \sqrt{y_1^2(t) + y_2^2(t)} \). The solution components are \( y_1(t) = \cos(u) - \varepsilon, \ y_2(t) = \sqrt{(1 + \varepsilon)(1 - \varepsilon)}\sin(u), \) where \( u \) is the solution of Kepler's equation \( t = u - \varepsilon \sin(u) \) and \( \varepsilon \) denotes the eccentricity of the orbit. In this example, we set \( \varepsilon = 0.3 \). As in the two preceding examples, the results listed in Table 4.5 show that the PITRKN methods are about twice as efficient as the PIRKN methods.

Table 4.5. Values of \( N_{CD}/N_{seq} \) for problem (4.5) obtained by various \( p^{th} \)-order parallel PC methods.

<table>
<thead>
<tr>
<th>( p^{th} )-order PC methods</th>
<th>( p )</th>
<th>( N_{steps} = 100 )</th>
<th>( N_{steps} = 200 )</th>
<th>( N_{steps} = 400 )</th>
<th>( N_{steps} = 800 )</th>
<th>( N_{steps} = 1600 )</th>
<th>( C )</th>
</tr>
</thead>
<tbody>
<tr>
<td>PIRKN</td>
<td>4</td>
<td>1.9 / 200</td>
<td>3.3 / 400</td>
<td>5.0 / 841</td>
<td>6.2 / 1995</td>
<td>7.3 / 4800</td>
<td>10^1</td>
</tr>
<tr>
<td>PITRKN</td>
<td>4</td>
<td>3.1 / 200</td>
<td>4.1 / 400</td>
<td>5.3 / 800</td>
<td>6.4 / 1601</td>
<td>7.6 / 3201</td>
<td>10^1</td>
</tr>
<tr>
<td>PIRKN</td>
<td>6</td>
<td>5.1 / 360</td>
<td>6.8 / 800</td>
<td>8.6 / 1600</td>
<td>10.4 / 3200</td>
<td>12.2 / 6400</td>
<td>10^{-1}</td>
</tr>
<tr>
<td>PITRKN</td>
<td>6</td>
<td>5.7 / 232</td>
<td>7.5 / 402</td>
<td>9.1 / 802</td>
<td>10.8 / 1602</td>
<td>12.6 / 3202</td>
<td>10^{-1}</td>
</tr>
<tr>
<td>PIRKN</td>
<td>8</td>
<td>7.7 / 450</td>
<td>10.1 / 917</td>
<td>12.5 / 1934</td>
<td>14.9 / 4000</td>
<td>17.3 / 8000</td>
<td>10^{-2}</td>
</tr>
<tr>
<td>PITRKN</td>
<td>8</td>
<td>9.4 / 268</td>
<td>10.6 / 497</td>
<td>12.9 / 890</td>
<td>15.2 / 1663</td>
<td>17.6 / 3203</td>
<td>10^{-2}</td>
</tr>
<tr>
<td>PIRKN</td>
<td>10</td>
<td>10.4 / 517</td>
<td>13.3 / 1050</td>
<td>16.2 / 2127</td>
<td>19.2 / 4306</td>
<td>22.2 / 8706</td>
<td>10^{-2}</td>
</tr>
<tr>
<td>PITRKN</td>
<td>10</td>
<td>10.8 / 297</td>
<td>13.7 / 546</td>
<td>16.8 / 1022</td>
<td>19.6 / 1898</td>
<td>22.6 / 3515</td>
<td>10^{-2}</td>
</tr>
</tbody>
</table>
4.2. Comparison with Sequential Methods

In Section 4.1, the PITRKN methods were compared with PIRKN methods (the most efficient parallel methods for nonstiff problems). In this section, we will compare the PITRKN methods with the sequential methods currently available.

We restricted our tests to the comparison of our tenth-order PITRKN method (PITRKN$^{10}$ method) with a few well-known sequential codes for the orbit problem (4.4). We selected some embedded RKN pairs presented in the form $p(p + 1)$ or $(p + 1)p$ constructed in [1,2,11,12] and the RKN code DOPRIN taken from [14]. We reproduced the best results obtained by these sequential methods given in the literature (cf., e.g., [6,12]) and added the results obtained by PITRKN$^{10}$ method. In spite of the fact that the results of the sequential methods are obtained using a stepsize strategy, whereas PITRKN$^{10}$ method is applied with fixed stepsizes, it is the PITRKN$^{10}$ method that performs most efficiently (see Table 4.6).

Table 4.6. Comparison with the sequential methods for problem (4.4).

<table>
<thead>
<tr>
<th>Methods</th>
<th>$N_{\text{steps}}$</th>
<th>NCD</th>
<th>$N_{\text{seq}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>11(10) pair (from [12])</td>
<td>919</td>
<td>20.7</td>
<td>15614</td>
</tr>
<tr>
<td>8(9)-pair (from [1])</td>
<td>1452</td>
<td>13.5</td>
<td>15973</td>
</tr>
<tr>
<td>9(10)-pair (from [2])</td>
<td>628</td>
<td>15.1</td>
<td>8793</td>
</tr>
<tr>
<td>11(12)-pair (from [11])</td>
<td>3235</td>
<td>21.4</td>
<td>45291</td>
</tr>
<tr>
<td>DOPRIN (from [14])</td>
<td>876</td>
<td>20.3</td>
<td>17521</td>
</tr>
<tr>
<td>11208</td>
<td>12.3</td>
<td>9665</td>
<td></td>
</tr>
<tr>
<td>4466</td>
<td>16.3</td>
<td>35729</td>
<td></td>
</tr>
<tr>
<td>16667</td>
<td>20.3</td>
<td>133337</td>
<td></td>
</tr>
<tr>
<td>PITRKN$^{10}$ (in this paper)</td>
<td>200</td>
<td>11.4</td>
<td>674</td>
</tr>
<tr>
<td>400</td>
<td>14.5</td>
<td>1156</td>
<td></td>
</tr>
<tr>
<td>800</td>
<td>18.1</td>
<td>2139</td>
<td></td>
</tr>
<tr>
<td>1600</td>
<td>21.1</td>
<td>4094</td>
<td></td>
</tr>
</tbody>
</table>

5. CONCLUDING REMARKS

In this paper, we proposed a new class of two-step RKN correctors of order $2k$, where $k$ is the number of implicit stages. When solved by parallel predictor-corrector iteration, the sequential costs are considerably less than those of the best parallel and sequential methods available in the literature.

REFERENCES


