Explicit parallel two-step Runge-Kutta-Nyström methods

Nguyen huu Cong

Department of Numerical Mathematics

Report NM-R9401 January 1994

Explicit Parallel Two-Step Runge-Kutta-Nyström Methods

Nguyen huu Cong
CWI
P. O. Box 94079, 1090 GB Amsterdam, The Netherlands
&
Faculty of Mathematics, Mechanics and Informatics
University of Hanoi, Thuong Dinh, Dong Da, Hanoi, Vietnam

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) righthand side evaluations per step of which each k evaluations 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.

AMS Subject Classification (1991): 65M12, 65M20
CR Subject Classification: G.1.7
Keywords and Phrases: Runge-Kutta-Nyström methods, predictor-corrector methods, parallelism.

Note: 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.

1. Introduction

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

$$\frac{d^2y(t)}{dt^2} = f(y(t)), \quad y(t_0) = y_0, \quad y'(t_0) = y'_0, \quad t_0 \le t \le T. \tag{1.1}$$

Methods up to order 10 can be found in [2], [3], [8] and [9]. In order to exploit the facilities of multi-processor computers, a class of predictor-corrector (PC) methods based on (one-step) RKN correctors have recently been considered in [15] and [19]. 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. [8] and [12]). Unfortunately, it turns out that the

Report NM-R9401
ISSN 0169-0388
CWI
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands

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. 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 [15], [19] 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 Subsections 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. [12]). For a scalar equation (1.1), this method assumes the form

(2.1a) $$\mathbf{U}_n = u_n\mathbf{e} + hu'_n\mathbf{c} + h^2Af(\mathbf{U}_n),$$

(2.1b) $$u_{n+1} = u_n + hu'_n + h^2\mathbf{b}^Tf(\mathbf{U}_n),$$

(2.1c) $$u'_{n+1} = u'_n + h\,\mathbf{d}^Tf(\mathbf{U}_n),$$

where A is an s-by-s matrix, **b, c, d, e** are s-dimensional vectors, **e** is the vector with unit entries, **c** is the collocation vector, and $\mathbf{U}_n$ is the stage vector corresponding to the n-th step. Furthermore, we use the convention that for any given vector $\mathbf{v} = (v_j)$, $f(\mathbf{v})$ 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 steppoint 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

$$A = \begin{pmatrix} A_{s-k,s-k} & A_{s-k,k} \\ A_{k,s-k} & A_{kk} \end{pmatrix},\ \mathbf{b} = \begin{pmatrix} \mathbf{b}_{s-k} \\ \mathbf{b}_k \end{pmatrix},\ \mathbf{c} = \begin{pmatrix} \mathbf{c}_{s-k} \\ \mathbf{c}_k \end{pmatrix},\ \mathbf{d} = \begin{pmatrix} \mathbf{d}_{s-k} \\ \mathbf{d}_k \end{pmatrix},\ \mathbf{e} = \begin{pmatrix} \mathbf{e}_{s-k} \\ \mathbf{e}_k \end{pmatrix},$$

where A_{ij} are i-by-j matrices, $\mathbf{c}_i$, $\mathbf{b}_i$, $\mathbf{d}_i$, $\mathbf{e}_i$ are i-dimensional vectors. Defining the vector $\mathbf{U}_n = ((\mathbf{U}_n^{(s-k)})^T, (\mathbf{U}_n^{(k)})^T)^T$, where $\mathbf{U}_n^{(s-k)}$, $\mathbf{U}_n^{(k)}$ are (s-k)-dimensional and k-dimensional stage subvectors, respectively, the generating RKN method (2.1) can be written in the form

(2.1a') $$\begin{aligned}\mathbf{U}_n^{(s-k)} &= u_n\mathbf{e}_{s-k} + hu'_n\mathbf{c}_{s-k} + h^2A_{s-k,s-k}f(\mathbf{U}_n^{(s-k)}) + h^2A_{s-k,k}f(\mathbf{U}_n^{(k)}),\\ \mathbf{U}_n^{(k)} &= u_n\mathbf{e}_k + hu'_n\mathbf{c}_k + h^2A_{k,s-k}f(\mathbf{U}_n^{(s-k)}) + h^2A_{kk}f(\mathbf{U}_n^{(k)}),\end{aligned}$$

(2.1b') $$u_{n+1} = u_n + hu'_n + h^2\mathbf{b}_{s-k}^Tf(\mathbf{U}_n^{(s-k)}) + h^2\mathbf{b}_k^Tf(\mathbf{U}_n^{(k)}),$$

(2.1c') $\quad u'_{n+1} = u'_n + h\,\mathbf{d}_{s-k}{}^T f(\mathbf{U}_n{}^{(s-k)}) + h\,\mathbf{d}_k{}^T f(\mathbf{U}_n{}^{(k)})$.

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

$$\mathbf{V}_n = y_n\mathbf{v} + B_{s-k,s-k}\mathbf{V}_{n-1} + B_{s-k,k}\mathbf{W}_{n-1}, \qquad (n \geq 1)$$

(2.2a)

$$\mathbf{W}_n = y_n\mathbf{e}_k + hy'_n\mathbf{c}_k + h^2A_{k,s-k}f(\mathbf{V}_n) + h^2A_{kk}f(\mathbf{W}_n), \qquad (n \geq 0)$$

(2.2b) $\quad y_{n+1} = y_n + hy'_n + h^2\mathbf{b}_{s-k}{}^T f(\mathbf{V}_n) + h^2\mathbf{b}_k{}^T f(\mathbf{W}_n), \qquad (n \geq 0)$

(2.2c) $\quad y'_{n+1} = y'_n + h\,\mathbf{d}_{s-k}{}^T f(\mathbf{V}_n) + h\,\mathbf{d}_k{}^T f(\mathbf{W}_n), \qquad (n \geq 0)$

where the B_{ij} are i-by-j extrapolation matrices and $\mathbf{v}$ is an (s-k)-dimensional vector. The vector $(\mathbf{V}_n{}^T, \mathbf{W}_n{}^T)^T$ 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 $(\mathbf{V}_n{}^T, \mathbf{W}_n{}^T)^T$ and $(\mathbf{V}_{n-1}{}^T, \mathbf{W}_{n-1}{}^T)^T$. We shall call $\mathbf{V}_n$ and $\mathbf{W}_n$ the stage subvectors of the TRKN method. The parameters $\mathbf{v}$ and B_{ij} in (2.2a) are defined by order conditions which will be discussed in the next subsection. 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 $\mathbf{V}_0$.

2.1. Order conditions for the explicit stages

In this subsection we describe the derivation of the parameter matrices $B_{s-k,s-k}$, $B_{s-k,k}$ and vector $\mathbf{v}$ in (2.2a). In this derivation, we assume that $\mathbf{V}_0$ 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 $\mathbf{U}^{(s-k)}(t_n)$ denote the vector with components $y(t_n + c_ih)$, i = 1, ..., 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 $\mathbf{U}_n{}^{(s-k)}(t_n) - \mathbf{V}_n = O(h^{q+1})$, then

$$\mathbf{U}_n{}^{(s-k)} - \mathbf{V}_n = O(h^{r^*+1}) + O(h^{q+1}), \;\; \mathbf{U}_n{}^{(k)} - \mathbf{W}_n = O(h^{r^*+3}) + O(h^{q+3}).$$

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, ..., s. The first relation is immediate from

$$\mathbf{U}_n{}^{(s-k)} - \mathbf{V}_n = \mathbf{U}_n{}^{(s-k)} - \mathbf{U}_n{}^{(s-k)}(t_n) + \mathbf{U}_n{}^{(s-k)}(t_n) - \mathbf{V}_n = O(h^{r^*+1}) + O(h^{q+1}).$$

Using this relation, we find

$$\begin{aligned}\mathbf{U}_n{}^{(k)} - \mathbf{W}_n &= [u_n\mathbf{e}_k + hu'_n\mathbf{c}_k + h^2A_{k,s-k}f(\mathbf{U}_n{}^{(s-k)}) + h^2A_{kk}f(\mathbf{U}_n{}^{(k)})] \\ &\quad - [y_n\mathbf{e}_k + hy'_n\mathbf{c}_k + h^2A_{k,s-k}f(\mathbf{V}_n) + h^2A_{kk}f(\mathbf{W}_n)] \\ &= h^2A_{k,s-k}[f(\mathbf{U}_n{}^{(s-k)}) - f(\mathbf{V}_n)] + h^2A_{kk}[f(\mathbf{U}_n{}^{(k)}) - f(\mathbf{W}_n)] \\ &= O(h^{r^*+3}) + O(h^{q+3}) + O(h^2)[\mathbf{U}_n{}^{(k)} - \mathbf{W}_n],\end{aligned}$$

which proves the second relation. []

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 $\mathbf{U}_n^{(s-k)}(t_n) - \mathbf{V}_n = O(h^{q+1})$, 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

$$\begin{aligned} u_{n+1} - y_{n+1} &= h^2\mathbf{b}_{s-k}^T(f(\mathbf{U}_n^{(s-k)}) - f(\mathbf{V}_n)) + h^2\mathbf{b}_k^T(f(\mathbf{U}_n^{(k)}) - f(\mathbf{W}_n)) \\ &= O(h^{r^*+3} + h^{q+3}) + O(h^{r^*+5} + h^{q+5}) = O(h^{r^*+3} + h^{q+3}) \\ u'_{n+1} - y'_{n+1} &= h\,\mathbf{d}_{s-k}^T(f(\mathbf{U}_n^{(s-k)}) - f(\mathbf{V}_n)) + h\,\mathbf{d}_k^T(f(\mathbf{U}_n^{(k)}) - f(\mathbf{W}_n)) \\ &= O(h^{r^*+2} + h^{q+2}) + O(h^{r^*+4} + h^{q+4}) = O(h^{r^*+2} + h^{q+2}). \end{aligned}$$

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 $\mathbf{V}_n$ ensuring that $\mathbf{U}_n^{(s-k)}(t_n) - \mathbf{V}_n = O(h^{q+1})$ are derived by replacing $\mathbf{V}_n$, y_n, $\mathbf{V}_{n-1}$, $\mathbf{W}_{n-1}$ by the exact solution values $y(t_n\mathbf{e}_{s-k} + \mathbf{c}_{s-k}h)$, $y(t_n)$, $y(t_{n-1}\mathbf{e}_{s-k} + \mathbf{c}_{s-k}h)$, $y(t_{n-1}\mathbf{e}_k + \mathbf{c}_k h)$, 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

(2.3) $$y(t_n\mathbf{e}_{s-k} + \mathbf{c}_{s-k}h) - y(t_n)\,\mathbf{v} - B_{s-k,s-k}\,y(t_{n-1}\mathbf{e}_{s-k} + \mathbf{c}_{s-k}h) - B_{s-k,k}\,y(t_{n-1}\mathbf{e}_k + \mathbf{c}_k h) = O(h^{q+1}).$$

Using (s+1)-point Lagrange interpolation formulas with abscissa vector $\mathbf{a} = (\mathbf{c}^T, 1)^T$, we obtain (see e.g. [1, p. 878])

(2.4)
$$y(t_n + th) = \sum_{j=1}^{s+1} L_j(t+1)\,y(t_{n-1} + a_j h) + C_{s+1}(t)\Big(h\frac{d}{dt}\Big)^{s+1} y(t^*),$$

$$L_j(x) := \prod_{i=1, i\neq j}^{s+1} \frac{x - a_i}{a_j - a_i}, \quad C_{s+1}(t) := \frac{1}{(s+1)!}\prod_{i=1}^{s+1}(t + 1 - a_i),$$

where t^* is a suitably chosen point in the interval containing the values t_n, $t_{n-1} + c_i h$, i = 1, ..., s+1. Hence,

(2.5a)
$$y(t_n + c_\mu h) - \sum_{j=1}^{s-k} L_j(c_\mu + 1)\,y(t_{n-1} + a_j h) - \sum_{j=s-k+1}^{s} L_j(c_\mu + 1)\,y(t_{n-1} + a_j h)$$

$$- L_{s+1}(c_\mu + 1)\,y(t_{n-1} + h) = C_{s+1}(c_\mu)\Big(h\frac{d}{dt}\Big)^{s+1} y(t_\mu^*),$$

where $t_\mu{}^*$ is a suitably chosen point in the interval containing the values t_n, $t_{n-1} + c_i h$, $i = 1, ..., s+1$, $\mu = 1, ..., s-k$. Using componentwise notation we obtain

$$\begin{aligned}
&y(t_n\mathbf{e}_{s-k} + \mathbf{c}_{s-k}h) - \big(L_1(\mathbf{c}_{s-k} + \mathbf{e}_{s-k}), ..., L_{s-k}(\mathbf{c}_{s-k} + \mathbf{e}_{s-k})\big)\, y(t_{n-1}\mathbf{e}_{s-k} + \mathbf{c}_{s-k}h) \\
&- \big(L_{s-k+1}(\mathbf{c}_{s-k} + \mathbf{e}_{s-k}), ..., L_s(\mathbf{c}_{s-k} + \mathbf{e}_{s-k})\big)\, y(t_{n-1}\mathbf{e}_k + \mathbf{c}_k h) - L_{s+1}(\mathbf{c}_{s-k} + \mathbf{e}_{s-k})\, y(t_n) \qquad (2.5b) \\
&= C_{s+1}(\mathbf{c}_{s-k})\Big(h\frac{d}{dt}\Big)^{s+1} y(\mathbf{t}^*),
\end{aligned}$$

where $\mathbf{t}^* = (t_1{}^*, ..., t_{s-k}{}^*)^T$. By defining

$$\begin{aligned}
&B_{s-k,s-k} := \big(L_1(\mathbf{c}_{s-k} + \mathbf{e}_{s-k}), ..., L_{s-k}(\mathbf{c}_{s-k} + \mathbf{e}_{s-k})\big), \\
&B_{s-k,k} := \big(L_{s-k+1}(\mathbf{c}_{s-k} + \mathbf{e}_{s-k}), ..., L_s(\mathbf{c}_{s-k} + \mathbf{e}_{s-k})\big), \quad \mathbf{v} := L_{s+1}(\mathbf{c}_{s-k} + \mathbf{e}_{s-k}),
\end{aligned} \qquad (2.6)$$

a comparison with (2.4) reveals that we achieve $q = s$ for any set of collocation points, and $q = s+1$ if $C_{s+1}(\mathbf{c}_{s-k})$ 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

$$\mathbf{Y}_n = R\mathbf{Y}_{n-1} + hS\mathbf{Y}_{n-1} + hPf(\mathbf{Y}_n) + h^2Qf(\mathbf{Y}_n), \qquad (2.7)$$

where $\mathbf{Y}_n := (\mathbf{V}_n, \mathbf{W}_n, y_{n+1}, y'_{n+1})^T$, and P, Q, R, S are all (s+2)-by-(s+2) matrices given by

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

$$P = \begin{pmatrix} O_{s-k,s-k} & O_{s-k,k} & \mathbf{0}_{s-k} & \mathbf{0}_{s-k} \\ O_{k,s-k} & O_{k,k} & \mathbf{0}_k & \mathbf{0}_k \\ \mathbf{0}_{s-k}{}^T & \mathbf{0}_k{}^T & 0 & 0 \\ \mathbf{d}_{s-k}{}^T & \mathbf{d}_k{}^T & 0 & 0 \end{pmatrix}, \quad Q = \begin{pmatrix} O_{s-k,s-k} & O_{s-k,k} & \mathbf{0}_{s-k} & \mathbf{0}_{s-k} \\ A_{k,s-k} & A_{k,k} & \mathbf{0}_k & \mathbf{0}_k \\ \mathbf{b}_{s-k}{}^T & \mathbf{b}_k{}^T & 0 & 0 \\ \mathbf{0}_{s-k}{}^T & \mathbf{0}_k{}^T & 0 & 0 \end{pmatrix},$$

and where $O_{i,j}$ and $\mathbf{0}_i$ 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 abscissas 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 2.1. Spectral radius $\rho(B_{s-k,s-k})$ of Gauss-Legendre based TRKN methods.

(s,k) =	(3,2)	(4,3)	(4,2)	(5,4)	(5,3)	(5,2)
$\rho(B_{s-k,s-k}) \approx$	.059	.023	3.05	.011	1.72	47.7

A second option minimizes the principal error vector associated with the extrapolation formula for the vector $\mathbf{V}_n$, i.e., the vector

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

where according to (2.4)

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

This vector vanishes if the set of components of the collocation vector $\mathbf{c}$ contains the set of components of the vector $\mathbf{c}_{s-k} + \mathbf{e}_{s-k}$. 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

Theorem 2.2. If the components of the collocation vector $\mathbf{c}$ contain the components of the vector $\mathbf{c}_{s-k} + \mathbf{e}_{s-k}$, then the associated TRKN method is zero-stable. []

3. Parallel iterated TRKN methods

Using (2.2) as corrector formula with predictor formula

$$\mathbf{W}_n^{(0)} = y_n\mathbf{w} + C_{k,s-k}\mathbf{V}_{n-1} + C_{kk}\mathbf{W}_{n-1}^{(m)}, \tag{3.1a}$$

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

$$\begin{aligned} \mathbf{V}_n &= y_n\mathbf{v} + B_{s-k,s-k}\mathbf{V}_{n-1} + B_{s-k,k}\mathbf{W}_{n-1}^{(m)}, \\ \mathbf{W}_n^{(j)} &= y_n\mathbf{e}_k + h\mathbf{c}_k y'_n + h^2A_{k,s-k}f(\mathbf{V}_n) + h^2A_{kk}f(\mathbf{W}_n^{(j-1)}),\ \ j = 1, \ldots, m, \\ y_{n+1} &= y_n + hy'_n + h^2\mathbf{b}_{s-k}^Tf(\mathbf{V}_n) + h^2\mathbf{b}_k^Tf(\mathbf{W}_n^{(m)}), \\ y'_{n+1} &= y'_n + h\mathbf{d}_{s-k}^Tf(\mathbf{V}_n) + h\mathbf{d}_k^Tf(\mathbf{W}_n^{(m)}). \end{aligned} \tag{3.1b}$$

The computational costs are measured by the number of sequential righthand side evaluations (f-evaluations) per step (notice that the (s-k) and k components of the vectors $f(\mathbf{V}_n)$ and $f(\mathbf{W}_n^{(j-1)})$ 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 $f(\mathbf{V}_n)$ 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 Subsection 2.1 we can prove that the conditions

$$\begin{aligned} C_{k,s-k} &= (L_1(\mathbf{c}_k + \mathbf{e}_k), \ldots, L_{s-k}(\mathbf{c}_k + \mathbf{e}_k)), \\ C_{k,k} &= (L_{s-k+1}(\mathbf{c}_k + \mathbf{e}_k), \ldots, L_s(\mathbf{c}_k + \mathbf{e}_k)), \quad \mathbf{w} = L_{s+1}(\mathbf{c}_k + \mathbf{e}_k) \end{aligned} \tag{3.2}$$

imply that

$$\mathbf{W}(t_n) - \mathbf{W}_n^{(0)} = O(h^{s+1}). \tag{3.3}$$

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

$$\mathbf{W}_n - \mathbf{W}_n^{(m)} = O(h^{2m+s+1}),$$

$$u_{n+1} - y_{n+1} = h^2\mathbf{b}_k^T[f(\mathbf{W}_n) - f(\mathbf{W}_n^{(m)})] = O(h^{2m+s+3}),$$

$$u'_{n+1} - y'_{n+1} = h\,\mathbf{d}_k^T[f(\mathbf{W}_n) - f(\mathbf{W}_n^{(m)})] = O(h^{2m+s+2}).$$

Thus, we have

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 [15], [11] and [16]. Using the model test equation $y''(t) = \lambda y(t)$, where λ runs through the eigenvalues of the Jacobian matrix $\partial\mathbf{f}/\partial\mathbf{y}$, we obtain the iteration error equation

$$\mathbf{W}_n^{(j)} - \mathbf{W}_n = zA_{kk}\,[\mathbf{W}_n^{(j-1)} - \mathbf{W}_n], \quad z := \lambda h^2, \quad j = 1, \ldots, m.$$

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

$$|z| < \frac{1}{\rho(A_{kk})} \quad \text{or} \quad h^2 < \frac{1}{\rho(A_{kk})\,\rho(\partial\mathbf{f}/\partial\mathbf{y})}.$$

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

$$\mathbf{E}_{s+1} = (0, \ldots, 0, 1, 0)^T,\ \mathbf{E}_{s+2} = (0, \ldots, 0, 1)^T,\ \mathbf{S}_{s+2} = (0, \ldots, 0, 1, 1)^T$$

and the matrices

$$\begin{aligned}
&Q_{s-k,s+2} = \mathbf{v}\ \mathbf{E}_{s+1}{}^T + B_{s-k,s-k}(I_{s-k,s-k}, O_{s-k,k+2}) + B_{s-k,k}(O_{k,s-k}, I_{kk}, O_{k,2}),\\
(3.4)\qquad &P_{k,s+2} = \mathbf{w}\ \mathbf{E}_{s+1}{}^T + C_{k,s-k}(I_{s-k,s-k}, O_{s-k,k+2}) + C_{kk}(O_{k,s-k}, I_{kk}, O_{k,2}),\\
&R_{k,s+2} = \mathbf{e}_k\mathbf{E}_{s+1}{}^T + \mathbf{c}_k\mathbf{E}_{s+2}{}^T,
\end{aligned}$$

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 λ is assumed to be negative. Defining

$$\mathbf{X}_n{}^{(m)} = ((\mathbf{V}_{n-1})^T, (\mathbf{W}_{n-1}{}^{(m)})^T, y_n, hy'_n)^T,$$

and using (3.4) we obtain

(3.5a) $\quad \mathbf{V}_n = Q_{s-k,s+2}\mathbf{X}_n{}^{(m)}$

(3.5b)
$$\begin{aligned}
&\mathbf{W}_n{}^{(m)} = (R_{k,s+2} + zA_{k,s-k}Q_{s-k,s+2})\mathbf{X}_n{}^{(m)} + zA_{kk}\mathbf{W}_n{}^{(m-1)} =\\
&(I + zA_{kk} + \ldots + (zA_{kk})^{m-1})(R_{k,s+2} + zA_{k,s-k}Q_{s-k,s+2})\mathbf{X}_n{}^{(m)} + (zA_{kk})^m P_{k,s+2}\mathbf{X}_n{}^{(m)} =\\
&[(I - zA_{kk})^{-1}(I - (zA_{kk})^m)(R_{k,s+2} + zA_{k,s-k}Q_{s-k,s+2}) + (zA_{kk})^m P_{k,s+2}]\mathbf{X}_n{}^{(m)},
\end{aligned}$$

(3.5c)
$$\begin{aligned}
&y_{n+1} = y_n + hy'_n + z\mathbf{b}_{s-k}{}^T\mathbf{V}_n + z\mathbf{b}_k{}^T\mathbf{W}_n{}^{(m)} =\\
&\mathbf{S}_{s+2}{}^T\mathbf{X}_n{}^{(m)} + z\mathbf{b}_{s-k}{}^T Q_{s-k,s+2}\mathbf{X}_n{}^{(m)} +\\
&z\mathbf{b}_k{}^T((I - zA_{kk})^{-1}(I - (zA_{kk})^m)(R_{k,s+2} + zA_{k,s-k}Q_{s-k,s+2}) + (zA_{kk})^m P_{k,s+2})\mathbf{X}_n{}^{(m)} =\\
&[\mathbf{S}_{s+2}{}^T + z\mathbf{b}_{s-k}{}^T Q_{s-k,s+2} + z\mathbf{b}_k{}^T((I-zA_{kk})^{-1}(I-(zA_{kk})^m)(R_{k,s+2}+zA_{k,s-k}Q_{s-k,s+2})+(zA_{kk})^m P_{k,s+2})]\mathbf{X}_n{}^{(m)},
\end{aligned}$$

(3.5d)
$$\begin{aligned}
&hy'_{n+1} = hy'_n + z\mathbf{d}_{s-k}{}^T\mathbf{V}_n + z\mathbf{d}_k{}^T\mathbf{W}_n{}^{(m)} =\\
&\mathbf{E}_{s+2}{}^T\mathbf{X}_n{}^{(m)} + z\mathbf{d}_{s-k}{}^T Q_{s-k,s+2}\mathbf{X}_n{}^{(m)} +\\
&z\mathbf{d}_k{}^T((I - zA_{kk})^{-1}(I - (zA_{kk})^m)(R_{k,s+2} + zA_{k,s-k}Q_{s-k,s+2}) + (zA_{kk})^m P_{k,s+2})\mathbf{X}_n{}^{(m)} =\\
&[\mathbf{E}_{s+2}{}^T + z\mathbf{d}_{s-k}{}^T Q_{s-k,s+2} + z\mathbf{d}_k{}^T((I-zA_{kk})^{-1}(I-(zA_{kk})^m)(R_{k,s+2}+zA_{k,s-k}Q_{s-k,s+2})+(zA_{kk})^m P_{k,s+2})]\mathbf{X}_n{}^{(m)}.
\end{aligned}$$

By introducing the matrices

$$\begin{aligned}
&M_{k,s+2}(z) = (I - zA_{kk})^{-1}(I - (zA_{kk})^m)(R_{k,s+2} + zA_{k,s-k}Q_{s-k,s+2}) + (zA_{kk})^m P_{k,s+2},\\
&\mathbf{M}_{s+2}(z) = \mathbf{S}_{s+2}{}^T + z\mathbf{b}_{s-k}{}^T Q_{s-k,s+2} + z\mathbf{b}_k{}^T((I-zA_{kk})^{-1}(I-(zA_{kk})^m)(R_{k,s+2}+zA_{k,s-k}Q_{s\ k,s+2})+(zA_{kk})^m P_{k,s+2}),\\
&\mathbf{M}^*{}_{s+2}(z) = \mathbf{E}_{s+2}{}^T + z\mathbf{d}_{s-k}{}^T Q_{s-k,s+2} + z\mathbf{d}_k{}^T((I-zA_{kk})^{-1}(I-(zA_{kk})^m)(R_{k,s+2}+zA_{k,s-k}Q_{s-k,s+2})+(zA_{kk})^m P_{k,s+2}),
\end{aligned}$$

the relations (3.5) yield the recursion

$$\text{(3.6)} \qquad X_{n+1}^{(m)} = M_m(z)\, X_n^{(m)}, \quad M_m(z) = \begin{pmatrix} Q_{s-k,s+2} \\ M_{k,s+2}(z) \\ M_{s+2}(z) \\ M^*_{s+2}(z) \end{pmatrix}.$$

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

$$(-\beta(m), 0) := \{z: \rho(M_m(z)) < 1, \ z \le 0\}.$$

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

$$\text{(4.1)} \qquad \mathbf{c} = (\mathbf{c}_{s-k}^T, \mathbf{c}_k^T)^T, \quad \mathbf{c}_{s-k} = (-c_k, ..., -c_1)^T, \quad \mathbf{c}_k = (c_1, ..., c_k)^T, \quad k = 2, ..., 5,$$

where $c_1, ..., c_k$ 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 $C_{s+1}(\mathbf{c}_{s-k})$ 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 [15] and [19]. However, a direct numerical computation reveals that the convergence factor as defined in Subsection 3.2 is much smaller than that of PIRKN methods (see Table 4.1).

Table 4.1. Convergence factors for various pth-order PITRKN and PIRKN methods

Parallel pth-order PC methods	p = 4	p = 6	p = 8	p = 10
Direct PIRKN methods (cf. [15])	0.048	0.029	0.018	0.013
Indirect PIRKN methods (cf. [15])	0.083	0.046	0.027	0.019
PITRKN methods	**0.026**	**0.015**	**0.009**	**0.006**

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

Table 4.2. Stability boundaries $\beta(m)$ for various pth-order PITRKN methods

pth-order PITRKN methods	p = 4	p = 6	p = 8	p = 10
m = 1	0.42	0.09	0.00	0.00
m = 2	4.15	1.37	0.51	0.10
m = 3	7.93	7.07	2.54	1.13
m = 4	8.50	16.20	7.48	3.74

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

$$(4.2) \qquad \| \mathbf{W}_n^{(m)} - \mathbf{W}_n^{(m-1)} \|_\infty \leq \mathrm{TOL} = C\,h^{p-1},$$

where p is 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 [19] 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_{seq} denotes the total number of sequential f-evaluations, and N_{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 pth-order PC methods to the linear problem (cf. [15])

$$(4.3) \qquad \frac{d^2y(t)}{dt^2} = \begin{pmatrix} -2\alpha(t)+1 & -\alpha(t)+1 \\ 2(\alpha(t)-1) & \alpha(t)-2 \end{pmatrix} y(t), \quad \alpha(t) = \max(2\cos^2(t), \sin^2(t)), \quad 0 \leq t \leq 20,$$

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 4.3. Values of NCD / N_{seq} for problem (4.3) obtained by various pth-order parallel PC methods.

pth-order PC methods	p	N_{steps}=80	N_{steps}=160	N_{steps}=320	N_{steps}=640	N_{steps}=1280	C
PIRKN	4	4.0 / 237	5.3 / 477	6.5 / 958	7.7 / 1919	8.9 / 3836	10^{-1}
PITRKN	4	4.8 / 161	6.2 / 321	7.5 / 641	8.7 / 1281	10.0 / 2561	10^{-1}
PIRKN	6	7.4 / 320	9.2 / 640	11.0 / 1280	12.8 / 2559	14.6 / 5119	10^{-3}
PITRKN	6	8.2 / 163	10.5 / 322	12.5 / 642	14.4 / 1282	16.2 / 2562	10^{-3}
PIRKN	8	11.0 / 399	13.4 / 799	15.8 / 1600	18.2 / 3198	20.6 / 6398	10^{-4}
PITRKN	8	12.1 / 211	14.2 / 380	17.9 / 683	20.2 / 1283	22.8 / 2563	10^{-4}
PIRKN	10	13.3 / 436	18.0 / 921	20.9 / 1881	23.8 / 3803		10^{-4}
PITRKN	10	14.2 / 233	17.3 / 407	20.3 / 750	24.1 / 1403		10^{-4}

4.1.2. Nonlinear Fehlberg problem

For the second numerical example, we consider the often-used orbit equation (cf. e.g. [2], [3], [5], [6])

$$(4.4)\qquad \frac{d^2y(t)}{dt^2} = \begin{pmatrix} -4t^2 & -2/r(t) \\ 2/r(t) & -4t^2 \end{pmatrix} y(t),\quad r(t) = \sqrt{y_1^2(t) + y_2^2(t)},\quad \sqrt{\pi/2} \le t \le 10.$$

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.

Table 4.4. Values of NCD / N_{seq} for problem (4.4) obtained by various pth-order parallel PC methods.

pth-order PC methods	p	N_{steps}=200	N_{steps}=400	N_{steps}=800	N_{steps}=1600	N_{steps}=3200	C
PIRKN	4	1.6 / 591	2.8 / 1197	4.0 / 2400	5.2 / 4800	6.4 / 9600	10^2
PITRKN	4	2.7 /441	3.8 / 802	5.1 / 1601	6.4 / 3201	7.6 / 6401	10^2
PIRKN	6	4.0 / 775	5.8 / 1532	7.6 / 3096	9.4 / 6257	11.2 / 12648	10^3
PITRKN	6	5.3 / 495	7.1 / 880	9.0 / 1601	11.0 / 3201	12.9 / 6401	10^3
PIRKN	8	6.6 / 1022	9.0 / 2032	11.5 / 4028	13.9 / 7966	16.3 / 15725	10^3
PITRKN	8	8.7 / 575	11.1 / 1051	13.5 / 1988	15.9 / 3672	18.3 / 6616	10^3
PIRKN	10	9.4 /1234	12.4 / 2458	15.5 / 4893	18.5 / 9734	21.5 / 19332	10^3
PITRKN	10	11.4 / 674	14.5 / 1156	18.1 / 2139	21.1 / 4094	23.8 / 7797	10^3

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 [18], p. 245):

$$(4.5)\qquad \begin{aligned} &\frac{d^2y_1(t)}{dt^2} = -\frac{y_1(t)}{(r(t))^3},\quad \frac{d^2y_2(t)}{dt^2} = -\frac{y_2(t)}{(r(t))^3},\quad 0 \le t \le 20, \\ &y_1(0) = 1-\varepsilon,\ y_2(0) = 0,\quad y'_1(0) = 0,\ y'_2(0) = \sqrt{\frac{1+\varepsilon}{1-\varepsilon}} \end{aligned}$$

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 ε 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 NCD / N_{seq} for problem (4.5) obtained by various pth-order parallel PC methods.

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

4.2. Comparison with sequential methods

In Subsection 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 [2], [3], [5], [6] and the RKN code DOPRIN taken from [10]. We reproduced the best results obtained by these sequential methods given in the literature (cf. e.g. [6], [19]) 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)

Methods	N_{steps}	NCD	N_{seq}
11(10) pair (from [6])	919	20.7	15614
8(9)-pair (from [2])	1452	13.5	15973
9(10)-pair (from [3])	628	15.1	8793
	3235	21.4	45291
11(12)-pair (from [5])	876	20.3	17521
DOPRIN (from [10])	1208	12.3	9665
	4466	16.3	35729
	16667	20.3	133337
$PITRKN_{10}$ (in this paper)	200	11.4	674
	400	14.5	1156
	800	18.1	2139
	1600	21.1	4094

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.

Acknowledgement

The author is grateful to Prof. Dr. P.J. van der Houwen and Dr. B.P. Sommeijer for their help during the preparation of this paper.

References

[1] **Abramowitz, M. & Stegun, I. A. (1970)**: *Handbook of Mathematical functions*, National Bureau of Standards Applied Mathematics Series 55, Dover, New York.

[2] **Fehlberg, E. (1972)**: *Klassische Runge-Kutta-Nyström Formeln mit Schrittweiten-Kontrolle für Differentialgleichungen $x''=f(t, x)$*, Computing **10**, 305-315.

[3] **Fehlberg, E. (1981)**: *Eine Runge-Kutta-Nyström formel 9-ter Ordnung mit Schrittweitenkoltrolle für Differentialgleichungen $x'' = f(t, x)$*, Z. Angew. Math. Mech. **61**, 477-485.

[4] **Fehlberg, E., Filippi, S. & Gräf J. (1986)**: *Eine Runge-Kutta-Nyström formelpaar der Ordnung 10(11) für Differentialgleichungen $y'' = f(t, x)$*, Z. Angew. Math. Mech. **66**, 265-270.

[5] **Filippi, S. & Gräf J. (1985)**:*Eine Runge-Kutta-Nyström formelpaar der Ordnung 11(12) Differentialgleichungen der form $y'' = f(t, x)$*, Computing **34**, 271-282.

[6] **Filippi, S. & Gräf J. (1986)**: *New Runge-Kutta-Nyström formula-pairs of order 8(7), 9(8) 10(9) and 11(10) for differential equations of the form $y'' = f(t, x)$*, J. Comp. Appl Math. **14**, 361-370.

[7] **Glasmacher, W. & Sommer, D. (1966)**: *Implizite Runge-Kutta-Formeln*, Westdeutscher Verlag, Köln und Opladen.

[8] **Hairer, E. (1977)**: *Methodes de Nyström pour l'équation différentielle $y''(t) = f(x,y)$*, Numer. Math. **27**, 283-300.

[9] **Hairer, E. (1982)**: *A one-step method of order 10 for $y'' = f(x,y)$*, IMA J. Numer. Anal. **2**, 83-94.

[10] **Hairer, E., Nørsett, S. P., & Wanner, G. (1987)**: *Solving Ordinary Differential Equations, I. Nonstiff Problems*, Springer-Verlag, Berlin.

[11] **Houwen, P.J. van der & Nguyen huu Cong (1993)**: *Parallel block predictor-corrector methods of Runge-Kutta type* , Appl. Numer. Math. **13**, 109-123.

[12] **Houwen, P.J. van der, Sommeijer, B.P. & Nguyen huu Cong (1991)**: *Stability of collocation-based Runge-Kutta-Nyström methods*, BIT **31**, 469-481.

[13] **Houwen, P.J. van der, Sommeijer, B.P. & Nguyen huu Cong (1992)**: *Parallel diagonally implicit Runge-Kutta-Nyström methods,* Appl. Numer. Math. **9**, 111-131.

[14] **Nguyen huu Cong (1993)**: *A-stable diagonally implicit Runge-Kutta-Nyström methods for parallel computers*, Numerical Algorithms **4**, 263-281.

[15] **Nguyen huu Cong (1993)**: *Note on performance of direct and indirect Runge-Kutta-Nyström methods*, J. Comp. Appl. Math. **45**, 347-355.

[16] **Nguyen huu Cong (1993)**: *Parallel iteration of symmetric Runge-Kutta methods for nonstiff problems,* to appear in J. Comp. Appl. Math.

[17] **Nguyen huu Cong**: *Highly parallel predictor-corrector methods of Runge-Kutta-Nyström type*, in preparation.

[18] **Shampine, L. F. & Gordon, M. K. (1975)**: *Computer solution of Ordinary Differential Equations, The Initial Value Problem*, W. H. Freeman and company, San Francisco.

[19] **Sommeijer, B.P. (1993)**: *Explicit, high-order Runge-Kutta-Nyström methods for parallel computers*, Appl. Numer. Math. **13**, 221-240.