Parallel iteration of symmetric Runge-Kutta methods for nonstiff initial value problems

Nguyen huu Cong

Department of Numerical Mathematics

Report NM-R9320 November 1993

CWI is the National Research Institute for Mathematics and Computer Science. CWI is part of the Stichting Mathematisch Centrum (SMC), the Dutch foundation for promotion of mathematics and computer science and their applications.
SMC is sponsored by the Netherlands Organization for Scientific Research (NWO). CWI is a member of ERCIM, the European Research Consortium for Informatics and Mathematics.

P.O. Box 94079, 1090 GB Amsterdam (NL)
Kruislaan 413, 1098 SJ Amsterdam (NL)
Telephone +31 20 592 9333
Telefax +31 20 592 4199

Parallel Iteration of Symmetric Runge-Kutta Methods for Nonstiff Initial Value Problems

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

This paper discusses parallel iteration schemes for collocation-based, symmetric Runge-Kutta (SRK) methods for solving nonstiff initial-value problems. Our main result is the derivation of four A-stable SRK corrector methods of orders 4, 6, 8, and 10 that optimize the rate of convergence when iterated by means of the highly parallel fixed point iteration process. The resulting PISRK method (parallel iterated SRK method) shows considerably increased efficiency when compared with fixed point iteration process applied to Gauss-Legendre correctors.

AMS Subject Classification (1991): 65M12, 65M20
CR Subject Classification: G.1.7
Keywords and Phrases: Runge-Kutta 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 CWI in Amsterdam.

1. Introduction

In the literature, a number of parallel numerical methods have been proposed to solve the initial-value problem (IVP) for the system of nonstiff first-order ordinary differential equations (ODEs)

$$\frac{d\mathbf{y}(t)}{dt} = f(\mathbf{y}(t)). \tag{1.1}$$

Most of them are based on the highly parallel fixed point iteration (or predictor-corrector iteration) using a Runge-Kutta (RK) corrector already available in the literature (e.g. the Gauss-Legendre methods (cf. [6], [8], [10])). These correctors possess a high-order of accuracy and excellent stability properties for generating parallel methods. In the present paper, we propose a new class of symmetric RK methods of collocation type, to be called SRK methods, in which the abscissas are chosen such that the RK matrix has a minimized spectral radius. This property leads to improved rate of convergence when applying the parallel iteration scheme. Like the conventional Gauss-Legendre methods, the resulting SRK methods are A-stable (cf. Subsection 3.3). However, the particular location of the abscissas decreases the order of accuracy of the SRK methods when compared with the Gauss-Legendre methods. To be more precise, in general, an s-stage SRK method is of order $p = s$ or $p = s+1$ depending on whether s is even or odd, whereas an s-stage Gauss-Legendre method has order $p = 2s$. On a sequential computer, this would be a serious sacrifice, because for a given order p, the increased number of stages of the SRK correctors increases the computational work per iteration considerably. But on parallel computers, the *sequential* computational work is independent of the number of stages.

The parallel iterated SRK methods (PISRK methods) developed in this paper have the same predictor-corrector nature as the parallel iterated RK methods (PIRK methods) proposed in [6] and the block PIRK methods (BPIRK methods) of [5]. The predictor formula is based on extrapolation of preceding stage and steppoint values (cf. Subsection 3.1). Stability investigations reveal that the PISRK methods have sufficiently large stability regions for nonstiff problems (see Subsection 3.3). In Section 4, we compare the efficiency of PISRK methods with that of the PIRK and the BPIRK methods by means of a number of numerical experiments. These comparisons show that for a given order

Report NM-R9320
ISSN 0169-0388
CWI
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands

of accuracy, the efficiency of the PISRK methods is much higher than the efficiency of the PIRK methods and comparable with or superior to that of the BPIRK methods. If we take into account that PISRK methods need much less processors for their implementation than needed by the BPIRK methods, we conclude that the PISRK methods are more attractive than the BPIRK methods.

2. Symmetric RK methods

In this section, we construct various symmetric RK methods that will serve as correctors for the parallel iteration scheme. For simplicity of notation, we assume that the equation (1.1) is an autonomous, scalar equation. However, all considerations below can be straightforwardly extended to a system of ODEs, and therefore, also to nonautonomous equations. For autonomous, scalar equations, the general s-stage RK method then assumes the form

$$\mathbf{Y}_n = \mathbf{e}y_n + hA\mathbf{f}(\mathbf{Y}_n), \quad y_{n+1} = y_n + h\mathbf{b}^T\mathbf{f}(\mathbf{Y}_n), \tag{2.1}$$

where A is an s-by-s matrix, **b** and **e** are s-dimensional vectors, **e** is the vector with unit entries, and $\mathbf{Y}_n$ is the stage vector corresponding to the n-th step. Furthermore, we use the convention that for any given vector $\mathbf{v} = (v_j)$, $\mathbf{f}(\mathbf{v})$ denotes the vector with entries $f(v_j)$. From now on, we assume that the RK method (2.1) is a collocation method based on symmetrically distributed, distinct collocation points (that is, the vector $\mathbf{c} = A\mathbf{e}$ is such that the abscissas $t_n + c_i h$ are symmetric with respect to $t_n + h/2$). These RK methods will be referred to as *SRK methods*. They form a special family of the class of symmetric RK methods (cf. [4] p. 217]).

The collocation principle ensures that the SRK method is of at least order p = s. The order can be increased by satisfying the orthogonality relation (cf., e.g. [4] p. 207)

$$\int_0^1 \prod_{i=1}^{s} (x - c_i)\, x^{j-1} dx = 0. \tag{2.2}$$

It is easily verified that this condition is automatically satisfied for j = 1 if s is odd. Thus, we have the result:

Theorem 2.1. *An s-stage SRK method is of order p = s if s is even and of order p = s + 1 if s is odd.* []

This leads us to restrict our considerations to SRK methods with an odd number of stages.

In Section 3, it will turn out that it is convenient to iterate A-stable SRK correctors. Therefore, we now briefly discuss the A-stability of SRK methods. It is well known that RK methods are A-stable if the stability function is analytic in the left-half plane $C^- := \{z \in C: \mathrm{Re}(z) < 0\}$ (i.e., if the eigenvalues of the matrix A lie in the right-half plane $C^+ := \{z \in C: \mathrm{Re}(z) > 0\}$) and if it is bounded by 1 on the imaginary axis. Since SRK methods possess stability functions of modulus 1 along the imaginary axis, we have the result:

Theorem 2.2. *An s-stage SRK method is A-stable if A has its eigenvalues in the right-half plane.* []

3. Parallel-iterated SRK methods

Starting with the RK method (2.1), we consider the following fixed-point iteration scheme

$$\mathbf{Y}_n^{(j)} = \mathbf{e}y_n + hA\mathbf{f}(\mathbf{Y}_n^{(j-1)}), \quad j = 1, \ldots, m, \tag{3.1b}$$

$$y_{n+1} = y_n + h\mathbf{b}^T\mathbf{f}(\mathbf{Y}_n^{(m)}). \tag{3.1c}$$

By using information from the preceding step, that is, the values of y_n and the stage vector $\mathbf{Y}_{n-1}^{(m)}$, we may define a predictor formula of the form

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

where V is an s-by-s matrix and **w** is an s-dimensional vector, both determined by order conditions (see Subsection 3.1). Notice that the s components of the vectors $\mathbf{f}(\mathbf{Y}_n^{(j)})$ can be computed in parallel, provided that s processors are available. Hence, the computational time needed for one iteration of (3.1b) is equivalent to the time required to evaluate one right-hand side function f on a sequential computer. Thus, in (3.1) the number of sequential evaluations of f per step of length h equals m+1.

Regarding the prediction formula (3.1a) as the predictor method and (2.1) as the corrector method, (3.1) may be considered as a conventional predictor-corrector (PC) method (in $P(EC)^mE$ mode). This parallel PC method (3.1) is of the same nature as the PIRK methods (parallel iterated RK methods) considered in [6], and only differs by its predictor (3.1a) and the underlying SRK corrector. In analogy with the PIRK methods, the method (3.1) will be called a *PISRK method* (parallel iterated SRK method).

3.1. Order conditions for the predictor method

The order conditions for the predictor formula (3.1a) can be derived straightforwardly using Taylor expansions. We obtain an order s predictor if

$$(3.2) \qquad \frac{1}{j!}\left[(\mathbf{c} + \mathbf{e})^j - (V, \mathbf{w})\, \mathbf{a}^j \right] = \mathbf{0}, \quad \mathbf{a} := (\mathbf{c}^T, 1)^T, \; j = 0, 1, ..., s.$$

These conditions determine the matrix $(V, \mathbf{w})$. In order to express $(V, \mathbf{w})$ explicitly in terms of $\mathbf{c}$, we define the s-by-(s+1) and (s+1)-by-(s+1) matrices P and Q

$$(3.3a) \qquad P = (\mathbf{e}, (\mathbf{c} + \mathbf{e}), (\mathbf{c} + \mathbf{e})^2, ..., (\mathbf{c} + \mathbf{e})^s), \quad Q = (\mathbf{e}^*, \mathbf{a}, \mathbf{a}^2, ..., \mathbf{a}^s),$$

where $\mathbf{e}^*$ is the (s+1)-dimensional vector with unit entries. Condition (3.2) can be written in the form $P - (V,\mathbf{w})Q = O$, where O is s-by-(s+1) matrix with zero entries. Since the abscissas c_j are assumed to be distinct, we can write

$$(3.3b) \qquad (V, \mathbf{w}) = PQ^{-1}.$$

If (3.3) is satisfied, then the iteration errors associated with the stage vector and step point value satisfy the order relations

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

where u_{n+1} denotes the corrector solution at the step point t_{n+1}. The local truncation error of PISRK methods can be written as the sum of the truncation error of the SRK corrector and the iteration error of the PISRK method:

$$y(t_{n+1}) - y_{n+1} = (y(t_{n+1}) - u_{n+1}) + (u_{n+1} - y_{n+1}) = O(h^{p+1}) + O(h^{m+s+2}) = O(h^{p^*+1}),$$

where p is the order of the SRK corrector, $p^* = \min(p, m+ s+1)$. Thus, we have:

Theorem 3.1. *If the generating SRK corrector method (2.1) is of order p and if (V, w) is defined by (3.3), then on s-processor computers the PISRK method (3.1) represents an explicit method of order $p^* = \min[p, m+s+1)$ requiring m+1 sequential right-hand side evaluations per step.* []

3.2. Construction of SRK corrector methods

In this subsection we concentrate on SRK methods with an odd number of implicit stages (s = 3, 5, 7, 9) and we will construct SRK correctors such that the corresponding PISRK methods have maximized rates of convergence. The rate of convergence of PISRK methods is defined by using the model test equation $y' = \lambda y$, where λ runs through the eigenvalues of the Jacobian matrix $\partial \mathbf{f}/\partial \mathbf{y}$ (cf. [5], [9]). For this equation, we obtain the iteration error equation

$$\mathbf{Y}_n^{(j)} - \mathbf{Y}_n = zA\,[\mathbf{Y}_n^{(j-1)} - \mathbf{Y}_n], \quad z := \lambda h, \quad j = 1, ..., m.$$

Hence, with respect to the model test equation, the rate of convergence is determined by the spectral radius $\rho(A)$ of the matrix A. We shall call $\rho(A)$ the *convergence factor* of the PISRK method. By requiring that $\rho(zA) < 1$, we are led to the convergence condition

$$(3.4) \qquad |z| < \frac{1}{\rho(A)} \quad \text{or} \quad h \le \frac{1}{\rho(A)\,\rho(\partial \mathbf{f}/\partial \mathbf{y})}.$$

We exploit the freedom in the choice of the collocation vector $\mathbf{c}$ for SRK correctors for minimizing the convergence factor $\rho(A)$, or equivalently, for maximizing the convergence region $\{z: \rho(zA) < 1\}$. By a numerical search, we found the collocation vectors and the corresponding convergence factors as listed in Table 3.1 (the

specification of the parameters of the associated SRK corrector methods can be found in the Appendix). Table 3.1 also lists the convergence factors for the Gauss-Legendre based PIRK methods (we note that PIRK and BPIRK methods have identical convergence factors). From these figures, we see that the convergence factors of the PISRK methods are substantially smaller than those of the PIRK methods of the same order.

Table 3.1. SRK collocation points and convergence factors of PISRK and (B)PIRK methods

Order	c_1	c_2	c_3	c_4	Convergence factors PISRK	(B)PIRK
p = 4	0.10300662				0.198	0.289
p = 6	0.04101173	0.21235714			0.123	0.215
p = 8	0.02180707	0.11383597	0.27544350		0.089	0.165
p = 10	0.01348800	0.07067122	0.17189713	0.31496835	0.070	0.137

3.3. Stability of PISRK methods

A numerical computation of the spectrum of the matrix A defining the SRK methods derived above shows that all the eigenvalues are lying in the right-half plane. Therefore, in view of Theorem 2.2, these SRK methods are A-stable. Evidently, when iterating until convergence, the stability region of the PISRK method is given by the intersection of its convergence region $\{z: |z| < 1 / \rho(A)\}$ and the stability region of the corrector. Hence, by virtue of the A-stability, we achieve that by maximizing the region of convergence, we have in fact maximized the region of stability. However, in actual computation, we often do not iterate until convergence, so that it is of interest to determine the stability regions as a function of m.

Applying (3.1) to the model test equation, we obtain

(3.5a) $$\mathbf{Y}_n^{(m)} = \mathbf{e}y_n + zA\mathbf{Y}_n^{(m-1)} = (I + zA + (zA)^2 + \ldots + (zA)^{m-1})\mathbf{e}y_n + (zA)^m\mathbf{Y}_n^{(0)}$$

$$= (zA)^m V\mathbf{Y}_{n-1}^{(m)} + ((I - zA)^{-1}(I - (zA)^m)\mathbf{e} + (zA)^m\mathbf{w})y_n$$

(3.5b) $$y_{n+1} = y_n + z\mathbf{b}^T\mathbf{Y}_n^{(m)}$$

$$= z\mathbf{b}^T(zA)^m V\mathbf{Y}_{n-1}^{(m)} + (1 + z\mathbf{b}^T((zA)^m\mathbf{w} + (I - zA)^{-1}(I - (zA)^m)\mathbf{e}))y_n.$$

From (3.5) we obtain the recursion

(3.6) $$\begin{pmatrix}\mathbf{Y}_n^{(m)}\\ y_{n+1}\end{pmatrix} = M_m(z)\begin{pmatrix}\mathbf{Y}_{n-1}^{(m)}\\ y_n\end{pmatrix},\quad M_m(z) = \begin{pmatrix}(zA)^m V & (zA)^m\mathbf{w} + (I - zA)^{-1}(I - (zA)^m)\mathbf{e}\\ z\mathbf{b}^T(zA)^m V & 1 + z\mathbf{b}^T((zA)^m\mathbf{w} + (I - zA)^{-1}(I - (zA)^m)\mathbf{e})\end{pmatrix}.$$

Similar to the stability considerations of block PIRK methods (cf. [5]), the (s+1)-by-(s+1) matrix $M_m(z)$ will be called the *amplification matrix*, and its spectral radius $\rho(M_m(z))$ the stability function. Notice that $\rho(M_m(z))$ converges to the stability function of the corrector method as $m \to \infty$, if z satisfies the convergence condition (3.4).

Using the familiar definition of the real and imaginary stability boundaries $\beta_{re}(m)$ and $\beta_{im}(m)$, we computed the stability pairs $(\beta_{re}(m), \beta_{im}(m))$ as listed in Table 3.2. We observe that for small m, the stability of PISRK methods is rather poor, but for $m \geq p/2$ (say), the stability boundaries are sufficiently large for nonstiff problems. Hence, already for relatively small numbers of iterations, the PISRK method is expected to perform stably.

Table 3.2. Stability pairs $(\beta_{re}(m), \beta_{im}(m))$ for various PISRK methods

Order	m = 1	m = 2	m = 3	m = 4	m = 5
p = 4	(0.04, 0.05)	**(0.43, 0.40)**	(0.96, 0.53)	(1.52, 0.42)	(2.13, 0.42)
p = 6	(0.00, 0.00)	(0.10, 0.10)	**(0.39, 0.40)**	(0.80, 0.82)	(1.25, 1.28)
p = 8	(0.00, 0.00)	(0.02, 0.02)	(0.15, 0.16)	**(0.42, 0.42)**	(0.77, 0.78)
p = 10	(0.00, 0.00)	(0.00, 0.00)	(0.06, 0.06)	(0.21, 0.21)	**(0.46, 0.46)**

4. Numerical experiments

In this section we report numerical results obtained by the PISRK, the PIRK and the BPIRK methods. The experiments were performed on a 28-digits arithmetic computer. The absolute error obtained at the end of integration interval is presented in the form 10^{-d} (d may be interpreted as the number of correct decimal digits (NCD)). We only compared methods of the *same order*, so that the accuracies are more or less comparable.

In order to see the efficiency of the PISRK, PIRK and BPIRK methods, we applied a dynamical strategy for determining the number of iterations in the successive steps. The stopping criterion is defined by

$$\| \mathbf{Y}_n^{(m)} - \mathbf{Y}_n^{(m-1)} \|_\infty \le \mathrm{TOL} = C\, h^p, \tag{4.1}$$

where C is a problem- and method-dependent parameter, p is order of the corrector. Notice that by this criterion the iteration error has the same order in h as the underlying corrector. Furthermore, in the tables of results, N_{seq} denotes the total number of sequential right hand side evaluations, N_{steps} denotes the total number of integration steps, k denotes number of processors needed for implementation. In the first integration step, we used the trivial predictor formula $\mathbf{Y}_1^{(0)} = y_n \mathbf{e}$.

4.1. Fehlberg problem

As a first numerical test, we integrate the often-used Fehlberg problem (cf. [3])

$$\begin{aligned} y_1'(t) &= 2\,t\,y_1(t)\,\log(\max\{y_2(t), 10^{-3}\}), \quad y_1(0) = 1, \\ y_2'(t) &= -2\,t\,y_2(t)\,\log(\max\{y_1(t), 10^{-3}\}), \quad y_2(0) = e, \end{aligned} \qquad 0 \le t \le 5, \tag{4.2}$$

with exact solution $y_1(t) = \exp(\sin(t^2))$, $y_2(t) = \exp(\cos(t^2))$. The results listed in Table 4.1 show that PISRK is always superior to PIRK. In the low accuracy range, the convergence of the PISRK methods in the integration process is slower than that of the BPIRK methods. This may be explained by the fact that the stability region of the PISRK methods is not sufficiently large for low m-values (see Table 3.2). However, for a given stepsize, the accuracy of the PISRK results turns out to be higher than the accuracy of the BPIRK method, so that the efficiency of PISRK is at least as high as that of BPIRK. Particularly, in the high accuracy range, the superiority of the PISRK methods over the BPIRK methods is evident.

Table 4.1. Values of NCD / N_{seq} for problem (4.2) obtained by various parallel PC methods

PC methods	k	p	N_{steps}=100	N_{steps}=200	N_{steps}=400	N_{steps}=800	N_{steps}=1600	C
BPIRK	8	4	3.2 / 200	4.3 / 406	5.4 / 844	6.5 / 1758	7.7 / 3759	10^3
PISRK	3	4	4.3 / 256	5.2 / 483	6.2 / 930	7.4 / 1820	8.7 / 3661	10^3
PIRK	2	4	2.7 / 392	4.0 / 842	5.2 / 1756	6.5 / 3650	7.7 / 7409	10^3
BPIRK	18	6	5.4 / 250	7.1 / 533	8.9 / 1150	10.7 / 2505	12.5 / 5317	10^3
PISRK	5	6	5.9 / 348	8.6 / 637	10.2 / 1194	12.2 / 2272	14.0 / 4398	10^3
PIRK	3	6	5.2 / 601	7.0 / 1245	8.9 / 2542	10.7 / 5199	12.5 / 10488	10^3
BPIRK	32	8	8.2 / 293	10.3 / 662	12.7 / 1432	15.0 / 2985	17.5 / 6233	10^3
PISRK	7	8	8.7 / 439	11.9 / 780	14.6 / 1439	17.3 / 2706	19.6 / 5116	10^3
PIRK	4	8	7.8 / 774	10.2 / 1603	12.6 / 3297	15.1 / 6674	17.5 / 13468	10^3
BPIRK	50	10	9.9 / 357	12.9 / 787	15.9 / 1710	18.9 / 3658	22.0 / 7540	10^3
PISRK	9	10	12.2 / 513	13.1 / 913	18.8 / 1654	21.7 / 3086	23.1 / 5919	10^3
PIRK	5	10	9.9 / 942	12.9 / 1947	15.9 / 3973	18.9 / 8134	22.0 / 16407	10^3

4.2. Orbit equation

Our second example is a well-known test problem in the RK-literature, viz. the orbit equation (cf. [7])

$$
\begin{aligned}
&y'_1(t) = y_3(t), && y_1(0) = 1 - \varepsilon,\\
&y'_2(t) = y_4(t), && y_2(0) = 0,\\
&y'_3(t) = \frac{-y_1(t)}{(y_1^2(t) + y_2^2(t))^{3/2}}, && y_3(0) = 0, && 0 \le t \le 20,\\
&y'_4(t) = \frac{-y_2(t)}{(y_1^2(t) + y_2^2(t))^{3/2}}, && y_4(0) = \sqrt{\frac{1+\varepsilon}{1-\varepsilon}}, && \varepsilon = \frac{3}{10}.
\end{aligned}
\tag{4.3}
$$

The performance of the methods is shown by the results given in Table 4.2. Again PISRK is superior to PIRK, but now, the BPIRK methods are slightly more efficient than PISRK in the low accuracy range. However, in the range of high accuracy, the PISRK methods are again superior to the BPIRK methods.

Table 4.2. Values of NCD / N_{seq} for problem (4.2) obtained by various parallel PC methods

PC methods	k	p	N_{steps}=100	N_{steps}=200	N_{steps}=400	N_{steps}=800	N_{steps}=1600	C
BPIRK	8	4	3.0 / 203	4.6 / 404	5.0 / 880	6.1 / 1861	7.3 / 3924	10^0
PISRK	3	4	2.7 / 270	5.0 / 499	5.8 / 958	7.7 / 1880	8.9 / 3739	10^0
PIRK	2	4	3.1 / 441	3.7 / 905	4.9 / 1947	6.1 / 4000	7.3 / 8000	10^0
BPIRK	18	6	4.8 / 237	6.8 / 511	8.7 / 1106	10.4 / 2516	12.2 / 5185	10^{-1}
PISRK	5	6	5.3 / 373	7.9 / 659	10.0 / 1172	12.6 / 2221	14.0 / 4363	10^{-1}
PIRK	3	6	5.0 / 643	7.2 / 1302	8.9 / 2637	10.5 / 5499	12.3 / 11200	10^{-1}
BPIRK	32	8	7.2 / 276	9.7 / 632	12.2 / 1382	14.7 / 2956	17.2 / 6277	10^{-2}
PISRK	7	8	7.9 / 458	10.9 / 808	14.0 / 1436	16.6 / 2695	19.0 / 5063	10^{-2}
PIRK	4	8	7.6 / 837	10.4 / 1686	12.8 / 3397	15.0 / 6845	17.3 / 13827	10^{-2}
BPIRK	50	10	9.5 / 265	12.8 / 637	16.0 / 1469	19.0 / 3187	22.1 / 6957	10^{-2}
PISRK	9	10	9.8 / 538	14.1 / 930	17.0 / 1651	19.6 / 2990	23.9 / 5625	10^{-2}
PIRK	5	10	9.3 / 926	12.8 / 1926	16.3 / 3927	19.2 / 8226	22.2 / 16532	10^{-2}

Concluding remarks

This paper shows the performance of a special class of symmetric Runge-Kutta methods when they are used as corrector methods for generating parallel PC methods for nonstiff problems. By two examples, we have shown that for a given order p the resulting PISRK method is by far superior to the PIRK method (about a factor from 2 to 5). However, the number of necessary processors is a factor 2 - 2/p larger. This modest increase of processors seems to be a low price for the substantially increased efficiency. The PISRK method is roughly competitive with BPIRK in the low accuracy range, but clearly more efficient in the high accuracy range. But here, it is the PISRK method that needs less processors. In fact, the number of processors needed by BPIRK is a factor $p^2 / (2p - 2)$ larger.

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 note.

References

[1] **Butcher, J. C. (1987):** *The Numerical Analysis of Ordinary Differential Equations, Runge-Kutta and General Linear Methods,* John Wiley & Sons, Chichester - New York - Brisbane - Toronto - Singapore.

[2] **Dekker, K. and Verwer, J.G. (1984):** *Stability of Runge-Kutta Methods for Stiff Nonlinear Dfferential Equations*, Noth-Holland, Amsterdam - New York - Oxford.

[3] **Fehlberg, E. (1969):** *Classical fith-, sixth- and eighth-order Runge-Kutta formulas with stepsize control*, NASA Technical Report 287, 1968; extract published in Computing 4, 93-106.

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

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

[6] **Houwen, P.J. van der & Sommeijer, B.P. (1990):** *Parallel iteration of high-order Runge-Kutta methods with stepsize control*, J. Comp. Appl. Math. 29, 111-127.

[7] **Hull, T.E., Enright, W.H., Fellen, B.M. & Sedgwick, A.E. (1972):** *Comparing numerical methods for ordinary differential equations*, SIAM J. Numer. Anal. 9, 603-637.

[8] **Lie, I. (1987):** *Some aspects of parallel Runge-Kutta methods,* Report No. 3/87, Division Numerical Mathematics, University of Trondheim, Norway.

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

[10] **Nørsett, S.P. & Simonsen, H.H. (1989):** *Aspects of parallel Runge-Kutta methods,* in: A. Bellen, C.W. Gear and E. Russo (Eds.): Numerical Methods for Ordinary Differential Equations, Proceedings L'Aquila 1987, LNM 1386, Springer-Verlag, Berlin.

Appendix

Here we give the parameters (**c**, A, **b**) in 24 decimals of the A-stable SRK methods of fourth-order, sixth-order, eighth-order and tenth-order based on optimal collocation vectors defined in Subsection 3.2. These parameters were computed on a 28-digits arithmetic computer.

Table A.1. Parameters (**c**, A, **b**) of the tenth-order A-stable SRK method

c(1)= 1.34880000000000000000000E-02
c(2)= 7.06712200000000000000000E-02
c(3)= 1.71897130000000000000000E-01
c(4)= 3.14968350000000000000000E-01
c(5)= 5.00000000000000000000000E-01
c(6)= 6.85031650000000000000000E-01
c(7)= 8.28102870000000000000000E-01
c(8)= 9.29328780000000000000000E-01
c(9)= 9.86512000000000000000000E-01

a(1,1)= 1.69233903381832193949744 1E-02
a(2,1)= 3.79881735624174811893251 2E-02
a(3,1)= 3.23412397350463850652009 6E-02
a(4,1)= 3.66499819112960377682619 1E-02
a(5,1)= 3.22656714193668143050637 4E-02
a(6,1)= 3.52357588948243440155151 3E-02
a(7,1)= 3.39281486578234588485651 7E-02
a(8,1)= 3.45868121118853271632823 0E-02
a(9,1)= 3.42304165397453566641532 2E-02
a(1,2)=-5.37287718694861097920438 3E-03
a(2,2)= 3.79304223993395262120739 5E-02
a(3,2)= 8.91406818843296070527915 4E-02
a(4,2)= 7.21482352260364382766842 6E-02
a(5,2)= 8.71581315601807490454548 7E-02
a(6,2)= 7.75388800207156408742966 8E-02
a(7,2)= 8.16699898534076045044148 5E-02
a(8,2)= 7.96139608959237267850334 6E-02
a(9,2)= 8.07197578727809467564191 5E-02
a(1,3)= 3.02006707596068797449605 1E-03
a(2,3)=-7.60579141639546401900070 7E-03
a(3,3)= 5.68786748505530513314948 1E-02
a(4,3)= 1.39469271102987048523714 9E-01
a(5,3)= 1.08503850256140007553873 7E-01
a(6,3)= 1.25515829632406780671115 6E-01
a(7,3)= 1.18626330051489620962407 8E-01
a(8,3)= 1.21965087300750751744052 1E-01
a(9,3)= 1.20192550676223397486085 6E-01
a(1,4)=-1.66814782151494577241470 3E-03
a(2,4)= 3.53155835665941763458872 7E-03
a(3,4)=-9.00759402959698800074277 4E-03
a(4,4)= 7.43111978414914803758747 3E-02
a(5,4)= 1.87354870875739420758512 2E-01
a(6,4)= 1.60617704717900651820617 5E-01
a(7,4)= 1.69974866015408730480806 9E-01
a(8,4)= 1.65689601477059923654339 3E-01
a(9,4)= 1.67906575431915862781961 5E-01
a(1,5)= 9.93220963186253929536972 6E-04
a(2,5)=-1.95932826820019081151092 1E-03
a(3,5)= 4.10181498599840020621922 1E-03
a(4,5)=-1.13438099677625467042218 9E-02
a(5,5)= 9.73583528482010408587690 3E-02
a(6,5)= 2.06060515664164628421759 9E-01
a(7,5)= 1.90614890710403681511318 8E-01
a(8,5)= 1.96676033964602272529048 9E-01
a(9,5)= 1.93723484733215827788001 2E-01
a(1,6)=-7.61388490430961616139117 0E-04
a(2,6)= 1.45558546442497751148324 1E-03
a(3,6)=-2.82967907392382931498433 6E-03
a(4,6)= 6.52748222358424934520498 7E-03
a(5,6)=-2.02096839342545195926897 0E-02
a(6,6)= 9.28339890999934207899477 7E-02
a(7,6)= 1.76152780971081889166565 3E-01
a(8,6)= 1.63613628584825483531233 7E-01
a(9,6)= 1.68813334762999846938237 2E-01
a(1,7)= 6.14536440237535007470217 7E-04
a(2,7)=-1.15800018428981925049661 6E-03
a(3,7)= 2.18075706497131153114770 9E-03
a(4,7)=-4.70874251594584817756000 6E-03
a(5,7)= 1.23032368603209249396818 6E-02
a(6,7)=-1.86621839865261160301593 3E-02
a(7,7)= 6.39284122659078811620607 1E-02
a(8,7)= 1.28412878532856396512556 3E-01
a(9,7)= 1.17787020040500244519059 6E-01
a(1,8)=-3.85253678760406324343203 8E-04
a(2,8)= 7.20543298096813647042438 8E-04
a(3,8)=-1.33548565938706407233894 8E-03
a(4,8)= 2.79562417330489955777917 9E-03
a(5,8)=-6.82362736616020861337903 3E-03
a(6,8)= 8.18626896798410215539158 1E-03
a(7,8)=-8.80617769030906662071568 6E-03
a(8,8)= 4.24040817946810142200019 4E-02
a(9,8)= 8.57073813809691514112802 7E-02
a(1,9)= 1.24452360087228385623751 6E-04
a(2,9)=-2.31943212052742113505250 8E-04
a(3,9)= 4.26720242009126201211797 4E-04
a(4,9)=-8.80889994991758965738152 1E-04
a(5,9)= 2.08919748046577074471324 4E-03
a(6,9)=-2.29511301146345271848493 0E-03
a(7,9)= 2.01362916478619998457601 2E-03
a(8,9)=-3.63330466258489613954815 1E-03
a(9,9)= 1.74314785616493656548025 6E-02

b(1)= 3.43548688998325850497769 6E-02
b(2)= 8.03345041940205404320760 8E-02
b(3)= 1.20807087116460932493555 4E-01
b(4)= 1 67145186941484901165822 6E-01
b(5)= 1.94716705696402081717538 0E-01
b(6)= 1.67145186941484901165822 6E-01
b(7)= 1.20807087116460932493555 4E-01
b(8)= 8.03345041940205404320760 8E-02
b(9)= 3.43548688998325850497769 6E-02

Table A.2. Parameters (**c**, A, **b**) of the eighth-order A-stable SRK method

c(1)= 2.18070700000000000000000E-02
c(2)= 1.13835970000000000000000E-01
c(3)= 2.75443500000000000000000E-01
c(4)= 5.00000000000000000000000E-01
c(5)= 7.24556500000000000000000E-01
c(6)= 8.86164030000000000000000E-01
c(7)= 9.78192930000000000000000E-01

a(1,1)= 2.71945852846834899969067 6E-02
a(2,1)= 6.16154356162785047384627 3E-02
a(3,1)= 5.18289859837328364425865 1E-02
a(4,1)= 5.95288624374945646617697 9E-02
a(5,1)= 5.48231876560905206652826 0E-02
a(6,1)= 5.68534987340026463796564 6E-02
a(7,1)= 5.58740746454937742028757 5E-02
a(1,2)=-8.22016779407128210263627 4E-03
a(2,2)= 6.01282061726677867537558 0E-02
a(3,2)= 1.44078529776929940052024 4E-01
a(4,2)= 1.15135242485371028401401 9E-01
a(5,2)= 1.30613952039300862693712 1E-01
a(6,2)= 1.24242089731935085319023 3E-01
a(7,2)= 1.27258085130406985890034 2E-01
a(1,3)= 4.31463025040686707235818 8E-03
a(2,3)=-1.12578016857505378461772 7E-02
a(3,3)= 8.93236400156221635575803 4E-02
a(4,3)= 2.20995978942976547717271 9E-01
a(5,3)= 1.90416317704594550158131 2E-01
a(6,3)= 2.01398792981060029593840 2E-01
a(7,3)= 1.96444024871552970631173 7E-01
a(1,4)=-2.42436113105964997188021 8E-03
a(2,4)= 5.32594330630770191714355 6E-03
a(3,4)=-1.44527664137933122644007 0E-02
a(4,4)= 1.19481431962505694270664 7E-01
a(5,4)= 2.53415630338804700805730 1E-01
a(6,4)= 2.33636920618703686624185 8E-01
a(7,4)= 2.41387225056071038513209 6E-01
a(1,5)= 1.61446053267280038826333 1E-03
a(2,5)=-3.34030757683425857440319 8E-03
a(3,5)= 7.64216769963122086130589 1E-03
a(4,5)=-2.29374935387507766978348 6E-02
a(5,5)= 1.08734845388603607461856 7E-01
a(6,5)= 2.09316287089976308865614 3E-01
a(7,5)= 1.93743855153818903947078 9E-01
a(1,6)=-9.98465101908612554266872 8E-04
a(2,6)= 2.01753029656328801674400 8E-03
a(3,6)=-4.35433201080248935794474 0E-03
a(4,6)= 1.11243775431273449343654 2E-02
a(5,6)=-1.78189097484315667162570 9E-02
a(6,6)= 6.61314138558305865820115 7E-02
a(7,6)= 1.34479787822569655438403 6E-01
a(1,7)= 3.26387959276387171255077 2E-04
a(2,7)=-6.53036129232485005525634 1E-04
a(3,7)= 1.37727494867964070884822 6E-03
a(4,7)=-3.32839983272440328763896 2E-03
a(5,7)= 4.37147662103732493154431 0E-03
a(6,7)=-5.41497301150834336433190 9E-03
a(7,7)= 2.90058773200866713772240 6E-02

b(1)= 5.62004626047701613741308 3E-02
b(2)= 1.26259620028498373335767 3E-01
b(3)= 1.98058485404225771019437 0E-01
b(4)= 2.38962863925011388541329 4E-01
b(5)= 1.98058485404225771019437 0E-01
b(6)= 1.26259620028498373335767 3E-01
b(7)= 5.62004626047701613741308 3E-02

Table A.3. Parameters (**c**, A, **b**) of the sixth-order A-stable SRK method

c(1)= 4.10117300000000000000000E-02
c(2)= 2.12357140000000000000000E-01
c(3)= 5.00000000000000000000000E-01
c(4)= 7.87642860000000000000000E-01
c(5)= 9.58988270000000000000000E-01

a(1,1)= 5.05986154333046438435088 8E-02
a(2,1)= 1.16419204368063987288154 1E-01
a(3,1)= 9.69756220749949329417367 4E-02
a(4,1)= 1.06677723412092392524838 0E-01
a(5,1)= 1.02816892662306584085754 3E-01
a(1,2)=-1.41434388653394951820526 0E-02
a(2,2)= 1.09410266541652197292860 8E-01
a(3,2)= 2.64917145828455550697859 4E-01
a(4,2)= 2.30095176480298008357207 6E-01
a(5,2)= 2.42459074248214894664500 6E-01
a(1,3)= 7.02232747235048354535633 8E-03
a(2,3)=-1.95096238875316302424116 2E-02
a(3,3)= 1.57189807129794153456557 6E-01
a(4,3)= 3.33889238147119937155526 8E-01
a(5,3)= 3.07357286787237823367758 9E-01
a(1,4)=-3.63469973568725229428481 9E-03
a(2,4)= 8.72919803222963401300812 3E-03
a(3,4)=-2.60927713159279083276436 2E-02
a(4,4)= 1.29414107970875445077354 9E-01
a(5,4)= 2.52967813377867137552268 4E-01
a(1,5)= 1.16892569537162008747219 4E-03
a(2,5)=-2.69190505441418835161150 0E-03
a(3,5)= 7.01019628268327123148980 4E-03
a(4,5)=-1.24333860103857831149276 1E-02
a(5,5)= 5.33872029243735603297176 5E-02

b(1)= 1.03985818357678204173226 5E-01
b(2)= 2.38824374512527642370215 8E-01
b(3)= 3.14379614259588306913115 2E-01
b(4)= 2.38824374512527642370215 8E-01
b(5)= 1.03985818357678204173226 5E-01

Table A.4. Parameters (**c**, A, **b**) of the fourth-order A-stable SRK method

c(1)= 1.03006620000000000000000E-01
c(2)= 5.00000000000000000000000E-01
c(3)= 8.96993380000000000000000E-01

a(1,1)= 1.24207508602896590564271 5E-01
a(2,1)= 2.89621416936229923897604 8E-01
a(3,1)= 2.56538635269563257230938 1E-01
a(1,2)=-2.90383758263742094570254 9E-02
a(2,2)= 2.35623877506959123876307 8E-01
a(3,2)= 5.00286130840292457209641 1E-01
a(1,3)= 7.83748722347761889275397 3E-03
a(2,3)=-2.52452944431890477739126 9E-02
a(3,3)= 1.40168613890144285559420 6E-01

b(1)= 2.64376122493040876123692 1E-01
b(2)= 4.71247755013918247752615 6E-01
b(3)= 2.64376122493040876123692 1E-01