1992

P.J. van der Houwen, Nguyen huu Cong

Parallel block predictor-corrector methods of Runge-Kutta type

Department of Numerical Mathematics Report NM-R9220 October

CWI is the research institute of the Stichting Mathematisch Centrum, which was founded on February 11, 1946, as a non-profit institution aiming at the promotion of mathematics, computer science, and their applications. It is sponsored by the Dutch Government through the Netherlands organization for scientific research (NWO).

Parallel Block Predictor-Corrector Methods of Runge-Kutta Type

P.J. van der Houwen and Nguyen huu Cong

CWI
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

Abstract In this paper, we construct block predictor-corrector methods using Runge-Kutta correctors. Our approach consists of applying the predictor-corrector method not only at step points, but also at off-step points (block points), so that, in each step, a whole block of approximations to the exact solution is computed. In the next step, these approximations are used for obtaining a high-order predictor formula by Lagrange or Hermite interpolation. By choosing the abscissas of the off-step points narrowly spaced, a much more accurately predicted value is obtained than by predictor formulas based on preceding step point values. Since the approximations at the off-step points to be computed in each step can be obtained in parallel, the sequential costs of these block predictor-corrector methods are comparable with those of a conventional predictor-corrector method. Furthermore, by using Runge-Kutta correctors, the predictor-corrector iteration scheme itself is also highly parallel. Application of these block predictor-corrector methods based on Lagrange-Gauss pairs to a few widely-used test problems reveals that the sequential costs are reduced by a factor ranging from 2 until 11 when compared with the best sequential methods.

1991 Mathematics Subject Classification: 65M12, 65M20
1991 CR Categories: G.1.7
Key Words and Phrases: numerical analysis, stability, parallelism.

Note: These investigations were supported by the University of Amsterdam who provided the second author with a research grant for spending a total of two years in Amsterdam.

1. Introduction

We will investigate a particular class of (explicit) predictor-corrector (PC) methods for solving the initial-value problem (IVP) for nonstiff, first-order differential equations

$$(1.1) \qquad \frac{dy(t)}{dt} = f(y(t))$$

on parallel computers. It is our aim to improve the conventional PC methods by using parallel processors. At a first level, PC methods can be characterized by the values $\{p, k, \beta\}$, where p is the order of the method, k is the number of right hand side evaluations per step, and β characterizes the stability of the integration process, e.g., β may denote the real or imaginary stability boundaries β_{re} and β_{im} of the method. Evidently, we would like to have a PC method in which for given order p, the value of k is small and β is sufficiently large. The magnitude of β should take into account the costs per step, which leads us to the definition of the *effective* or *scaled* stability boundary β/k.

For *sequential* computers, the PC methods of Adams type belong to the most efficient nonstiff IVP solvers. The PECE mode of these methods are characterized by $\{p, 2, \beta\}$, where the effective stability boundaries $(\beta_{re}, \beta_{im})/2$ monotonically decrease from (1.20, 0.60) for p = 3 to (0.16, 0.09) for p = 10. Less popular are PC methods based on PC pairs consisting of 'last step value predictors' and Runge-Kutta (RK) correctors. In $P(EC)^{p-1}E$ mode, these RK-type PC methods are characterized by $\{p, s(p-1)+1, \beta\}$, where s is the number of stages of the generating corrector. The effective stability boundaries $(\beta_{re}, \beta_{im})/(s(p-1)+1)$ strongly depend on the particular corrector chosen, but are extremely small for the higher-order RK correctors. The advantage of the RK type PC methods is their one-step nature facilitating easy implementation and stepsize control. However, the relatively large number of right hand side evaluations per step make them unattractive from a computational point of view.

With the introduction of *parallel* computers, several authors have proposed *parallel* methods (mostly of PC type) and have tried to improve on the sequential PC methods. Parallel PC methods can again be characterized by $\{p, k, \beta\}$ if we define k as the *sequential* number of right hand side evaluations per step, that is, the wall-clock time per step corresponds to the time needed to evaluate k right hand side functions. With this meaning of k, the *effective* stability boundary on *parallel* computers can again be defined by β/k. Let us first consider the parallel implementation of the Adams PECE methods and RK-type PC methods in $P(EC)^{p-1}E$ mode. The Adams PECE methods are again characterized by $\{p, 2, \beta\}$ indicating that these methods do not have intrinsic parallelism. For future reference, the effective stability boundaries are listed in Table 1.1. If the RK-type PC methods are implemented on a parallel computer, then we can characterize them by $\{p, p, \beta\}$ which shows that the sequential costs are reduced by about a factor

s. PC methods of this type have been discussed in [14, 16, 10, 13]. An actual implementation, including a stepsize strategy, and a detailed performance analysis can be found in [10] where they were called *PIRK methods* (parallel iterated RK methods). The effective stability boundaries of PIRK methods using 'last step value' predictors are listed in Table 1.1 (these methods possess stability boundaries that do not depend on the particular corrector chosen).

Table 1.1. Effective stability boundaries $(\beta_{re}, \beta_{im})/k$ of PC methods

	p=3	p=4	p=5	p=6	p=7	p=8	p=9	p=10
Adams PECE	(1.20,0.60)	(0.96,0.58)	(0.70,0.48)	(0.52,0.35)	(0.39,0.26)	(0.29,0.18)	(0.22,0.13)	(0.16,0.09)
PIRK (k = p)	(0.84,0.57)	(0.69,0.70)	(0.63,0.00)	(0.59,0.00)	(0.56,0.25)	(0.54,0.42)	(0.52,0.00)	(0.50,0.00)

Table 1.2. Effective stability boundaries $(\beta_{re}, \beta_{im})/k$ of various parallel methods

Method	p	k	$\beta := (\beta_{re}, \beta_{im})$
Multiblock method [5, Method {(2.7),(2.9)}]	3	2	(2.49, -)
BRK method [13, Method (4.1)]	3	1	(0.64,0.65)
Miranker-Liniger method [19]	4	1	(0.50,0.04)
Shampine-Watts-Worland [21, 23]	4	2	(0.44,0.58)
Multiblock method [5, Method {(2.11),(2.13)}]	4	2	(1.67, -)
Hermite-Gauss method [18]	4	2	-
BRK method [13, Method (4.7)]	4	1	(0.53,0.05)
BRK method [13, Method {(4.3),(4.6)}]	4	2	(0.06,0.05)
Cyclic mutistep method [7, Table 2]	6	2	-
BRK method [13, Method {(4.12),(4.13)}]	6	2	(0.87,0.29)
Cyclic mutistep method [7, Table 2]	8	2	-
BRK method [13, Method {(4.14),(4.15)}]	8	2	(0.15,0.07)

There have been several attempts to construct parallel methods without starting from a conventional sequential method [15, 19, 5, 11]. For a number of these parallel methods, Table 1.2 lists the corresponding {p, k, β/k} values (as far as available). We remark that the cyclic multistep methods mentioned in this table refer to parallel modifications of the original methods of Donelson and Hansen [6].

A further increase of the amount of parallelism in step-by-step methods consists of computing parallel solution values not only at step points, but also at off-step points, so that, in each step, a whole block of approximations to the exact solution is computed. This approach was successfully used in [7] for obtaining reliable defect control in explicit RK methods. In this paper, we want to use this approach for constructing parallel PC methods where the value of k is substantially less than the order p and where, at the same time, the effective stability boundaries are acceptably large. In our case, the block of approximations is used for obtaining a high-order predictor formula in the next step by some interpolation formula, e.g., Lagrange or Hermite interpolation. By choosing the abscissas of the off-step points narrowly spaced, we achieve much more accurately predictor values than can be obtained by predictor formulas based on preceding step point values. Moreover, the precise location of the off-step points can be used for minimizing the interpolation errors or for maximizing stability boundaries. Since the approximations at the off-step points to be computed in each step can be obtained in parallel, the sequential costs of this block PC method are equal to those of conventional PC methods. Furthermore, by using RK correctors, the PC iteration scheme itself is also highly parallel (cf. [16, 9]). The RK-based block PC methods may be considered as block versions of the aforementioned PIRK methods, and will therefore be termed *block PIRK methods* (BPIRK methods).

We concentrated on BPIRK methods based on Lagrange predictors and Gauss correctors. The number of sequential function calls per step of Lagrange-Gauss BPIRK methods equals k = m+1, m denoting the number of iterations performed. Using p-point Lagrange interpolation predictors (i.e., the dimension of the block of approximations equals p resulting in predictor formulas of order p-1) and pth-order Gauss correctors, we obtain a p-dimensional BPIRK method whose order equals p for all m (even m = 0). The abscissas of the off-step points were used for minimizing the predictor errors (in some sense, see Section 2.3). For these BPIRK methods, we computed the effective stability boundaries. It turned out that for $k \le 4$, the scaled stability boundary β_{re}/k assumes values in the range [0.31, 0.44]. The values of β_{im}/k are less constant and are often quite small (see Table 2.2 in Section 2.4). Table 1.3 lists cases where k is minimal while both β_{re}/k and β_{im}/k are 'substantial'. These figures show that the requirement of 'substantial' scaled stability boundaries makes the fourth- and tenth-order BPIRK methods relatively expensive. However, our numerical experiments reveal that in actual applications, the BPIRK methods of order four and ten already perform efficiently for k = 1 or k = 2. Hence, we conclude that minimizing the interpolation error leads to sufficiently stable methods requiring only one or two sequential function calls per step.

Table 1.3. Effective stability boundaries $(\beta_{re}, \beta_{im})/k$ of BPIRK methods based on {Lagrange, Gauss} pairs

p=4, k=3	p=6, k=2	p=8, k=1	p=10, k=4
(0.42, 0.42)	(0.39, 0.15)	(0.39, 0.20)	(0.37, 0.36)

In Section 3, we present comparisons with sequential and parallel methods from the literature for two widely-used test examples from the literature, viz. FEHL: the Fehlberg problem (cf. [8, p. 174]) and JACB: the Jacobian elliptic functions problem (cf. [8, p. 236]). Let R be the factor by which the sequential costs (i.e., wall-clock time) are reduced by applying the BPIRK methods to obtain the same accuracy. Then, from a comparison with sequential methods, we find the following reduction factors:

Problem	Method from the literature	BPIRK	R
FEHL	Dormand-Prince method of order 5	order 4	2
	Dormand-Prince method of order 8	order 8	5
JACB	Runge-Kutta-Hairer method of order 10	order 10	11

These conclusions encourage us to pursue the analysis of BPIRK methods. In particular, we will concentrate on a performance analysis of other predictors and on stepsize strategies that exploit the special structure of BPIRK methods.

2. Block PIRK Methods

For simplicity of notation, let the IVP be a scalar problem and let us consider the s-stage implicit RK method

$$\text{(2.1)} \qquad \mathbf{U} = y_n\mathbf{e} + hA\mathbf{f}(\mathbf{U}),\ y_{n+1} = y_n + h\mathbf{b}^T\mathbf{f}(\mathbf{U}),$$

where A is an s-by-s matrix, **b** is an s-dimensional vector, **e** is the unit vector, **U** is the stage vector with components U_j, and where **f**(**U**) denotes the vector with components $f(U_j)$. Suppose that we apply (2.1) at t_n with distinct stepsizes a_ih, where $i = 1, \ldots, r$ and $a_1 = 1$. Then we obtain a block of r numerical approximations $y_{n+1,i}$ to the exact solution values $y(t_n + a_ih)$ defined by

$$\text{(2.2)} \qquad \mathbf{U}_i = y_n\mathbf{e} + a_ihA\mathbf{f}(\mathbf{U}_i),\ y_{n+1,i} = y_n + a_ih\mathbf{b}^T\mathbf{f}(\mathbf{U}_i),\ i = 1, \ldots, r.$$

Let

$$\text{(2.3)} \qquad \mathbf{Y}_n := (y_{n,1}, \ldots, y_{n,r})^T,\ y_{n,1} := y_n,$$

and let us approximate the stage vectors $\mathbf{U}_i$ by

$$\text{(2.4)} \qquad \mathbf{U}_i^{(0)} = V_i\mathbf{Y}_n + hW_i\mathbf{f}(\mathbf{Y}_n),\ i = 1, \ldots, r,$$

where V_i and W_i are s-by-r matrices determined by order conditions (see Section 2.1). Regarding (2.2) as correctors and (2.4) as predictors for the stage vectors, we arrive at the PC method (in PE(CE)^{m}E mode)

$$\text{(2.5)} \qquad \begin{aligned} &\mathbf{U}_i^{(0)} = V_i\mathbf{Y}_n + hW_i\mathbf{f}(\mathbf{Y}_n),\ \mathbf{U}_i^{(j)} = \mathbf{e}_1^T\mathbf{Y}_n + a_ihA\mathbf{f}(\mathbf{U}_i^{(j-1)}),\ j = 1, \ldots, m, \\ &y_{n+1,i} = \mathbf{e}_1^T\mathbf{Y}_n + a_ih\mathbf{b}^T\mathbf{f}(\mathbf{U}_i^{(m)}),\ \mathbf{Y}_{n+1} := (y_{n+1,1}, \ldots, y_{n+1,r})^T, \end{aligned}$$

where $i = 1, \ldots, r$ and where $\mathbf{e}_1$ denotes the first unit vector. We may distinguish the following types of predictors:

Hermite:	$\mathbf{U}_i^{(0)} = V_i\mathbf{Y}_n + hW_i\mathbf{f}(\mathbf{Y}_n)$,	Lagrange:	$\mathbf{U}_i^{(0)} = V_i\mathbf{Y}_n$,
Adams:	$\mathbf{U}_i^{(0)} = y_{n,1}\mathbf{e} + hW_i\mathbf{f}(\mathbf{Y}_n)$,	Explicit BDF:	$\mathbf{U}_i^{(0)} = V_i\mathbf{Y}_n + hW_i\mathbf{f}(y_{n,r}\mathbf{e})$.

In the case of a Lagrange predictor, the PE(CE)^{m}E mode reduces to P(CE)^{m}E mode. If r = 1, then (2.5) reduces to the PIRK method studied in [10]. We shall call (2.5) an *r-dimensional BPIRK method.*

Given the vector $\mathbf{Y}_n$, the r values $y_{n+1,i}$ can be computed in parallel, and on a second level, the components of the ith stage vector iterate $\mathbf{U}_i^{(j)}$ can also be evaluated in parallel. Hence, r-dimensional BPIRK methods based on s-stage RK correctors can be implemented on a computer possessing r parallel processors each of which is itself a parallel system with s parallel processors. The number of sequential evaluations of f per step of length h equals k = m+2. If the matrices W_i vanish, then k = m+1.

2.1. Order conditons for the predictor

The order conditions for the predictor to be of order q are derived by replacing both Y_n and $U_i^{(0)}$ by exact solution values. On substitution of $y(t_{n-1}e + ha)$ and $y(t_ne + a_ihc)$, respectively, setting $c := Ae$, and by requiring that the residue is of order q+1 in h, we are led to the conditions

(2.6) $\quad y(t_ne + a_ihc) - V_iy(t_ne + h(a-e)) - hW_iy'(t_ne + h(a-e)) = O(h^{q+1}), \quad i = 1, \dots, r.$

Using the relation $y(te + hx) = \exp\left(hx\frac{d}{dt}\right) y(t)$, we can expand the lefthand side of (2.6) in powers of h:

(2.6') $$\left[\exp\left(h(a_ic+e)\frac{d}{dt}\right) - (V_i + W_i\, h\frac{d}{dt})\exp\left(ha\frac{d}{dt}\right)\right]y(t_{n-1})$$
$$= \sum_{j=0}^{q} C_i^{(j)}\left(h\frac{d}{dt}\right)^j y(t_{n-1}) + C_i^{(q+1)}\left(h\frac{d}{dt}\right)^{q+1} y(t^*) = O(h^{q+1}),$$

where t^* is a suitable chosen point in the interval containing the values $t_{n-1}+a_ih$, i = 1, ... , r, and where

(2.7a) $\quad C_i^{(j)} := \frac{1}{j!}[(a_ic+e)^j - V_ia^j - jW_ia^{j-1}] = 0, \quad j = 0, 1, \dots, q; \quad i = 1, \dots, r.$

The $C_i^{(j)}$, i = 1, ... , r, represent the error vectors of the predictor formula. From (2.6') we obtain the order conditions

(2.7b) $\quad C_i^{(j)} = 0, \quad j = 0, 1, \dots, q; \quad i = 1, \dots, r.$

The error vectors $C_i^{(q+1)}$ are the *principal* error vectors of the predictor (it is assumed that $C_i^{(q+1)}$ does not vanish).

If the conditions (2.7) are satisfied, then the iteration error associated with the stage vector and the step point value satisfy the order relations

$$U_i - U_i^{(m)} = O(h^{q+m+1}), \quad v_{n+1,i} - y_{n+1,i} = a_ihb^T[f(U_i) - f(U_i^{(m)})] = O(h^{q+m+2}),$$

where $v_{n+1,i}$ denote the exact corrector solutions. Thus, we have

Theorem 2.1. If the conditions (2.7) are satisfied and if the generating corrector (2.1) is of order p, then the orders of the iteration error and the BPIRK method (2.5) are $p_{iter} = q+m+1$ and $p^* := \min\{p, p_{iter}\}$, respectively. []

Let $q \geq r-1$ and define the matrices

$$P_i := \left(e, a_ic+e, (a_ic+e)^2, \dots, (a_ic+e)^{r-1}\right), \quad Q := \left(e, a, a^2, \dots, a^{r-1}\right), \quad R := \left(0, e, 2a, 3a^2, \dots, (r-1)a^{r-2}\right),$$
$$P_i^* := \left((a_ic+e)^r, \dots, (a_ic+e)^q\right), \qquad Q^* := \left(a^r, \dots, a^q\right), \qquad R^* := \left(ra^{r-1}, \dots, qa^{q-1}\right),$$

where the matrices P_i^*, Q^* and R^* are assumed to be zero if q = r-1. Then the conditions (2.7) can be presented in the form

(2.7') $\quad P_i - V_iQ - W_iR = O, \quad P_i^* - V_iQ^* - W_iR^* = O, \quad i = 1, \dots, r.$

Since the abscissas a_j are assumed to be distinct, we may write

$$V_i = [P_i - W_iR]Q^{-1}, \quad P_i^* - [P_i - W_iR]Q^{-1}Q^* - W_iR^* = O, \quad i = 1, \dots, r.$$

Using Theorem 2.1, explicit expressions for the predictor matrices V_i and W_i can be derived. The following theorem presents these matrices for Lagrange predictors and Hermite predictors:

Theorem 2.2. Let $\theta = 1$ and $\theta = 2$ respectively indicate the Lagrange and Hermite predictors. If

$$q = \theta r - 1, \quad V_i = [P_i - (\theta - 1)W_i R]Q^{-1}, \quad W_i = (\theta - 1)\,[P_i Q^{-1}Q^* - P_i^*][RQ^{-1}Q^* - R^*]^{-1}, \quad i = 1, \ldots, r,$$

then $p_{iter} = \theta r + m$, $p^* = \min\{p, p_{iter}\}$, $k = m + \theta$, where $RQ^{-1}Q^* - R^*$ is assumed to be nonsingular. []

In the application of BPIRK methods, we have two natural PC pairs, viz. Lagrange-Gauss pairs and Hermite-Radau pairs. The Lagrange-Gauss pairs have the advantage of (i) a maximal corrector-order for a given number of stages, (ii) no additional evaluations of f in the predictor (since we are aiming at a small number of iterations, say one or two, one extra f-evaluation substantially increases the total effort per step), and (iii) less round-off if the abscissas a_i are narrowly spaced. The disadvantage of Gauss correctors of being only A-stable, is not relevant here, since BPIRK methods are designed for nonstiff problems, so that more stable correctors such as the L-stable Radau correctors, are not needed. In the case of Radau correctors where the last component of the stage vector is identical to the step point value $y_{n+1,i}$, Hermite predictors are more natural because the additional f-evaluation needed in Hermite interpolation formulas is already available. An important advantage of using Hermite interpolation is the reduction of the number of processors needed for the implementation of BPIRK methods.

In this paper, we confine our considerations to Lagrange predictors and Gauss correctors. In the near future, we intend to compare BPIRK methods employing Lagrange, Hermite, Adams and BDF predictors.

2.2. Region of convergence

In actual integration, the number of iterations m is determined by some iteration strategy, rather than by order considerations. Therefore, it is of interest to know how the integration step affects the rate of convergence. The stepsize should be such that a reasonable convergence speed is achieved.

We shall determine the convergence factor for the test equation $y' = \lambda y$, where λ runs through the eigenvalues of the Jacobian matrix $\partial f/\partial y$. For this equation, we obtain the iteration error equation

$$(2.8) \qquad \mathbf{U}_i^{(j)} - \mathbf{U}_i = a_i z A\,[\mathbf{U}_i^{(j-1)} - \mathbf{U}_i], \quad z := h\lambda, \quad j = 1, \ldots, m.$$

Hence, with respect to the test equation, the convergence factor is defined by the spectral radius $\rho(a_i zA)$ of the iteration matrix $a_i zA$, $i = 1, \ldots, r$. Requiring that $\rho(a_i zA)$ is less than a given number α, leads us to the convergence condition

$$(2.9) \qquad a_i h \le \frac{\gamma(\alpha)}{\rho(\partial f/\partial y)}, \quad \gamma(\alpha) := \frac{\alpha}{\rho(A)},$$

where $\gamma(\alpha)$ presents the convergence boundary of the method. In Table 2.1, the maximal convergence boundaries $\gamma(\alpha)$ are given for Gauss correctors of orders up to 10. In actual computation, the stepsize should of course be substantially smaller than allowed by $\gamma(1)$. Notice that for a given integration step h, the maximal damping factor is given by $\alpha = a_i h \rho(\partial f/\partial y)/\gamma(1)$, so that the higher-order correctors listed in Table 2.1 give rise to faster convergence.

Table 2.1. Convergence boundaries $\gamma(\alpha)$.

	p = 4	p = 6	p = 8	p = 10
Gauss-Legendre	3.46α	4.65α	6.06α	7.30α

2.3. On the choice of the abscissas a_i

The accuracy of Lagrange interpolation formulas improves if the abscissas of the interpolating values are narrowly spaced. However, this will increase the magnitude of the entries of the matrix V_i, causing serious round-off errors. There are several ways to reduce this round-off effect: (i) multi-precision arithmetic, (ii) direct computation of the extrapolated values, and (iii) limitation of the spacing of the abscissas. The use of multi-precision arithmetic is the most simple remedy, but not always available and usually rather costly. Direct interpolation of the values $y_{n,1}, \ldots, y_{n,r}$ requires in each step and for each component equation of the system of IVPs the solution of a linear system of dimension θr. Again, this option is rather costly. Probably, the most realistic option is a limitation on the minimal spacing of the abscissas a_i. In [7] where Hermite interpolation formulas were used for deriving reliable error estimates for defect control, it was found that on a Silicon Grafics Inc. Power Iris 4D/240S-64 machine with 15 digits precision, the abscissas should be separated by 0.2 in order to suppress rounding errors. For the more stable Lagrange interpolation formulas, we expect that slightly smaller spacings are still acceptable.

In order to derive further criteria for the choice of suitable values for the absissas a_i, we need insight into the propagation of a perturbation ε of the block vector $\mathbf{Y}_n$ within a single step. We shall study this for the test equation $y' = \lambda y$. First we express $y_{n+1,i}$ in terms of $\mathbf{Y}_n$. Applying (2.5) and (2.8), we obtain the recursions

$$U_i^{(0)} = [V_i + zW_i]\, Y_n,\quad U_i^{(j)} - U_i = a_i zA\,[U_i^{(j-1)} - U_i),\quad j = 1, \dots, m.$$

Hence,

$$\begin{aligned}(2.10)\qquad y_{n+1,i} &= e_1^T Y_n + a_i z b^T [U_i^{(m)} - U_i] + a_i z b^T U_i = e_1^T Y_n + a_i z b^T [a_i zA]^m [U_i^{(0)} - U_i] + a_i z b^T U_i \\ &= (e_1^T + a_i z b^T [I - a_i zA]^{-1} e e_1^T) Y_n + a_i z b^T [a_i zA]^m [V_i + zW_i - [I - a_i zA]^{-1} e e_1^T] Y_n \\ &= R(a_i z) e_1^T Y_n + a_i z b^T [a_i zA]^m [V_i + zW_i - [I - a_i zA]^{-1} e e_1^T] Y_n,\quad i = 1, \dots, r,\end{aligned}$$

where $R(z)$ is the stability function of the RK corrector. Let us now replace Y_n by $Y_n^* = Y_n + \varepsilon$. Then, the perturbed value of $y_{n+1,i}$ is given by

$$(2.10')\qquad y^*_{n+1,i} = y_{n+1,i} + R(a_i z) e_1^T \varepsilon + a_i z b^T [a_i zA]^m [V_i + zW_i - [I - a_i zA]^{-1} e e_1^T] \varepsilon.$$

This relation shows that the first component of the perturbation ε is amplified by a factor of $O(1)$, whereas all other components are amplified by a factor of $O(h^{m+1})$.

Let us now return to the choice of the absissas a_i. The values of the a_i influence the accuracy of the predicted stage values, and hence the accuracy of the block vectors Y_n. Let ε represent the effect on Y_n of using inaccurate interpolation formulas in the preceding steps. Then, from the preceding discussion, we may conclude that the first component of ε is not damped. Since the components of the block vectors Y_n are calculated independently from the predicted stage values, it is important that the interpolation error corresponding to the predicted stage values used for the first component of the block vector are small. Thus, we should try to minimize the magnitude of the principal error vector $C_1^{(q+1)}$.

In the case of Lagrange predictors where $q = r-1$, we have to minimize the magnitude of $C_1^{(r)}$. Although we may use (2.7a) for minimizing $C_1^{(r)}$, it is more convenient to start with the usual expression for the remainder term in Lagrange interpolation formulas. For sufficiently differentiable functions $y(t)$, the r-point Lagrange interpolation formula can be written in the form (see e.g. [1, formulas 25.2.1 - 25.2.3])

$$(2.11)\qquad y(t_n + \tau h) = \sum_{i=1}^{r} L_i(\tau)\, y(t_{n-1} + a_i h) + C^{(r)}(\tau) \left(h \frac{d}{dt}\right)^r y(t^*),\quad C^{(r)}(\tau) := \frac{1}{r!} \prod_{i=1}^{r} [\tau + 1 - a_i],$$

where $L_i(\tau)$ are the interpolation coefficients and t^* is a suitably chosen point in the interval containing the values $t_{n-1}+a_i h$, $i = 1, \dots, r$. The principal error vectors of the Lagrange predictor formulas defined by Theorem 2.2 are given by $C_i^{(r)} = C^{(r)}(ca_i)$, $i = 1, \dots, r$. Recalling that $a_1 = 1$, we are led to minimize the magnitude of the values

$$C^{(r)}(c_j) = \frac{1}{r!} \prod_{i=1}^{r} [c_j + 1 - a_i],\quad j = 1, \dots, s.$$

Confining our considerations to block dimensions $r \geq s+1$, we set

$$(2.12a)\qquad a_1 = 1,\quad a_i = 1 + c_{i-1},\quad i = 2, \dots, s+1.$$

By this choice, the principal error vector $C_1^{(r)}$ vanishes, so that now all inaccuracies introduced by the predictor formula are damped by a factor of $O(h^{m+1})$ (cf. (2.10')). If $r > s+1$, then we have additional abscissas for improving the predictor formula. It is tempting to use these additional abscissas for reducing the magnitude of the other error vectors. From (2.11) it follows that the largest error constant (corresponding to the largest values of a_i and c_j) can be minimized by choosing the remaining abscissas close to 1. However, as already observed, the minimal spacing of the abscissas should be sufficiently large to avoid round-off. From (2.12a) it follows that the *averaged* spacing of the abscissas $a_1, \dots, a_{s+1}$ is $1/(s+1)$ for correctors with $c_s \neq 1$ and $1/s$ otherwise, the *minimal* spacing being, in general, smaller. Therefore, it seems recommendable to choose the remaining abscissas outside the interval $[1, 1 + c_s]$. In our numerical experiments, we have chosen the remaining abscissas such that averaged spacing equals that of the abscissas $a_1, \dots, a_{s+1}$. This leads us to define the remaining abscissas according to

$$(2.12b)\qquad \text{if } c_s \neq 1, \text{ then } a_i = \frac{s+i}{s+1},\ i = s+2, \dots, r,\ \text{else } a_i = \frac{s+i-1}{s},\ i = s+2, \dots, r.$$

For Gauss correctors, the order $p = 2s$, resulting in an averaged spacing $2/(p+2)$. Recalling that the 15 digits experiments reported in [7] indicate that a minimal spacing of 0.2 is acceptable in the case of Hermite interpolation, we expect that on 15 digits computers and for orders up to $p = 10$, an averaged spacing of $2/(p+2)$ should be acceptable in the case of the more stable Lagrange interpolation formulas. We remark that the optimal location of the off-step points for defect control as derived in [7] is in the interval where the defect is to be computed, rather than advancing the current step point as in (2.12).

Finally, we remark that the abscissas defined by (2.12) enable us to develop various cheap strategies for stepsize control. For example, if $r \geq s+2$, then the difference $y_{n-1,s+2} - y_{n,1}$ can be used for obtaining an error estimate.

2.4. Stability

From (2.10) it follows that we may write

$$Y_{n+1} = M_{mr}(z)\, Y_n, \quad M_{mr}(z) := \begin{pmatrix} R(a_1z)e_1^T + a_1zb^T[a_1zA]^m\,[V_1 + zW_1 - [I - a_1zA]^{-1}ee_1^T] \\ \cdots \\ R(a_rz)e_1^T + a_rzb^T[a_rzA]^m\,[V_r + zW_r - [I - a_rzA]^{-1}ee_1^T] \end{pmatrix}.$$

Evidently, the *asymptotic* stability region for $m \to \infty$ is the intersection in the z-plane of the stability region S_{corr} of the generating corrector and the region of convergence defined by the points z where the eigenvalues of a_izA are within the unit disk. Hence, if the corrector is A-stable, then the asymptotic stability region in the left half plane is completely determined by the region of convergence (see Table 2.1 for convergence boundaries).

For *finite* m, the stability regions are given by

$$S_{stab}(m,r) := \{z: \rho(M_{mr}(z)) < 1\}.$$

The associated real and imaginary stability boundaries β_{re} and β_{im} can be defined in the usual way.

Let us consider methods where r = p and where the number of iterations is chosen dynamically by some iteration strategy. This type of methods use 'maximal' block dimension r (in the sense that the order of the predictor equals that of the corrector) and iterate until a stable result is obtained assuming that the process converges. Again restricting our considerations to Lagrange-Gauss pairs, we obtain the results listed in Table 2.2. Because the effective, *real* stability boundaries are almost constant for all k, we may use k = 1 if only the real stability boundary plays a role. The *imaginary* stability boundaries show a less regular behaviour. BPIRK methods with (r, p, k) = (4, 4, 3), (6, 6, 4), (8, 8, 2), (10, 10, 4) possess reasonably large effective real and imaginary stability boundaries (these cases are collected in Table 1.3). Notice that in all cases the convergence regions contains the real and imaginary stability intervals, so that the integrations step will not be limited by convergence conditions, but rather by accuracy or stability conditions.

Table 2.2. Effective stability boundaries $(\beta_{re}, \beta_{im})/k$ of BPIRK methods of order $p^* = p$ using Lagrange-Gauss pairs with r = p.

p	k = 1	k = 2	k = 3	k = 4
4	(0.44, 0.00)	(0.40, 0.00)	(0.42, 0.42)	(0.37, 0.37)
6	(0.40, 0.08)	(0.39, 0.15)	(0.39, 0.03)	(0.38, 0.39)
8	(0.39, 0.20)	(0.38, 0.28)	(0.38, 0.35)	(0.37, 0.05)
10	(0.31, 0.00)	(0.37, 0.00)	(0.36, 0.03)	(0.37, 0.36)

3. Numerical experiments

We tested accuracy and efficiency aspects of BPIRK methods based on Lagrange-Gauss pairs. All experiments are performed on a 28 digit computer, so that the effect of rounding errors is negligible. In Section 3.1, we will concentrate on the accuracy of the methods. In particular, the effective order and the influence of the number of iterations on the efficiency will be tested. In Section 3.2, we compare the BPIRK methods with block RK methods, and in Section 3.3 a number of tenth-order methods are compared. In all experiments, the abscissas a_i are defined according to (2.12).

The maximal absolute error obtained at t = T is presented in the form $10^{-\Delta}$ (Δ may be interpreted as the number of correct decimal digits). Negative values of Δ are indicated by *. If the order of accuracy shown in the experiments equals the theoretical order p^*, then, on halving the (fixed) stepsize, the number of correct decimal digits should increase by $0.3p^*$. Hence, the number of steps denoted by N_{steps} and Δ are related according to $N_{steps} = c\, 2^{\Delta/(0.3p^*)}$, where c is a constant depending on the problem. In order to verify this theoretical relation, we define the effective order

$$(3.1) \qquad p_{eff} := \frac{\Delta(h) - \Delta(2h)}{0.3}.$$

In the first step, we always set r = 1 and k = m+1 = p, k being the number of *sequential* function calls per step. For the subsequent steps, we used either r = 1 (PIRK methods) or r = p, while k is specified in the tables of results. These methods will be denoted by PIRK(p, k) and BPIRK(p, k). The stepsize is chosen such that the total number of sequential function calls (approximately) equals a prescribed number N_{seq}. Since $N_{seq} = p + k(N_{steps} - 1)$, we have

$$N_{steps} = 1 + \left[\frac{N_{seq} - p}{p - r + 1} + \frac{1}{2} \right], \quad h := \frac{T - t_0}{N_{steps}},$$

where [.] denotes the integer part function and T denotes the end point of the integration interval (the effect of the integer part operation causes that the actual number of sequential right hand sides may be slightly different from the prescribed number N_{seq}).

3.1. Accuracy tests

Consider the often-used test problem of Fehlberg (cf. [8, p. 174])

$$\begin{aligned} y_1' &= 2\,t\,y_1 \log(\max\{y_2, 10^{-3}\}), \quad y_1(0) = 1, \\ y_2' &= -2\,t\,y_2 \log(\max\{y_1, 10^{-3}\}), \quad y_2(0) = e, \end{aligned} \qquad 0 \le t \le T, \tag{3.2}$$

with exact solution $y_1(t)=\exp(\sin(t^2))$, $y_2(t)=\exp(\cos(t^2))$. The Tables 3.1 and 3.2 present results for the fourth- and eighth-order Gauss correctors. We listed values of Δ for prescribed numbers N_{seq} of sequential function calls and the effective orders p_{eff} corresponding to the smallest stepsize h. In order to appreciate the accuracy of the BPIRK methods, we added the Δ-values produced by the PIRK methods and by the 'best' sequential methods currently available. In Table 3.1 we included results obtained by the 5(4) Dormand-Prince RK pair (DOPRI5) taken from [8, Figure 4.3], and in Table 3.2 we included results obtained by the 8(7) Dormand-Prince RK pair (DOPRI8) (see [9, Table 5]). Unlike the BPIRK results, the DOPRI results are obtained using a stepsize strategy, so that at first sight, a comparison may not be fair. However, the BPIRK methods can be provided with a stepsize strategy without additional costs per step (see [9]) and, for problem (3.1), stepsize strategies do not change the (N_{seq}, Δ) results very much. This may be concluded from a comparison of the PIRK(8, 8) results of Table 3.2 with the results reported in [9, Table 5] for the stepsize control version of PIRK(8, 8), i.e. the code PIRK8. Therefore, it seems fair to conclude that for the Fehlberg problem (3.1) the BPIRK(4, 1) method is at least a factor two faster than DOPRI5, and BPIRK(8, 2) beats DOPRI8 by at least a factor five.

Table 3.1. Correct decimal digits at t = T = 5 for problem (3.2)

N_{seq}	DOPRI5	PIRK(4, k)	BPIRK(4, k)		
		k = 4	k = 1	k = 2	k = 3
240		1.2	3.5	3.5	2.4
480	2.9	2.7	5.1	4.8	3.7
960	4.6	3.9	6.7	6.0	4.9
1920	6.0	5.1	8.2	7.2	6.1
p_{eff}		4.0	5.0	4.0	4.0

Table 3.2. Correct decimal digits at t = T = 5 for problem (3.2)

N_{seq}	DOPRI8	PIRK(8, k)	BPIRK(8, k)		
		k = 8	k = 1	k = 2	k = 3
240		1.5	6.8	8.1	7.4
480		6.0	10.8	11.7	9.7
960	7.0	8.3	13.8	14.2	12.1
1920	9.9	10.3	16.9	16.7	14.5
p_{eff}		6.7	10.3	8.3	8.0

3.2. Comparison with other parallel methods

In [11] parallel block Runge-Kutta methods (BRK methods) of orders up to 8 for nonstiff problems have been constructed and were shown to be highly efficient when compared with sequential methods. One of the test examples in [11] is the equation of motion of a rigid body without external forces (problem JACB in [8, p. 236]):

$$\begin{aligned} y_1' &= y_2 y_3, & y_1(0) &= 0, \\ y_2' &= -y_1 y_3, & y_2(0) &= 1, \quad 0 \le t \le T. \\ y_3' &= -.51 y_1 y_2, & y_3(0) &= 1, \end{aligned} \tag{3.3}$$

Table 3.3 presents a comparison of the most efficient BRK methods with BPIRK methods of the same order. These (fixed-stepsize) results show that the BPIRK methods are about four times as efficient as the BRK methods. However, the BRK methods are all two-processor methods, whereas the BPIRK methods require $p^2/2$ processors.

Table 3.3. Comparison with methods from the literature for problem (3.3) with T = 20.

Sequential righthand sides N_{seq}	120	240	480	960	p_{eff}	$p^*=p$
BRK [10, PC pair (4.3)-(4.6) of Table 5.4]	*	3.3	4.7	6.0	4.3	4
BPIRK(4, 1)	4.3	5.8	7.2	8.7	5.0	4
BRK [10, PC pair (4.12)-(4.13) of Table 5.4]	3.2	5.1	6.9	8.7	6.0	6
BPIRK (6, 1)	6.8	9.3	11.3	13.4	7.0	6
BRK [10, PC pair (4.14)-(4.15)]	2.9	7.4	9.8	12.2	8.0	8
BPIRK (8, 2)	8.7	11.4	13.8	16.2	8.0	8

3.3. Comparison of tenth-order methods

We repeat the (fixed-stepsize) experiment performed in [8], where a number of methods were compared by applying them to problem (3.3) with T = 60 and by counting the number of (sequential) function calls needed to obtain 10 digits accuracy. In Table 3.4, we reproduce the values given in [8], and [10] for a few tenth-order methods, and we added the results obtained by our tenth-order BPIRK method. From these results we conclude that the BPIRK(10, 3) method is about 11 times cheaper than the sequential Runge-Kutta-Hairer method and about 4 times cheaper than the PIRK(10,10) method.

Table 3.4. Comparison with tenth-order methods from the literature for problem (3.3) at T = 60.

Method	k	p	N_{steps}	Δ	N_{seq}
Runge-Kutta-Curtis (cf. [8])	18	10	240	9.9	4320
Runge-Kutta-Hairer [8]	17	10	240	10.1	4080
PIRK(10,k) method [10, Table 4]	10	10	150	10.0	1560
BPIRK(10, k)	1	10	410	10.1	419
	2	10	190	10.1	389
	3	10	120	10.0	369

Acknowlegement. The authors are grateful to Dr. B.P. Sommeijer for his interest in our investigations and for his careful reading of the paper.

References

[1] Abramowitz, M.& Stegun, I.A. (1970): Handbook of mathematical functions, Nat. Bur. Stand., Appl. Math. Ser. 55, Dover, New York.

[2] Burrage, K. (1991): The error behaviour of a general class of predictor-corrector methods, Appl. Numer. Math., 201-216.

[3] Burrage, K. (1992): The search for the Holy Grail, or Predictor-Corrector methods for solving ODEIVPs, to appear in Appl. Numer. Math..

[4] Burrage, K. (1993): Efficient block predictor-corrector methods with a small number of iterations, to appear in J. Comp. Appl. Math.

[5] Chu, M.T. & Hamilton, H. (1987): Parallel solution of ODE's by multi-block methods, SIAM J. Sci. Stat. Comput. 8, 342-353.

[6] Donelson, J. & Hansen, E. (1971): Cyclic composite multistep predictor-corrector methods, SIAM Numer. Anal. 8, 137-157.

[7] Enright, W.H. & Highman, D.J. (1991): Parallel defect control, BIT 31, 647-663.

[8] Hairer, E. (1978): A Runge-Kutta method of order 10, J. Inst. Math. Applics. 21, 47-59.

[9] Hairer, E., Nørsett, S.P. & Wanner, G. (1987): Solving ordinary differential equations I, Nonstiff problems, Springer-Verlag, Berlin.

[10] 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.

[11] Houwen, P.J. van der & Sommeijer, B.P. (1992): Block Runge-Kutta methods on parallel computers, Z. angew. Math. Mech. 68, 3-10.

[12] Houwen, P.J. van der & Nguyen huu Cong (1992): Parallel block predictor-corrector methods of Runge-Kutta type, Report NM-R92xx, Centre for Mathematics and Computer Science, Amsterdam.

[13] Jackson, K.R. & Nørsett, S.P. (1990): The potential for parallelism in Runge-Kutta methods.
Part I: RK formulas in standard form, Technical Report No. 239/90, Department of Computer Science, University of Toronto.
Part II: RK predictor-corrector formulas, in preparation.

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

[15] Miranker, W.L. & Liniger, W. (1967): Parallel methods for the numerical integration of ordinary differential equations, Math. Comput. 21, 303-320.

[16] 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.

[17] Shampine, L.F. & Watts, H.A. (1969): Block implicit one-step methods, Math. Comput. 23, 731-740.

[18] Sommeijer, B.P.: Stability boundaries of block Runge-Kutta methods, in preparation.

[19] Worland, P.B. (1976): Parallel methods for the numerical solution of ordinary differential equations, Trans. Comput. C-25, 1045-1048.