Numerical Algorithms 8(1994)293–312

Parallelism across the steps in iterated Runge–Kutta methods for stiff initial value problems*

P.J. van der Houwen, B.P. Sommeijer and W.A. van der Veen
CWI, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands

Communicated by J.C. Butcher
Received 26 November 1993; revised 3 September 1994

For the parallel integration of stiff initial value problems (IVPs), three main approaches can be distinguished: approaches based on "parallelism across the problem", on "parallelism across the method" and on "parallelism across the steps". The first type of parallelism does not require special integration methods and can be exploited within any available IVP solver. The method-parallel approach received some attention in the case of Runge–Kutta based methods. For these methods, the required number of processors is roughly half the order of the generating Runge–Kutta method and the speed-up with respect to a good sequential IVP solver is about a factor 2. The third type of parallelism (step-parallelism) can be achieved in any IVP solver based on predictor–corrector iteration. Most step-parallel methods proposed so far employ a large number of processors, but lack the property of robustness, due to a poor convergence behaviour in the iteration process. Hence, the effective speed-up is rather poor. The step-parallel iteration process proposed in the present paper is less massively parallel, but turns out to be sufficiently robust to solve the four-stage Radau IIA corrector used in our experiments within a few effective iterations per step and to achieve speed-up factors up to 10 with respect to the best sequential codes.

Keywords: Numerical analysis, Runge–Kutta methods, parallelism.

Subject classification: G.1.7.

1. Introduction

Recently, various attempts have been made to solve stiff initial value problems (IVPs)

$$y'(t) = f(y(t)), \quad y(t_0) = y_0, \quad y, f \in \mathbb{R}^d, \tag{1.1}$$

on parallel computers. Using the familiar terminology of parallelism "across the problem", "across the steps" and "across the method", we mention the *problem-parallel* methods based on wave form relaxation (cf. the survey paper of Burrage

* The research reported in this paper was partly supported by the Technology Foundation (STW) in the Netherlands.

[3]), the *step-parallel* methods of Bellen and coworkers [1,2] and Chartier [6], and the *method-parallel* solvers proposed in [10] based on parallel iteration of Runge–Kutta (RK) methods. To some extent, these three types of parallelism are orthogonal in the sense that they can often be combined. In this paper, we shall be concerned with step-parallelism.

Our starting point is a stiff IVP method that is both highly accurate and highly stable. This method is used as a corrector that is solved to convergence using parallel iteration techniques. In the selection of a suitable corrector, we are automatically led to the classical implicit Runge–Kutta methods such as the Radau IIA methods. These methods fulfil the requirements of accuracy and stability and belong to the best correctors for stiff problems. For the iteration method we choose the PDIRK (Parallel Diagonally Implicit RK) approach developed in [10] that solves the RK corrector by diagonally implicit iteration using s processors, s being the number of stages of the corrector. In [10], we advocated alternative correctors (called Lagrange correctors) which possessed stage order $s+1$, whereas s-stage Radau methods have only stage order s. Since the stage order is important for the accuracy in many stiff problems and because the number of processors equals s, the Lagrange correctors may have advantages if the number of processors is small. However, recent developments indicate that the number of processors is no longer an important issue. Therefore, we adopt the Radau IIA methods as the correctors to be used in this paper. Using a predictor based on extrapolation of preceding stage values and the four-stage Radau IIA corrector, we obtained for the PDIRK approach a speed-up factor of about 2 with respect to the best sequential codes for stiff problems, viz. the variable order LSODE code and the fifth-order RADAU5 code [9]. An interesting feature of the PDIRK-based code (called PSODE in [16]) is the highly efficient performance of high-order correctors in the low accuracy range. Hence, assuming that sufficiently many processors are available, we may equally well use a Radau corrector with more than four stages without increasing the sequential costs, while the high order is effective both in the low and high accuracy range.

A drawback of the PDIRK methods is that the number of iterations needed to achieve corrector accuracy is still high (about the order of the corrector). To reduce the number of iterations, we introduced preconditioning into the PDIRK methods by which the number of iterations reduces substantially (cf. [11]). In this paper, we apply step-parallelism to the PDIRK methods. The analysis given here partly parallels the derivations in [13] for nonstiff problems.

2. Parallelism across the steps

Following [13], we write the RK method in the General Linear Method (GLM) form introduced by Butcher [4] (see also [5, p. 340]):

$$\boldsymbol{Y}_n = (E \otimes I_d)\,\boldsymbol{Y}_{n-1} + h_n(A \otimes I_d)\,\boldsymbol{F}(\boldsymbol{Y}_n), \quad n = 1, \ldots, N. \tag{2.1a}$$

Here, h_n denotes the stepsize $t_n - t_{n-1}$, the matrix A contains the RK parameters, and $\boldsymbol{F}(\boldsymbol{Y}_n)$ contains the derivative values $(\boldsymbol{f}(\boldsymbol{Y}_{n,i}))$, where $\boldsymbol{Y}_{n,i}$, $i = 1,2,\ldots,s$, denote the d-dimensional components of the stage vector $\boldsymbol{Y}_n$. In this paper we will assume that (2.1a) possesses s implicit stages and that the last stage corresponds to the step point t_n (e.g. Radau IIA type methods). The first $s-1$ stage vector components $\boldsymbol{Y}_{n,i}$ represent numerical approximations at the intermediate points $t_{n-1} + c_i h_n$, $i = 1,2,\ldots,s-1$, where $\boldsymbol{c} = (c_i) = A\boldsymbol{e}$, $\boldsymbol{e}$ being the vector with unit entries. In (2.1a), the matrix E is of the form

$$E := \begin{pmatrix} 0 & \cdots & 0 & 1 \\ \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdots & \cdot & \cdot \\ 0 & \cdots & 0 & 1 \end{pmatrix}, \tag{2.1b}$$

the matrix I_d is the d-by-d identity matrix, $\otimes$ denotes the Kronecker product, and we define $\boldsymbol{Y}_0 = \boldsymbol{e} \otimes \boldsymbol{y}_0$. In the following, the dimension of I and $\boldsymbol{e}$ may change, but will always be clear from the context.

We approximate the solution $\boldsymbol{Y}_n$ of (2.1) by successive iterates $\boldsymbol{Y}_n^{(j)}$ satisfying the iteration scheme

$\boldsymbol{Y}_n^{(1)}$ defined by a predictor formula,

$$\boldsymbol{Y}_n^{(j)} - h_n(D \otimes I_d)\boldsymbol{F}(\boldsymbol{Y}_n^{(j)}) = (E \otimes I_d)\,\boldsymbol{Y}_{n-1}^{(q(n-1,j))} + h_n((A-D) \otimes I_d)\boldsymbol{F}(\boldsymbol{Y}_n^{(j-1)}),$$
$$j = 2,\ldots,m(t_n), \tag{2.2}$$

where $n = 1,2,\ldots,N$ and $\boldsymbol{Y}_0^{(j)} = \boldsymbol{e} \otimes \boldsymbol{y}_0$ for all j. The predictor formula and the integer-valued function $q(n,j)$ will be discussed below. The number of iterations $m(t_n)$ performed at t_n is defined by the condition that for $j = m(t_n)$ the iterates $\boldsymbol{Y}_n^{(j)}$ *numerically* satisfy the corrector equation (2.1) (evidently, if the iterates $\boldsymbol{Y}_n^{(j)}$ satisfying (2.2) converge to fixed vectors $\boldsymbol{V}_n$ as $j \to \infty$, then $\boldsymbol{V}_n = \boldsymbol{Y}_n$). The matrix D will be assumed to be diagonal with s positive diagonal entries. In the case of the s-stage Radau IIA correctors ($s = 2,3,4$), suitable matrices $D = D_s$ have been derived in [10]. For future reference, these matrices are here reproduced:

$$D_2 = \frac{1}{30}\begin{pmatrix} 20 - 5\sqrt{6} & 0 \\ 0 & 12 + 3\sqrt{6} \end{pmatrix},$$

$$D_3 = \begin{pmatrix} \dfrac{4365}{13624} & 0 & 0 \\ 0 & \dfrac{1032}{7373} & 0 \\ 0 & 0 & \dfrac{1887}{5077} \end{pmatrix}, \tag{2.3}$$

$$D_4 = \begin{pmatrix} \frac{3055}{9532} & 0 & 0 & 0 \\ 0 & \frac{531}{5956} & 0 & 0 \\ 0 & 0 & \frac{1471}{8094} & 0 \\ 0 & 0 & 0 & \frac{1848}{7919} \end{pmatrix}.$$

Irrespective of the definition of the function $q(n, j)$, the correction formula (2.2) possesses *parallelism across the method*, because the diagonal structure of the matrix D enables us to compute the components of $Y_n^{(j)}$ in parallel. In addition, a suitable definition of the function $q(n, j)$ may determine an ordering by which the iterates $Y_n^{(j)}$ are computed that facilitates *parallelism across the steps*. We shall discuss various options.

2.1. *Order of computation of the iterates*

The conventional PC approach is defined by

$$q(n, j) = m(t_n),$$

for all n and j. By this definition, the only possibility is to compute first the iterates $Y_1^{(j)}, j = 1, 2, \ldots, m(t_1)$, next the iterates $Y_2^{(j)}, j = 1, 2, \ldots, m(t_2)$, etc. This ordering generates the PDIRK method of [12] and [10]. Thus, representing the iterates by points in the (n, j)-plane, the PDIRK method computes the iterates *column-wise*. Obviously, this method does not allow for parallelism across the steps. If the predictor formula defining $Y_n^{(1)}$ requires the same sequential costs as the correction formula in (2.2), then the sequential computational complexity of the PDIRK method is given by $N_{\text{seq}} = \Sigma_n m(t_n)$, N_{seq} denoting the number of implicit systems to be solved.

A second option defines

$$q(n, j) = j - 1, \quad j > 1. \tag{2.4}$$

In this case, there are various possibilities in the ordering by which the iterates can be computed, leading to the same set of iterates. For example, it can again be done column-wise, but also *row-wise*. In the latter case, the iteration scheme {(2.2), (2.4)} may be considered as Jacobi-type iteration possessing a large degree of parallelism across the steps, because for fixed j, all iterates $Y_n^{(j)}$, $n = 1, 2, \ldots, N$, can be computed concurrently. Therefore, it will be called the PDIRKAS J method (PDIRK Across the Steps using Jacobi iteration). The sequential computational complexity of the PDIRKAS J method is reduced to the sequential costs of computing all initial iterates $Y_n^{(1)}$ and the sequential costs of solving $\max_n\{m(t_n)\} - 1$ implicit systems. In actual application, one wants to limit the sequential costs of the predictor formula. In the extreme case, one sets $Y_n^{(1)} = e \otimes y_0$ for all n, so that $N_{\text{seq}} =$

$\max_n\{m(t_n)\} - 1$. The PDIRKAS J method using this strategy has similarities with the step-parallel methods studied by Bellen and co-workers [1,2]. However, such a PDIRKAS J method is expected to exhibit poor convergence due to the inaccuracy of $Y_n^{(1)}$ as n increases and can only be applied on small subintervals (windows). A more robust approach computes $Y_n^{(1)}$ by a predictor formula of at least order one that is sufficiently stable (see section 2.2). Assuming that this predictor formula requires the same sequential costs as the correction formula in (2.2), the total sequential computational costs are given by $N_{\text{seq}} = N - 1 + \max_n\{m(t_n)\}$. Initially, the number of processors needed in this strategy is sN. However, in an actual implementation, iteration at a particular point t_n will be stopped as soon as the corrector solution is obtained within some given tolerance (see section 4), so that the number of processors needed will gradually decrease.

Convergence will often be improved substantially by defining

$$q(n, j) = j. \tag{2.5}$$

Again, many algebraic equivalent orderings are possible (i.e., orderings that generate the same set of iterates). But now, neither the column-wise, nor the row-wise ordering does allow for step-parallelism. The only ordering by which a certain amount of step-parallelism is achieved, computes the iterates along the diagonals $n + j =$ constant, that is, all iterates $Y_n^{(j)}$ with $n + j$ constant are computed concurrently. If we again restrict our considerations to predictor formulas that are equally expensive as the correction formula and if we assume that the iteration process at the end point of the integration interval is stopped only if iteration at all preceding step points has converged, then the total sequential computational costs are now given by $N_{\text{seq}} = N - 1 + m(t_N)$, where again the number of processors is at most sN (but usually less than sN as we saw in our previous discussion of the Jacobi iteration strategy). The iteration scheme defined by (2.5) may be considered as Gauss–Seidel-type iteration and the corresponding integration method will therefore be called the PDIRKAS GS method. We remark that the PDIRKAS GS method {(2.2), (2.5)} is the stiff version of the PIRKAS GS method developed in [13].

In the remainder of this paper, we analyse the PDIRKAS J and PDIRKAS GS methods.

2.2. *The predictor formula*

There are several possibilities in defining a predictor formula for the PDIRKAS method. An implementationally convenient choice defines $Y_n^{(1)}$ by applying a backward differentiation formula (BDF) to the preceding iterate $Y_{n-1}^{(1)}$ to obtain the implicit *stage vector* predictor formula

$$Y_n^{(1)} - h_n(D^* \otimes I_d) F(Y_n^{(1)}) = (E^* \otimes I_d) Y_{n-1}^{(1)}, \tag{2.6}$$

where D^* is assumed to be diagonal. If we choose $D^* = D$, then we can achieve predictor order $q = s - 1$, while the predictor formula and the correction formula

share the same LU decomposition. If both D^* and E^* are defined by order conditions, then we have order $q = s$. However, we need an additional set of s processors in order to compute the LU-decompositions for the predictor formula concurrently with those for the correction formula.

An alternative to (2.6) defines $Y_n^{(1)}$ by applying a BDF to preceding step values $(E \otimes I_d)Y_{n-1}^{(1)}$, $(E \otimes I_d)Y_{n-2}^{(1)}, \ldots$. For example, we may define the *step point* predictor formula

$$Y_n^{(1)} - h_n(D^* \otimes I_d)F(Y_n^{(1)}) = (E_1 \otimes I_d)Y_{n-1}^{(1)} + (E_2 \otimes I_d)Y_{n-2}^{(1)}, \tag{2.7}$$

where D^* is again diagonal, and where the first $s-1$ columns of E_1 and E_2 vanish. Locally, this formula is at most third-order accurate. If $D^* = D$, then only second-order local accuracy can be achieved.

3. Stability and convergence

The stability region and the convergence region of the PDIRKAS methods will be discussed for the familiar basic test equation $y'(t) = \lambda y(t)$, where λ is assumed to run through the spectrum of $\partial f/\partial y$. With respect to this test equation, the stability properties of the PDIRKAS method are determined by the stability of the predictor–corrector pair and the convergence properties of the iteration process. Unlike the situation in conventional PC methods, step-parallel methods as considered here, require the predictor to be stable for integration over the whole interval (row-wise ordering of the iterates). Assuming that the underlying corrector is unconditionally stable (with respect to the basic test equation), the stability region of the PDIRKAS method is the intersection of the region of convergence of the correction formula and the stability region of the predictor formula. At first sight, the stage value predictor formula (2.6) is more attractive, because of its higher order (we recall that the predictor order q equals s or $s-1$), whereas the step point predictor formula (2.7) is at most second-order accurate. However, (2.6) is less stable than (2.7). To see this, we consider the case where all coefficients are determined by order conditions. Furthermore, let the stepsizes be constant, i.e. $h_n = h$. Then, each of the s components of $Y_n^{(1)}$ defined by (2.6) may be considered as the result of applying an s-step BDF with the $s+1$ abscissas $\{t_{n-1} + c_i h, i = 1, \ldots, s; t_{n-1} + h + c_k h\}$ where $k = 1, \ldots, s$. In the case (2.7), each component of $Y_n^{(1)}$ is defined by a two-step BDF with abscissas $\{t_{n-2}, t_{n-2} + h, t_{n-2} + h + c_k h\}$ where $k = 1, \ldots, s$. BDFs with non-uniformly distributed abscissas have been investigated in [8] and were shown to lead to poor stability regions if the spacing of the abscissas is *increasing*. Since in general the spacing of the last two abscissas in formula (2.6) is relatively large for $k > 1$, we cannot expect that (2.6) is sufficiently stable, whereas (2.7) is expected to be L-stable, because its stepsizes are nonincreasing. We also considered the case of (2.6) with $D^* = D$ and we did prove the existence of a family of first-order

predictors which are L(α)-stable for the two-, three- and four-stage Radau IIA correctors using the matrices D as given in (2.3). For example, for the four-stage Radau IIA corrector we computed the angle α and found $\alpha \approx 70°$. Because the stability of the predictor formula is crucial in step-parallel methods, we decided to use the second-order, L-stable step point predictor formula (2.7) with D^*, E_1 and E_2 defined by order conditions.

3.1. *Region of convergence of the correction formula*

In this section, we shall derive the region of convergence for the recursion (2.2) when applied to the test equation. Let us define the stage vector iteration errors

$$\epsilon_n^{(j)} := Y_n^{(j)} - Y_n. \tag{3.1}$$

Subtracting (2.1) and (2.2), we find the linear recursion

$$\begin{aligned} \epsilon_n^{(j)} &= K_n \epsilon_{n-1}^{(q(n-1,j))} + Z_n \epsilon_n^{(j-1)}, \\ K_n &:= (I - z_n D)^{-1} E, \\ Z_n &:= z_n D (I - z_n D)^{-1} (D^{-1} A - I), \\ z_n &:= \lambda h_n, \end{aligned} \tag{3.2}$$

where $n = 1, \ldots, N$. We shall study the convergence of the iteration error vectors

$$\epsilon^{(j)} := \begin{pmatrix} \epsilon_1^{(j)} \\ \epsilon_2^{(j)} \\ \cdots \\ \epsilon_n^{(j)} \end{pmatrix}, \quad z := (z_1, \ldots, z_n)^{\mathrm{T}}. \tag{3.3a}$$

In particular, we are interested in the rate of convergence of the error vectors as function of n. The recursion (3.2) can be represented in the form

$$\epsilon^{(j)} = Q(z)\, \epsilon^{(j-1)} = Q(z)^{j-1} \epsilon^{(1)}, \tag{3.3b}$$

where in the case of the PDIRKAS J and PDIRKAS GS methods the n-by-n block iteration matrix $Q(z)$ is respectively given by

$$Q_{\mathrm{J}}(z) := \begin{pmatrix} Z_1 & O & O & O & \cdots \\ K_2 & Z_2 & O & O & \cdots \\ O & K_3 & Z_3 & O & \cdots \\ O & O & K_4 & Z_4 & \cdots \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{pmatrix},$$

$$Q_{GS}(z) := \begin{pmatrix} Z_1 & O & O & O & \cdots \\ K_2Z_1 & Z_2 & O & O & \cdots \\ K_3K_2Z_1 & K_3Z_2 & Z_3 & O & \cdots \\ K_4K_3K_2Z_1 & K_4K_3Z_2 & K_4Z_3 & Z_4 & \cdots \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{pmatrix}. \tag{3.4}$$

If all matrices Z_i in (3.4) are nondefective, then the spectrum of the matrix $Q(z)$ consists of the eigenvalues of the n matrices Z_i. This observation leads us to the condition of convergence

$$\rho(z_i D(I - z_i D)^{-1}(D^{-1}A - I)) < 1, \quad i = 1, \ldots, n,$$

where $\rho(\cdot)$ denotes the spectral radius function. This condition is identical to that of the PDIRK method. Constant stepsize plots of the convergence region $\mathbf{C} := \{z: \rho(zD(I - zD)^{-1}(D^{-1}A - I)) < 1\}$ for the Radau IIA correctors of orders $p = 3$, 5 and 7 reveal that the whole left halfplane is contained in $\mathbf{C}$. Hence, the Radau IIA based PDIRKAS J and PDIRKAS GS methods may be considered as "A-convergent". Thus, we may conclude that using these Radau IIA correctors leads to PDIRKAS methods whose stability region is completely determined by the stability region of the predictor.

3.2. *Rate of convergence*

Although the *regions* of convergence of the PDIRKAS and PDIRK methods are identical, the *rate* of convergence of the PDIRKAS method may be much worse because of ill-conditioning (or even defectiveness) of the eigensystem of the iteration matrix $Q(z)$. For example, if we integrate with fixed stepsizes, then $Q(z)$ possesses s eigenvalues of geometric multiplicity n leading to rather poor convergence as n increases. The condition of the eigensystem may improve if all stepsizes are distinct, but convergence can still be slow.

In order to get insight into the convergence properties as a function of j and n, we need an estimate for the rate of convergence of the iteration process. In this paper, we shall adopt a definition as given in [17, p. 88], where the *averaged rate of convergence* of the recursion (3.3) is given by

$$R(n, j, z) := -\log\left(\sqrt[j]{\| Q(z)^j \|}\right). \tag{3.5}$$

Let the iteration error associated with $\boldsymbol{Y}_i^{(j)}$, $i = 1, \ldots, n$, be of magnitude $10^{-\Delta(j)}$ (that is, the iterates $\boldsymbol{Y}_i^{(j)}$ and the corrector solutions $\boldsymbol{Y}_i$, $i \leqslant n$, differ by $\Delta(j)$ decimal digits). Then, taking logarithms to base 10, the number of iterations j needed to achieve this is at most

$$j \approx 1 + \frac{\Delta(j) - \Delta(1)}{R(n, j-1, z)}. \tag{3.6}$$

We shall separately discuss the rate of convergence at the origin (*nonstiff* rate of

convergence), at infinity (*stiff* rate of convergence), and the rate of convergence at intermediate points in the whole left halfplane. At the origin, the matrices Q_J and Q_{GS} can be approximated by

$$Q_J(z) = K + \operatorname{diag}(z)L + O(z^2),$$

$$K := \begin{pmatrix} O & O & O & \cdots \\ E & O & O & \cdots \\ O & E & O & \cdots \\ O & O & E & \cdots \\ \cdots & \cdots & \cdots & \cdots \end{pmatrix}, \quad L := \begin{pmatrix} A-D & O & O & O & \cdots \\ DE & A-D & O & O & \cdots \\ O & DE & A-D & O & \cdots \\ O & O & DE & A-D & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \end{pmatrix}, \tag{3.7a}$$

$$Q_{GS}(z) = M \operatorname{diag}(z) + O(z^2),$$

$$M := \begin{pmatrix} A-D & O & O & O & \cdots \\ H & A-D & O & O & \cdots \\ H & H & A-D & O & \cdots \\ H & H & H & A-D & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \end{pmatrix}, \quad H := E(A-D), \tag{3.7b}$$

and at infinity, we obtain

$$Q_J(z) = Q_{GS}(z)$$

$$:= \begin{pmatrix} I - D^{-1}A & O & O & O & \cdots \\ O & I - D^{-1}A & O & O & \cdots \\ O & O & I - D^{-1}A & O & \cdots \\ O & O & O & I - D^{-1}A & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots \end{pmatrix} + O(z^{-1}). \tag{3.8}$$

In the following subsections, the maximum norm is used in the definition of $R(n, j, z)$, and in the tables of computed convergence rates, the underlying corrector is the four-stage Radau IIA method iterated by means of the matrix $D = D_4$ as defined in (2.3).

3.2.1. Convergence of nonstiff error components

From (3.7a) it follows that

$$[Q_J(z)]^j = [K + \operatorname{diag}(z)L + O(z^2)]^j = K^j + O(z). \tag{3.9}$$

Since $\| K^j \|_\infty$ equals 1 for $j < n$ and vanishes as $j \geqslant n$, we have that

$$\begin{aligned} R_J(n, j, z) &= O(z) && \text{for } j < n, \\ R_J(n, j, z) &= O\left(\frac{1}{j} |\log \| z \| |\right) && \text{for } j \geqslant n, \end{aligned} \tag{3.10a}$$

whereas (3.7b) immediately reveals that

$$R_{GS}(n, j, z) = O(|\log||z|||) \quad \text{for all } j. \tag{3.10b}$$

These formulas indicate that with respect to the nonstiff error components the convergence of Jacobi iteration is unacceptably slow. Therefore, in the remainder of this paper, we confine our discussions to the PDIRKAS GS method.

Let us consider convergence in more detail for *fixed* stepsizes, i.e. $z_i = z$ for all i. From (3.7b) it follows that

$$R_{GS}(n, j, z) = -\log|z| - \log\left(\sqrt[j]{||M^j||_\infty} + O(z)\right). \tag{3.11}$$

The following theorem provides explicit formulas for the asymptotic behaviour of the nonstiff rate of convergence for large values of j and n, respectively.

Theorem 3.2

For fixed values of n, the nonstiff rate of convergence of the PDIRKAS GS method satisfies the asymptotic relation

$$R_{GS}(n, j, z) = -\log(\rho(A - D)|z|) - O(j^{-1}\log(j)) \quad \text{as } j \to \infty \quad \text{and} \quad z \to 0. \tag{3.12}$$

If the matrices A and D satisfy the conditions $a_{ss} < d_s < 1$ and $a_{sk} > 0$ $(k = 1, \ldots, s)$, then for fixed j

$$R_{GS}(n, j, z) = -\log(n|z|) - \log\left[(1 - d_s)\sqrt[j]{\frac{1 - 2a_{ss} + d_s}{j!(1 - d_s)}} + O(n^{-1})\right]$$
$$\text{as } n \to \infty \quad \text{and} \quad z \to 0. \tag{3.13}$$

Proof

The formula (3.12) immediately follows from the asymptotic formula for the norm of powers of matrices (see e.g. [17]). Assertion (3.13) can be proved along the lines of a similar theorem given in [13]. According to this proof, it is first shown that

$$||M^j||_\infty = ||M_0^j||_\infty + O(n^{j-1}) \quad \text{as } n \to \infty, \tag{3.14}$$

where

$$M_0 := \begin{pmatrix} O & O & O & \cdots & O \\ H & O & O & \cdots & O \\ H & H & O & \cdots & O \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ H & H & \cdots & H & O \end{pmatrix}, \quad H := E(A - D).$$

Next, it is shown that

$$||M_0^j||_\infty = \frac{1}{j!}n^j||H^j||_\infty + O(n^{j-1}) \quad \text{as } n \to \infty. \tag{3.15}$$

By observing that H satisfies the recursion $H^j = (1 - d_s)^{j-1}H$, and using the assumptions $a_{sk} > 0$, $a_{ss} < d_s < 1$, we find

$$\| H^j \|_\infty = (1 - d_s)^{j-1}(1 - 2a_{ss} + d_s). \tag{3.16}$$

On substitution into (3.15) and into (3.14) formula (3.13) is immediate. □

If we consider the error over the whole integration interval, i.e. $n = N$, then this theorem shows that the nonstiff rate of convergence R_{GS} rapidly converges to a constant value as N increases and j is kept fixed. In this connection, we remark that the nonstiff rate of convergence of the PDIRK method is given by

$$R_{PDIRK}(j, z) = -\log\left(\sqrt[j]{\| Z^j \|}\right) = -\log|z| - \log\left(\sqrt[j]{\| (A - D)^j \|_\infty} + O(z)\right), \tag{3.17}$$

showing that the nonstiff rate of convergence R_{PDIRK} behaves as $O(\log(N))$ as N increases.

3.2.2. Convergence of stiff error components

If $z \to \infty$, then (3.8) immediately yields the following theorem:

Theorem 3.3

If $z \to \infty$ and if n is finite, then for any corrector (2.1), the rate of convergence of the PDIRKAS GS method is given by

$$R_{GS}(n, j, \infty) = -\frac{1}{j}\log \| (I - D^{-1}A)^j \|_\infty. \tag{3.18}$$ □

We remark that the stiff rate of convergence of the PDIRKAS GS method is identical to that of the PDIRK method and does not depend on n. Table 1 lists the values for a few values of j.

3.2.3. Convergence at intermediate values of z

The preceding subsections indicate that the stiff and nonstiff rates of convergence of the PDIRKAS GS method are quite satisfactory, even for larger values of n. However, as soon as we move away from the origin or from infinity, then the rate of convergence deteriorates. Tables 2a and 2b respectively list values of Min $\{R_{GS}(n, j, z): z \leqslant 0\}$ and Min $\{R_{GS}(n, j, z): \mathrm{Re}(z) \leqslant 0\}$ for the four-stage Radau IIA corrector for a few values of n. In the latter case, the minimal rate of

Table 1
Stiff $R_{GS}(n, j, \infty)$ values for the four-stage Radau IIA corrector.

$j = 1$	$j = 2$	$j = 4$	$j = 8$	$j = 16$	$j = 32$	$j = \infty$
−0.67	−0.52	0.15	0.82	1.22	1.40	1.60

Table 2a
Values of Min $\{R_{GS}(n, j, z): z \leqslant 0\}$ and j_Δ for the four-stage Radau IIA corrector.

j	$n=1$	$n=2$	$n=4$	$n=8$
1	−0.67	−0.67	−0.67	−0.67
2	−0.52	−0.52	−0.52	−0.54
4	−0.01	−0.25	−0.36	−0.43
8	0.33	0.09	−0.16	−0.33
16	0.52	0.35	0.12	−0.12
32	0.60	0.51	0.35	0.14
$j_\Delta(n)$	19	23	31	42
$S(n)$	1	1.6	2.2	3.1

Table 2b
Values of Min $\{R_{GS}(n, j, z): \mathrm{Re}(z) \leqslant 0\}$ and j_Δ for the four-stage Radau IIA corrector.

j	$n=1$	$n=2$	$n=4$	$n=8$
1	−0.67	−0.81	−0.94	−1.05
2	−0.52	−0.60	−0.75	−0.90
4	−0.14	−0.38	−0.57	−0.75
8	0.09	−0.12	−0.35	−0.45
16	0.18	0.02	−0.13	−0.29
32	0.23	0.15	0.03	−0.11
64	0.25	0.21	0.14	0.02
$j_\Delta(n)$	42	52	69	102
$S(n)$	1	1.6	2.3	3.1

convergence was always found on the imaginary axis. Tables 2 show that for larger values of n, the rate of convergence only becomes positive if the number of iterations is relatively large, particularly in the case of table 2b. The effect on the corresponding iteration error components is disastrous (even for relatively low values of n), because these components start to grow exponentially and will only be damped if j is relatively large. In order to illustrate this, tables 2a and 2b also list the values of $j = j_\Delta(n)$ given by (3.6) with $\Delta - \Delta_1 = 10$ and $z = ze$. The rather large values of $j_\Delta(n)$ as n increases indicate that the iterates may easily become so bad that we have overflow before the iteration process starts to converge. Therefore, some strategy should be employed that controls when it is safe to advance to the next step point (see section 4.3). Finally, we remark that by means of the values of j_Δ we can compute an estimate of the speed-up factor of PDIRKAS GS with respect to PDIRK. Setting $n = N$, the number of sequential iterations of these methods are respectively given by $N + j_\Delta(N) - 1$ and $Nj_\Delta(1)$, resulting in the speed-up factor $S(N) = Nj_\Delta(1)[N + j_\Delta(N) - 1]^{-1}$ (notice that the speed-up factors along the negative and imaginary axis are roughly equal).

4. Numerical experiments

The PDIRKAS GS method {(2.2), (2.5)} described above was applied using the four-stage Radau IIA corrector equation and the predictor formula (2.7). Since the number $m(t_n)$ of outer iterations needed to solve the corrector equation will strongly depend on n, we applied a dynamic iteration strategy with stopping criterium (cf. [13])

$$\Delta_n^{(j)} = \frac{\| (e_s^{\mathrm{T}} E \otimes I)(Y_n^{(j-1)} - Y_n^{(j)}) \|_1}{\| (e_s^{\mathrm{T}} E \otimes I) Y_n^{(j-1)} \|_1} \leqslant TOL_{\mathrm{corr}}.$$

In all our experiments, we set $TOL_{\mathrm{corr}} = 10^{-12}$. The number of necessary processors is determined by the number of step points at which this stopping criterion is not yet satisfied. The maximal number of processors needed during the integration equals $sK_{\max}$, where $K_{\max}$ denotes the maximal number of step points where the stopping criterion is not yet satisfied. For the inner iteration process for solving the correction formula in (2.2) we used a modified Newton method which was solved to convergence.

In addition to the PDIRKAS GS method, we shall also apply the PDIRK method that may be considered as the PDIRKAS GS method in one-processor mode. In this paper, we want to compare characteristic properties of the methods like the rate of convergence and sequential costs, rather than strategy aspects such as stepsize and error control. Therefore, we restrict the experiments to problems that can be integrated with fixed stepsizes $h = N^{-1}T$. In a sequel to this paper, we will develop a stepsize and error control strategy [18].

The calculations were performed using 15-digits arithmetic. The accuracy is given by the number of correct digits Δ, obtained by writing the maximum norm of the absolute error at the endpoint in the form $10^{-\Delta}$.

4.1. *Test problems*

Our first problem is the well known stability test problem of Prothero and Robinson

$$\frac{dy}{dt} = -\epsilon^{-1}(y - g(t)) + g'(t), \quad y(0) = g(0), \quad 0 \leqslant t \leqslant T, \tag{4.1a}$$

where the exact solution equals $g(t)$ and ϵ is a small parameter. Prothero and Robinson used this problem to show the order reduction of RK methods when ϵ is small. In our experiments we set

$$g(t) = \cos(t), \quad \epsilon = 10^{-3}. \tag{4.1b}$$

The second test problem is the "nonlinearization" of problem (4.1):

$$\frac{dy}{dt} = -\epsilon^{-1}(y^3 - g(t)^3) + g'(t), \quad y(0) = g(0), \quad 0 \leqslant t \leqslant T, \tag{4.2a}$$

with exact solution $y(t) = g(t)$ for all values of the parameter ϵ. As in the preceding problem, we set

$$g(t) = \cos(t), \quad \epsilon = 10^{-3}. \tag{4.2b}$$

The third test problem is that of Kaps [14]:

$$\begin{aligned} \frac{dy_1}{dt} &= -(2+\epsilon^{-1})y_1 + \epsilon^{-1}(y_2)^2, \\ \frac{dy_2}{dt} &= y_1 - y_2(1+y_2), \\ y_1(0) &= y_2(0) = 1, \quad 0 \leqslant t \leqslant T, \end{aligned} \tag{4.3}$$

with the smooth exact solution $y_1 = \exp(-2t)$ and $y_2 = \exp(-t)$ for all values of the parameter ϵ. This problem belongs to the class of problems for which stiffly accurate RK methods do not suffer order reduction whatever small ϵ is (cf. [9]).

The test set of Enright et al. [7] contains the following system of ODEs describing a chemical reaction:

$$\frac{dy}{dt} = -\begin{pmatrix} 0.013 + 1000y_3 & 0 & 0 \\ 0 & 2500y_3 & 0 \\ 0.013 & 0 & 1000y_1 + 2500y_2 \end{pmatrix} y, \tag{4.4a}$$

with $y(0) = (1, 1, 0)^T$. Since we want to use fixed step sizes in our experiments, we avoided the initial phase by choosing the starting point at $t_0 = 1$. The corresponding initial and end point values at $t = T = 51$ are given by

$$y(1) \approx \begin{pmatrix} 0.990731920827 \\ 1.009264413846 \\ -0.366532612659 \times 10^{-5} \end{pmatrix}, \quad y(51) \approx \begin{pmatrix} 0.591045966680 \\ 1.408952165382 \\ -0.186793736719 \times 10^{-5} \end{pmatrix}. \tag{4.4b}$$

The final test example is taken from Lambert [15, p. 228]:

$$\frac{dy}{dt} = \begin{pmatrix} 42.2 & 50.1 & -42.1 \\ -66.1 & -58.0 & 58.1 \\ 26.1 & 42.1 & -34.0 \end{pmatrix} y, \tag{4.5a}$$

with $y(0) = (1, 0, 2)^T$. As soon as the fast transient e^{-50t} has died out, the exact solution is sinusoidal with a slowly increasing amplitude:

$$y(t) = \begin{pmatrix} e^{t/10}\sin(8t) + e^{-50t} \\ e^{t/10}\cos(8t) - e^{-50t} \\ e^{t/10}(\sin(8t) + \cos(8t)) + e^{-50t} \end{pmatrix}. \tag{4.5b}$$

The eigenvalues of this system are given by -50 and $1/10 \pm 8i$, hence we are faced with a stiff problem, the nonstiff solution components of which are nondissipative.

As pointed out in Lambert's discussion of the system (4.5), such problems are a difficult test for L-stable methods like our Radau IIA based PDIRKAS GS method. As in the preceding example, the initial phase is avoided by starting the integration at $t_0 > 0$. In fact, we integrated the interval [0.5, 1.5].

4.2. *Comparison of the PDIRKAS GS and the PDIRK method*

In our first tests, we compare results obtained by the PDIRKAS GS and the PDIRK method. We apply the PDIRKAS GS in unlimited-number-of-processors mode and in one-processor mode (by which we generate the PDIRK method). The sequential computational complexity is measured by the total number $N_{\text{seq}} = N - 1 + m(t_N)$ of sequential implicit systems to be solved during the integration process. Furthermore, we define the average number of iterations per step and the average number of *sequential* iterations per step by $m^* := N^{-1}\Sigma_n m(t_n)$ and

Table 3
Results for the linear Prothero–Robinson problem (4.1) with $T = 1$.

N	Δ	$K_{\max}$	m^*	N_{seq}	m^*_{seq}	$S(N)$
1	6.3	1	10.0	10	10.0	1.0
2	7.4	2	10.5	13	6.5	1.5
4	8.6	4	12.8	19	4.8	2.2
8	9.8	8	17.3	32	4.0	2.8
16	11.0	13	26.9	59	3.7	3.1

Table 4
Results for the nonlinear Prothero–Robinson problem (4.2) with $T = 1$.

N	Δ	$K_{\max}$	m^*	N_{seq}	m^*_{seq}	$S(N)$
1	6.3	1	10.0	10	10.0	1.0
2	7.3	2	9.5	12	6.0	1.6
4	8.5	4	11.3	18	4.5	2.1
8	9.7	7	15.4	28	3.5	2.9
16	11.0	13	22.7	53	3.3	3.2

Table 5a
Results for the Kaps problem (4.3) with $\epsilon = 10^{-3}$ and $T = 1$.

N	Δ	$K_{\max}$	m^*	N_{seq}	m^*_{seq}	$S(N)$
1	5.0	1	12.0	12	12.0	1.0
2	6.4	2	13.0	15	7.5	1.7
4	7.8	4	15.3	22	5.5	2.3
8	9.1	8	20.9	36	4.5	2.8
16	10.3	14	31.4	64	4.0	3.4

Table 5b
Results for the Kaps problem (4.3) with $\epsilon = 10^{-8}$ and $T = 1$.

N	Δ	$K_{\max}$	m^*	N_{seq}	m^*_{seq}	$S(N)$
1	6.6	1	12.0	12	12.0	1.0
2	8.7	2	10.5	13	6.5	1.6
4	10.8	4	9.3	14	3.5	2.6

Table 6
Results for the chemical reaction problem (4.4).

N	Δ	$K_{\max}$	m^*	N_{seq}	m^*_{seq}	$S(N)$
1	7.9	1	9.0	9	9.0	1.0
2	9.8	2	7.5	10	5.0	1.5
4	11.8	4	7.0	11	2.8	2.5

Table 7
Results for the nondissipative problem (4.5) with $T = 1.5$.

N	Δ	$K_{\max}$	m^*	N_{seq}	m^*_{seq}	$S(N)$
10	5.9	10	18.5	33	3.3	4.7
20	8.1	14	20.7	53	2.6	4.6
40	10.2	21	25.1	85	2.1	4.6
80	12.3	33	30.4	138	1.7	4.8

$m^*_{\text{seq}} := N^{-1}N_{\text{seq}}$, respectively. For the PDIRK method, we obviously have $N_{\text{seq}} = \Sigma_n m(t_n)$ and $m^* = m^*_{\text{seq}} = N^{-1}N_{\text{seq}}$. The ratio of the values of m^*_{seq} for the PDIRKAS GS and PDIRK methods determines the speed-up factor $S(N)$.

Tables 3 to 7 present multi-processor results for the PDIRKAS GS method and the speed-up factors $S(N)$. From these results, we conclude that the PDIRKAS GS method becomes more efficient as the number of step points N increases and that the speed-up factors $S(N)$ are in good agreement with the theoretical speed-up factors listed in tables 2a and 2b.

In order to see the effect of larger intervals of integration, we repeated the experiments for the linear Prothero–Robinson problem (4.1) and the Kaps problem (4.3), but now on the interval [0, 10]. The results in tables 8 and 9 reveal that the speed-up factor is much larger than in tables 3 and 5 (for the same stepsize) with a maximal speed-up factor of about 5, but we also see that the convergence behaviour in the Prothero–Robinson problem and the mildly stiff Kaps problem now becomes worse as N becomes too large. This is due to the deterioration of the rate of convergence as discussed in subsection 3.2.3. For the Kaps problem with $\epsilon = 10^{-8}$, table 9b shows that this deterioration does not occur and hence the limiting value $m^*_{\text{seq}} = 1$ is almost obtained.

Table 8
Results for the linear Prothero–Robinson problem (4.1) with $T = 10$.

J	Δ	K_{max}	m^*	N_{seq}	m^*_{seq}	$S(N)$
10	6.9	9	17.2	31	3.1	3.6
20	7.7	16	22.4	46	2.3	4.7
40	8.7	29	32.6	77	1.9	5.7
80	10.0	55	69.2	172	2.1	5.2
60		...	divergence		...	

Table 9a
Results for the Kaps problem (4.3) with $\epsilon = 10^{-3}$ and $T = 10$.

J	Δ	K_{max}	m^*	N_{seq}	m^*_{seq}	$S(N)$
10	9.5	10	22.0	39	3.9	4.1
20	11.6	17	30.6	65	3.3	3.9
40	13.7	30	47.5	110	2.8	4.3
80	15.8	57	92.6	245	3.1	3.8
60		...	divergence		...	

Table 9b
Results for the Kaps problem (4.3) with $\epsilon = 10^{-8}$ and $T = 10$.

J	Δ	K_{max}	m^*	N_{seq}	m^*_{seq}	$S(N)$
10	9.5	10	20.3	36	3.6	4.5
20	11.6	15	20.1	49	2.5	5.1
40	13.7	21	22.7	75	1.9	5.6
80	16.0	28	24.3	117	1.5	5.9
60	17.5	33	24.8	198	1.2	6.4

4.3. *Dynamic PDIRKAS GS method*

The preceding experiments indicate that the performance of the PDIRKAS GS method strongly depends on the problem to be solved. Therefore, we should apply a strategy that controls when it is safe to move to the next time point t_n. One strategy is to take the local truncation error of the last component of the iterate $Y^{(j)}_{n-1}$ as a measure for safety (in fact, such a strategy was used in [13] for *nonstiff* problems). However, in the present case of *stiff* problems, the BDF predictor formula (2.7) often computes highly accurate first iterates $Y^{(1)}_n$ for all n, so that control of its local truncation error will not be effective. The cause of a potential bad performance is a strong initial grow of the iteration error. Hence, an alternative strategy might be a check on the behaviour of the iteration error, e.g. by means of the residue of the corrector equation (2.1a). If this residue does not grow anymore, then it should be safe to advance to the next step point.

Let us define the residual function

$$\boldsymbol{R}_n^{(j)} := \boldsymbol{Y}_n^{(j)} - (E \otimes I_d)\boldsymbol{Y}_{n-1}^* - h_n(A \otimes I_d)\boldsymbol{F}(\boldsymbol{Y}_n^{(j)}), \tag{4.6}$$

where $\boldsymbol{Y}_{n-1}^*$ denotes the most recent iterate available at t_{n-1} (this iterate will depend on j too). One option is to require that for some iteration index i

$$\| e_s^{\mathrm{T}} \boldsymbol{R}_{n-k}^{(i)} \|_\infty < a \| e_s^{\mathrm{T}} \boldsymbol{R}_{n-k}^{(1)} \|_\infty, \quad i \geqslant 2, \quad k \geqslant 1, \quad a \leqslant 1, \tag{4.7}$$

before advancing to t_n. Here, a is some safety parameter and k determines the point where the iteration errors are checked. Since the first few iterations at t_{n-k} may have an erratic behaviour, we should choose k greater than 1. Setting $j^*(t_{n-1})$ equal to the current iteration index j at t_{n-1}, as soon as the residue at t_{n-k} is sufficiently small, we can compute the iterate $\boldsymbol{Y}_n^{(j)}$ according to the formula

$$\boldsymbol{Y}_n^{(j)} - h_n(D \otimes I_d)\boldsymbol{F}(\boldsymbol{Y}_n^{(j)}) = (E \otimes I_d)\boldsymbol{Y}_{n-1}^{(j+j^*-1)} + h_n((A-D) \otimes I_d)\boldsymbol{F}(\boldsymbol{Y}_n^{(j-1)}),$$
$$j = 2, \ldots, m(t_n). \tag{4.8}$$

The sequential costs are now proportional to the number

$$N_{\mathrm{seq}} = \sum_{n=1}^{N-1} (j^*(t_n)) + m(t_N).$$

Tables 10, 11 and 12 are the analogues of tables 8, 9 and 7, respectively. By virtue of the strategy (4.7), using $a = 10^{-2}$ and $k = 3$, divergence of the iteration process is avoided at the costs of a modest increase of the sequential costs. However, $K_{\max}$ is also much lower, which decreases the number of processors substantially.

Table 10
Results for the linear Prothero–Robinson problem (4.1) with $T = 10$.

N	Δ	$K_{\max}$	m^*	N_{seq}	m^*_{seq}	$S(N)$
10	6.9	7	14.3	31	3.1	3.6
20	7.6	8	16.0	55	2.8	3.9
40	8.8	8	17.5	108	2.7	3.9
80	10.0	8	19.3	230	2.9	3.8
160	11.3	8	19.8	513	3.2	3.6

Table 11a
Results for the Kaps problem (4.3) with $\epsilon = 10^{-3}$ and $T = 10$.

N	Δ	$K_{\max}$	m^*	N_{seq}	m^*_{seq}	$S(N)$
10	9.5	7	18.4	39	3.9	4.1
20	11.6	10	20.7	65	3.3	3.9
40	13.7	14	23.2	116	2.9	4.2
80	15.8	17	25.4	248	3.1	3.8
160	17.7	9	23.8	532	3.3	3.6

Table 11b
Results for the Kaps problem (4.3) with $\epsilon = 10^{-8}$ and $T = 10$.

N	Δ	K_{max}	m^*	N_{seq}	m^*_{seq}	$S(N)$
10	9.5	8	17.8	36	3.6	4.5
20	11.6	7	13.8	49	2.5	5.1
40	13.7	9	13.0	76	1.9	5.3
80	15.8	9	10.5	127	1.6	5.0
160	16.9	10	8.9	233	1.5	5.1

Table 12
Results for the nondissipative problem (4.5) with $T = 1.5$.

N	Δ	K_{max}	m^*	N_{seq}	m^*_{seq}	$S(N)$
10	5.9	8	16.9	34	3.4	4.5
20	8.1	10	16.9	54	2.7	4.5
40	10.2	15	19.0	85	2.1	4.6
80	12.3	20	20.6	138	1.7	4.8

Summarizing, we may conclude that the PDIRKAS GS method using the strategy defined by (4.7) is rather robust. As to the sequential costs, it follows from the tables of results, that the effective number of iterations per step for solving the four-stage Radau IIA RK corrector varies from 1.5 to at most 4 iterations per step. With respect to the PDIRK method, the speed-up factors are in the range 3.5 until 5. Taking into account that the variable step version of the PDIRK method (viz. the PSODE code described in [16]) is about twice as fast as the best sequential codes such as LSODE, we may expect that the variable step version of the PDIRKAS GS method will give rise to speed-up factors in the range 7 until 10 with respect to LSODE. This variable step version will be discussed in a future paper [18].

References

[1] A. Bellen, Parallelism across the steps for difference and differential equations, in: *Numerical Methods for Ordinary Differential Equations*, Lecture Notes in Mathematics 1386 (Springer, 1987) pp. 22–35.

[2] A. Bellen, R. Vermiglio and M. Zennaro, Parallel ODE-solvers with stepsize control, J. Comp. Appl. Math. 31 (1990) 277–293.

[3] K. Burrage, The search for the Holy Grail, or Predictor–Corrector methods for solving ODEIVPs, Appl. Numer. Math. 11 (1993) 125–141.

[4] J.C. Butcher, On the convergence of numerical solutions to ordinary differential equations, Math. Comp. 20 (1966) 1–10.

[5] J.C. Butcher, *The Numerical Analysis of Ordinary Differential Equations, Runge–Kutta and General Linear Methods* (Wiley, Chichester/New York/Brisbane/Toronto/Singapore, 1987).

[6] P. Chartier, Parallelism in the numerical solution of initial value problems for ODEs and DAEs, Thesis, Université de Rennes I, France (1993).
[7] W.H. Enright, T.E. Hull and B. Lindberg, Comparing numerical methods for stiff systems of ODEs, BIT 15 (1975) 10–48.
[8] C.W. Gear and K.W. Tu, The effect of variable mesh size on the stability of multistep methods, SIAM J. Numer. Anal. 11 (1974) 1025–1043.
[9] E. Hairer and G. Wanner, *Solving Ordinary Differential Equations, II: Stiff and Differential Algebraic Problems* (Springer, Berlin, 1991).
[10] P.J. van der Houwen and B.P. Sommeijer, Iterated Runge–Kutta methods on parallel computers, SIAM J. Sci. Stat. Comp. 12 (1991) 1000–1028.
[11] P.J. van der Houwen and B.P. Sommeijer, Preconditioning in parallel Runge–Kutta methods for stiff initial value problems, to appear in CMA (1994).
[12] P.J. van der Houwen, B.P. Sommeijer and W. Couzy, Embedded diagonally implicit Runge–Kutta algorithms on parallel computers, Math. Comp. 58 (1992) 135–159.
[13] P.J. van der Houwen, B.P. Sommeijer and W.A. van der Veen, Parallel iteration across the steps of high order Runge–Kutta methods for nonstiff initial value problems, to appear in JCAM (1994).
[14] P. Kaps, Rosenbrock-type methods, in: *Numerical Methods for Stiff Initial Value Problems*, eds. G. Dahlquist and R. Jeltsch, Bericht nr. 9, Inst. für Geometrie und Praktische Mathematik der RWTH Aachen (1981).
[15] J.D. Lambert, *Numerical Methods for Ordinary Differential Equations*, (Wiley, New York, 1991).
[16] B.P. Sommeijer, Parallelism in the numerical integration of initial value problems, Thesis, University of Amsterdam (1992).
[17] D.M. Young, *Iterative Solution of Large Linear Systems* (Academic Press, New York, 1971).
[18] W.A. van der Veen, Performance of step-parallelism in Runge–Kutta methods for stiff initial-value problems, in preparation (1994).