1992

P.J. van der Houwen, B.P. Sommeijer

Parallel Jacobi iteration in implicit step-by-step methods

Department of Numerical Mathematics Report NM-R9218 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 Jacobi Iteration in Implicit Step-by-Step Methods

P.J. van der Houwen & B.P. Sommeijer
CWI
Post box 4079, 1009 AB Amsterdam, The Netherlands

Abstract. An iteration scheme is described to solve the implicit relations that result from the application of an implicit integration method to an initial value problem (IVP). In this iteration scheme the amount of implicitness is still free so as to comprise a large variety of methods, running from fully explicit (functional iteration) to fully implicit (Newton's method). In the intermediate variants (the so-called Jacobi-type methods), the influence of the Jacobian matrix of the problem is gradually increased. Special emphasis is placed on the 'stage-value-Jacobi' iteration which uses only the diagonal of the Jacobian matrix. Therefore, the convergence of this method crucially depends on the diagonally dominance of the Jacobian. Another characteristic of this scheme is that it allows for massive parallelism: for a d-dimensional IVP, d uncoupled systems of dimension s have to be solved, where s is the number of stages in the underlying implicit method (e.g., an s-stage Runge-Kutta method). Hence, on a parallel architecture with d processors (d>>1), we may expect an efficient process (for high-dimensional problems).

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

1. Introduction

We shall be concerned with parallel predictor-corrector iteration of implicit step-by-step methods for solving initial value problems (IVPs). For a wide class of functional equations, including ordinary differential equations (ODEs), Volterra integral equations (VIEs), Volterra integro-differential equations (VIDEs), delay-differential equations (DDEs), etc., these step-by-step methods (referred to as corrector equations) can be represented in the form:

$$\begin{aligned} &\mathbf{Y} = F_n(h, \mathbf{U}_0, \mathbf{U}_1, \ldots, \mathbf{U}_n) + h^{\nu}(M\otimes I)G_n(\mathbf{Y}), \quad \mathbf{U}_{n+1} = H_n(h, \mathbf{U}_0, \mathbf{U}_1, \ldots, \mathbf{U}_n, \mathbf{Y}), \\ &\mathbf{Y} := (\mathbf{Y}_1^T, \mathbf{Y}_2^T, \ldots, \mathbf{Y}_s^T)^T, \quad \mathbf{U}_n := (\mathbf{U}_{n1}^T, \mathbf{U}_{n2}^T, \ldots, \mathbf{U}_{nr}^T)^T, \quad n = 0, 1, 2, \ldots . \end{aligned} \tag{1.1}$$

Here, h is the stepsize, ν is the order of the IVP, M is an s-by-s matrix characterizing the corrector, $\mathbf{U}_n$ and $\mathbf{Y}$ present an r-dimensional and s-dimensional block vector of numerical approximations to the exact solution of the IVP. If the IVP has dimension d, then $\mathbf{U}_n$ and $\mathbf{Y}$ are vectors in rd-dimensional and sd-dimensional vector spaces, respectively, and F_n, G_n and H_n are functions depending both on the IVP and the step-by-step method. Furthermore, $M\otimes I$ denotes the direct product of the matrices M and I. In each step, the block vectors $\{\mathbf{U}_0, \mathbf{U}_1, \ldots, \mathbf{U}_n\}$ are the input vectors, $\mathbf{U}_{n+1}$ is the output vector, and $\mathbf{Y}$ is the internal stage vector. We shall say that the corrector method has s internal stages and r output points. The representation (1.1) is similar to the partitioned general linear method (GLM) format introduced in [5].

On sequential computers, multi-stage corrector equations are seldom used in predictor-corrector iteration methods, because of the increased computational complexity if s > 1. However, parallel computers have changed the scene. A number of papers [15, 9, 11, 2, 3, 4, 14, 16] discuss the parallel aspects of *functional* iteration of Runge-Kutta-type correctors in the case of first-order and second-order, nonstiff ODEs and show that the *sequential* costs can be reduced to such an extent that they are at least competitive with, but often superior to, the best sequential codes. For stiff ODEs and VIEs, it has been shown in [10, 7] that so-called *diagonally implicit* iteration of Runge-Kutta-type correctors is suitable for implementation on parallel computers (see also Section 2 of the present paper). In [8], these functional and

Report NM-R9218
ISSN 0169-0388
CWI
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

diagonally implicit iteration methods are discussed for solving the general class of correctors defined by (1.1) and preconditioning techniques for accelerating their convergence are studied.

In this paper, we consider another approach to accelerating the convergence of parallel iteration methods, which leads us to *Jacobi-type* iteration methods. For nonstiff problems, we investigate Jacobi iteration methods that are implicit in the s stage values Y_{iq} $(i = 1, \dots, s)$ corresponding to the qth component of the stage vectors Y_i, and we show that its computational costs per step are hardly higher than those of function iteration. This type of Jacobi iteration will be called *stage-value-Jacobi* iteration. It turns out that diagonal dominancy of the Jacobian of the function G_n plays a crucial role in the rate of convergence of stage-value-Jacobi iteration. This is not surprising, because, as is well known, diagonal dominancy also plays an important role in classical (point-)Jacobi iteration. For example, we have the following classical theorem, the proof of which can be found in Collatz [6]:

Theorem 1.1. Let the matrix A in the linear system $Ax = b$ be irreducibly diagonally dominant. Then the point-Jacobi iteration method

$$x_{n+1} = x_n - D^{-1}\,[Ax_n - b], \quad n = 0, 1, \dots$$

converges for any starting vector x_0. □

For linear problems, we shall derive a safe estimate for the convergence factor of stage-value-Jacobi iteration and it will be shown that for IVPs with strongly diagonally dominant Jacobian matrix, we obtain fast convergence, in spite of the modest degree of implicitness of the method. For a number of numerical examples, we compare its efficiency with that of function iteration and we test the reliability of the convergence factor estimate.

2. Parallel iteration methods

We shall study iterative methods for solving the stage vector equation on parallel computers. Let us write the stage vector equation in (1.1) in the form

$$(2.1) \qquad R_n(h, \mathbf{Y}) := \mathbf{Y} - F_n(h, \mathbf{U}_0, \mathbf{U}_1, \dots, \mathbf{U}_n) - h^\nu (M \otimes I) G_n(\mathbf{Y}) = \mathbf{0},$$

and consider Jacobi-type iteration methods of the form

$$(2.2) \qquad I_q\big(\mathbf{Y}^{(j)} - h^\nu Q G_n(\mathbf{Y}^{(j-1)} + I_q(\mathbf{Y}^{(j)} - \mathbf{Y}^{(j-1)}))\big) = I_q\big(\mathbf{Y}^{(j-1)} - h^\nu Q G_n(\mathbf{Y}^{(j-1)}) - P R_n(h, \mathbf{Y}^{(j-1)})\big), \quad q = 1, \dots, k,$$

where the iteration index j runs from 1 to m. Here, P and Q are real, nonzero sd-by-sd matrices, and for a given value of q, I_q is an sd-by-sd diagonal matrix of which the diagonal entries are either 1 or 0 (if $Q = O$, then (2.2) becomes fully explicit and reduces to *functional iteration*).

Each iteration in (2.2) requires itself the application of an iteration process for computing $\mathbf{Y}^{(j)}$. This iteration process will be called the *inner* iteration method and the iteration method (2.2) will be called the *outer* iteration method. It will be assumed that the inner iteration is defined by the modified Newton method.

P may be considered as a *preconditioning matrix* and the matrices Q and I_q determine the degree of implicitness of the iteration scheme. It will be assumed that the matrix obtained by summing all matrices I_q equals the identity matrix I, so that all components of the stage vector are iterated. Each iteration of the iteration method (2.2) requires the solution of a set of k uncoupled, implicit subsystems of dimension Trace(I_q). Hence, it can be efficiently implemented on a k-processor computer.

There are various obvious options for choosing the 'partitioning' matrices I_q. Denoting the unit vector (with only unit entries) and the qth unit vector by **e** and $\mathbf{e}_q$, respectively, both having dimension sd, we recognize the following special cases:

Point-Jacobi:	$k = sd$	$\mathrm{Trace}(I_q) = 1$	$I_q\mathbf{e} = \mathbf{e}_q$
Stage-value-Jacobi:	$k = d$	$\mathrm{Trace}(I_q) = s$	$I_q\mathbf{e} = \mathbf{e} \otimes \mathbf{e}_q$
Stage-vector-Jacobi:	$k = s$	$\mathrm{Trace}(I_q) = d$	$I_q\mathbf{e} = \mathbf{e}_q \otimes \mathbf{e}$
Newton:	$k = 1$	$\mathrm{Trace}(I_q) = sd$	$I_q = I$,

where $q = 1, \dots, k$. The most simple option is point-Jacobi iteration. It has optimal parallelism in the sense that k is as large as possible. The next simple option is stage-value-Jacobi iteration. It allows for massive parallelism for large systems ($k = d$). The qth processor iterates on the s stage values Y_{iq} $(i = 1, \dots, s)$ corresponding to the qth component of the stage vectors $\mathbf{Y}_i$, so that per step each processor has to solve m systems of equations of equal dimension s.

However, in actual computations, the major part of the computational effort per step per processor usually goes into the evaluation of ms components of the residual function $R_n(h, Y^{(j-1)})$. A disadvantage of stage-value iteration may be the poor load balancing if the computational complexity of the components of the residual function vary widely. This disadvantage disappears in the case of stage-vector-Jacobi iteration, where the qth processor iterates on the d components Y_{qi} (i = 1, ... , d) of the qth stage vector Y_q. Now the systems of equations have dimension d, so that for larger dimensions d the major part of the computational effort per step per processor consists of solving these d-dimensional systems. For IVPs originating from ODEs and VIEs, this iteration method has been analysed in the case where $P = I$ and $Q = D\otimes I$ with D an s-by-s matrix (cf. [10, 7] where this type of iteration was called *diagonally implicit iteration*). It was shown that the sets of equations are of comparable computational complexity, so that we have more or less equal load balancing of the processors. Stage-vector-Jacobi iteration has the additional advantage of using the full Jacobian matrix of the IVP in the inner iteration which enables us to solve *stiff* systems efficiently. The disadvantage is the low number of processors that can efficiently be employed ($k = s$). At the end of the scale, we have Newton iteration with $k = 1$ and hence no intrinsic parallelism.

In a more sophisticated partitioning approach, the matrices I_q are chosen such that sets of strongly coupled equations are taken together on one processor. However, this requires precise information on the IVP to be solved, and can only be analysed for specific classes of problems.

Finally, we remark that the iteration scheme (2.2) can be generalized by allowing the matrix M occurring in the residual function R_n, to depend on the partitioning index q. This enables us to adapt the iteration method and the corrector to the particular subsystem to be iterated. However, in this paper, we confine our considerations to constant M.

2.1. The iteration error

In order to analyse the behaviour of the iteration error $Y^{(j)} - Y$ we consider the error equation associated with (2.2) in the case where G_n is linear in Y, satisfying the relation

$$(2.3)\qquad G_n(V) - G_n(W) = (I\otimes J_n)\,[V - W],$$

with J_n the d-by-d Jacobian matrix of G_n (evaluated at t_n). Omitting in J_n the step index n, the inner-outer iteration method reduces to the recursion

$$(2.4)\qquad I_q\left[I - h^{\nu}Q(I\otimes J)I_q\right]\left[Y^{(j)} - Y^{(j-1)}\right] = -\,I_q\,PR_n(h, Y^{(j-1)}),$$

from which we deduce the iteration error equation

$$(2.5)\qquad I_q\left[I - h^{\nu}Q(I\otimes J)I_q\right]\left[Y^{(j)} - Y\right] = I_q\left[I - P + h^{\nu}P(M\otimes J) - h^{\nu}Q(I\otimes J)I_q\right]\left[Y^{(j-1)} - Y\right].$$

The combined effect of the iteration process for q = 1, 2, ... , k can be studied by considering the summed recursions given by

$$(2.6)\qquad \left[I - h^{\nu}S\right]\left[Y^{(j)} - Y\right] = \left[I - P - h^{\nu}(S - P(M\otimes J))\right]\left[Y^{(j-1)} - Y\right],\quad S := \sum_{q=1}^{k} I_qQ(I\otimes J)I_q$$

(recall that the summing the matrices I_q was assumed to yield the identity matrix). The matrix S can be expressed as

$$(2.7)\qquad S := \sum_{q=1}^{k} I_q\begin{pmatrix} Q_{11}J & \dots & Q_{1s}J \\ . & \dots & . \\ Q_{s1}J & \dots & Q_{ss}J \end{pmatrix} I_q,\quad Q := \begin{pmatrix} Q_{11} & \dots & Q_{1s} \\ . & \dots & . \\ Q_{s1} & \dots & Q_{ss} \end{pmatrix},$$

where the Q_{ij} are d-by-d matrices. In the cases of stage-vector-Jacobi iteration ($I_qe = e_q\otimes e$), point-Jacobi iteration ($I_qe = e_q$), and stage-value-Jacobi iteration ($I_qe = e\otimes e_q$), we respectively obtain

$$(2.8) \qquad S = \begin{pmatrix} Q_{11}J & \dots & O \\ . & \dots & . \\ O & \dots & Q_{ss}J \end{pmatrix}; \quad S = \begin{pmatrix} S_{11} & \dots & O \\ . & \dots & . \\ O & \dots & S_{ss} \end{pmatrix}; \quad S = \begin{pmatrix} S_{11} & \dots & S_{1s} \\ . & \dots & . \\ S_{s1} & \dots & S_{ss} \end{pmatrix},$$

where $S_{ij} := \mathrm{diag}\,(Q_{ij}J)$. From these representations it follows that stage-vector-Jacobi iteration does not split the Jacobian matrix, while the diagonal operation in the point-Jacobi and stage-value-Jacobi iteration methods will in general not preserve the complete Jacobian.

In the remainder of this paper, we restrict our considerations to point-Jacobi iteration and stage-value-Jacobi iteration *without* preconditioning ($P = I$).

3. Jacobi iteration versus functional iteration

In this section, we discuss various aspects of Jacobi iteration with

$$(3.1) \qquad P = I, \;\; Q = M \otimes I.$$

Assuming that the Jacobian of $G_n(Y^{(j)})$ at t_n is given by $I \otimes J_n$ (cf. (2.3)) and solving (2.2) for $Y^{(j)}$ by just one Newton iteration, we obtain

$$(3.2) \qquad I_q\big[I - h^\nu(M \otimes J_n)I_q\big]\,\big[Y^{(j)} - Y^{(j-1)}\big] = - I_q R_n(h, Y^{(j-1)}), \;\; q = 1, \dots, d; \;\; j = 1, \dots, m.$$

This equation shows that for point- and stage-value-Jacobi iteration methods only diagonal entries of the Jacobian matrix of the IVP enter into the iteration process, so that stiff systems can only be solved if the Jacobian J_n is sufficiently diagonally dominant. Hence, in practice, one should consider the methods using point- and stage-value-Jacobi iteration as *nonstiff* solvers. This immediately raises the question whether Jacobi iteration has any advantage over (explicit) functional iteration obtained for $P = I$ and $Q = O$. Let us first compare the computational costs of the two type of methods when implemented with some stepsize and iteration error strategy.

3.1. Computational costs. Denoting the total number of steps in the integration process by N and the number of steps where we need a new LU-decomposition of the matrix $I - I_q h^\nu(M \otimes J_n)I_q$ by θN, we conclude that the major costs of the stage-value-Jacobi iteration method are:

N	evaluations of the sd components of $Y^{(0)}$
mN	evaluations of the sd components of the residual function R_n
mN	estimates of the sd components of the iteration error
θN	evaluations of the d diagonal entries of J_n
$d\theta N$	LU-decompositions of s-by-s matrices of the form $I - I_q h^\nu(M \otimes J_n)I_q$
mdN	backward/forward substitutions by s-by-s matrices.

Here, m should be interpreted as the averaged number of iterations over all N steps. To the iteration costs listed above, we have to add the costs of

N	evaluations of the rd components of the function H_n defining the step point formula in (1.1)
N	estimates of the rd components of the truncation error associated with U_{n+1}

These costs have intrinsic parallelism of degree at least d, so that d processors can efficiently be employed.

Suppose that the evaluation of one (block)component of R_n and H_n, and the evaluation of the diagonal entries of J_n require F_R, F_H, and F_J floating-point operations (flops), respectively, and let us assume that F_R also contains the costs of $Y^{(0)}$ and iteration error costs, and that truncation error costs are included in F_H. Then, the total number of flops per processor per step required by functional iteration and stage-value-Jacobi iteration are given by $F_{FI} := msF_R + rF_H$ and $F_{SVJ} := msF_R + \theta F_J + 2\theta s^3/3 + 2ms^2 + rF_H$, respectively. Thus,

$$\frac{F_{SVJ}}{F_{FI}} = 1 + \frac{\theta F_J + 2\theta s^3/3 + 2ms^2}{msF_R + rF_H} < 1 + \frac{\theta F_J + 2\theta s^3/3 + 2ms^2}{msF_R}.$$

In general, $F_J < F_R$, so that we find

$$\frac{F_{SVJ}}{F_{FI}} < \frac{ms + \theta}{ms} + \frac{2s}{3m}\frac{\theta s + 3m}{F_R} \approx 1 + \frac{2s}{3m}\frac{\theta s + 3m}{F_R} .$$

This costs-increase factor changes per step and per processor because the value of F_R usually varies with t (e.g., in the case of Volterra equations) and with the components of R_n. It is larger as F_R is smaller. On the other hand, the run time per processor per step is largest for the processor to which the most expensive components of the residual function are assigned. Hence, the relevant costs-increase factor is bounded by $1 + s(1 + \theta s/3m)/\max\{F_R\}$. In most applications, this factor is only marginally larger than 1.

For example, using an s-stage Gauss-Legendre corrector and iterating until the order of the corrector is reached leads to $m = 2s-1$ iterations per step. Hence, stage-value-Jacobi iteration is about a factor $1 + s/\max\{F_R\}$ more expensive than functional iteration.

In the case of point-Jacobi iteration, we have similar costs, except for the LU-decompositions and backward/forward substitutions which are negligible because only scalarly implicit relations are involved. As a consequence, the main costs have parallelism of degree sd. We find

$$\frac{F_{PJ}}{F_{FI}} = 1 + \frac{\theta F_J}{msF_R + rF_H} < 1 + \frac{\theta}{ms},$$

so that point-Jacobi increases the computational costs only marginally.

Summarizing, we conclude that point-Jacobi and stage-value-Jacobi are generally not much more expensive than functional iteration.

3.2. The convergence factor. Next, we consider the convergence of the Jacobi method (3.2). The error equation corresponding to (3.2) reads

$$(3.3) \qquad Y^{(j)} - Y = Z\left[Y^{(j-1)} - Y\right], \; Z := h^{\nu}(I - h^{\nu}K\otimes J_D)^{-1}(M\otimes J - K\otimes J_D), \quad J_D := \mathrm{diag}\,(J),$$

where for functional iteration, point-Jacobi and stage-value-Jacobi we have $K = O$, $K = \mathrm{diag}\,(M)$ and $K = M$, respectively. We shall call Z the *iteration matrix* and its spectral radius $\rho(Z)$ the *convergence factor* of the iteration method. The expression (3.3) shows that we always have convergence (i.e., $\rho(Z) < 1$) if h is sufficiently small.

For functional iteration the iteration matrix reduces to

$$Z := h^{\nu}M\otimes J,$$

so that we have convergence factor

$$(3.4) \qquad \rho(Z) = h^{\nu}\rho(M)\rho(J).$$

For Jacobi iteration, it is convenient to factorize Z according to

$$(3.5) \qquad Z := Z_1Z_2, \quad Z_1 := (h^{\nu}K\otimes J_D)(I - h^{\nu}K\otimes J_D)^{-1}, \quad Z_2 := K^{-1}M\otimes J_D{}^{-1}J - I,$$

where K and J_D are assumed to be nonsingular. This representation shows that, unlike functional iteration, Jacobi iteration has a bounded iteration matrix Z for all h and J, provided that the entries of the 'diagonally scaled' Jacobian $J_D{}^{-1}J$ are bounded. Furthermore, the matrix Z_1 can be partitioned into a matrix with diagonal blocks $c_{ij}J_D$, whereas the blocks in the partitioning of Z_2 contains the full matrix $J_D{}^{-1}J$. Therefore, the matrix Z_2 will largely determine the convergence behaviour of the iteration process.

The convergence wil be faster as the magnitude (in some sense) of the iteration matrix $Z = Z_1Z_2$ is smaller. We shall estimate the magnitude of this matrix by the quantity $\rho(Z_1)\rho(Z_2)$. The following theorem presents an easy estimate for $\rho(Z_1)\rho(Z_2)$ and specifies a few cases where $\rho(Z_1)\rho(Z_2)$ provides an estimate for the convergence factor $\rho(Z)$. In this theorem, it is convenient to use the minimal value of the real parts of the eigenvalues of a matrix A. Denoting the spectrum of A by $\sigma(A)$, this quantity is defined by

$$(3.6) \qquad \mu(A) := \min\,\{\mathrm{Re}(\alpha):\ \alpha \in \sigma(A)\}.$$

Theorem 3.2. Let

$$(3.7)\qquad \sigma(J_D) \in \mathbb{R}^-, \ \ \sigma(K) \in \mathbb{C}^+, \ \ E(h) := \frac{h^\nu \rho(K)\rho(J_D)\,\rho(K^{-1}M \otimes J_D^{-1}J - I)}{\sqrt{1 + 2h^\nu \mu(K)\mu(-J_D) + h^{2\nu}\rho(K)^2\rho(J_D)^2}}\ .$$

Then the following assertions hold:

(a)	Arbitrary K, M and J_D	=>	$\rho(Z_1)\rho(Z_2) \le E(h)$.
(b)	$KM = MK, \ J_D = \delta I$	=>	$\rho(Z) \le \rho(Z_1)\rho(Z_2) \le E(h)$.
(c)	$K = M, \ J_D = \delta I$	=>	$\rho(Z) = \rho(Z_1)\rho(Z_2) \le E(h)$.
(d)	$K = M, \ \mathrm{Re}\,(\sigma(K)) = \mu(K), \ J_D = \delta I$	=>	$\rho(Z) = \rho(Z_1)\rho(Z_2) = E(h)$.
(e)	$K = \kappa I, \ J_D = \delta I$	=>	$\rho(Z) = \rho(Z_1)\rho(Z_2) = E(h)$.

Proof. Let κ and δ denote the eigenvalues of K and J_D, respectively. From the definition of Z_1 and Z_2 it then follows that

$$(3.8)\qquad \rho(Z_1)\,\rho(Z_2) \ = \max_{\kappa\delta} \frac{h^\nu|\kappa\delta|\,\rho(Z_2)}{|1 - h^\nu\kappa\delta|} = \max_{\kappa\delta} \frac{h^\nu|\kappa\delta|\,\rho(Z_2)}{\sqrt{1 + 2h^\nu \mathrm{Re}(-\kappa\delta) + h^{2\nu}|\kappa\delta|^2}}$$

$$\le \max_{\kappa\delta} \frac{h^\nu|\kappa\delta|\,\rho(Z_2)}{\sqrt{1 + 2h^\nu\mu(K)\mu(-J_D) + h^{2\nu}|\kappa\delta|^2}}\ .$$

Since the righthand side in this inequality is an increasing function of $|\kappa\delta|$, we obtain the result (a). The convergence factor $\rho(Z)$ is bounded by $\rho(Z_1)\rho(Z_2)$ if Z_1 and Z_2 commute, or equivalently, if the matrices $K\otimes J_D$ and $K^{-1}M\otimes J_D^{-1}J$ commute. This happens if both K and M, and J_D and J commute. The condition on J_D and J implies that the Jacobian matrix J has constant diagonal entries, to obtain the result (b). Thirdly, if also K = M, then Z becomes the direct product of the matrices Z_1 and Z_2, so that we have $\rho(Z) = \rho(Z_1)\rho(Z_2)$, leading to (c). The assertions (d) and (e) follow by observing that in these cases we have strict equality in (3.8). []

In the case of stage-value-Jacobi iteration (K = M), the estimate E(h) reduces to

$$(3.7')\qquad E(h) := \frac{h^\nu \rho(M)\,\rho(J_D)\,\rho(J_D^{-1}J - I)}{\sqrt{1 + 2h^\nu\mu(M)\mu(-J_D) + h^{2\nu}\rho(M)^2\rho(J_D)^2}},$$

showing that, independent of the particular corrector used, fast convergence can be expected when applied to IVPs possessing strongly diagonal dominant Jacobian matrices, i.e., $\rho(J_D^{-1}J - I) \ll 1$. Therefore, from now on, we concentrate on stage-value-Jacobi iteration. For future reference, we list in Table 3.1 the radius $\rho(M)$ and the minimal real part $\mu(M)$ of the spectrum of the matrices M of Gauss-Legendre correctors.

Table 3.1. Values of $\rho(M)$ and $\mu(M)$ for s-stage Gauss-Legendre correctors.

	s = 2	s = 3	s = 4	s = 5	s = 6
$\rho(M)$	0.289	0.216	0.166	0.133	0.115
$\mu(M)$	0.250	0.143	0.092	0.064	0.048

3.3. Transformation to constant diagonal entries in the Jacobian

In general, the Jacobian J will have variable diagonal entries, so that the condition $J_D = \delta I$ in Theorem 3.2 (b) - (e) is not satisfied and consequently the estimate E(h) is not necessarily an upper bound for the convergence factor $\rho(Z)$. In order to gain some apriori insight into the true convergence factors for problems with nonconstant diagonal entries in the Jacobian, we may try to transform the problem into a problem with constant diagonal entries in its Jacobian. If the integration method applied to the original and transformed problems show a comparable convergence behaviour of the iteration process, then the convergence factor corresponding to the transformed problem is indicative for the convergence factor corresponding to the original problem. We illustrate this for the IVP for ODEs. Let the ODE be given by $y'(t) = f(y(t))$, and define $z(t) = Ty(t)$ with T a constant nonsingular d-by-d matrix. In terms of z(t), we have the ODE $z'(t) = g(z(t)) := Tf(T^{-1}z(t))$ with Jacobian matrix TJT^{-1}, where $J = J(y)$ denotes the Jacobian of the original right hand side function f. Suppose that we can find a matrix T such that at $y = y(t_n)$ the matrix TJT^{-1} has constant diagonal entries δ. Then, instead of integrating the equation $y'(t) = f(y(t))$ from t_n to t_{n+1}, we can integrate the equation $z'(t) = g(z(t))$ over this interval, while satisfying the condition of constant diagonal entries. The iteration matrix defining the iteration process for the transformed problem is given by

$$(3.9) \qquad Z := Z_1Z_2, \quad Z_1 := (h^{\nu}K\otimes\delta I)\,(I - h^{\nu}K\otimes\delta I)^{-1}, \quad Z_2 := K^{-1}M\otimes\delta^{-1}TJT^{-1} - I.$$

A comparison of the iteration matrices defined by (3.5) and (3.8) reveals that they are rather similar indicating that we can expect comparable convergence behaviour. We shall call the iteration method with iteration matrix (3.9) the *transformed iteration method.*

Let us consider the case of triangular transformation matrices T. In order to construct such a transformation matrix T, we write $T = L + D$, where L is strictly lower triangular and D is diagonal. To obtain constant diagonal entries δ in TJT^{-1}, we have to satisfy the relation

$$(3.10) \qquad \mathrm{diag}((L + D)\,J\,(L + D)^{-1}) = \delta I.$$

Given the matrix L and δ, this equation presents a system of d equations for the d diagonal entries of D.

Theorem 3.3 presents an extremely simple transformation that can be used for deriving apriori estimates for the convergence factor in cases where the Jacobian contains at least one row with nonzero off-diagonal elements.

Theorem 3.3. Let J be an d-by-d matrix with entries a_{ij} and let T be the triangular matrix defined by

$$T := \begin{pmatrix} d_1 & 0 & 0 & 0 & \dots \\ 1 & d_2 & 0 & 0 & \dots \\ 1 & 0 & d_3 & 0 & \dots \\ 1 & 0 & 0 & d_4 & \dots \\ . & . & . & . & \dots \end{pmatrix}, \quad d_i := \frac{a_{1i}}{\delta - a_{ii}}, \quad \delta := \frac{\mathrm{Trace}(J)}{d}, \quad i = 2, \dots, d.$$

If d_1, a_{1i} and $\delta - a_{ii}$ do not vanish for $i = 2, \dots, d$, then $\mathrm{diag}\,(TJT^{-1}) = \delta I$.

Proof. Substitution of $L + D = T$ and

$$(L + D)^{-1} = T^{-1} = \begin{pmatrix} (d_1)^{-1} & 0 & 0 & \dots & 0 \\ -(d_1d_2)^{-1} & (d_2)^{-1} & 0 & \dots & 0 \\ -(d_1d_3)^{-1} & 0 & (d_3)^{-1} & \dots & 0 \\ . & . & . & \dots & . \end{pmatrix}$$

into (3.9) yields the following system for the diagonal entries d_i:

$$a_{11} - \sum_{j=2}^{n} a_{1j}\,(d_j)^{-1} = \delta; \quad a_{1i}(d_i)^{-1} + a_{ii} = \delta, \; i = 2, \dots, d.$$

Choosing $\delta = \mathrm{Trace}(J)/d$, this system is solved by $d_i = a_{1i}/(\delta - a_{ii})$, $i = 2, \dots, d$, leaving d_1 free. []

4. Numerical experiments

In this section, we report numerical comparisons of results obtained by functional iteration and by stage-value-Jacobi iteration for IVPs for first-order ODEs $y'(t) = f(t, y(t))$. In our experiments, we used the fourth-order Gauss-Legendre corrector, so that the residual function occurring in (3.2) is given by

$$R_n(h, Y) = Y - y_n e - h(M \otimes I)\, f(et_n + ch, Y), \quad M = \frac{1}{12}\begin{pmatrix} 3 & 3-2\sqrt{3} \\ 3+2\sqrt{3} & 3 \end{pmatrix}.$$

We used the simple 'last step value' predictor $Y^{(0)} = ey_n$. In order to 'tune' the arguments of f, we set $c = 0$ in the computation of $R_n(h, Y^{(0)})$, and $c = Me$ otherwise.

In particular, we check the relevance of the estimate E(h) defined by (3.7) as an indicator for convergence of the iteration method. For functional iteration and stage-value-Jacobi iteration the estimates E(h) are respectively given by

$$\text{(4.1)} \qquad E_{FI}(h) = 0.29h\rho(J), \quad E_{SVJ}(h) = \frac{0.29h\rho(J_D)\rho(J_D^{-1}J - I)}{\sqrt{1 + 0.5h\mu(-J_D) + 0.084h^2\rho(J_D)^2}}.$$

4.1. Effect of the constant-diagonal-transformation

Firstly, we compare the convergence of functional iteration and of stage-value-Jacobi iteration for the untransformed and transformed problem. Consider the linear problem

$$\text{(4.2)} \qquad \frac{dy(t)}{dt} = Jy(t) + v, \quad y(0) = \mathbf{0}, \quad J := \begin{pmatrix} -1 & 1 & 1 \\ 0 & -2 & 1 \\ 1 & 1 & -1/2 \end{pmatrix}, \quad v := \begin{pmatrix} 1 \\ -1 \\ 2 \end{pmatrix}, \quad 0 \le t \le T.$$

At $t = T = 5$, the solution is approximately given by $y(5) = (41.529764, 18.516263, 51.537861)^T$. The rapid increase of the solution values is caused by a positive eigenvalue of the Jacobian matrix J (they are approximately given by -2.19, -2 and +0.69). Since $\rho(J) \approx 2.2$, $\rho(J_D) = 2$ and $\rho(J_D^{-1}J - I) \approx 1.9$, the estimates E(h) for functional iteration and stage-value-Jacobi iteration are given by

$$\text{(4.3a)} \qquad E_{FI}(h) = 0.64h, \quad E_{SVJ}(h) = \frac{1.1h}{\sqrt{1 + 0.25h + 0.336h^2}}.$$

Next we consider the transformed version of (4.1). According to Theorem 3.3, we define the matrices

$$T := \begin{pmatrix} 1 & 0 & 0 \\ 1 & 6/5 & 0 \\ 1 & 0 & -3/2 \end{pmatrix}, \quad T^{-1} := \begin{pmatrix} 1 & 0 & 0 \\ -5/6 & 5/6 & 0 \\ 2/3 & 0 & -2/3 \end{pmatrix},$$

so that we can transform (4.2) to the constant-Jacobian-diagonal-form:

$$\text{(4.4)} \qquad \frac{dz(t)}{dt} = TJT^{-1}z(t) + Tv, \quad z(0) = \mathbf{0}, \quad TJT^{-1} = \begin{pmatrix} -7/6 & 5/6 & -4/6 \\ 49/30 & -7/6 & -22/15 \\ -11/12 & -5/12 & -7/6 \end{pmatrix}, \quad 0 \le t \le T.$$

We now have $\rho(J_D) = 7/6$ and $\rho(J_D^{-1}J - I) = \rho\big(-(6/7)TJT^{-1} - I\big) = \rho\big(-(6/7)J - I\big) \approx 1.59$. Denoting the estimate E(h) for transformed stage-value iteration by $E_{TSVJ}(h)$, we find

$$\text{(4.3b)} \qquad E_{TSVJ}(h) := \frac{0.54h}{\sqrt{1 + 0.58h + 0.11h^2}}.$$

We integrate (4.2) and (4.4) from $t = 0$ until $t = 5$ using stepsizes $h = T/N$ with $N = 1, \dots, 5$. In the case (4.4), the numerical solution y_N at $t = 5$ is obtained by the back transformation $y_N = T^{-1}z_N$. For the two-point Gauss-

Legendre corrector, Table 4.2 lists the estimates E(h) defined by (4.2) and (4.4), and the numbers of *correct significant decimal* digits Δ at the endpoint defined by (devision is meant componentwise)

$$\Delta := -\log_{10}\left(\left\| \frac{y(t_N) - y_N}{y(t_N)} \right\|_\infty \right).$$

These results show that direct and transformed stage-value-Jacobi iteration perform similarly, but for transformed stage-value-Jacobi iteration the estimate $E_{TSVJ}(h)$ is a much better predictor for the actual performance of the iteration process than the estimate $E_{SVJ}(h)$ corresponding to direct stage-value-Jacobi iteration. Furthermore, the convergence region of stage-value-Jacobi is considerably larger than that of functional iteration. However, if the functional iteration method does converge, then its *true* convergence factor seems to be smaller than that of stage-value-Jacobi.

Table 4.2. Correct significant decimal digits Δ for problem (4.1) and (4.3) at $t = T = 5$ obtained for the two-point Gauss-Legendre corrector (* indicates $\Delta < 0$).

Iteration mode	T/h	E(h)	m=2	m=3	m=4	m=5	...	m=10
Functional iteration	3	1.1	*	*	0.8	0.8	...	0.2
	4	.80	0.5	1.2	2.7	2.7	...	2.6
	5	.64	1.5	2.4	3.0	3.0	...	2.9
Direct	1	1.7	0.1	0.2	0.3	0.4	...	1.5
stage-value iteration	2	1.4	0.4	0.6	0.8	1.2	...	1.6
	3	1.2	0.6	0.9	1.4	2.0	...	2.1
	4	1.0	0.8	1.3	1.8	2.6	...	2.6
	5	.88	1.0	1.5	2.2	3.2	...	3.0
Transformed	1	1.04	*	0.1	0.1	0.3	...	1.4
stage-value iteration	2	.76	0.3	0.5	0.7	1.0	...	2.3
	3	.59	0.5	0.8	1.2	1.6	...	2.4
	4	.49	0.7	1.1	1.6	2.3	...	2.7
	5	.41	0.9	1.4	2.0	2.9	...	3.0

4.2. Widely spaced diagonal entries

Our next test problem is a system of 10 nonlinear equations:

$$\text{(4.5)} \qquad \frac{dy(t)}{dt} = A\,[y(t) - e\sin(t)] + e\cos(t), \quad A := \begin{pmatrix} -1 & y_2(t) & 0 & \dots & 0 & 0 & 0 \\ y_1(t) & -2 & y_3(t) & \dots & 0 & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & \dots & y_8(t) & -9 & y_{10}(t) \\ 0 & 0 & 0 & \dots & 0 & y_9(t) & -10 \end{pmatrix}, \quad 0 \le t \le T,$$

with exact solution $y(t) = e\sin(t)$. The problem is constructed such that the diagonal entries of its Jacobian are widely varying, so that the constant-diagonal condition occurring in Theorem 3.2 is far from being satisfied. Along the solution, the Jacobian of (4.5) is given by the matrix A, so that using Gerschgorin's disk theorem, we have for $\rho(J)$ and $\rho(J_D^{-1}J - I)$ the estimates $-10 + |\sin(t)|$ and $|\sin(t)|$, respectively. Hence, the diagonal dominance of the Jacobian depends on t, resulting in intervals of strong, weak or no diagonal dominancy. The estimates E(h) are given by

$$E_{FI}(h) = 3.19h, \quad E_{SVJ}(h) = \frac{2.9h}{\sqrt{1 + 0.5h + 8.4h^2}}.$$

Table 4.3 lists the number of correct decimal digits, defined by

$$\Delta := -\log_{10}\left(\| y_N - y(T) \|_\infty \right), \quad N := \frac{T}{h}.$$

These results clearly demonstrate the superior convergence behaviour of stage-value-Jacobi iteration.

Table 4.3. Correct decimal digits for problem (4.5) at t = T = 5 obtained by two-point Gauss-Legendre corrector (* indicates divergence).

Iteration mode	h	E(h)≤	m=1	m=2	m=3	m=4	m=5	m=6	m=7	m=8	m=9	m=10
Functional iteration	1/2	1.6	*	*	*	*			...			*
	1/4	0.80	*	2.1	1.5	4.5	2.3	1.9	2.3	2.1	2.7	4.7
	1/8	0.40	2.1	2.9	3.4	5.9	4.3	4.4	4.8	5.1	6.1	5.9
Stage-value-Jacobi	1	0.92	0.6	1.0	1.6	2.0			...			2.0
	1/2	0.79	1.1	2.5	3.1	4.1			...			4.1
	1/4	0.56	2.8	3.6	4.7				...			4.7
	1/8	0.33	3.0	4.3	6.1	5.8	5.9		...			5.9

4.3. Reaction-diffusion equations

In order to see the effect of stage-value-Jacobi iteration in the case of a large system, we consider the two-dimensional reaction-diffusion equation

$$\text{(4.6a)} \qquad \frac{\partial u(t,x_1,x_2)}{\partial t} = \varepsilon\Delta u(t,x_1,x_2) - f(u(t,x_1,x_2)),$$

defined on the unit square. Here ε is a small parameter and Δ denotes the Laplacian in the spatial variables x_1 and x_2. We selected a problem from combustion theory for which f(u) is defined as

$$\text{(4.6b)} \qquad f(u) := D(1 + a - u)\exp\Big(-\frac{\delta}{u}\Big), \quad D := \frac{R\exp(\delta)}{a\delta}.$$

Details about this model can be found in [12]. The temperature $u(t, x_1, x_2)$ is subject to the initial and boundary conditions

$$\text{(4.6c)} \qquad u(0, x_1, x_2) = 1, \quad \frac{\partial u}{\partial n} = 0 \ \text{ at } x_1 = 0,\ x_2 = 0, \qquad u = 1 \ \text{ at } x_1 = 1,\ x_2 = 1.$$

Semidiscretization of (4.6a) on a uniform grid of width Δx, using symmetric second-order differences and incorporating the boundary conditions leads to a system of ODEs

$$\text{(4.6')} \qquad \frac{dy(t)}{dt} = \varepsilon(\Delta x)^{-2}Ay(t) - f(y(t)),$$

where f(y) has to be understood componentwise. For this problem we have

$$J = \varepsilon(\Delta x)^{-2}A - \mathrm{diag}\big(\partial f(y(t))/\partial y\big) \ \text{ and } \ J_D = -\,\mathrm{diag}\big(4\varepsilon(\Delta x)^{-2}e + \partial f(y(t))/\partial y\big).$$

In our test, we selected the following parameter values: $R = 5$, $\delta = 10$, $a = 1$ (see also [1]). Furthermore, ε was set to 10^{-3} and $\Delta x = 1/40$, resulting in a set of 1600 ODEs. The effect of this parameter choice is that the solution u increases from $u = 1$ (at $t = 0$) to the 'steady state' $u \approx 2$ at $t = 0.5$, the endpoint of the integration interval.

The main difficulty in this problem is caused by the reaction term which changes sign in the interval of integration: $\partial f(u)/\partial u = D\exp(-\delta/u)[(1 + a - u)\delta/u^2 - 1]$ is positive until u reaches the value $u \approx 1.71$ (the so-called 'ignition' point where a reaction front is formed running to the outer Dirichlet boundaries). For components having a value >1.71, $\partial f/\partial u$ is negative, ending at $\partial f/\partial u \approx -74$ for u-values close to the steady state. As a consequence of this behaviour, the elements of the matrix J_D are small in some parts of the integration interval, resulting in large values of the factor $\rho(J_D^{-1}J - I)$. Once the ignition point has been reached, $\partial f/\partial u$ becomes negative, the diagonal dominance of the Jacobian is re-established and $\rho(J_D^{-1}J - I)$ quickly decreases; at the end of the integration interval we have $\rho(J_D^{-1}J - I) \approx 0.02$. Hence, for this problem the estimate E(h) is only relevant in part of the integration interval (we remark that the assumption $\sigma(J_D) \in \mathbb{R}^-$ of Theorem 3.2 is even violated for some t-values). Nontheless, we have applied the algorithms to this problem, particularly because reaction-diffusion equations have great practical relevance. The results of this test are collected in Table 4.4.

Table 4.4. Correct decimal digits for problem (4.6') at $t = T = 0.5$ obtained by two-point Gauss-Legendre corrector (* indicates divergence).

Iteration mode	h	m=1	m=2	m=3	m=4	m=5	...	m=10
Functional iteration	1/10	*	*	*	*		...	*
	1/20	0.5	-0.1	0.8	1.1	1.2	...	1.3
	1/40	1.9	3.9	3.7	5.1		...	5.1
	1/80	3.6	4.6	5.3	6.6		...	6.4
Stage-value-Jacobi	1/10	*	0.0	0.0	*	0.2	...	*
	1/20	2.6	4.1	3.5			...	3.6
	1/40	4.3	5.2				...	5.1
	1/80	5.4	6.4				...	6.4

We see that stage-value-Jacobi shows a much better convergence behaviour than functional iteration: 2 or 3 iterations are sufficient (for $h \le 1/20$), whereas functional iteration needs at least 4 iterations. Hence, in spite of the aforementioned deficiencies of the stage-value-Jacobi method for this problem, it seems to possess a rather wide applicability.

4.4. Mildly stiff problems

Finally, we show that stage-value-Jacobi iteration can even be applied to mildly stiff problems. Consider a test problem proposed by Kaps [13]:

$$\frac{dy_1(t)}{dt} = -(2+\varepsilon^{-1})\, y_1(t) + \varepsilon^{-1}(y_2(t))^2,$$

$$(4.8) \qquad\qquad y_1(0) = y_2(0) = 1, \quad 0 \le t \le 1,$$

$$\frac{dy_2(t)}{dt} = y_1(t) - y_2(t)\,(1 + y_2(t)),$$

with exact solution $y_1 = \exp(-2t)$ and $y_2 = \exp(-t)$ for all values of the parameter ε. For this problem we have

$$J = \begin{pmatrix} -(2+\varepsilon^{-1}) & 2\varepsilon^{-1}y_2 \\ 1 & -(1+2y_2) \end{pmatrix}, \quad J_D = \begin{pmatrix} -(2+\varepsilon^{-1}) & 0 \\ 0 & -(1+2y_2) \end{pmatrix}.$$

We integrate this problem using the two-point Gauss-Legendre corrector. For small ε we have $\rho(J) \approx \varepsilon^{-1}$, $\rho(J_D) \approx \varepsilon^{-1}$, and $\rho(J_D^{-1}J - I) \approx (2y_2/(1+2y_2))^{1/2}$, leading to

$$(4.9) \qquad E_{FP}(h) = 0.29\,(h/\varepsilon), \quad E_{SVJ}(h) = \frac{0.29(h/\varepsilon)(2y_2/(1+2y_2))^{1/2}}{\sqrt{1 + 0.5h(1+2y_2) + 0.084(h/\varepsilon)^2}}\,.$$

For $\varepsilon = .01$, Table 4.5 lists the numbers of *correct decimal* digits (in absolute sense) for various values of the stepsize h. As in the preceding example, the convergence region of stage-value-Jacobi is considerably larger than that of functional iteration (assuming that the numerical approximation to y_2 varies from 1 until $\exp(-1) \approx 0.37$, the interval for $E_{SVJ}(h)$ is easily calculated and given in the table). Furthermore, although the Jacobian of this problem is only weakly diagonally dominant (i.e., $\rho(J_D^{-1}J - I)$ is not much smaller than 1), the rate of convergence of the stage-value-Jacobi method appears to be substantially larger than that of functional iteration.

Table 4.5. Correct decimal digits for problem (4.8) at t = 1 obtained by two-point Gauss-Legendre corrector for $\varepsilon = .01$ (* indicates $\Delta < 0$).

	h	E(h)	m=1	m=2	m=3	m=4	...	m=10
Functional iteration	≥ 1/20	≥ 1.02	*	*	*	*	...	*
	1/40	0.73	*	1.9	4.1	7.3	...	7.0
Stage-value-Jacobi	1/2	[0.65, 0.82]	*	*	*	1.8	...	1.9
	1/5	[0.64, 0.80]	*	1.9	0.8	3.3	...	3.2
	1/10	[0.61, 0.77]	0.0	3.2	2.4	4.9	...	4.6
	1/20	[0.53, 0.66]	1.5	3.9	3.8	6.1	...	5.9
	1/40	[0.38, 0.47]	2.3	4.7	5.0	7.3	...	7.1

References

[1] Adjerid, S. & Flaherty, J.E. (1988): A local refinement finite element method for two dimensional parabolic systems, SIAM J. Sci. Stat. Comput. 9, 792-811.

[2] Burrage, K. (1991): The error behaviour of a general class of predictor-corrector methods, Appl. Numer. Math. 8, 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] Burrage, K. & Butcher, J.C. (1980): Nonlinear stability of a general class of differential equations methods, BIT 20, 185-203.

[6] Collatz, L. (1950): Über die Konvergentzkriterien bei Iterationsverfahren fur lineare Gleichungssysteme, Math. Z. 53, 149-61.

[7] Crisci, M.R., Houwen, P.J. van der, Russo, E. & Vecchio, A. (1992): Stability of parallel Volterra-Runge-Kutta methods, to appear in Appl. Numer. Math..

[8] Houwen, P.J. van der (1993): Preconditioning in implicit initial value problem methods on parallel computers, to appear in Advances in Comp. Math...

[9] 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.

[10] Houwen, P.J. van der, & Sommeijer, B.P. (1991): Iterated Runge-Kutta methods on parallel computers, SIAM J. Sci. Stat. Comput. 12, 1000-1028.

[11] 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.

[12] Kapila, A.K. (1983): Asymptotic treatment of chemically reacting systems, Pitman Advanced Publ. Company.

[13] Kaps, P. (1981): Rosenbrock-type methods, in: Numerical Methods for Stiff Initial Value Problems, G. Dahlquist and R. Jeltsch, eds., Bericht nr. 9, Inst. für Geometrie und Praktische Mathematik der RWTH Aachen, Aachen, Germany.

[14] Nguyen huu Cong (1993): Note on the performance of direct and indirect Runge-Kutta-Nyström methods, to appear in J. Comp. Appl. Math..

[15] 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.

[16] Sommeijer, B.P. (1993): Explicit, high-order Runge-Kutta-Nyström methods for parallel computers, submitted for publication.