Stability Control for Approximate Implicit Time Stepping Schemes with Minimum Residual Iterations

Mikhail A. Botchev* Gerard L. G. Sleijpen†

Henk A. van der Vorst†

December 1, 1998

Abstract

Implicit schemes for the integration of ODE's are popular when stability is more of concern than accuracy, for instance for the computation of a steady state solution. However, in particular for very large systems the solution of the involved linear systems may be very expensive. When these systems are solved iteratively to a certain tolerance, it is often not known which tolerance has to be taken. We propose a different strategy, where the number of iterations is fixed, but the step size is controlled with respect to stability. Numerical tests show the effectiveness of this approach in comparison with an implicit scheme that iterates to a certain tolerance.

Keywords: Time Stepping schemes; Stability step size control; Minimum Residual Iterations

1 Introduction

Implicit schemes for the integration of ODE's are popular when stability is more of concern than accuracy, for instance for the computation of a steady state solution. However, in particular for very large systems the solution of the involved linearized systems may be very expensive. In this paper we study the solution of these linear systems by a moderate number of the minimum residual iterations [14, 1]. Of course, this puts limits to the step size since these approximate schemes may be viewed as explicit schemes and

*CWI, P.O.Box 94079, 1090 GB Amsterdam, the Netherlands. E-mail: botchev@cwi.nl. This work has been done while this author was with Mathematical Institute, Utrecht University.

†Mathematical Institute, Utrecht University, P.O.Box 80.010, 3508 TA Utrecht, the Netherlands. E-mail: [sleijpen, vorst]@math.uu.nl

these are never unconditionally stable. However, in our approach we obtain a different explicit scheme for each time step. As we will see, a succession of these different schemes will be much more stable than each of them separately. This is mainly because the minimum residual approximation adjusts itself to a given right-hand side. It turns out that even a modest degree of approximation allows rather large time steps and we propose a simple mechanism for the control of the step size with respect to stability.

The usage of a few minimum residual iterations is attractive, since this involves only matrix-vector operations. Specially on parallel computers this may be an advantage because matrix-vector operations are usually easy to parallelize. If one uses k iterative steps, then the resulting approximate method may be viewed as an explicit integration scheme and such schemes have been studied heavily. Most often, these schemes are studied with fixed coefficients of Chebyshev approximations [21, 12, 23, 17, 16, 11]. Performance of these methods depends on a priori knowledge of a region containing the spectrum of the Jacobian. Another approach, that is based on Krylov subspace information, is to use an approximation for the vector $e^{\Delta t A}\boldsymbol{y}^n$ (where A is the Jacobian), for accurate time integration of stiff systems [8, 9].

The minimum residual iterations lead to different coefficients for each time step. The usage of minimum residual iterative schemes, e.g. GMRES, for the approximate solution of the involved linear systems is not new either, but the resulting schemes have been studied from an accuracy point of view, that is the main focus is on strategies to obtain bounded residuals and higher order consistency [6, 5, 4, 15]. Typically, one solves the linear systems to a certain tolerance, but it is often not known which tolerance has to be taken.

We study the usage of a few minimum residual iterations from the point of view of stability. A nice aspect of minimum residual iterations is that they construct implicitly an integration polynomial of which the coefficients are adjusted to the specific right-hand side. If after some time steps components in the solution in higher frequencies have not sufficiently damped out, then they are present in the right-hand side. As soon as these components are too large, they are automatically damped out by the minimum residual polynomial, provided that the time step is not too large with respect to the number of iterations. We propose an easy strategy to control the size of the time step. This strategy is based on information that we obtain from the iterative process itself. As we will see from our numerical experiments, the proposed stability control can be quite efficient as compared to iterating to a certain tolerance.

The stability control is useful in situations where accuracy is not the restrictive factor. However, our approximated implicit schemes can be used for high accurate computations in an ODE code as well. Our findings on this respect will be reported elsewhere.

The remainder of our paper has been organized as follows. Our basic approach has been formulated in Section 2. In the third Section we present the stability analysis and discuss how the stability can be monitored. We will do this for two different schemes. Implementation issues are discussed in Section 4. Numerical experiments are presented in the fifth Section, and we have listed some conclusions at the end.

2 Minimum Residual Approximated Implicit schemes

The general idea behind our approach is as follows. Our aim is to obtain a cheap alternative for an implicit integration scheme, without loosing too much in terms of stability. We start with a given implicit scheme of order p in time, and we approximate the solution of the associated implicit system with a few steps of a minimum residual iterative solver, say GMRES. This leads to a new scheme, and in order to let this new scheme have at least the same order of consistency, we need to start GMRES with the result of an explicit scheme of order p. This is different from the popular predictor–corrector approach, where the corrector is solved by successive substitution (this Richardson type of iteration gives an integration scheme with limited stability properties, comparable to the stability of the predictor).
We will work out the idea in more detail for the simple Euler Backward (EB) scheme for the system of ODE's

$$\frac{d\boldsymbol{y}}{dt} = \boldsymbol{f}(t,\boldsymbol{y})\,, \qquad \boldsymbol{y}\big|_{t=0} = \boldsymbol{y}^0 \in \mathbb{R}^N. \tag{1}$$

EB is given by:

$$\boldsymbol{y}^{n+1}_{\text{impl}} = \boldsymbol{y}^n + \Delta t \boldsymbol{f}(t_{n+1}, \boldsymbol{y}^{n+1}_{\text{impl}}). \tag{2}$$

In order to solve this equation for $\boldsymbol{y}^{n+1}_{\text{impl}}$, we take Euler Forward (EF)

$$\boldsymbol{y}^{n+1}_0 = \boldsymbol{y}^n + \Delta t \boldsymbol{f}(t_n, \boldsymbol{y}^n), \tag{3}$$

as an initial guess, and we write

$$\boldsymbol{y}^{n+1}_{\text{impl}} = \boldsymbol{y}^{n+1}_0 + \Delta\boldsymbol{y}. \tag{4}$$

The vector $\Delta\boldsymbol{y}$ satisfies

$$\boldsymbol{y}^{n+1}_0 + \Delta\boldsymbol{y} = \boldsymbol{y}^n + \Delta t \boldsymbol{f}(t_{n+1}, \boldsymbol{y}^{n+1}_0 + \Delta\boldsymbol{y}). \tag{5}$$

This leads to

$$\begin{array}{rcl} \Delta\boldsymbol{y} & = & \boldsymbol{y}^n + \Delta t \boldsymbol{f}(t_{n+1}, \boldsymbol{y}^{n+1}_0 + \Delta\boldsymbol{y}) - \boldsymbol{y}^{n+1}_0 \\ & \approx & \boldsymbol{y}^n + \Delta t \left\{ \boldsymbol{f}(t_{n+1}, \boldsymbol{y}^{n+1}_0) + A\Delta\boldsymbol{y} \right\} - \boldsymbol{y}^{n+1}_0, \end{array}$$

where A denotes the Jacobian, evaluated in $(t_{n+1}, \boldsymbol{y}_0^{n+1})$. In the predictor–corrector approach one ignores the $\Delta\boldsymbol{y}$ in the right-hand side, and one repeats the preceding steps, which leads to an iterative application of (4):

$$\boldsymbol{y}_{(j)}^{n+1} = \boldsymbol{y}_{(j-1)}^{n+1} + \Delta\boldsymbol{y}_{(j-1)}. \tag{6}$$

In our approach, we take for $\Delta\boldsymbol{y}$ the approximate solution from

$$(I - \Delta t A)\boldsymbol{x} = \boldsymbol{r}^n \equiv \boldsymbol{y}^n - \boldsymbol{y}_0^{n+1} + \Delta t \boldsymbol{f}(t_{n+1}, \boldsymbol{y}_0^{n+1}). \tag{7}$$

Note that $\boldsymbol{r}^n$ can be interpreted as the residual for (2) that we get when $\boldsymbol{y}_0^{n+1}$ is inserted.

We solve (7) by k steps of GMRES, with starting solution $\boldsymbol{x}_0 = 0$, so that the approximated solution $\boldsymbol{x}_k$ can be represented as

$$(\Delta\boldsymbol{y} :=) \quad \boldsymbol{x}_k = P_{k-1}^{(n)}(I - \Delta t A)\boldsymbol{r}^n,$$

where $P_{k-1}^{(n)}$ is the so-called iteration polynomial of degree $k-1$, associated with GMRES (in the case that A is symmetric, we take the MINRES method instead of GMRES). The number of iterations k is kept fixed. The superindex $\cdot^{(n)}$ has been included in order to indicate that we usually get a different polynomial for each time step. We will refer to the resulting scheme as an *Minimum Residual Approximated Implicit (MRAI)* scheme. Obviously, we can interpret the combination of (7) with minimum residual iterations as an inexact Newton method for (2).

We leave it to the reader to verify that the order of the local error is the same as for EB or EF, irrespective of the number of minimum residual iterations used for the approximation of $\boldsymbol{x}$. For more general implicit schemes, one has to take care that the starting vector is constructed with an explicit scheme of at least the same consistency order.

3 Stability of the MRAI scheme

3.1 Euler Backward based MRAI

The MRAI scheme of the previous section can be represented by

$$\boldsymbol{y}^{n+1} = \boldsymbol{y}_0^{n+1} + P_{k-1}^{(n)}(I - \Delta t A)(\boldsymbol{y}^n - \boldsymbol{y}_0^{n+1} + \Delta t \boldsymbol{f}(t_{n+1}, \boldsymbol{y}_0^{n+1})) \tag{8}$$

with $\boldsymbol{y}_0^{n+1} = \boldsymbol{y}^n + \Delta t \boldsymbol{f}(t_n, \boldsymbol{y}^n)$.

This explicit scheme is difficult to analyze, because the polynomial $P_{k-1}^{(n)}$ depends on spectral properties of the matrix as well as on the vector on which it is acting. This means that we can not simply apply the MRAI scheme to a linear homogeneous system and consider its effect in each eigenvector direction independently.

For the stability we look at the recursions obtained when $\boldsymbol{f}(t, \boldsymbol{y}) = A\boldsymbol{y}$, that is we consider, as is usual, the behavior for the linear homogeneous case. The MRAI scheme (8) then takes the form

$$\boldsymbol{y}^{n+1} = (I + \Delta t A)\boldsymbol{y}^n + P_{k-1}^{(n)}(I - \Delta t A)\boldsymbol{r}^n, \tag{9}$$

$$\boldsymbol{r}^n = (\Delta t A)^2 \boldsymbol{y}^n. \tag{10}$$

After multiplying the right-hand side of (9) by $(I - \Delta t A)^{-1}(I - \Delta t A)$, we obtain

$$\boldsymbol{y}^{n+1} = (I - \Delta t A)^{-1} \left(I - \left[I - (I - \Delta t A) P_{k-1}^{(n)} (I - \Delta t A) \right] (\Delta t A)^2 \right) \boldsymbol{y}^n. \tag{11}$$

The polynomial $[I - (I - \Delta t A) P_{k-1}^{(n)}(I - \Delta t A)]$ is just the k-degree minimum residual polynomial $R_k^{(n)}(I - \Delta t A)$, that is the polynomial that describes how the initial residual $\boldsymbol{r}^n \equiv \boldsymbol{r}_0^n$ is reduced to $\boldsymbol{r}_k^n$, after k minimum residual iterations.

Assuming that $A \in \mathbb{R}^{N \times N}$ can be diagonalized as $A = S \Lambda S^{-1}$, where Λ is a diagonal matrix with real eigenvalues λ_j on its main diagonal, we arrive, with $\boldsymbol{z}^n \equiv S^{-1}\boldsymbol{y}^n$, at

$$\boldsymbol{z}^{n+1} = (I - \Delta t \Lambda)^{-1} \left(I - R_k^{(n)}(I - \Delta t \Lambda)(\Delta t \Lambda)^2 \right) \boldsymbol{z}^n, \tag{12}$$

or for the j-th component $z^n = z_j^n$ of $\boldsymbol{z}^n$, and the j-th eigenvalue $\mu = \Delta t \lambda_j$:

$$z^{n+1} = (1 - \mu)^{-1} \left(1 - R_k^{(n)}(1 - \mu)\mu^2 \right) z^n. \tag{13}$$

We have written the above expressions for the residual polynomial $R_k^{(n)}$, rather than for $P_{k-1}^{(n)}$, since $R_k^{(n)}$ satisfies an optimality property: $\|R_k^{(n)}(I - \Delta t A)\boldsymbol{r}^n\|_2$ is minimal over all polynomials R of degree k, satisfying $R(0) = 1$. Note that *exactly* the same recursions as (12) and (13) describe how the initial residual $S^{-1}\boldsymbol{r}^{n+1}$ is related to its predecessor $S^{-1}\boldsymbol{r}^n$.

From experiences with GMRES, it is well-known that this polynomial attempts to reduce the largest components of $S^{-1}\boldsymbol{r}^n$, by putting a root close to the corresponding eigenvalue. Therefore, if the scheme leads to a large component in the vector $S^{-1}\boldsymbol{r}^n$, because of instability, then this component will be damped and in the next time step a new polynomial is automatically constructed that tries to reduce other new dominating components. The heuristic argument behind our stability control strategy is that we assume that in average, over a number of successive time steps, no component of the vector $S^{-1}\boldsymbol{r}^n$ dominates, or in other words, that all its components are more or less equal in absolute value.

The j-th component of $S^{-1}\boldsymbol{r}^n$ is multiplied in step n by a factor to $R_k^{(n)}(1-\mu)\mu^2$, and following the above heuristics, it is likely that the polynomial

$$Q(\mu) \equiv \mu^2 R_k^{(n)}(1-\mu)$$

has almost the minimax property over the interval $[\Delta t\lambda_N, \Delta t\lambda_1]$ (we assume that the eigenvalues have been ordered as $\lambda_N \leqslant \ldots \leqslant \lambda_1$). Note that this leads to a weaker condition than the requirement that the reduction polynomial is small over the entire spectrum in *each* time step. The latter requirement is made for schemes based upon Chebyshev polynomial approximations.

We will further assume that $\lambda_1 < 0$. The stability condition now reads

$$\left|\frac{1-R_k^{(n)}(1-\mu)\mu^2}{1-\mu}\right| \leqslant 1, \qquad \text{or}$$

$$\left|\frac{1-Q(\mu)}{1-\mu}\right| \leqslant 1. \tag{14}$$

The last condition in (14) is equivalent to

$$\mu \leqslant Q(\mu) \leqslant 2-\mu. \tag{15}$$

Note that $Q(1) = R_k^{(n)}(0) = 1$, and 0 is a root of Q of multiplicity 2. The other k roots η_i satisfy $R_k^{(n)}(1-\eta_i) = 0$. Suppose that $\eta_k \leqslant \ldots \leqslant \eta_1$. Since $\mu < 0$ over the interval of interest, we have that η_i, in particular $\eta_1 < 0$.
In Figure 1 we have plotted the situation when Q has the exact minimax property. With the estimate

$$|Q(\mu)| = \mu^2 \left|\prod_{i=1}^{k} \frac{\mu-\eta_i}{1-\eta_i}\right| \leqslant \mu^2 \frac{\mu-\eta_1}{1-\eta_1}, \qquad \mu \in [\eta_1\,;\,0), \tag{16}$$

it is possible to show (see [2] for a detailed proof) that (15) holds when

$$-7 \leqslant \eta_1 \qquad (<0). \tag{17}$$

In fact, condition (17) guarantees that the rightmost local maximum of $Q(\mu)$ (cf. Figure 1) is below the line $2-\mu$. The other local extrema should be between μ and $2-\mu$. In other words, condition (17) works for polynomials $Q(\mu)$ which may only very roughly resemble the minimax polynomial. To see whether our conclusions are confirmed in practice, in Figure 2 we have plotted $S^{-1}\boldsymbol{r}^n$ [1] and $Q(\mu)$ observed during numerical integration of the model problem

$$\begin{gathered} \boldsymbol{y}' = A\boldsymbol{y}, \quad \boldsymbol{y}^0 = (1,\ldots,1)^T \in \mathbb{R}^{500}, \\ A = \mathrm{Diag}(-1 : 0.002 : -0.01). \end{gathered} \tag{18}$$

[1] Note that $S = I$ in our example.

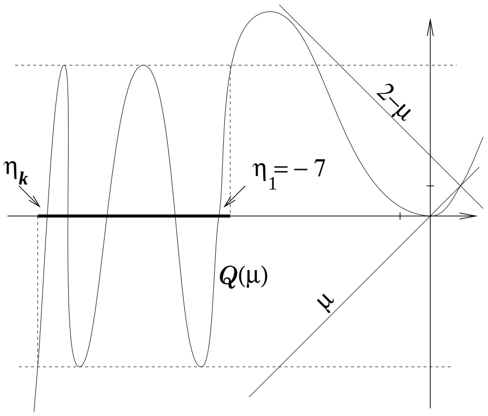


Figure 1: Stability governed by the roots of $Q(\mu)$ (cf. (15),(17))

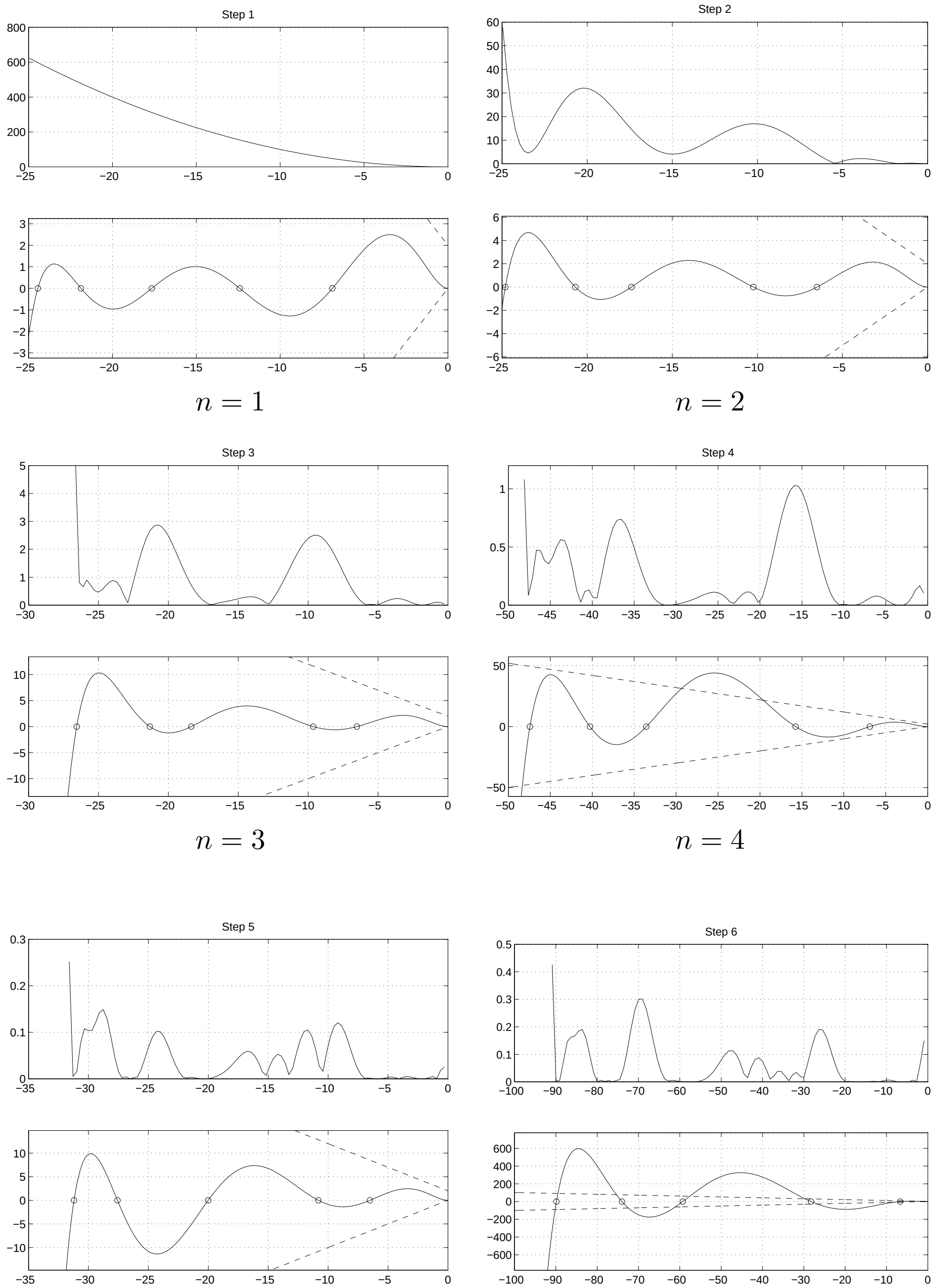
The fact that matrix A is diagonal does not affect behavior of the minimum residual iterations. Here we used the described MRAI scheme based on EB with $k = 5$ steps of GMRES and the step size chosen to have $-7 \leqslant \eta_1 \leqslant -6.5$. It turns out that the condition (15) is true for most time steps; each time step where for some μ's the condition (15) is not fulfilled is followed by one or two "safe" steps. As we see, our assumptions leaded to realistic stability control.

The same conclusions would be valid for other initial values $\boldsymbol{y}^0$ and for more "stiff" situations, for example, for A with eigenvalues evenly distributed in $[-1, -0.00001]$.

The roots η_i of $R_k^{(n)}(1-\mu)$ are easily obtained from the minimum residual iterative process: they are the so-called harmonic Ritz values [13]. The rightmost root η_1 indicates how important the spectral information is for eigenvalues closest to the origin, and we simply have to check that this stays above -7. If $\eta_1 < -7$, then we have to decrease the step size (see Section 4).

Condition (17) is necessary and sufficient for stability when $k = 1$. Assuming that $Q(\mu)$ has the minimax property, for $k > 1$ we can easily determine the limit values for η_1 numerically (see [2] for specific values). However, these limit values may be too optimistic in practice (as we just saw in the presented experiment, the polynomial $Q(\mu)$ is not precisely of the minimax type). Therefore, for the safety reasons, we will use condition (17) for $k > 1$ as well.

Another simple bound for the time step can be derived under the assumption that $R_k^{(n)}(1-\mu)$ resembles in average the k-th degree Chebyshev polynomial T_k shifted to a segment containing all μ's, for instance $[-\Delta t\lambda_N\,;\,0]$. This polynomial T_k is also scaled to be 1 at 0 (recall that $R_k^{(n)}(0) = 1$). The largest root of R_k is $1 - \eta_k$. Since we know η_k from the minimum residual



$n = 1$ $n = 2$

$n = 3$ $n = 4$

$n = 5$ $n = 6$

Figure 2: The absolute values of components of $S^{-1}\boldsymbol{r}^n$ plotted against the μ values and polynomials $Q(\mu)$, observed within first 6 successive time steps. The lines μ and $2 - \mu$ are dashed.

iterations, we can approximate μ_N using the relation

$$\Delta t\lambda_N = \mu_N \approx \frac{2\eta_k}{1+\cos(0.5\pi/k)}.$$

Of course, this estimate may not be accurate. Nevertheless, it gives a good impression of the largest *effective* eigenvalue that restricts the step size. Moreover, the stability condition (14) is satisfied when $|R_k(1-\mu)|\,|\mu| \leqslant 1$. Then, from the estimate $|R_k(1-\mu)| \leqslant 2\left(\frac{\sqrt{1-\mu_N}-1}{\sqrt{1-\mu_N}+1}\right)^k$, we obtain

$$2\left(\frac{\sqrt{1-\mu_N}-1}{\sqrt{1-\mu_N}+1}\right)^k |\mu_N| \leqslant 1. \tag{19}$$

The last inequality can be solved numerically, which gives an upper bound $\mathcal{C}$ for $|\Delta t\lambda_N|$, and, hence, also for Δt:

$$\Delta t|\lambda_N| = |\mu_N| \leqslant \mathcal{C} \quad \text{so that} \quad \Delta t \leqslant \mathcal{C}/|\lambda_N|. \tag{20}$$

It should be remarked that the estimate (19) is typically stronger than (14), and in practice it should be relaxed, especially for small k ($k \lesssim 3$). For instance, the constant 2 is often too crude in (19), and does not reflect realistically observed values.

3.2 An example of a higher order MRAI scheme

Of course, the ideas outlined in the previous section may also be applied to other implicit schemes. In this section we discuss briefly the situation for the implicit midpoint rule

$$\boldsymbol{y}^{n+1}_{\text{impl}} = \boldsymbol{y}^n + \Delta t\boldsymbol{f}(t_{n+1/2}, \frac{1}{2}(\boldsymbol{y}^n + \boldsymbol{y}^{n+1}_{\text{impl}})), \tag{21}$$

where $t_{n+1/2} = t_n + \frac{\Delta t}{2}$. This scheme is the second order accurate. We consider as before the linearized form:

$$(I - \frac{\Delta t}{2}A)\boldsymbol{y}^{n+1}_{\text{impl}} = (I + \frac{\Delta t}{2}A)\boldsymbol{y}^n + \Delta t\boldsymbol{g}^{n+1/2}, \tag{22}$$

with

$$A = \left[\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{y}}\right](t_{n+1/2}, \boldsymbol{y}^n), \quad \boldsymbol{g}^{n+1/2} = \boldsymbol{f}(t_{n+1/2}, \boldsymbol{y}^n) - A\boldsymbol{y}^n.$$

For the starting vector for the minimum residual iterations, we select the second order explicit scheme

$$\boldsymbol{y}^{n+1}_0 = \boldsymbol{y}^n + (I + \frac{\Delta t}{2}A)\Delta t\boldsymbol{f}(t_{n+1/2}, \boldsymbol{y}^n). \tag{23}$$

The initial residual delivered by (23) is $\boldsymbol{r}^n = \frac{(\Delta t)^3}{4} A^2 \boldsymbol{f}(t_{n+1/2}, \boldsymbol{y}^n)$, or, and for linear homogeneous system $\boldsymbol{y}' = A(t)\boldsymbol{y}$, we obtain

$$\boldsymbol{r}^n = \frac{(\Delta t)^3}{4} A^3 \boldsymbol{y}^n. \tag{24}$$

In a similar way as for the EB-based MRAI, we obtain after some manipulations, with $\tilde{A} = I - \frac{\Delta t}{2} A$:

$$\boldsymbol{y}^{n+1} = \tilde{A}^{-1} \left(I + \frac{\Delta t}{2} A - 2(\frac{\Delta t}{2} A)^3 R_k^{(n)}(\tilde{A}) \right) \boldsymbol{y}^n$$

Assuming that A can be diagonalized, and has real negative eigenvalues, we get for the components of the transformed solutions $z^n = z_j^n$:

$$z^{n+1} = \frac{1 + \mu - 2\mu^3 R_k^{(n)}(1-\mu)}{1-\mu} z^n$$

with $\mu = \frac{1}{2}\Delta t \lambda_j$.
The scheme is stable if for all time steps:

$$\left| \frac{1 + \mu - 2\mu^3 R_k(1-\mu)}{1-\mu} \right| \leqslant 1. \tag{25}$$

Following the same heuristic arguments as in the previous section, we conclude that the polynomial

$$Q(\mu) = 2\mu^3 R_k(1-\mu)$$

has in average almost the minimax property. This leads to the bound

$$-2.375 \leqslant \eta_1 \qquad (< 0), \tag{26}$$

where η_1 is defined as in Section 3.1 (cf. (17)).

4 Minimum residual iterations and stability control

We assume that the minimum residual iterations are carried out with GMRES [14, 1]. The iterations are applied to the linear system with matrix $\tilde{A}$ (for the EB-based MRAI, $\tilde{A} = I - \Delta t A$). In k steps of GMRES an orthogonal basis $\{\boldsymbol{v}_1, \dots, \boldsymbol{v}_k\}$ for the Krylov subspace $\mathsf{span}\{\boldsymbol{r}^n, \tilde{A}\boldsymbol{r}^n, \dots, \tilde{A}^{k-1}\boldsymbol{r}^n\}$ is built up. We take these vectors $\boldsymbol{v}_i$ as the columns of the matrix V_k. GMRES constructs a small $(k+1) \times k$ upper Hessenberg system

$$\underline{\tilde{H}} u = b, \qquad b = (\|\boldsymbol{r}^n\|_2, 0, \dots, 0)^T \in \mathbb{R}^{k+1}, \tag{27}$$

that is solved in the least squares sense, where $\underline{\tilde{H}} = V_{k+1}^* \tilde{A} V_k$.
The construction of V_{k+1} and $\underline{\tilde{H}}$ requires k matrix–vector multiplications with $\tilde{A}$ and $k(k+1)/2 + k$ inner products. The Jacobian matrix is not needed in explicit form, only its action on a vector is required. For instance, this may be approximated directly by a Frechét derivative in the direction of the vector on which the Jacobian acts. The MRAI algorithm is explicit and allows straightforward parallelization [3].

During the time-stepping process, the stability of MRAI can easily be checked with conditions like those inferred by (17) and (26), and, if necessary, the step size can be adjusted. We note that $\underline{\tilde{H}}$ depends on Δt as $\underline{\tilde{H}} = I - \Delta t \underline{H}$, where elements of $\underline{H}$ do not depend on Δt. The required value η_1 can be computed from the small upper Hessenberg system, since the $1 - \eta_i$ are the eigenvalues of the matrix

$$\tilde{H}^{-*}(\underline{\tilde{H}}^* \underline{\tilde{H}}),$$

where $\tilde{H}$ is the $k \times k$ upper part of $\underline{\tilde{H}}$. The values η_i are the harmonic Ritz values of $\tilde{A}$ [13].
The conditions (17) and (26) have been derived for a negative real spectrum of A. It turned out experimentally that when $k \geqslant 3$ they may also be used when the spectrum of A is contained in the left complex half-plane. Instead of η_1, we then use its real part.
For the simplified situation where $\boldsymbol{f} = \boldsymbol{f}(\boldsymbol{y})$, we have collected the major elements of two MRAI schemes, based on EB and Implicit Midpoint rule, in Tables 1 and 2. These allow adjustment of the step size for almost no extra price, when the Krylov basis has already been computed. This is possible because with the used starting vectors the initial residual vector $\boldsymbol{r}^n$ depends on Δt only in its direction and, hence, the Krylov basis matrix V does not depend on Δt.

For other starting vectors, as, for example, for starting vector (28) considered in the following Section, when the computed V depends on Δt, the stable value of Δt delivered by the stability control is used on the next time step.

5 Numerical experiments

In this chapter results for several numerical experiments are reported. They demonstrate the efficiency of the MRAI approach in cases where implicit time stepping is necessary but expensive.

We have tested our schemes for the typical situations where high accuracy is not needed in the time stepping, but fast stable integration is important. We illustrate this by two numerical examples: the heat conduction problem in 3D (Section 5.1) and an MHD (magnetohydrodynamical) 2D problem from astrophysical simulations (Section 5.2).

Table 1: First order MRAI based on Euler Backward

1.	$\boldsymbol{y}^n$ is given, $\boldsymbol{f}^n := \boldsymbol{f}(\boldsymbol{y}^n)$ $\boldsymbol{r}^n := A\boldsymbol{f}^n$, compute b in (27)	Function evaluation Jacobian action
2.	Modified Gram-Schmidt $\Rightarrow$ matrices V_{k+1} and $\underline{\tilde{H}}$	k Jacobian actions, $k+1$ vectors to store
3.	Choose Δt (cf. (17)), $b := (\Delta t)^2 b$, solve the least-square problem (27)	$\mathcal{O}(k^3)$ operations
4.	Starting vector: $\boldsymbol{y}_0^{n+1} := \boldsymbol{y}^n + \Delta t \boldsymbol{f}^n$ MRAI step: $\boldsymbol{y}^{n+1} := \boldsymbol{y}_0^{n+1} + V_k u$	k vector updates

Table 2: Second order MRAI based on Implicit Midpoint rule

1.	$\boldsymbol{y}^n$ is given, $\boldsymbol{f}^n := \boldsymbol{f}(\boldsymbol{y}^n)$ $\boldsymbol{p}^n := A\boldsymbol{f}^n$, $\boldsymbol{r}^n := \frac{1}{4}A\boldsymbol{p}^n$, compute b in (27)	Function evaluation 2 Jacobian actions
2.	Modified Gram-Schmidt $\Rightarrow$ matrices V_{k+1} and $\underline{\tilde{H}}$	k Jacobian actions, $k+1$ vectors to store
3.	Choose Δt (cf. (26)), $b := (\Delta t)^3 b$, solve the least-square problem (27)	$\mathcal{O}(k^3)$ operations
4.	Starting vector: $\boldsymbol{y}_0^{n+1} := \boldsymbol{y}^n + \Delta t \boldsymbol{f}^n + \frac{(\Delta t)^2}{2}\boldsymbol{p}^n$ MRAI step: $\boldsymbol{y}^{n+1} := \boldsymbol{y}_0^{n+1} + V_k u$	k vector updates

5.1 3D heat conduction problem

This test problem has been taken from [17]; it is a linear heat conduction problem over the 3D unit cube. The inhomogeneous term is chosen in order to have the analytical solution $\tanh(5(x+2y+1.5z-0.5-t))$. The spatial discretization is done with central differences on a grid of $79\times 39\times 39$ internal nodes. This yields a system of $N = 120\,159$ equations. The integration is done for $0 \leqslant t \leqslant t_{\text{END}} = 5.0$.

We report here on the EB-based MRAI scheme (EB/MRAI). We have incorporated the MRAI technique into the LSODE code [7], and the EB/MRAI variant has been triggered by restricting the consistency order in LSODE to one. The EB/MRAI code is a simple integration scheme with inexact Newton iterations where linear solves are done with k minimum residual iterations. In LSODE with consistency order restricted to one, the starting vector is chosen as

$$\boldsymbol{y}_0^{n+1} = \boldsymbol{y}^n + \frac{t_{n+1}-t_n}{t_n - t_{n-1}}(\boldsymbol{y}^n - \boldsymbol{y}^{n-1}), \tag{28}$$

and analysis together with experiments suggest that the stability condition is (cf. (17),(26))

$$-11 \leqslant \eta_1. \tag{29}$$

The value η_1 and the corresponding stable Δt are obtained at the first Newton iteration. This stable Δt value will be used in the stability control at the next time step. We do $k = 5$ minimum residual iterations in all Newton steps.

In the integration process the local error $\boldsymbol{e}^n$ is controlled to satisfy

$$|e_i^n| \leqslant \texttt{rtol}|y_i^n| + \texttt{atol}.$$

We have selected `rtol`=`atol`= 0.1. The step size is chosen to comply with local error and stability control. With such a crude accuracy tolerance, stability control prevails.

The size and the structure of the problem suggest the use of an iterative solver in implicit time stepping. Therefore we compare our EB/MRAI code with the implicit EB scheme with GMRES(m), $m = 20$ as a linear solver.

Of course in EB it is not necessary to solve the linear equations to full accuracy. In principle, this has a limiting effect on the stability but in practice it may lead to more efficiency. In our tests we have tried different stopping criteria for the GMRES iterations in EB/GMRES. The results are collected in Table 3. Here the CPU time is given in sec for one processor of an SGI Origin O2000. Clearly, MRAI outperforms the implicit scheme for a wide range of reasonable stopping criteria.

For a coarser grid, direct LU linear solve becomes feasible. For a $19 \times 19 \times 19$ grid, we have compared EB/MRAI and the fastest variant of the

Table 3: EB/MRAI versus EB/GMRES

	stopping criterion	CPU time	fevals /steps
EB/MRAI	—	253	1015/143
EB/GMRES	$\|\boldsymbol{r}_l^n\|/\|\boldsymbol{r}_0^n\| \leqslant 0.75$	366	1457/162
EB/GMRES	$\|\boldsymbol{r}_l^n\|/\|\boldsymbol{r}_0^n\| \leqslant 0.5$	318	1235/65
EB/GMRES	$\|\boldsymbol{r}_l^n\|/\|\boldsymbol{r}_0^n\| \leqslant 0.1$	391	1351/22
EB/GMRES	$\|\boldsymbol{r}_l^n\|/\|\boldsymbol{r}_0^n\| \leqslant 0.01$	532	1853/22
EB/GMRES	$\|\boldsymbol{r}_l^n\|/\|\boldsymbol{r}_0^n\| \leqslant 10^{-3}$	588	2048/20
EB/GMRES	$\|\boldsymbol{r}_l^n\| \leqslant 0.75$	1035	3596/20
EB/GMRES	$\|\boldsymbol{r}_l^n\| \leqslant 0.1$	1530	4940/20
EB/GMRES	$\|\boldsymbol{r}_l^n\| \leqslant 0.01$	1878	6549/20

Table 4: EB/MRAI and EB/GMRES versus EB/LU

	CPU time	fevals/steps
EB/MRAI	3.4	241/23
EB/GMRES	4.3	311/39
EB/LU	82	2194/21

EB/GMRES (the one with relative stopping criterion $\|\boldsymbol{r}_l^n\|/\|\boldsymbol{r}_0^n\| \leqslant 0.5$) with the implicit scheme employing direct banded LU linear solves. It is worthwhile to note that the LU factorizations are reused in the code, that is they are computed only once in a while. In our test only 3 LU factorizations were necessary. Still, the direct approach was by far more expensive (Table 4). To assure that we indeed had stable computations we checked the actual error (the maximum component in the difference of the computed and a reference solutions). For EB/MRAI it was 0.19, which is of order of the required tolerance. The error delivered by EB/LU scheme is much smaller: $7 \cdot 10^{-5}$. However, a similar accuracy can be achieved by EB/MRAI also in significantly less CPU time: error $8.2 \cdot 10^{-5}$ for `rtol`=`atol`=10^{-4} in 59 sec.

5.2 Magnetohydrodynamical simulation example

In this section we consider a slightly more complicated problem where we use also preconditioning. The problem comes from the magnetohydrodynamical (MHD) simulation Versatile Advection Code[2] (VAC) [18], [10].

This model problem is a two-dimensional MHD problem which models

[2] See URL `http://www.fys.ruu.nl/~toth/`.

Table 5: MRAI for an MHD simulation

Method	CPU time	matvecs	time steps
MRAI	225	1575	315
EB	291	3744	36
EF	533	–	1710

Table 6: MRAI for an MHD simulation: preconditioning

Method	CPU time	matvecs	time steps
MRAI+prec.	51	190	38
EB+prec.	64	511	32

the formation of a steady bow shock occurring in a super-fast flow around a perfectly conducting cylinder (for more details see [19]).

The equations are discretized spatially by high-resolution shock-capturing finite volume schemes [20]. For our example, a polar 60×60 grid is used, so that the discretization results in a system of $N = 21600$ nonlinear ODE's (for six different discretized variables). For this simulation the steady state solution is obtained by the time-stepping process until the relative difference in solution becomes sufficiently small:

$$\|\boldsymbol{y}^{n+1} - \boldsymbol{y}^{n}\|_{\star} \leqslant \varepsilon = 10^{-4}, \tag{30}$$

The norm $\|\cdot\|_{\star}$ is a special weighted norm with different weights for each of the conserved MHD variables.

We compare three time-stepping strategies: the MRAI scheme, the linearized EB scheme, and the EF scheme. In EB, the linear systems have been solved iteratively. We have chosen GMRES(m), $m = 20$ as an iterative solver. The number of restarts is restricted to 5. A simple strategy to stop the iterations is to stop whenever $\boldsymbol{r}_l^n \leqslant \varepsilon$ (cf. (30)), so that at every time step the residual in the implicit scheme is not larger than the prescribed steady state accuracy ε. The iterations in MRAI and EB can be preconditioned with an efficient modified block ILU (MBILU) preconditioning [22]. We compare the strategies both with and without preconditioning. The situation without preconditioning is meaningful in itself since it provides information about effectiveness of our approach for problems for which preconditioning is not readily available.

In EB the step size is chosen in such a way that the CFL number is 100, which means the step size is 100 times larger than the maximum allowed step size for EF. For EF Δt is chosen to have CFL number 0.8.

The MRAI variant is based on linearized EB. For MRAI, the starting vector has been chosen as $\boldsymbol{y}_0^{n+1} = \boldsymbol{y}^n$. This means that the formal consistency order of MRAI is zero, but consistency is not much of concern for steady state problems. A stability analysis similar to that of Section 3 suggests that this MRAI scheme has a much larger linear stability region than the first and second order MRAI schemes described in Sections 3.

It is important for the overall efficiency that MRAI allows for a cheap and flexible change of Δt (as in Section 4).

In the stability step size control used, the value of Δt was accepted when $-12.5 \leqslant \eta_1$ (cf. (17), (26)). In our experience this stability condition is quite useful for situations where the elements of $A + A^T$ have the same order of magnitude as $A - A^T$ (A is the Jacobian matrix).

Unfortunately, we do not have an appropriate stability control when preconditioning is involved. In the tests with preconditioning here we simply perform $k = 5$ minimum residual iterations per time step. The step size is chosen in the same way as in EB, that is the CFL number is 100.

The resulting MRAI is almost identical to the one represented in Table 1; the difference is in several simplifications: $\boldsymbol{r}^n := \boldsymbol{f}^n$, $b := \Delta t\, b$ in step 3, and $\boldsymbol{y}_0^{n+1} := \boldsymbol{y}^n$.

In Tables 5 and 6 we present results of the test runs carried out on one processor of an SGI Origin O2000. As we see, the MRAI scheme is about 25% faster than EB and much faster than EF.

For the unpreconditioned case, the simple strategy to adjust the step size to have the CFL number 100 leads to worse results for MRAI.

In column “matvecs” of Tables 5 and 6, the number of Jacobian-vector multiplications indicates the work involved for the efficient stable time stepping. This number is not directly proportional to the total CPU time for the following reason. In this simulation, after each time step it is necessary to update the computed magnetic field in order to make it divergence free, and this is rather expensive. Therefore schemes that require more time steps have a larger overhead in this case.

6 Conclusions

Stabilized explicit methods and implicit methods based on iterative linear solves become attractive when the Jacobian matrix is not easy to obtain and/or invert, or, in parallel processing.

We have studied the effect of approximate linear solves in implicit time stepping processes. A small fixed number of minimum residual iterations are performed in each linear solve. The resulting schemes can be seen as approximate implicit schemes, which are referred to as MRAI (Minimum Residual Approximated Implicit) schemes.

The schemes are in fact explicit, so that the stability is now of concern. We propose a convenient way to control the stability and adjust the step size adaptively.

Usually, the convergence in linear solves is checked by displaying the residual norm. In many cases it is not clear what tolerance has to be taken here. In our approach, different information coming from the linear solver itself is used for the stability control, and this is shown to be more efficient in practice.

Unlike many other explicit stabilized methods (see e.g. [23]), such as RKC, MRAI successfully copes with the complex spectrum Jacobian.

Our numerical experiments show that, when standard implicit schemes are too expensive and high accuracy is not required, MRAI can be very attractive.

Acknowledgments. The first author thanks Jos van Dorsselaer for helpful discussions. We thank referees for helping us to improve the presentation.
This work was supported by the Netherlands organization for scientific research NWO, project 95MPR04.

References

[1] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. A. van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods.* SIAM, Philadelphia, PA, 1994. Available at www.netlib.org/templates/.

[2] M. A. Botchev, G. L. G. Sleijpen, and H. A. van der Vorst. Low-dimensional Krylov subspace iterations for enhancing stability of time-step integration schemes. Technical Report 1004, Department of Mathematics, Utrecht University, Mar. 1997. Available at URL http:// www.math.uu.nl/publications/.

[3] M. A. Botchev and H. A. van der Vorst. Approximated implicit time-stepping schemes in a distributed memory parallel environment. Technical Report 1054, Department of Mathematics, Utrecht University, Mar. 1998. Available at URL http:// www.math. ruu.nl/publications/.

[4] G. D. Byrne, A. C. Hindmarsh, and P. N. Brown. VODPK, large non-stiff or stiff ordinary differential equation initial-value problem solver. Available at URL http://www.netlib.org, 1997.

[5] T. F. Chan and K. R. Jackson. The use of iterative linear-equation solvers in codes for large systems of stiff IVPs for ODEs. *SIAM J. Sci. Stat. Comput.*, 7(2):378–417, 1986.

[6] C. W. Gear and Y. Saad. Iterative solution of linear equations in ODE codes. *SIAM J. Sci. Stat. Comput.*, 4(4):583–601, 1983.

[7] A. C. Hindmarsh. LSODE: Livermore solver for ordinary differential equations. Available at URL http://www.netlib.org, 1987.

[8] M. Hochbruck and C. Lubich. On Krylov subspace approximations to the matrix exponential operator. *SIAM J. Numer. Anal.*, 34(5):1911–1925, Oct. 1997.

[9] M. Hochbruck, C. Lubich, and H. Selhofer. Exponential integrators for large systems of differential equations. *SIAM J. Sci. Comput.*, 19(5):1552–1574, 1998.

[10] R. Keppens, G. Tóth, M. A. Botchev, and A. van der Ploeg. Implicit and semi-implicit schemes in the Versatile Advection Code: algorithms. *Int. J. Numer. Methods in Fluids*, 30:335–352, 1999.

[11] V. I. Lebedev. Explicit difference schemes for solving stiff systems of ODEs and PDEs with complex spectrum. *Russian Journal of Numerical Analysis and Mathematical Modelling*, 13(2):107–116, 1998.

[12] V. O. Lokutsievskii and O. V. Lokutsievskii. On numerical solution of boundary value problems for equations of parabolic type. *Sov. Math. Dokl.*, 34(3):512–516, 1987.

[13] C. C. Paige, B. N. Parlett, and H. A. van der Vorst. Approximate solutions and eigenvalue bounds from Krylov subspaces. *Numer. Lin. Alg. with Appl.*, 2(2):115–133, 1995.

[14] Y. Saad and M. H. Schultz. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7(3):856–869, 1986.

[15] B. A. Schmitt and R. Weiner. Matrix-free W-methods using a multiple Arnoldi iteration. *Appl. Num. Math.*, 18(1–3):307–320, 1995.

[16] A. S. Shvedov and V. T. Zhukov. Explicit iterative difference schemes for parabolic equations. *Russian Journal of Numerical Analysis and Mathematical Modelling*, 13(2):133–148, 1998.

[17] B. P. Sommeijer, L. F. Shampine, and J. G. Verwer. RKC: An explicit solver for parabolic PDEs. *J. Comput. Appl. Math.*, 88:315–326, 1997.

[18] G. Tóth. General code for modeling MHD flows on parallel computers: Versatile Advection Code. *Astrophysical Letters & Communications*, 34:245–258, 1996.

[19] G. Tóth, R. Keppens, and M. A. Botchev. Implicit and semi-implicit schemes in the Versatile Advection Code: numerical tests. *Astronomy and Astrophysics*, 332:1159–1170, 1998.

[20] G. Tóth and D. Odstrčil. Comparison of some flux corrected transport and total variation diminishing numerical schemes for hydrodynamic and magnetohydrodynamic problems. *J. Comput. Physics*, 128:82–100, 1996.

[21] P. van der Houwen and B. P. Sommeijer. On the internal stability of explicit m-stage Runge–Kutta methods for large values of m. *Z. Angew. Math. Mech.*, 60:479–485, 1980.

[22] A. van der Ploeg, R. Keppens, and G. Tóth. Block incomplete LU-preconditioners for implicit solution of advection dominated problems. In B. Hertzberger and P. Sloot, editors, *High performance computing and networking*, Lecture Notes in Computer Science 1225, pages 421–430. Springer-Verlag, 1997.

[23] J. G. Verwer. Explicit Runge–Kutta methods for parabolic partial differential equations. *Appl. Num. Math.*, 22:359–379, 1996.