ZAMM · Z. angew. Math. Mech. 68 (1988) 9, 445 – 453

THIJE BOONKKAMP, J. H. M. TEN

Residual Smoothing for Accelerating the ADI Iteration Method for Elliptic Difference Equations

Residuale Glättung ist eine einfache Technik zur Erhöhung der Konvergenzrate von Iterationsverfahren für elliptische Differenzengleichungen. In dieser Arbeit kombinieren wir die residuale Glättung mit dem ADI-*Iterationsverfahren. Dies kann auf verschiedene Weise erfolgen. In geeigneter Weise angewandt kann residuale Glättung die Anzahl der Iterationen beträchtlich verringern und damit auch die Rechenzeit für das* ADI-*Schema reduzieren. Die Parameterwerte des geglätteten* ADI-*Schemas werden so gewählt, daß die hoch- wie auch die niederfrequenten Komponenten im Iterationsfehler sehr gut gedämpft werden. Durch die residuale Glättung werden auch die anderen Komponenten im Fehler geeignet gedämpft. Numerische Beispiele zeigen die Leistungsfähigkeit des geglätteten* ADI-*Schemas gegenüber dem* ADI-*Schema.*

Residual smoothing is a simple technique to increase the rate of convergence of iterative methods for elliptic difference equations. In this paper, we combine residual smoothing with the ADI *iteration method, which can be done in several ways. When applied in the proper way, residual smoothing can considerably reduce the number of iterations and thus the computing time of the* ADI *scheme. The parameter values of the smoothed* ADI *scheme are chosen such that the high- and low-frequency components in the iteration error are damped very well. Due to the residual smoothing, the other components in the error are also properly damped. Numerical examples demonstrate the performance results of the* ADI *scheme and the smoothed* ADI *scheme.*

Резидуальное сглаживание является простым методом для ускорения скорости сходимости итерационных методов для решения эллиптических разностных уравнений. В этой статье сочетаем резидуальное сглаживание с итерационным методом альтернирующих направлений (АДИ в ангийской литературы — здесь МАН), что можно делать различным образом. Если применяется резидуальное сглаживание правильным образом, этот метод значительно может уменьшать число итераций и таким образом и время вычисления схемы МАН. Значения параметров сглаженной схемы МАН выбираются так, что высокочастотные и низкочастотные компоненты в ошибке итерации тормозируются очень хорошо. Благодаря резидуального сглаживания остальные компоненты в ошибке тоже тормозируются должным образом. Численные примеры демонстрируют производительность схемы МАН и сглаженной схемы МАН.

1. Introduction

We consider the first boundary-value problem for the two-dimensional elliptic partial differential equation (PDE)

$$(p(x, y)\, u_x)_x + (q(x, y)\, u_y)_y - w(x, y)\, u = f(x, y)\,, \qquad (x, y) \in \Omega = [0, 1] \times [0, 1]\,, \tag{1.1}$$

where $p(x, y) > 0$, $q(x, y) > 0$ and $w(x, y) \geqq 0$. As a special case of (1.1) we employ the Poisson equation

$$u_{xx} + u_{yy} = f(x, y) \tag{1.2}$$

as a model problem.

For space discretization, we cover Ω with a uniform space grid with gridsize h, where $h = 1/(M + 1)$ and M is the number of internal gridpoints in x- and y-direction. Space discretization of (1.1), using standard central differences, yields a difference system

$$D_{xx}U + D_{yy}U = B\,. \tag{1.3}$$

In (1.3) U is a vector, with components U_{ij}, and B is a vector originating from the right hand side f and the boundary conditions for u. The component U_{ij} is the finite difference approximation to $u(ih, jh)$. The matrices D_{xx} and D_{yy} in (1.3) are the finite difference replacements of respectively

$$\frac{\partial}{\partial x}\left(p(x, y)\,\frac{\partial}{\partial x}\right) - \frac{1}{2}\,w(x, y) \qquad \text{and} \qquad \frac{\partial}{\partial y}\left(q(x, y)\,\frac{\partial}{\partial y}\right) - \frac{1}{2}\,w(x, y)$$

and are defined by

$$(D_{xx}U)_{ij} := \frac{1}{h^2}\left(p_{i-(1/2),j}\,U_{i-1,j} - (p_{i-(1/2),j} + p_{i+(1/2),j})\,U_{ij} + p_{i+(1/2),j}U_{i+1,j}\right) - \frac{1}{2}\,w_{ij}U_{ij}\,, \tag{1.4a}$$

$$(D_{yy}U)_{ij} := \frac{1}{h^2}\left(q_{i,j-(1/2)}U_{i,j-1} - (q_{i,j-(1/2)} + q_{i,j+(1/2)})\,U_{ij} + q_{i,j+(1/2)}U_{i,j+1}\right) - \frac{1}{2}\,w_{ij}U_{ij}\,, \tag{1.4b}$$

with $p_{i\pm(1/2),j} = p((i \pm 1/2)\,h, jh)$ (analogous definitions for $q_{i,j\pm(1/2)}$ and w_{ij}). The matrices D_{xx} and D_{yy} are tridiagonal, symmetric and negative definite.

For the iterative solution of (1.3) we examine the ADI scheme of PEACEMAN and RACHFORD [3, 5]. For the model problem, the ADI scheme is known to be a fast scheme if one chooses its parameter values in the right way. However, the scheme is very sensitive to the parameter values used, i.e., the iteration count grows rapidly when the the computation is carried out away from the optimal parameter values. Therefore, the ADI scheme is in general not a fast iteration technique. It is the purpose of this paper to apply residual smoothing for improving the rate of convergence of the ADI scheme and, most importantly, to make the scheme less sensitive to the choice of the parameter values. This paper is inspired by [2], where residual smoothing is applied to Jacobi iteration.

The contents of the paper is the following. In Section 2 a short outline of the theory of residual smoothing is given. The ADI scheme and the smoothed ADI scheme are discussed in Section 3 and parameter values for both schemes are given in Section 4. Section 5 is devoted to a numerical comparison between the ADI scheme and the smoothed ADI scheme. This comparison also involves a nonlinear example. In Section 6, an alternative smoothed ADI scheme is briefly discussed. Some conclusions are formulated in Section 7.

2. Residual smoothing

In this section we give a short outline of the theory of residual smoothing as a means of accelerating the convergence of iterative methods for elliptic difference equations. For a more extensive treatment of the special type of explicit residual smoothing used here, the reader is referred to [2].

Consider the linear system

$$AU = B\,, \tag{2.1}$$

obtained by discretizing a linear elliptic boundary value problem. We assume that A has negative eigenvalues. Iterative methods for solving (2.1) are based upon the splitting $A = P - Q$, where P is a non-singular and easily invertible matrix [1, 5]. The iteration scheme thus takes the form

$$PU^{n+1} = QU^n + B\,, \tag{2.2}$$

or equivalently, in residual form,

$$PU^{n+1} = PU^n - (AU^n - B)\,. \tag{2.2'}$$

The idea of *residual smoothing* is now to multiply the residual in (2.2') by a matrix S such that the condition number of SA is much smaller than the condition number of A. The iteration scheme then reads

$$PU^{n+1} = PU^n - S(AU^n - B)\,. \tag{2.3}$$

Thus, instead of solving (2.1), we solve the preconditioned system $SAU = SB$ with the original iteration method.

Following [2], S is taken of the form $S = P_k(D)$, where $P_k(z)$ is a polynomial of degree k satisfying $P_k(0) = 1$ and D is a scaled difference matrix with eigenvalues in the interval $[-1, 0]$. In order to analyse the residual smoothing technique we choose

$$D = \frac{1}{\varrho} A\,, \tag{2.4}$$

where $\varrho = \varrho(A)$ is the spectral radius of A. In [2], for this choice, an optimal smoothing matrix $S = P_k(D)$ is derived, in the sense that SA has negative eigenvalues and the smallest possible condition number. The *condition number* $\gamma(A)$ of a matrix A is defined as $\gamma(A) = \varrho(A)/\delta(A)$, where $\delta(A)$ is the in absolute value smallest eigenvalue of A. The polynomial $P_k(z)$ is given by

$$P_k(z) = \frac{T_{k+1}(1 + 2z) - 1}{2(k+1)^2 z}\,, \tag{2.5}$$

where $T_k(z)$ is the kth degree Chebyshev polynomial of the first kind. Because of the factorization properties of the Chebyshev polynomials, the smoothing matrix S can be computed very efficiently if $k = 2^q - 1$ for some positive integer q.

We emphasize, however, that in actual computations we do not use the difference matrix D defined by(2.4), because it is much too expensive in the general case. Instead, for one-dimensional problems, the matrix D is given by

$$D = \frac{1}{4}\begin{pmatrix} 0 & & & & & & & \\ 1 & -2 & 1 & & & & & \\ & 1 & -2 & 1 & & & & \\ & & \ddots & \ddots & \ddots & & & \\ & & & \ddots & \ddots & \ddots & & \\ & & & & \ddots & \ddots & \ddots & \\ & & & & & 1 & -2 & 1 \\ & & & & & & & 0 \end{pmatrix}. \tag{2.6}$$

For two-dimensional computations we do not use the two-dimensional analogue of (2.6) because the computation of $S = P_k(D)$ is then not attractive [2]. Therefore, we consider an alternative which only uses one-dimensional smoothing matrices. The residual $r = AU^n - B$ in (2.2') can in the two-dimensional case be written as $r = D_{xx}U^n + D_{yy}U^n - B$ (Cf (1.3)). The residual is then smoothed by applying the one-dimensional smoothing matrix to D_{xx} and/or D_{yy}. In other words, let the residual r be arranged in a two-dimensional array in the natural way, then r is smoothed by applying the one-dimensional smoothing matrix to all rows and/or columns of r.

3. ADI- and smoothed ADI iteration

Consider equation (1.3)

$$AU = B\,, \qquad A = D_{xx} + D_{yy}\,. \tag{3.1}$$

The ADI scheme for (3.1) can be written in residual form as [5]

$$(D_{xx} - \nu_1 I)\, U^* = (D_{xx} - \nu_1 I)\, U^n - (AU^n - B)\,, \tag{3.2a}$$

$$(D_{yy} - \nu_2 I)\, U^{n+1} = (D_{yy} - \nu_2 I)\, U^* - (AU^* - B)\,, \tag{3.2b}$$

where $\nu_1, \nu_2 > 0$ and are supposed to be independent of n.

The first stage (3.2a) of the ADI scheme is implicit in x-direction and explicit in y-direction. This suggests to apply in (3.2a) a smoothing matrix $\tilde{S}_y$ for the preconditioning of D_{yy}. In other words, we multiply each column of the residual in (3.2a) by a (one-dimensional) smoothing matrix S_y, where S_y is such that $S_y D_{yy}$ has the smallest possible condition number. In the same way, we apply a smoothing matrix $\tilde{S}_x$ at the second stage (3.2b) for the preconditioning of D_{xx}. Each row of the residual in (3.2b) is then multiplied by a (one-dimensional) smoothing matrix S_x. The *smoothed* ADI (SADI) *scheme* then reads

$$(D_{xx} - \nu_1 I)\, U^* = (D_{xx} - \nu_1 I) U^n - \tilde{S}_y (AU^n - B)\,, \tag{3.3a}$$

$$(D_{yy} - \nu_2 I)\, U^{n+1} = (D_{yy} - \nu_2 I)\, U^* - \tilde{S}_x (AU^* - B)\,. \tag{3.3b}$$

In the analysis, the operators S_x and S_y are defined by $S_x = P_k\left(\frac{1}{\varrho_1} D_{xx}\right)$ and $S_y = P_k\left(\frac{1}{\varrho_2} D_{yy}\right)$ where $\varrho_1 = \varrho(D_{xx})$ and $\varrho_2 = \varrho(D_{yy})$. We emphasize once more, that in practice the matrices $\frac{1}{\varrho_1} D_{xx}$ and $\frac{1}{\varrho_2} D_{yy}$ are replaced by difference matrices like the one defined in (2.6). The degree k of the polynomial $P_k(z)$ will be specified later.

If $D_{xx} - \frac{1}{2}(\nu_1 - \nu_2) I$ and $D_{yy} + \frac{1}{2}(\nu_1 - \nu_2) I$ are negative definite then the ADI scheme is *convergent* [5]. Likewise, *the* SADI *scheme is convergent if* $D_{xx} - D_{yy} + S_x A - (\nu_1 - \nu_2) I$ *and* $-D_{xx} + D_{yy} + S_y A + (\nu_1 - \nu_2) I$ *are negative definite.* The proof is along the same lines as the proof for ADI.

In order to get an indication about the performance of both the ADI scheme and the SADI scheme, we consider the eigenvalues of the iteration matrix of both schemes. These eigenvalues are called the *damping factors of the iteration scheme.* In the remainder of the paper we consider the following two cases:

$$\text{case 1:}\quad \varrho(D_{xx}) = \varrho(D_{yy}) = \varrho\,, \qquad \delta(D_{xx}) = \delta(D_{yy}) = \delta\,,$$

$$\text{case 2:}\quad \varrho_1 = \varrho(D_{xx}) \neq \varrho_2 = \varrho(D_{yy})\,, \qquad \delta_1 = \delta(D_{xx}) \neq \delta_2 = \delta(D_{yy})\,.$$

For simplicity, we take $\nu_1 = \nu_2 = \nu$, unless stated otherwise, and assume that D_{xx} and D_{yy} commute.

First, we restrict ourselves to case 1. The damping factor of the ADI scheme is given by

$$\xi = \xi(\lambda_x, \lambda_y; \nu) = \frac{(\lambda_x + \nu)\,(\lambda_y + \nu)}{(\lambda_x - \nu)\,(\lambda_y - \nu)}\,, \tag{3.4}$$

where λ_x and λ_y are the eigenvalues of D_{xx} and D_{yy}, respectively ($\lambda_x, \lambda_y < 0$). It is convenient to write ξ as a function of the scaled eigenvalues $\mu_x := \lambda_x/\varrho$ and $\mu_y := \lambda_y/\varrho$, so that

$$\xi = \xi(\mu_x, \mu_y; \omega) = \frac{(\mu_x + \omega)\,(\mu_y + \omega)}{(\mu_x - \omega)\,(\mu_y - \omega)}\,, \tag{3.5}$$

where $\omega := \nu/\varrho$. The parameter ω should be chosen in the range $0 < \omega \leqq 1$ [5]. In Fig. 1 $\xi(\mu_x, \mu_y; \omega)$ is plotted for $\mu_x = \mu_y$ and for $\omega = 1,\ 10^{-1},\ 10^{-2},\ 10^{-3}$. For $\mu_y = a\mu_x$ ($a \neq 1$) the graph of $|\xi(\mu_x, \mu_y; \omega)|$ displays a similar behaviour.

From (3.3) one can easily see that the damping factor of the SADI scheme is given by

$$\xi = \xi(\lambda_x, \lambda_y; \nu) = \frac{\lambda_y - \nu - P_k(\lambda_x/\varrho)\,(\lambda_x + \lambda_y)}{\lambda_x - \nu} \cdot \frac{\lambda_x - \nu - P_k(\lambda_y/\varrho)\,(\lambda_x + \lambda_y)}{\lambda_y - \nu}\,, \tag{3.6a}$$

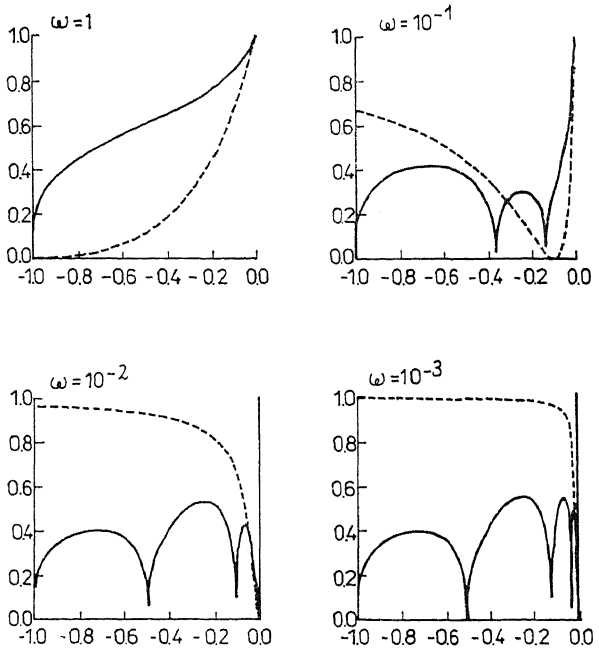
or equivalently as a function of μ_x and μ_y

$$\xi = \xi(\mu_x, \mu_y; \omega) = \frac{\mu_y - \omega - P_k(\mu_x)\,(\mu_x + \mu_y)}{\mu_x - \omega} \cdot \frac{\mu_x - \omega - P_k(\mu_y)\,(\mu_x + \mu_y)}{\mu_y - \omega}\,. \tag{3.6b}$$

Note that $\xi(\mu_x, \mu_y; \omega) = 1$ in all points where $P_k(\mu_x) = P_k(\mu_y) = 0$. This implies that we should not iterate with a fixed value of k and ω. Therefore, we consider cyclic methods where $k = k_q$ and $\omega = \omega_q$, k_q and ω_q being periodic functions of q: $k_q = k_{q+N}$, $\omega_q = \omega_{q+N}$ with N fixed. In our experiments we choose $k_q = 2^q - 1$ ($q = 0(1)\, N - 1$) since then the smoothing matrices can be computed very efficiently [2]. The integer N will be specified later. Instead of $\xi = \xi_q(\mu_x, \mu_y; \omega_q)$ we thus consider the *average damping factor*

$$\alpha = \alpha(\mu_x, \mu_y; \omega_0, \ldots, \omega_{N-1}) := \Big[\prod_{q=0}^{N-1} |\xi_q(\mu_x, \mu_y; \omega_q)|\Big]^{1/N}\,. \tag{3.7}$$

Since $\xi_0(-1, -1; \omega_0) = \left(\frac{\omega_0 - 1}{\omega_0 + 1}\right)^2$ and $\xi_q(-1, -1; \omega_q) = 1$ for $q > 0$, we choose $\omega_0 = 1$ in order to damp the eigenvector components in the iteration error which correspond to values of μ_x, μ_y close to -1. These components



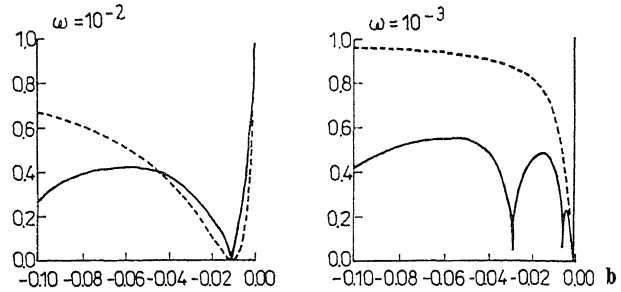


Fig. 1a. The damping factors for the ADI scheme and the average damping factors for the SADI scheme on the interval $-1 \leqq \mu \leqq 0$ for $\omega = 1, 10^{-1}, 10^{-2}, 10^{-3}$. - - - - - - ADI, ——— SADI

Fig. 1b. The damping factors for the ADI scheme and the average damping factors for the SADI scheme on the interval $-0.1 \leqq \mu \leqq 0$ for $\omega = 10^{-2}, 10^{-3}$. - - - - - - ADI, ——— SADI

are the high frequency components. Likewise, the low frequency components correspond to values of μ_x, μ_y close to 0. The other ω_q values are chosen equal: $\omega_q = \omega$ for $q > 0$. The average damping factor $\alpha = \alpha(\mu_x, \mu_y; \omega) := \alpha(\mu_x, \mu_y; 1, \omega, \ldots, \omega)$ of the SADI scheme is also plotted in Fig. 1. for $\mu_x = \mu_y$, $N = 6$ and $\omega = 1, 10^{-1}, 10^{-2}, 10^{-3}$. Also in this case, the graph of $\alpha(\mu_x, a\mu_x; \omega)$ $(a \neq 1)$ is very similar to the graph of $\alpha(\mu_x, \mu_y; \omega)$.

Comparing both damping factors, we see that for small ω-values $(10^{-3} \leqq \omega \leqq 10^{-2})$ the SADI scheme has substantial better damping properties than the ADI-scheme. In particular, with the exception of the lowest ones $(\mu \approx 0)$, SADI damps all error components with a factor of a least 0.6.

4. Choice of the parameter values

In this section we derive parameter values for the SADI scheme. The derivation of parameter values for the ADI scheme (3.2) is extensively described in [5], therefore we only present the results.

The damping factor $\xi(\lambda_x, \lambda_y; \nu)$ of the ADI scheme in case 1 is given by (3.4). We choose the ν-parameter to minimize the function

$$\psi = \psi(\nu; \varrho, \delta) := \max_{-\varrho \leqq \lambda_x, \lambda_y \leqq -\delta} |\xi(\lambda_x, \lambda_y; \nu)| . \tag{4.1}$$

Asymptotically, the eigenvector corresponding to the maximum damping factor dominates the error. Therefore, in order to minimize the number of iterations, we have to minimize $\psi(\nu; \varrho, \delta)$. We emphasize, however, that this only applies if we compute the solution sufficiently accurate. For moderate accurate computations, the ν-value thus obtained can be far from optimal, i. e., the corresponding number of iterations is far from minimal. A simple analysis gives that the optimal parameter is given by $\nu^* = (\delta\varrho)^{1/2}$ [5].

Example 1: Consider the Poisson equation. The eigenvalues λ_x and λ_y of D_{xx} and D_{yy} are given by $\lambda_{x,i} = \lambda_{y,i} = -\frac{4}{h^2}\sin^2\left(\frac{\pi}{2} ih\right)$, $i = 1(1) M$, with $h = 1/(M+1)$. In this case $\varrho(D_{xx}) = \varrho(D_{xx}) = \varrho \approx \frac{4}{h^2}$ and $\delta(D_{xx}) = \delta(D_{yy}) = \delta \approx \pi^2$, so that $\nu^* \approx \frac{2\pi}{h}$.

In case 2, the function ψ to be minimized is defined by

$$\psi = \psi(\nu; \varrho_1, \delta_1, \varrho_2, \delta_2) := \max_{\substack{-\varrho_1 \leqq \lambda_x \leqq -\delta_1 \\ -\varrho_2 \leqq \lambda_y \leqq -\delta_2}} |\xi(\lambda_x, \lambda_y; \nu)| . \tag{4.1'}$$

Assume that $\varrho_1\delta_1 \leqq \varrho_2\delta_2$. Then one can prove the following result for the ADI scheme [5]: *if* $\delta_1 \geqq \delta_2$ *or* $\delta_1 \leqq \delta_2$ *and* $\delta_1\varrho_2 \geqq \delta_2\varrho_1$ *then* $\nu^* = (\delta_1\varrho_1)^{1/2}$, *and if* $\varrho_1 \geqq \varrho_2$ *or* $\varrho_1 \leqq \varrho_2$ *and* $\delta_1\varrho_2 \leqq \delta_2\varrho_1$ *then* $\nu^* = (\delta_2\varrho_2)^{1/2}$.

Consider the SADI scheme. In case 1, the damping factor $\xi(\lambda_x, \lambda_y; \nu)$ is given by (3.6a). Since $\xi(\lambda_x, \lambda_y; \nu) = 1$ for all λ_x, λ_y for which $P_k(\lambda_x/\varrho) = P_k(\lambda_y/\varrho) = 0$, we have to iterate with varying $k = k_q$ and $\nu = \nu_q$ (see Section 3). Therefore, instead of $\xi = \xi_q(\lambda_x, \lambda_y; \nu_q)$ we consider the average damping factor α defined by (cf. (3.7))

$$\alpha = \alpha(\lambda_x, \lambda_y; \nu_0, \ldots, \nu_{N-1}) =: [\prod_{q=0}^{N-1} |\xi(\lambda_x, \lambda_y; \nu_q)|]^{1/N} . \tag{4.2}$$

In order to damp the high frequency components, we require $\xi_0(-\varrho, \lambda_y; \nu_0) = \xi_0(\lambda_x, -\varrho; \nu_0) = 0$, which gives $\nu_0 = \varrho$. For the other ν_q-values we choose $\nu_q = \nu$, $q > 0$. This ν-value is chosen to minimize $\alpha(-\delta, -\delta; \nu) := \alpha(-\delta, -\delta; \varrho, \nu, \ldots, \nu)$ because of the following reasons:

(i) the lowest frequency eigenvector corresponding to $\lambda_x = \lambda_y = -\delta$ has often a large weight in the error,
(ii) the eigenvalue $\lambda_x = \lambda_y = -\delta$ is either known or can be approximated.

In this way we construct a SADI scheme which damps the high- and low-frequency components in the iteration error very well. It turns out that a SADI scheme constructed this way also damps the remaining error components very well, as illustrated before in Fig. 1.

So consider $\alpha(-\delta, -\delta; \nu)$, which can be written as

$$\alpha(-\delta, -\delta; \nu) = \left(\left(\frac{\varrho - \delta}{\varrho + \delta}\right)^2 \prod_{q=1}^{N-1} \xi_q(-\delta, -\delta; \nu)\right)^{1/N}, \tag{4.3a}$$

with

$$\xi_q(-\delta, -\delta; \nu) = \left(1 - P_k\left(-\frac{\delta}{\varrho}\right)\frac{2\delta}{\delta + \nu}\right)^2, \qquad k = 2^q - 1, \qquad q = 1(1)\,N - 1. \tag{4.3b}$$

If $\xi_q(-\delta, -\delta; \nu) = 0$ for some $q > 0$ then $\alpha(-\delta, -\delta; \nu) = 0$, and thus $\alpha(-\delta, -\delta; \nu)$ is minimal. From (4.3b) one can easily see that $\xi_q(-\delta, -\delta; \nu) = 0$ if $\nu = \nu_k = \delta\left(2P_k\left(-\frac{\delta}{\varrho}\right) - 1\right)$, provided $P_k\left(-\frac{\delta}{\varrho}\right) > \frac{1}{2}$. A Taylor series expansion yields

$$P_k\left(-\frac{\delta}{\varrho}\right) \approx 1 - a_k\frac{\delta}{\varrho}, \qquad a_k := \frac{1}{3}k(k+2), \tag{4.4}$$

if $b_k := \frac{2}{45}\left(\frac{\delta}{\varrho}\right)^2 (k+1)^4 \ll 1$. For k sufficiently small, this condition is fulfilled and ν_k is approximately given by $\nu_k = c_k\delta$, $c_k := 1 - 2a_k\frac{\delta}{\varrho}$. In our numerical experiments we take $\nu^* = \nu_1 \approx \delta$ (see Table 1).

Example 2: Consider again the Poisson equation for which $\varrho \approx 4/h^2$ and $\delta \approx \pi^2$. In this case we have

$$P_k\left(-\frac{\delta}{\varrho}\right) \approx 1 - \frac{\pi^2}{12}\left(\frac{k+1}{M+1}\right)^2, \qquad b_k = \frac{\pi^4}{360}\left(\frac{k+1}{M+1}\right)^4, \qquad c_k \approx 1 - \frac{\pi^2}{6}\left(\frac{k+1}{M+1}\right)^2.$$

These values for $k = 2^q - 1$ ($q = 1(1)\,5$) and for $M = 39$ are given in Table 1. Note that the value c_{31} does not make sense since $P_{31}\left(\approx\frac{\delta}{\varrho}\right) < \frac{1}{2}$. For the general elliptic case one finds similar results since the ratio $\frac{\delta}{\varrho} = O(h^2)$ just as for the Poisson equation.

Table 1. $P_k(-\delta/\varrho)$-, b_k- and c_k-values for the Poisson equation for $k = 2^q - 1 (q = 1(1)\,5)$ and $M = 39$

k	$P_k(-\delta/\varrho)$	b_k	c_k
1	0.9979	$1.69^* 10^{-6}$	0.9959
3	0.9918	$2.71^* 10^{-5}$	0.9836
7	0.9671	$4.33^* 10^{-4}$	0.9342
15	0.8684	$6.93^* 10^{-3}$	0.7368
31	0.4736	$1.11^* 10^{-1}$	−0.0528

In case 2, the damping factor of the SADI scheme can be written as (cf. (3.6a))

$$\xi = \xi(\lambda_x, \lambda_y; \nu_1, \nu_2) = \frac{\lambda_y - \nu_2 - P_k(\lambda_x/\varrho_1)(\lambda_x + \lambda_y)}{\lambda_x - \nu_1} \cdot \frac{\lambda_x - \nu_1 - P_k(\lambda_y/\varrho_2)(\lambda_x + \lambda_y)}{\lambda_y - \nu_2}. \tag{4.5}$$

Note that in (4.5) we assume that $\nu_1 \neq \nu_2$. The corresponding average damping factor is given by (4.2) with, $\xi = \xi_q(\lambda_x, \lambda_y; \nu_{1q}, \nu_{2q})$ defined by (4.5). For the damping of the high frequency components we require $\xi_0(-\varrho_1, \lambda_y; \nu_{10}, \nu_{20}) = \xi_0(\lambda_x, -\varrho_2; \nu_{10}, \nu_{20}) = 0$, which implies that we indeed should iterate with two different ν-values (cf. (3.3)). This gives $\nu_{10} = \varrho_2$ and $\nu_{20} = \varrho_1$. For $q > 0$ we choose $\nu_{1q} = \nu_1$ and $\nu_{2q} = \nu_2$. These two values are chosen to minimize $\alpha(-\delta_1, -\delta_2; \nu_1, \nu_2)$, which can be written as

$$\alpha(-\delta_1, -\delta_2; \nu_1, \nu_2) = \left(\frac{\varrho_1 - \delta_1}{\varrho_2 + \delta_1} \cdot \frac{\varrho_2 - \delta_2}{\varrho_1 + \delta_2} \prod_{q=1}^{N-1} |\xi_q(-\delta_1, -\delta_2; \nu_1, \nu_2)|\right)^{1/N}, \tag{4.6a}$$

with

$$\xi_q(-\delta_1, -\delta_2; \nu_1, \nu_2) = \left(1 - P_k\left(-\frac{\delta_1}{\varrho_1}\right)\cdot\frac{\delta_1 + \delta_2}{\delta_2 + \nu_2}\right)\cdot\left(1 - P_k\left(-\frac{\delta_2}{\varrho_2}\right)\cdot\frac{\delta_1 + \delta_2}{\delta_1 + \nu_1}\right). \tag{4.6b}$$

Also in this case, if $\xi_q(-\delta_1, -\delta_2; \nu_1, \nu_2) = 0$ for some $q > 0$ then $\alpha(-\delta_1, -\delta_2; \nu_1, \nu_2)$ is minimal as a function of ν_1 and ν_2. From (4.6b) one can readily see that this condition is fulfilled if $\nu_1 = \nu_{1,k} = P_k(-\delta_2/\varrho_2)(\delta_1 + \delta_2) - \delta_1$ or $\nu_2 = \nu_{2,k} = P_k(-\delta_1/\varrho_1)(\delta_1 + \delta_2) - \delta_2$ provided that $P_k(-\delta_2/\varrho_2) > \delta_1/(\delta_1 + \delta_2)$ or $P_k(-\delta_1/\varrho_1) > \delta_2/(\delta_1 + \delta_2)$. Substitution of the approximation $P_k(-\delta_i/\varrho_i) = 1 - a_k\delta_i/\varrho_i$, $i = 1, 2$ (see (4.4)), then yields the following expression for $\nu_{1,k}$ and $\nu_{2,k}$: $\nu_{1,k} = \delta_2 - a_k(\delta_2/\varrho_2)(\delta_1 + \delta_2)$ and $\nu_{2,k} = \delta_1 - a_k(\delta_1/\varrho_1)(\delta_1 + \delta_2)$. As in case 1, we choose the following approximation: $\nu_1^* = \nu_{1,1} \approx \delta_2$ and $\nu_2^* = \nu_{2,1} \approx \delta_1$.

For the computation of the parameter values for both schemes the values of $\delta(D_{xx}), \delta(D_{yy}), \varrho(D_{xx})$ and $\varrho(D_{yy})$ are required. As we have seen, for the Poisson equation $\varrho(D_{xx}) = \varrho(D_{yy}) \approx \frac{4}{h^2}$ and $\delta(D_{xx}) = \delta(D_{yy}) \approx \pi^2$. For the general elliptic equation (1.3) these values can only be approximated as follows. Consider the general matrix D_{xx} defined by (1.4a). Let $\overline{p} := \max_{0 \leqq x,y \leqq 1} p(x, y)$, $\underline{p} := \min_{0 \leqq x,y \leqq 1} p(x, y)$ and analogous definitions for $\overline{q}, \underline{q}, \overline{w}$ and $\underline{w}$. Let the matrices $\overline{D}_{xx}$ and $\underline{D}_{xx}$ be defined by replacing $p_{i \pm (1/2),j}$ and w_{ij} in (1.4a) by $\overline{p}$ and $\overline{w}$ respectively $\underline{p}$ and $\underline{w}$. In other words, $\overline{D}_{xx} = \overline{p}\delta_{xx} - \frac{1}{2}\overline{w}I$ and $\underline{D}_{xx} = \underline{p}\delta_{xx} - \frac{1}{2}\underline{w}I$, with δ_{xx} denoting the standard central difference approximation to $\partial^2/\partial x^2$. Then one can easily show that

$$\varrho(\underline{D}_{xx}) \leqq \varrho(D_{xx}) \leqq \varrho(\overline{D}_{xx}) \quad \text{and} \quad \delta(\underline{D}_{xx}) \leqq \delta(D_{xx}) \leqq \delta(\overline{D}_{xx}) .$$

The values $\varrho(D_{xx})$ and $\delta(D_{xx})$ can then be approximated by

$$\varrho(D_{xx}) \approx \frac{1}{2}\left(\varrho(\overline{D}_{xx}) + \varrho(\underline{D}_{xx})\right) = \frac{2}{h^2}(\overline{p} + \underline{p}) + \frac{1}{4}(\overline{w} + \underline{w})$$

and

$$\delta(D_{xx}) \approx \frac{1}{2}\left(\delta(\overline{D}_{xx}) + \delta(\underline{D}_{xx})\right) = \frac{\pi^2}{2}(\overline{p} + \underline{p}) + \frac{1}{4}(\overline{w} + \underline{w}) .$$

In the same way one finds

$$\varrho(D_{yy}) \approx \frac{2}{h^2}(\overline{q} + \underline{q}) + \frac{1}{4}(\overline{w} + \underline{w}) \quad \text{and} \quad \delta(D_{yy}) \approx \frac{\pi^2}{2}(\overline{q} + \underline{q}) + \frac{1}{4}(\overline{w} + \underline{w}) .$$

5. Numerical examples

In this section we present a few numerical examples, in order to compare the ADI scheme and the SADI scheme. We restrict ourselves to Dirichlet problems. The solution is computed for $h = \frac{1}{20}, \frac{1}{40}, \frac{1}{80}$ with the parameter values derived in Section 4. In addition, we compute the solution for $h = \frac{1}{40}$ for several ν-values, in order to check whether the ν-values derived in Section 4 are good enough. Further, to demonstrate the power of residual smoothing, we apply the SADI scheme to a nonlinear problem.

For the degree k of the smoothing matrices we choose $k = k_q = 2^q - 1$, $q = 0(1)\, N - 1$, such that k_{N-1} is the largest k_q smaller than $M = h^{-1} - 1$. The reason for this is, that for $k_q > M$ for some q, the computation of the smoothing matrices becomes cumbersome. Thus for $h = \frac{1}{20}, \frac{1}{40}, \frac{1}{80}$ we choose, respectively, $N = 5, 6, 7$. We emphasize once more that the choice $k_q = 2^q - 1$ admits an efficient computation of the smoothing matrices [2], which is a prerequisite for accelerating the ADI scheme. In all computations, the initial approximation is defined by forming linear interpolations of the boundary values on $x = 0$, $x = 1$ and on $y = 0$, $y = 1$, respectively, and by taking the average value of these functions. The iteration is stopped if the scaled residual

$$r(n) := \frac{\|AU^n - B\|_1}{\|AU^0 - B\|_1} \tag{5.1}$$

has dropped below a certain tolerance TOL.

The examples we consider are the following.

Example 1 [4, p. 427]:

$$u_{xx} + u_{yy} = f(x, y) , \qquad u(x, y) = 3\,\mathrm{e}^{x+y}(x - x^2)(y - y^2) , \qquad f(x, y) = 6xy\,\mathrm{e}^{x+y}(xy + x + y - 3) ,$$

$$\varrho = \varrho(D_{xx}) = \varrho(D_{yy}) = \frac{4}{h^2}, \qquad \delta = \delta(D_{xx}) = \delta(D_{yy}) = \pi^2 .$$

Example 2:

$$(\mathrm{e}^x u_x)_x + (\mathrm{e}^y u_y)_y = f(x, y) , \qquad u(x, y) = (xy)^3 , \qquad f(x, y) = 3xy\left((2 + x)\, y^2\,\mathrm{e}^x + x^2(2 + y)\,\mathrm{e}^y\right) ,$$

$$\varrho = \varrho(D_{xx}) = \varrho(D_{yy}) = \frac{2}{h^2}(\mathrm{e} + 1) , \qquad \delta = \delta(D_{xx}) = \delta(D_{yy}) = \frac{\pi^2}{2}(\mathrm{e} + 1) .$$

Example 3:

$$(\mathrm{e}^{-xy} u_x)_x + (\mathrm{e}^{xy} u_y)_y - (x + y)\, u = f(x, y) ,$$

$$u(x, y) = (xy)^3 , \qquad f(x, y) = 3xy^3(2 - xy)\,\mathrm{e}^{-xy} + 3x^3y(2 + xy)\,\mathrm{e}^{xy} - (x + y)(xy)^3 ,$$

$$\varrho_1 = \varrho(D_{xx}) = \frac{1}{\mathrm{e}} \cdot \frac{2}{h^2}(\mathrm{e} + 1) + \frac{1}{2}, \qquad \delta_1 = \delta(D_{xx}) = \frac{1}{\mathrm{e}}\frac{\pi^2}{2}(\mathrm{e} + 1) + \frac{1}{2},$$

$$\varrho_2 = \varrho(D_{yy}) = \frac{2}{h^2}(\mathrm{e} + 1) + \frac{1}{2}, \qquad \delta_2 = \delta(D_{yy}) = \frac{\pi^2}{2}(\mathrm{e} + 1) + \frac{1}{2}.$$

Example 4:

$$(e^u u_x)_x + (e^u u_y)_y - w(x, y, u) = 0\,, \qquad u(x, y) = (xy)^2\,, \qquad w(x, y, u) = 2(x^2 + y^2)(1 + 2x^2y^2)\, e^u\,.$$

Note that the matrices D_{xx} and D_{yy} commute for the first two examples but not for the third one. Note that Example 4 is a nonlinear problem. Like the ADI scheme, the SADI scheme can be applied to nonlinear problems in a straightforward manner. We have included this example, in order to show the power of the residual smoothing technique.

Consider the first three examples. First we present results for $h = \frac{1}{20}, \frac{1}{40}, \frac{1}{80}$ obtained with the ν-values derived in Section 4. The results are collected in Table 2, which contains the following values: the total number of iterations n_0, the average reduction factor $\bar{r}$ defined by $\bar{r} := r(n_0)^{1/n_0}$ (cf. (5.1)) and the computing time (CT) in seconds needed for the iteration process. For the tolerance we take TOL $= 10^{-8}$; similar results are obtained for larger values of TOL. From Table 2 we see that, especially on the finer grids, the SADI scheme needs much less iterations than the ADI scheme, which results in a considerable reduction of CT.

Next we present results obtained on a 40*40 grid for several ν-values, with the purpose of testing the ν-parameter values derived in Section 4. Case 1 ($\varrho = \varrho(D_{xx}) = \varrho(D_{yy})$, $\delta = \delta(D_{xx}) = \delta(D_{yy})$) applies to the first two examples. Instead of ν, consider for these two examples the scaled parameter $\omega = \nu/\varrho$. One can readily see that $\omega^* = \frac{\nu^*}{\varrho} = \left(\frac{\delta}{\varrho}\right)^{1/2} = 0.039269908$ for the ADI scheme and $\omega^* = \frac{\delta}{\varrho} = 0.001542126$ for the SADI scheme. Case 2 ($\varrho_1 = \varrho(D_{xx}) \neq \varrho_2 = \varrho(D_{yy})$, $\delta_1 = \delta(D_{xx}) \neq \delta_2 = \delta(D_{yy})$) applies to Example 3. Let in this case $\omega := \nu/\varrho_1$, then one can easily see that for the ADI scheme $\omega^* = \left(\frac{\delta_1}{\varrho_1}\right)^{1/2} = 0.040696$. Since $\varrho_2 \approx e\,\varrho_1$ and $\delta_2 \approx e\,\delta_1$, it is obvious to choose $\nu_1 = e\,\nu$ and $\nu_2 = \nu$ for the SADI scheme. The ω^*-value is then given by $\omega^* = \frac{\delta_1}{\varrho_1} = 0.001656164$. The number of iterations, for TOL $= 10^{-8}$, are presented in Table 3. We may conclude that the parameter values derived in Section 4 are fairly good since the corresponding number of iterations is nearly minimal. Furthermore, we see that in the range $10^{-3} \leqq \omega \leqq 10^{-2}$, the SADI scheme is less sensitive to the choice of the parameter values than the ADI scheme. Thus, an ω-value which differs a little from the ω^*-value can lead to considerably extra computing time for the ADI scheme, but not so for the SADI scheme.

Consider Example 4. Application of the ADI scheme or the SADI scheme to this nonlinear problem requires at each iteration the solution of a set of nonlinear tridiagonal systems, for which we use Newton iteration. Results

Table 2. The n_0-, $\bar{r}$- and CT-values for the first three examples

ADI

h^{-1}	example 1			example 2			example 3		
	n_0	$\bar{r}$	CT	n_0	$\bar{r}$	CT	n_0	$\bar{r}$	CT
20	58	0.73	0.702	67	0.76	1.263	76	0.78	1.397
40	116	0.85	5.301	138	0.87	11.069	155	0.89	11.042
80	231	0.92	41.196	279	0.94	76.486	312	0.94	86.092

SADI

h^{-1}	example 1			example 2			example 3		
	n_0	$\bar{r}$	CT	n_0	$\bar{r}$	CT	n_0	$\bar{r}$	CT
20	18	0.33	0.369	21	0.42	0.512	26	0.49	0.747
40	21	0.40	1.781	27	0.49	3.306	34	0.58	4.080
80	25	0.45	9.219	31	0.55	15.490	43	0.64	17.712

Table 3. The n_0-values for $h = \frac{1}{40}$ and various values of ω, for the first three examples

ω	example 1		example 2		example 3	
	ADI	SADI	ADI	SADI	ADI	SADI
$5^* 10^{-2}$	147	200	143	188	166	159
10^{-2}	100	41	267	39	220	34
$5^* 10^{-3}$	199	21	>500	26	440	31
10^{-3}	>500	22	>500	27	>500	37
ω^*	116	21	138	27	155	34

Table 4. The n_0-, $\bar{r}$- and CT-values for Example 4

h^{-1}	ADI			SADI		
	n_0	$\bar{r}$	CT	n_0	$\bar{r}$	CT
20	27	0.71	13.013	12	0.45	5.961
40	95	0.91	194.378	14	0.51	30.030

for $h = \frac{1}{20}, \frac{1}{40}$ and for TOL $= 10^{-4}$ are presented in Table 4. The best ω-values are experimentally found to be $\omega^* = 10^{-1}$ for the ADI scheme and $\omega^* = 10^{-2}$ for the SADI scheme. From this table we see that residual smoothing leads to a considerable saving of the number of iterations and hence also of the computing time. Note that in this case the gain in computing time is even more than for the first three examples, since one ADI iteration is now very expensive compared to the computation of the smoothing matrices.

6. An alternative smoothed ADI scheme

In this section we briefly consider an alternative to the SADI scheme (3.3). For this purpose, we rewrite the ADI scheme (3.2) in the one-stage form

$$(D_{xx} - v_1 I)\,(D_{yy} - v_2 I)\,U^{n+1} = (D_{xx} - v_1 I)\,(D_{yy} - v_2 I)\,U^n + (v_1 + v_2)\,(AU^n - B)\,. \tag{6.1}$$

The idea is now to multiply the residual in (6.1) by the smoothing matrices $\tilde{S}_x$ and $\tilde{S}_y$ (see Section 3):

$$(D_{xx} - v_1 I)\,(D_{yy} - v_2 I)\,U^{n+1} = (D_{xx} - v_1 I)\,(D_{yy} - v_2 I)\,U^n + (v_1 + v_2)\,\tilde{S}_y \tilde{S}_x (AU^n - B)\,. \tag{6.2}$$

For brevity, we restrict ourselves to case 1 and assume that $v_1 = v_2 = v$. The damping factor of scheme (6.2), as a function of μ_x and μ_y can then be written as

$$\xi = \xi(\mu_x, \mu_y; \omega) = 1 + \frac{2\omega(\mu_x + \mu_y)}{(\mu_x - \omega)\,(\mu_y - \omega)} \cdot P_k(\mu_x)\,P_k(\mu_y)\,, \tag{6.3}$$

where $\omega = v/\varrho$. The corresponding average damping factor α is then given by (3.7) with $\xi = \xi_q(\mu_x, \mu_y; \omega_q)$ defined in (6.3). In order to damp the high frequency error components, we choose $\omega_0 = 1$ and $\omega_q = \omega$ for $q = 1(1)\,N - 1$ (see Section 3). The average damping factor $\alpha = \alpha(\mu_x, \mu_y; \omega)$ is plotted in Fig. 2 for $\mu_x = \mu_y$, $N = 6$ and $\omega = 1, 10^{-1}, 10^{-2}, 10^{-3}$. Comparing Fig. 1 and Fig. 2 it is apparent that the SADI scheme gives a much better "overall" damping of the iteration error than the alternative scheme.

As an illustration, we apply the alternative scheme (6.2) to Example 1 for $h = \frac{1}{40}$ and for various values of the parameter ω. For the tolerance TOL we take TOL $= 10^{-8}$. The results are presented in Table 5. From Table 3 and Table 5 one can readily see that scheme (6.2) is slightly faster than the ADI scheme, however, much slower than the SADI scheme. Thus, the SADI scheme is clearly to be preferred to the alternative scheme (6.6).

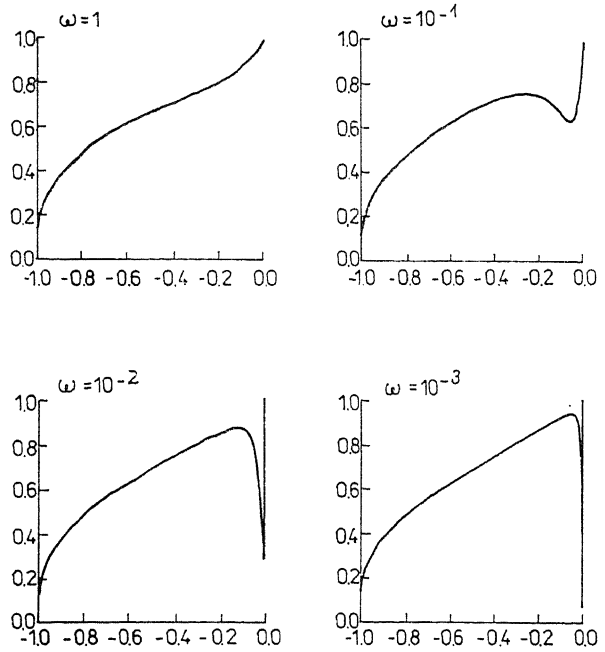


Fig. 2. The average damping factor for scheme (6.2) for $\omega = 1, 10^{-1}$ $10^{-2}, 10^{-3}$

Table 5. n_0-values for $h = \frac{1}{40}$ and various ω-values for Example 1

ω	$5^* 10^{-2}$	10^{-2}	$5^* 10^{-3}$	10^{-3}
n_0	219	79	105	229

7. Concluding remarks

In this paper we considered residual smoothing as a means to accelerate the convergence of the ADI scheme for elliptic difference equations. Concerning this technique we note the following.

(i) Residual smoothing can be easily applied to general elliptic problems, even to nonlinear problems, to speed up iterative methods such as the ADI method.

(ii) For a proper choice of the degree of smoothing k ($k = 2^q - 1$ for some integer $q \geqq 0$), residual smoothing can be implemented very efficiently.

(iii) Residual smoothing can be combined with the ADI scheme in several ways. When it is applied in the right way, as is done for the SADI scheme (3.3), residual smoothing can lead to a considerable reduction of the number of iterations and the computing time for the ADI scheme.

(iv) The parameters for the SADI scheme are chosen such that the high- and low-frequency components in the iteration error are rapidly damped. Due to the residual smoothing, the other components in the error are also properly damped.

(v) For a certain range of the parameter values, the SADI scheme is much less sensitive to the choice of these values that the ADI scheme.

References

1 BIRKHOFF, G.; LYNCH, R. E.: Numerical solution of elliptic problems. SIAM, Philadelphia 1984.
2 HOUWEN, P. J. VAN DER; BOON, C.; WUBS, F. W.: Analysis of smoothing matrices for the preconditioning of elliptic difference equations. Z. angew. Math. Mech. 68 (1988) 1, 3—10.
3 PEACEMAN, D.W.; RACHFORD, H. H.: The numerical solution of parabolic and elliptic differential equations. J. Soc. Ind. Appl. Math. 3 (1955), 28—41.
4 RICE, J. R.; BOISVERT, R. F.: Solving elliptic problems using ELLPACK. Springer-Verlag, New York 1984.
5 YOUNG, D. M.: Iterative solution of large linear systems. Academic Press, New York 1971.

Received November 20, 1987

Address: J. H. M. TEN THIJE BOONKKAMP, Centre for Mathematics and Computer Science, P.O. Box 4079, NL-1009 AB Amsterdam, The Netherlands