THEORETICAL AND PRACTICAL ASPECTS
OF A MULTIGRID METHOD*

P. WESSELING

Abstract. A multigrid method is described. A novel item is the use of incomplete LU decomposition
for smoothing. Numerical experiments show that its speed and robustness compare favorably with other
multigrid methods. A fairly simple rate of convergence proof is presented.

Key words. multigrid methods, numerical methods for elliptic equations

1. Introduction. A multigrid method will be presented. Numerical experiments
show that this method is faster than other multigrid methods for which numerical
experiments had been reported in sufficient detail when this paper was written (Brandt
(1977), Hackbusch (1978), Nicolaides (1979)). The method to be described is also
robust in the sense that it does not need to be adapted to the problem at hand, does
not require "tuning", and converges fast for a large variety of problems.

A difficulty with the development of multigrid methods is that there are many
ways in which the basic ideas underlying these methods can be implemented. The
way in which this is done has great influence on the speed and robustness of the
method obtained. In order to justify as far as possible some of the choices made here
(namely, incomplete LU smoothing and Galerkin approximation on coarser grids)
some theoretical background material will be presented, including a rate of conver-
gence proof, and some comparative experiments are described.

It is assumed that the reader has some familiarity with multigrid methods (cf.
Brandt (1977)).

2. Description of the method. The method to be presented is used to solve the
algebraic system that results from finite difference approximation of the following
elliptic partial differential equation, denoted in Cartesian tensor notation as follows:

\[(a_{ij} u_{ij})_{ij} - (b_{i} u)_{i} + cu = f,\]

with \(a_{ij}, b_{i}, c, f\) and \(u\) functions of two variables \(x_{1}, x_{2}\) with \((x_{1}, x_{2}) \in \Omega = (0, 1) \times (0, 1)\).

A computational grid \(\Omega^{l}\) and a corresponding set of grid functions \(U^{l}\) are defined by

\[\Omega^{l} = \{(x_{1}, x_{2}) | x_{1} = m_{i} \cdot 2^{-l}, m_{i} = 0(1)2^{l}\}, \quad U^{l} = \{u^{l} : \Omega^{l} \to \mathbb{R}\}.\]

Equation (2.1) is approximated by a finite difference scheme, for example,

\[\frac{3}{2} (\nabla [a_{ij} \Delta_{i}] + \Delta_{i} a_{ij} \nabla) u^{l} - \frac{1}{2} (\nabla^{l} + \Delta_{i}) (b_{i} u^{l}) + cu^{l} = f^{l},\]

with \(u^{l}, f^{l} \in U^{l}\). The operators \(\nabla^{l}\) and \(\Delta^{l}\) are backward and forward difference operators
in the \(x_{i}\)-direction. In singular perturbation problems one may wish to approximate
\(b_{i} u_{i}\) by one-sided differences; this does not affect the performance of the multigrid
method to be described. The linear algebraic system to which the difference scheme
together with the (as yet unspecified) boundary conditions give rise will be denoted by

\[A^{l} u^{l} = f^{l},\]

with \(A^{l} : U^{l} \to U^{l}\).

* Received by the editors October 30, 1980.
† Department of Mathematics, Delft University of Technology, Julianalaan 132, Delft,
the Netherlands.

387
The multigrid method makes use of a hierarchy of computational grids \( \Omega^k \) and corresponding sets of grid functions \( U^k, k = l-1(-1)1 \), defined by (2.2) with \( l \) replaced by \( k \). The step size on \( \Omega^k \) is \( 2^{-k} \), and as \( k \) decreases \( \Omega^k \) gets coarser. On the coarser grids (2.4) is approximated by

\[
A^k u^k = f^k, \quad k = l-1(-1)1.
\]

\( A^k \) and \( f^k \) can be chosen in many ways. Several alternatives will be discussed in the sequel. The choice that is made has a significant influence on the performance of the multigrid method.

Furthermore, let there be given restriction operators \( r^k \) and prolongation operators \( p^k \),

\[
r^k: U^k \to U^{k-1}, \quad p^k: U^{k-1} \to U^k.
\]

Examples will follow.

With the foregoing definitions, multigrid methods for linear partial differential equations can be described in general by the following quasi-Algor program:

**Algorithm 1.**

**Procedure** multigrid method \((k)\), value \( k, \) integer \( k \);  
begin integer \( n \);  
if \( k = 1 \) then  
begin for \( n := 1(1) sa[k] + sb[k] \) do \( S(k, u^k, A^k, f^k); \) end else  
begin for \( n := 1(1) sa[k] \) do \( S(k, u^k, A^k, f^k); \)  
\( f^{k-1} := r^k(f^k - A^k u^k); u^{k-1} := 0; \)  
for \( n := 1(1) sc[k] \) do multigrid method \((k - 1)\);  
\( u^k := u^k + p^k u^{k-1}; \)  
for \( n := 1(1) sb[k] \) do \( S(k, u^k, A^k, f^k) \)  
end  
end multigrid method;

Here \( S \), to be called the smoothing operator, is some iterative method with a smoothing effect on the error. The choice of \( S \) has an important influence on the efficiency of the multigrid method. Various possibilities will be discussed. Note that \( f^{k-1} \) is a coarse grid approximation, not to \( f^k \), but to \( f^k - A^k u^k \). The following statements,

initialize \( u^l \);  
for \( n := 1(1) N \) do multigrid method \((l)\);  
result in the execution of \( N \) multigrid iterations. By special choices of the integer arrays \( sa, sb, sc[k] \), every multigrid strategy that has been proposed for linear problems in the literature can be recovered.

For the prolongation operator \( p^k \) the following two possibilities will be considered, both exact only for linear functions:

**7-point prolongation:**

\[
(p^k u^{k-1})_{2s,2t} = u^{k-1}_{s,t}, \quad (p^k u^{k-1})_{2s+1,2t} = \frac{1}{2}(u_{2s}^{k-1} + u^{k-1}_{2s+1,t}),
\]

\[
(p^k u^{k-1})_{2s,2t+1} = \frac{1}{2}(u_{2s}^{k-1} + u^{k-1}_{s,t+1}),
\]

\[
(p^k u^{k-1})_{2s+1,2t+1} = \frac{1}{2}(u_{2s+1}^{k-1} + u^{k-1}_{s,t+1})
\]

where the value of the grid function \( u^k \) in the grid point with coordinates \((s \cdot 2^{-k}, t \cdot 2^{-k})\) is denoted by \( u^k_{st} \).
9-point prolongation: Like 7-point prolongation, except
\[ (p^k u^{k-1})_{2+1,2+1} = \frac{1}{4}(u_{6+1}^{k-1} + u_{5+1}^{k-1} + u_{4+1}^{k-1} + u_{3+1}^{k-1}). \]

The following three options will be studied for the restriction operator:

\[ (r^k u^k)_n = u_{2n}^k \]

7-point restriction:
\[ (r^k u^k)_n = \frac{1}{4} (u_{2+1}^k + u_{1+1}^k + u_{2+1}^{k+1} + u_{1+1}^{k+1} + u_{2+1+1}^k + u_{1+1+1}^k + u_{2+1+1}^{k+1} + u_{1+1+1}^{k+1}). \]

9-point restriction:
\[ (r^k u^k)_n = \frac{1}{4} (u_{2+1}^k + u_{1+1}^k + u_{2+1}^{k+1} + u_{1+1}^{k+1} + u_{2+1+1}^k + u_{1+1+1}^k + u_{2+1+1}^{k+1} + u_{1+1+1}^{k+1}). \]

On \( U^k \) the following inner product is defined:
\[ (u^k, v^k)_k = 4^{-k} \sum_{n} u_n^k v_n^k. \]

Between \( p^k \) and \( r^k \) we have the following special relation, for (2.7) and (2.10) or for (2.8) and 2.11):
\[ r^k = (p^k)^T, \]
i.e.,
\[ (r^k u^k, v^{k-1})_{k-1} = (u^k, p^k v^{k-1})_k \quad \forall u^k \in U^k, \quad \forall v^{k-1} \in U^{k-1}. \]

For \( r^k \) one could think of many possible weighted averages of neighboring function values; it is useful that (2.13) selects a few distinguished cases, which will turn out to be advantageous both theoretically and practically. The operators (2.7) and (2.10) have not yet been discussed in the literature; we will find them to be preferable to the other possibilities.

For the operators \( A^k (k < l) \) two possibilities will be studied:

\[ \text{finite difference approximation}, \]
for example, (2.3) with \( l \) replaced by \( k \); and

\[ \text{Galerkin approximation} \]
\[ A^{k-1} = (p^k)^T A^k p^k, \quad k = l(1-1/2). \]

The reason for the appellation “Galerkin approximation” will become clear in the sequel. Equation (2.15) has been used by Fedorenko (1962), (1964), Bakhvalov (1966) and Brandt (1977), (1979); equation (2.16) has been used by Frederickson (1975), Hackbusch (1978), Wesseling and Sonneveld (1980) and Wesseling (1980).

An important property of (2.16) is that if \( A^l \) is a 7-point operator (i.e., corresponding to a 7-point molecule), as for (2.3) for example, and \( p^k \) is defined by (2.7) or (2.8), then \( A^k (k < l) \) are 7- or 9-point operators respectively (cf. Fig. 2.1).

A discussion of the smoothing operator \( S \) is deferred to § 4.

In order to provide some insight into the question of which of the various options outlined above is preferable, some theoretical framework is developed in the next two sections.
3. Computational complexity of multigrid methods. First, a bound will be derived for the computational complexity of one multigrid iteration as defined by the procedure in the preceding section. Let $\max_k (sa[k], sb[k])$ be independent of $l$. The total computational complexity of the operations $S$ and the computation of $f^{k-1}$ and $u^k + p^k u^{k-1}$ for one execution of the statement: multigrid method ($k$) can be bounded by $a \cdot 4^k$, where $a$ is some constant. Let $\max_k (sc[k]) \equiv \sigma$. Let the total computational complexity of one execution of the statement: multigrid method ($k$) be denoted by $W_k$; then

$$W_k \leq \sigma W_{k-1} + a \cdot 4^k.$$  

We is bounded from above by the solution of the following difference equation:

$$z_k = \sigma z_{k-1} + a \cdot 4^k.$$

Hence

$$W_l \leq W_i \sigma^{l-i} + a \left( \frac{4^{l-i} - \sigma^{l-i}}{4 - \sigma} \right), \quad \sigma \neq 4,$$

(3.2)

$$W_l \leq 4^l \frac{W_i}{4 + a (l - 1)} \quad \sigma = 4.$$  

Therefore $W_l = O(4^l), \sigma < 4$; $W_l = O(l 4^l), \sigma = 4$; $W_l = O(\sigma^l), \sigma > 4$.

The overall computational complexity depends on the accuracy required and on the rate of convergence of the multigrid method. The following authors have shown theoretically that for various special cases various specific multigrid algorithms have a rate of convergence that is independent of $l$ (eq. numbers are for this paper):

- Fedorenko (1964): Poisson equation; $A^k$: (2.15); $p^k$: (2.8); $r^k$: (2.9).
- Bakhvalov (1966): (2.3); $A^k$: (2.15); $p^k$: must be exact for second degree polynomials; $r^k$: (2.9).
- Frederickson (1975): Poisson equation; $A^k$: (2.16); $p^k$: (2.8); $r^k$: (2.11).
- Wesseling (1980): (2.3); $A^k$: (2.16); $p^k$: (2.8); $r^k$: (2.11).

A very general rate of convergence proof in an abstract setting admitting arbitrary regions and elliptic equations of higher order has been given Hackbusch (1980).

The rate of convergence proofs referred to above are rather involved and technical. Therefore it seems useful to present here an elementary proof for the selfadjoint and positive definite case (i.e., (2.3) with $b = c = 0$), using only arguments that can be extended to the general case. It is assumed that $A^k$ is given by (2.16); for (2.15) the proof would seem to be more difficult, as evidenced by the use of higher order interpolation in the theory of Bakhvalov (1966). A homogeneous Dirichlet boundary condition is assumed.

The most difficult part of the proof is to estimate how well $A^{k-1}$ approximates $A^k$, i.e., to estimate $u^k - p^k u^{k-1}$ with $A^k u^k = f^k$ and $A^{k-1} u^{k-1} = r^k$. We proceed to do this.

Define

$$V^k \equiv \{ u^k : \Omega^k \to \mathbb{R} | u^k |\partial \Omega^k = 0 \}.$$  

(3.3)
Whenever in some formula a function value on \( \partial \Omega^k \) or outside \( \Omega^k \) is needed, this value is taken as zero. When there is no danger of confusion the indices \( k \) and \( l \) will be deleted for brevity. On \( V \) the following norms are defined:

\[
\|u\|_0 = (u, u), \quad \|u\|^2_2 = (\nabla u, \nabla u), \quad \|u\|^2_\Delta = (\nabla_1 \Delta u, \nabla_1 \Delta u),
\]

(3.4)

\[
\|u\|_{-1} = \sup_{\|v\|_{-1} = 1} (u, v),
\]

where the summation convention applies. These quantities are indeed norms because of the boundary condition, and are analogous to Sobolev norms. Without mentioning this explicitly, we will often use the following partial summation formula:

\[
(u, \nabla v) = -(\Delta u, v).
\]

The following well-known inequalities hold:

\[
\|u\|_{-1} \leq \|u\|_0 \leq \|u\|_1 \leq \|u\|_2.
\]

Furthermore, it is easy to derive that

\[
\|u_i^k\|_{l,k} \leq 2^{2k+1} \|u_i^k\|_{l-1,k}, \quad i = 1, 2.
\]

The prolongation and restriction operators \( p^k \) and \( r^k \) are to have the following properties. Define \( p^{m-k} = p^m p^{m-1} \cdots p^{k+1} \).

\[
C_i \|u_i^k\|_{1,k} \leq \|p^m u_i^k\|_{1,m} \leq \|u_i^k\|_{1,k}, \quad m > k,
\]

(3.8a)

\[
\|u_i^k - p^k r^k u_i^k\|_{1,k} \leq C 2^{-k} \|u_i^k\|_{2,k},
\]

(3.8b)

\[
C_3 \|u_i^{k-1}\|_{0,k-1} \leq \|p^{k-1} u_i^{k-1}\|_{0,k} \leq \|u_i^{k-1}\|_{0,k-1},
\]

(3.8c)

\[
\|u_i^k - r^{k+1} p^{k+1} u_i^k\|_{-1,k} \leq C_4 4^{-k} \|u_i^k\|_{1,k},
\]

(3.8d)

\[
\|r^k u_i^k\|_{2,k-1} \leq \|u_i^k\|_{2,k}.
\]

(3.8e)

\( C_i \) denote positive constants independent of \( k \) and \( l \).

Define \( B(u, v) = (Au, v) \). Assume uniform ellipticity and boundedness for (2.3):

\[
C_5 \xi_i \xi_j \leq a_{ij} \xi_i \xi_j, \quad a_{ij} = a_{ji}, \quad \forall \xi_i \in \mathbb{R}, \quad \forall x \in \Omega.
\]

(3.9)

We have

\[
B'(u', v') = \frac{1}{2}(a_{ij} \Delta u_i \Delta v_j) + \frac{1}{2}(a_{ij} \nabla u_i \nabla v_j).
\]

(3.10)

Hence

\[
B'(u', u') = B(u', u'),
\]

(3.11)

\[
B'(u', u') \geq C_6 \|u\|^2_{1,0}.
\]

(3.12)

Furthermore, with the inequality of Schwarz,

\[
(a_{ij} \Delta u_i \Delta v_j) \leq \sup_{t \in \Omega} |a_{ij}| \sum_{t_j} \|\Delta u_j\|_{0,j} \|\Delta v_j\|_{0,j} \leq 2C_6 \|u\|_{1,0} \|v\|_{1,0},
\]

(3.13)

since from (3.9): \( \sup_{t \in \Omega} |a_{ij}| \leq C_6 \). It follows that

\[
|B'(u', v')| \leq 2C_6 \|u\|^2_{1,0} \|v\|^2_{1,0}.
\]

(3.14)

It will be shown that if \( A^k \) is constructed according to (2.16) and if \( p^k \) satisfies (3.8a), then \( B^k(u^k, v^k) \) also satisfies (3.11), (3.12) and (3.14) (with other constants).
We have \( B^{k-1}(u^{k-1}, v^{k-1}) = ((p^k) A^k p^k u^{k-1}, v^{k-1})_{k-1} = (A^k p^k u^{k-1}, p^k v^{k-1})_k = B^k(p^k u^{k-1}, p^k v^{k-1}) \). Hence:

\[
B^k(u^k, v^k) = B^k(p^k u^k, p^k v^k). 
\]

With (3.8a) it follows that for \( k = 1(1) l \)
\[
B^k(u^k, v^k) = B^k(v^k, u^k), 
\]
\[
B^k(u^k, u^k) \geq C_1^2 C_3 \| u^k \|^2_{1,k}, 
\]
\[
\| B^k(u^k, v^k) \| \leq 2 C_0 \| u^k \|_{1,k} \| v^k \|_{1,k}.
\]

We are now in a position to show that \( A^k \) is a Galerkin approximation to \( A^m \), \( m > k \), and to derive error estimates. Define, for arbitrary (deleted) \( k \),

\[
a(u, v) = B(u, v) - 2(b, v) \quad \forall u, v \in V, \quad b \in V \text{ fixed}.
\]

In the standard way it can be shown that \( A u = b \) and \( a(u, u) = \inf_{v \in V} a(v, v) \) are equivalent. Furthermore, \( a^k(u^k, v^k) = a^m(p^m u^k, p^m v^k) \), \( m > k \). Let \( A^m u^m = b^m \) and \( A^k u^k = (p^m)^T b^m \). We have

\[
a^k(u^k, u^k) = \inf_{v^k \in V^k} a^k(v^k, v^k) = \inf_{v^k \in V^k} a^m(p^m v^k, p^m v^k).
\]

Hence, the designation of (2.16) as a Galerkin approximation is justified. Using (3.16)–(3.18), we derive the following result similarly to its equivalent in finite element theory:

\[
\| u^m - p^m u^k \|_{1,k} \leq D_1 \inf_{w^k \in V^k} \| u^m - p^m w^k \|_{1,k}
\]

with \( D_1 = (2 C_0 / C_1^2 C_2)^{1/2} \). This estimate is not quite what we need, because in the rate of convergence proof an estimate of the following kind will be required: \( \| u^k - p^k u^k \|_{1,k} = O(4^{-k}) \). This can be obtained as follows, using what is known in the finite element literature as Nitsche’s trick; and specializing (3.21) to the case \( m = k \), \( k = m - 1 \). Define \( w^k \in V^k \) by

\[
A^k w^k = u^k - p^k u^k - 1; 
\]

then \( \| u^k - p^k u^{k-1} \|_{0,k} \leq B^k(w^k, u^k - p^k u^{k-1}) \). Since \( B^k(p^k u^{k-1}, p^k v^{k-1}) = (k, p^k v^{k-1})_k \) for all \( v^{k-1} \in V^{k-1} \) and \( B^k(u^k, v^k) = (k, k)_k \) for all \( v^k \in V^k \), we have \( B^k(u^k - p^k u^{k-1}, p^k v^{k-1}) = 0 \). Hence

\[
\| u^k - p^k u^{k-1} \|_{0,k}^2 = B^k(w^k - p^k u^{k-1}, u^k - p^k u^{k-1}) \quad \forall v^{k-1} \in V^{k-1}. 
\]

Using (3.18) and (3.21), we obtain

\[
\| u^k - p^k u^{k-1} \|_{0,k} \leq 2 C_0 D_1 \| w^k - p^k u^{k-1} \|_{1,k} \inf_{v^{k-1} \in V^{k-1}} \| u^k - p^k v^{k-1} \|_{1,k}, 
\]

for all \( v^{k-1} \in V^{k-1} \). Choosing \( v^{k-1} = r^k w^k, z^{k-1} = r^k u^k \), we find with (3.8b)

\[
\| u^k - p^k u^{k-1} \|_{0,k}^2 \leq 2 C_0 D_1 C_2 - 2^{-k} \| w^k \|_{2,k} \| u^k \|_{2,k}. 
\]

In Appendix A it will be shown that

\[
\| u^k \|_{2,k} \leq D_2 \| u^k \|_{0,k} \quad \forall v^k \in V^k. 
\]

Using (3.26) and (3.22) we finally obtain

\[
\| u^k - p^k u^{k-1} \|_{0,k} \leq D_2 \| u^k \|_{0,k} D_3 - 2^{-k} \| u^k \|_{0,k} \quad D_3 = 2 C_0 D_1 C_2 D_2. 
\]
ASPECTS OF A MULTIGRID METHOD

This estimate of the quality of the coarse grid approximations is one ingredient needed in the rate of convergence proof to be presented. We also need some information concerning the smoothing operator $S$.

Using (3.7) and (3.18), we have

$$\frac{(A^k u^k, u^k)}{(u^k, u^k)} \leq D_4 \cdot 4^k, \quad D_4 = 16 C_6.$$  \hspace{1cm} (3.28)

Hence

$$\lambda(A^k) \in (0, D_4 \cdot 4^k).$$  \hspace{1cm} (3.29)

For purposes of illustration only, the smoothing operator $S$ is chosen as follows (Jacobi iteration):

$$u^{k,v+1} = u^{k,v} - \alpha (A^k u^{k,v} - f^k),$$  \hspace{1cm} (3.30)

with $\alpha$ a parameter to be specified shortly.

The rate of convergence of the multigrid method of § 2 will be estimated for the special case $\sigma[k] = 0$, $s_{b[k]} = m$, $s_{c[k]} = \sigma$. Numerical experiments to be described show that this is a profitable multigrid strategy. More generality would merely add to the technicalities of the arguments to be presented.

Let $u^{k,0}$, $u^{k,1/2}$ and $u^{k,1}$ denote the iterand just before the execution of the statement, multigrid method $(k)$, just before the $s_{b[k]} = m$ executions of the smoothing operation $S$, and just after the execution of multigrid method $(k)$, respectively. Define

$$e^{k,\mu} = u^{k,\mu} - u^k, \quad u^k = (A^k)^{-1} f^k, \quad \mu = 0, \frac{1}{2}, 1.$$  \hspace{1cm} (3.31)

Then

$$e^{k,1} = (I^k - \alpha A^k)^m e^{k,1/2},$$  \hspace{1cm} (3.32)

with $I^k$ the identity operator. Define

$$V_1^{k,\gamma} = \text{span of eigenvectors of } A^k \text{ belonging to eigenvalues } \varepsilon \in (0, \gamma D_4 \cdot 4^k), \quad 0 \leq \gamma < 1,$$  \hspace{1cm} (3.33)

$$V_2^{k,\gamma} = \text{orthogonal complement of } V_1^{k,\gamma} \text{ in } V^k.$$  \hspace{1cm} (3.34)

Let $e_i^{k,\mu} = e_i^{k,0} + e_i^{k,\mu}$ with $e_i^{k,\mu} \in V_i^{k,\gamma}$, $i = 1, 2$. Choose

$$\alpha = \frac{4^{-k}}{D_4},$$  \hspace{1cm} (3.35)

then

$$\|e_1^{k,1}\|_{0,k} \leq \|e_1^{k,1/2}\|_{0,k}, \quad \|e_2^{k,1}\|_{0,k} \leq (1 - \gamma)^m \|e_2^{k,1/2}\|_{0,k}.$$  \hspace{1cm} (3.36)

Because of (3.27) we have

$$\|e^{k,0} + p^{k} u^{k-1}\|_{0,k} \leq D_4 \cdot 4^{-k} \|A^k e^{k,0}\|_{0,k}.$$  \hspace{1cm} (3.37)

Let the result of the coarse grid correction (computed by $s_{c[k]} = \sigma$ executions of the statement multigrid method $(k)$) be denoted by $v^{k-1}$, and assume

$$\|v^{k-1} - u^{k-1}\|_{0,k-1} \leq \delta_{k-1} \|u^{k-1}\|_{0,k-1},$$  \hspace{1cm} (3.38)

with $\delta_{k-1}$ a constant to be determined later. Then

$$\|e^{k,1/2}\|_{0,k} \leq D_3 \cdot 4^{-k} \|A^{k,0}\|_{0,k} + \delta_{k+1} \|u^{k-1}\|_{0,k-1}.$$  \hspace{1cm} (3.39)
With (3.37) \( p^k u^{k-1} \) can be bounded, and with (3.8c) one obtains

\[
\| e^{k,1/2} \|_{\Omega,k} \leq D_3 \cdot 4^{-k} \left( \frac{1 + \delta_{k-1}}{C_3} \right) \| A^k e^{k,0} \|_{\Omega,k} + \frac{\delta_{k-1}}{C_3} \| e^{k,0} \|_{\Omega,k}.
\]

From (3.36) it follows that

\[
\| e^{k,1} \|_{\Omega,k} \leq (\xi + \eta \delta_{k-1}) \| e^{k,0} \|_{\Omega,k},
\]

with \( \xi = D_3(\gamma + (1 - \gamma)^m), \) \( \eta = (1 + \xi)/C_3. \) Hence

\[
\| e^{k-1} \|_{\Omega,k} \leq (\xi + \eta \delta_{k-1}) \| e^{k} \|_{\Omega,k},
\]

and therefore,

\[
\delta_{k} \leq (\xi + \eta \delta_{k-1})^\nu.
\]

In order to simplify this proof we assume \( \delta_1 = 0, \) i.e., on the coarsest grid \( \Omega^1 \) the solution is to be calculated exactly, rather than approximately as in the procedure multigrid method of \$2. \) Since \( \eta > 1 \) we will not be able to show that \( \delta_k < 1 \) if \( \sigma = 1, \) although the method works well for \( \sigma = 1, \) as we shall see. Choose \( \sigma > 1. \) Assume (induction) that \( \delta_{k-1} \leq \xi, \) and choose \( m \) and \( \gamma \) such that

\[
\gamma^{\sigma-1}(1 + \eta)^\nu < 1,
\]

(hence \( \xi < 1; \) then

\[
\delta_k \leq \xi.
\]

Thus we have proved the following theorem.

Theorem. Let the multigrid method of \$2 satisfy \( sa[k] = 0, sb[k] = m, sc[k] = \sigma > 1 \) and equations (2.16), (3.8) and (3.44), and let the smoothing operator \( S \) be defined by (3.30) and (3.35). Then one execution of the statement multigrid method (1) reduces the error by a factor \( \xi < 1, \) for equation (2.3) with \( b_i = c = 0 \) and \( a_i \) satisfying (3.9).

Because \( \xi \) is independent of \( l, \) the number of multigrid iterations required for a given precision is fixed. Choosing \( \sigma < 4 \) and taking (3.2) into account, we conclude that the computational complexity of the multigrid method is \( O(4^l). \)

As an example, we show that the conditions of the theorem, i.e., (3.8) and (3.44), are satisfied with \( p^k \) and \( r^k \) defined by (2.7), (2.8), (2.10), (2.11). The right-hand side of (3.8a) follows from the easily verifiable fact that \( \| \nabla u^k \| \| u^k \| \| u^{k-1} \| \| u^{k-1} \| \| u^k \| \| u^{k-1} \| \) is the left-hand side can be checked by employing the following identity, valid for 9-point interpolation:

\[
\| \nabla u^k \| \| u^k \| \| u^{k-1} \| \| u^{k-1} \| \leq 2^{k-m} \| \nabla u^k \| \| u^k \| \| u^{k-1} \| \| u^{k-1} \| \}
\]

with \( w^k = \nabla u^k, \) \( n = 2^{m-k-1}, \) and by using the inequality \( \Sigma (a_i + b_i)^2 \leq \Sigma (a_i^2 + b_i^2) - 2\Sigma a_i b_i \) for \( a_i^2 + b_i^2. \) The case of 7-point interpolation can be treated similarly. It is easy to write \( V_{\lambda} (u^k - p^k r^k u^k) \) as a linear combination of quantities of the type \( V_{\lambda} \Delta u^k, \) such that (3.8b) follows. The derivation of (3.8c) is straightforward. There are constants \( C_1 \) and \( C_2 \) such that \( u^k - r^k + p^k - u^k = C_1 \cdot 4^{-k} V_{\lambda} \Delta u^k + C_2 \cdot 16^{-k} V_{\lambda} \Delta u^k \) using (3.8d) is easy to derive. Equation (3.8e) is trivial. Finally, choose \( \gamma = z/(2D_3) \) and \( m > \log(z/(2D_3))/\log(1 - \gamma) \) with \( z \) to be determined; then \( \xi \leq z. \) Choose \( z = (1+2/C_3)^\nu; \) then (3.44) is satisfied.

4. Smoothing. This section is devoted to a discussion of possible smoothing operators (procedure \( S \) in Algorithm 1). A useful yardstick by which the merits of
various smoothing operators may be measured is the smoothing factor introduced by Brandt (1977). The error \( e^k \in U^k \) before application of a multigrid iteration can be represented by a Fourier series as follows (deleting the superscript \( k \)):

\[
e_{mn} = \sum_{s,t=-M}^M c_{st} \exp \left( i m \theta_s + i n \phi_t \right),
\]

with \( \theta_s = (2s - 1) \pi / 2M + 1, \phi_t = (2t - 1) \pi / (2M + 1), \quad M = 2^{k-1} \). For periodic boundary conditions and constant coefficients in the differential equation, many smoothing operators have the property that the error after application of the smoothing operator \( \hat{e}_{mn} \) satisfies

\[
\hat{e}_{mn} = \sum_{s,t=-M}^M \hat{c}_{st} \exp \left( i m \theta_s + i n \phi_t \right),
\]

with

\[
\hat{c}_{st} = \rho(\theta_s, \phi_t) c_{st}.
\]

The philosophy underlying multigrid methods requires that those Fourier components which cannot be resolved on coarser grids be eliminated as fast as possible. One therefore defines the smoothing factor \( \bar{\rho} \) as follows: (cf. Brandt (1977)):

\[
\bar{\rho} = \sup_{(\theta, \phi) \in G} \left| \rho(\theta, \phi) \right|, \quad G = \left\{ (\theta, \phi) \left| -\pi \leq \theta, \phi \leq \pi, |\theta| \leq \pi / 2 \text{ or } |\phi| \leq \pi / 2 \right\}.
\]

For convenience, \( G \) is not restricted to the discrete set of values occurring in (4.1) and (4.2). Of two smoothing operators, the one with the smaller \( \bar{\rho} \) (for a given amount of computational work) is judged to be the better of the two.

Two smoothing operators that are often used in multigrid methods are pointwise and line Gauss–Seidel iteration (Brandt (1977), (1979), Nicolaides (1979)). Hackbusch (1978) uses pointwise and line Gauss–Seidel iteration with a special ordering of the grid points. Wesseling and Sonneveld (1980) have introduced incomplete \( LU \) (ILU) decomposition as a smoothing operator for multigrid methods.

In order to describe \( ILU \) decomposition, some notation will be introduced. The operator \( A^k \) is denoted as follows, under the summation convention and deleting the superscript \( k \):

\[
(Au)_i = \sigma_{\alpha \beta} u_{i+\alpha,j+\beta},
\]

with \( \alpha, \beta \in \{-1, 0, 1\} \). If the coefficients in the differential equation are variable, one must keep in mind that \( \sigma_{\alpha \beta} \) depends on \( i \) and \( j \). The difference molecules that (4.3) may represent are depicted in Fig. 2.1.

Incomplete \( LU \) decomposition is defined as follows:

\[
(Lu)_i = \lambda_{\alpha \beta} u_{i+\alpha,j+\beta}, \quad (Uu)_i = \mu_{\alpha \beta} u_{i+\alpha,j+\beta},
\]

\[
LU = A + C, \quad (Cu)_i = \gamma_{\alpha \beta} u_{i+\alpha,j+\beta}.
\]

with the elements of the error matrix \( C \) zero in those places where the elements of \( A \) are nonzero. The \( ILU \) decompositions that will be used are fully defined in Table 4.1, which lists the values of \( (\alpha, \beta) \) for which \( \lambda_{\alpha \beta} \neq 0, \gamma_{\alpha \beta} \neq 0 \), for what will be called 5-point, 7-point and 9-point \( ILU \) decomposition. Furthermore, \( \mu_{\alpha \beta} \neq 0 \) if and only if \( \lambda_{-\alpha,-\beta} \neq 0 \).
$L$ and $U$ may be constructed by a standard LU-decomposition algorithm, with zero outside the nonzero patterns described above. A version in which $I_{ij} = u_{ij}$ may be computed as follows (cf. Wesseling and Sonneveld (1980)). $P$ denotes the nonzero pattern of the ILU decomposition. $A$ is an $N \times N$ matrix.

**Algorithm 2.**

$A^0 := A$

for $r := 1(1)N$ do

begin $a^r_{ii} := \text{sqrt}(a^r_{ii})$;

for $j > r \land (r, j) \in P$ do $a^r_{ij} := a^r_{ij} / a^r_{ii}$;

for $i > r \land (i, r) \in P$ do $a^r_{ij} := a^r_{ij} / a^r_{jj}$;

for $(i, j) \in P \land i > r \land j > r \land (i, r) \in P \land (r, j) \in P$ do $a_{ij} := a_{ij} - a_{ii} a_{jj}$

end ILU decomposition;

$A^N$ will contain $L$ and $U$.

For a practical smoothing factor, we will construct 5-point and 7-point ILU decompositions for the constant coefficient case. The 9-point ILU decomposition equals the 7-point decomposition, unless $A$ is a 9-point difference operator, which is never the case on the finest grid. Since the bulk of the computational work takes place on the finest grid, we will not study 9-point ILU decompositions.

In order to make the ILU decomposition unique, we (arbitrarily) require

\[(4.7) \quad \lambda_{00} = \mu_{00}\]

(as in Algorithm 2). The following equations result from (4.6):

\[
\begin{align*}
\sigma_{00} &= \lambda_{00} - 1 \mu_{00}, \\
\sigma_{10} &= \lambda_{00} - 1 \mu_{10} + \lambda_{10} - 1 \mu_{00}, \\
\sigma_{11} &= \lambda_{10} - 1 \mu_{01} + \lambda_{00} - 1 \mu_{10}, \\
\sigma_{01} &= \lambda_{00} - 1 \mu_{01}.
\end{align*}
\]

(4.8)

For the case of the Poisson equation, the system (4.8) is solved as follows. Let $L^T = U$, i.e., $\lambda_{ab} = \mu_{a-b}$. Eliminating $\mu_{01}$ with the first equation, $\mu_{-1,1}$ with the second and $\mu_{10}$ with the third, one finds, with $\mu = \mu_{00}$, that $\mu_{01} = -1/\mu$, $\mu_{-1,1} = \mu/(1 - \mu^2)$, $\mu_{10} = \mu^3/(1 - \mu^4)$, and

\[\mu^{12} - 4 \mu^{10} + 8 \mu^6 - 4 \mu^2 + 1 = 0.\]

In order to get $\gamma_{ab}$ as small as possible the largest root is preferred, which is found to be

\[(4.10) \quad \mu^2 = 3.294168413.\]

This defines the 7-point ILU decomposition for the Poisson equation. In the 5-point
case we have \( \mu_{-1,1} = 0 \), and the equations in (4.8) in which \( \sigma_{-1,1} \) and \( \sigma_{1,1} \) occur are dropped. We find \( \mu_{01} = \mu_{10} = -1/\mu \), and

\[
\mu^2 - 4\mu^2 + 2 = 0,
\]

of which the largest root is

\[
(4.12) \quad \mu^2 = 2 + \sqrt{2},
\]

The smoothing factors of these ILU decompositions are determined as follows. A smoothing operation is defined by

\[
(4.13) \quad x := x + (LU)^{-1}(f-Ax)
\]
or \((A + C)x := Cx + f\). One finds

\[
(4.14) \quad \rho(\theta, \phi) = \frac{\gamma_{1,-1} \cos (\theta - \phi)}{2 - \cos \theta - \cos \phi + \gamma_{1,-1} \cos (\phi - \theta)} \quad (5\text{-point ILU}),
\]

with \( \gamma_{1,-1} = \mu_{01} \mu_{10} = 1 - \sqrt{2} \),

\[
(4.15) \quad \rho(\theta, \phi) = \frac{2\gamma_{2,-1} \cos(2\theta - \phi)}{4 - 2 \cos \theta - \cos \phi + 2\gamma_{2,-1} \cos(2\theta - \phi)} \quad (7\text{-point ILU}),
\]

with \( \gamma_{2,-1} = \mu_{-1,1} \mu_{10} \approx 0.11181 \). By analytical means it follows that

\[
(4.16) \quad \bar{p} = \frac{\mu}{\sqrt{3 - \mu^2}} = 0.204 \quad (5\text{-point ILU})
\]

(cf. Hemker (1980)). For the 7-point case it has been found numerically that

\[
(4.17) \quad \bar{p} \approx 0.126 \quad (7\text{-point ILU}).
\]

For the Gauss-Seidel smoothing operators, Brandt (1977) has found the following smoothing factors:

\[
(4.18) \quad \bar{p} = \frac{1}{2} \quad (\text{pointwise Gauss-Seidel}),
\]

\[
(4.19) \quad \bar{p} = \frac{1}{\sqrt{2}} \approx 0.447 \quad (\text{line Gauss-Seidel}).
\]

The smoothing operators proposed by Hackbusch (1978) are not amenable to the Fourier analysis described above, so that smoothing factors cannot be defined.

We will roughly determine the computational complexity of the smoothing operators defined above by counting operations from the set \{+, -, \times, /, \sqrt{\cdot}\}. We will distinguish between the variable coefficient case and the case of the Poisson equation, to be denoted general and Poisson case respectively. Equation (4.13) is rewritten as follows:

\[
(4.20) \quad (LU)x := Cx + f.
\]

For the 5-point ILU decomposition we have \((Cx + f)_i = y_{i-1,i,j+1} + y_{i-1,i,j-1} + f_i\) (general), \((Cx + f)_i = y_{i-1,i,j+1} + y_{i+1,i,j-1} + f_i\) (Poisson), hence computation of \(Cx + f\) takes 4 or 3 operations per grid point respectively. For the 7-point ILU
decomposition the computational complexity of $Cx + f$ is the same. The solution of a system $Lu = y$ is computed as follows, for the 5-point ILU decomposition:

\begin{align}
\text{(4.21)} & \quad u_i^j := \frac{(y_i^j - \lambda_{0,0} u_{i-1,j} - \lambda_{-1,0} u_{i,j-1})}{\lambda_{0,0}} \quad \text{(general)}, \\
\text{(4.22)} & \quad u_i^j := \frac{(y_i^j - (u_{i-1,j} + u_{i,j-1}))}{\mu} \quad \text{(Poisson)},
\end{align}

and we count 5 or 4 operations per grid point in the general or Poisson case, respectively. Similarly for the 7-point ILU decomposition, it is found that $Lu = y$ takes 7 or 6 operations per grid point in the general or Poisson case, respectively.

The computational complexity of the Gauss–Seidel smoothing operators is as follows. The pointwise Gauss–Seidel smoothing operation is performed according to:

\begin{align}
\text{(4.23)} & \quad u_i^j := \left( f_i^j - \sigma_{0,0} u_{i,j-1} - \sigma_{-1,0} u_{i-1,j} - \sigma_{0,1} u_{i,j+1} - \sigma_{-1,1} u_{i-1,j+1} \right) / \sigma_{0,0} \quad \text{(general)}, \\
\text{(4.24)} & \quad u_i^j := \left( f_i^j + u_{i,j-1} + u_{i-1,j} + u_{i,j+1} + u_{i-1,j+1} \right) / 4 \quad \text{(Poisson)}.
\end{align}

If $A$ is a 5-point operator, $\sigma_{-1,1} = \sigma_{1,-1} = 0$. The number of operations per grid point is 13, 9 or 5 in the general 7-point, general 5-point or Poisson case, respectively. The line Gauss–Seidel smoothing operation for lines in the $x_1$-direction is performed according to:

\begin{align}
\text{(4.25)} & \quad \sigma_{-1,0} u_{i-1,j} + \sigma_{0,0} u_{i,j} + \sigma_{1,0} u_{i+1,j} \\
& \quad = f_i^j - \sigma_{0,-1} u_{i,j-1} - \sigma_{0,1} u_{i,j+1} - \sigma_{1,-1} u_{i+1,j} - \sigma_{1,1} u_{i-1,j+1} \quad \text{(general)}, \\
\text{(4.26)} & \quad - u_{i-1,j} + 4 u_{i,j} - u_{i+1,j} = f_i^j + u_{i,j-1} + u_{i,j+1} \quad \text{(Poisson)}.
\end{align}

The right-hand side takes 8, 4 or 2 operations per grid point in the general 7-point, general 5-point or Poisson case, respectively. One can solve one or $m$ identical tridiagonal $n \times n$ systems in about $10n$ or $6nm$ operations respectively (Isaacs and Keller (1966, p. 57)); (by our definition of an operator, our count is twice that of Isaacs and Keller). Hence, solving the tridiagonal systems (4.25), (4.26) takes the following number of operations per grid point: 10, 10 or 6 in the general 7-point, general 5-point or Poisson case, respectively.

The number of operations per grid point for the various smoothing operators is summarized in Table 4.2.

<table>
<thead>
<tr>
<th>Operations per grid point for various smoothing operators</th>
</tr>
</thead>
<tbody>
<tr>
<td>General 7-point</td>
</tr>
<tr>
<td>5-point ILU</td>
</tr>
<tr>
<td>7-point ILU</td>
</tr>
<tr>
<td>G-S by points</td>
</tr>
<tr>
<td>G-S by lines</td>
</tr>
</tbody>
</table>

Clearly, on the basis of this table and the smoothing factors, 7-point ILU comes out best. The Poisson equation can be considered to be representative for the class of selfadjoint elliptic equations with smoothly varying coefficients of the same order of magnitude. In the next section numerical experiments with other typical cases will also be described.
The above analysis is not exact for ILU smoothing, because the coefficients in L and U vary near the boundaries. The decisive test of the merits of the various options lies in numerical experiments.

If the ILU decomposition is constructed by means of Algorithm 2, then the computation of L and U takes 9 or 21 operations per grid point in the 5- or 7-point case respectively. If one also takes into account the computation of booleans of the type \((i,j) \in P\), then the total work becomes proportional to \(8^l\), thus destroying the theoretical \(O(4^l)\) computational complexity of the present multigrid method. This can be avoided by computing the ILU decomposition by means of the following recursive formulae. Consider the 7-point case. Let, in left to right order, \(g, f, e, a, b, c, d\) denote the nonzero diagonals of the given matrix \(A\), and let the subscript \(k\) denote the \(k\)th element, e.g., \(a_k\) is the element in the main diagonal in row \(k\). Similarly, let \(\eta, \zeta, \epsilon\) and \(\alpha\) denote the nonzero diagonals of \(L\), and \(\alpha, \beta, \gamma, \sigma\) those of \(U\). Then we have on an \(m \times n\) grid:

\[
\eta_k = \frac{g_k}{\alpha_{k-1}}, \quad \zeta_k = \frac{f_k - \eta_k \beta_{k-m}}{\alpha_{k-m+1}}, \quad k \geq m; \\
\epsilon_k = \frac{e_k - \eta_k \gamma_{k-m}}{\alpha_{k-1}}, \quad k \geq 2; \\
\alpha_k = \left(\alpha_{k-1} - \zeta_k \delta_{k-m+1} - \eta_k \delta_{k-m} - \epsilon_k \gamma_{k-m+1}\right)^{1/2}, \quad k \geq 1; \\
\beta_k = \frac{b_k - \zeta_k \delta_{k-m+1}}{\alpha_k}, \quad k \equiv mn - 1; \\
\gamma_k = \frac{c_k - \epsilon_k \delta_{k-1}}{\alpha_k}, \quad k \equiv mn - m; \\
\delta_k = \frac{d_k}{\alpha_k}, \quad k \equiv mn - m.
\]

Quantities that are not defined are to be replaced by 0. The recursion formulae (4.27) require 21 operations per grid point, just as the quasi-Algod program (Algorithm 1) (neglecting computation of booleans). However, one might prefer the quasi-Algod program because of its flexibility in changing the sparsity pattern \(P\) and its ease of generalization to systems of partial differential equations.

An efficient way to compute \(A^k\) as defined by (2.16) is given in Appendix B.

5. Numerical experiments. It has already been remarked that multigrid methods in general for linear partial differential equations can be represented by the procedure multigrid method \((k)\) of § 2. It remains to choose \(A^k(k < l), p^k, r^k, sa[k]_k, sb[k], sc[k]_k\) and the smoothing operator \(S\). To judge the merits of all possibilities on purely theoretical grounds seems out of the question, although the fact that (2.16) allows a nicer theory to be developed than does (2.15) induces a preference for (2.16). However, numerical experiments are decisive. Some numerical experiments are described below, and comparison is made with experiments reported by Hackbusch (1978) and Nicolaides (1979). The following cases will be treated, all of them special cases of (2.1):

(5.1) Case 1: \(u_{ii} = 4,\)
(5.2) Case 2: \(u_{11} + 0.01u_{22} = 2.02,\)
(5.3) Case 3: \(u_{11} + 1.7u_{12} + u_{22} = 4.\)

For these three cases, which are among the cases treated by Hackbusch, the boundary condition is \(u_{|\partial D} = x_0x_0\), exact solution \(u = x_0x_0\), starting iterate and \(u = 0.\)
In the experiments to be described, the boundary conditions were not eliminated from the equations.

Table 5.1 shows variants of the multigrid method of § 2 that have been applied to case 1. The columns $p$, $r$ and $A$ give the numbers of the equations where the prolongation, restriction and coarse grid operators are defined. The column $ILU$ gives the number of points in the nonzero structure of the $ILU$ decomposition as described in § 4.

**Table 5.1**

<table>
<thead>
<tr>
<th>Variant</th>
<th>$sa[k]$</th>
<th>$sb[k]$</th>
<th>$sc[k]$</th>
<th>$p$</th>
<th>$r$</th>
<th>$A$</th>
<th>$ILU$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>(2.7)</td>
<td>(2.10)</td>
<td>(2.16)</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>(2.7)</td>
<td>(2.10)</td>
<td>(2.15)</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>(2.8)</td>
<td>(2.9)</td>
<td>(2.15)</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>(2.8)</td>
<td>(2.11)</td>
<td>(2.16)</td>
<td>9</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>(2.7)</td>
<td>(2.10)</td>
<td>(2.16)</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>(2.7)</td>
<td>(2.10)</td>
<td>(2.15)</td>
<td>7</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>(2.7)</td>
<td>(2.10)</td>
<td>(2.16)</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>(2.7)</td>
<td>(2.10)</td>
<td>(2.16)</td>
<td>5</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>(2.7)</td>
<td>(2.10)</td>
<td>(2.16)</td>
<td>7</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>(2.7)</td>
<td>(2.10)</td>
<td>(2.15)</td>
<td>7</td>
</tr>
</tbody>
</table>

Table 5.2 gives the number of iterations $M$ carried out and the average reduction factor $\rho$ (defined by $\rho^M = $quotient of Euclidean norms of residue $Ax - f$ after and before $M$ multigrid iterations) for case 1.

**Table 5.2**

<table>
<thead>
<tr>
<th>Variant</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M$</td>
<td>4</td>
<td>6</td>
<td>7</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>7</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>$\rho$</td>
<td>0.023</td>
<td>0.077</td>
<td>0.11</td>
<td>0.022</td>
<td>0.0079</td>
<td>0.015</td>
<td>0.022</td>
<td>0.12</td>
<td>0.086</td>
<td>0.078</td>
</tr>
</tbody>
</table>

From Table 5.2 the following conclusions are drawn. Comparing variants 1, 2 and 3, we can see that (2.16) is better than (2.15). (In the interior, (2.15) and (2.16) are identical in this case, but at the boundary they differ.) Comparison of variants 1 and 4 shows that the prolongation and restriction operators (2.7), (2.10) perform just as well as the more expensive operators (2.8), (2.11). Comparison of variants 1, 5 and 6 shows that the additional smoothing of variants 5 and 6, which almost doubles the computational work, does not reduce $\rho$ enough to make variants 5 and 6 competitive. Comparing variants 1 and 7, we conclude that improvement of the accuracy of the coarse grid corrections by an increase of $sc[k]$ beyond 1 is not worthwhile. Comparing 1 and 8, one finds that it does not pay to make the $ILU$ decomposition more sparse. Comparison of 1, 9 and 10 shows that smoothing after, rather than before, coarse grid correction is preferable. Variant 1 is clearly superior to the other variants considered, although there is no certainty that this is also true for other problems; the other variants (except variant 2) have not been tested on other problems. (The reason for including further tests with variant 2 is to make the case for (2.16), rather than (2.15), more convincing, which seems desirable because (2.15) seems to be rather more popular at present.)
The performance of variant 1 as a function of \( l \) (mesh size = \( 2^{-l} \)) is given in Table 5.3. Clearly, the rate of convergence is bounded away from 1 as the mesh size tends to zero, as predicted by the theory.

**Table 5.3**  
*Variant 1 applied to case 1*

<table>
<thead>
<tr>
<th>( l )</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>( M )</td>
<td>6</td>
<td>8</td>
<td>9</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>( \rho )</td>
<td>0.013</td>
<td>0.037</td>
<td>0.047</td>
<td>0.040</td>
<td>0.033</td>
</tr>
</tbody>
</table>

Next, variants 1 and 2 and the method tested by Hackbusch (1978) are compared for cases 1, 2 and 3 (Table 5.4). The latter method is denoted by \( H \) in the sequel. Results for \( H \) are directly quoted from Hackbusch (1978). Unfortunately, similar detailed results are not available for other methods in the literature at this moment.

As in Table 5.2, variant 1 is faster than variant 2, which strengthens our preference for (2.16) over (2.15).

**Table 5.4**  
*Results for cases 1, 2, 3; \( l=6 \)*

<table>
<thead>
<tr>
<th>Case:</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>2</th>
<th>2</th>
<th>3</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variant:</td>
<td>1</td>
<td>2</td>
<td>( H )</td>
<td>1</td>
<td>2</td>
<td>( H )</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>( M )</td>
<td>8</td>
<td>10</td>
<td>8</td>
<td>10</td>
<td>10</td>
<td>8</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>( \rho )</td>
<td>0.033</td>
<td>0.088</td>
<td>0.038</td>
<td>0.15</td>
<td>0.37</td>
<td>0.063</td>
<td>0.025</td>
<td>0.051</td>
</tr>
</tbody>
</table>

Before any further conclusions can be drawn from Table 5.4, the computational complexities per iteration must be estimated for variant 1 and \( H \). Estimation of the computational complexity is a tedious business which can be carried out only roughly. But because efficiency is what multigrid methods are all about, and because efficiency and robustness of multigrid methods vary widely (although not very much between variant 1 and \( H \), as it turns out), the computational complexity needs to be determined and reported as well as possible.

There are two main differences between variant 1 and \( H \). The first is that for \( p^k \) and \( r^k \) variant 1 uses the 7-point operators defined by (2.7) and (2.10), whereas \( H \) uses the 9-point operators given by (2.8) and (2.11). This makes application of \( p^k \) and \( r^k \) somewhat cheaper for variant 1, and furthermore, the coarse matrices \( A^k \), \( k < l \), are 7-point rather than 9-point operators, as for \( H \). This makes a difference for the work done on coarser grids. More important for the efficiency is the second main difference between both methods, namely the smoothing operators that are used, because this also affects the work done on the finest grid. For the problems of Table 5.4, \( H \) uses line relaxation along lines in the \( x_1 \)-direction, with a special ordering of the lines. Half the lines are visited before coarse grid correction, the other half, and all lines once more, after coarse correction. The total smoothing work is therefore equivalent to two line relaxations, or, on the finest grid, to 28 (cases 1, 2) or 36 (case 3) operations per grid point, as follows from Table 4.2. (We take the variable coefficient case in order to make the estimates more representative of the general case.) As seen from Table 4.1, for variant 1, smoothing takes 18 operations on the finest grid for cases 1, 2, 3. We expect therefore that the computational work for variant 1 for one
multigrid iteration is roughly 0.65 (cases 1, 2) or 0.5 (case 3) times that of $H$. Taking the average reduction factors $\rho$ of Table 5.4 into account, one comes to the conclusion that the total work of variant 1 will be roughly 0.42 (case 1), 0.96 (case 2) or 0.24 (case 3) times the total work of $H$.

For cases 1 and 3, variant 1 is better than $H$. But for case 2, variant 1 does much worse than for cases 1 and 3, and is not significantly faster than $H$. The following tables give results of further experiments with anisotropic model problems of the type of case 2.

<table>
<thead>
<tr>
<th>$\epsilon$</th>
<th>$\Phi_{xx} + \epsilon \Phi_{yy} = 2 + 2\epsilon$</th>
<th>$\epsilon \Phi_{xx} + \Phi_{yy} = 2 + 2\epsilon$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.5</td>
<td>0.1</td>
<td>0.01</td>
</tr>
<tr>
<td>9</td>
<td>10</td>
<td>7</td>
</tr>
<tr>
<td>$\rho$</td>
<td>0.047</td>
<td>0.10</td>
</tr>
</tbody>
</table>

**Table 5.6**

| $\Phi_{xx} + 0.01 \Phi_{yy} = 2.2$ for various step sizes |
|------------|---------------------------------|---------------------------------|
| $t$ |
| 2 | 3 | 4 | 5 | 6 |
| $M$ | 6 | 10 | 10 | 10 | 10 |
| $\rho$ | 0.0077 | 0.065 | 0.100 | 0.11 | 0.095 |

Because the pivots are chosen in a certain sequence in the $ILU$ decomposition, the $ILU$ smoothing properties are sensitive to rotational transformations, quite like line relaxation. This effect is visible in Table 5.5. For the present $ILU$ decomposition, $\Phi_{xx} + \epsilon \Phi_{yy}$ is the worst anisotropic case (the best for $x_1$-line relaxation), and $\epsilon \Phi_{xx} + \Phi_{yy}$ is the best (here $x_1$-line relaxation would not work at all). Table 5.6 shows that for this problem also the rate of convergence is independent of the mesh size.

Our conclusion for anisotropic problems of the type of case 2 is that, although its performance is somewhat variable, variant 1 is always fast, and never outperformed by $H$. $H$ can be made just as robust as variant 1 for this case by including $x_2$-line relaxation, which would double the work.

Case 1 has also been treated by method $H$ using point relaxation as smoothing operator. The computational work is equivalent to two pointwise sweeps, i.e., 18 operations per point. Reasoning as before, we expect that variant 1 and $H$ require about the same amount of work per iteration. For $H$, a value of $\rho = 0.048$ is reported, and we conclude that the computational complexity of variant 1 will be about 0.61 times that of $H$. Note that this version of $H$ would not work for cases 2 and 3.

In order to be able to compare with results reported by Brandt (1977) and Nicolaides (1979), the computational complexity of variant 1 has been measured in terms of workunits (WU). One WU, as introduced by Brandt (1977) and denoted here by WUGS, is the work of one pointwise Gauss-Seidel sweep on the finest grid. One may similarly define one WULU as the work of one $ILU$ smoothing operation on the finest grid. For the computations reported in Tables 5.2, 5.3, 5.4., 5.5 and 5.6, it has been measured that the average cost of one multigrid iteration with variant 1 is 1.99 WULU with variance 0.07 WULU. The smallness of this variance is another
indication of the robustness of variant 1, and makes it reasonable to assume henceforth
a fixed cost of 2 WULU for one multigrid iteration with variant 1. In order to be able
to compare with authors who have measured work in terms of WUGS, we will assume,
on the basis of Table 4.1, that 1 WULU = 2 WUGS (general 5-point case) or
1 WULU = 3 WUGS (Poisson) exploiting the fact that the coefficients are constant.
Of course, in this case one would be better off with a Fourier-analysis–cyclic-reduction
method.) This figure may also be obtained by a theoretical operations count as follows.
Table 5.7 gives the operations count per grid point for the operations that occur in
the procedure multigrid method of § 2.

In Table 5.7, savings near the boundaries are neglected. We consider variant 1.
Table 5.8, which expresses the operations count of one multigrid iteration in WULU
units, is easily compiled. The work on the coarsest grid is neglected; $A^1$ is assumed
to be a 5-point operator and $A^k$, $k < l$, are 7-point operators. $S$ is applied once for
$k = 1(l), r(f - Au)$ is executed for $k = 1(l)l - 1$ and $u + pu$ is computed for $k = 2(l)l$.
One WULU is 18 operations per gridpoint.

| TABLE 5.7 |
|---|---|
| Operations per grid point for portions of procedure multi-grid method |
| $A$: 5-point | $A$: 7-point |
| $S$ | 18 | 18 |
| $r(f - Au)$ | 18 | 22 |
| $u + pu$ | 3 | 3 |

| TABLE 5.8 |
|---|---|---|---|---|---|
| Operations count for multigrid method, variant 1, expressed in WULU |
| $l$ | 2 | 3 | 4 | 5 | 6 | 7 |
| WULU | 1.89 | 2.08 | 2.05 | 1.99 | 1.95 | 1.93 |

Nicolaides (1979) reports results for combinations of a multigrid method and two
finite element methods. One of these, namely linear elements, turns out to be
equivalent to variant 1 with pointwise Gauss–Seidel smoothing instead of ILU smooth-
ing, if the coefficients are constant. After varying $sa[k]$ and $sc[k]$ (see procedure
multigrid method $(k)$ § 2) a best result of 3.9 WUGS per decimal digit is measured
for the first 3 multigrid iterations for the Poisson equation with smooth initial error
and a $64 \times 64$ grid. We find (case 1, Table 5.3) $\rho = 0.033$, hence 0.67 iterations per
decimal digit, at a cost of 1.34 WULU = 2.68 WUGS (3.02 if the constant coefficients
are exploited). During the first three iterations the rate of convergence is $\rho = 0.0064,$
and a cost per decimal digit is found of 0.91 WULU = 1.8 WUGS. Apart from the
Poisson equation, several interesting variable coefficient cases are treated successfully
by Nicolaides (1979). One of these is (2.1) with $a_{11} = a_{22} = abs (\sin kx_1 \sin kx_2), a_{12} =
a_{21} = b_1 = c = f = 0, u/\pi = 0$. The initial values for $u$ are uniformly randomly dis-
tributed in $[-1, 1]$. Three multigrid iterations are performed. The results for variant
1 are given in Table 5.9. For this test case, Nicolaides (1979) used bilinear elements.
Hence $A^1$, $k = l(-1)l$, are 9-point operators in his case, so that one Gauss–Seidel
sweep takes 17 operations per grid point, and 1 WULU = 18/17 WUGS. Table 5.10
TABLE 5.9
ρ for test case of Nicolaides

<table>
<thead>
<tr>
<th>l/k</th>
<th>2</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>32</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>0.060</td>
<td>0.063</td>
<td>0.067</td>
<td>0.067</td>
<td>0.061</td>
</tr>
<tr>
<td>6</td>
<td>0.065</td>
<td>0.060</td>
<td>0.062</td>
<td>0.070</td>
<td>0.067</td>
</tr>
</tbody>
</table>

gives the work per decimal digit expressed in WUGS. Before the comma: variant 1; after the comma: method of Nicolaides.

TABLE 5.10
Cost per decimal digit for test case of Nicolaides

<table>
<thead>
<tr>
<th>l/k</th>
<th>2</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>32</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1.7, 4.7</td>
<td>1.7, 5.2</td>
<td>1.8, 5.4</td>
<td>1.8, 5.9</td>
<td>1.7, 5.0</td>
</tr>
<tr>
<td>6</td>
<td>1.8, 4.5</td>
<td>1.7, 4.8</td>
<td>1.7, 5.8</td>
<td>1.9, 6.3</td>
<td>1.8, 6.1</td>
</tr>
</tbody>
</table>

Distinguishing features of the multigrid methodology advocated by Brandt (1977), (1979) are: (2.15) is used for the coarse grid operators, (2.8) and (2.9) or (2.11) are used for prolongation and restriction and Gauss–Seidel relaxation by points or by lines is used for smoothing. At certain instances, higher order interpolation is used for $\rho^k$. Furthermore, in contrast with the fixed smoothing strategy of the methods discussed in the foregoing, the smoothing strategy is variable: The number of smoothing operations preceding and succeeding a coarse grid correction depends on the rate of convergence which is monitored by the algorithm. Brandt (1977, p. 349) introduces the quantity $\mu_0$ as the factor by which the errors are reduced per WU of computational work, counting only smoothing work; Brandt derives a relation between $\mu_0$ and $\bar{\rho}$ (defined in (4.4)), which in the present context reads

$$\mu_0 = \bar{\rho}^{3/4}. \tag{5.4}$$

This relation is not rigorous, but is found to be very realistic according to Brandt (1977). With Gauss–Seidel smoothing, $\bar{\rho} = \frac{1}{2}$ for the Poisson equation (see (4.18)), and one finds $\mu_0 = 0.595$, whereas with $ILU$ smoothing $\bar{\rho} = 0.126$ (see (4.17)), hence $\mu_0 = 0.211$. Taking into account that 1 WULU = 2 WUGS (Table 4.1, general 5-point case), $ILU$ smoothing comes out best. Counting only relaxation work, 1 multigrid iteration with variant 1 takes somewhat less than $\frac{1}{2}$ WULU, and the observed value of $\mu_0$ comes out to be 0.077 per 1 WULU for case 1, $l = 6$ (on the basis of Table 5.3), which is considerably better than the estimate $\mu_0 = 0.211$ based on (5.4). In (5.4) the effects of the prolongation and restriction operators and the quality of the coarse grid operators are not taken into account: apparently these have a very beneficial effect in variant 1, and our preference for (2.7), (2.10) and (2.16) is strengthened once more. In an experiment on the Poisson equation using the variable smoothing strategy, Brandt (1977, p. 354) measures $\mu_0 = 0.537$ per 1 WUGS; cf. our just-quoted value $\mu_0 = 0.077$ per 1 WULU $\sim 0.278$ per 1 WUGS. One might ask whether the inclusion of a variable smoothing strategy would further improve the performance of variant 1. Because of the (possibly problem-dependent) tuning this would require, and in view of the speed and robustness already obtained, we do not think it worthwhile to pursue this matter further.
This concludes our comparison with other methods. Another important model problem, which has not been treated by other authors, is the convection-diffusion equation. In Wesseling and Sonneveld (1980) this equation is treated with variant 1; furthermore, results are reported for the Navier–Stokes equations.

6. Final remarks. It has been shown that the Galerkin coarse grid approximation, 7-point prolongation and restriction, incomplete $LU$ smoothing and a fixed multigrid strategy (i.e., in procedure multigrid method (k) of §2 $sa[k] = 0$, $sb[k] = sc[k] = 1$ for all problems) result in a fast and robust multigrid method. Coarse grid Galerkin approximation was found to be better than finite difference approximation, and $ILU$-smoothing is better than other smoothing methods that have been proposed. Furthermore, 7-point prolongation and restriction is competitive with 9-point prolongation and restriction, which is important for three-dimensional problems.

It seems likely that multigrid methods will soon come to be widely used as fast solvers for elliptic problems. An important aspect that has not been touched upon here is adaptive mesh generation (cf. Brandt (1977), (1979), Hemker (1980a)). Furthermore, the application of multigrid methods is not restricted to partial differential equations (cf. Hemker and Schippers (1981)).

Appendix A

Proof of (3.26). The operator $A$ is defined by (2.3) with $b_1 = c = 0$. Define (deleting the superscript $l$ for brevity) $ar{A} = -\frac{1}{2}a_i(
abla_i \Delta_l + \Delta_i \nabla_l)$. It is not difficult to verify that

(A1) \[ \|\bar{A}u\|_0 \leq \alpha \|u\|_1, \quad \alpha^2 = 2 \sum_{j=1}^2 \sup \|\nabla_j a_j\|^2. \]

Nitsche and Nitsche (1969) show that

(A2) \[ \|u\|_2 \leq \sqrt{2} C_1^{-2} C_0 \|\bar{A}u\|_0. \]

(Their definition of $\bar{A}$ and $\|\|_2$ is slightly different, but their proof holds also in our case.) Furthermore, using (3.6) and (3.12),

(A3) \[ \|Au\|_0 \leq \|Au\|_{-1} \leq \|u\|_1 \leq C_1 \|u\|_0. \]

Combining (A1)–(A3) one obtains

(A4) \[ \|u\|_2 \leq D_3 \|A^lu\|_{0,h}, \quad D_3 = \sqrt{2} C_1^{-2} C_0 \left( \frac{1 + \alpha}{C_5} \right). \]

Next, (A4) is generalized to coarser grids. First some useful consequences of (3.8) are derived. Define $r^{km} = r^{k+1}r^{k+2} \ldots r^m$, $m > k$. We have

(A5) \[ \|u^{km}\|_{-1,k} = \sup_{x \in \Omega_{k+1}} |(u^{m}, p^{m,k}v^{k})_m| \leq \sup_{x \in \Omega_{k+1}} |(u^{m}, p^{m,k}v^{k})_m| \leq \|u^{m}\|_{-1,m}. \]

Furthermore,

(A6) \[ \|u^{m} - p^{m,k}u^{km}\|_{1,m} \leq 2^{-k} C_2 \|u^{m}\|_{2,m}, \quad m > k, \]

which may be derived by noting that $1 - p^{k,km} = 1 - p^{m,m} + p^{m}(1 - p^{m-1,m-1})r^{m} + \ldots + p^{m,k+1}(1 - p^{k+1,k+1})r^{k+1,m}$, and from (3.8a, b, e). Similarly
one may derive
\[(A7) \quad \|u^k - r^km_p u^k\|_{1,k} \leq \frac{3}{2} C_4 \cdot 4^{-k}\|u^k\|_{1,k}, \quad m > k.\]

Next we note that, for arbitrary (deleted) \(k\)
\[(A8) \quad \|u\|_1 \leq C_1^{-1} C_3^{-1} \|Au\|_1,\]

since \(\|Au\|_1 = \sup_{(u,v) \in \Omega} |B(u,v)| \leq C_1^2 C_3 \|u\|_1\), because of (3.17).

Now (3.26) will be derived. Let \(A^k u^k = f^k, \quad k < l\). Define \(\hat{u}^k\) and \(\hat{u}^l\) by
\[A^k \hat{u}^k = r^k f^k, \quad A^l \hat{u}^l = r^l f^l.\]
Using (3.8a), (3.21) and (A6), we get
\[(A9) \quad \|u^k - r^k \hat{u}^k\|_{1,k} \leq C_1^{-1} \|r^k \hat{u}^k - r^k \hat{u}^l\|_{1,l},\]
\[\leq C_1^{-1} \|r^k \hat{u}^k - \hat{u}^l\|_{1,l} + C_1^{-1} \|\hat{u}^l - r^k \hat{u}^l\|_{1,l},\]
\[\leq C_1^{-1} (1 + D_1) \|\hat{u}^l - r^k \hat{u}^l\|_{1,l} \leq C_1^{-1} C_2 (1 + D_1) 2^{-k} \|\hat{u}^l\|_{2,l}.\]

From (A4) and (3.8c) it follows that
\[(A10) \quad \|\hat{u}^l\|_{2,l} \leq D_2 |f^l|_{0,k}.\]

Since \(A^k (u^k - \hat{u}^k) = f^k - r^k p^k f^k\), we conclude from (A7), (A8) and (3.7) that
\[(A11) \quad \|u^k - \hat{u}^k\|_{1,k} \leq \frac{3}{4} \sqrt{2} C_1^{-2} C_4 C_5^{-1} 2^{-k} \|f^k\|_{0,k}.\]

From (A9), (A10), (A11) and (3.7) we have
\[(A12) \quad \|u^k - r^k \hat{u}^l\|_{2,k} \leq C_1^{-2} (\frac{32}{3} \frac{C_4}{C_5} + \sqrt{8} C_1 C_2 D_2 (1 + D_1)) \|f^l\|_{0,k}.\]

With (3.8c) and (A10) one obtains
\[(A13) \quad \|p^k \hat{u}^l\|_{2,k} \leq D_3 |f^k|_{0,k}.\]

Combination of (A12) and (A13) results in
\[(A14) \quad \|u^k\|_{2,k} \leq D_2 |f^k|_{0,k},\]

with \(D_2\) following in an obvious way from (A12), (A13). This is the desired result.

**Appendix B.** One way to compute \(A^k\) as defined by (2.16) is as follows, following Frederickson (1975). Let (2.7) and (2.10) hold, and let \(A^k\) be a 7-point operator; the following treatment is easily extended to other cases. Define \(Z = \{m1, m2\} |m2 = 0, \pm 1, (1, 1), (-1, -1)\), \(R^k = \{m1, m2\} |m2 = 0, 1\}.\) \(M\) will be used to enumerate the atoms of the difference molecule corresponding to \(A^k\), and \(R^k\) to enumerate the points of \(\Omega\). Elements of the matrix \(A^k\) will be denoted by \(A^k_{ij}, i \in R^k,\)
\(j \in M.\) With this notation, matrix-vector multiplication is defined by \((A^k u)^i = \sum_{i \in M} A^k_{ij} u^j,\)
Prolongation and restriction may be defined as follows. Define \(\iota: Z \to R\) by \(\iota(0, 0) = 1; \iota (f) = \frac{1}{2}, \forall f \in M(0, 0); \iota (f) = 0, \forall f \not\in M.\) Then
\[(B1) \quad (p^k u^{k \pm 1})_i = \sum_{i \in Z} \iota (i \pm 2j) u^{k \pm 1}_j, \quad i \in R^k,\]
\[(B2) \quad (r^k u^k)_i = \frac{1}{2} \sum_{i \in Z} \iota (j) u^k_{2i+j}, \quad i \in R^{k-1},\]
\[(B3) \quad A^k_{l,j} = \frac{1}{2} \sum_{u \in Z} \iota (u) A^k_{2i+u,2i+u \pm 1}, \quad i \in R^{k-1}, \quad j \in M.\]
The (obvious) key to efficient computation of $A^{k-1}$ is to pick a combination $(j, u, v)$ and to check whether $2j + u - v \in M$ before sweeping through $R^{k-1}$. The obtained contribution to $A^{k-1}_{j}$ is stored and is augmented by further sweeps through $R^{k-1}$ for different combinations $(j, u, v)$, until all relevant combinations $(j, u, v)$ have been exhausted. By checking off all possible combinations $(j, u, v)$, one arrives at an operations count of about 110 per point of $\Omega^{k-1}$. Thus, computation of $A^k, k = l - 1(-1)$, takes less than 37 operations per point of $\Omega$, or 2 WULU.

REFERENCES


P. O. Frederiksson (1975), Fast approximate inversion of large sparse linear systems, Mathematics Report 7-75, Lakehead University, Thunder Bay, Ontario, Canada.


