CWI Tracts

CWI
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands
Telephone 31 - 20 592 9333, telex 12571 (mactr nl),
telefax 31 - 20 592 4199

CWI is the nationally funded Dutch institute for research in Mathematics and Computer Science.

Contributions to multigrid

A selection of contributions to
the Fourth European Multigrid Conference,
held in Amsterdam, July 6-9, 1993

P.W. Hemker, P. Wesseling (eds.)

1991 Mathematics Subject Classification: 65M55, 65N55.
ISBN 90 6196 439 3
NUGI-code: 811

Preface

This volume contains a selection from the papers presented at the Fourth European Multigrid Conference, held in Amsterdam, July 6-9, 1993. Another selection, including the contributions by the invited speakers, appears as the proceedings of the Fourth European Multigrid Conference and is published by Birkhäuser Verlag, Basel.

When the first multigrid conference was held in 1981 there was no doubt about the usefulness of a conference dedicated specially to multigrid, because multigrid was a new and relatively unexplored subject, still in a pioneering stage, and pursued by specialists. The past twenty years have shown a rapid growth in theoretical understanding, useful applications and widespread acceptance of multigrid in the applied disciplines. Hence, one might ask whether there is still a need today to continue organising conferences specially dedicated to multigrid. The general consensus is that the answer is affirmative. New issues arise that are best addressed or need also be addressed from a special multigrid point of view. Most prominent among these issues are parallel computing, adaptive computations and applications other than elliptic boundary value problems. Multigrid has much impact on computational fluid dynamics. This influence is much reflected in the present selection of contributions: more than 10 of the 17 contributions are devoted to CFD.

Beside meetings, exchange of information on multigrid research is also aided by MGNet, in which papers and software are stored electronically, and may be retrieved by ftp. MGNet is maintained by C. Douglas of Yale University. Information on MGNet can be obtained by sending email to mgnet-requests@cs.yale.edu.

The conference was made possible by the Centre of Mathematics and Computer Science (CWI), Amsterdam, and the University of Amsterdam. Financial support was provided by Akzo NV, IBM Nederland NV and the Royal Dutch Academy of Science (KNAW). We are also greatly indebted to Mr Frans Snijders and Ms Simone van der Wolff for their help in organising the conference in the historic setting of old Amsterdam.

Amsterdam / Delft, December 1993
P.W. Hemker
P. Wesseling

Contents

1

Burgers Equation and Multi-Grid Techniques

Alfio Borzì[1]

ABSTRACT We analyze a discretization of the Burgers equation. For the resulting discrete problem we prove the existence of solutions for a class of initial-boundary conditions. Then we use this introductory evolution equation to present an algebraic multi-grid method for fluid flow problems. This method consists of a smoothing iteration, transfer of defects on a coarser space, and prolongation of obtained corrections to the previous space. But these operators are here interpreted in terms of a direct multi-grid substitution method. This fact seems to suggest that multi-grid methods are "approximating" versions of suitable chosen direct methods. This algorithm is compared with a standard multi-grid method by means of numerical experiments. It turns out that the algebraic algorithm provides the exact solution of the discrete elliptic problem to be solved at each time step.

1 Introduction

Today there exists a large choice of methods for the numerical solution of partial differential equations. Among them we find the modern multi-grid (MG) scheme. The first application of this method was to solve elliptic boundary value problems [1]. Nowadays the range of applications of MG algorithms is quite large. One of the most important is for solving fluid flow problems.

The first fluid dynamics computations were based on finite difference methods. That is, the differential equation which governs the fluid is discretized approximating each partial derivative with a difference-quotient operator. Therefore, the differential problem is approximated by a large system of algebraic equations. Fortunately this algebraic problem is normally sparse, i.e., has relatively few non-vanishing elements. Hence it can be solved using iterative methods. However, these methods present a problem: they effectively reduce only a part of the spectrum of the solution error components. This principal problem has been solved by means of multi-grid techniques.

---

[1]SISSA, Scuola Internazionale Superiore di Studi Avanzati, Via Beirut 2-4, 34013 Trieste, Italy. Address after 15 Novenber 1993: OUCL, Wolfson Building, Parks Road, Oxford, OX1 3QD, UK

The basic MG idea is to treat the different spectral components of the solution error on different grid spaces. On each space an iterative method is used to solve those components of the error which are more oscillating in character. As a consequence, on a given grid, the error remains smooth and can be approximately represented and computed on a coarser grid space as a solution of a "coarse" problem. Therefore the coarse solution is used to correct the finer solution for the smooth components of the error. This last step is called coarse grid correction. Because the smoothness can be considered a geometric concept, sometimes this approach is referred to as "geometric" multi-grid method.

However, in many complex problems it is difficult to handle all geometrical aspects. Therefore a multi-grid method based only on the algebraic features of the system of equations to be solved could be convenient. For this purpose an algebraic MG method has been proposed [2,3]: the vector solution u is a set of variables which represent the value of the solution on the grid points. Then one selects a subset of these variables so that the remaining ones are "strongly connected" [2] to them. This subset can be represented in a smaller algebraic space as a vector, solution of a coarser algebraic problem. Therefore these variables are part of the entire solution and using the "strong connection" it is possible to compute all the elements of u on the fine space.

Based on these algorithms it is possible to construct MG methods for evolution equations. The time evolution is described by a sequence of systems of equations which give the solution at different time steps. Therefore the multi-grid methods are used to solve each problem of the sequence. Here we present two MG schemes. The first is a standard MG method for evolution equations. The second is based on algebraic considerations and constitutes our first attempt to construct an algebraic multi-grid method for evolutionary problems.

Now in order to present them, we choose as a model problem a nonlinear diffusive wave equation known as the Burgers equation [4],

$$\partial_t u + u\partial_x u - \delta\partial_{xx} u = 0 \ , \tag{1}$$

where $\partial_x \phi \equiv \frac{\partial \phi}{\partial x}$, $\partial_t \phi \equiv \frac{\partial \phi}{\partial t}$ and $\partial_{xx}\phi \equiv \frac{\partial^2 \phi}{\partial x^2}$. δ is the so called *diffusion coefficient.*

This equation describes a balance between the nonlinear convective term and the linear dissipative term. These terms are present in the same way in the incompressible Navier-Stokes (NS) equation, making (1) a qualitatively correct approximation to the one-dimensional NS equation. However, for some dissipative flow problems such as shock propagation, compressible turbulence, etc., (1) is considered to be the appropriate mathematical model. On the other hand, the Burgers equation provides a suitable model for testing computational algorithms for problems involving the evolution of shocks.

Nevertheless, this equation does not represent the main difficulties encountered in the numerical solution of higher dimensional flow problems. Actually we choose this equation to provide an introductory problem in order to study an algebraic multi-grid (AMGM) approach to the solution of evolution equations.

In the following section we present our algorithm based on the formal requirements given in [2,3]. It is constructed by introducing a MG substitution method which mimics the algebraic multi-grid process. In section 2 we also sketch a standard MG scheme. In section 3 we analyze a discretization of (1) using the M-matrix formalism. We prove, under suitable conditions, the existence of the solution for a class of initial-boundary value problems. Finally, in section 4, we compare numerically the algebraic method with the standard MG approach to (1) we sketch here. In section 5 we present our conclusions.

2 Multi-Grid Methods and Evolution Equations

For generality of presentation we consider the implicit time-discretization of an initial boundary-value problem in a bounded domain Ω,

$$\partial_t u + \mathcal{A}(u,t) = 0, \quad t \geq 0, \ x \in \Omega \ , \tag{2}$$

$$u(x,0) = u_0(x) \quad in \ \Omega \ , \tag{3}$$

$$u(.,t) = g_\Gamma(t) \ , \ \text{on the boundary } \Gamma \ , \tag{4}$$

where $\mathcal{A}(.,t)$ is an elliptic differential operator which could be linear or nonlinear. The functions $u_0(x)$ and $g_\Gamma(t)$ give the initial and boundary values respectively.

Let us choose as Ω the bounded domain (a,b) and $\Gamma = \{a, b\}$. We construct a grid Ω_h on Ω with mesh size $h = \frac{(b-a)}{n+1}$, n being the number of interior grid points. We suppose to discretize equation (2) on the grid Ω_h at the time step $t = k\Delta t$ ($\Delta t > 0$, step size) so that we obtain a linear discrete elliptic problem defined on Ω_h,

$$A^h u^{h,k} = F^{h,k} \ , \tag{5}$$

where $u^{h,k}$ is the approximate solution and $F^{h,k}$ defines the right-hand side for the current time level k. Therefore one gets a sequence of discrete elliptic problems which should be solved at each time step. We use multi-grid techniques to solve this boundary value problem.

2.1 The FAS Algorithm

The MG method works efficiently because on each grid it handles only the high frequency (HF) components of the solution, i.e., those components whose wavelength $\lambda \in (2h, 4h]$. First of all, for these methods one needs a convergent iterative scheme which is used to treat with just the HF components of the solution, or equivalently, to dump effectively the HF components of the error. We denote such smoothing iteration by:

$$u^h_{new} = \mathcal{S}^h(u^h_{old}, F^h) \ , \tag{6}$$

where we have omitted the time level k.

Because (1) is nonlinear, a nonlinear multi-grid method is appropriate for its resolution. We report here a standard nonlinear multi-grid scheme, that is the so called FAS (Full approximation storage) of A. Brandt [2]. Let us introduce a sequence of grids with mesh sizes $h_1 > h_2 > ... > h_M$, so that $h_{\ell-1} = 2h_\ell$. Equation (5) with discretization parameter h_ℓ will be denoted as

$$A^\ell u^\ell = F^\ell \ . \tag{7}$$

If one applies (6) to (7), an approximated solution $\tilde{u}^\ell$ is obtained. Because of the relaxing property of (6), the approximated solution has an error, $\tilde{e}^\ell = \tilde{u}^\ell - u^\ell$, with only low frequency (LF) components. Then it can be represented well on a coarser grid. Now transfer $\tilde{u}^\ell$ onto the next coarser grid, $\tilde{u}^{\ell-1} = \hat{I}_\ell^{\ell-1}\tilde{u}^{h_\ell}$, $\hat{I}_\ell^{\ell-1}$ being a restriction operator. Hence the coarsening procedure applies and defines the coarse grid equation,

$$A^{\ell-1}\hat{u}^{\ell-1} = \hat{F}^{\ell-1} \ , \tag{8}$$

where

$$\hat{F}^{\ell-1} = I_\ell^{\ell-1}F^\ell + A^{\ell-1}\tilde{u}^{\ell-1} - I_\ell^{\ell-1}A^\ell\tilde{u}^\ell \ . \tag{9}$$

$I_\ell^{\ell-1}$ is a fine-to-coarse grid transfer operator not necessarily equal to $\hat{I}_\ell^{\ell-1}$. Moreover, $A^{\ell-1}$, as A^ℓ, is obtained by the discretization of the differential problem but with respect to $h_{\ell-1}$.

Having obtained the solution of the coarse grid equation $\hat{u}^{\ell-1}$, the difference $\hat{u}^{\ell-1} - \tilde{u}^{\ell-1}$ approximates in the coarse grid $\Omega_{h_{\ell-1}}$ the error $\tilde{e}^\ell$ of $\tilde{u}^\ell$ in the finer grid [2]. This we use to obtain a coarse grid (CG) correction to the finer-grid solution

$$\tilde{u}^\ell \leftarrow \tilde{u}^\ell + \hat{I}_{\ell-1}^\ell(\hat{u}^{\ell-1} - \tilde{u}^{\ell-1}) \ , \tag{10}$$

where $\hat{I}_{\ell-1}^\ell$ is a coarse-to-fine grid interpolation operator. Finally one applies relaxation (6) at level ℓ, in order to smoothen errors coming from the interpolation procedure (post-smoothing). To solve equation (7) one employes a CG correction recursively, i.e. equation (8) is itself solved by iteration sweeps combined with a further CG correction. The FAS algorithm is very efficient, it requires no linearization and the n equations obtained by the discretization of an elliptic differential problem on Ω_{h_M} are solved to the desired accuracy in $O(n)$ operations.

Notice that $A^{\ell-1}$ and A^ℓ gives the same discrete problem. Hence also $\hat{u}^{\ell-1}$ should approximate $\tilde{u}^\ell$ on the coarse grid. Therefore $\tilde{u}^\ell$ must be smooth so that it can be represented on $\Omega_{h_{\ell-1}}$. This property provides the justification for the use of common restriction and interpolation operators. However, for some problems, it is difficult to choose appropriately these operators. In this case, an alternative strategy is provided by an algebraic approach.

2.2 The Algebraic Multi-Grid Method

An algebraic multi-grid method is formally equivalent to the method sketched above. However, in this case $A^{\ell-1}$ is given by $I_\ell^{\ell-1}A^\ell\hat{I}_{\ell-1}^\ell$ (Galerkin approach). In

addition, the restriction and the interpolation operators should be defined using only the algebraic features of the problem in hand. That is for example the values of the entries of the matrix A^ℓ.

In order to present the essentials of the algebraic multi-grid algorithm we will consider that A^ℓ is a $n_\ell \times n_\ell$ tridiagonal matrix, whose elements a_{ij} with $i = j$ or $|i - j| = 1$ are nonzero, and the matrix is not *ill-conditioned* [5].

The algebraic approach, as the geometric one, is also based on the notion of smoothness; however it has now an algebraic sense: the error $\tilde{e}^\ell$ is smooth if the corresponding defect $d^\ell = A^\ell \tilde{u}^\ell - F^\ell$ is such that $d^\ell << \tilde{e}^\ell$ [2,3]. Again the smoothness provides the justification for the coarsening and the interpolation procedure. For the latter consider the defect equation resulting from few smoothing iterations $a_{ii-1}\tilde{e}^\ell_{i-1} + a_{ii}\tilde{e}^\ell_i + a_{ii+1}\tilde{e}^\ell_{i+1} = d^\ell_i$, $i = 1, \ldots, n_\ell$, where $e^\ell_0 = e^\ell_{n_\ell+1} = 0$ in case of Dirichlet boundary conditions.

Now select, in analogy with the geometric MG approach, the subset of "coarse" variables $e^{\ell-1} = (e^\ell_2, e^\ell_4, \ldots, e^\ell_{n_\ell-1})$ and notice that the remaining variables can be obtained as

$$e^\ell_i = -\frac{a_{ii-1}}{a_{ii}} e^\ell_{i-1} - \frac{a_{ii+1}}{a_{ii}} e^\ell_{i+1} \ , \qquad i = 1, 3, \ldots, n_\ell \ , \tag{11}$$

where the defect is considered to be approximately zero. We notice that (11) gives exactly the piecewise linear interpolation in case of the simplest one-dimensional Dirichlet problem [6]. In that case the restriction operator is defined (up to a constant) as the transpose of the matrix operator $\hat{I}^\ell_{\ell-1}$. So, in algebraic multi-grid, there is the attempt to define the coarsening procedure using the assumption $I^{\ell-1}_\ell = (\hat{I}^\ell_{\ell-1})^T$ (T means transpose). However this result seems to be restricted to symmetric problems as we will see later.

For the moment let us construct a simple substitution procedure which result to be formally equivalent to the algebraic MG coarsening. First consider the ith (i odd integer) equation of the above defined tridiagonal system and express the ith element of e^ℓ in terms of the neighboring elements:

$$e^\ell_i = -\frac{a_{ii-1}}{a_{ii}} e^\ell_{i-1} - \frac{a_{ii+1}}{a_{ii}} e^\ell_{i+1} + \frac{d^\ell}{a_{ii}} \ , \qquad i = 1, 3, \ldots, n_\ell \ . \tag{12}$$

Then use (12) in order to eliminate the variables e^ℓ_i , $i = 1, 3, \ldots, n_\ell$ from the complete set of equations, obtaining the following coarse problem

$$\begin{aligned}(-\tfrac{a_{ii-1}a_{i-1i-2}}{a_{i-1i-1}})e^\ell_{i-2} + (-\tfrac{a_{ii-1}a_{i-1i}}{a_{i-1i-1}} + a_{ii} - \tfrac{a_{ii+1}a_{i+1i}}{a_{i+1i+1}})e^\ell_i + (-\tfrac{a_{ii+1}a_{i+1i+2}}{a_{i+1i+1}})e^\ell_{i+2} = \\ -\tfrac{a_{ii-1}}{a_{i-1i-1}} d^\ell_{i-1} + d^\ell_i - \tfrac{a_{ii+1}}{a_{i+1i+1}} d^\ell_{i+1} \ ,\end{aligned} \tag{13}$$

where $i = 2, 4, \ldots, n_\ell - 1$.

We notice that the matrix of coefficients of the coarse problem obtained by substitution is equal to $I^{\ell-1}_\ell A^\ell \hat{I}^\ell_{\ell-1}$ with $\hat{I}^\ell_{\ell-1}$ given by (11) and $I^{\ell-1}_\ell$ acting as

$$d_I^{\ell-1} = -\frac{a_{ii-1}}{a_{i-1i-1}} d_{i-1}^{\ell} + d_i^{\ell} - \frac{a_{ii+1}}{a_{i+1i+1}} d_{i+1}^{\ell} , \qquad i = 2I \ , \ I = 1, 2, \ldots, n_{\ell-1} \ . \quad (14)$$

Equation (14) means that the restriction operator can be expressed as the transpose of the interpolation operator corresponding to $(A^{\ell})^T$. Obviously if the problem is symmetric we have $I_{\ell}^{\ell-1} = (\hat{I}_{\ell-1}^{\ell})^T$ as in [2,3].

Now suppose to solve (13) obtaining the coarse variables defined above, then the remaining variables are found using (12). However, as in the geometric case, we can solve the algebraic problem recursively. At each level is applied the coarsening procedure as far as the coarsest space, with only few variables, is reached. Then the following finer variables are computed by means of (12).

There are only two differences between the algebraic MG algorithm and the substitution multi-grid method presented above. First of all, the former requires a smoothing iteration whereas the second not. The second difference is in the coarse correction: that of the substitution contains a term which depends on the defect and coincides with (11) only if $d_i^{\ell} = 0$, $i = 1, 3, \ldots, n_{\ell}$. These results suggest that the action of the smoothing iteration is to arrange the defect so that (12) can be well approximated by (11). Hence, in principle, the convergence of the relaxation is not required.

We notice that the particular structure of the matrix A^{ℓ} suggests that a red-black (RB) Gauss-Seidel [7] iteration should be used. That is, split the set of interior grid points $x_i = a+ih$, $i = 1, 2, \ldots, n_{\ell}$ into a 'red' and a 'black' subset: into points whose index i is even and odd respectively. Then each subset is numbered lexicographically and the Gauss-Seidel iteration is applied first to the red subset and subsequently to the black one. It is clear that after this iteration one obtains $d_i^{\ell} = 0$, $i = 1, 3, \ldots, n_{\ell}$. Therefore, in this case, the substitution and the algebraic multi-grid methods will give the same results. That is, the AMGM solves exactly the discrete problem.

2.3 Multi-Grid Methods for Evolution Equations

We consider now the problem of how to advance the solution in time using multi-grid methods. There are indeed many multi-grid approaches to evolutionary problems [6,8,9,10]. Because of its simplicity let us use the modified nested iteration (MNI) proposed by W. Hackbusch [10]. The first step of this method is to solve the discrete problem associated to the coarsest grid Ω_{h_1} where the solution is represented in few grid points. The corresponding system of equations (7), for the current time step $k\Delta t$, is solved by applying a few steps of the iteration procedure (6) obtaining the solution $\tilde{u}^{h_1,k}$. On this grid ($\ell = 1$) the difference between the obtained solution and the one relative to the previous time step is computed, $\tilde{u}^{h_1,k} - \tilde{u}^{h_1,k-1}$. This difference is used to construct the approximation to the solution for the current time step in the following finer level. That is by means of a

prolongation operator $P_{\ell-1}^{\ell}$ we have

$$\tilde{u}^{\ell,k} = \tilde{u}^{\ell,k-1} + P_{\ell-1}^{\ell}(\tilde{u}^{\ell-1,k} - \tilde{u}^{\ell-1,k-1}) \ . \tag{15}$$

This approximation is then used as a starting value for the multi-grid solution process in order to get the solution on the finer level. This procedure is repeated until the finest level ($\ell = M$) is reached. We summarize here the algorithm just described

- Modified nested iteration (MNI)

  1. Compute the solution for the current time step in the coarsest level, $\tilde{u}^{h_1,k}$, $(\ell = 1)$;
  2. Increase ℓ by one and use the computed values at level $\ell - 1$ to approximate the solution in the next finer grid ℓ :

  $$\tilde{u}^{\ell,k} = \tilde{u}^{\ell,k-1} + P_{\ell-1}^{\ell}(\tilde{u}^{\ell-1,k} - \tilde{u}^{\ell-1,k-1});$$

  3. Solve with FAS the discretized equation (7) at level ℓ using the given initial approximation $\tilde{u}^{\ell,k}$;
  4. Repeat the procedure starting from 2. until the finest level M is reached.
  5. Increase the time level k by one and go to 1.

We observe that the prolongation (15) is also based on geometric considerations: it provides a good approximation to the solution for following finer levels if the solution changes smoothly from one time step to another.

Therefore we will not define a MNI scheme in case of an algebraic multi-grid method and the solution will be advanced in time solving (5) at each time step as a pure algebraic problem, whose coefficients will be determined by the solution at the previous time level.

3 A Discretization of the Burgers Equation

We have supposed that the solution $u^{h,k}$ is a grid function

$$u^{h,k} = (u_1^{h,k}, u_2^{h,k}, ..., u_n^{h,k})^T \ , \tag{16}$$

where $u_i^{h,k}$ represents the value of the solution corresponding to the interior grid point $x = a + ih$ of a given grid space Ω_h, at the time step $k\Delta t$. For the discretization of (1) we focus our attention on two facts. First when the diffusion coefficient becomes small, equation (1) tends to coincide with the inviscid Burgers equation which can produce discontinuous solutions [4]. Therefore one must be careful in the discretization of the gradient $\partial_x u$ in the nonlinear term of (1). On the other hand as $\delta \to 0$, any centered discretization of (1) becomes unstable. Hence we use

a noncentered discretization for the nonlinear term because it makes the finite difference scheme stable (independently of the mesh size h). In this way a numerical viscosity arises and there is no need to introduce an artificial viscosity (as it is done in [11] where one has the additional problem of choosing a viscosity parameter). In addition, the so obtained discrete equation is consistent with the same coarser or finer (grid) equation [6]. Then starting from a scheme whose consistency order is 2 the consistency order of the resulting discrete equation is only 1.

Now let us denote with ∂_i^h and ∂_{ii}^h the difference-quotient operators approximating ∂_x and ∂_{xx} respectively. Following [11] we linearize the nonlinear term using a Taylor series expansion with respect to time. This procedure leads to the following discrete problem (for economy of notation we do not put here the discretization parameter h):

$$\frac{u_i^k - u_i^{k-1}}{\Delta t} + \frac{1}{2}\partial_i[u^k u^{k-1}] - \delta \partial_{ii} u^k = 0 \ , \tag{17}$$

with initial condition

$$u_i^0 = u_0(a + ih) \ , \ 1 \leq i \leq n \tag{18}$$

and boundary conditions

$$u_0^k = g_0(k\Delta t) \quad , \quad u_{n+1}^k = g_{n+1}(k\Delta t) \ . \tag{19}$$

For the reason explained above we make the discretization stable by using a backward (or forward) formulae $\partial_i^h \equiv \frac{[-1,1,0]}{h}$ (or $\partial_i^h \equiv \frac{[0,-1,1]}{h}$) and $\partial_{ii}^h \equiv \frac{[1,-2,1]}{h^2}$, which is the so-called difference stencil notation. This produces a $O(\Delta t) + O(h)$ scheme. The use of backward or forward discretization depends on the sign of u. Later we will restrict to the case $u \geq 0$ for which the backward discretization is appropriate. It is obvious that the following results are valid also for the case $u \leq 0$ but in that case they are referred to a forward discretization.

Now notice that (17) is a tridiagonal system of n linear equations for the n unknowns u_i^k , $i = 1, \ldots, n$. That is

$$-(\frac{u_{i-1}^{k-1}}{2h} + \frac{\delta}{h^2})u_{i-1}^k + (\frac{1}{\Delta t} + \frac{u_i^{k-1}}{2h} + \frac{2\delta}{h^2})u_i^k - (\frac{\delta}{h^2})u_{i+1}^k = \frac{u_i^{k-1}}{\Delta t} \ , \tag{20}$$

where u_0^k , u_{n+1}^k are known from the boundary conditions. Let us denote this system by $A(u^{k-1})u^k = f(u^{k-1})$. The boundary conditions are incorporated in the right hand side. Notice that the AMGM described above applies to this algebraic problem.

Now, in order to investigate the above algebraic problem, we introduce the formalism of M-matrices [5]:

Definition 1 *A matrix $A \in L(R^n)$ is an M-matrix if A is invertible, $A^{-1} \geq 0$, and $A_{ij} \leq 0$ for all $i, j = 1, \ldots, n$, $i \neq j$.*

for the following results we need also

Definition 2 *A matrix $A \in L(R^n)$ is irreducible if and only if for any two distinct indices $1 \leq i \leq j \leq n$, there is a sequence of nonzero elements of A of the form $\{a_{ii_1}, a_{i_1 i_2}, \ldots, a_{i_m j}\}$.*

Therefore we can recall a useful theorem for M-matrices [5]

Theorem 1 *Let $A \in L(R^n)$ be irreducibly diagonally dominant and assume $A_{ij} \leq 0$, $i \neq j$, and that $A_{ii} > 0$, $i = 1, \ldots, n$. Then A is an M-matrix.*

We can now state a lemma which tells us a sufficient condition to be satisfied so that the above discretization of (1) gives a matrix $A(u^{k-1})$ which is an M-matrix

Lemma 1 *If $u_i^{k-1} \geq 0$ and $\frac{1}{\Delta t} \geq \max_i \frac{1}{2}|\frac{u_i^{k-1} - u_{i-1}^{k-1}}{h}|$, then $A(u^{k-1})$ is an M-matrix.*

Proof. The condition of (weak) diagonal dominance, where strict inequality holds for at least one i, is expressed by

$$|A(u^{k-1})_{ii}| \geq \sum_{j=1, j\neq i}^{n} |A(u^{k-1})_{ij}| \ , \qquad i = 1, \ldots, n \ . \tag{21}$$

This is immediately satisfied by $A(u^{k-1})$ because

$$\begin{aligned} \sum_{j=1, j\neq i}^{n} |A(u^{k-1})_{ij}| &= \frac{\delta}{h^2} + (\frac{u_{i-1}^{k-1}}{2h} + \frac{\delta}{h^2}) = \frac{2\delta}{h^2} - \frac{u_i^{k-1} - u_{i-1}^{k-1}}{2h} + \frac{u_i^{k-1}}{2h} \\ &\leq \frac{2\delta}{h^2} + |\frac{u_i^{k-1} - u_{i-1}^{k-1}}{2h}| + \frac{u_i^{k-1}}{2h} \leq A(u^{k-1})_{ii} \ , \qquad i = 1, \ldots, n \ . \end{aligned}$$

Then $A(u^{k-1})$ is diagonal dominant. In addition we have $A(u^{k-1})_{ii} > 0$, $i = 1, \ldots, n$ and $A(u^{k-1})_{ij} \leq 0$, $i \neq j$. Moreover, for any $i < j$ there is the sequence $\{a_{ii+1}, a_{i+1i+2}, \ldots, a_{j-1j}\}$ and for $j < i$ we have $\{a_{jj+1}, a_{j+1j+2}, \ldots, a_{i-1i}\}$, then $A(u^{k-1})$ is irreducible. Hence, by theorem 1, $A(u^{k-1})$ is an M-matrix.

¿From now on we will refer to the conditions $u_i^{k-1} \geq 0$ and $\frac{1}{\Delta t} \geq \max_i \frac{1}{2}|\frac{u_i^{k-1} - u_{i-1}^{k-1}}{h}|$, as the M-conditions. We then have a result for our analysis stated in

Lemma 2 *If the M-conditions are satisfied at level $k-1$ and the boundary values are positive or zero then u^k exists and is nonnegative.*

Proof. Because the M-conditions are satisfied $A(u^{k-1})$ is an M-matrix. Then it is invertible and the solution at the time step $k\Delta t$ for the discretization (17) exists and is given by $u^k = A(u^{k-1})^{-1} f(u^{k-1})$. Moreover if $u_i^{k-1} \geq 0, i = 0, \ldots, n+1$, and $u_0^k \geq 0$ and $u_{n+1}^k \geq 0$ one immediately sees that $f(u^{k-1})_i \geq 0$, $i = 1, \ldots, n$. Then the solution u^k is positive thanks to the positive definiteness of inverse $A(u^{k-1})^{-1}$ and of the vector $f(u^{k-1})$.

The last lemma states that starting from a nonnegative initial data the solution will evolve remaining positive if at any time level k the boundary values are nonnegative and $\frac{1}{\Delta t} \geq \max_i \frac{1}{2}|\frac{u_i^{k-1}-u_{i-1}^{k-1}}{h}|$. With these conditions $A(u^{k-1})$ will remain an M-matrix during the evolution.

The above conditions pose a limitation in the step size Δt. Depending on the behaviour of the solution with the time, Δt could be increased or must be reduced during the evolution. In any case the diffusive term in (1) prevents the development of steep wave profiles and tends to spread the sharp discontinuities into smooth profiles so that the step size Δt remains finite. For simplicity we shall consider an example where Δt is fixed. In any case the MNI applies also with variable step sizes [3, 10].

4 Numerical Investigation

We solve the Burgers equation in the domain $\Omega = (a, b)$ with the initial condition

$$u_0(x) = u_1 + \frac{u_2 - u_1}{2}\{1 - \tanh[\frac{u_2 - u_1}{4\delta}x]\} , \tag{22}$$

and boundary conditions $u_1 = u_0(b)$ and $u_2 = u_0(a)$. We choose a and b so that the boundary conditions approximate well the asymptotic values of the steady state solution (together with the asymptotic vanishing of the spatial derivative of u)

$$u(x,t) = u_1 + \frac{u_2 - u_1}{2}\{1 - \tanh[\frac{u_2 - u_1}{4\delta}(x - Ut)]\} , \tag{23}$$

where $U = \frac{u_1+u_2}{2}$. This solution is known as the Taylor shock solution [4], U being the velocity of the shock. Let us assume that $u_1 = 0$. With these conditions the sum of the n discrete equations (17) gives

$$S^k = S^{k-1} + \frac{\Delta t U}{h}u_2 , \tag{24}$$

where $S^k = \sum_{i=1}^n u_i^k$. The term $\frac{\Delta t U}{h}$ can be interpreted as the number of new grid points reached by the shock in one time step. Then (24) means that the evolution of the discrete Burgers equation (17) is that of a Taylor shock satisfying the given initial-boundary conditions.

The numerical experiments are performed for $\delta = 0.01$ and boundary values given by $u_1 = 0$ and $u_2 = 1$, so that $U = 0.5$. The numerical domain is $\Omega = (-2, 10.8)$. The number of the interior points of the finest grid is 127 ($h_M = 0.1$) and $M = 6$, this means that the coarsest space has 3 interior grid points. In all numerical experiments the time step size is $\Delta t = \frac{1}{5}$. We verify numerically that with the above values of the discretization parameters the M-conditions are satisfied during the evolution.

For both FAS and AMGM we use the RB Gauss-Seidel smoothing iteration. Notice that $A(u^{k-1})$ is an M-matrix and this is a sufficient condition for the above iteration to converge to the solution u^k for any starting approximation [7]. Moreover, this iteration is normally recommended to be used in multi-grid methods for the numerical solution of parabolic problems [9]. For the AMGM the use of the RB Gauss-Seidel iteration is motivated by the results given in section 2.

Now we summarize the numerical results. We consider the AMGM at each time level. It employes one sweep of the RB Gauss-Seidel iteration at any level as far as the coarsest one is reached. There the algebraic problem is solved exactly (zero defect) with just few iterations. Then the exact solution is obtained using (11) recursively to go back to the finest level, without any need for a post-smoothing.

On the other hand the FAS algorithm requires also a smoothing iteration when coming back to the finest grid. With one step of the FAS algorithm we remain with a defect of order $\sim 10^{-3}$ and the discrete problem is solved to the order of truncation error.

Moreover, it results that both the modified nested iteration and the repeated application of the algebraic multi-grid method have a good convergence in time: after few time steps the defect reaches its stationary value (but it is zero using AMGM), and the relative solution error with respect to (23) is also stationary of order $\sim 10^{-2}$, but the exact discrete solution is obtained using AMGM. Moreover, the velocity of the resulting shock coincides with the analytical one. However, because the algebraic method does not require a post-smoothing it results to be faster than the FAS algorithm, where 2 sweeps of pre-, and post-smoothing are applied at each space level. In fact we observe that the CPU time used by AMGM is only 36% of the time used by the FAS scheme.

For a comparison of these results with others presented in the literature we refer to [4] and references therein. For example, in [12], the same accuracy of the solution is obtained using an accurate space derivative (ASD) pseudo-spectral approach with values of the discretization parameters ($\Delta t = 0.001$ and $h_M = 0.01$) which are much more severe than ours.

5 Conclusions

In this work we have defined and tested an algebraic multi-grid method for evolution equations of parabolic type. This algorithm is very interesting because it provides the exact discrete solution of the discrete problem, imitating the standard MG solution process. However, the AMGM is itself interesting: all components of the algebraic MG procedure are given explicitly and are not restricted to symmetric problems. This was possible introducing a MG substitution method. In this way we have explained also why the convergence of relaxation is not a crucial property in a multi-grid solver. Actually we have showed that the importance of a smoothing iteration (here we considered Colored Gauss-Seidel relaxation) is to arrange

the defect so that the coarse grid correction solves exactly the discrete equation on the finest grid. Hence, in principle, the AMGM does not require convergence of the smoothing iteration.

We have done some numerical experiments solving the Burgers equation. For the resulting discrete problem we have proved the existence of solutions for a class of initial-boundary conditions. These solutions have been obtained using AMGM and MNI (with FAS) and their accuracy is comparable with that obtained with other methods presented in the literature. However, the values of the discretization parameters used here are much less severe.

A method which is limited to the one-dimensional case is not very interesting. In fact this serves as an introductory work where all features of an algebraic multigrid approach to fluid flow problems are exhibited. The formulation of an identical algebraic algorithm for two dimensional flow problems is in progress.

ACKNOWLEDGEMENT

I am grateful to A. Lanza for continuous encouragement and guidance in doing this work.

References

[1] A. Brandt, *Multi-Level Adaptive Solutions to Boundary-Value Problems. Math. Comp.* **31**, 333 (1977).

[2] A. Brandt, *Algebraic Multigrid Theory: The Symmetric Case. Appl. Math. Comp.* **19**, 23 (1986).

[3] J.W. Ruge and K. Stüben, *Algebraic Multigrid.* In: S.F. McCormick, *Multigrid Methods*, Frontiers in Applied Mathematics, SIAM, Philadelphia (1987).

[4] P.L. Sachdev, *Nonlinear diffusive waves.* Cambridge University Press, Cambridge (1987).

[5] J.M. Ortega, *Numerical Analysis, A Second Course.* Academic Press, New York (1972).

[6] W. Hackbusch, *Multi-Grid Methods and Applications.* Springer, Heidelberg (1985).

[7] D.M. Young and R.T. Gregory, *A Survey of Numerical Mathematics, Vol.II.* Addison-Wesley Publishing Company (1973).

[8] A. Brandt, *Multi-grid techniques: 1984 guide with applications to fluid dynamics.* GMD-Studien. no 85, St. Augustin (1984).

[9] A. Brandt and J. Greenwald, *Parabolic Multigrid Revisited.* In: W. Hackbusch and U. Trottenberg, *Multigrid Methods III*, III. Proc. of the European Conference on Multigrid Methods, Bonn, Oct. 1990, Birkhäuser, Berlin (1991).

[10] W. Hackbusch, *Parabolic Multi-Grid Methods.* In: R. Glowinski and J.L. Lions, *Computing Methods in Applied Sciences and Engineering*, VI. Proc. of the sixth international symposium, Versailles, Dec.1983, North-Holland, Amsterdam (1984).

[11] C.A.J. Fletcher, *Computational Techniques for Fluid Dynamics, Vol.I.* Springer, Heidelberg (1988).

[12] J. Gazdag, *Numerical Convective Schemes Based on Accurate Computation of Space Derivatives. J. Comp. Phys.* **13** 100 (1973).

2

On the Extension of the Twolevel Method for Operator Equations in Hilbert Space

Alfio Borzì[1]

ABSTRACT We present an extension of the so-called twolevel method for operator equations in infinite dimensional Hilbert spaces. After a short review of some iterative methods for problems in this space, we introduce two level spaces using the notion of anti-reduction properties stated here. Hence we give suitable prolongation and restriction operators which map between the levels. By means of these operators we obtain a reduced problem, that is the auxiliary problem on the coarse level, whose solution allows to solve the given operator equation. The last step is explicitly defined and the resulting twolevel method discussed.

1 Introduction

The purpose of this work is to present an extension of the so-called twolevel method, used in numerical analysis, for operator equations in infinite dimensional Hilbert spaces. The motivation of this paper is twofold. First it represents a first step toward the statement of a new iterative method in functional analysis. Second it provides an *explicit* formulation of the twolevel scheme for the solution of a large class of algebraic problems.

The multilevel method originated by a careful study of iterative methods, like Jacobi or Gauss-Seidel [1], and since its efficient formulation in 1977 by A. Brandt [2], it has been applied successfully to solve numerically partial differential equations, integral equations, etc. (see [3], and references therein). The basic multilevel (ML) idea is to treat the different spectral components of the solution error on different discretization spaces. On each space an iterative method is applied to solve those components of the error which are highly oscillating. Consequently, on the given space, the error remains smooth and can be approximately represented and computed on a "coarser" space, solving a "coarse" problem, and the corre-

[1]SISSA, Scuola Internazionale Superiore di Studi Avanzati, Via Beirut 2-4, 34013 Trieste, Italy. Address after 15 November 1993: OUCL, Wolfson Building, Parks Road, Oxford, OX1 3QD, UK

sponding coarse solution is used to correct the finer one for the smooth components of the error. This last step is called coarse level (CL) correction. In the multilevel version, to solve the given problem one employees a CL correction recursively, i.e., the coarse problem is itself solved by iteration combined with a further coarse level correction. However, for simplicity we restrict ourselves to consider only twolevel (TL) methods.

What we have sketched above can be considered the multilevel method in a narrow sense. Indeed, historically, the use of recursive levels was also considered in connection with direct methods [4]. In this approach, called total reduction method, elimination techniques are used which transform the original problem "equivalently" to coarser spaces. That is, one selects a subset of the set of variables which represent the solution on the given numerical domain. Thereby, the problem resulting by the elimination of the remaining variables constitutes the coarse problem. However this alternative point of view was not followed up.

In a previous paper [5], we studied an algebraic twolevel method for tridiagonal matrices. In particular, we gave an algebraic interpretation of the entire TL algorithm. This was possible by defining a particular reduction procedure, and we arrived at the conclusion that the standard twolevel method can be seen as an "approximative" version of a reduction method. Moreover, we obtained to express all TL operators in a formalism which, as we will see, allows to extend the twolevel scheme to problems in infinite dimensional Hilbert spaces.

This formalism is similar to that used by W.V. Petryshyn [6, 7], in order to extend the classical Jacobi and Gauss-Seidel iterations for infinite dimensional Hilbert spaces. In this case, the present work could be considered as a continuation of that, as the TL method represents a development of classical iterative methods. Hence for completeness of presentation, to introduce the formalism, and to explain the connection between the twolevel and the reduction methods, we report the main results regarding the above iterative schemes in section 2.

However, the crucial point in order to extend the twolevel method in infinite dimensional Hilbert spaces, is how to define the different "level" spaces. The standard approach of TL numerical analysis is to define high and low frequency subspaces with respect to which one defines the coarse level. This is not suitable to be applied here since one has infinite frequency components. On the other, also the reduction method seems not immediately applicable. Nevertheless, in section 3, we will present a general approach to this problem which generalizes the two standard method mentioned above.

Having defined a suitable auxiliary space, we need to construct, in correspondence, an operator equation whose resolution helps to derive the solution of the given equation in Hilbert space. This "coarse" problem is presented in section 4. We obtain it following the standard procedure of algebraic multilevel methods, that is the Galerkin approach [3, 8], making use of prolongation and a restriction operators which are suitable for our purpose. It turns out that the solution of the auxiliary problem allows to solve the given equation through a coarse level correction explicitly defined. Our conclusions are presented in section 5.

2 Iterative Methods in Hilbert Space

Let $\mathcal{H}$ be an infinite dimensional Hilbert space. Denote with $A : \mathcal{H} \to \mathcal{H}$ a bounded, continuously invertible, self-adjoint linear operator. Moreover assume it is of the special form $A = D - L - U$ with bounded operators D, L, and U subject to the conditions:

(a) D is self-adjoint;

(b) The operators $G_\omega = \frac{2-\omega}{\omega} D - L^* + U$, are positive definite, i.e., there exists $\beta = \beta(\omega) > 0$ such that

$$(G_\omega u, u) \geq \beta \|u\|^2, \qquad \text{for } \omega \in \Omega \ ,$$

where L^* be the adjoint of L and Ω a set of reals $\omega > 0$;

(c) $(D - \omega L)$ has a bounded inverse defined on all of $\mathcal{H}$ for $\omega \in \Omega$.

Under these conditions it is possible to define an iterative method that determines an approximate solution u_n of the equation

$$Au = f \ , \qquad f \in \mathcal{H} \ , \tag{1}$$

by the iteration

$$u_n = (D - \omega L)^{-1}\{(1-\omega)D + \omega U\}u_{n-1} + \omega(D - \omega L)^{-1} f \ , \tag{2}$$

where u_0 is an arbitrary initial approximation. We denote the *iteration operator* with $T = (D - \omega L)^{-1}\{(1-\omega)D + \omega U\}$.

Let us report now a theorem [6, 7], which states a necessary and sufficient condition so that (2) defines a sequence $\{u_n\}$ which converges to the unique solution of (1).

Theorem 1 *If D, L, U and Ω satisfy the conditions (a), (b) and (c), then the iterative method (2) converges to the unique solution of Eq. (1), for every f in $\mathcal{H}$ and any u_0 in $\mathcal{H}$, if and only if $A = D - L - U$ is positive definite.*

Furthermore, the above conditions allow to estimate a bound for the spectral radius of the iteration operator. It is given by the following theorem [7]:

Theorem 2 *Let D, L, U and Ω satisfy the conditions (a), (b) and (c). Suppose also that $A = D - L - U$ is positive definite. Then the spectral radius $r(T)$ of the operator T satisfies the inequality*

$$r(T) \leq 1 - \frac{\gamma_\omega}{\|D - \omega L\|} \ , \qquad \omega \in \Omega \ ,$$

where

$$\gamma_\omega = \inf_{\|u\|=1} \{|((D - \omega L)u, u)| - |(P(\omega)u, u)|\} > 0 \ ,$$

with $P(\omega) = (1-\omega)D + \omega U$.

Notice that depending on the choice of D, L, U and ω, many different special cases are obtained. For example, if $\omega = 1$, then (2) is the generalization of the Gauss-Seidel iteration, and when $L = 0$ one gets the Jacobi method.

However, there are many other iterative methods for the solution of a linear equation in Hilbert space, and for a beautiful survey of the subject we refer to [9].

3 Two Levels in Hilbert Space

Let us analyze two different ways in order to define level spaces. For example, let us consider the $n \times n$ (n even) tridiagonal matrix A_n, with $a_{ii} = 2$, $i = 1, \dots n$, and $a_{ij} = -1$, $|i - j| = 1$.

We take $D_n = \text{diag}\{a_{11}, a_{22}, \dots, a_{nn}\}$, and with $-L_n$, and $-U_n$ we denote the strictly lower and upper part of A_n. We consider the Jacobi iteration matrix $T_n = D_n^{-1}(L_n + U_n)$. This is obtained from (2) with $\omega = 1$, $L = 0$, and $U = U_n + L_n$. The n eigenvectors of T_n are given by $e^k = (\sin(k\frac{\pi}{n+1}j))_{j=1}^n$, $k = 1, \dots, n$. They correspond to the eigenvalues $\lambda_k = 1 - 2\sin^2(k\frac{\pi}{2(n+1)})$, $k = 1, \dots, n$. Notice that the spectrum of T_n is symmetric with respect to the origin. Now, let us denote by e_+ and e_- any two eigenvectors corresponding to a positive eigenvalue λ and $-\lambda$ respectively. One obtains immediately

$$T_n(e_+ \pm e_-) = \lambda(e_+ \mp e_-) \ , \qquad \lambda > 0 \ .$$

That is $T_n X_\pm = X_\mp$ where $X_\pm = span\{(e_+ \pm e_-)\}$. A similar situation occurs using the algebraic approach [5]. In fact, denote with X_+ and X_- the subspaces of vectors $(u_j)_{j=1}^n$ whose nonzero elements u_j are those with index j even or odd, respectively. Then it is easy to see that $T_n X_\pm = X_\mp$, as before. Notice that in both cases we have also $(L_n + U_n)X_\pm = X_\mp$.

Let us now go back to the infinite dimensional case. We assume from now on that D and $(L + U)$ have a bounded inverse. Following the above examples we consider the existence of two subspaces $\mathcal{H}_\pm$ of $\mathcal{H}$ such that

$$\mathcal{H} = \mathcal{H}_+ \oplus \mathcal{H}_- \ , \tag{3}$$

and

$$(L + U)\mathcal{H}_\pm = \mathcal{H}_\mp \ , \qquad D^{-1}(L + U)\mathcal{H}_\pm = \mathcal{H}_\mp \ . \tag{4}$$

We will refer to (3) and (4) as the *anti-reduction* properties.

Hence, in the coming part of this section we investigate some additional conditions on the operators D and $(L + U)$ so that (3) and (4) hold. First let us prove the following theorem

Theorem 3 *Let $B : \mathcal{H} \to \mathcal{H}$ be a bounded self-adjoint linear operator with bounded inverse such that*

$$B(L + U) + (L + U)B = 0 \ , \qquad \textit{and} \qquad BD = DB \ . \tag{5}$$

Then there exist subspaces $\mathcal{H}_+$ and $\mathcal{H}_-$ of $\mathcal{H}$, which satisfy (3) and $(L+U)\mathcal{H}_\pm = \mathcal{H}_\mp$, and $D^{-1}(L+U)\mathcal{H}_\pm = \mathcal{H}_\mp$.

Proof. Let us define $\mathcal{B} = (\sqrt{B^2})^{-1}B$, and denote $\tilde{T}_1 = (L+U)$ and $\tilde{T}_2 = D^{-1}(L+U)$. First of all we have $BD^{-1} = D^{-1}B$ and $B^2\tilde{T}_i = -B\tilde{T}_iB = \tilde{T}_iB^2$, $i = 1,2$. In particular the square root, as the inverse of the square root, of B^2, commutes with every bounded linear operator which commutes with B^2. This implies that $\mathcal{B}$ has the following properties

$$\mathcal{B}\tilde{T}_i + \tilde{T}_i\mathcal{B} = 0 \ , \ i = 1,2 \ , \text{and} \quad \mathcal{B}^2 = I \ , \quad \sigma(\mathcal{B}) = \pm 1 \ .$$

By means of $\mathcal{B}$, we can construct two projection operators denoted by $P_+ = \frac{1+\mathcal{B}}{2}$, $P_- = I - P_+$, with which we define $\mathcal{H}_\pm = P_\pm\mathcal{H}$. Clearly we have $\mathcal{H} = \mathcal{H}_+ \oplus \mathcal{H}_-$. Moreover, we obtain

$$\tilde{T}\mathcal{H}_\pm = \tilde{T}P_\pm\mathcal{H} = \tilde{T}\frac{1 \pm \mathcal{B}}{2}\mathcal{H} = \frac{1 \mp \mathcal{B}}{2}\tilde{T}\mathcal{H} = \mathcal{H}_\mp \ .$$

This completes the proof.

The example given in the beginning of this section suggests that the existence of such an operator $\mathcal{B}$ has consequences on the spectrum of $(L+U)$. In fact, we prove now a theorem which is useful in order to define the splitting $A = D - L - U$.

Theorem 4 *Let $(L+U) : \mathcal{H} \to \mathcal{H}$ be a bounded, continuously invertible, self-adjoint linear operator. Then there exists a bounded linear operator $\mathcal{B}$, with bounded inverse, which anti-commutes with $(L+U)$ if and only if the spectrum $\sigma(L+U)$ is symmetric, and each two eigenvalues λ and $-\lambda$ have the same multiplicity.*

Proof.

Necessity. Assume there exists an operator $\mathcal{B}$ so that $\mathcal{B}(L+U)+(L+U)\mathcal{B} = 0$. We have that $\lambda \in \sigma(L+U)$ if and only if there exists a sequence of elements $\psi_n \in \mathcal{H}$, $\|\psi_n\| = 1$, so that $\|((L+U) - \lambda)\psi_n\| \to 0$. Now consider the sequence $\phi_n = \mathcal{B}\psi_n/\|\mathcal{B}\psi_n\|$ in correspondence we have $((L+U) + \lambda)\phi_n = -\mathcal{B}((L+U) - \lambda)\psi_n/\|\mathcal{B}\psi_n\|$. That is, in terms of norms

$$\|((L+U) + \lambda)\phi_n\| = \|\mathcal{B}((L+U) - \lambda)\psi_n\|/\|\mathcal{B}\psi\| \leq \frac{\|\mathcal{B}\|}{\|\mathcal{B}\psi_n\|}\|((L+U) - \lambda)\psi_n\| \ .$$

This implies that $\|((L+U) + \lambda)\phi_n\| \to 0$, whenever $\|((L+U) - \lambda)\psi_n\| \to 0$, and $-\lambda \in \sigma(L+U)$ whenever $\lambda \in \sigma(L+U)$. Then $\sigma(L+U)$ is symmetric.

Sufficiency. Suppose $(L+U)$ has a symmetric spectrum. Then its spectral representation becomes

$$(L+U) = \int_0^{\|(L+U)\|} \lambda(dE_{+\lambda} - dE_{-\lambda}) \ ,$$

where $\mathcal{E} = (E_{\pm\lambda})_{\lambda>0}$ is the spectral family associated with $(L+U)$.

Now let us define the operator $\mathcal{B}$ as $\mathcal{B}dE_{\pm\lambda} = dE_{\mp\lambda}$. We obtain

$$\begin{aligned}\mathcal{B}(L+U) &= \mathcal{B}\int_0^{\|(L+U)\|} \lambda(dE_{+\lambda} - dE_{-\lambda}) \\ &= -\int_0^{\|(L+U)\|} \lambda(dE_{+\lambda} - dE_{-\lambda})\mathcal{B} = -(L+U)\mathcal{B}\ .\end{aligned}$$

That is, $\mathcal{B}$ anti-commutes with $(L+U)$, and the proof is complete.

Notice that the two subspaces $\mathcal{H}_\pm$ are in some sense identical. But in the simple case studied in [5], the coarse space chosen there could be identified with either $\mathcal{H}_+$ or $\mathcal{H}_-$. Hence we take (arbitrarily) $\mathcal{H}_+$ as our coarse or auxiliary space, whereas $\mathcal{H}$ is, in the ML terminology, the fine space.

4 The Twolevel Algorithm

The twolevel theory, as well as the total reduction formulation, are essentially based on the same idea. That is an operator equation $Au = f$ on a given space $\mathcal{H}$ is solved first defining an auxiliary problem $A_c u_c = f_c$ on $\mathcal{H}_c \subset \mathcal{H}$, such that u_c represents part of the entire solution u.

This coarse problem is constructed by means of two operators called prolongation and restriction operators, respectively. The first of them will be denoted by $P : \mathcal{H}_c \to \mathcal{H}$, and the second by $R : \mathcal{H} \to \mathcal{H}_c$. Then we can use the Galerkin approach and define $A_c = RAP$ and $f_c = Rf$. Obviously P and R should be in such a way that the coarse level correction gives the approximation $u \simeq Pu_c$.

In order to construct the twolevel method we need to define the prolongation and restriction operators ($\mathcal{H}_c$ being identified with $\mathcal{H}_+$). Actually we assume that R and P given in [5] are suitable in the context of this work and in accordance with our formalism we have:

$$P = \{I + D^{-1}(L+U)\}|_{\mathcal{H}_+}\ , \tag{6}$$

$$R = P_+ + (L+U)D^{-1}P_-\ , \tag{7}$$

which map the right spaces if (3) and (4) are satisfied.

We compute the operator product RAP using the splitting $A = D - L - U$ and the fact that the anti-reduction properties hold. Therefore we obtain

$$A_c = \{D - (L+U)D^{-1}(L+U)\}|_{\mathcal{H}_+}\ . \tag{8}$$

The subspace $\mathcal{H}_+$ is *invariant* under $\tilde{A} = \{D - (L+U)D^{-1}(L+U)\}$, i.e., $\tilde{A}\mathcal{H}_+ \subset \mathcal{H}_+$ [10]. Moreover $\tilde{A}\mathcal{H}_- \subset \mathcal{H}_-$ and $\mathcal{H}_+$ is said to *reduce* $\tilde{A}$. The point is

that we can consider solely the restriction of this operator given by (8). In order to construct a TL procedure we need to prove the invertibility of A_c. This is done by the following

Lemma 3 *Assume that the conditions of theorem 2 are satisfied together with the requirement that both D and $(L+U)$ have bounded inverse and the anti-reduction properties hold. Then A_c^{-1} exists and is bounded.*

Proof. Let us denote $T = D^{-1}(L+U)$. Because of theorem 2 (which applies to T with the substitution $L \to 0$ and $U \to L+U$), the spectrum $\sigma(T)$ of T lies in the interior of the unit circle. Therefore $I - T^2$ is continuously invertible [6]. Further we can rewrite $\tilde{A} = D(I - T^2)$ and the existence and boundedness of $\tilde{A}^{-1}$ follows. This means that $\tilde{A}\mathcal{H}_+ = \mathcal{H}_+$ and A_c^{-1} is given by $\tilde{A}^{-1}|_{\mathcal{H}_+}$.

We have completed the presentation of all components of a twolevel algorithm. With the prolongation and restriction operators, given by (6) and (7), we have obtained a coarse problem which is well defined. It remains to investigate the relationship between the coarse solution $u_c = A_c^{-1} f_c$ and the solution of $Au = f$. For this purpose we give now a theorem which shows this connexion:

Theorem 5 *Assume that the conditions of lemma 3 are satisfied. Denote with $u_c \in \mathcal{H}_+$ the solution of the coarse problem $A_c u_c = f_c$, $f_c = Rf$. Then $u = Pu_c + D^{-1}P_- f$ solves the fine equation (1).*

Proof. Let us apply A to $Pu_c + D^{-1}P_- f$. We have

$$\begin{aligned}
A(Pu_c + D^{-1}P_- f) &= APu_c + AD^{-1}P_- f \\
&= (D - L - U)(I + D^{-1}(L+U))u_c + (D - L - U)D^{-1}P_- f \\
&= Du_c - (L+U)D^{-1}(L+U)u_c + (I - (L+U)D^{-1})P_- f \\
&= \{D - (L+U)D^{-1}(L+U)\}u_c + (I - (L+U)D^{-1})P_- f \\
&= f_c + (I - (L+U)D^{-1})P_- f \\
&= (P_+ + (L+U)D^{-1}P_-)f + (I - (L+U)D^{-1})P_- f \\
&= (P_+ + P_-)f = f \ .
\end{aligned}$$

This theorem shows how a twolevel solution procedure works. First a coarse problem must be defined and solved and second the result has to be "prolongated" to obtain the solution of the given equation. In this form the method could be considered of an extension of the reduction algorithm.

We remark that the exact result given above has been obtained because of the special choice of P and R and thanks to the anti-reduction properties. However, it is interesting to consider some approximation to the method just described. For this purpose let us denote with $d = A\tilde{u} - f$ and $e = \tilde{u} - u$ the *defect* and the *error* relative to the approximation $\tilde{u}$ to the solution u. Obviously the two equations $Au = f$ and $Ae = d$ are equivalent. Now suppose to choose (2) so that the resulting

approximation $\tilde{u}$ gives $d << e$, with respect to some norm. In this case (2) is said to have the *smoothing* property [8]. Therefore we have $e = Pe_c + D^{-1}P\_d \simeq Pe_c$. Then the CL correction becomes $u = \tilde{u} - Pe_c$ and a new approximation to the solution is obtained. This is the standard twolevel method.

From another point of view, one could recognize that the smoothing property makes possible the term $D^{-1}P\_d$ to disappear. That is, the coarse level correction does not depend explicitly on the splitting of which D is part (however P depends on this splitting). In addition, it is possible to define iterative methods equivalent to (2) which are based only on the operator A [9, 11]. These facts seem to suggest the possibility to transcend the splitting formulation followed in this work.

5 Conclusions

Let us summarize here the main points of this paper and take this occasion to add some remarks. Our starting point was the generalization of classical iterative methods for the approximate solution of $Au = f$ on a Hilbert space $\mathcal{H}$. This is motivated by the fact that multilevel methods are a natural development of iteration procedures. In fact the ML algorithms are based on the observation that pure iterations are efficient in solving only part of the spectral components of the solution u. But it is possible to obtain uniform convergence if one handles the different spectral components on different discretization spaces.

Therefore to generalize the ML methods we needed to introduce the concept of levels in Hilbert space. Following [5] we have constructed a subspace $\mathcal{H}_c \subset \mathcal{H}$ which in many respects looks like the coarse level of the ML theory. In particular we have proved that the existence of this space is related to the spectral properties of the Jacobi iterative operator. Then we have explicitly given suitable prolongation and restriction operators relative to $\mathcal{H}_c$. In the multilevel formulation these operators provide, using the Galerkin approach, a well defined "coarse" problem. Finally we have obtained the solution of the fine equation on $\mathcal{H}$ in terms of that of the reduced problem given on the "coarse" space.

All these results, as those relative to the iterative methods, were obtained assuming a particular splitting of the operator A. This fact could give the feeling that it is not easy to find applications of these methods in infinite dimensional Hilbert space. On the contrary one should notice that this machinery applies, for example, to the interesting class of the Fredholm integral equations of the second kind with symmetric kernels ($D = I$, $U = L^*$). In addition these results apply to a large part of the class of *consistently ordered* matrices [1], because of the spectral properties of the associated Jacobi iteration. Notice that these matrices are very important in applications since the discretization of boundary value problems leads naturally to them.

ACKNOWLEDGEMENTS

We wish to thank Prof. F. Strocchi, for his help in proving the results of section 3, and E. Aldrovandi, C. Grosche and A. Lanza for many valuable comments.

References

[1] J. Stoer and R. Bulirsch, *Introduction to Numerical Analysis.* New York, Springer-Verlag (1980).

[2] A. Brandt, *Multi-Level Adaptive Solutions to Boundary-Value Problems. Math. Comp.* **31**, (1977) 333-390.

[3] W. Hackbusch, *Multi-Grid Methods and Applications.* Heidelberg, Springer-Verlag (1985).

[4] J. Schröder, U. Trottenberg and K. Witsch, *On the Fast Poisson Solvers and Applications. Numerical Treatment of Partial Differential Equations. Lecture Notes in Mathematics* **631**. Berlin, Springer-Verlag (1978).

[5] A. Borzì, *Algebraic Twolevel Methods and Tridiagonal $\mathcal{M}$-Matrices.* SISSA-ISAS 28/93/FM (1993). Submitted to *IMA Journal of Numerical Analysis.*

[6] W.V. Petryshyn, *On the Generalized Overrelaxation Method for Operator Equations. Proc. Amer. Math. Soc.* **14**, (1963) 917-924.

[7] W.V. Petryshyn, *Remarks on the Generalized Overrelaxation and the Extrapolated Jacobi Methods for Operator Equations in Hilbert Space. J. Math. Anal. Appl.* **29**, (1970) 558-568.

[8] A. Brandt, *Algebraic Multigrid Theory: The Symmetric Case. Appl. Math. Comp.* **19**, (1986) 23-56.

[9] W.M. Patterson, 3rd *Iterative Methods for the Solution of a Linear Operator Equation in Hilbert Space - A Survey. Lecture Notes in Mathematics* **394**. Heidelberg, Springer-Verlag (1974).

[10] E. Kreyszig, *Introductory Functional Analysis with Applications.* New York, John Wiley & Sons (1978).

[11] H. Bialy, *Iterative Behandlung Linearer Funktionalgleichungen. Arch. Rational Mech. Anal.* **4**, (1959) 166-176.

3

Multigrid Convergence Acceleration for the 2D Euler Equations Applied to High-lift Systems

K.M.J. de Cock[1]

ABSTRACT In this paper, a multigrid convergence acceleration technique for the 2D Euler equations is described. Applications to high-lift flows are made. The multigrid method has been tested for this type of flows.

1 Introduction

In the last two years, at NLR a feasibility study of turn-around time reduction in CFD methods for complicated geometries has been conducted. It has been found that the main reduction of turn-around time can be obtained by automating the grid generation. Unstructured grid methods allow to attain a higher level of automation of the grid generation than structured grids, especially for the complicated shapes encountered in high-lift devices or iced airfoils.

However, the best unstructured grid flow solvers are in general more expensive to use than the best structured grid flow solvers. This is due to the indirect addressing which is inherent to the use of unstructured grids.

For both the structured and the unstructured grid flow solvers, convergence acceleration techniques are often necessary to reduce the computing time of the basic single grid flow solvers for which the efficiency reduces drastically with increasing number of grid nodes. In the literature, various convergence acceleration techniques for the Euler and Navier Stokes equations have been reported. These can be used and combined to improve the performance of flow solvers, see for instance ref. [1]. So, even in case a powerful computer is available, additional turn-around time reduction can be obtained by implementing one or more convergence accelerators. In this paper, a Full Approximation Scheme has been adopted. The price for the computing time reduction by using a multigrid method is an increase

[1]National Aerospace Laboratory (NLR), Anthony Fokkerweg 2, 1059 CM Amsterdam, The Netherlands

in memory use. The present flow solver uses fully nested multigrid levels, so in that case the multigrid workspace in two dimensions is typically 33% larger than the single grid workspace.

2 Grid generation

2.1 Generation of the lowest grid level by successive refinement.

In this section the single grid generator is described briefly. A more extensive description can be found in ref. [3]. The grid generation method is based on successive refinement of some initial grid. The method has much in common with the quad- and octree grid generation approach of ref. [2]. The following assumptions are made.

Assumption 1 *Let $\mathcal{M}$ be a given geometry around which the flow is to be calculated. Let the geometry $\mathcal{M}$ consist of i non-intersecting contours $\mathcal{K}^i$. Let each contour $\mathcal{K}^i$ consist of j_i segments $\mathcal{S}^{j_i}$.*

Assumption 2 *Let $\mathcal{S}_a^{j_i}$ be the analytical representation of a segment $\mathcal{S}^{j_i}$. This analytical representation of segment $\mathcal{S}^{j_i}$ can be used to generate k_{j_i} points $\mathcal{P}^{k_{j_i}}$ on segment $\mathcal{S}^{j_i}$. Let $\mathcal{S}_s^{j_i}$ be the spline representation of a segment $\mathcal{S}^{j_i}$ based on the points $\mathcal{P}^{k_{j_i}}$.*

Assumption 3 *Let all spline representations $\mathcal{S}_s^{j_i}$ either be modified or supplemented with extra splines, such that the contours $\mathcal{K}^i$ are closed.*

Assumption 4 *Let an initial grid $\mathcal{G}$ be given. It can either be a default grid consisting of 7 nodes (start from scratch) or a previously generated grid (restart). Let $\mathcal{G}$ cover the geometry $\mathcal{M}$.*

Assumption 5 *Let $\mathcal{S}_g^{j_i}$ be the momentary polygonal representation of a segment $\mathcal{S}^{j_i}$ based on the points of the grid $\mathcal{G}$ that lie on the spline representation $\mathcal{S}_s^{j_i}$ at a particular moment.*

Assumption 6 *Let $\mathcal{F}$ be a given flow solution on grid $\mathcal{G}$. In the case at hand $\mathcal{F}$ is the solution of the two dimensional Euler equations.*

In order to ensure a succesful and reliable simulation of the flow around a given geometry $\mathcal{M}$, with a given flow solver, some requirements of the grid $\mathcal{G}$ can be identified *a priori*.

Requirement 1 *The far field boundary of the grid $\mathcal{G}$ should lie sufficiently far away from the geometry $\mathcal{M}$ (for instance 30 chord lengths).*

Requirement 2 *The resolution of the grid $\mathcal{G}$ should highest near to the geometry $\mathcal{M}$.*

Requirement 3 *Each node of the grid $\mathcal{G}$ should be nearly centered in its surrounding finite volume.*

Requirement 4 *All momentary polygonal representations $\mathcal{S}_g^{j_i}$ should at least consist of two line pieces.*

Requirement 5 *The grid spacing ΔS along the geometry $\mathcal{M}$ should be sufficiently small such that the required resolution is obtained. The following condition should hold*

$$\left(\frac{\Delta S}{S}\right)_{maximum} < \left(\frac{\Delta S}{S}\right)_{prescribed}. \tag{1}$$

Requirement 6 *The grid curvature along the geometry $\mathcal{M}$ should be sufficiently small such that the expected gradients can be represented accurately. If a,b and c are three subsequent points on the momentary polygonal representation $\mathcal{S}_g^{j_i}$ then the following condition should hold*

$$\frac{\overrightarrow{ab}}{\|\overrightarrow{ab}\|} \cdot \frac{\overrightarrow{bc}}{\|\overrightarrow{bc}\|} > \cos\alpha_{prescribed}. \tag{2}$$

Requirement 7 *All segments $\mathcal{S}_g^{j_i}$ should form closed and non-intersecting contours with the same topology as the segments $\mathcal{S}^{j_i}$.*

The idea behind the current grid generation algorithm and its advantages can be summarized as follows

- *Grid generation* is considered as adaptation of a given grid $\mathcal{G}$ with respect to the properties of the momentary *polygonal* representations $\mathcal{S}_g^{j_i}$ of the segments $\mathcal{S}^{j_i}$ of the geometry $\mathcal{M}$. The only user interaction is specifying the prescribed quantities in requirements 1, 5 and 6.
- *Grid adaptation* is considered as adaptation of a given grid $\mathcal{G}$ with respect to the properties of the flow solution $\mathcal{F}$ around the geometry $\mathcal{M}$. Extension of the grid generation to grid adaptation is straightforward.

So, following these basic ideas, the lowest grid level is generated by recursion of the following algorithms :

Algorithm 1 *Assignment of an "in-contour" flag.*

Algorithm 2 *Construction of the momentary polygonal representation $\mathcal{S}_g^{j_i}$ of the segments $\mathcal{S}^{j_i}$, using the "in-contour" flags.*

Algorithm 3 *Control of the gridgenerator based on requirements 1 to 7 and the momentary polygonal representation $\mathcal{S}_g^{j_i}$. This algorithm decides to stop the grid generation or assigns refinement flags.*

Algorithm 4 *A refinement algorithm for flagged nodes of the grid $\mathcal{G}$.*

Algorithm 5 *A smoother to attain requirement 3.*

Algorithm 6 *A projection algorithm to project nodes of the momentary polygonal representation $\mathcal{S}_g^{j_i}$ on to the spline representation $\mathcal{S}_s^{j_i}$ of the segments $\mathcal{S}^{j_i}$.*

2.2 Generation of the higher grid levels

The different grid levels involved in the multigrid cycle are fully *nested*. This means that when going from a coarse grid level to a finer level, a new node is inserted on all edges of the coarse grid. This choice is made in order to avoid additional interpolation data structures for the prolongation operator to finer grid levels.

The disadvantage of fully *nested* grids is the direct link between the properties of the fine mesh and the properties of the coarsest mesh. If a clustering of grid nodes is wanted in the fine mesh, this clustering should also be present in the coarse mesh. With respect to the efficiency of the multigrid method, the coarse mesh should be as coarse as possible. Clearly, these requirements are conflicting. Crucial for making this choice is the use of the multigrid method. Multigrid methods can be seen as a tool to make for instance grid refinement studies feasible with respect to the computational time (the result of such a study can be found figures 17 and 18).

3 Euler solver

3.1 Single grid solver

The current flow solver is based on the steady two dimensional Euler equations. These equations can be written in the *primitive* variables ξ,

$$\xi = [\rho, u, v, p]^t, \tag{3}$$

as follows :

$$\frac{\partial}{\partial x}\begin{bmatrix} \rho u \\ \rho uu + p \\ \rho uv \\ \rho u\left(\frac{\gamma p}{(\gamma-1)\rho} + \frac{1}{2}\left(u^2+v^2\right)\right) \end{bmatrix} + \frac{\partial}{\partial y}\begin{bmatrix} \rho v \\ \rho vu \\ \rho vv + p \\ \rho v\left(\frac{\gamma p}{(\gamma-1)\rho} + \frac{1}{2}\left(u^2+v^2\right)\right) \end{bmatrix} = 0. \tag{4}$$

In equations 3 and 4, ρ is the density, p is the static pressure, and u and v are the Cartesian x and y components of the velocity vector in the two dimensional physical space and γ is the specific heats ratio. In what follows, ξ is called the vector of the *primitive* variables (in the four dimensional solution space). Equation 4 is supplemented by the perfect slip boundary condition on solid walls,

$$n_x u + n_y v = 0, \tag{5}$$

and the far field boundary condition, making use of prescribed values for

$$M_\infty, \alpha_\infty, \rho_\infty, p_\infty. \tag{6}$$

In equation 5, n_x and n_y are the Cartesian components of the unit normal to the solid wall.

Equations 4, 5 and the far field boundary equation using the prescribed values 6 are to be solved for the vector of *primitive* variables ξ. In the grid generation phase, the physical space is divided into triangles, as in figure 1. The corresponding finite volumes are formed by connecting the centroids of the triangles. Most of these finite volumes have a hexagonal shape. For each grid node, the vector of *primitive* variables ξ is stored. The grid nodes are lying approximately in the center of the surrounding finite volume so the present spatial discretisation can be called *vertex centered.* The unknowns, ξ_i for each grid node, i should obey equations 4 expressed for the finite volume Ω_i (taking into account the solid wall boundary condition 5 or the far field boundary condition using the prescribed values 6, when i is a boundary node) :

$$\int_{\Omega_i} \left(\frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} \right) d\Omega = 0. \tag{7}$$

The finite volume Ω_i is bounded by the surface S_i. In equation 7, F and G are the two components of the flux vector. The unit outward normal of surface S_i is called $\overrightarrow{n}$ with components n_x and n_y. Applying the Gauss Theorem, eq. 7 becomes :

$$\int_{S_i} (F n_x + G n_y)\, dS = 0. \tag{8}$$

We shorten the last equation 8 as :

$$\int_{S_i} N dS = 0, \tag{9}$$

defining the normal flux function $N = F n_x + G n_y$ at the surface S_i of the control volume Ω_i. The normal flux function N is to be approximated at the surface S_i, using the vector of *primitive* variables ξ in the neighbourhood of node i. Within the finite volume, the vector of *primitive* variables $\xi\,(x, y)$ can be reconstructed as

$$\xi\,(x, y) = \xi\,(x_i, y_i)\,, \tag{10}$$

which will finally result in a first order accurate scheme, or as

$$\xi\,(x, y) = \xi\,(x_i, y_i) + \left.\frac{\partial \xi}{\partial x}\right|_i (x - x_i) + \left.\frac{\partial \xi}{\partial y}\right|_i (y - y_i)\,, \tag{11}$$

giving a second order accurate scheme, according to the MUSCLE approach (ref. [5]). Van Albada limiting (ref. [6]) is applied in calculating the gradient in equation 11. An upwind flux difference discretization of the normal flux function N is

adopted in the present code,

$$N = \frac{1}{2}\left[N\left(\xi_L\right) + N\left(\xi_R\right)\right] - \frac{1}{2}\left|A\left(\xi_L, \xi_R\right)\right|\left(\xi_R - \xi_L\right), \tag{12}$$

in which the notations L and R indicate the left and right state vector ξ. Equation 12 can be rewritten as

$$N = N\left(\xi_L\right) + A^{-}\left(\xi_L, \xi_R\right)\left(\xi_R - \xi_L\right), \tag{13}$$

in which A^{-} is the part of the discrete Jacobian of the normal flux with respect to the the vector of *primitive* variables, ξ, having only negative eigenvalues. This discrete Jacobian is based on the polynomial character of the normal flux with respect to the primitive variables ξ, as introduced in ref. [4] . Introducing equation 13 into equation 9 for the first order accurate scheme, finally results in

$$\left(\sum_j A_{ij}^{-} \Delta S_{ij}\right) \widetilde{\xi}_i^n = \sum_k A_{ik}^{-} \Delta S_{ik} \xi_k^n. \tag{14}$$

In equation 14, summations are made over all neighbours of node i, and n indicates the iteration level. The system can be solved by a point relaxation method. Here a point Jacobi relaxation is adopted for vectorization reasons,

$$\xi_i^{n+1} = \omega \widetilde{\xi}_i^n + (1 - \omega)\xi_i^n. \tag{15}$$

The second order accurate counterpart of equation 14 cannot be solved by a point relaxation method, unless the second order accurate part is introduced as a defect correction of the first order accurate equation 14, see ref. [7].

3.2 Multigrid solver

The Full Approximation Scheme is used, meaning that on all grid levels an approximation of the original equations is made. Due to the way the multigrid levels are generated, the prolongation reduces to linear interpolation. The restriction and projection operators are full weighting operators for internal nodes and injection on the boundaries. The sum of the weights is one for the projection operator and two for the restriction of defects, compensating for the fact that going from a fine to a coarse grid the surface of the finite volume doubles. For the injection of defects on the boundaries also a factor two is used. A suitable initial guess of the solution on the finest grid level is found by nested iteration. The multigrid iteration consists of V-Cycles, with two pre- and one postrelaxation. The convergence history of the multigrid method is expressed in *work units*. One *work unit* is defined as the work needed to perform one relaxation on the highest grid level. The work involved in the calculation of defects is taken as one *work unit*, although it is cheaper than performing one relaxation. To compensate for this, we do not account for the other operators.

Name	# nodes	% boundary nodes	max. spacing	max. curvature
M1	1916	18.2	3.8728E-2	4.5235E-2
N1	3697	18.4	2.0209E-2	9.6318E-4
P1	5170	18.7	4.9102E-3	2.8921E-3

TABLE 1. Description of the grids : Coarsest grid levels

Name	# grid nodes	% boundary nodes	max. aspect ratio
M2	7297	9.6	36.5
N2	14088	9.7	28.2
P2	19697	9.8	6.1
M3	28448	4.9	60.3

TABLE 2. Description of the grids : second and third grid levels

4 Results

The performance of the current multigrid method will be analysed in the context of a typical multi element (or *high-lift*) application. For this type of geometries, usually a large variety of conditions (Mach,α) require study. Also a large number of slat and flap settings need to be considered. Within a numerical simulation of the inviscid flow around a multi element airfoil, the flow can be complicated (recirculations and/or small supersonic regions can exist). Hence, this application of multigrid methods is relevant for aircraft industry. In this section, the influence of the flow conditions, the grid and some multigrid parameters on the performance of the present multigrid scheme is investigated.

First, the different grids used are introduced. In figure 1, the M1 grid around the slat, wing and flap of the NLR422 three element airfoil is shown. A view of the P1 grid around the slat is shown in figure 2. Properties of the coarsest grid levels are given in table 1. The difference between grid M1 and N1 is mainly due to the grid curvature, while grids N1 and P1 differ mostly with respect to the grid spacing. Remark that these grids have almost the same percentage of grid nodes on the boundaries, so the resolution on the boundaries increases going from grid M1 to grid P1. From these grids, a second grid level is derived by global refinement. Some characteristics of these grids M2, N2 and P2 are given in table 2. Also a third grid level M3 is derived whose characteristics can be found in the same table.

α	-5.0	0.0	5.0	10.0	15.0	20.0
Name	M2.1	M2.2	M2.3	M2.4	M2.5	M2.6

TABLE 3. Calculated cases : Influence of angle of attack

M_∞	0.15	0.20	0.25
Name	M2.7	M2.8	M2.9

TABLE 4. Calculated cases : Influence of free stream Mach number

4.1 Influence of the angle of attack

The high lift flows considered usually involve a wide range of angles of incidence. For the present configuration, a number of angles of attack have been analysed, as listed in table 3. Some flows have been more difficult to compute than others depending on the flow phenomena involved. For an angle of attack α of -5 degrees a large recirculation filling the slat cove is found, while for α equal to 20 degrees almost no recirculation is present. In the later case, the compressibility effects on the slat are important. These effects can be observed in the Mach number distributions on the slat, shown in figure 13. The corresponding convergence histories are shown in figures 3 and 4. Figure 3 illustrates that the residual reduction for a fixed number of *work units* can differ up to one order of magnitude due to the change in angle of attack.

4.2 Influence of the free stream Mach number

For high values of the free stream Mach number compressibility effects become more important. This fact can be observed in figure 14, showing the Mach number distribution on the slat for the calculated cases of table 4. Especially in case M2.9, a small supersonic zone exists on the slat, terminated by a shock. The convergence histories depicted in figures 5 and 6 shown that for the most extreme case, M2.9, the maximum residual of the mass equation is oscillating. This effect disappears when calculating the same flow condition on the P2 grid; so it is likely to be related to the grid resolution in the neighbourhood of the supersonic zone. Apart from this grid related effect, it seems that the influence of the Mach number on the convergence properties of the scheme is small.

4.3 Influence of the coarsest grid level

Three types of grids are generated, using the available parameters of the grid generator. Some properties can be found in table 1 and 2. The lowest grid level

Grid family	M	N	P
Name	M2.6	N2.6	P2.6

TABLE 5. Calculated cases : Influence of the coarsest grid level

ω	0.4	0.5	0.6	0.7	0.8	0.9
Name	M2.10	M2.11	M2.12	M2.13	M2.14	M2.15

TABLE 6. Calculated cases : Influence of the relaxation factor

determines the family of nested grids derived from the lowest grid level by global refinement. This means that the properties of the lowest grid level will be reflected in the higher levels. On the highest level, the grid should have some properties such as a suitable density of grid points in regions of high curvature. So the lowest grid level should also have a higher density of grid points in those regions. It can be expected that this property of the lowest grid level affects the smoothing properties and so the convergence acceleration. This effect is investigated, for the calculated cases found in table 5. The angle of attack is equal to 20 degrees and the free stream Mach number is equal to 0.20.

The solutions on the different grids are compared in fig. 15, showing the Mach number distribution on the slat. Due to the finer grid at the slat nose, the Mach number behaves more inviscidly for case N2.6 than for case M2.6. This effect is also illustrated in figure 16. The convergence histories of figures 7 and 8 show that the finer the coarsest grid level, the lower the convergence rate.

4.4 Influence of the relaxation factor

The relaxation factor is directly related to the smoothing properties of the relaxation method. So it is expected to have a large influence on the convergence rate of the multigrid method. In table 6, some testcases are defined. The angle of attack is equal to 10 degrees and the free stream Mach number is equal to 0.20 while the relaxation factor, ω, is varied. In figure 9 and 10 the results are shown. The fastest convergence is obtained for a relaxation factor of 0.9.

4.5 Influence of the number of gridlevels

Two computations on the M3 grid are compared, see table 7. The flow conditions are the same as for case M2.3, while the number of grid levels used, is respectively 2 and 3. The convergence histories are compared in figure 11 and 12. It is clear that, for the same amount of *work units*, the overall convergence of case M3.33 (incorporating three grid levels) is faster than for case M3.32 (using only the two

number of gridlevels	2	3
Name	M3.32	M3.33

TABLE 7. Calculated cases : Influence of the number of grid levels

highest grid levels).

5 Conclusions

A multigrid convergence acceleration technique for the two dimensional Euler equations has been implemented and tested. The different grid levels involved in the multigrid cycle are derived from the coarsest level by nesting. The influence of the flow conditions, the grid and multigrid parameters on the performance of the multigrid method has been investigated. It seems that for complicated flows, due to numerical transient effects, the overall residual reduction of the multigrid method after a fixed number of *work units* depends on the flow conditions and the coarsest grid level used.

ACKNOWLEDGEMENTS

The results presented here are obtained by methods developed under contract 01308N of the Netherlands Agency for Aerospace Programs (NIVR). The author wishes to thank Dr. F. Brandsma and C. Tovee for their constructive revisions of this paper.

References

[1] V. Venkatakrishnan, D. Mavriplis, "Implicit solvers for unstructured meshes", ICASE Report No. 91-40, 1991.

[2] D. Moore, J. Warren, "Adaptive mesh generation I : Packing space", submitted to The Internal Journal of Computational Geometry and Applications, 1990.

[3] K. de Cock, "High-Lift system analysis method using unstructured meshes", NLR TP 92351, 1992.

[4] E. Dick, "A flux-difference splitting method for steady Euler equations", J. Comp. Phys., 76,19-32, 1988.

[5] B. Van Leer, "Towards the ultimate conservation difference scheme V, a second order sequel to Godunov 's method", J. Comp. Phys., 32, 101-136, 1979.

[6] G. Van Albada et al., "A comparative study of computational methods in cosmic gas dynamics", Astron. Astrophys.108, 76-84, 1982.

[7] P. Hemker, "Defect correction and higher order schemes for the multigrid solution of the steady Euler equations", Lecture notes in mathematics, 1228, 149-165, 1986.

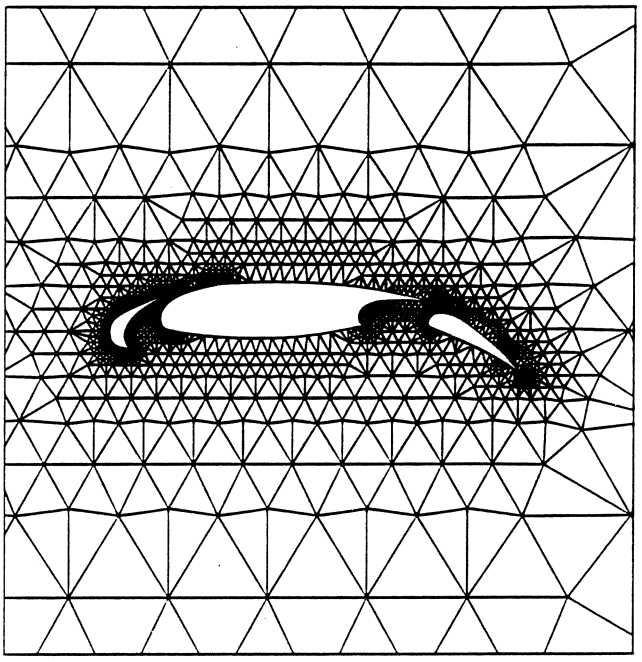


FIGURE 1. NLR422 three element airfoil : Global view of the M1 grid

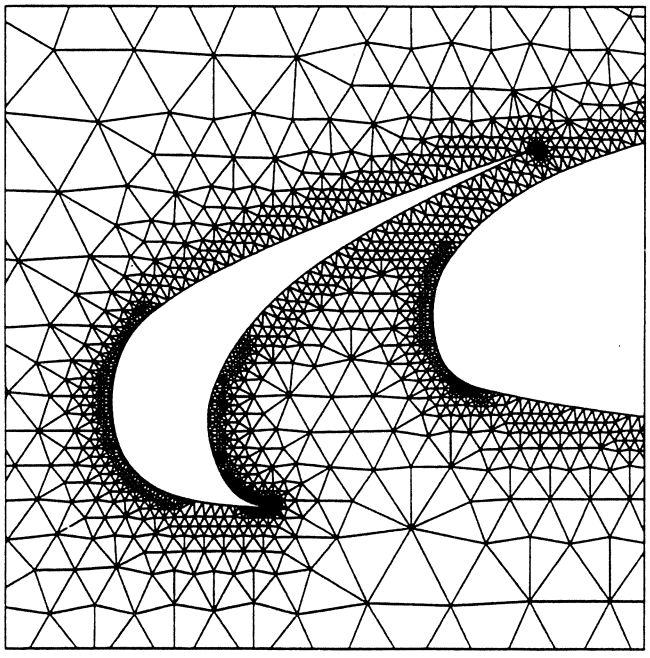


FIGURE 2. NLR422 three element airfoil : View of the P1 grid around the slat

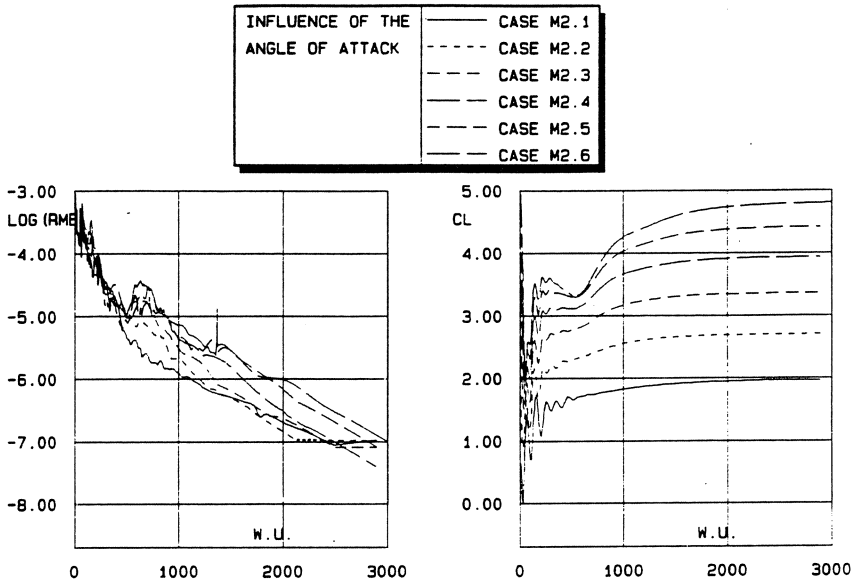


FIGURE 3. Influence of the angle of attack : Convergence history, Logarithm of the maximum residual of the mass equation vs. Work Units.

FIGURE 4. Influence of the angle of attack : Convergence history, Lift coefficient of the total configuration vs. Work Units.

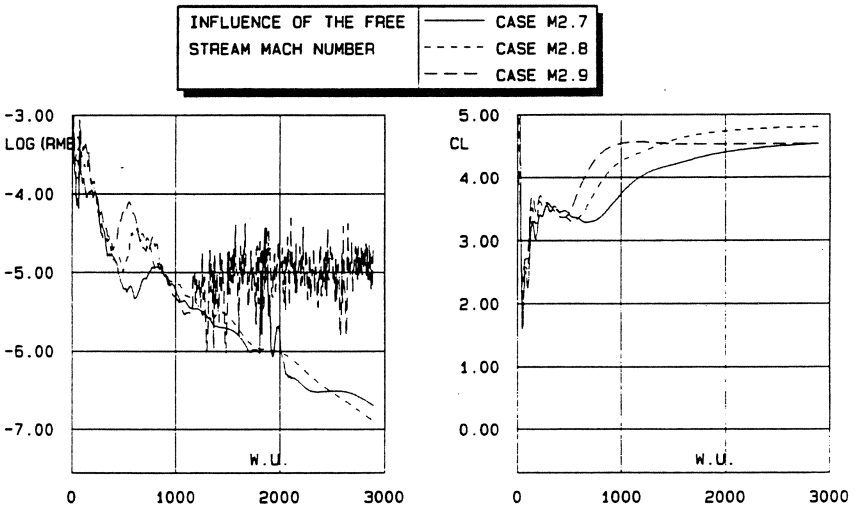


FIGURE 5. Influence of the free stream Mach number : Convergence history, Logarithm of the maximum residual of the mass equation vs. Work Units.

FIGURE 6. Influence of the free stream Mach number : of the total configuration vs. Work Units.

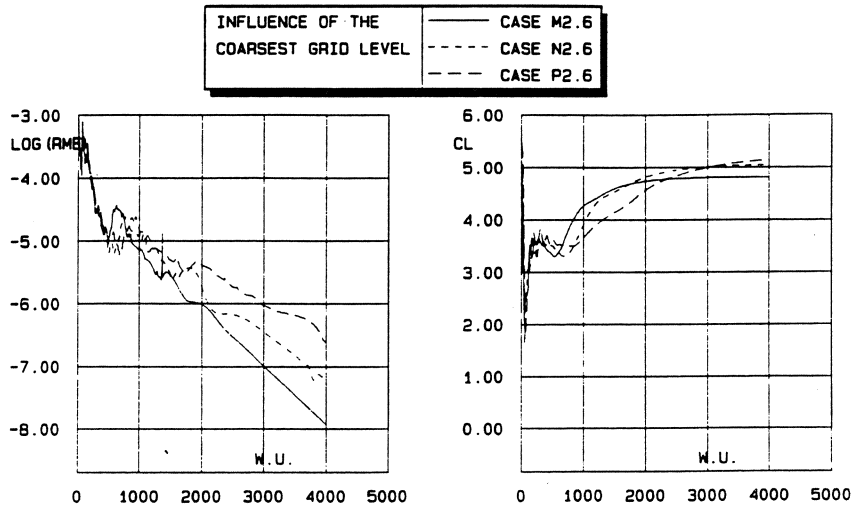


FIGURE 7. Influence of the coarsest grid level : Convergence history, Logarithm of the maximum residual of the mass equation vs. Work Units.

FIGURE 8. Influence of the coarsest grid level : Convergence history, Lift coefficient of the total configuration vs. Work Units.

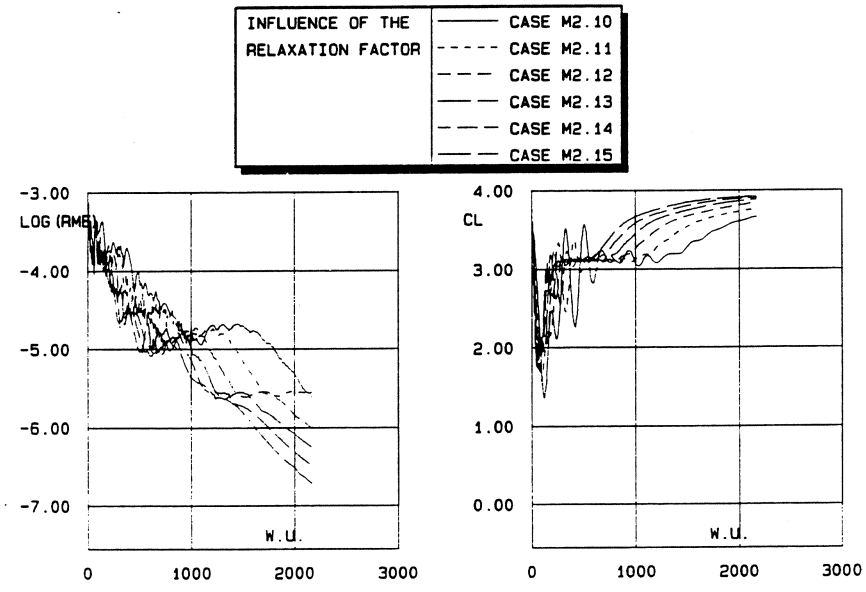


FIGURE 9. Influence of the relaxation factor : Convergence history, Logarithm of the maximum residual of the mass equation vs. Work Units.

FIGURE 10. Influence of the relaxation factor : Convergence history, Lift coefficient of the total configuration vs. Work Units.

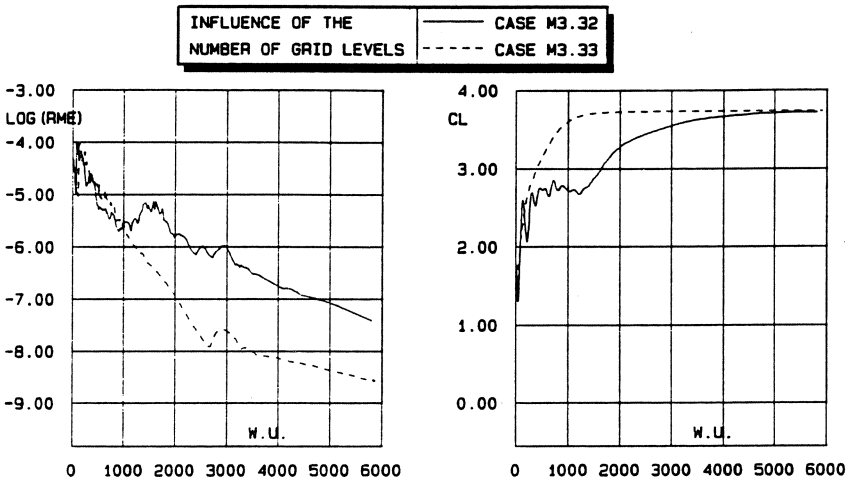


FIGURE 11. Influence of the number of grid levels : Convergence history, Logarithm of the maximum residual of the mass equation vs. Work Units.

FIGURE 12. Influence of the number of grid levels : Convergence history, Lift coefficient of the total configuration vs. Work Units.

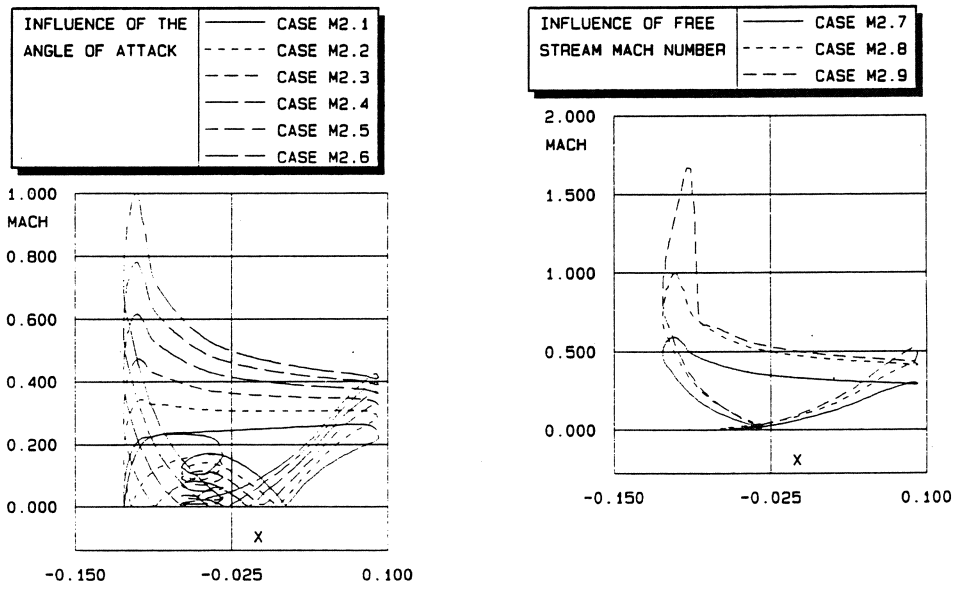


FIGURE 13. Influence of the angle of attack : Mach number on the slat vs. X-coordinate.

FIGURE 14. Influence of the free stream Mach number : Mach number on the slat vs. X-coordinate.

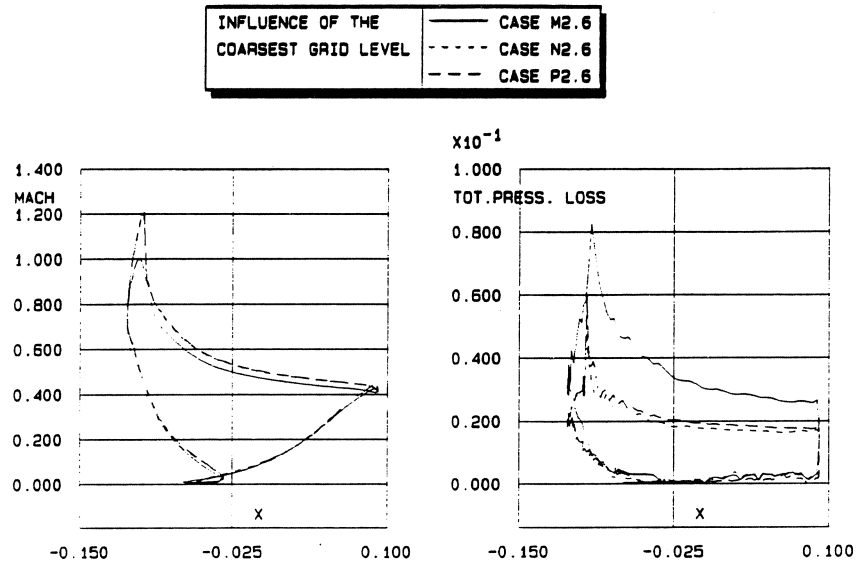


FIGURE 15. Influence of the coarsest grid level : Mach number on the slat vs. X-coordinate.

FIGURE 16. Influence of the coarsest grid level : Total pressure losses on the slat vs. X-coordinate.

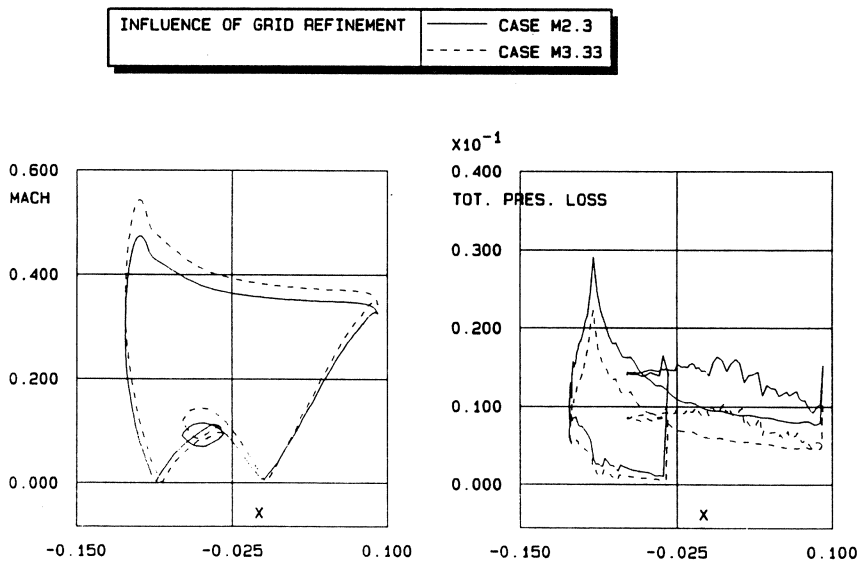


FIGURE 17. Influence of grid refinement : Mach number on the slat vs. X-coordinate.

FIGURE 18. Influence of grid refinement : Total pressure losses on the slat vs. X-coordinate.

4

A Multigrid Method for Solving the Boussinesq Approximation of the Navier-Stokes Equations

O. Dorok, F. Schieweck and L. Tobiska[1]

1 Introduction

Let us consider the Boussinesq approximation of the Navier-Stokes equations

$$\begin{aligned} -\frac{1}{Re}\Delta u + u\cdot\nabla u + \nabla p &= \frac{Ra}{Re^2 Pr} f(T) \quad \text{in} \quad \Omega \\ \nabla\cdot u &= 0 \quad \text{in} \quad \Omega \\ -\frac{1}{RePr}\Delta T + u\cdot\nabla T &= 0 \quad \text{in} \quad \Omega \end{aligned} \tag{1}$$

under the following boundary conditions

$$u_{|\Gamma} = 0, \quad T_{|\Gamma_D} = T_D, \quad \frac{\partial T}{\partial n}\Big|_{\Gamma_N} = 0 \tag{2}$$

in a bounded domain $\Omega \subset \mathbb{R}^d$, $d \in \{2,3\}$ with Lipschitz continuous boundary $\partial\Omega = \Gamma = \Gamma_D \cup \Gamma_N$. We have used the notations Re,Pr and Ra for the Reynolds-, Prandtl- and Rayleigh-number, respectively. The function $f : \mathbb{R} \to \mathbb{R}^2$ is assumed to be linear. In our case there is no natural choice of a reference velocity and thus of a Reynolds number. Therefore, we choose in the numerical calculations $Re = Ra^{1/2}/Pr$. For simplicity we discuss here the case of homogenous boundary conditions

$$u_{|\Gamma} = 0, \quad T_{|\Gamma_D} = 0, \quad \frac{\partial T}{\partial n}\Big|_{\Gamma_N} = 0$$

and introduce the Sobolev-spaces $V = H_0^1(\Omega)^d$, $Q = L_0^2(\Omega)$, $Z = \{z \in H^1(\Omega) : z_{|\Gamma_D} = 0\}$. The corresponding weak formulation of this problem reads:

Find $[u,p,T] \in V \times Q \times Z$ such that for all $[v,q,z] \in V \times Q \times Z$

$$\nu(\nabla u, \nabla v) + (u\cdot\nabla u, v) - (p, \nabla\cdot v) = (\alpha f(T), v)$$

[1]Institut für Analysis und Numerik, TU "Otto von Guericke" Magdeburg
Postfach 4120, 39016 Magdeburg, Germany

$$\begin{aligned} (q, \nabla \cdot u) &= 0 \\ \kappa(\nabla T, \nabla z) + (u \cdot \nabla T, z) &= (g, z) \end{aligned} \tag{3}$$

where ν and κ denote the inverse of the Reynolds and Peclet number, respectively, and $\alpha = \nu\kappa Ra$. In [BMPT91] it is proved that the problem (3) admits at least one solution which is unique provided that the data of the problem are sufficiently small.

We are interested in numerical methods for solving (3) for a large range of Ra. For this there are at least two difficulties for designing a stable finite element method for the class of problems considered above:

- In order to fulfil the Babuška-Brezzi condition we can not use arbitrary pairs of finite element spaces for the approximation of velocity and pressure, respectively.
- In the case of high Rayleigh numbers the problem becomes singularly perturbed and the standard discretization leads to oscillations unless the mesh is very fine.

In the next section we describe a nonconforming finite element method satisfying the Babuška-Brezzi condition and combine it with an upstream technique analyzed in [ST89] for the isothermal case. Section 3 is devoted to the multigrid method for solving the nonlinear algebraic system of equations. Finally, we discuss some numerical experiments in Section 4.

2 Discretization by a nonconforming finite element method of upstream type

In order to get a stable discretization we start with the nonconforming Crouzeix-Raviart element satisfying the discrete version of the Babuška-Brezzi condition [GR86]. Each component of the velocity field and the temperature field are approximated by piecewise linear functions and the pressure is assumed to be piecewise constant. This finite element choice guarantees that $\nabla \cdot u_h = 0$ on each element is satisfied in a pointwise manner which is a useful property for calculating incompressible flows.
Let us denote by V_h, Q_h and Z_h the finite dimensional spaces approximating V, Q and Z, respectively. Then, the discrete Babuška-Brezzi condition

$$\exists \beta > 0 \quad \forall q_h \in Q_h \quad \beta \|q_h\|_0 \leq \sup_{v_h \in V_h} \frac{(q_h, \nabla \cdot v_h)}{|v_h|_{1,h}}$$

is fulfilled and the discrete pressure field depends continously on the discrete velocity field. The standard discretization technique is appropriate only for low Reynolds and Rayleigh numbers. For stabilization in the singular perturbation case (i.e. for

dominating convective terms) we apply an upstream technique analyzed for the isothermal case in [ST89], [TT89]. Our method is based on a modification in the discretization of the convective terms

$$\begin{aligned}(u_h \cdot \nabla u_h, v_h) &\sim n_h(u_h, u_h, v_h)\\ (u_h \cdot \nabla T_h, z_h) &\sim N_h(u_h, T_h, z_h).\end{aligned}$$

In the following we consider only the ideas to derive N_h. The technique to construct n_h is completely analogous. For each midpoint $i = 1, ..., M$ of an inner edge Γ_i of the mesh, we define a quadrilateral box R_i by joining the endpoints of Γ_i with the barycentres of the two adjacent elements. We use the identity

$$(u \cdot \nabla T, z) = (\nabla \cdot (uT), z) - (\nabla \cdot u, Tz) \tag{4}$$

and simplify z and Tz, respectively, to be constant functions on R_i (lumping)

$$z(x) \approx z(x_i) \quad T(x)z(x) \approx T(x_i)z(x_i) \quad \forall x \in R_i.$$

Now, the resulting integrals in (4) can be transformed via integration by parts into boundary integrals over the edges Γ_{ij}, $j \in \Lambda_i$, of the boundary ∂R_i (j denotes the midpoint of the edge Γ_j corresponding to Γ_{ij}; see Figure 1).

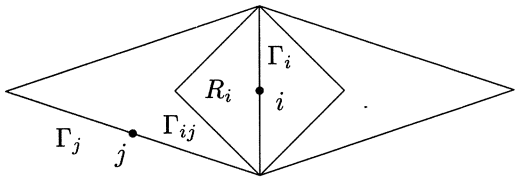


FIGURE 1. *Lumping region R_i for an inner node i*

We approximate the variable T in the first integral on Γ_{ij} by

$$T_{|\Gamma_{ij}} := \lambda_{ij} T_i + (1 - \lambda_{ij}) T_j$$

where λ_{ij} is a function of the flux $F_{ij} = \int_{\Gamma_{ij}} u \cdot n_{ij}\, ds$ through the part Γ_{ij} of the boundary of R_i, i.e. $\lambda_{ij} = \Phi(F_{ij}/\kappa)$. A simple upstream technique is the choice

$$\Phi(t) := \frac{1}{2}(1 + \operatorname{sign}(t)). \tag{5}$$

Other possibilities are described in [TT89].

In this way we obtain

$$N_h(u, T, z) := \sum_{i=1}^{M} \sum_{j \in \Lambda_i} \int_{\Gamma_{ij}} u \cdot n_{ij}\, ds\, (1 - \lambda_{ij})(T_j - T_i) z_i.$$

Note that N_h is not a trilinear form, because λ_{ij} depends on the flux through Γ_{ij}. The generalization to the three-dimensional case is straightforward.

Now the discrete problem of our nonconforming finite element method of upstream type reads:

Find $[u_h, p_h, T_h] \in V_h \times Q_h \times Z_h$ such that for all $[v_h, q_h, z_h] \in V_h \times Q_h \times Z_h$

$$\begin{aligned} \nu(\nabla u_h, \nabla v_h) + n_h(u_h, u_h, v_h) - (p_h, \nabla \cdot v_h) &= (\alpha f(T_h), v_h) \\ (q_h, \nabla \cdot u_h) &= 0 \\ \kappa(\nabla T_h, \nabla z_h) + N_h(u_h, T_h, z_h) &= (g, z_h)\,. \end{aligned} \tag{6}$$

THEOREM 2.1 *In the case of small data the solution of the discrete problem is unique and it holds the error estimate*

$$|u - u_h|_{1,h} + \|p - p_h\|_0 + |T - T_h|_{1,h} \leq c\, h.$$

For a proof of this error estimate we refer to [DST93].

3 The solution method

The corresponding nonlinear system of equations for the vector $\mathbf{z}$ of unknowns, containing the vector of velocity values $\mathbf{u} = (\mathbf{u_1}, \mathbf{u_2})$, the vector of pressure values $\mathbf{p}$ and the vector of temperature values $\mathbf{T}$, has the following structure

$$\mathbf{G(z)z} = \mathbf{F} \tag{7}$$

with

$$\mathbf{G(z)} := \begin{pmatrix} \mathbf{A}_\nu(\mathbf{u}) & \mathbf{0} & \mathbf{B_1} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_\nu(\mathbf{u}) & \mathbf{B_2} & \mathbf{D} \\ \mathbf{B_1^T} & \mathbf{B_2^T} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{A}_\kappa(\mathbf{u}) \end{pmatrix} \quad \mathbf{z} := \begin{pmatrix} \mathbf{u_1} \\ \mathbf{u_2} \\ \mathbf{p} \\ \mathbf{T} \end{pmatrix} \quad \mathbf{F} := \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{g} \end{pmatrix}. \tag{8}$$

A nonlinear full multigrid method **NFMG** is used for solving the equation (7). An advantage of our simple upwind discretization (5) is that for an arbitrary $\mathbf{u}$ the matrix blocks $\mathbf{A}_{\nu,\kappa}(\mathbf{u})$ become M-matrices if the angles of the triangles of the mesh are not greater than $\frac{\pi}{2}$. On the other hand $\mathbf{A}_{\nu,\kappa}(\mathbf{u})$ is not differentiable. Therefore, we only apply the simple iteration within the smoother of our nonlinear multigrid method. In our notation, the subscript k indicates the actual grid level. The accepted solution is called $\mathbf{z}_k^*$.

$\underline{\mathbf{z}_k^* = \mathbf{NFMG(maxcyc,res,qmin,qmax,maxsmooth,}\omega_s\mathbf{,}\omega_c\mathbf{)}}$

For i = 1,2... (i = number of the cycle), perform a multigrid V-cycle defined by steps A1 - A4.
A new iterate $\mathbf{z}_k^i$ is obtained by :

$$\mathbf{z}_k^i = \mathbf{VCYCLE(qmin,qmax,maxsmooth,}\omega_s\mathbf{,}\omega_c\mathbf{,}\mathbf{z}_k^{i-1}\mathbf{)}$$

We set $\mathbf{w}_k^0 := \mathbf{z}_k^{i-1}$. The initial guess $\mathbf{z}_k^0$ is computed by prolongation of the accepted solution $\mathbf{z}_{k-1}^*$ on the previous grid level k-1.

A1) Pre-Smoothing

Perform for j = 1,2,... several smoothing steps

$$\mathbf{w}_k^j = \mathbf{Smoother(maxsmooth, qmin, qmax,}\ \omega_\mathbf{S}\mathbf{,}\ \mathbf{w}_k^{j-1}\mathbf{)}.$$

Stop the iteration in the following cases:

- j = **maxsmooth**
- $\frac{\|\mathbf{G}_k(\mathbf{w}_k^j)\mathbf{w}_k^j - \mathbf{F}_k\|}{\|\mathbf{G}_k(\mathbf{w}_k^0)\mathbf{w}_k^0 - \mathbf{F}_k\|} \leq$ **qmin** (prescribed damping is achieved)
- $\frac{\|\mathbf{G}_k(\mathbf{w}_k^j)\mathbf{w}_k^j - \mathbf{F}_k\|}{\|\mathbf{G}_k(\mathbf{w}_k^{j-1})\mathbf{w}_k^{j-1} - \mathbf{F}_k\|} >$ **qmax** (avoiding oszillations in the smoothing process)

Let the result of this procedure be denoted by $\bar{\mathbf{w}}_k := \mathbf{w}_k^j$.

A2) Coarse grid iteration

On the coarser grid levels (l = k-1,...,1) we solve approximately a problem with the structure:

$$\mathbf{G}_l(\mathbf{w}_l^*)\mathbf{w}_l^* = \mathbf{F}_l := \mathbf{G}_l(\mathbf{z}_l^*)\mathbf{z}_l^* + \mathbf{d}_l \tag{9}$$

where $\mathbf{z}_l^*$ is the accepted solution of (7) on grid level l and $\mathbf{d}_l$ is defined as restriction of the fine grid defect

$$\mathbf{d}_{l+1} := \mathbf{F}_{l+1} - \mathbf{G}_{l+1}(\bar{\mathbf{w}}_{l+1})\bar{\mathbf{w}}_{l+1}. \tag{10}$$

We compute $\tilde{\mathbf{w}}_l \approx \mathbf{w}_l^*$ by applying

$$\tilde{\mathbf{w}}_l = \mathbf{VCYCLE(qmin, qmax, maxsmooth,}\ \omega_\mathbf{S}\mathbf{,}\ \omega_\mathbf{C}\mathbf{,}\mathbf{z}_l^*\mathbf{)}$$

with starting vector $\mathbf{z}_l^*$.

A3) Correction

Let $\mathbf{v}_k$ be the prolongation of the correction

$$\mathbf{v}_{k-1} = \tilde{\mathbf{w}}_{k-1} - \mathbf{z}^*_{k-1}, \tag{11}$$

obtained by step A2 on grid level l = k-1. To get the new approximation $\hat{\mathbf{w}}_k$ we use the following damped correction

$$\hat{\mathbf{w}}_k = \bar{\mathbf{w}}_k + \omega_c \mathbf{v}_k. \tag{12}$$

A4) Post-Smoothing

Perform the smoothing procedure of step A1 applied to the starting vector $\hat{\mathbf{w}}_k$. The result is our new approximation $\mathbf{z}^i_k$.

The multigrid procedure **NFMG** is stopped in the following cases
- $\|\mathbf{G}_k(\mathbf{z})^i_k)\mathbf{z}^i_k - \mathbf{F}_k\| \leq$ **res** (final residual is achieved)
- i = **maxcyc**.

3.1 SMOOTHING PROCEDURE

One smoothing step

$$\mathbf{w}^j_k = \mathbf{Smoother}(\mathbf{maxsmooth}, \mathbf{qmin}, \mathbf{qmax}, \omega_\mathbf{S}, \mathbf{w}^{j-1}_k)$$

consists of three steps.

A) Applying the forward Gauss-Seidel iteration to the energy equations in (8) we get a new vector **T** for the temperature. In the y-momentum equation of (8) we use this vector **T** and put the term associated with **T** to the right hand side.

B) In this step we update the values for velocity and pressure. We take a block-wise Gauss-Seidel smoother described in [RS88] and [Van86].
That means on every element in our mesh the local system of corresponding unknowns is generated and the off-diagonal terms are taken to the right hand side. The local system contains 6 unknowns for velocity in the midpoints of a triangle and one for pressure.

C) Denoting the result from step A and step B by $\bar{\mathbf{w}}^j_k$ we perform the following relaxation

$$\mathbf{w}^j_k = \mathbf{w}^{j-1}_k + \omega_s(\bar{\mathbf{w}}^j_k - \mathbf{w}^{j-1}_k). \tag{13}$$

3.2 Prolongation and restriction

Restriction and prolongation are defined as projection in $L^2(\Omega)$ and $(L^2(\Omega))^*$, respectively. For a detailed description see [RS88]. For example, the prolongation $\mathbf{I}_{2h}^h r^{2h}$ of a given coarse grid function $\mathbf{r}^{2h}$ is determined by

$$(\mathbf{I}_{2h}^h \mathbf{r}^{2h} - \mathbf{r}^{2h}, \phi^h) = 0 \quad \forall \phi^h \in V_h \tag{14}$$

and the restriction $\mathbf{I}_h^{2h} r^h$ of a given fine grid function $\mathbf{r}^h$ by

$$(\mathbf{I}_h^{2h} \mathbf{r}^h - \mathbf{r}^h, \phi^{2h}) = 0 \quad \forall \phi^{2h} \in V_{2h}. \tag{15}$$

A nice property of this choice is that, the canonical basis functions are L^2-orthogonal which leads to explicit formulas for $\mathbf{I}_{2h}^h r^{2h}$ and $\mathbf{I}_h^{2h} r^h$.

3.3 Coarsest grid solver

For solving problem (7) on the coarsest grid level we use the smoothing procedure too. The smoothing iteration on the coarsest level is stopped if a prescribed residual norm is obtained or the number of allowed iteration steps is exceeded. Of course, this number of iteration steps is much higher than the number used in the multigrid algorithm.

4 Numerical results

Our algorithm is applied to the non-isothermal problem of laminar free convection in a closed cavity. It is one of the most popular test-problems to compare numerical algorithms for solving of the Navier-Stokes equations of incompressible flows. The geometry of the cavity and the boundary conditions are shown in Figure 2. In the

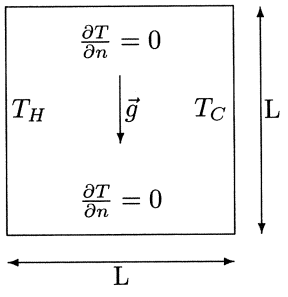


FIGURE 2. Geometry of the cavity and boundary conditions

dimensionless coordinates the computational domain Ω extends from 0 to 1 in the

Ra	h	$\|u_h - u_{2h}\|_0$	$\|T_h - T_{2h}\|_0$	$\|p_h - p_{2h}\|_0$	$\|u_h - u_{2h}\|_{1,h}$	$\|T_h - T_{2h}\|_{1,h}$
10^5	1/8	0.64 -1	0.51 -1	0.63 -2	0.13 +1	0.95 +0
	1/16	0.30 -1	0.21 -1	0.27 -2	0.11 +1	0.70 +0
	1/32	0.12 -1	0.95 -2	0.14 -2	0.72 +0	0.40 +0
	1/64	0.35 -2	0.26 -2	0.70 -3	0.41 +0	0.22 +0
	1/128	0.11 -2	0.93 -3	0.35 -3	0.22 +0	0.12 +0
10^6	1/8	0.76 -1	0.72 -1	0.80 -2	0.19 +1	0.16 +1
	1/16	0.47 -1	0.43 -1	0.42 -2	0.20 +1	0.15 +1
	1/32	0.25 -1	0.19 -1	0.15 -2	0.16 +1	0.11 +1
	1/64	0.11 -1	0.97 -2	0.80 -3	0.11 +1	0.72 +0
	1/128	0.31 -2	0.28 -2	0.39 -3	0.62 +0	0.41 +0
10^7	1/8	0.81 -1	0.73 -1	0.67 -2	0.22 +1	0.20 +1
	1/16	0.60 -1	0.68 -1	0.62 -2	0.29 +1	0.23 +1
	1/32	0.37 -1	0.31 -1	0.22 -2	0.30 +1	0.22 +1
	1/64	0.21 -1	0.17 -1	0.83 -3	0.25 +1	0.18 +1
	1/128	0.11 -1	0.98 -2	0.44 -3	0.17 +1	0.12 +1

TABLE 1. Numerical h-convergence for different Rayleigh numbers

x and y directions. The boundary conditions are:

- $\mathbf{u} = 0$ on $\partial\Omega$
- $T_H = 0.5$ and $T_C =$ - 0.5.

4.1 Streamlines, isotherms and isobars

We present figures of streamlines and isotherms for the Rayleigh numbers $Ra = 10^5$, $Ra = 10^6$ and $Ra = 10^7$. Isobars are shown for $Ra = 10^6$ and $Ra = 10^7$. The isotherms are plotted in eleven contourlines corresponding to the temperature increment. Looking to the plotted streamlines we remark that we used nonequidistant isovalues.

4.2 Convergence history and accuracy

Table 1 gives the numerical convergence of velocity, temperature and pressure in the L_2-Norm $\|\cdot\|_0$ and the discrete H_1-seminorm $|\cdot|_{1,h}$, respectively, for Rayleigh numbers $Ra = 10^5$, 10^6 and 10^7. We see that for high Rayleigh numbers we get only first order convergence in L_2 for velocity and temperature. In [Tob89] it was shown for an one-dimensional example that using the simple upwind technique we can not expect in general a better convergence in L_2 than first order. For pressure we see first order convergence for all cases. In the discrete H_1-seminorm we get first order convergence only for $Ra = 10^5$ and 10^6.

In order to compare our method UPW-FEM with other methods we consider the maximal velocities at the midlines of the cavity and their localization. For

Ra	code	h	$u_{\max}$	y	$v_{\max}$	x
10^5	[LA93]	1/16	35.7300	0.83600	62.0720	0.06250
		1/32	36.4170	0.84400	68.2900	0.06250
		1/64	35.9210	0.85900	68.6730	0.06250
	[HPS90]	1/20	34.5813	0.87500	66.4046	0.07500
		1/40	34.7396	0.86250	68.8438	0.06250
		1/80	34.7499	0.85625	68.5600	0.06875
	UPW-FEM	1/16	37.2051	0.87500	66.1852	0.06250
		1/32	36.4822	0.84375	67.1288	0.06250
		1/64	36.4123	0.85938	68.0585	0.06250
10^6	[ZW89]	1/32	68.8600	0.85900	235.680	0.04690
		1/64	62.6900	0.84800	225.290	0.03910
	[HPS90]	1/40	65.3710	0.86250	223.412	0.03750
		1/80	64.9944	0.85625	218.312	0.03125
		1/160	64.8659	0.85312	219.861	0.04062
	UPW-FEM	1/32	70.2753	0.84375	201.974	0.03125
		1/64	64.6953	0.84375	214.116	0.04688
		1/128	64.8373	0.84375	220.592	0.03906
10^7	[LeQ91]	1/32	148.444	0.87900	699.496	0.02100
		1/64	148.582	0.87900	699.233	0.02100
		1/80	148.583	0.87900	699.237	0.02100
	UPW-FEM	1/32	185.081	0.81250	650.366	0.03125
		1/64	168.565	0.84375	658.032	0.03125
		1/128	146.479	0.85156	676.699	0.02344

TABLE 2. Comparison of maximal velocities at the midlines of the cavity for several codes

comparision we use the reference results of [HPS90], [ZW89], [LA93] and [LeQ91]. In [HPS90] the problem is solved by a multigrid method using the finite volume discretization based on central differences. In [ZW89] the same method is combined with a hybrid difference scheme (defect correction) using artificial viscosity and artificial heat conductivity. A conforming P1-approximation for velocity, temperature and pressure and Galerkin/least-squares stabilization is used in [LA93]. The time-dependent equations are integrated in [LeQ91]. The algorithm is of pseudo-spectral type and combines spatial expansions in series of Chebyshev polynomials with a finite-difference time-stepping scheme. Table 2 illustrates the results of the several codes.

Figures 3 and 4 present the convergence history of the nonlinear multigrid method. For a finer mesh, the efficiency of the multigrid algorithm is enhanced. The number of smoothing steps was set to **maxsmooth** = 10 iterations. For the other parameters of our algorithm we used the values **res** = 5.e-8, $\omega_{\mathbf{C}} = \omega_{\mathbf{S}} = 0.5$, **maxcyc** = 100, **qmin** = 0.7 and **qmax** = 1.0 .

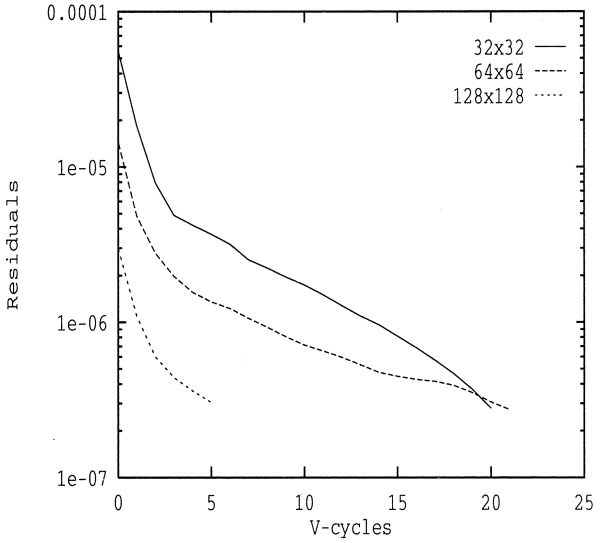


FIGURE 3. Convergence history, Ra = 10^6

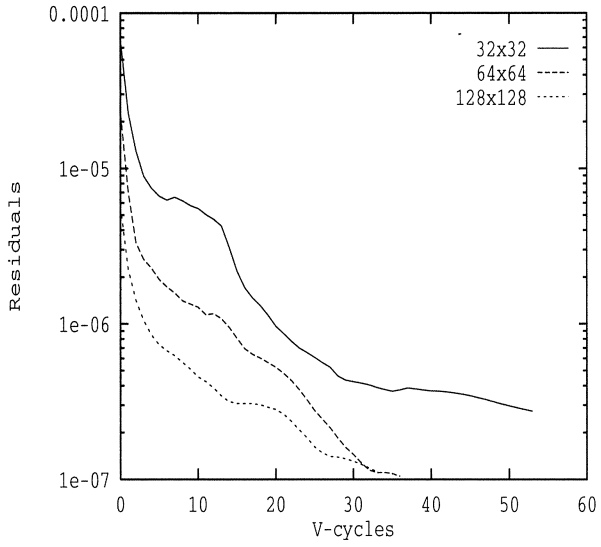


FIGURE 4. Convergence history, Ra=10^7

References

[BMPT91] C. Bernardi, B. Métivet, and B. Pernaud-Thomas. Couplage des equations de Navier-Stokes et de la chaleur: le modéle et son approximation par elements finis. *Report 91014, Univ. Pierre et Marie Curie,CSR*, 1991.

[DST93] O. Dorok, F. Schieweck, and L. Tobiska. Error estimates for an upwind discretization of the Boussinesq approximation of the Navier-Stokes equations (in preparation). 1993.

[GR86] V. Girault and P.-A. Raviart. *Finite Element Methods for Navier-Stokes equations.* Springer-Verlag, Berlin-Heidelberg-New York, 1986.

[HPS90] M. Hortmann, M. Peric, and G. Scheuerer. Finite volume multigrid prediction of laminar natural convection : Bench-mark solutions. *Intern. J. Numer. Meths. Fluids*, 11:189–207, 1990.

[LA93] G. Lube and A. Auge. Regularized mixed finite element approximations of non-isothermal incompressible flow problems. *to appear in ZAMM*, 73, 1993.

[LeQ91] P. LeQuéré. Accurate solutions to the square thermally driven cavity at high Rayleighnumbers. *Computer Fluids*, 20(1):29–41, 1991.

[RS88] U. Risch and F. Schieweck. A multigrid method for solving the stationary Navier- Stokes equations by FEM. *Preprint Math 5/88 TU Magdeburg*, 1988.

[RS90] U. Risch and F. Schieweck. Experiences with the multigrid method applied to high Reynolds number steady incompressible flows. In G. Telschow, editor, *Fourth Multigrid Seminar*, 1990.

[ST89] F. Schieweck and L. Tobiska. A nonconforming finite element method of upstream type applied to stationary Navier-Stokes equations. *M^2AN*, 23:627–647, 1989.

[Tob89] L. Tobiska. Full and weighted upwind finite element methods. In J.W. Schmidt and H. Späth, editors, *Splines in Numerical Analysis*, volume 52 of *Mathematical Research*, pages 181–188. Akademie-Verlag Berlin, 1989.

[TT89] A. Thiele and L. Tobiska. A weighted upwind finite element method for solving the stationary Navier-Stokes equations. *WZ TU Magdeburg*, 33:13–20, 1989.

[Van86] S. Vanka. Block-implicit multigrid calculation of two-dimensional recirculating flows. *Comp. Meth. Appl. Mech. Eng.*, 59(1):29–48, 1986.

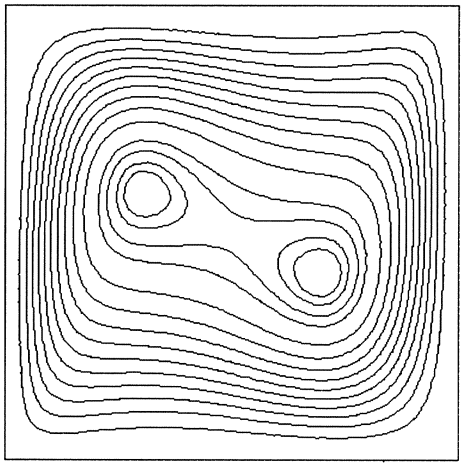


FIGURE 5. Streamlines,$h = 1/32$,Ra=10^5

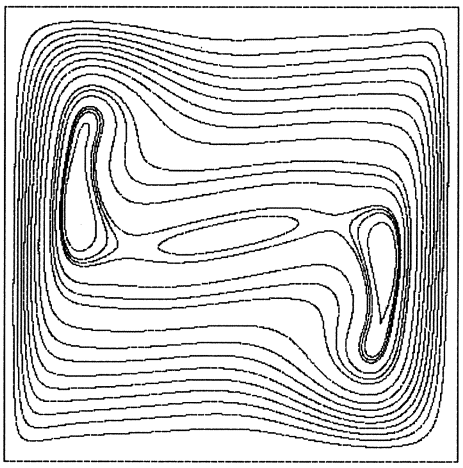


FIGURE 6. Streamlines, $h = 1/128$, Ra=10^6

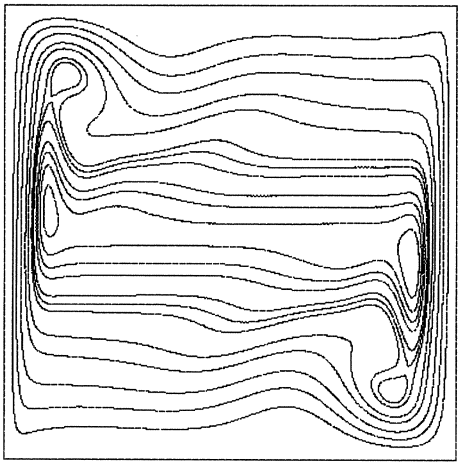


FIGURE 7. Streamlines, $h = 1/128$, Ra=10^7

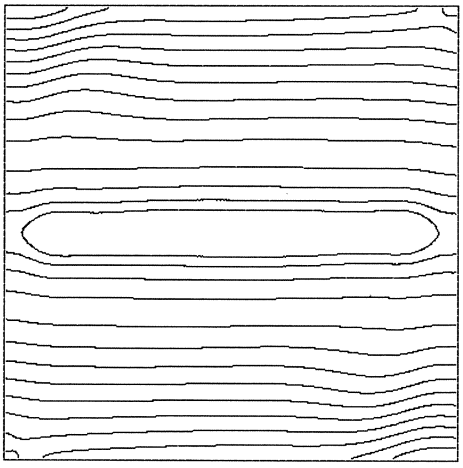


FIGURE 8. Isobars, $h = 1/128$, Ra=10^6

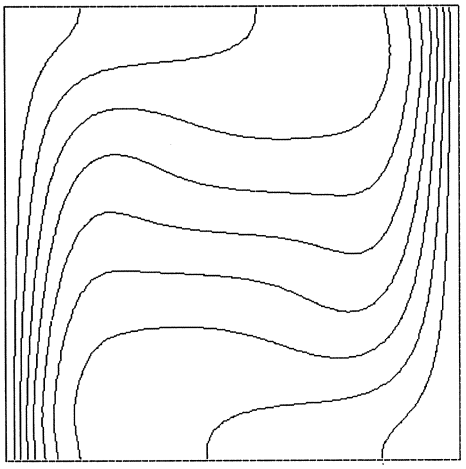


FIGURE 9. Isotherms, $h = 1/32$, Ra=10^5

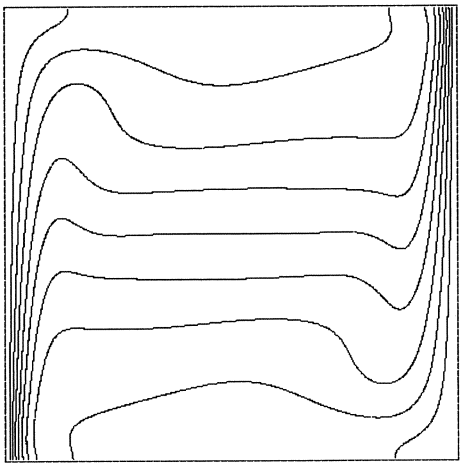


FIGURE 10. Isotherms, $h = 1/128$, Ra=10^6

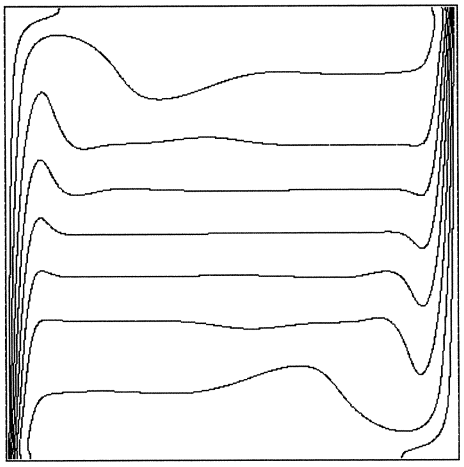


FIGURE 11. Isotherms, $h = 1/128$, Ra=10^7

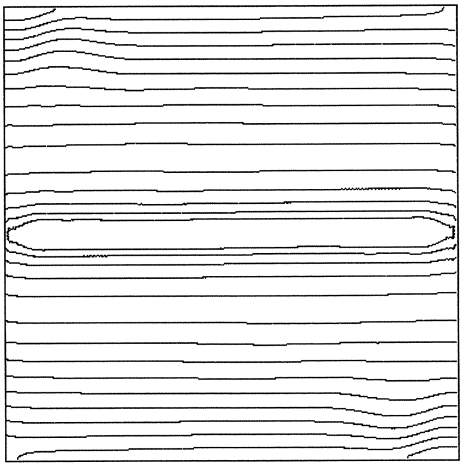


FIGURE 12. Isobars, $h = 1/128$, Ra=10^7

[ZW89] S. Zeng and P. Wesseling. A multigrid method combined with defect correction for free convection problems at high Rayleigh numbers. *Report 89-27, TU DELFT*, 1989.

5

On matrix data structures and the stability of multigrid algorithms

Jürgen Fuhrmann[1] and Klaus Gärtner[2]

ABSTRACT [3] A multigrid preconditioner is presented which can be used to to solve symmetric and certain nonsymmetric problems with strongly varying coefficients in two and three space dimensions. It is semi-algebraic, that means, the coarse grid matrices are generated algebraically from the fine grid matrix using the knowledge of the coarse mesh structure. It preserves symmetry with respect to weighted scalar products. Based on the columnwise diagonal dominance of the discrete operators generated by exponentially fitted upwind schemes, a diagonal sparse matrix storage scheme which stores instead of the matrix main diagonal the difference between the main diagonal and the negative sum of the non-diagonal column entries ensures the numerical stability of the multigrid components.

1 Introduction

For numerical methods to solve linear problems occurring during the process of the solution of real world problems, like semiconductor device equations or porous media flow calculations, a high robustness against coefficient jumps and operator asymmetries is demanded. This raises problems for the development of multigrid methods. While ILU methods provide good smoothers, the design of the other basic multigrid components – the intergrid transfer operators and the coarse grid operators – in the case of inverse averaging and exponential fitting discretizations is not an easy task.

In this paper, an algebraic scheme as proposed in [Fuh92] is used to set up the multigrid components. It is shortly described in section 3. It has been successfully implemented on two- and threedimensional rectangular grids. The main features of the preconditioner are

---

[1]Institut für Angewandte Analysis und Stochastik, D-10117 Berlin, Mohrenstraße 39, fuhrmann@iaas-berlin.d400.de

[2]Interdisciplinary Project Center for Supercomputing, ETH Zürich, ETH-Zentrum, CH-8092 Zürich, gaertner@ips.id.ethz.ch

[3]Partially supported by the SFB 123 of the Deutsche Forschungsgemeinschaft.

- dependence only on the algebraic structure of the finest grid operator
- operator dependent intergrid transfer operators
- five (2D) and seven (3D) point stencils on all grids using recursively created approximate Galerkin coarse grid operators
- preservation of symmetry with respect to certain weighted scalar products
- fixed Tchebyshev polynomials as an approximate inverse on the coarsest grid to ensure the preconditioner to be constant during conjugated gradient iterations.

The implementation of the method on tensor product meshes is based on a sparse matrix diagonal storage scheme which holds the operators on different grids. The canonical scheme with the storage of the assembled main diagonal of the stiffness matrix results in numerically unstable representations for the intergrid transfer and coarse grid operators and does not guarantee the propagation of the fine grid matrix properties onto the coarse grid in the floating point arithmetic. In section 4 a sparse matrix diagonal storage scheme with unassembled main diagonal is proposed which results in numerically stable formulae for the multigrid components. A test example in section 5 in fact shows very different convergence behaviour depending on the data structure chosen. The method has been tested and optimized in different fields of applications.

In section 6, an example from semiconductor device simulation is shortly discussed. For results concerning a nonlinear diffusion equation we refer to [Fuh93b].

2 The problem

We assume to be given the following boundary value problem on a rectangular resp. quadrilateral domain $\Omega \subset \mathbf{R}^d, d = 1, 2, 3$:

$$-\nabla \cdot (k(x)\nabla u(x)) + c(x)u(x) = f(x) \tag{1}$$

with mixed boundary conditions where k, c and f are assumed to be given functions.

A second problem we will refer to is the following:

$$-\nabla \cdot (\nabla v(x) + v(x)\nabla \psi(x)) + d(x)u(x) = f(x) \tag{2}$$

Again, we assume mixed boundary conditions and ψ, d, f to be given functions. Such an equation occurs as the linearized carrier transport equation in semiconductor device simulation (see (13) in section 6). One has to remark that the variable substitution

$$v \mapsto u = e^{\psi} v \tag{3}$$

generates from (2) a selfadjoint problem similar to (1) with $k = e^{-\psi}$.

We are interested in the case, when the coefficient functions show large variations which cannot be resolved by the discretization mesh. This is the case if one

considers rectangular meshes. Because of the fact that adaptive mesh refinement for problems of type (2) is not yet a standard technique, one even then has to deal with this problem.

Problem (1) can be discretized by homogeneous differece schemes using logarithmically harmonic averaging of the coefficient k and mass lumping for c resulting in a linear problem

$$A_s u_h = f_h \tag{4}$$

with a five or seven diagonal symmetric, positive definite M-matrix A_s acting in a finite dimensional Euclidean space $\mathcal{M}$ [Sam83].

For problem (2) we use a box method together with an Il'in scheme resulting in a five or seven diagonal operator A_n

$$A_n v_h = f_h \tag{5}$$

which in our case is equivalent to the Scharfetter-Gummel exponential fitting scheme described e.g. in [Sel84].

There are various approaches to put these kinds of discretizations into a finite element context, possibly using mixed methods and "nonstandard" numerical integration [BMP87, Son89, Fuh90, HM91].

Let $E = E_h = e^{\psi_h}$ be the diagonal matrix containing the node values of e^{ψ} and let

$$((\cdot,\cdot))_E = (E\cdot,\cdot)$$

be the Euclidean scalar scalar product with weight E.

Then we have the following facts:

- similar to (3), if $k = e^{-\psi_h}$,
$$A_n = A_s E. \tag{6}$$
- A_s is a Stieltjes matrix.
- A_n is selfadjoint in the scalar product $((\cdot,\cdot))_E$.
- A_n is a columnwise diagonally dominant M-matrix.

3 The multigrid method

Let $\mathcal{M}_0, \ldots \mathcal{M}_j = \mathcal{M}$ be a sequence of Euclidean vector spaces with growing dimension. In order to define a standard multigrid algorithm to solve $A_j u = f$ for $A_j = A_s$ or $A_j = A_s$ there need to be defined the following components (in the terminology of [BPX91]):

- scalar products $((\cdot,\cdot))_k : \mathcal{M}_k \times \mathcal{M}_k \to R$
- symmetric, positive definite with respect to $((\cdot,\cdot))_k$ operators $A_k : \mathcal{M}_k \to \mathcal{M}_k$,
- interpolations $I_k : \mathcal{M}_{k-1} \to \mathcal{M}_k$
- restrictions $P_k^0 : \mathcal{M}_k \to \mathcal{M}_{k-1}$.
- smoothers $R_k : \mathcal{M}_k \to \mathcal{M}_k$

While the smoothers R_k can provided by one or more steps of a "classical" iteration method (Jacobi, Gauß-Seidel, ILU), the design of other components in cases of strongly varying coefficients or missing standard finite element background is unclear. Here, we try a "semi-algebraic" method as described at previous stages in [FG91]. Rather similar ideas of constructing multigrid or multilevel preconditioners have been used in [AV90, DE89, Kuz90, Pop91].

In what follows, we omit the level-k-indices. Fot given k, a fine grid corresponds to $\mathcal{M}_k$, a coarse grid then corresponds to $\mathcal{M}_{k-1}$.

On a three-dimensional grid with quadrilateral cells generated by standard refinement from a coarser one, we have the splitting of the grid vertex set $V(A) = V_C \cup V_F \cup V_E \cup V_N$ into sets of coarse grid cell midpoints, coarse grid cell face midpoints, coarse grid cell edge midpoints and coarse grid node points, respectively. We get the matrix partitioning

$$A = \begin{pmatrix} \begin{pmatrix} A_C & B_{CF} & 0 \\ B_{FC} & A_F & B_{FE} \\ 0 & B_{EF} & A_E \end{pmatrix} & \begin{pmatrix} 0 \\ 0 \\ B_{EN} \end{pmatrix} \\ \begin{pmatrix} 0 & 0 & B_{NE} \end{pmatrix} & A_N \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix},$$

where the off diagonal blocks are nonpositive and the diagonal blocks $A_C = A_{CF} + M_C$, $A_F = A_{FC} + A_{FE} + M_F$, $A_E = A_{EF} + A_{EN} + M_E$ and $A_N = A_{NE} + M_N$, are positive diagonal matrices which consist of sums of off diagonal column entries and a nonnegative "mass" term. The assumptions made on A imply that at least one entry of the "mass" M_* is positive.

Let $\tilde{A}_F = A_{FE} + M_F$, $\tilde{A}_E = A_{EN} + M_E$, and choose

$$F = \begin{pmatrix} A_C & B_{CF} & 0 \\ 0 & \tilde{A}_F & B_{FE} \\ 0 & 0 & \tilde{A}_E \end{pmatrix}, \qquad G = A_{12}, \qquad U = \begin{pmatrix} F & G \\ 0 & I \end{pmatrix}.$$

U can be seen as a transformation matrix to an approximate harmonic basis [HLM91]. Then for $\cdot^T$ being the transposition with respect to the $((\cdot,\cdot))$-scalar product we have

$$A = U^T \begin{pmatrix} F^{-T} A_{11} F^{-1} & \Delta \\ \Delta^T & S \end{pmatrix} U, \tag{7}$$

with

$$\Delta = F^{-T}(A_{12} - A_{11}F^{-1}G)$$

and

$$S = \hat{S} + \Delta^T (F^{-T} A_{11} F^{-1})^{-1} \Delta$$

where

$$\hat{S} = A_{22} - A_{21}(A_{11})^{-1} A_{12}$$

is the Schur complement. To create a block diagonal preconditioner for A in the new basis, one takes the decomposition (7) and omits the off diagonal blocks

Δ. Omitting the error correction in the fine grid part, too, yields a coarse grid correction by projecting the error vector onto the fine grid space in the new basis. It has the form

$$B = U^{-1} \begin{pmatrix} 0 & 0 \\ 0 & S^{-1} \end{pmatrix} U^{-T} = I_k S^{-1} P_k^0$$

with

$$I_k = \begin{pmatrix} -F^{-1}G \\ I \end{pmatrix} = \begin{pmatrix} \begin{pmatrix} -A_C^{-1} B_{CF} \tilde{A}_F^{-1} B_{FE} \tilde{A}_E^{-1} B_{EN} \\ \tilde{A}_F^{-1} B_{FE} \tilde{A}_E^{-1} B_{EN} \end{pmatrix} \\ \begin{pmatrix} -\tilde{A}_E^{-1} B_{EN} \\ I \end{pmatrix} \end{pmatrix} \tag{8}$$

and P_k^0 being its $((\cdot,\cdot))$-adjoint. S is the Galerkin coarse grid operator corresponding to the given choice of the intergrid transfer operators:

$$\begin{aligned} S &= P_k^0 A I_k \\ &= (A_N - B_{NE}\tilde{A}_E^{-1}B_{EN}) + B_{NE}\tilde{A}_E^{-1}(A_{EF} - B_{EF}\tilde{A}_F^{-1}B_{FE})\tilde{A}_E^{-1}B_{EN} \\ &\quad + B_{NE}\tilde{A}_E^{-1}B_{EF}\tilde{A}_F^{-1}(A_{FC} - B_{FC}A_C^{-1}B_{CF})\tilde{A}_F^{-1}B_{FE}\tilde{A}_E^{-1}B_{EN}. \end{aligned}$$

Some geometrical considerations and numerical experiments [Fuh92, Fuh93a] suggest that in the sense of spectral equivalences, it should hold that

$$S \approx 4(A_N - B_{NE}\tilde{A}_E^{-1}B_{EN}) =: A_{k-1}, \tag{9}$$

when the coefficients vary not too strongly. This suggests replacing S by A_{k-1} in B. At the other hand, A_{k-1} is the Schur complement of the positive definite matrix

$$4 \begin{pmatrix} A_{EN} + M_E & B_{EN} \\ B_{NE} & A_N \end{pmatrix} \tag{10}$$

and inherits the $((\cdot,\cdot))$-symmetry, the M-property, and the seven-diagonal structure of A, so the process described above can be continued recursively.

It can be shown [Fuh92, Fuh93a] that for $((\cdot,\cdot))$-symmetric, positive definite operators, the convergence of a multigrid method with components defined this way depends on a number of reasonable factors:

- the spectral equivalences of A_{11} and F, and of A_{11} and its diagonal;
- the cosines of the angles between coarse grid and fine grid spaces in the A-energy scalar product;
- the spectral equivalence of S and A_{k-1};
- a smoothing property for R_k, which for Jacobi and Gauß-Seidel smoothers is valid for any symmetric M-matrix [RS87]

The whole multigrid operator described above is selfadjoint in the $((\cdot,\cdot))_j$-scalar product provided the smoothers are, and one can use it as a preconditioner for conjugated gradients in this scalar product.

So, for the Euclidean scalar product $(\cdot,\cdot)$, we have a preconditioner for A_s.

If we use $((\cdot,\cdot))_{E_j,j} = ((\cdot,\cdot))_E$ and define via E_k for $k < j$ resulting from straight injection the scalar products on the coarse grids, we get a preconditioner for A_n which can be used for conjugated gradients in a weighted scalar product.

4 The data structure

To ensure the preservation of the M-property in a floating point representation, one has to choose a matrix data structure where, on all the levels, instead of the main diagonal entries of A, the difference between these entries and the sum of the remaining entries of the same column is stored.

The discrete operator A can be described on a three-dimensional quadrilateral mesh with $N_x \times N_y \times N_z$ nodes by the generic stencil

$$\begin{aligned}(Au)_{ijk} &= -d^{l,x}_{i-1jk}u_{i-1jk} + (d^{u,x}_{i-1jk} + d^{l,x}_{ijk})u_{ijk} - d^{u,x}_{ijk}u_{i+1jk}) \\ &- d^{l,y}_{ij-1k}u_{ij-1k} + (d^{u,y}_{ij-1k} + d^{l,y}_{ijk})u_{ijk} - d^{u,y}_{ijk}u_{ij+1k}) \\ &- d^{l,z}_{ijk-1}u_{ijk-1} + (d^{u,z}_{ijk-1} + d^{l,z}_{ijk})u_{ijk} - d^{u,z}_{ijk}u_{ijk+1}) \\ &+ d^{m}_{ijk}u_{ijk}.\end{aligned} \tag{11}$$

It is symmetric, if $d^{l,*}_* = d^{u,*}_*$.

If one uses a penalty method for the approximation of the Dirichlet boundary conditions, d^m holds the boundary data, too, and it is possible to develop an easily vectorizable matrix data structure [GTG+89].

The storage of the operator data can be done in two ways:

- store $d^{l,*}$,$d^{u,*}$ and the assembled main diagonal

$$d^{\text{main}}_{ijk} = d^{m}_{ijk} + d^{u,x}_{i-1jk} + d^{l,x}_{ijk} + d^{u,y}_{ij-1k} + d^{l,y}_{ijk} + d^{u,z}_{ijk-1} + d^{l,z}_{ijk}$$

  as one-dimensional vectors filled by zeroes whenever it is necessary
- store $d^{l,*}$,$d^{u,*}$ as above, but instead of assembling the main diagonal, store only d^m

According to (9), this storage scheme can be used on all multigrid levels. The link between the numerical stability and these storage schemes becomes clear if one regards the implementation of the multigrid components - restriction P^0_k, prolongation I_k and coarse grid matrix condensation (calculation of A_{k-1} from A_k using (9)). To be more precise, we look at the operators $\tilde{A}^{-1}_F B_{FE}$ and $\tilde{A}^{-1}_E B_{EN}$ from section 3 involved into the calculation of all these components. The point is the calculation of the diagonal operators $\tilde{A}_*$. The B_* shouldn't make trouble because they are always nonnegative. At the generic x-parallel coarse grid cell edges we have

$$\begin{aligned}(\tilde{A}_E)_{ijk} &= d^{\text{main}}_{ijk} - d^{u,y}_{ij-1k} - d^{l,y}_{ijk} - d^{u,z}_{ijk-1} - d^{l,z}_{ijk} \\ &= d^{m}_{ijk} + d^{u,x}_{i-1jk} + d^{l,x}_{ijk},\end{aligned}$$

and at the generic x-orthognonal coarse grid cell faces we have to calculate

$$\begin{aligned}(\tilde{A}_F)_{ijk} &= d^{\text{main}}_{ijk} - d^{u,x}_{i-1jk} - d^{l,x}_{ijk} \\ &= d^{m}_{ijk} + d^{u,y}_{ij-1k} + d^{l,y}_{ijk} + d^{u,z}_{ijk-1} + d^{l,z}_{ijk}.\end{aligned}$$

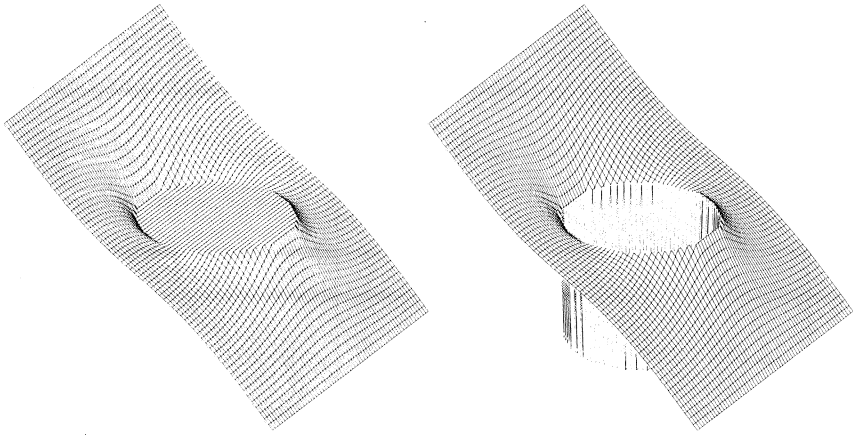


FIGURE 1. Solutions of problems (1) and (2) for $\kappa = 10$.

Similar formulae are valid for the y- and z-directions. It is obvious that the formulae involving d^m are the better ones, and in fact, one is able to guarantee in equation (9) the propagation of the M-property onto the coarse grid in the floating point arithmetic, too.

Of course, for coefficients varying not very strongly there is no big difference, but if one considers the semiconductor device example below, this little difference becomes the question of "to be or not to be" for multigrid !

5 An experiment

To verify the considerations above, we make a simple 2D-experiment. Let $\Omega = [0,1] \times [0,1]$ be discretized using a 64×64 mesh with 5 coarse levels. Let $\psi : \Omega \to \mathbf{R}$ be a given by

$$\psi(x_1, x_2) = \kappa \begin{cases} -1 & , \quad (x_1 - \frac{1}{2})^2 + (x_2 - \frac{1}{2})^2 < (\frac{1}{4})^2 \\ 1 & , \quad \text{else.} \end{cases} \tag{12}$$

We look for u,v being the solutions of (1) with $k = e^{-\psi}$ and (2) with the Dirichlet boundary conditions

$$\begin{aligned} u &= 1 \quad \text{on} \quad 0 \times [0,1] \\ u &= 0 \quad \text{on} \quad 1 \times [0,1]. \end{aligned}$$

On all other parts of the boundary, homogeneous Neumann boundary conditions are assumed. The solutions are shown in figure 1.

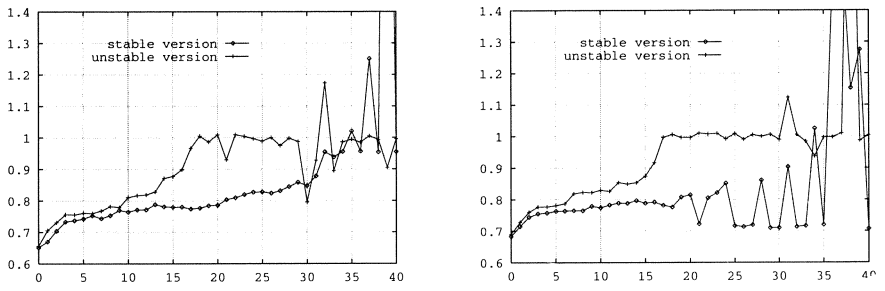


FIGURE 2. Dependence of the average per-work-unit contraction rate on κ for problem (1) (left) and problem (2) (right).

Now, we trace the behaviour of the multigrid preconditioned CG method while increasing κ succesively from 0 to 40 by steps of 1 for both data structures (fig. 2) We measure the norm of the residual vector emerging during the CG calculations. It is of course "virtual" , but can tell us something about the stability of the algorithm. We stop the iteration when the overall residual contraction becomes 10^{-10} and measure the contraction per work unit. As one work unit we take the amount of CPU time necessary for one Jacobi step. One multigrid step then takes 10 - 12 work units depending on the data structure, so that a per-work-unit contraction of 0.7 corresponds to a per-step contraction of 0.028 and a per-work-unit contraction of 0.9 still yields 0.35 per multigrid step. We see, that in fact, for growing values of κ, the stability considerations made above become more and more important.

6 A problem from semiconductor device simulation

The multigrid preconditioned CG method in the weighted scalar product has been build into the 3D semiconductor device simulation program MEDEA [GTG+89] as a solver for the carrier transport equations in the stationary van Roosbroek system [Sel84]

$$\begin{aligned} -\nabla\cdot(\epsilon\nabla u) + q(n-p) &= qf \\ -\nabla\cdot(D_n\nabla n - \mu_n n\nabla u) + R(n,p) &= 0 \\ -\nabla\cdot(D_p\nabla p + \mu_p p\nabla u) + R(n,p) &= 0. \end{aligned} \tag{13}$$

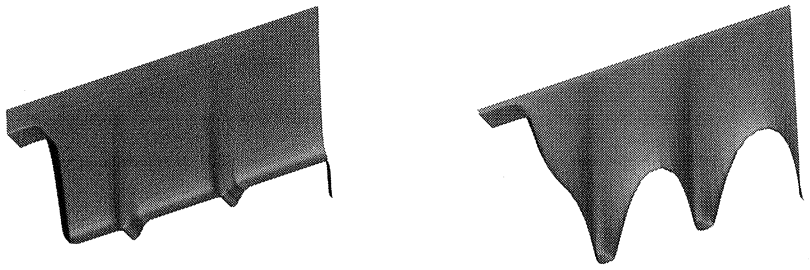


FIGURE 3. Electrostatic potential at $0V$ (equilibrium) and $6V$, $x = 30\mu m$

One example which was impossible to solve using multigrid methods without the numerically stable data structure has been a photo diode sized $30 \times 5 \times 60\mu m^3$ with multiple differently doped horizontal layers. The signed logaritm of the absolute value of the doping concentration f has nearly the same graph as the equilibrium potential which is shown in fig. 3. During the simulation, the diode should be depleted and the recombination current is to be calculated. The changing of the slope at $3.5V$ in the U-I curve (figure 4) is a phenomenon the designers are interested in.

The results calculated using the multigrid method have been compared on different grids with a Tchebyshev-ILU-preconditioned weighted conjugated gradient method (figure 4). For the multigrid method, in both cases two coarser grids and Tchebyshev polynomials of degree 25 as an approximate coarse grid solver have been used. There has been no visible difference between the U-I-curves calculated with the two methods.

References

[AV90] O. Axelsson and P. S. Vassilevski, *Algebraic multilevel preconditioning methods, III*, preprint, Cath. Univ. Dept. of Math., Nijmegen, 1990. Report 9045.

[BMP87] F. Brezzi, L. Marini, and P. Pietra, *Two-dimensional exponential fitting and applications to semiconductor device equations*, preprint, IANCR, Pavia, 1987. Pubblicazioni N.597.

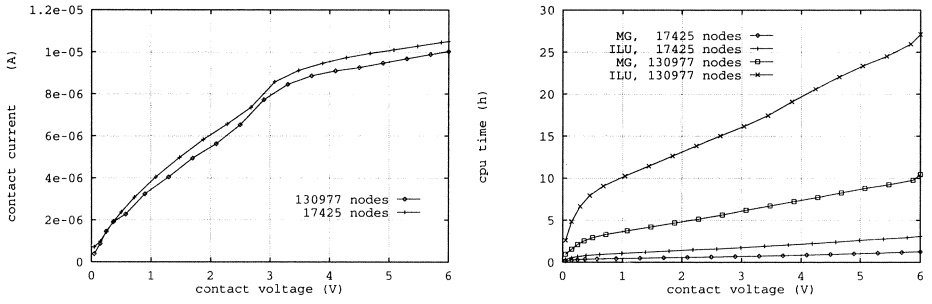


FIGURE 4. I-U-Curve and comparision of CPU time between poly-ILUWCG and MG-WCG (CONVEX C220, 1 processor)

[BPX91] J. Bramble, J. Pasciak, and J. Xu, *The analysis of multigrid algorithms with nonnested spaces or noniherited quadratic forms*, Mathematics of Computation, 56 (1991), 1–34.

[DE89] W. Dahmen and L. Elsner, *Algebraic multigrid methods and the Schur complement*, in Robust Multigrid-Methods, W. Hackbusch, ed., vol. 23 of Notes on numerical fluid mechanics, Vieweg, Braunschweig, 1989, 58–69.

[FG91] J. Fuhrmann and K. Gärtner, *A multigrid method for the solution of a convection - diffusion equation with rapidly varying coefficients*, in Hackbusch and Trottenberg [HT91].

[Fuh90] J. Fuhrmann, *An interpretation of the Scharfetter-Gummel scheme as a mixed finite element discretization*, in Fourth Multigrid Seminar, Unterwirbach, May 1989, G. Telschow, ed., Berlin, 1990, Karl-Weierstraß-Institut für Mathematik. Report R-MATH-03/90.

[Fuh92] ——, *On the convergence of algebraically defined multigrid methods*, preprint no.3, Institut für Angewandte Analysis und Stochastik Berlin, 1992.

[Fuh93a] ——, *An algebraically defined multigrid method*, Journal of Numerical Linear Algebra with Applications, (1993). submitted.

[Fuh93b] ——, *Calculation of saturated-unsaturated flow in porous media with a Newton-multigrid method*, in GAMM-Seminar on Multigrid Meth-

ods, Gosen, September 1992, S. Hengst, ed., Berlin, 1993, Institut für Angewandte Analysis und Stochastik Berlin. Report No. 5.

[GTG+89] K. GÄRTNER, G. TELSCHOW, F. GRUND, H. LANGMACH, J. FUHRMANN, H. SZILLAT, AND C. KEUSCH, *MEDEA- Anwendungsbeschreibung*, technical documentation, Karl-Weierstraß-Institut für Mathematik, Berlin, 1989. unpubl.

[HLM91] G. HAASE, U. LANGER, AND A. MEYER, *The approximate Dirichlet domain decomposition method. part I: An algebraic approach*, Computing, 47 (1991), 137–151.

[HM91] P. W. HEMKER AND J. MOLENAAR, *An adaptive multigrid approach for the solution of the 2d semiconductor equations*, in Hackbusch and Trottenberg [HT91], 41–60.

[HT91] W. HACKBUSCH AND U. TROTTENBERG, eds., *Proceedings of the Third European Multigrid Conference, October 1 - 4,1990, Bonn, Germany*, vol. 98 of ISNM, Basel, 1991, Birkhäuser Verlag.

[Kuz90] Y. KUZNETSOV, *Multigrid domain decomposition methods*, in Proceedings of the Third International Symposium on Domain Decomposition Methods for Partial Differential Equations, Houston, Texas, March 20-22,1989, T. Chan, R. Glowinski, J. Periaux, and O. Widlund, eds., Philadelphia, 1990, SIAM, 290–313.

[Pop91] C. POPA, *ILU decomposition for coarse grid correction step on algebraic multigrid*, in Proceedings of the Third European Multigrid Conference, October 1 - 4,1990, Bonn, Germany, vol. 189 of GMD-Studien, Sankt Augustin, 1991.

[RS87] J. RUGE AND K. STÜBEN, *Algebraic multigrid*, in Multigrid methods, S. McCormick, ed., vol. 4 of Frontiers in Applied Mathematics, SIAM, Philadelphia, 1987, ch. 4, 73–130.

[Sam83] A. SAMARSKIJ, *Teorija raznostnych schem (Theory of difference schemes)*, Nauka, Moskva, 1983.

[Sel84] S. SELBERHERR, *Analysis and simulation of semiconductor devices*, Computational Microelectronics, Springer Verlag, Wien, 1984.

[Son89] W. SONG, *Numerical Methods for the solution of the stationary semiconductor device equations in two and three dimensions*, PhD thesis, University of Dublin, School of Mathematics, 1989.

6

Spectral Multigrid Methods for the Reformulated Stokes Equations

Wilhelm Heinrichs[1]

ABSTRACT We present a spectral multigrid method for the reformulated Stokes equations. Here the continuity equation is replaced by a Poisson equation for the pressure. This system is discretized by a spectral collocation method using only one grid with the Chebyshev Gauss-Lobatto nodes. We present an efficient finite difference preconditioner which is employed for relaxation in the spectral multigrid method. Numerical results are presented which show the efficiency of our treatment.

1 Introduction

We consider a reformulation of the Stokes equations where the continuity equation is replaced by a Poisson-like equation for the pressure. To guarantee full equivalence between the two systems of equations (provided all functions are sufficiently smooth) the continuity equation has to hold at the boundary of the domain. This treatment has been proposed in the famous article of Harlow and Welch [3]. They proposed the MAC method which obtains an equation containing Δp (p denotes the pressure) by differentiating the time-dependent momentum equations and adding them. For finite difference discretizations this approach turned out to be very useful for the efficient solution with multigrid techniques (see [6]). Furthermore this algorithm is not difficult to parallelize and to vectorize.

We were interested in the performance of spectral multigrid methods for the reformulated Stokes equations. Here the spectral discretization is accomplished by a pseudospectral (or collocation) method using Chebyshev polynomials. The collocation points are given by the standard Chebyshev Gauss-Lobatto nodes. In particular, we introduce a Chebyshev collocation method which has the following desirable properties:

- high spectral accuracy,

[1]Mathematisches Institut der Heinrich-Heine-Universität, Universitätsstr. 1, D-40225 Düsseldorf, Germany

- no spurious modes except the physical one (constant mode),
- no staggered grids.

If the continuity equation is directly discretized by a spectral method, the pressure is affected by spurious modes (parassite modes) which deteriorate the accuracy of the method. By our spectral discretization of the reformulated Stokes equations spurious modes (except the constant) are avoided. A similar approach was already introduced by Kleiser and Schumann where the parassite modes are implicitly filtered out by the solution process. A quite good survey about the occurrence of spurious modes for spectral discretizations of the Stokes equations is given in [2, Chapter 11.3].

Furthermore it is well known (see, e.g., [1]) that spurious modes can also be avoided by introducing staggered grids. In [1] the continuity equation is discretized by using the Gauss points instead of the Gauss-Lobatto points. However, this approach is quite expensive in numerical computations since one has to interpolate between these two meshes. Furthermore fast Fourier transforms (FFTs) are no more applicable. Our approach yields high spectral accuracy and can be efficiently implemented by FFTs.

We also present an efficient finite difference preconditioner for the spectral system. The eigenvalues of the preconditioned spectral operator are complex and lie in a circle C which intersects the real axes in the points 1 and $\frac{\pi^2}{4}$. These eigenvalue bounds are already well known from similar considerations for the Poisson equation. Due to the good performance of finite difference preconditioning techniques we prefer a preconditioned Richardson relaxation for the iterative solution of the spectral systems. Finally we show the efficiency of a spectral multigrid method which employs the standard multigrid components (see [4], [5]).

2 Reformulation of the Stokes equations

We consider the steady Stokes equations

$$\begin{aligned} -\Delta u + p_x &= f^u \text{ in } \Omega = (-1,1)^2, && (1)\\ -\Delta v + p_y &= f^v \text{ in } \Omega, && (2)\\ u_x + v_y &= 0 \text{ in } \overline{\Omega} = [-1,1]^2 && (3) \end{aligned}$$

with Dirichlet boundary conditions for the velocity field:

$$u = u_0, \quad v = v_0 \quad \text{on } \partial\Omega.$$

Here f^u, f^v denote given functions defined in Ω. Since the pressure is only determined up to a constant, we impose the average pressure to be zero, i.e., $\int_\Omega p dx = 0$.

In the following theorem we derive a new system which is equivalent to (1)–(3) and which is more appropriate for iterative solvers.

Theorem 2.1. *Let* $u, v \in C^3(\Omega) \cap C^1(\overline{\Omega})$, $p \in C^2(\Omega)$, $f^u, f^v \in C^1(\Omega)$. *Then the Stokes system* (1)–(3) *is equivalent to the following modified system*

$$\begin{aligned} -\Delta u + p_x &= f^u \text{ in } \Omega, && (4)\\ -\Delta v + p_y &= f^v \text{ in } \Omega, && (5)\\ \Delta p &= f^u_x + f^v_y \text{ in } \Omega, && (6)\\ u_x + v_y &= 0 \text{ on } \partial\Omega. && (7) \end{aligned}$$

Proof. The sum of the derivatives of equation (1) with respect to x and (2) with respect to y yields (6). Here we make use of the fact that due to (3)

$$-\Delta(u_x + v_y) = 0 \text{ in } \overline{\Omega}. \quad (8)$$

In the other direction, equation (3) can be regained from the equations (4) -(7) because the difference of the sum of the derivatives of equations (4) with respect to x and (5) with respect to y and equation (6) yields equation (8) and this partial differential equation with the homogeneous Dirichlet boundary conditions (7) interpreted as a boundary value problem for $u_x + v_y$ has the unique solution (3). $\oplus$

This result can easily be generalized to the Navier–Stokes equations.
For the new Stokes systems we have three boundary conditions for the velocity components but no condition for the pressure. A similar problem also occured for the biharmonic equation. Here we propose a splitting into a second order system with the streamfunction and vorticity. There are two boundary conditions for the streamfunction and no condition for the vorticity.

3 Spectral discretization

In order to present the spectral discretization of the reformulated Stokes equations (4)–(7) we first give some notations. For $N \in \mathbf{N}$ we introduce the following polynomial subspaces:

$$\begin{aligned} \mathbf{P}_N &= \{\text{polynomials of degree less or equal } N \text{ in } x, y\},\\ \mathbf{P}^0_{N+2,N} &= \{(1-x^2)^2(1-y^2)p_{N-2} : \ p_{N-2} \in \mathbf{P}_{N-2}\},\\ \mathbf{P}^0_{N,N+2} &= \{(1-x^2)(1-y^2)^2 p_{N-2} : \ p_{N-2} \in \mathbf{P}_{N-2}\},\\ \mathbf{P}^0_N &= \{(1-x^2)(1-y^2)p_{N-2} : \ p_{N-2} \in \mathbf{P}_{N-2}\}. \end{aligned}$$

Hence the polynomials $p \in \mathbf{P}^0_{N+2,N}$ fulfill $p = p_x = 0$ for $x = \pm 1$, $p = 0$ for $y = \pm 1$ and the polynomials $p \in \mathbf{P}^0_{N,N+2}$ fulfill $p = p_y = 0$ for $y = \pm 1$, $p = 0$ for $x = \pm 1$.

The collocation points are given by the standard Chebyshev Gauss-Lobatto nodes:

$$(x_i, y_j) \;=\; (\cos\frac{i\pi}{N}, \cos\frac{j\pi}{N})\;, \quad i,j = 0,\ldots,N.$$

Furthermore we introduce the following collocation grids:

$$\begin{aligned}
\overline{\Omega}_N &= \{(x_i,y_j):\ i,j=0,\ldots,N\},\\
\Omega_N &= \overline{\Omega}_N \cap \Omega,\\
\Omega_N^{co} &= \{(1,1),(-1,1),(1,-1),(-1,-1)\},\\
\Omega_N^{u} &= \{(x_i,y_j):\ i=0,\ldots,N,\ j=1,\ldots,N-1\}\ \cup\ \Omega_N^{co},\\
\Omega_N^{v} &= \{(x_i,y_j):\ i=1,\ldots,N-1,\ j=0,\ldots,N\}.
\end{aligned}$$

Now the pseudospectral (or collocation) discretization of the reformulated Stokes system (4)–(7) associated with homogeneous Dirichlet boundary conditions (i.e. $u_0 = v_0 = 0$) reads as follows: Find $u_{N+2,N} \in \mathbf{P}^0_{N+2,N}$, $v_{N,N+2} \in \mathbf{P}^0_{N,N+2}$, $p_N \in \mathbf{P}_N$ such that

$$-\Delta u_{N+2,N} + p_{N,x} = f^u \text{ in } \Omega_N^u, \tag{9}$$

$$-\Delta v_{N,N+2} + p_{N,y} = f^v \text{ in } \Omega_N^v, \tag{10}$$

$$\Delta p_N = f^u_x + f^v_y \text{ in } \Omega_N. \tag{11}$$

In (9)–(11) we choose an implicit treatment of the boundary conditions. Since $u = v = 0$ on $\partial\Omega$ we obtain from (7) $u_x = 0$ along the axes $x = \pm 1$ and $v_y = 0$ along the axes $y = \pm 1$. This means that we have two boundary conditions for u in $x = \pm 1$ and two boundary conditions for v in $y = \pm 1$. We approximate u, v by polynomials $u_{N+2,N} \in \mathbf{P}^0_{N+2,N}$, $v_{N,N+2} \in \mathbf{P}^0_{N,N+2}$ which have two more degrees of freedom in the direction where two boundary conditions have to be enforced. Since the boundary conditions are treated implicitly we have to give collocation conditions on the momentum equations in u for $x = \pm 1$ and v for $y = \pm 1$. Furthermore the boundary conditions $u_x + v_y = 0$ in the corners are automatically fulfilled if $u = v = 0$ on $\partial\Omega$. Hence these four conditions have to be replaced by four conditions belonging to the momentum equations. Here we impose that (4) also holds in the four corners of the domain. Since $\Delta u = 0$ in the four corners we obtain the four additional conditions:

$$p_{N,x} = f^u \text{ in the corners.}$$

These conditions are enclosed in the equations (9) which have to be valid in Ω_N^u. The system (9)–(11) requires

$$M \;=\; 2(N-1)^2 + (N+1)^2 \;=\; 3N^2 - 2N + 3$$

conditions of collocation for the M unknown coefficients of $u_{N+2,N}$, $v_{N,N+2}$, p_N. Clearly, p_N is only determined up to a constant and has to be normalized such

that $\int_\Omega p_N dx = 0$.

A well known problem for spectral methods applied to the Stokes equation results from the occurrence of certain "spurious modes" or "parassite modes" for the pressure. They are given by those non-trivial polynomials $q_N \in \mathbf{P}_N$ which yield zero for all collocation conditions. For the implicit scheme (9)–(11) it is easy to show that the spectral discretization does not introduce spurious modes for the pressure.

4 Preconditioning

Since the spectral operator has a very large condition number growing as $O(N^4)$ we present an efficient finite difference (FD) preconditioner. For the Laplace operator it is well known (see [4], [5]) that the preconditioned spectral operator has real, positive eigenvalues lying in the interval $[1, \frac{\pi^2}{4}]$. This can still be improved by using bilinear finite element preconditioning where the eigenvalues are confined to the interval $[0.693, 1]$. Here we consider FD preconditioning.

For the second derivative we employed standard central differences with respect to the Chebyshev mesh. In the boundary points we introduced outer points which were eliminated by using the Neumann boundary conditions (for the velocity components). The first derivatives in boundary points are approximated by one-sided finite differences. Now we consider the spectral system (9)–(11) where the continuity equation is treated implicitly. The corresponding spectral resp. FD operators are written as

$$L_{sp} \in \mathbf{R}^{M,M} \quad \text{resp.} \quad L_{FD} \in \mathbf{R}^{M,M}.$$

Clearly, the eigenvalues of L_{sp}, L_{FD} are complex where the absolute value increases as $O(N^4)$. One eigenvalue is zero due to the presence of one spurious mode which is identical to the constant. In order to get regular operators we considered the operators which are obtained by eliminating the last column and row of the system. In table I we present the (absolutely) minimal and maximal eigenvalues λ_{min} and λ_{max} of the preconditioned spectral operator.

N	λ_{min}	λ_{max}
4	1.00	2.4284
8	1.00	2.3867
12	1.00	2.4231

Table I. λ_{min}, λ_{max} of $(L_{FD})^{-1} L_{sp}$.

The eigenvalues belonging to λ_{min} and λ_{max} are real. Hence the absolute values of the eigenvalues lie in the interval $[1, \frac{\pi^2}{4}]$. In the figure we plotted the eigenvalues for $N = 12$. It can be seen that the imaginary parts are relatively small, always

less than 0.5. The eigenvalues with the largest imaginary parts lie in the "middle" of the eigenvalue spectrum with a real part of about $1.6 - 1.7$.

Numerically it can be seen that almost all eigenvalues lie in the circle C given by

$$C = \{(\rho,\sigma): (\rho - \frac{1}{2}(1+\frac{\pi^2}{4}))^2 + \sigma^2 \leq \frac{1}{4}\left(\frac{\pi^2}{4}-1\right)^2\}.$$

Only the complex eigenvalue with smallest real part (about 0.99) is a little bit outside of C. But this will not disturb the convergence if we choose the relaxation parameters based on C. For a Richardson relaxation (see [4]) we therefore choose the relaxation parameter ω equal to

$$\omega = \frac{2}{1+\frac{\pi^2}{4}}.$$

5 Spectral multigrid method

In the spectral multigrid (SMG) method we employ the standard grid transfer operators (see [4], [5]) for interpolation and restriction. For smoothing we employ the Richardson relaxation with finite difference preconditioning. The finite difference problem is approximately solved by one or two steps of an (alternating) zebra line Gauss-Seidel relaxation. Here one first solves along lines of constant x and afterwards along lines of constant y. Vectorization is achieved by first solving for the odd lines and then for the even lines, resulting in zebra line relaxation. In the numerical computations we generally use a V-cycle with two ($N = 4, 8$), three ($N = 4, 8, 16$) or four ($N = 4, 8, 16, 32$) grids. In order to measure the convergence speed of the (SMG) method we calculated the spectral radius ρ of the SMG operator by means of the power method. A convergence factor which is related to the computational work can be defined by $\rho_W = \rho^{\frac{1}{W}}$ where $W = n_d + n_u$ and n_d resp. n_u denote the number of relaxations on each grid in the downward resp. upward branches. ρ_W does not take the total computational work into account but it should be near the smoothing rate and provide an estimate of efficiency. From numerical experiments we found that the number of relaxations should be

$$n_d = n_u = 3$$

for both the stationary Richardson (SR) and nonstationary Richardson (NSR) relaxation. The number of preconditioniong (PC) steps should be 2. In table II we give the results for the SMG method.

Number of grids	Relaxation	1 PC step	2 PC steps
2	SR	0.5834	0.4346
2	NSR	0.5286	0.4000
3	SR	0.6223	0.4641
3	NSR	0.5420	0.4269
4	SR	0.7104	0.4769
4	NSR	0.5415	0.4440

Table II. ρ_W for the SMG method.

The numerical results substantiate the usefulness of SMG. Furthermore it shows the improvements by choosing 2 PC steps of the Gauss-Seidel line relaxation. 1 PC step is not enough for a good smoothing of the high frequencies. If we employ 2 steps the convergence rates are quite similar to the rates we already observed for the Poisson equation (see [4]). Similar results were also obtained by other cycle structures. For instance, the W-cycle could not improve the convergence factors. The rates were nearly the same as for the V-cycle.

6 References

1. Bernardi,C. and Maday,Y.: A collocation method over staggered grids for the Stokes problem. Int. J. Num. Meth. Fluids **8**, 537-557(1988)

2. Canuto,C., Hussaini,M.Y., Quarteroni,A., and Zang,T.A.: Spectral Methods in Fluid Dynamics, 1st Ed., Springer Series in Computational Physics, Springer-Verlag, Berlin-Heidelberg-New York, 1989

3. Harlow,F.H. and Welch,J.E.: Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. Physics of Fluids **8**, 2182-2189(1965)

4. Heinrichs,W.: Line relaxation for spectral multigrid methods. J. Comp. Phys. **77**, 166-182(1988)

5. Heinrichs,W.: Multigrid methods for combined finite difference and Fourier problems. J. Comp. Phys. **78**, 424-436(1988)

6. Schüller,A.: A multigrid algorithm for the incompressible Stokes equations, *in* Numerical Treatment of the Stokes Equations, Proceedings of the Fifth GAMM-Seminar, edited by W.Hackbusch and R.Rannacher, Vieweg, Braunschweig 1990

7

Multigrid Waveform Relaxation on Spatial Finite Element Meshes

Jan Janssen [1] **and Stefan Vandewalle**[2]

ABSTRACT [3] We analyze in this paper the convergence of multigrid waveform relaxation methods, applied to a system of ordinary differential equations, obtained by a finite element discretization of a parabolic initial boundary value problem. We restrict the discussion to linear systems and consider both the continuous-time and the discrete-time cases.

1 Introduction

Consider a parabolic partial differential equation (PDE)

$$\frac{\partial \mathbf{u}}{\partial t}(x,t) = \mathcal{L}\mathbf{u}(x,t) + \mathbf{f}(x,t) \quad x \in \Omega,\ t > 0\ , \tag{1}$$

with a linear boundary condition and given initial values. $\mathcal{L}$ is a linear second order elliptic operator with time-independent coefficients and Ω is a compact spatial domain. A spatial finite element discretization on a mesh Ω_h with mesh size h transforms (1) into a linear system of ordinary differential equations (ODEs),

$$B\dot{u} + Au = f\ ,\ \text{ with }\ u(0) = u_0\ ,\ \ t > 0\ , \tag{2}$$

where B is a non-singular constant-coefficient matrix.

The standard waveform relaxation method, also called the dynamic iteration method, is an iterative technique for solving systems of ordinary differential equations. Its background is in electrical network simulation, [4]. It differs from

[1]Katholieke Universiteit Leuven, Department of Computing Science, Celestijnenlaan 200A, B-3001 Leuven, Belgium.

[2]Senior Research Assistent of the National Fund for Scientific Research (N.F.W.O.), Belgium. Katholieke Universiteit Leuven, Department of Computing Science, Celestijnenlaan 200A, B-3001 Leuven, Belgium.

[3]The following text presents research results of the Belgian Incentive Program "Information Technology" - Computer Science of the future, initiated by the Belgian State - Prime Minister's Service - Science Policy office. The scientific responsibility is assumed by its authors.

most standard iterative techniques in that it is a continuous-time method, iterating with functions. A discrete-time variant can be obtained by discretizing the continuous-time method in time, with, e.g., a linear multistep or Runge-Kutta method.

The discretization of (1) with finite differences results in a system of ODEs of the form (2) where B is the identity matrix. For such systems, the convergence of the standard waveform relaxation method has been studied exhaustively. Miekkala and Nevanlinna considered the continuous-time case in [7] and [9]. They formulated the convergence characteristics of the method in terms of a complex iteration matrix, obtained after Laplace-transforming the problem. Discrete-time results are obtained by the same authors in [8] and [10]. The multigrid acceleration of the waveform relaxation method was studied by Lubich and Ostermann in [5], where both the continuous-time and the discrete-time cases are considered. A complete survey and a discussion of a parallel implementation of these methods can be found in the work by Vandewalle, [11], and in the references cited therein. We also mention a paper by Miekkala, [6], where the convergence properties of the standard waveform relaxation method are studied for differential-algebraic systems of the form (2) where B is possibly singular.

In this paper, we shall concentrate on (2) with non-singular B and generalize some of the results of [5], [7] and [8]. This paper is a summary of [2] and [3] where further theoretical results and numerical experiments are reported. Miekkala's results are briefly recalled in section 2. These results are completed with a convergence analysis for the discrete-time standard waveform relaxation method. In section 3, the multigrid acceleration of this method for (2) is theoretically investigated. In section 4, we give some specific theoretical results for the heat equation, which are validated by numerical experiments.

2 Standard waveform relaxation

2.1 The continuous-time case

The continuous-time waveform relaxation operator

Consider the linear initial value problem (2) where B and A are complex $d \times d$ matrices, and u and f are $\mathbb{C}^d$-valued functions in time. B is assumed to be non-singular. Its solution is formally given by

$$u(t) = e^{-tB^{-1}A}u_0 + \int_0^t e^{(s-t)B^{-1}A}B^{-1}f(s)ds\ . \tag{3}$$

The waveform relaxation method for solving (2) is defined by introducing a splitting of the matrices B and A. With $B = M_B - N_B$ and $A = M_A - N_A$, the basic continuous-time waveform relaxation iteration can be written as

$$M_B\dot{u}^{(\nu)} + M_A u^{(\nu)} = N_B\dot{u}^{(\nu-1)} + N_A u^{(\nu-1)} + f\ ,\ \text{with}\ u^{(\nu)}(0) = u_0\ ,\ t > 0\ , \tag{4}$$

which is usually started by choosing the zeroth iterate $u^{(0)}(t) = u_0$, $t > 0$. We shall always assume M_B to be invertible. By using (3), we can rewrite (4) as an explicit successive approximation scheme:

$$u^{(\nu)} = \mathcal{K}u^{(\nu-1)} + \varphi \ . \tag{5}$$

The right-hand side function φ can be found in the companion report, [2]. The *continuous-time waveform relaxation operator* $\mathcal{K}$ is found to be

$$\mathcal{K}u(t) = M_B^{-1}N_B u(t) + \mathcal{K}_c u(t) \ , \tag{6}$$

where $\mathcal{K}_c$ is a linear Volterra operator with a continuous kernel,

$$\mathcal{K}_c u(t) = \int_0^t k_c(t-s)u(s)ds \ , \tag{7}$$

$$k_c(t) = e^{-tM_B^{-1}M_A}M_B^{-1}(N_A - M_A M_B^{-1}N_B) \ . \tag{8}$$

Let $e^{(\nu)}$ be the error of the ν-th waveform relaxation iterate, i.e., $e^{(\nu)} = u^{(\nu)} - u$. It satisfies $e^{(\nu)} = \mathcal{K}e^{(\nu-1)}$. That is, it is the solution to

$$M_B\dot{e}^{(\nu)} + M_A e^{(\nu)} = N_B\dot{e}^{(\nu-1)} + N_A e^{(\nu-1)} \ , \quad \text{with} \quad e^{(\nu)}(0) = 0 \ , \quad t > 0 \ . \tag{9}$$

Denoting by $\tilde{e}^{(\nu)}(z)$ the Laplace-transform of $e^{(\nu)}$, we obtain by Laplace-transforming (9) that $\tilde{e}^{(\nu)} = \mathbf{K}(z)\tilde{e}^{(\nu-1)}$, with

$$\mathbf{K}(z) = (zM_B + M_A)^{-1}(zN_B + N_A) \ . \tag{10}$$

$\mathbf{K}(z)$ is called the dynamic iteration matrix of operator $\mathcal{K}$.

Convergence analysis

The convergence of the waveform relaxation operator is often studied in general Banach spaces. In particular, we consider the space of p-th power integrable Lebesgue measurable functions $L_p((0,\infty);\mathbb{C}^d)$, or $L_p(0,\infty)$ for short, with the usual mean p-norm, and the space of continuous functions $C([0,T];\mathbb{C}^d)$, or $C[0,T]$, equipped with the maximum norm.

The spectral radius $\rho(\mathcal{U})$ of a bounded linear operator $\mathcal{U}$ in a complex normed linear space is characterized by

$$\rho(\mathcal{U}) = \lim_{n\to\infty} \sqrt[n]{||\mathcal{U}^n||} \ ,$$

and convergence of the general successive approximation scheme,

$$x^{(\nu)} = \mathcal{U}x^{(\nu-1)} + \varphi \ ,$$

is guaranteed if and only if $\rho(\mathcal{U}) < 1$.

In [2], we analyzed the convergence behaviour of the continuous-time waveform relaxation operator, both on finite and infinite time-intervals. The two main results are stated below. Supplied with some extra assumptions, Theorem 2 also holds for differential-algebraic systems with singular B, [6].

Theorem 1 (finite time-interval) *Consider $\mathcal{K}$ as an operator in $C[0,T]$. Then,*

$$\rho(\mathcal{K}) = \rho(M_B^{-1} N_B) \ . \tag{11}$$

Theorem 2 (infinite time-interval) *Consider $\mathcal{K}$ as an operator in $L_p(0,\infty)$ with $1 \leq p \leq \infty$, and assume that all eigenvalues of $M_B^{-1} M_A$ have positive real parts. Then,*

$$\rho(\mathcal{K}) = \sup_{\xi \in I\!R} \rho(\mathbf{K}(i\xi)) \ . \tag{12}$$

Remark 1. The previous results can also be applied when B is the identity matrix, i.e., when we discretize our PDE using finite differences. Indeed, if $M_B = I$ and $N_B = 0$, we obtain the same results as in [7].

2.2 The discrete-time case

The discrete-time waveform relaxation operator

We recall the general linear multistep formula for calculating the solution to the ordinary differential equation $\dot{y} = f(t,y)$, $y(0) = y_0$,

$$\frac{1}{\tau} \sum_{j=0}^{k} \alpha_j y_{n+j} = \sum_{j=0}^{k} \beta_j f_{n+j} \ .$$

τ denotes a constant step-size; α_j and β_j are real constants, and y_j approximates the ODE solution at time-level $t = j\tau$. The fully discrete solution $\{y_i\}_{i=0}^{N_t}$, with N_t the number of time-steps, will be denoted further by y_τ. The characteristic polynomials of the linear multistep method are $a(\xi) = \sum_{j=0}^{k} \alpha_j \xi^j$ and $b(\xi) = \sum_{j=0}^{k} \beta_j \xi^j$. We shall adhere to the usual assumptions: $a(\xi)$, $b(\xi)$ have no common roots; $a(1) = 0$, $a'(1) = b(1)$; all roots of $a(\xi)$ are inside the closed unit disk, and every root with modulus one is simple.

Application of the linear multistep formula to (9) leads to

$$\frac{1}{\tau} \sum_{j=0}^{k} \alpha_j M_B e_{n+j}^{(\nu)} + \sum_{j=0}^{k} \beta_j M_A e_{n+j}^{(\nu)} = \frac{1}{\tau} \sum_{j=0}^{k} \alpha_j N_B e_{n+j}^{(\nu-1)} + \sum_{j=0}^{k} \beta_j N_A e_{n+j}^{(\nu-1)} \ , \tag{13}$$

with $n \geq 0$ and $e_\tau^{(\nu)} = u_\tau^{(\nu)} - u_\tau$ the error of the ν-th discrete waveform relaxation iterate. For simplicity's sake, we assume that there are k fixed starting values supplied, i.e., $u_j^{(\nu)} = u_j^{(\nu-1)} = u_j$ for $j < k$. So, we do not iterate on the starting values. Iteration (13) can be rewritten as $e_\tau^{(\nu)} = \mathcal{K}_\tau e_\tau^{(\nu-1)}$. In [3] it is shown that the *discrete-time waveform relaxation operator* $\mathcal{K}_\tau$ is a discrete convolution operator:

$$(\mathcal{K}_\tau x_\tau)_j = (k_\tau \star x_\tau)_j = \sum_{i=0}^{j} k_{j-i} x_i \ . \tag{14}$$

Its kernel, k_τ, is related to the dynamic iteration matrix of $\mathcal{K}_\tau$ by the ξ-transform (or discrete Laplace-transform),

$$\mathbf{K}_\tau(\xi) := \sum_{i=0}^{\infty} k_i \xi^{-i} = \left(\frac{1}{\tau}\frac{a}{b}(\xi) M_B + M_A\right)^{-1} \left(\frac{1}{\tau}\frac{a}{b}(\xi) N_B + N_A\right) , \tag{15}$$

which is equal to $\mathbf{K}(z)$ for $z = \frac{1}{\tau}\frac{a}{b}(\xi)$.

Convergence results

The convergence of the discrete waveform relaxation is studied in the Banach spaces of p-summable sequences $l_p(0..N_t)$, equipped with the mean p-norm.

The discrete versions of Theorem 1 and 2 are given below. Their proof is given in [3].

Theorem 3 (finite time-interval) *Consider $\mathcal{K}_\tau$ as an operator in $l_p(0..N_t)$ with $1 \le p \le \infty$. Let the number of time-steps, N_t, be finite and suppose that $\frac{\alpha_k}{\beta_k} \notin \sigma(-\tau M_B^{-1} M_A)$. Then,*

$$\rho(\mathcal{K}_\tau) = \rho\left(\mathbf{K}(\frac{1}{\tau}\frac{\alpha_k}{\beta_k})\right) . \tag{16}$$

Theorem 4 (infinite time-interval) *Consider $\mathcal{K}_\tau$ as an operator in $l_p(\mathbb{N})$ with $1 \le p \le \infty$ and suppose $\sigma(-\tau M_B^{-1} M_A) \subset$ int S. Then,*

$$\rho(\mathcal{K}_\tau) = \sup_{|\xi|=1} \rho\left(\mathbf{K}\left(\frac{1}{\tau}\frac{a}{b}(\xi)\right)\right) . \tag{17}$$

3 Multigrid waveform relaxation

3.1 The continuous-time case

The continuous-time two-grid waveform relaxation operator

Multigrid methods are known to be very efficient solvers for elliptic partial differential equations. We refer to [1] and [12] for a detailed analysis. The principle can easily be extended to time-dependent problems by choosing all the operations in the multigrid cycle as operations on functions. A two-grid cycle for the initial value problem (2) is stated below. It is defined on two nested grids Ω_H and Ω_h, with $\Omega_H \subset \Omega_h$, and determines a new iterate $u^{(\nu)}$ from the former waveform $u^{(\nu-1)}$ in three steps: pre-smoothing, coarse grid correction, and post-smoothing. In the following, the subscripts h and H are used to denote fine and coarse grid quantities respectively.

- **Pre-smoothing.** Set $x^{(0)} = u^{(\nu-1)}$, and perform ν_1 waveform relaxation steps: for $\nu = 1, 2, \ldots, \nu_1$, solve

$$M_{B_h} \dot{x}^{(\nu)} + M_{A_h} x^{(\nu)} = N_{B_h} \dot{x}^{(\nu-1)} + N_{A_h} x^{(\nu-1)} + f_h , \tag{18}$$

with $x^{(\nu)}(0) = u_0$, $t > 0$.

- **Coarse grid correction.** Compute the defect,

$$\begin{aligned} d_h &= B_h \dot{x}^{(\nu_1)} + A_h x^{(\nu_1)} - f_h \\ &= N_{B_h}(\dot{x}^{(\nu_1-1)} - \dot{x}^{(\nu_1)}) + N_{A_h}(x^{(\nu_1-1)} - x^{(\nu_1)}) \ . \end{aligned}$$

Solve the coarse grid equivalent of the defect equation,

$$B_H \dot{v}_H + A_H v_H = r d_h \ , \quad \text{with} \quad v_H(0) = 0 \ , \quad t > 0 \ , \tag{19}$$

with $r : \Omega_h \to \Omega_H$ the restriction operator. Then interpolate the correction v_H to Ω_h, and correct the current approximation,

$$\bar{x} = x^{(\nu_1)} - p v_H \ ,$$

with $p : \Omega_H \to \Omega_h$ the prolongation operator.

- **Post-smoothing.** Perform ν_2 iterations of type (18), starting with $x^{(0)} = \bar{x}$, and set $u^{(\nu)} = x^{(\nu_2)}$.

Since (19) is formally equal to (2), this two-grid cycle can be applied in a recursive way to obtain a multigrid cycle.

The two-grid cycle can be written as a successive approximation scheme:

$$u^{(\nu)} = \mathcal{M} u^{(\nu-1)} + \varphi \ . \tag{20}$$

The *continuous-time two-grid waveform relaxation operator* $\mathcal{M}$ is given by

$$\mathcal{M}u(t) = (M_{B_h}^{-1} N_{B_h})^{\nu_2} (I - p B_H^{-1} r B_h)(M_{B_h}^{-1} N_{B_h})^{\nu_1} u(t) + \mathcal{M}_c u(t) \ . \tag{21}$$

$\mathcal{M}_c$ is a linear Volterra convolution operator with kernel m_c, whose Laplace-transform is denoted by $\mathbf{M_c}(z)$. Let $e^{(\nu)} = u^{(\nu)} - u$ be the error of the ν-th two-grid waveform relaxation iterate. This error satisfies the relation $e^{(\nu)} = \mathcal{M} e^{(\nu-1)}$. Laplace-transforming yields

$$\tilde{e}^{(\nu)} = \mathbf{M}(z) \tilde{e}^{(\nu-1)} \ , \tag{22}$$

with $\mathbf{M}(z)$ the two-grid dynamic iteration matrix of $\mathcal{M}$,

$$\mathbf{M}(z) = \mathbf{S}^{(\nu_2)}(z)(I - p(zB_H + A_H)^{-1} r (zB_h + A_h))\mathbf{S}^{(\nu_1)}(z) \ , \tag{23}$$

$$\mathbf{S}(z) = (zM_{B_h} + M_{A_h})^{-1}(zN_{B_h} + N_{A_h}) \ . \tag{24}$$

Convergence analysis

The following theorems are the two-grid analogues of Theorem 1 and Theorem 2, and are proven in [2].

Theorem 5 (finite time-interval) *Consider $\mathcal{M}$ as an operator in $C[0,T]$. Then,*

$$\rho(\mathcal{M}) = \rho\left((M_{B_h}^{-1}N_{B_h})^{\nu_2}(I - pB_H^{-1}rB_h)(M_{B_h}^{-1}N_{B_h})^{\nu_1}\right) . \tag{25}$$

Theorem 6 (infinite time-interval) *Consider $\mathcal{M}$ as an operator in $L_p(0,\infty)$ with $1 \leq p \leq \infty$, and assume that all eigenvalues of $B_H^{-1}A_H$ and $M_{B_h}^{-1}M_{A_h}$ have positive real parts. Then,*

$$\rho(\mathcal{M}) = \sup_{\xi \in I\!R} \rho(\mathbf{M}(i\xi)) . \tag{26}$$

Remark 2. The previous results can also be applied when both B_H and B_h are identity matrices, i.e., when we discretize our PDE using finite differences. In that case, we obtain the same results as in [11, Th. 3.4.1] and [5, Prop. 1].

3.2 The discrete-time case

The discrete-time two-grid waveform relaxation operator

We discretize the two-grid operator in time using a linear multistep formula, and we assume again that we do not iterate on the k given starting values. The error $e_\tau^{(\nu)} = u_\tau^{(\nu)} - u_\tau$ of the fully discrete ν-th two-grid iterate then satisfies $e_\tau^{(\nu)} = \mathcal{M}_\tau e_\tau^{(\nu-1)}$. In an analogous manner as in the previous section, it can be shown that the *discrete-time two-grid waveform relaxation operator* $\mathcal{M}_\tau$ is a discrete convolution operator. The ξ-transform of its kernel m_τ equals

$$\mathbf{M}_\tau(\xi) := \sum_{i=0}^{\infty} m_i \xi^{-i} = \mathbf{M}\left(\frac{1}{\tau}\frac{a}{b}(\xi)\right) . \tag{27}$$

Convergence analysis

The convergence of the discrete-time two-grid waveform relaxation is studied in [3]. The following results are very similar to the results obtained for the discrete-time standard waveform relaxation operator.

Theorem 7 (finite time-interval) *Consider $\mathcal{M}_\tau$ as an operator in $l_p(0..N_t)$ with $1 \leq p \leq \infty$. Let the number of time-steps, N_t, be finite and assume none of the poles of $\mathbf{M}(z)$ is equal to $\frac{\alpha_k}{\beta_k}$. Then,*

$$\rho(\mathcal{M}_\tau) = \rho\left(\mathbf{M}(\frac{1}{\tau}\frac{\alpha_k}{\beta_k})\right) . \tag{28}$$

Theorem 8 (infinite time-interval) *Consider $\mathcal{M}_\tau$ as an operator in $l_p(I\!N)$ with $1 \leq p \leq \infty$ and suppose all the poles of $\mathbf{M}(z)$ are in the interior of the scaled stability region $\frac{1}{\tau}S$. Then,*

$$\rho(\mathcal{M}_\tau) = \sup_{|\xi|=1} \rho\left(\mathbf{M}(\frac{1}{\tau}\frac{a}{b}(\xi))\right) . \tag{29}$$

4 Model problem analysis and numerical results

In this section, we verify our theoretical results on the basis of numerical experiments with two model problems.

Model problem 1: the one-dimensional heat equation

$$\frac{\partial \mathbf{u}}{\partial t}(x,t) = \frac{\partial^2 \mathbf{u}}{\partial x^2}(x,t) \quad x \in [0,1] \ , \quad t > 0 \ , \tag{30}$$

completed with homogeneous Dirichlet boundary conditions and an initial condition. The analytical solution is given by $\mathbf{u}(x,t) = \sin(\pi x)\exp(-\pi^2 t)$.
Model problem 2: the two-dimensional heat equation

$$\frac{\partial \mathbf{u}}{\partial t}(x,y,t) = \frac{\partial^2 \mathbf{u}}{\partial x^2}(x,y,t) + \frac{\partial^2 \mathbf{u}}{\partial y^2}(x,y,t) \quad (x,y) \in [0,1]\times[0,1] \ , \quad t > 0 \ , \tag{31}$$

completed with Dirichlet boundary conditions and an initial condition such that the analytical solution is given by $\mathbf{u}(x,y,t) = 1+\sin(\pi x/2)\sin(\pi y/2)\exp(-\pi^2 t/2)$.

Discretizing these model problems on a discrete mesh Ω_h yields systems of the form (2). The matrices B and A are listed for several finite element basis functions in [2].

4.1 Gauss-Seidel waveform relaxation

Theoretical results

In order to determine the spectral radius of the continuous-time Gauss-Seidel waveform relaxation method, the spectral radius of $\mathbf{K}(z)$ is to be calculated for every value of z along the imaginary axis. This is generally a very difficult task. However, there are some cases where $\rho(\mathcal{K})$ can be calculated explicitly.

Assume B and A to be decomposed as $B = -L_B + D_B - U_B$ and $A = -L_A + D_A - U_A$, with D_B and D_A diagonal matrices, L_B and L_A strictly lower and U_B and U_A strictly upper triangular matrices. Let $\mathcal{K}_{GS}$ and $\mathcal{K}_{JAC}$ denote the Gauss-Seidel and Jacobi waveform relaxation operator respectively. Following [9], their dynamic iteration matrices are given by:

$$\begin{aligned} \mathbf{K_{GS}}(z) &= (z(D_B - L_B) + (D_A - L_A))^{-1}(zU_B + U_A) \ , \\ \mathbf{K_{JAC}}(z) &= (zD_B + D_A)^{-1}(z(L_B + U_B) + (L_A + U_A)) \ . \end{aligned}$$

Lemma 4 *Assume that all eigenvalues of $(D_B - L_B)^{-1}(D_A - L_A)$ have positive real parts. Let A and B be such that $(zB + A)$ is a consistently ordered matrix for $Re(z) \geq 0$. Then, in $L_p(0,\infty)$ with $1 \leq p \leq \infty$,*

$$\rho(\mathcal{K}_{GS}) = \rho(\mathcal{K}_{JAC})^2 \ . \tag{32}$$

This Lemma is proven in [2]. Discretizing model problem 1 with linear basis functions yields

$$\rho(\mathbf{K_{JAC}}(z)) = \left|\frac{-2zh^2+12}{4zh^2+12}\right| \cos(\pi h) \ .$$

Hence, as a consequence of Theorem 2, equation (12), we find that

$$\begin{aligned} \rho(\mathcal{K}_{JAC}) &= \sup_{\xi \in I\!R} \left|\frac{-2i\xi h^2+12}{4i\xi h^2+12}\right| \cos(\pi h) = \cos(\pi h) \\ &\approx 1 - \pi^2 h^2/2 \ . \end{aligned}$$

Since the assumptions of Lemma 4 are satisfied, we have for small h

$$\rho(\mathcal{K}_{GS}) \approx 1 - \pi^2 h^2 \ . \tag{33}$$

Numerical results

For the time-discretization of (2), we use the Crank-Nicolson formula. When the time-step τ is taken sufficiently small, the continuous-time theoretical results should fit the obtained results of the numerical experiments. For the ν-th waveform relaxation iterate, we determine the l_2-norm of the defect $d_\tau^{(\nu)} = B\dot{u}_\tau^{(\nu)} + Au_\tau^{(\nu)} - f_\tau$. The iteration convergence factor is then calculated as

$$\rho^{(\nu)} = \frac{||d_\tau^{(\nu)}||_2}{||d_\tau^{(\nu-1)}||_2} \ .$$

After a sufficiently large number of iterations this factor takes a nearly constant value, the *averaged convergence factor.*

The numerical results for model problem 1, discretized using linear basis functions, are given in Table 1. Even though the time-interval in this experiment is finite, the measured convergence factors closely match the infinite interval theoretical spectral radii of (33). For a discussion of this phenomenon, we refer to [11].

Table 1 also reports the averaged convergence factors for the same problem, discretized using quadratic basis functions. Observe that these factors seem to satisfy a relation of the form

$$\rho(\mathcal{K}_{GS}) \approx 1 - O(h^2) \ ,$$

although no explicit theoretical formula was found. The same is true for the results of Table 2, where we reported the averaged convergence factors for model problem 2, discretized with bilinear basis functions on an equidistant rectangular grid.

h	1/5	1/10	1/15	1/20	1/30	1/40
Linear	0.630	0.898	0.953	0.973	0.988	0.993
$1-\pi^2h^2$	0.605	0.901	0.956	0.975	0.989	0.994
Quadratic	0.917	0.978	0.991	0.995	0.998	0.999

TABLE 1. Gauss-Seidel waveform relaxation averaged convergence factors for the one-dimensional heat equation.

h	1/5	1/10	1/15	1/20	1/30	1/40
Bilinear	0.530	0.854	0.933	0.962	0.983	0.990

TABLE 2. Gauss-Seidel waveform relaxation averaged convergence factors for the two-dimensional heat equation.

4.2 MULTIGRID WAVEFORM RELAXATION

Theoretical results

The coarse grid Ω_H is derived from the fine grid Ω_h by standard coarsening ($H = 2h$). For the prolongation operator $p : \Omega_H \rightarrow \Omega_h$, we use piecewise linear interpolation. The restriction operator $r : \Omega_h \rightarrow \Omega_H$ is then defined as the transpose of the prolongation operator, $r = p^t$, see e.g. [1, p. 66] or [12, p. 70].

Lemma 5 *The two-grid operator $\mathcal{M}$ for the one-dimensional heat equation, discretized using linear basis functions, with red-black Gauss-Seidel smoothing, and with the prolongation and restriction operator defined as above, satisfies*

$$\rho(\mathcal{M}) \leq \sqrt{3}\sqrt{\eta_0(2\nu-1)} \ , \quad \textit{with} \ \ \eta_0(\nu) = \frac{\nu^\nu}{(\nu+1)^{\nu+1}} \ , \tag{34}$$

for $\nu = \nu_1 + \nu_2 \geq 1$.

The lemma states that the spectral radius of the two-grid operator $\mathcal{M}$ is bounded by a constant, independent of h. A table of the bound is given in Table 3. The proof is given in [2].

Numerical results

Since the bound in Lemma 5 is not optimal, we numerically computed the spectral radius of the two-grid operator by evaluation of (26), for $\nu = 2$ and for several values of h. These results are reported in the first line of Table 4, and are compared to the averaged convergence factors of different multigrid cycles. We use $V(\nu_1, \nu_2)$ and $W(\nu_1, \nu_2)$ as shorthands for a standard V-cycle and W-cycle with ν_1 presmoothing, ν_2 post-smoothing steps, and standard coarsening down to a grid with

ν	1	2	3	4
$\sqrt{3}\sqrt{\eta_0(2\nu-1)}$	0.866	0.563	0.448	0.384

TABLE 3. Values of the upper bound for $\rho(\mathcal{M})$.

mesh size $h = 0.5$. The other parameters are as above: red-black Gauss-Seidel smoothing, linear interpolation and corresponding restriction.

Finally, we report some averaged convergence factors for the two-dimensional model problem, discretized using bilinear basis functions on an equidistant rectangular grid.

h	1/8	1/16	1/32	1/64
$\rho_{\text{num}}(\mathcal{M})$	0.217	0.263	0.276	0.280
$V(1,1)$	0.228	0.301	0.325	0.332
$W(1,1)$	0.211	0.253	0.265	0.268

TABLE 4. Multigrid waveform relaxation averaged convergence factors for the one-dimensional heat equation.

h	1/4	1/8	1/16	1/32
$V(1,1)$	0.138	0.299	0.352	0.361
$W(1,1)$	0.138	0.294	0.343	0.355

TABLE 5. Multigrid waveform relaxation averaged convergence factors for the two-dimensional heat equation.

References

[1] W. Hackbusch. *Multi-Grid Methods and Applications.* Springer Verlag, Berlin, 1985.

[2] J. Janssen and S. Vandewalle. Multigrid waveform relaxation on spatial finite element meshes: the continuous-time case. Technical report TW 201, Department of Computing Science, Katholieke Universiteit Leuven, Belgium, November 1993.

[3] J. Janssen and S. Vandewalle. Multigrid waveform relaxation on spatial finite element meshes: the discrete-time case. Technical report TW, Department of Computing Science, Katholieke Universiteit Leuven, Belgium, 1994. (to appear).

[4] E. Lelarasmee, A. Ruehli, and A. Sangiovanni-Vincentelli. The waveform relaxation method for time-domain analysis of large scale integrated circuits. *IEEE Trans. Computer-Aided Design*, 1:131–145, 1982.

[5] Ch. Lubich and A. Ostermann. Multi-grid dynamic iteration for parabolic equations. *BIT*, 27:216–234, 1987.

[6] U. Miekkala. Dynamic iteration methods applied to linear DAE systems. *Journal of CAM*, 25:133–151, 1989.

[7] U. Miekkala and O. Nevanlinna. Convergence of dynamic iteration methods for initial value problems. *SIAM J. Sci. Stat. Comput.*, 8(4):459–482, July 1987.

[8] U. Miekkala and O. Nevanlinna. Sets of convergence and stability regions. *BIT*, 27:554–584, 1987.

[9] O. Nevanlinna. Remarks on Picard-Lindelöf iteration, PART I. *BIT*, 29:328–346, 1989.

[10] O. Nevanlinna. Remarks on Picard-Lindelöf iteration, PART II. *BIT*, 29:535–562, 1989.

[11] S. Vandewalle. *Parallel Multigrid Waveform Relaxation for Parabolic Problems.* B.G. Teubner Verlag, Stuttgart, 1993.

[12] P. Wesseling. *An Introduction to Multigrid Methods.* Wiley, Chicester, 1992.

8

A Subspace Decomposition Twogrid Method for Hyperbolic Equations

Edgar Katzer[1]

ABSTRACT A new twogrid iteration for hyperbolic equations is presented. It is based on a subspace decomposition technique and is a modification of the frequency decomposition approach [1].
We introduce a finite volume discretisation on a triangular grid which avoids the use of a secondary grid. The discrete equations are solved by a least square minimization procedure. The system is uniformly stable for all characteristic directions.
Numerical results for a hyperbolic model problem show good convergence rates for all characteristic directions. Thus an efficient and robust twogrid iteration is obtained.

1 Introduction

This paper presents a new twogrid iteration for the numerical solution of hyperbolic partial differential equations. Preliminary results for the extension to hyperbolic and elliptic systems are discussed.

The solution of first order systems of partial differential equations is of considerable interest. The Euler equations, which govern inviscid compressible flows, are such a system. For steady flows they are of a combined elliptic/hyperbolic type and this complicates their numerical solution considerably. The use of time marching methods for solving the steady state equations is common but not efficient. Low frequency errors are primarily convected through the computational domain and time step limitations leads to slow convergence. Even with multigrid acceleration techniques, convergence rates are slower than 0.9 (see e. g. [2], [3], [4]).

A direct approach for solving the steady equations is more complicated and is often restricted to first order accuracy. Excellent results were obtained with multigrid methods by [5],[6], ([7] for the Navier–Stokes equations), [8], [9], and [10]. The

[1]Institute for Analysis and Numerical Mathematics
Otto-von-Guericke-University, D-39016 Magdeburg, Germany

efficiency of the approach depends on carefully designed relaxation schemes and/or on multiple coarse grid corrections of the semi-coarsening approach [9]. Higher order accuracy is obtained with defect correction. Robustness of the convergence is difficult to obtain. In many cases, convergence rates depend on flow characteristics, e. g. the flow direction, or deteriorate for hypersonic flows; therefore a special treatment of these cases is necessary ([6]).

A new approach to robust multigrid methods, the frequency decomposition multigrid method, has been introduced by Hackbusch [11, 1] and independently by Ta'asan and Brandt (see [1]). The use of several coarse grid corrections is similar to the semi-coarsening approach of Mulder [9]. In section 3 we introduce a modification of the frequency decomposion approach. The name "subspace decomposition twogrid method" is proposed because the new prolongations defined in section 3 are not directly related to some parts of the frequency domain.

The development of a robust and efficient multigrid method is still an open problem and motivates the interest in hyperbolic problems. The present paper deals with twogrid methods for simple model problems and is a first step for the construction of efficient Euler solvers. It is not recommended to solve a *pure* hyperbolic problem with a multigrid method, because in this case alternative procedures like the method of characteristics are very efficient.

The hyperbolic model problem

As a hyperbolic problem, we define a scalar advection equation with constant coefficients:

$$a_1 \, \partial_x u + a_2 \, \partial_y u = 0 \quad , \qquad a_2 > 0 \quad , \tag{1}$$

and periodic initial and boundary conditions on the unit square:

$$u(x,0) = u_0(x) \quad ,$$
$$u(x+1,y) = u(x,y) \quad , \qquad \forall \;\; 0 \le y \le 1 \quad .$$

A conservative integral formulation is given by Stoke's theorem:

$$\int_{\partial\tau} a_1 \, u \, dy - a_2 \, u \, dx = 0 \quad , \tag{2}$$

for sufficiently smooth control elements τ with boundary $\partial\tau$.

In section 2 a finite volume discretisation on a triangular grid and a minimization problem are introduced. In section 3 we introduce a subspace decomposition approach. Numerical results for the hyperbolic model problem and preliminary results for first order systems are shown in section 4.

2 Discretisation

The integral formulation (2) is approximated on a structured triangular grid based on an equidistant grid with step size $h = 1/n'$:

$$\Omega_h = \{(x_j, y_k) \,|\, x_j = jh \,, \; y_k = kh \,, \; 1 \le j,k \le n'\} \quad .$$

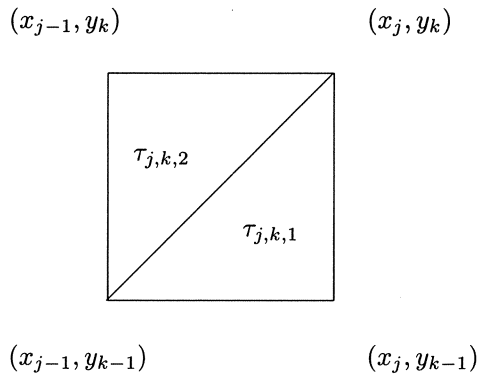


FIGURE 1. Subdivision of a square into two triangles.

Figure 1 shows the subdivision of a square into two triangles. The grid is equivalent to the regular triangular grid shown in figure 2.

Usually, finite volume methods depend on a second grid of control volumes defined e. g. by the centroids of the triangles (see e. g. [12]). The present vertex centered flux discretisation avoids such a dual grid. Fluxes along triangle edges are approximated by the trapezoidal rule and yield for a triangle of type $\tau_{j,k,1}$:

$$Lu\,(\tau_{j,k,1}) = \frac{1}{h}\,(a_2\,u_{j,k} - (a_2 - a_1)u_{j,k-1} - a_1\,u_{j-1,k-1}) \tag{3}$$

and for a triangle of type $\tau_{j,k,2}$:

$$Lu\,(\tau_{j,k,2}) = \frac{1}{h}\,(a_1\,u_{j,k} + (a_2 - a_1)u_{j-1,k} - a_2\,u_{j-1,k-1}) \quad . \tag{4}$$

As there are twice as many triangles $\tau \in T$ than grid points involved, the dimension of the flux space $F = \{f : T \to \mathbf{R}\}$ is larger than the dimension of the grid space $U = \{u : \Omega_h \to \mathbf{R}\}$. The discrete system

$$Lu = f \quad , \qquad L : U \to F$$

is overspecified and no solution of the flux equations exists in the general case. Therefore a discrete minimization problem is defined:

Minimization problem

For L given by (3) and (4) we minimize the squared Euclidean norm

$$E(u) = \|Lu - f\|_2^2 \tag{5}$$

and solve:

$$E(u^*) \leq E(u) \quad , \qquad \forall\, u \in U \quad . \tag{6}$$

A minimum exists and is unique, as the author [13] has shown. The well known solution of this problem is given in lemma 1.

Lemma 1:
The minimum u^* of problem (6) solves the normal equations:

$$Au = L^*Lu = L^*f = b \quad . \tag{7}$$

On a cartesian grid, matrix A is represented by the seven-point stencil:

$$A = \frac{1}{h^2} \begin{bmatrix} 0 & -g_2 & -g_3 \\ -g_1 & 2(g_1+g_2+g_3) & -g_1 \\ -g_3 & -g_2 & 0 \end{bmatrix} \quad , \tag{8}$$

where

$$\begin{aligned} g_1 = 2a_1(a_1-a_2) &= r^2(1-\sin(2\phi)+\cos(2\phi)) \\ g_2 = 2a_2(a_2-a_1) &= r^2(1-\sin(2\phi)-\cos(2\phi)) \\ g_3 = 2a_1a_2 &= r^2\sin(2\phi) \end{aligned}$$

and the flow in polar coordinates is given by $a_1 = r\sin(\phi)$, $a_2 = r\cos(\phi)$.

Matrix A is positive definite but not an M-matrix, except in special cases when two of the $g_1 \ldots g_3$ vanish.

Proof:
Simple calculation yields (8). As $g_1g_2g_3 = -8a_1^2a_2^2(a_1-a_2)^2 \leq 0$ and $g_1+g_2+g_3 = r^2(2-\sin(2\phi)) \geq r^2$, not all coefficients are simultaniously positive and A is not an M-matrix. ■

Discretisations of hyperbolic equations may be unstable when the CFL condition is not fulfilled. For the minimization problem uniform stability for all characteristic directions is obtained in

Lemma 2:
Let u^* be the solution of the minimization problem (6), u_0 be the discrete inflow profile at $y = 0$ and let $f = 0$ at inner points. Then we have uniform stability for all a_1, a_2 and h in the discrete L_2–norm on Ω_h and Γ_h :

$$||u^*||_{\Omega,h} \leq ||u_0||_{\Gamma,h}$$

A proof based on a Fourier analysis is given in [13], corollary A.9. ■

Furthermore the discrete solution is second order accurate on an equidistant grid (see [13]). This is rather surprising, because we do not fulfill the flux equations (3, 4) exactly but only in the mean. Apparently, the minimization procedure does not deteriorate the accuracy.

3 Subspace decomposition twogrid method

We present a new twogrid method based on a subspace decomposition technique and introduce a decomposition of the grid space

$$U = \sum_{\kappa=1}^{K} U_\kappa \quad . \tag{9}$$

On each subspace U_κ smaller minimization problems are given by the

Coarse grid correction:
For given $u \in U$ and subspace U_κ, $1 \leq \kappa \leq K$, minimize:

$$E(u + v_\kappa) \leq E(u + v) \quad \forall\, v \in U_\kappa \quad . \tag{10}$$

This defines the coarse grid correction $G_\kappa(u) = u + v_\kappa$.

The subspaces are defined by prolongations p_κ on a standard coarse grid space V:

$$p_\kappa : V \to U \quad , \quad U_\kappa = \text{Range}\,(p_\kappa) \quad .$$

The coarse grid operator is then given by:

$$G_\kappa(u) = M_\kappa u + N_\kappa b$$

where

$$N_\kappa = p_\kappa (p_\kappa^* A p_\kappa)^{-1} p_\kappa^* \quad , \quad M_\kappa = I - N_\kappa A \quad .$$

A simple gradient iteration is applied for **smoothing:**

$$S(u) = u - \lambda(Au - b) \quad , \quad \lambda = <r, r> / <Ar, r> \quad .$$

The prolongations
The prolongations are given in stencil notation:

$$p_0 = \tfrac{1}{2}\begin{bmatrix} 0 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 0 \end{bmatrix} \quad , \quad p_1 = \tfrac{1}{2}\begin{bmatrix} 0 & 1 & -1 \\ -1 & 2 & -1 \\ -1 & 1 & 0 \end{bmatrix}$$

$$p_2 = \tfrac{1}{2}\begin{bmatrix} 0 & -1 & -1 \\ 1 & 2 & 1 \\ -1 & -1 & 0 \end{bmatrix} \quad , \quad p_3 = \tfrac{1}{2}\begin{bmatrix} 0 & -1 & 1 \\ -1 & 2 & -1 \\ 1 & -1 & 0 \end{bmatrix} \quad .$$

Here, p_0 is the well known seven-point prolongation. The other prolongations are defined on shifted coarse grids. As the stencil notation gives no information on

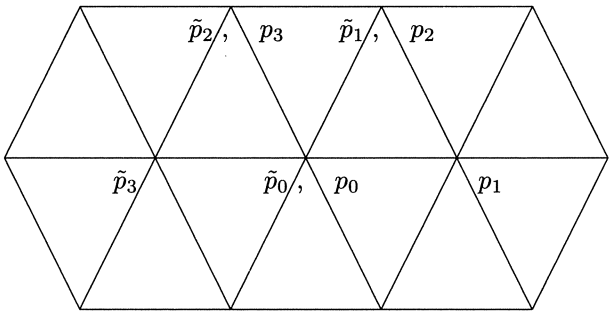


FIGURE 2. Location of coarse grid prolongations.

the location of the coarse grid, figure 2 shows the location of the center of the prolongations in the triangular grid.

Obviously, the seven-point prolongations are more natural than nine-point prolongations on a structured triangular grid. Numerical tests showed, that these four prolongations are not sufficient for a robust method. It was necessary to introduce four additional prolongations $\tilde{p}_0 \ldots \tilde{p}_3$ similar to $p_0 \ldots p_3$ which are located at shifted coarse grids given in figure 2 (notice that $\tilde{p}_0 = p_0$).

Twogrid iteration

At the moment we apply a multiplicative Schwarz method combined with smoothing steps:

$$\Phi^{TG} = \tilde{\Phi}_1 \Phi_1 \tag{11}$$

where

$$\Phi_1 = G_3 S G_2 S G_1 S G_0 S$$
$$\tilde{\Phi}_1 = \tilde{G}_3 S \tilde{G}_2 S \tilde{G}_1 S \tilde{G}_0 S$$

In section 4 we also present convergence rates for the sequence of prolongations given by the frequency decompositon approach ([11, 1]):

$$\Phi^{HA} = G_3^{HA} S G_2^{HA} S G_1^{HA} S G_0^{HA} S \quad . \tag{12}$$

Furthermore results for an iteration without smoothing is presented:

$$\Phi^{nos} = \tilde{G}_3 \tilde{G}_2 \tilde{G}_1 \tilde{G}_0 G_3 G_2 G_1 G_0 \quad . \tag{13}$$

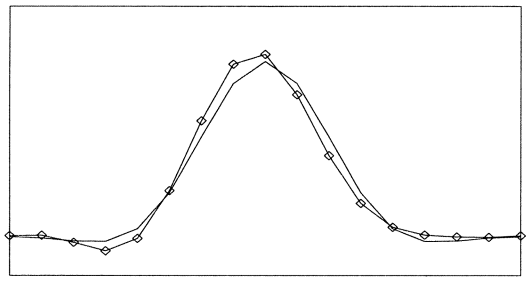


FIGURE 3. Minimization (—) and Lax–Wendroff (◇) solution.

4 Numerical results

The accuracy of solutions of the minimization problem (6) and the efficiency of the twogrid iteration are analysed in this section. For two inflow profiles the discrete solution at the outflow boundary, $y = 1$, is compared with the continuous solution and the second order accurate Lax–Wendroff solution. The Lax–Wendroff solution is obtained by marching in y-direction. In both cases the parameters are: $a_1 = 0.5$, $a_2 = 1.0$.

Figure 3 compares the minimization solution with the Lax–Wendroff solution for a continuous cosine-type profile. A rather coarse grid of 17×9 makes small differences clearly visible. Both numerical solutions are close together which demonstrates the second order accuracy of the minimisation solution.

Figure 4 compares the minimization solution with the exact solution of problem (1) for a discontinuous inflow profile. At the outflow boundary the discontinuities are diffused over several points of the 65×65 grid. Small overshoots of about 5% are observed. The isolines in figure 5 confirm the small smearing of the numerical solution.

Convergence rates for twogrid iterations are shown in table 1 to 4. Unless otherwise stated, all results are asymptotic error reduction rates obtained on an 32×32 grid. The influence of the characteristic direction is represented in the parameter $q = a_1/a_2$.

Table 1 shows convergence rates for different twogrid operator sequences and flow directions. The convergence rates for Φ_1 and $\tilde{\Phi}_1$, deteriorate for some parameters. It is the combination of both, Φ^{TG}, which is robust. For the prolongations given by Hackbusch [1], we obtain very good error reduction for $q = 0$, but almost no reduction for $q = 1$. The hyperbolic problem with $q = 1$ is not in the robustness class of the frequency decomposition method. The results for the case without any smoothing Φ^{nos} shows, that smoothing could be omitted on the cost of increased convergence rates.

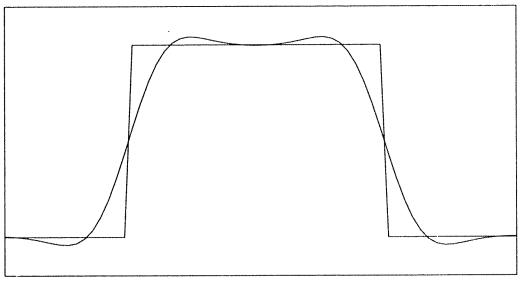


FIGURE 4. Minimization solution for a discontinuous profile.

q	-1.0	0.0	0.5	0.75	1.0	2.0
Φ^{HA}	0.38	0.03	0.78	0.92	0.97	0.83
Φ_1	0.76	0.50	0.71	0.72	0.89	0.74
$\tilde{\Phi}_1$	0.71	0.90	0.70	0.71	0.50	0.75
Φ^{TG}	0.55	0.44	0.50	0.52	0.44	0.56
Φ^{nos}	0.64	0.47	0.67	0.65	0.47	0.64

TABLE 1. Convergence rates for different corse grid sequences.

q	-4.0	-2.0	-1.0	-0.5	0.0	0.25
$\rho(\Phi^{TG})$	0.64	0.54	0.55	0.53	0.44	0.50
q	0.50	0.75	1.0	1.5	2.0	4.0
$\rho(\Phi^{TG})$	0.50	0.52	0.44	0.54	0.56	0.59

TABLE 2. Influence of flow direction, $q = a_1/a_2$ (robustness).

The robustness of the twogrid iteraton is analysed in table 2. The characteristic direction q has only minor influence on the convergence.

First, and at the moment preliminary, results for hyperbolic and elliptic systems are shown in tables 3 and 4. Here we solve a 2×2 system:

$$A_1 \, \partial_x u + A_2 \, \partial_y u = 0$$

where $A_2 = I$ and, in the hyperbolic case:

$$A_1 = q \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

and in the elliptic case ($q \neq 0$):

$$A_1 = q \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \quad .$$

q	0.5	1.0	2.0
$\rho(\Phi^{TG})$	0.55	0.55	0.56

TABLE 3. Robustness for a hyperbolic system.

q	1.0	0.5	0.25	0.1
$\rho(\Phi^{ell})$	0.12	0.14	0.21	0.34
$\rho(\Phi_2)$	0.20	0.38	0.75	0.94

TABLE 4. Robustness for an elliptic system.

For $q = 1$ we obtain the Cauchy–Riemann equations. For the hyperbolic system we specify periodic Dirichlet conditions at $y = 0$. For the elliptic case we specify periodic conditions at all boundaries and impose zero mean values.

Table 3 shows convergence rates for the hyperbolic system. They are independent of the parameter q and of similar order as in the scalar case. The results for the elliptic system are obtained with four and one coarse grid corrections, respectively:

$$\begin{aligned} \Phi^{ell} &= G_3 S G_2 S G_1 S G_0 S \\ \Phi_2 &= G_0 S^4 \end{aligned}$$

Results for Φ^{ell} in table 4 show very small convergence rates which are independent of q. We obtain robustness similar as in the scalar case. The iteration Φ_2 is similar to usual multigrid procedures with smoothing and only one coarse grid correction. Convergence rates are small for $q = 1$ but deteriorate for $q \to 0$. Notice that the minimization procedure solves the normal equations ($A = L^*L$). Brandt and Dinar [14] successfully used the adjoint operator L^* for the construction of a distributed smoother for solving the Cauchy–Riemann equations. It is then no surprise, that Φ_2 shows fast convergence for $q = 1$ but is not robust because here a simple smoother is used.

5 Conclusion

An efficient twogrid iteration for hyperbolic equations is presented. The discrete system is based on a new vertex centered finite volume discretisation on a triangular grid. The solution of a minimization problem for the flux equations has several advantages. The minimum is unique and second order accurate on an equidistant grid. The system of normal equations is positive definite and uniformly stable for all characteristic directions and mesh sizes.

Discontinuous solutions are captured without additional numerical diffusion. They are smeared over several grid spacings and only small overshoots are obsered.

FIGURE 5. Isolines for a discontinuous minimization solution.

The discrete minimization problem is solved by a twogrid iteration based on a subspace decompositon technique. These subspaces are created by carefully designed prolongations on a coarse grid. The minimization problems on the subspaces yield the coarse grid corrections. A multiplicative Schwarz iteration with additional smoothing steps defines the twogrid iteration. The present approach is a modification of the frequency decomposition approach of Hackbusch [1, 11].

We obtain a robust twogrid iteration with uniformly bounded convergence rates for all characteristic directions. Robustness is important for the extension of the algorithm to solve hyperbolic or elliptic systems. Primary results for linear systems are promising.

It is expected, that the present approach can be extended to nonlinear systems of first order equations. The robustness and uniform stability is a prerequisite for future applications to Euler and possibly the Navier–Stokes equations. A parallel version based on additive Schwarz iteration is currently being developed (see [15]).

Acknowledgements

Many fruitful and inspiring discussions with Prof. Hackbusch, Kiel, are gratefully acknowledged. The present work was supported by the "Deutsche Forschungsgemeinschaft" (German Research Council).

References

[1] Hackbusch, W.: *The frequency decomposition multigrid method.* Fourth European Multigrid Conference, EMG '93, Amsterdam, 1993.

[2] Jameson, A.: *Solution of the Euler equations for two dimensional transonic flow by a multigrid method.* Applied Math. Computation 13 (1983), pp. 327–356.

[3] de Cock, K.: *Multigrid acceleration of the 2D Euler equations applied to high lift systems.* Fourth European Multigrid Conference, EMG '93, Amsterdam, 1993.

[4] Kloppmann, Ch.; Schwamborn, D.; Singh, J., P.: *Multigrid solution of the 2D Navier–Stokes equations for transonic internal and external flows.* Fourth European Multigrid Conference, EMG '93, Amsterdam, 1993.

[5] Hemker, P., W.; Spekreijse, S., P.: *Multiple grid and Osher's scheme for the efficient solution of the Euler equations.* Applied Numer. Math. 2 (1986), pp. 475–493.

[6] Koren, P., W.; Hemker, P. W.: *Damped, direction-dependent multigrid for hypersonic flow computation.* Applied Numerical Mathematics 7 (1991), pp. 309–328.

[7] Shaw, G.; Wesseling, P.: *A multigrid method for the Navier–Stokes equations.* Deft University of Technology, Dept. Mathematics and Informatics. Report No.: 86–13, 1986.

[8] Dick, E.: *Multigrid formulation of polynomial flux-difference splitting for steady Euler equations.* J. Computational Physics 91 (1990), pp. 161–173.

[9] Mulder, W., A.: *A high resolution Euler solver based on multigrid, semi-coarsening, and defect correction.* J. Computational Physics 100 (1992), pp. 91–104.

[10] Ta'asan, S.: *Optimal multigrid solvers for steady state inviscid flow problems.* Fourth European Multigrid Conference, EMG '93, Amsterdam, 1993.

[11] Hackbusch, W.: *A new approach to robust multi-grid solvers.* First. Int. Conf. on Industrial and Applied Mathematics. ICIAM '87. Proceedings, Philadelphia, PA: SIAM 1988, pp. 114–126.

[12] Löhner, R.; Morgan, K.; Zienkiewicz, O., C.: *The solution of non–linear hyperbolic equation systems by the finite element method.* Int. J. Numer. Meth. Fluids 4 (1984), pp. 1043–1063.

[13] Katzer, E.: *A subspace decomposition twogrid method for hyperbolic equations.* Inst. für Informatik und Praktische Mathematik, Kiel: Universität Kiel, 1992, Bericht Nr. 9218.

[14] Brandt, A.; Dinar, N.: *Multigrid solutions to elliptic flow problems.* In: Parter, S.V. (ed.): Numerical Methods for Partial Differential Equations. New York: Academic Press, 1979, pp. 53–147.

[15] Katzer, E.: *A parallel subspace decomposition method for hyperbolic equations.* 7^{th} Int. Conf. on Domain Decomposition Methods, Pennsylvania State College, 1993. Preprint MBI-93-4, 1993. Graduiertenkolleg Modellierung, Berechnung und Identifikation mechanischer Systeme. Otto-von-Guericke-Universität, Magdeburg.

9

Multigrid Solution of the 2-D Navier-Stokes Equations for Transonic Internal and External Flows

Ch. Kloppmann[1], D. Schwamborn[2] and J.P. Singh[3]

ABSTRACT A FAS multigrid scheme is presented to accelerate the solution process of the two-dimensional, turbulent, compressible Navier-Stokes equations. Simulations of transonic internal and external flows are carried out. Multigrid variations like V- and W- cycling, FMG, different time step numbers and grid levels are tested and their advantages are shown. In most cases, machine accuracy can be reached without difficulty. The results obtained are in good agreement with the available experimental data.

1 Introduction

An enormous amount of computational time is required for the numerical solution of the Navier-Stokes equations, particulary in case of threedimensional (3-d) problems. Therefore, it is important to find adequate techniques for the acceleration of the solution process. Among various techniques, the meanwhile widespread multigrid schemes belong to the most effective techniques to proceed the solution to a steady state.
The present work is based on a 3-d compressibe, Reynolds averaged Navier-Stokes solver developed by Schwamborn [3] with explicit Runge-Kutta time stepping, finite volume discretization and a cell centered treatment of the flow variables and is along the ideas of Jameson [4]. It is the aim here to reduce the computational time by implementation of the nonlinear Full-Approximation-Scheme (FAS), which is a suitable multigrid technique for hyperbolic equations. At this stage, the twodimensional formulation of the problem is treated first. The main subject is to obtain

[1]DLR- Institute for Theoretical Fluid Dynamics, Bunsenstr. 10, D-3400 Göttingen
[2]DLR- Institute for Theoretical Fluid Dynamics, Bunsenstr. 10, D-3400 Göttingen
[3]CTFD, National Aeronautical Lab., Bangalore 560017, India

accurate solutions and to achieve a faster convergence.

2 Governing Equations and Spatial Discretization

The solver is based on the time-dependent, compressible Navier-Stokes equations in conservative formulation

$$\frac{\partial}{\partial t}\int\int\int_{Vol} \overline{\mathbf{U}}\, dVol + \int\int_{S} \overline{\overline{\mathbf{H}}} \cdot \overline{n}\, dS = 0,$$

where the solution vector $\overline{\mathbf{U}}$ consists of mass, momentum and total energy $\overline{\mathbf{U}} = (\rho, \rho u, \rho w, \rho E)^T$. The total energy E per unit mass $E = e + (u^2 + w^2)/2$ contains the mass averaged internal energy

$$e = \frac{p}{\rho\,(\gamma - 1)}$$

whith the specific heat ratio γ and the pressure p. The flux tensor $\overline{\overline{\mathbf{H}}}$ writes

$$\overline{\overline{\mathbf{H}}} = \begin{pmatrix} \rho u & \rho w \\ \rho u^2 - \sigma_x & \rho w u - \tau_{zx} \\ \rho u w - \tau_{xz} & \rho w^2 - \sigma_z \\ (\rho E - \sigma_x)\, u - \tau_{xz} w + q_x & (\rho E - \sigma_z) w - \tau_{zx} u + q_z \end{pmatrix}$$

wherein σ, τ are the stress tensor components and q_x, q_z denote the heat flux vector with the thermal conductivity k

$$q_x = -k\, T_x \qquad q_z = -k\, T_z.$$

The surface S of a volume Vol owns the normal vector $\overline{n}$. Sutherland's equation is used to calculate the molecular viscosity and heat conductivity while the eddy viscosity coefficient is computed with the Baldwin-Lomax turbulence model. During the multigrid procedure, the eddy viscosity is computed and updated only on the finest grid level. The physical domain is discretizated by a structured mesh which forms quadrilateral cells. Flow variables are located in the cell centers and are assumed to be constant over the entire cell volume. For one cell, the spatial discretization can be formulated as

$$\frac{d}{dt}\,\overline{\mathbf{U}}_{i,k} + (Q^{conv}_{i,k} + Q^{visc}_{i,k} - D_{i,k})\,\overline{\mathbf{U}}_{i,k} = 0.$$

Herein, Q^{conv}, Q^{visc} are the convective and viscous flux operators and D is the artificial dissipation operator. The convective fluxes are obtained by averaging the variables at each side of the cell faces equivalent to central differencing, while the derivatives in the viscous fluxes are obtained from a local coordinate transformation [3]. The numerical dissipation is required to stabilize the solution at shocks,

stagnation points or in case of small physical diffusion. Its damping terms consist of second and fourth order differences following Jameson et. al. [4] and are written as

$$D(\overline{U}) = (D_\xi^2 - D_\xi^4 + D_\eta^2 - D_\eta^4)\,\overline{U}$$

where the ξ- Operators are

$$D_\xi^2 \overline{U} = \nabla_\xi(\Lambda_{i+\frac{1}{2},k}\,\varepsilon^{(2)}_{i+\frac{1}{2},k})\Delta_\xi \overline{U}_{i,k}$$

$$D_\xi^4 \overline{U} = \nabla_\xi(\Lambda_{i+\frac{1}{2},k}\,\varepsilon^{(4)}_{i+\frac{1}{2},k})\Delta_\xi \nabla_\xi \Delta_\xi \overline{U}_{i,k}$$

and i,k the indices with ξ, η - directions and ∇_ξ, Δ_ξ are forward and backward difference operators in ξ.
To avoid large unphysical diffusion in viscous wall regions, we use the eigenvalue scaling of Martinelli [1] and Arnone [8]. The coefficients Λ are defined by

$$\Lambda_{i+\frac{1}{2},k} = \frac{1}{2}\left((\Lambda_\xi)_{i,k} + (\Lambda_\xi)_{i+1,k}\right)$$

with

$$\Lambda_\xi = \Phi_\xi \lambda_\xi \qquad and \qquad \Phi_\xi = 1 + \left(\frac{\lambda_\eta}{\lambda_\xi}\right)^\sigma$$

where $\lambda_\xi, \lambda_\eta$ are the scaled spectral radii of the Jacobian of the convective fluxes. The expression for the second component Λ_η is equivalent. In our calculations, the exponent σ is set to $\sigma = .67$. The coefficients $\varepsilon^{(2)}$ are used as a sensor for shocks and they are evaluated with respect to the second differences of the local pressure, while the coefficients $\varepsilon^{(4)}$ serve for the background dissipation and are switched off near shocks.

3 Runge- Kutta Time Stepping

The time integration of the discretizated equations is performed with a Runge Kutta scheme with local time steps Δt as in the work of Jameson [4]. In the present calculations we use the 5-stage hybrid scheme

$$\begin{aligned} U^{(0)} &= \overline{U}^n \\ U^{(\mathbf{m})} &= U^{(0)} - \alpha_{\mathbf{m}}\,\Delta t\,[Q^{conv}(U^{(\mathbf{m-1})}) + Q^{visc}(U^{(0)}) - D_{\mathbf{m-1}}] \\ &\quad with\ \mathbf{m} = 1, 2, ..., 5 \\ \overline{U}^{n+1} &= U^{(5)}. \end{aligned}$$

The solution $\overline{U}^n$ at a certain timestep is denoted by n, $U^{(\mathbf{m})}$ is the data at stage number $\mathbf{m}$. The viscous terms are held constant after the first stage and the dissipative operator D is recomputed only at the first, third and fifth stage employing a combination of actual and previous values:

$$D_0 = D_1 = D(U^{(0)})$$

$$D_2 = D_3 = \beta D(U^{(2)}) + (1-\beta)D_0$$
$$D_4 = \gamma D(U^{(4)}) + (1-\gamma)D_2.$$

Following Mavripilis et.al. [2], this scheme has good stability characteristics. The weighting factors of the dissipation are $\beta = .56$ and $\gamma = .44$. The coefficients for the different timestep stages are $\alpha_1 = 1/4$, $\alpha_2 = 1/6$, $\alpha_3 = 3/8$, $\alpha_4 = 1/2$ and $\alpha_5 = 1$.

4 Multigrid Method

The application of a multigrid technique accelerates the solution process on the way to the steady state. Low frequency error components of an estimated solution are damped out slowly on fine grids. On coarsed grids, this error appears to have a short wavelength and is reduced faster since the Runge-Kutta solver smoothes out the high frequency error rapidly. We use the FAS scheme and work on three or four grid levels. This is in accordance to Jameson and Baker [5]. Every grid is coarsened by ommitting every second gridpoint in both directions of the mesh. Until the coarsest mesh level is reached, the flow variables are restricted by a volume (Vol) averaging

$$\overline{U}_H = \frac{\sum \overline{U}_h \, Vol_h}{\sum Vol_h}$$

where $\overline{U}_h, \overline{U}_H$ are the solution vectors on the finer and coarser mesh, respectively, and the sum is taken over those four neighbouring fine grid cells which build one coarse grid cell. This procedure conserves mass, momentum and energy. The residuals R_h from the finer grid are summarized over the four cells to find the forcing function τ_H on the coarser grid

$$\tau_H = \sum R_h - R_H^{(0)},$$

where $R_H^{(0)}$ is the coarse grid residual from the first Runge-Kutta stage

$$R_H^{(0)} = [Q^{conv}(U^{(0)}) + Q^{visc}(U^{(0)}) - D_0].$$

The coarse grid stage $\mathbf{m}$ writes

$$U_H^{(\mathbf{m})} = U_H^{(0)} - \alpha_{\mathbf{m}} \, \Delta t \, [Q^{conv}(U_H^{(\mathbf{m-1})}) + Q^{visc}(U_H^{(0)}) - D_{\mathbf{m-1}} + \tau_H]$$

and its first stage result is only influenced by the fine grid residual as is easily seen by taking $\mathbf{m} = 1$ and substituting τ_H. The correction $(\overline{U}_H^{new} - \overline{U}_H^{old})$ found on a coarser grid from the difference of the computed and previously restricted solution vector is prolongated to the next finer mesh level by linear interpolation. To overcome high frequency errors, this prolongated correction for each fine grid

cell is smoothed using a weighted average of the corrections in the surrounding eight cells. Denoting the prolongation operator with I_H^h, the solution on finer grid levels is updated by

$$\overline{U}_h^{new} = \overline{U}_h^{old} + I_H^h (\overline{U}_H^{new} - \overline{U}_H^{old}).$$

5 Boundary Conditions

The boundary conditions are no-slip at walls, which are assumed to be adiabatic while the pressure can be found from the assumption of zero pressure gradient normal to the wall. The latter is justified in turbulent Navier-Stokes calculations due to the very small step size normal to the wall. At the farfield boundary, one-dimensional Riemann invariants are used. For the tunnel cases, constant total pressure and isentropic flow are assumed at the inflow plane while the inflow velocity is extrapolated prescribing the flow direction. In the exit plane, the pressure p_{out} is known and a non-reflecting boundary condition [7] $p_t = \rho c u_t + \beta (p - p_{out})$ is used where p_t, u_t indicate partial differentation with respect to time and c is the speed of sound. The formulation allows pressure waves to leave the computational domain, but ensures the prescribed pressure in case of convergence for nonzero β. During the multigrid process,the boundary conditions are updated at every Runge Kutta stage as well as after the restriction to a coarser grid and again after the solution updating with the coarse grid correction on a finer grid level.

6 Implicit Residual Smoothing

To improve the convergence of the scheme, an implicit residual averaging is introduced. Instead of the unsmoothed local residual R, the smoothed residual $\overline{R}$ is computed from

$$(1 - \beta_\xi \nabla_\xi \Delta_\xi)(1 - \beta_\eta \nabla_\eta \Delta_\eta)\, \overline{R} = R$$

wherein only an inversion of tridiagonal matrices is required. The coefficients β_ξ, β_η depend on the cell aspect ratio and are defined as

$$\beta_\xi = max\left(0, \frac{1}{4}\left[\left(\frac{CFL}{CFL^*}\,\frac{\lambda_\xi}{\lambda_\xi + \lambda_\eta}\,\Phi_\xi\right)^2 - 1\right]\right)$$

$$\beta_\eta = max\left(0, \frac{1}{4}\left[\left(\frac{CFL}{CFL^*}\,\frac{\lambda_\eta}{\lambda_\xi + \lambda_\eta}\,\Phi_\eta\right)^2 - 1\right]\right).$$

This improves the robustness and the stability limit for viscous calculations on streched meshes and is in accordance with the smoothing method described by Martinelli [1] and Arnone et. al. [8].

7 Results

Some typical results concerning the effectivity of the multigrid scheme and the agreement with experimental data are shown.
At first, the computation of the flow past the CAST 7 airfoil at $\alpha = 2$ degrees incidence, $Ma_{inf} = .7$, $Re = 4 \cdot 10^6$ and transition fixed at 7 percent cord length is presented. The mesh consists of 257 x 41 points and is of C- type. A multigrid computation is performed using V-cycles on three grid levels. Mach isolines ($\Delta Ma = .0325$) of the solution are shown in Figure 1. They are in agreement with the single grid computation. The computed lift and drag coefficients are $c_L = .817$, $c_D = .0192$. In Figure 2, the convergence history of this run is given. The maximal density residual is plotted as a function of work units. A work unit is the amount of computational work during a Runge-Kutta time step on the finest grid. Time steps on coarser grids need a fraction of a work unit. A steady state solution is achieved within the range of machine accuracy. This multigrid computation converges about 14 times faster than the single grid computation. To demonstrate the influence of different grid levels on convergence speed, the performance ist plotted in Figure 2. A run on two grid levels only is less effective. If the grid depth is extended to four, an improvement of convergence speed is achieved. A full multigrid (FMG) computation starting on the coarsest level was found to reduce the computation time to 60 percent (Figure 3).
The convergence history is dependent on the number of timesteps on different grid levels. If one single step is applied on each level, no converged solution is achieved. At least, two Runge-Kutta steps are needed on the finest grid level (graph 2-1-1 in Figure 3) or alternatively some steps on the coarsed grids (graph 1-16-32 for example). Better results were obtained with a few steps on each level as in the case of 2-16-32 steps from fine to coarser grids or with 2-1-16 steps. It is found that 2 Runge-Kutta steps on the finest grid are optimal but this is not demonstrated here. The application of one timestep during the prolongation phase could improve the convergence rate in any case.
The next example is the RAE 2822 airfoil at $\alpha = 2.54$ degrees incidence, $Ma_{inf} = .734$, $Re = 6.5 \cdot 10^6$ and a transition fix at 3 percent. The C- type grid has 257 x 65 nodes. Again, the multigrid procedure uses V-cycles on three grid levels. The surface pressure distribution is compared with experimental data [6] indicated by triangles in Figure 4. Differences are at least partly attributed to the chamber correction of the airfoil which was used in the mesh generation to compensate for windtunnel effects over a range of Mach numbers and incidences. Lift and drag coefficients are computed to $c_L = .759$ and $c_D = .0205$. Figure 5 shows the Mach number contour lines ($\Delta Ma = .0325$). The residual history in Figure 6 indicates the fast residual drop versus the single grid computation. All multigrid computations are performed with implicit residual smoothing during the Runge Kutta stages and we use a 2.6 times larger CFL- number than single grid calculations allow. To find the speedup of this treatment, a multigrid run is made for the RAE 2822 airfoil applying the single grid CFL number. Figure 6 gives the convergence

development which is 1.5 times slower compared to the smoothed case. Another interesting variation of multigrid stepping is the W- cycle type, which has a small advantage against V- cycling (Figure 7).
An internal flow case is presented with the ONERA Bump A [9] testcase. The exit pressure is fixed to .658 times the total pressure in the reservoir which corresponds to an isentropic Mach number of 1.05. The mesh has 193 x 65 nodes on the finest grid. In Figure 8, the mesh and the Mach isolines ($\Delta Ma = .035$) are plotted to show the resulting flow field. Implicit residual smoothing, three grid levels and V-cycling were applied for the multigrid computation. The convergence rate is about 6 times greater than the single grid computation (Figure 9), but the solution does not drop to the same accuracy as the single grid computation. This seems to result from the outflow boundary condition. The agreement between experimental and numerical isentropic Mach number along the wall and along the centerline are fairly well (Figure 10). Here, the x- coordinate is made dimensionless with two times the bump length L.

8 Conclusions

An explicit 5-stage finite volume Runge-Kutta scheme has been presented. A multigrid algorithm was implemented which accelerates the convergence to a steady state in an effective way. A number of aspects of the method has been examined and discussed. The computed flows are in good agreement with the single grid results and with the available experimental data thus the multigrid method is expected to become a helpful tool for a variety of transonic flows.

References

[1] Martinelli,L., Jameson, A.; Validation of a Multigrid Method for the Reynolds Averaged Equations, AIAA Paper 88-0414 (1988)

[2] Mavripilis, D.J., Jameson, A., Martinelli, L.; Multigrid Solution of the Navier-Stokes Equations on Triangular Meshes, AIAA Paper 89-0120 (1989)

[3] Schwamborn, D.; Simulation of the DFVLR-F5 Wing Experiment Using a Block Structured Explicit Navier-Stokes Method., Vieweg Series, Notes on Numerical Fluid Dynamics, Vol.22, pp.244-268, (1988)

[4] Jameson, A., Schmidt, W., Turkel, E.; Numerical Solution of the Euler Equations by Finite Volume Methodes Using Runge-Kutta Time-Stepping Schemes, AIAA Paper 81-1259, (1981)

[5] Jameson, A. , Baker, T.J.; Multigrid Solution of the Euler Equations for Aircraft Configurations, AIAA Paper 84-0093 (1984)

[6] Cook,P.H., McDonald, M.A. and Firmin, M.C.P.; Aerofoil RAE2822 - Pressure Distributions and Boundary Layer and Wake Measurements, AGARD-AR-138, (1979)

[7] Rudi, D.M., Strickwerda. J.C.; A Nonreflecting Outflow Boundary Condition for Subsonic Navier-Stokes Calculations, Comp. Physics 36, (1980), pp. 55-70

[8] Arnone, A., Liou, M.S. and Povinelli, L.A.; Multigrid Calculations of Thredd-Dimensional Viscous Cascade Flows; NASA TM 105257 (1991)

[9] Délery, J.; Experimental Investigation of Turbulence Properties in Transonic Shock/Boundary Layer Interactions, AIAA J.Vol.21, No.2 pp. 180/185 (1983)

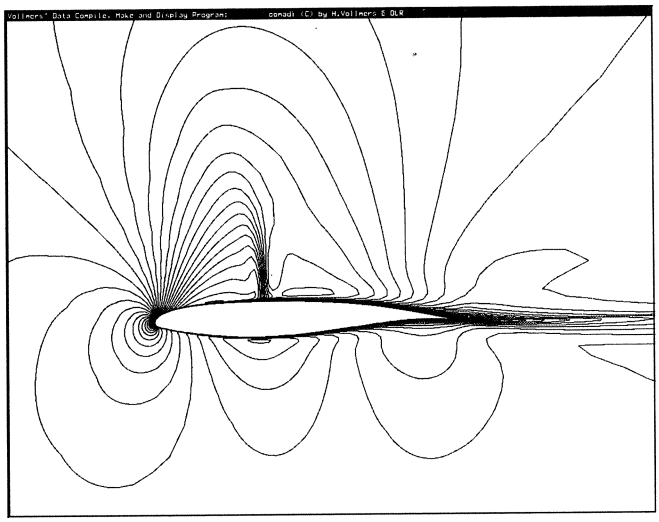


FIGURE 1. Cast 7: Contour plot of Mach number

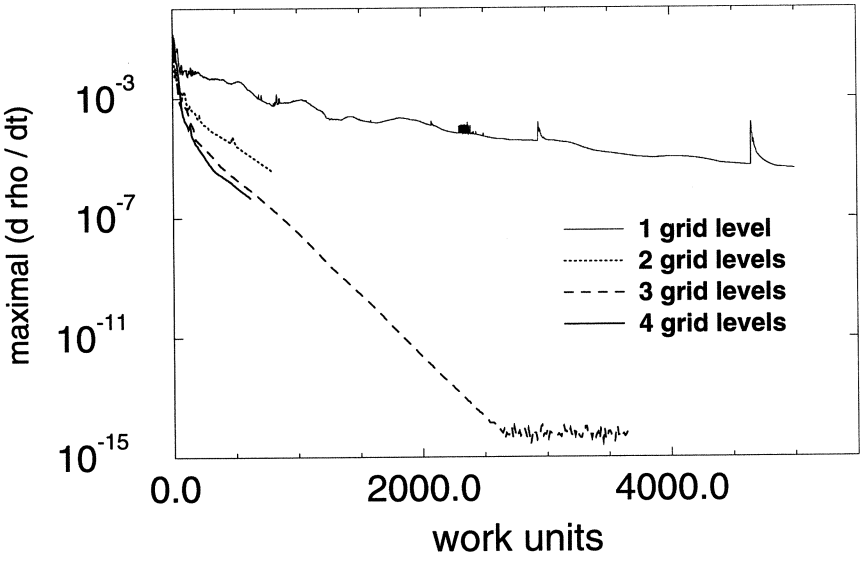


FIGURE 2. Cast 7: residual histories using different grid levels

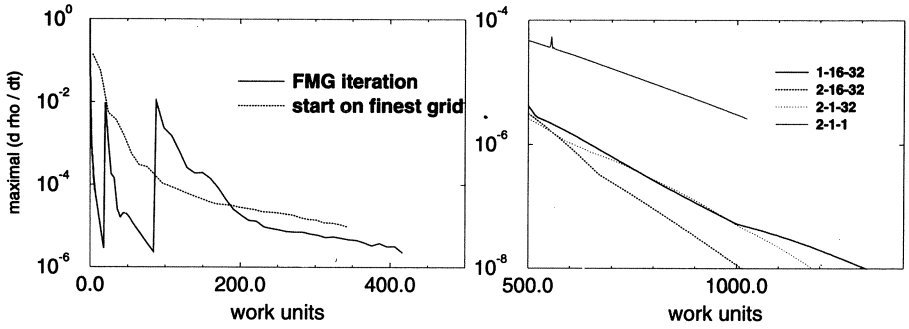


FIGURE 3. Cast 7: FMG iteration (left) and timestep variations (right)

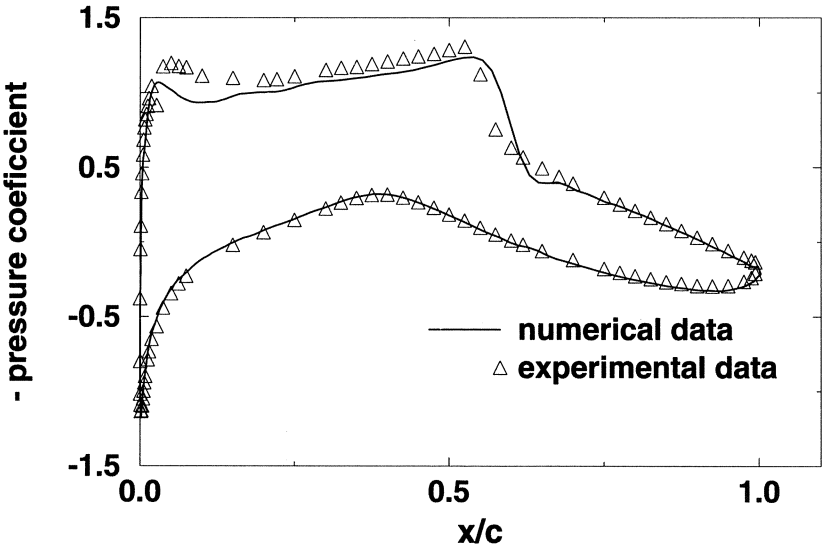


FIGURE 4. RAE 2822:pressure coefficient

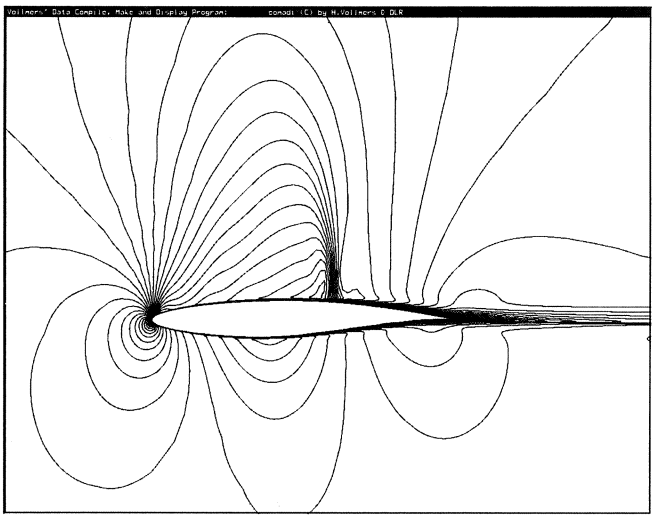


FIGURE 5. RAE 2822: Contour plot of Mach number

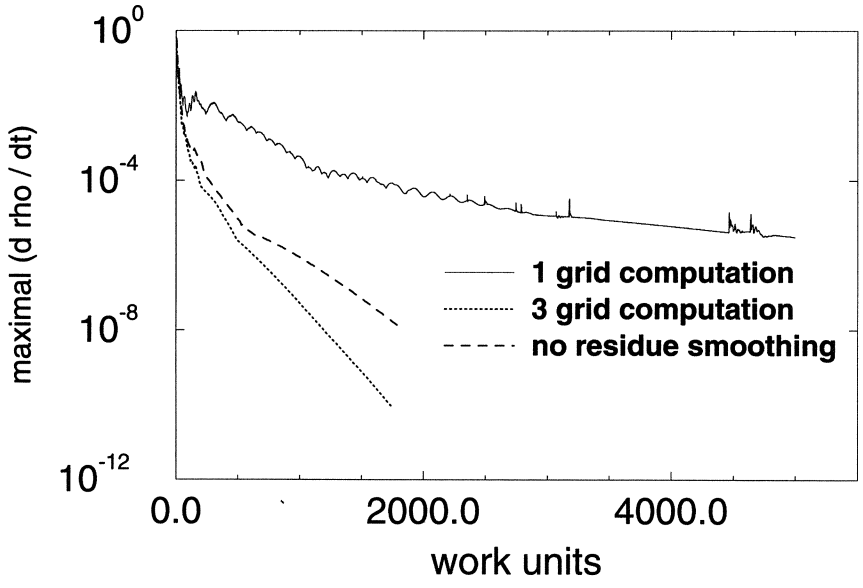


FIGURE 6. RAE 2822: Residual with and without implicit smoothing

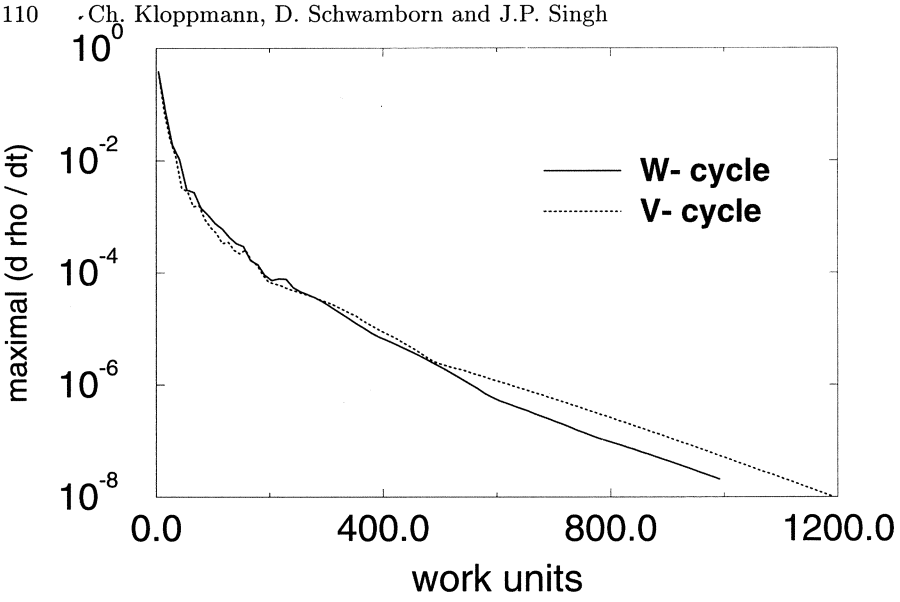


FIGURE 7. RAE 2822: comparison of V- and W- cycle performance

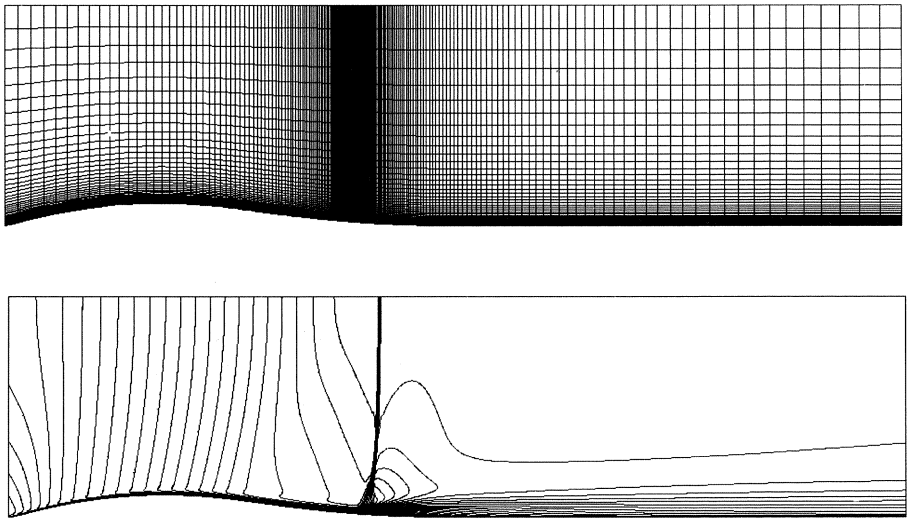


FIGURE 8. Bump A: Grid lines and contour plot of Mach number

TABLE 2. Speed-Up

Three Dimensional V-cycle
with Variable Length Messages
N^3 = 1,000,000,000

Processors Fixed, Variable	256	1024	4096	16,384
2500, 200	253.3	1006.1	3965.4	15,192.1
1000, 200	253.9	1010.3	3999.2	15,555.7
500, 100	254.2	1012.3	4016.9	15,773.9
100, 50	254.4	1013.7	4029.3	15,924.2
3600, 0	253.0	1004.2	3953.3	15,105.8

lems. The ability to bundle messages requires an optimized data partition which minimizes the maximum length of any subdomain border as well as minimizing the number of neighboring regions and load balancing the computation. With structured grids this type of optimized partition is straightforward. With unstructured meshes, however, more sophisticated partitioning techniques are required to produce well-shaped partitions. Several directions for unstructured multigrid partitioning strategies can be found in [3]. With both fixed, constant length and variable length message transmission, analysis on the Current Generation Models suggests that the F-cycle can be more efficient than the V-cycle. With fixed message lengths, this is due mainly to the reduced amount of fine grid communication of the F-cycle. With the ability to send large messages, the F-cycle outperformed the V-cycle in three dimensions because of the reduction in the amount of required computation.

Concurrent Algorithms

The medium granularity of the current generation leaves unexploited much of the subspace parallelism in the concurrent algorithms, making them far too expensive to be practical. With a machines in the range of 256-4096 processors producing more fine grained parallelism is inefficient for even moderate problems sizes. Figure 5 shows the performance of the standard V-cycle and the Chan-Tuminaro algorithm in three dimensions on a problem with 1,000,000 points. Within the medium grain range of machine sizes, the V-cycle is more efficient by at least a factor of two than the concurrent algorithm, even with favorable convergence assumptions.

11 Conclusions

The architectural characteristics of each class of massively parallel computers motivate different optimization strategies to facilitate the realization of the considerable processing power of these machines. Fine grain machines with a high variable cost component in communication costs motivate the optimization of the domain to processor topology mapping. The mapping needs to minimize the topological distance of the required communications. The medium grain machines of the current generation with their high fixed cost of a communication motivate an optimized domain partition. The partition needs to consist of "well-shaped" subdomains in which the maximum size of a subdomain border is minimized and the number of neighboring regions is small. This facilitates the transmission of large messages which allow the fixed cost to be amortized over a large number of words, significantly reducing the average fixed cost per word. With simple optimized strategies the F-cycle is a potentially more efficient than the V-cycle on massively parallel machines. The reduced fine grid activity potentially outweighs the additional coarse grid communications costs on many problems. Concurrent methods even with optimized strategies, require an accurate subspace decomposition and an unreasonably large number of processors to provide a practical alternative to the standard algorithms. This analysis suggests that this remains true regardless of the characterization of inter-processor communications costs.

References

[1] Gannon, D., and Van Rosendale, J., "On the Structure of Parallelism in a Highly Concurrrent PDE Solver", Journal of Parallel and Distributed Computing, vol. 3, pp. 106-135.

[2] JAJA J., "An Introduction to Parallel Algorithms", Addison-Wesley, Reading, MA., 1992.

[3] Matheson, L., "Multigrid Algorithms on Massively Parallel Computers", Phd. Dissertation, Princeton University, 1993.

[4] Tuminaro, R., "Multigrid Algorithms on Parallel Processing Systems", Phd. Dissertation, Stanford University, 1989.

[5] Briggs, W., Multigrid Tutorial, SIAM, Philadelphia, 1987

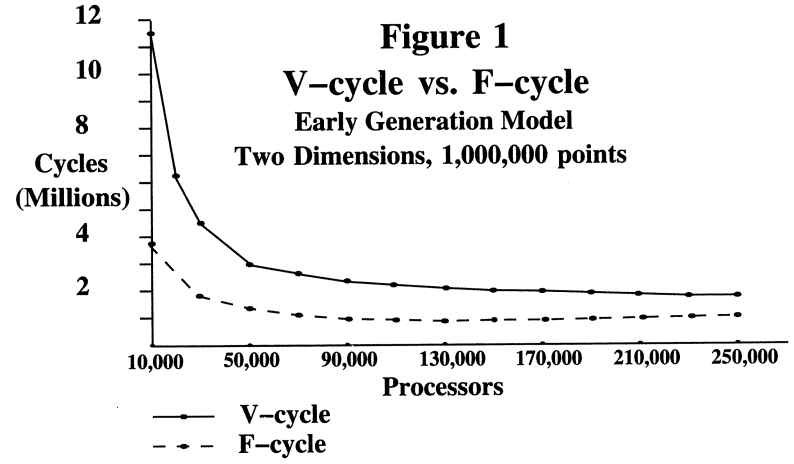


Figure 1
V–cycle vs. F–cycle
Early Generation Model
Two Dimensions, 1,000,000 points

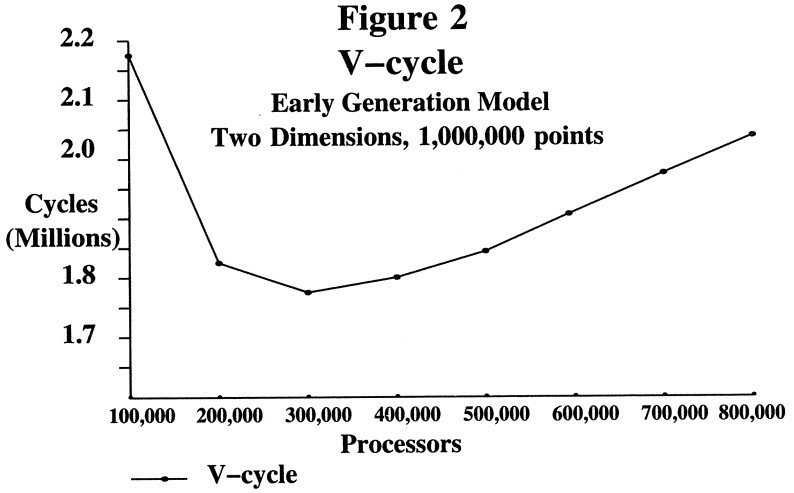


Figure 2
V–cycle
Early Generation Model
Two Dimensions, 1,000,000 points

Figure 3

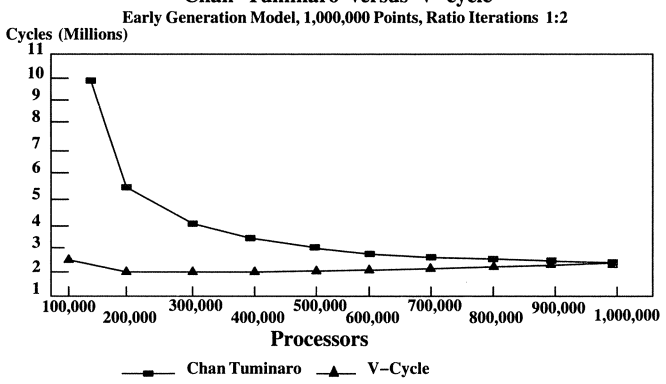


Figure 4

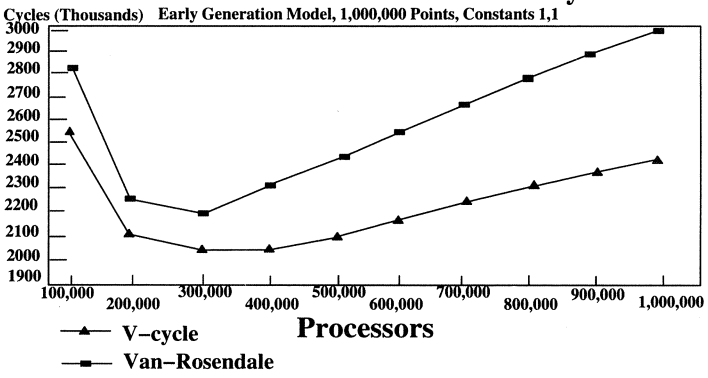
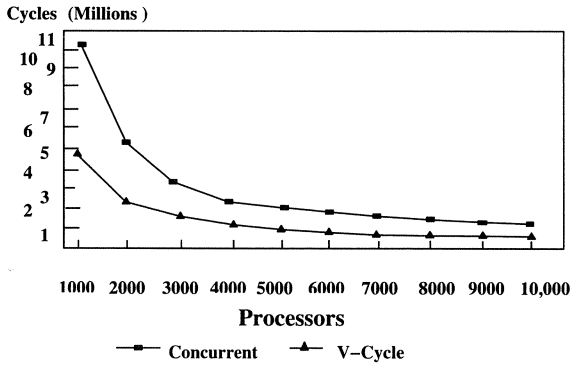


Figure 5



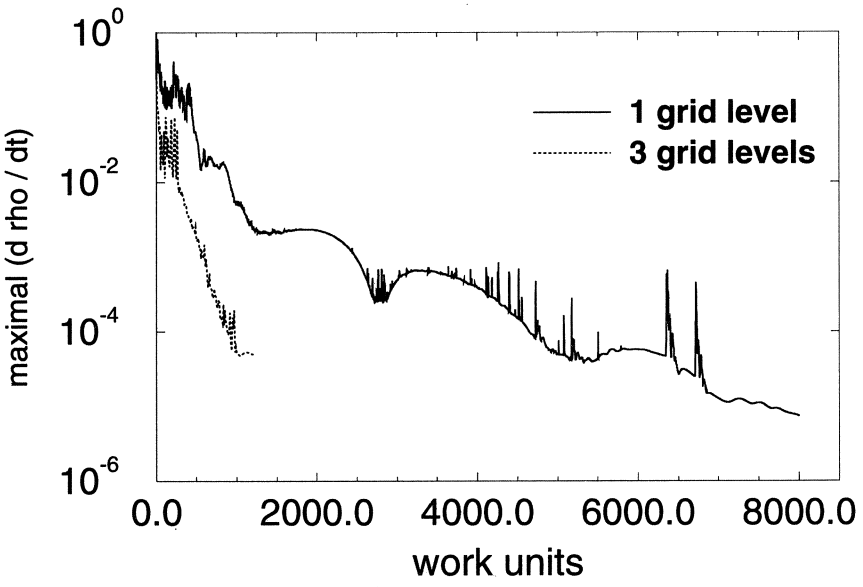


FIGURE 9. Bump A: Maximal density residual history

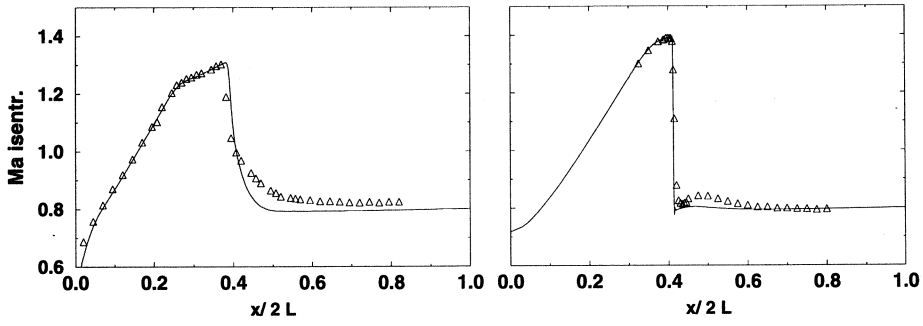


FIGURE 10. Bump A: isentropic Mach number at the wall (left) and at the centerline (right)

10

Unstructured Multigrid by Volume Agglomeration for diffusion problems

B. Koobus[1], M-H. Lallemand[2], G. Carré[3] and A. Dervieux[4]

ABSTRACT We present a Multigrid (MG) strategy for solving second order elliptic PDE's in the unstructured grid context [6], as an extension of the generalized *Finite Volume Agglomeration Multigrid* Euler solver developed in [7, 8]. This agglomeration approach has as prime advantage to allow a fast and automatic generation of the coarse levels from an arbitrary given mesh. A brief description of this *Correction Scheme* MG solver together with numerical results on the 2-D convection/diffusion equation, are given in this paper. Extension to the MG solution of the linearized 2-D compressible Euler and Navier-Stokes equations is also discussed, and a number of preliminary illustrating results are presented. A detailed version of the present paper is found in [5].

Introduction

Applying Multigrid (MG) methods to unstructured meshes is a rather difficult problem today. One way consists in building one mesh for each level as in the so-called non-embedded MG option as in [11, 9, 12, 3].

The introduction of the unstructured Finite Volume Agglomeration MG method [7, 8] has set the problem of dealing with still more general meshes. The agglomeration coarsening consists of grouping fine control volumes together to define coarser control volumes where a coarse grid problem has to be discretized. This coarsening approach allows a fast automatic generation of all the nested coarse levels needed when a MG solver on a Finite Volume discretized system is used. On such general

[1]INRIA, Sophia-Antipolis, 2004 route des Lucioles, B.P. 93, 06902 Valbonne Cedex.
[2]INRIA, Rocquencourt, Domaine de Voluceau, B.P. 105, 78153 Le Chesnay Cedex, FRANCE.
[3]SNECMA Villaroche and INRIA, Sophia-Antipolis, 2004 route des Lucioles, B.P. 93, 06902 Valbonne Cedex.
[4]INRIA, Sophia-Antipolis, 2004 route des Lucioles, B.P. 93, 06902 Valbonne Cedex.

meshes, the difficulty is to define both a suitable approximation scheme and a smoother (iterative relaxation method), such that (at least):

(i) the resulting coarse level discretization scheme is sufficiently accurate to verify the *approximation property* introduced by Hackbusch in [4],

(ii) the specific frequencies of the discrete error are efficiently damped at each level (*smoothing property*, [4]).

For first-order hyperbolic systems such as the steady compressible Euler equations, the definition of such ingredients is quite easy to get and several unstructured Volume Agglomeration MG methods have proved to be at least as efficient as standard ones, and easier to use [7, 8, 16, 21]. For second order PDE's, whereas other MG methods perform well, we are confronted with more severe contraints and these imply no obvious way of defining both the coarse level approximation scheme and the intergrid transfer operators. In [6], we propose an algebraic way of defining the discrete problem on the coarse levels. Property (ii) is kept and we preserve both the symmetry and the positiveness of the discrete operator at each level, while property (i) becomes clearly false. One remedy (a corrective multiplication factor) is proposed in [6] where the efficiency of the new resulting agglomeration MG scheme is shown through its application on a sample of numerical tests for solving the Poisson equation.

When considering multi-level optimisation by unstructured agglomeration, a comparable inconsistancy problem occurs and may be resolved in a slightly different way. We refer to [10] where promising remedies are introduced to verify property (i).

In this paper, we present the method proposed in [6] and an extended version for the solution of a class of compressible viscous flow problems. We give some numerical results and compare them to those obtained in the Euler version of [8].

1 Unstructured agglomeration MG for the Poisson equation

Let us first focus on the simple 2-D Poisson equation

$$-\Delta u = f \ , \ \text{on } \Omega \subset \mathrm{R}^2 \ , \tag{1}$$

with homogeneous Dirichlet boundary conditions, where f is a given function in $L^2(\Omega)$. We set Ω to be simply the unit square and we denote by Γ its boundary.

1.1 The fine grid discretization

We use a Finite Volume / Finite Element (FV/FE) discretization (with P1 Galerkin projection) for the integral formulation of problem (1). Let us denote by τ_h a given triangulation of Ω and by N_h the total number of vertices in τ_h. We denote by φ_j the basis function attached to node s_j (that we will refer as j for simplicity) and by V_h the discrete space associated to the unknown function u_h. For simplicity

again, let us forget the h indices when no confusion appears. From τ_h, we derive a new Finite Volume (FV) partition of Ω, called the dual mesh of τ_h, by defining each control volume (or cell) C_i around each vertex s_i (see Figure 1). Use the P1 Galerkin projection of $u : u = \sum_j u_j \varphi_j$, multiply (1) by χ_i (characteristic function of cell C_i) and then integrate over the whole domain to get

$$- \sum_{j \in K(i) \cup \{i\}} \sum_{T_{ij}} \int_{\partial C_i \cap T_{ij}} u_j \nabla \varphi_j \cdot \vec{n} \, d\sigma = \int_{C_i} f \, dx \, dy \ , \tag{2}$$

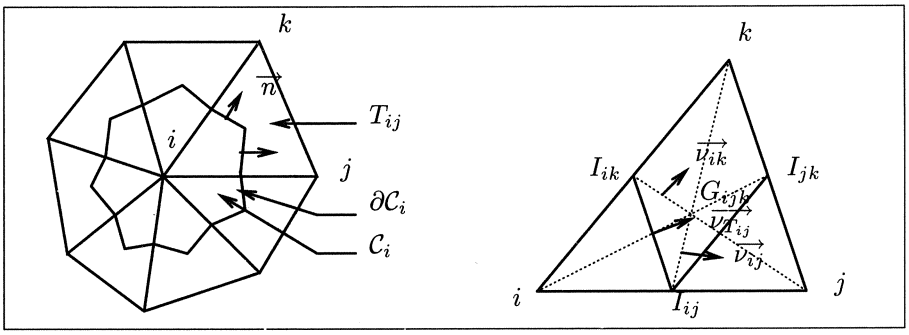


FIGURE 1. Cell C_i and integration path

where $K(i)$ is the set of vertex indices neighbors of i, $\vec{n}$ is the unit outward normal vector to the boundary ∂C_i of C_i, $\partial C_i \cap T_{ij} = [I_{ik}, G_{ijk}] \cup [G_{ijk}, I_{ij}]$, where i, j and k are the three indices of vertices defining T_{ij}; G_{ijk} is the isobarycenter of T_{ij}, I_{ij} and I_{ik} are the midpoints of segments $[i,j]$ and $[i,k]$ respectively. For $j = i$, T_{ii} denotes the support of φ_i, i.e. $\partial C_i \cap T_{ii} = \left(\bigcup_{j \in K(i)} T_{ij} \right) \cap \partial C_i = \partial C_i$. Equation (2) can in turn be rewritten by

$$\sum_{j \in K(i) \cup \{i\}} a_{ij} \, u_j = F_i \ , \tag{3}$$

where F_i is some approximation of the RHS, and

$$\begin{aligned} a_{ij} &= \sum_{k \in T_{ij}} a_{ij}^k \quad , \ a_{ij}^k = \int_{T_{ij}} \nabla \varphi_i \cdot \nabla \varphi_j \, dx \, dy \ , \\ a_{ii} &= \sum_{j \in K(i)} a_{ii}^j \quad , \ a_{ii}^j = \int_{T_{ij}} \| \nabla \varphi_i \|^2 \, dx \, dy \ . \end{aligned} \tag{4}$$

This is exactely what we get in classical FEM. We refer to [5] for more details.

1.2 The coarse grid approximation scheme

In order to keep the Galerkin standpoint, we decide to condens our system by replacing the above basis functions by a smaller set of functions which are linear combinations of the original ones. Let I^f be the set of fine indices i whose cardinal is N_h. We

(i) define the following partition $I^f = \{1, \ldots, i, \ldots, N_h\} = I_1 \cup I_2 \cup ... \cup I_{N_H}$, where $N_H \ll N_h$;

(ii) define the new (coarse) basis functions, ϕ_J, for any subset I_J, $J = 1, \ldots, N_H$ by setting $\phi_J = \sum_{j \in I_J} \varphi_j$.

Defining the N_H–component coarse unknown vector U by the following identification $U_J \equiv u_j$, $\forall j \in I_J$, $J = 1, \ldots, N_H$, the new (coarsened) system to be solved reads as

$$\int_\Omega \left(\sum_{I=1,\ldots,N_H} U_I \nabla\phi_I \cdot \nabla\phi_J - f\phi_J \right) dv = 0 \qquad \forall J = 1, \ldots, N_H . \tag{5}$$

In practise, the subsets I_J are built from neighboring relations; the coarsening algorithm is straightly deduced from an efficient algorithm which has been extensively used in [8] and defined in [6]. The resulting compressed/agglomerated system is inconsistent, as shown in the 1-D case in [6], for a simple scalar advection/diffusion equation, because the resulting coarse grid discrete space V_H is not an approximation space of $H^1(\Omega)$ anymore, but is still an approximation space of $L^2(\Omega)$ (with corresponding norms). In order to overcome the coarse grid inconsistency problem, in [6], we have introduced the following correction, which is empirically derived from simplified cases: in the corrected system, viscous terms are multiplied by the factor $K_{\mathcal{N}} = 2\,(\mathcal{N}-1)^2/(2\mathcal{N}-1)^2$, where $\mathcal{N}$ is the number of nodes in one direction (if $\mathcal{N}_x \neq \mathcal{N}_y$, or when using an unstructured mesh, $\mathcal{N}$ is chosen to be equal to $\sqrt[d]{NS_k}$, where d is the space dimension and NS the total number of control volumes at level k. With this correction, we have the

Lemma 1.1 *The above correction yields a consistent coarsened approximation in the case of a cartesian mesh (orthogonal, regular) discretizing a rectangle.*

A proof is given in [6] with $f = 1$. Despite this inconsistency, note that this agglomeration procedure preserves positiveness and symmetry of the discrete operator at each level, if the fine grid matrix is symmetric, positive definite (this is true when the triangulation contains no obtuse angles, otherwise positiveness of the discrete operator is lost).

2 The simplified linearized Navier-Stokes model

2.1 The 2-D isenthalpic Navier-Stokes equations

In this paper, we focus on a simplified conservative law form of the 2-D isenthalpic Navier-Stokes equations given by

$$\begin{aligned} &\nabla_t(W) + \nabla_s \cdot \mathcal{F}(W) = \frac{1}{Re} \nabla_s \cdot \nabla_s \mathcal{N}(W) \ , \text{ in } \Omega_t \ , \\ &+\text{initial and boundary conditions,} \end{aligned} \tag{6}$$

where $\Omega_t = \Omega \times \mathrm{R}^+$, Ω is an open bounded subset of R^2, $\nabla_t = \partial/\partial t$, and $\nabla_s = (\partial/\partial x \ , \ \partial/\partial y)^T$; $\mathcal{F}(W) = (F(W), G(W))^T$, $\mathcal{N}(W) = (0, u, v, 0)^T$, and

$$W = \begin{pmatrix} \rho \\ \rho\, u \\ \rho\, v \\ E \end{pmatrix} \ , \ F(W) = \begin{pmatrix} \rho\, u \\ \rho\, u^2 + p \\ \rho\, u\, v \\ (E+p)\, u \end{pmatrix} \ , \ G(W) = \begin{pmatrix} \rho\, v \\ \rho\, u\, v \\ \rho\, v^2 + p \\ (E+p)\, v \end{pmatrix} \ , \tag{7}$$

W is the vector of the conservative variables. For a zero RHS ($Re \to +\infty$), we get the compressible Euler equations, otherwise we get a simplified version of the full Navier-Stokes equations (viscous terms are reduced to the Laplacian of the velocity components). Let Γ denote the boundary of Ω which is splitted into the farfield boundary Γ_∞ and the solid boundary Γ_B. The usual slip boundary condition is applied on Γ_B for Euler computations, otherwise the no-slip condition is used. We set a uniform flow at Γ_∞.

The spatial approximation used to discretize system (6) is based on a FV/FE discretization. As in the previous section, let τ_h be some triangulation discretizing the initial domain Ω, from which a dual FV mesh is derived. For the inviscid terms we use a FV formulation and we apply a FE (P1 Galerkin) formulation for the viscous terms. For a fixed time t_n, i.e. for a given solution W_n, we are solving the linearized isenthalpic Navier-Stokes system which can be formally written by

$$\left(\mathbf{Id} + \mathbf{D}^{-1}\,\mathbf{M}\right)\, \delta W = \mathbf{D}^{-1}\, RHS \ , \tag{8}$$

where $\mathbf{D}$ is a 4×4 block diagonal matrix corresponding to discretization of the time derivative with mass lumping, $\mathbf{M} \equiv A + B$, with A standing for the discretized (FV + flux splitting procedure [14, 18, 20]) Jacobian $\partial \mathcal{F}(W_0)/\partial W$ of the Euler flux function $\mathcal{F}(W)$, and B designing the discretized Jacobian (P1 FEM) of the diffusive flux function $\nabla_s \mathcal{N}(W)$; $\delta W \equiv W^{n+1} - W^n$, and $RHS \equiv -(\mathcal{F}(W^n) + \nabla_s \mathcal{N}(W^n))$. This linear system has to be solved at each time step to provide the change δW of the upated solution W^{n+1}. In our computations, the first order accurate version of one selected splitting function is used for the inviscid Jacobian, and first order or second order flux splittings are used for the approximation of the inviscid RHS. The viscous Jacobian is easily derived since it corresponds to a linear operator, and is discretized by classical P1 FEM. For more details (linearization, accurate flux splitting) see [2, 15, 5].

3 The coarse grid discretization

We refer to [7, 8] for a complete description of the coarse grid discretization of the inviscid terms. We just recall that we use a first order extension of the upwind scheme used on the finest level. For the viscous terms, they are computed following

the describtion given in Section 1.2. We apply a similar linearization as the one done on the finest grid, giving the coarse linear system that we would have to solve during the MG cycling for solving the change δW for the updating of the solution W at time t_{n+1}.

4 MG ingredients: transfer operators, smoothers and cycles

Transfer operators:
Due to the algebraic way of constructing the coarse grid equations (see Remark 1.1 in [6]), some of the good properties of the fine–grid matrix will also hold for all the successive coarse–grid matrices; in particular, if the initial matrix is an M–matrix, then the resulting coarse matrices are also M–matrices. Residual restriction is done by summing fine residuals belonging to the same coarse cell, and either trivial injection or weighted interpolation are used as residual prolongations.
Smoothers:
We refer to [19, 8, 12] for more details (stability and smoothing analysis) on the smoothers used in the MG cycling. In addition to the classical pointwise block Gauss-Seidel (GS) relaxation, two types of smoothers are used and have been chosen for their low-storage requirements:

1. Pointwise Block Jacobi with relaxation parameter ω, refered as PBJ(ω),
2. Pointwise Block Runge-Kutta 4 steps Jacobi, refered as PBJ-RK4.

Cycles:
For the numerical tests, we have just used the classical V(ν_1,ν_2)-cycle, with $\nu_1 = \nu_2 = 1$.

5 Numerical results

5.1 2-D CONVECTION/DIFFUSION EQUATION

The problem to be solved is $div(\vec{V}u) - \epsilon\,\Delta u = 1$, with homogeneous Dirichlet boundary conditions. We have done two MG computations on a NACA0012 mesh (3114 nodes) with two smoothers PBJ(ω) and GS. A V(1,1)-cycle with 6 levels is used for both calculations. The corresponding grids are depicted in Figure 2. In Table 1, we give the corresponding reduction factors (denoted by μ) and the number of cycles needed to reduce the initial residual by a factor of 10^{-6} (denoted by α_{conv}) when solving the Poisson equation.

For this example, the MG-GS is more than 40 times more efficient than the single grid GS solver (which needs more than 3000 iterations to get the same reduction level). To illustrate the coarse grid inconsistancy, and the effect of the correction rule applied to the diffusive operator, we have depicted in Figure 3 the coarse grid solutions for $y = 0$ (first coarser level) with and without the correction factor and compare them to the fine grid solution.

	MG-PBJ(0.8)	MG-GS
μ	0.690	0.496
α_{conv}	39	21

TABLE 1. $\varepsilon = 1$, $\mathbf{V} = (0,0)^T$.

In Table 2, we give the different values of μ and corresponding α_{conv} when the MG-GS solver is used for $\vec{V} = (1,0)^T$ and decreasing values of ϵ. Note that

ε	1	10^{-1}	10^{-3}
μ	0.422	0.402	0.399
α_{conv}	17	17	17

TABLE 2. $\mathbf{V} = (1,0)^T$, MG-GS.

there is a slight improvement in μ as ϵ goes to zero.

5.2 2-D ISENTHALPIC NAVIER-STOKES EQUATIONS

Here are presented some numerical tests for a transonic ($M_\infty = .85$, zero angle of attack) flow around a NACA0012 airfoil profile. Three triangulations are used, containing 800 nodes (G_1), 3114 nodes (G_2) and 12284 nodes (G_3). We are here interested in the linear convergence of the MG algorithm defined with PBJ(0.9) and a V(1,1)-cycle. The CFL number is set to 1000. A first experiment is done with the ideal two-grid solver (I2G) (i.e. the second grid is the first coarser level and the problem is solved accurately here), for the 3114 node problem, to compare the corresponding μ's gotten when solving the 2-D Euler system or the isenthalpic Navier-Stokes (INS) system, for which Re is set to 100. We get $\mu = 0.76$ for Euler and a slighty better $\mu = 0.714$ for INS. For the same fine grid, we use this time the MG solver (6 levels). The resulting reduction factor for Euler is quite sensitive to the splitting used for the Jacobian matrix. Indeed, when the Steger-Warming splitting is used, $\mu = 0.884$, which reduces to $\mu = 0.803$ when using the van Leer splitting; corresponding linear convergence histories are depicted in Figure 4. For INS (with the van Leer splitting for the inviscid terms), we obtain $\mu = 0.763$, which is rather close to what obtained by I2G. We have a quasi mesh independancy if we compare the asymptotic I2G and MG reduction factors (μ) in the linear phase of INS obtained on the different grids G_i, $i = 1,2,3$ which are given in Table 3. Numbers in parenthesis correspond to the total iteration numbers needed to reach machine zero. Additional tests are included here in order to check on the sensitivity of the linear convergence versus the variations the Mach number. In Figure 5, three Mach numbers are used : .85, .085 and .0085: the asymptotic convergence factors improve with decreasing Mach numbers when the same CFL number (1000) is used for all calculations. A possible explanation of this improvement is that the physical allowable time step increases as the Mach

number decreases, leading consequently to better conditioned matrices (diagonal dominancy). If we now increase accordingly the CFL as the Mach number decreases by the same factor, then the asymptotic convergence behaviour is not so good (Figure 6).

Conclusion

This work is a first step in solving the full linearized compressible Navier-Stokes equations. A new definition of the coarse grid basis functions allows a more accurate (but yet not consistent) variational formulation via a corrective multiplication factor for the diffusive terms. The result can be considered as an answer to the specific need of a linear (or nonlinear) method applying transparently to compressible flows computed on unstructured finite volumes.
Several causes of less good efficiency have been pointed out. However a rather good robustness is obtained for a large range of Mach number, Reynolds number and mesh size.
To show the global efficiency, we have tested a $V(1,1)$ cycle with Jacobi iteration. For decreasing the residual by a factor 10, arround 10 cycles and 40 WU (1 Jacobi = 1 WU) are needed. In the case of an unnested MG scheme, the figure of 40 WU was also obtained in [13], for a non-linear system. Figures of this type are also obtained for structured meshes [17] of comparable size.

References

[1] L. Fézoui, "Résolution des équations d'Euler par un schéma de van Leer en éléments finis", Rapport de Recherche INRIA, N 358, 1985.

[2] L. Fézoui, B. Stoufflet, "A class of implicit upwind schemes for Euler simulations with unstructured meshes", J.C.P., 84, pp. 174-206, 1989.

[3] H. Guillard, "Node-Nested Multi-Grid with Delaunay Coarsening", Rapport de Recherche INRIA, N 1898, 1993.

[4] W. Hackbusch, "Multi-Grid Methods and Applications", Springer series in Computational Mathematics, **4**, Springer-Verlag Berlin, Heidelberg 1985.

[5] B. Koobus, M.-H. Lallemand, G. Carré and A. Dervieux, "Unstructured Multigrid by Volume Agglomeration: application for viscous compressible flow problems", Rapport de Recherche INRIA, in preparation.

[6] B. Koobus, M.-H. Lallemand and A. Dervieux, "Unstructured Volume-Agglomeration MG: Solution of the Poisson Equation", Rapport de Recherche INRIA, N 1846, 1993.

[7] M.-H. Lallemand, A. Dervieux, "A Multigrid Finite Element Method For Solving The Two Dimensional Euler Equations", Proceedings of the Third Copper Mountain Conference on Multigrid Methods, April 6-10, 1987.

[8] M.-H. Lallemand, H. Steve and A. Dervieux, "Unstructured Multigridding by Volume Agglomeration: Current Status", Computer Fluids, 21, **3**, pp. 397-433, 1992.

[9] M.-P. Leclercq and B. Stoufflet, "Characteristic Multigrid Method, Application to solve the Euler equations with unstructured and unnested grids", International Conference on Hypersonic Problems, Upsala, 1989.

[10] N. Marco, "Résolution de l'équation de Poisson par optimisation hiérarchique", Rapport de Recherche INRIA, in preparation.

[11] D. Mavriplis and A. Jameson, "Multigrid Solution of the two-dimensional Euler equations on unstructured meshes", AIAA paper, **87-0350**, January 1987.

[12] E. Morano, "Résolution des Equations d'Euler par une méthode Multigrille stationnaire", Thèse de Docteur, Université de Nice, 1992.

[13] E. Morano and A. Dervieux, "Looking for $O(N)$ Navier-Stokes solutions on non-structured meshes", to be published by Society for Industrial and Applied Mathematics.

[14] S. Osher, S. Chakravarty, "Upwind schemes and boundary conditions with applications to Euler equations in general geometries", J.C.P., 50, **3**, pp. 447-481, 1983.

[15] P. Rostand, B. Stoufflet, "TVD schemes to compute compressible viscous flows on unstructured meshes", Notes on Numerical Fluid Mechanics, 24, Vieweg, Braunschweig 1989, pp. 510-520, 1989.

[16] W.A. Smith, "Multigrid Solution of Transonic Flow on Unstructured Grids", Recent Advances and Applications in Computational Fluid Dynamics, Proceedings of the ASME Winter Annual Meeting, O. Baysal (ed.), November 1990.

[17] S.P. Spekreijse, "Multigrid solution of the steady Euler equations",Thesis, CWI Amsterdam (1987).

[18] J. Steger, R.F. Warming, "Flux vector splitting for the inviscid gas dynamic with applications to finite difference methods", J.C.P., 40, pp. 263-293, 1981.

[19] H. Steve, "Schémas implicites linéarisés décentrés pour la résolution des équations d'Euler en plusieurs dimensions ", Thèse de doctorat, Université de Provence, 1988.

[20] B. van Leer, "Flux vector splitting for the Euler equations", Lecture Notes in Physics, 170, pp. 405-512, 1982.

[21] V. Venkatakrishnan, D.J. Mavriplis, M.J. Berger, " Unstructured Multigrid through Agglomeration", 6th Copper Mountain Conf. on MG Methods, April 4-9, 1993.

	G_1	G_2	G_3
I2G	0.684 (72)	0.703 (73)	0.775 (112)
V(1,1)	0.703 (78)	0.740 (95)	0.813 (140)

TABLE 3. Asymptotic I2G and MG-PBJ(0.9) reduction factors.

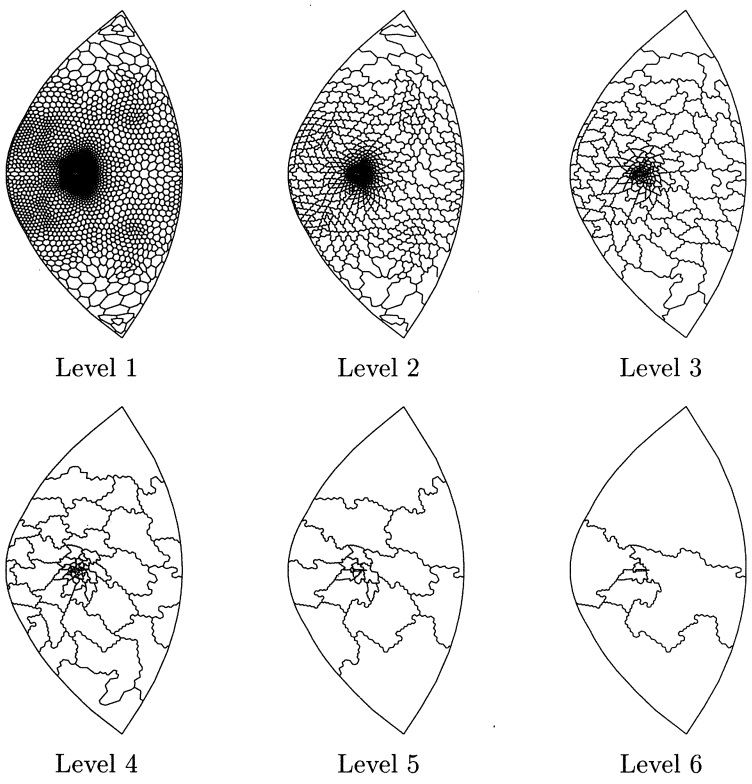


FIGURE 2. The NACA0012 airfoil finite-volume partition grids.

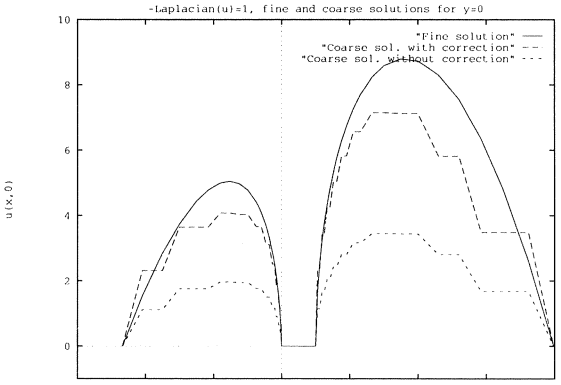


FIGURE 3. Poisson equation on the 3114 node mesh - Fine and coarse solutions (with and without correction).

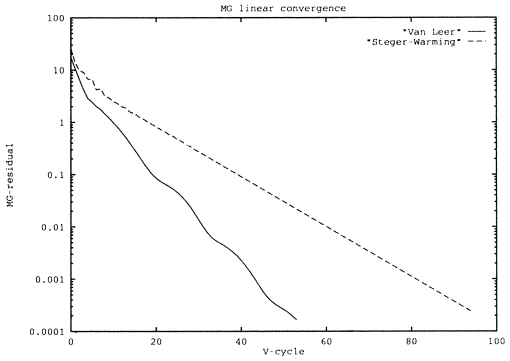


FIGURE 4. MG linear convergence histories for 2-D Euler system vs. the use of different splittings of the Jacobian matrix

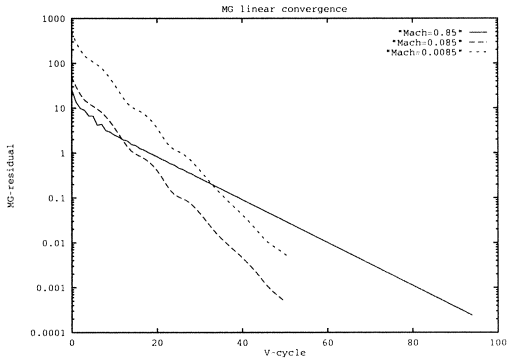


FIGURE 5. Euler: linear convergence sensitivity vs. M_∞ (1)

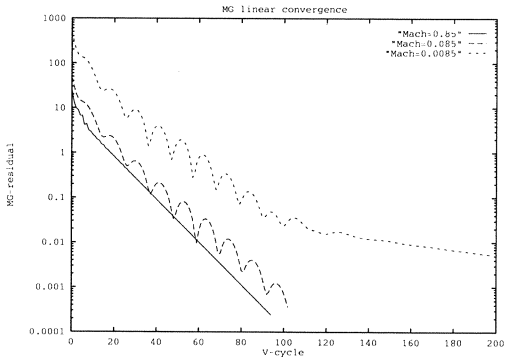


FIGURE 6. Euler: linear convergence sensitivity vs. M_∞ (2).

11

A Multigrid Multiblock Solver for Compressible Turbulent Flow

Hans Kuerten and Bernard Geurts[1]

ABSTRACT We describe a multigrid multiblock method for compressible turbulent flow simulations and present results obtained from calculations on a two-element airfoil. A vertex-based spatial discretization method and explicit multistage Runge-Kutta time-stepping are used. The slow convergence of a single grid method makes the multigrid method, which yields a speed up with a factor of about 12.5, indispensable. The numerical predictions are in good agreement with experimental results. It is shown that the convergence of the multigrid process depends considerably on the ordering of the various loops. If the block loop is put inside the stage loop the process converges more rapidly than if the block loop is situated outside the stage loop in case a three-stage Runge-Kutta method is used. If a five-stage scheme is used the process does not converge in the latter block ordering. Finally, the process based on the five-stage method is about 60% more efficient than with the three-stage method, if the block loop is inside the stage loop.

1 Introduction

Numerical simulations of turbulent flow in aerodynamic applications are frequently based on the Reynolds-averaged Navier-Stokes equations. One of the main problems in aeronautics is the prediction of flow quantities in complicated geometries, such as the multi-element airfoil (see figure 1). The simulation of turbulent flow

FIGURE 1. Geometry of a two-element airfoil

around such a multi-element airfoil configuration was one of the applications selected for the compressible flow solver which was developed by our group and

[1]Department of Applied Mathematics, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands

NLR as a part of the Dutch ISNaS[2] project [1]. For this application the use of a single-block, boundary-conforming, structured grid is impossible and one may select either an unstructured grid approach or a block-structured grid approach. Although the former technique has been successfully applied by others [2], we selected the block-structured approach in view of the transparent data structure in the coding, ease of implementation of the turbulence model and a high flexibility with respect to the use of different physical models in different parts of the computational domain.

In a previous paper [3] it has been shown that for laminar and turbulent flow around a single airfoil the introduction of the multiblock structure has no influence on the results, with respect to both the steady-state solution and the convergence rate. Furthermore, invoking the Euler equations instead of the Navier-Stokes equations in blocks outside the boundary layer appeared to have no significant influence on the results. In this paper we describe the application of the multiblock concept to the multi-element airfoil. If the Euler equations are used throughout the computational domain a converged steady-state solution is obtained within a reasonable calculation time. However, if the Reynolds-averaged Navier-Stokes equations are solved in the boundary layers, the rate of convergence is unacceptably low. Therefore, a multigrid technique was implemented in order to accelerate the convergence. The resulting gain in computational effort is close to a factor of 12.5, and the converged solution is in good agreement with wind-tunnel measurements.

In section 2 the numerical technique, which is based on a combination of a finite volume method with central spatial differencing and a Runge-Kutta explicit time-stepping method, is described. The results, both for inviscid and for viscous simulations are presented in section 3. Finally, in section 4 some conclusions are summarized.

2 Numerical Method

In this section we describe the numerical method used in the flow solver. The two-dimensional, compressible Navier-Stokes equations can be written in integral form as

$$\frac{\partial}{\partial t}\left[\iint_\Omega U dx dy\right] + \int_{\partial\Omega}(F dy - G dx) = 0, \tag{1}$$

where U represents the vector of dependent variables,

$$U = [\rho, \rho u, \rho v, E]^T, \tag{2}$$

with ρ the density, u and v the Cartesian velocity components, and E the total energy density. Further, Ω is an arbitrary part of the two-dimensional space with

[2]ISNaS is an abbreviation for Information System for flow simulation based on the Navier-Stokes equations, and is a cooperation of Delft Hydraulics, the National Aerospace Laboratory NLR and the Universities of Delft and Twente, The Netherlands.

boundary $\partial\Omega$ and F and G are the Cartesian components of the total flux vector. This flux vector consists of two parts: the non-dissipative or 'convective' part and the dissipative or 'viscous' part, which describes the effects of viscosity and heat conduction, and involves first order spatial derivatives. The Navier-Stokes equations (1) are averaged over a sufficiently large time interval. Due to the nonlinear terms in the convective fluxes, the resulting 'Reynolds-averaged Navier-Stokes' equations involve averages of products of two velocity components. These terms are modeled by a suitable turbulence model. In the present paper the algebraic Baldwin-Lomax turbulence model, in which the unknown terms are modeled by eddy viscosity terms, is adopted [4].

The discretization of the Navier-Stokes equations follows the method of lines, i.e. the spatial discretization is performed first, and subsequently the resulting set of ordinary differential equations is integrated in time, until the steady state solution is approximated. First the computational domain is divided into blocks and each block is partitioned in quadrilateral cells with the help of a structured, boundary-conforming grid. The variables are stored in the grid points. A finite volume method is used in which the integral form of the Navier-Stokes equations is applied to a control volume Ω, bounded by the dashed lines in figure 2. The

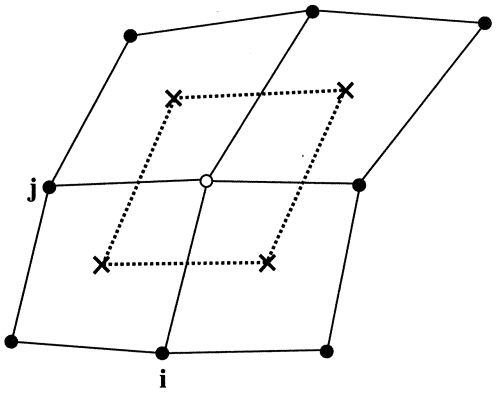


FIGURE 2. Control volume in the vertex-based method

convective flux through a boundary of this control volume is approximated using the value of the convective flux vector in the midpoint of the boundary. The latter is calculated by averaging over the two neighboring grid points. The viscous flux vector involves spatial derivatives of the state vector U and is approximated in the corner points of the control volume with the use of Gauss' theorem on a grid cell. The viscous flux is subsequently calculated using the trapezoidal rule. This method is called the vertex-based method.

The method of central differencing leads to a decoupling of odd and even grid points and to oscillations near shock waves. Even in viscous flow calculations the presence of the viscous dissipation is insufficient to damp these instabilities outside

shear layers. Therefore, nonlinear artificial dissipation is added to the basic numerical scheme. This artificial dissipation consists of two contributions: fourth order difference terms which prevent odd-even decoupling, and second order difference terms to resolve shock waves. The second order terms are controlled by a shock sensor, which detects discontinuities in the pressure. In the present flow solver the artificial dissipation in the boundary layers, where the viscous dissipation should be dominant, may be reduced by multiplication with the ratio of the local and free-stream Mach number. The role of the artificial dissipation in relation to the viscous dissipation is discussed in more detail in reference [5].

At the solid wall boundaries the no-slip condition is used. The density and energy density in the grid points on a solid wall are calculated by solving the corresponding discrete conservation laws, using the two adjacent cells within the computational domain and their mirror images inside the wall as control volume. The values of the density and energy density in the grid points inside the walls are adjusted such that the adiabatic wall condition is approximated. The boundary conditions at a (subsonic) far-field boundary are based on characteristic theory. The extent of the computational domain can be reduced without affecting the accuracy if a vortex is superimposed on the incoming free stream outside the computational domain [6].

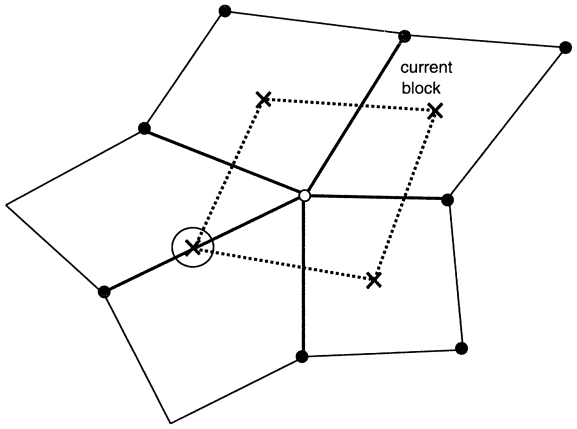


FIGURE 3. Control volume for a special point

Due to the topology of the two-element airfoil geometry special points in the computational grid are unavoidable. The computational grids used contain two special points at block boundaries, where five cells meet. These points can be treated in an elegant way within the same numerical scheme, if the dummy vertices outside the 'current' block are defined appropriately. The multi-valuedness of the variables at the special point, caused by this asymmetric treatment, is eliminated by taking the average of the five different values after all blocks have been treated. This is sketched in figure 3.

The system of ordinary differential equations, which results after spatial discretization, is integrated in time using a time-explicit multistage Runge-Kutta method. In the present flow solver a three-stage scheme in which the dissipative fluxes (both viscous and artificial) are calculated once per time-step, and a five-stage scheme in which the dissipative terms are calculated only at the odd stages, are implemented. With this treatment both calculation time is saved and the stability region of the method is increased. Extra calculation time is saved by advancing each grid point at the maximum local time-step according to its own stability limit. In this way the evolution from the initial solution to the steady state is no longer time accurate, but the steady state solution obtained is unaffected.

The above time-stepping method acts as the relaxation method and coarse grid operator in the multigrid solver (see reference [6]). In this solver an initial solution on the finest grid is obtained with a full multigrid method. This initial solution is corrected in the FAS-stage, where either V- or W-cycles can be chosen. A fixed number of pre- and post-relaxations is performed before turning to the next coarser or finer grid. The solution is transferred to a coarser grid by injection, the residuals by full weighting and the corrections to the solution are prolonged by bilinear interpolation. In order to increase the smoothing properties of the Runge-Kutta time-stepping technique an implicit averaging of the residuals is applied with frozen residuals at the block boundaries. For mono-block applications this method has given satisfactory results for both two-dimensional and three-dimensional flows [5].

In the multi-element airfoil application care has to be taken in the definition of the residual-vector in the special points. The proposed treatment of a special point implies that the control volume is different in each of the five blocks where such a point is found. In the required averaging the five residual-vectors in a special point are weighed with their corresponding time-steps. Without this weighing the multigrid process cannot converge to the single grid stationary state solution.

In this multigrid, multiblock solver with a multistage time-stepping method there are various possibilities for intertwining the different loops. In the present study the grid loop is chosen as the outer loop and the effect of interchanging the block and the stage loop will be studied. Several 'competing' requirements serve as possible guidance for selecting a specific ordering of these loops. On the one hand an anticipated parallel processing of the different blocks is more efficient, if the data transfer between the blocks is kept to a minimum, i.e. with the stage loop inside the block loop. On the other hand the good convergence of the multigrid mono-block solver may be reduced as the dummy variables near the block boundaries are kept frozen during more stages of the time-step. This would suggest to put the block loop inside the stage loop. In order to study this dilemma we implemented these two loop orders in a flexible way: a single parameter determines whether the block loop is situated inside or outside the stage loop.

3 Results

3.1 Description of the test-case

We will present results for a two-component airfoil geometry consisting of the NLR7301 wing section, from which a flap has been cut out at a deflection angle of 20° and with a gap width of 2.6% chord length [7] (see figure 1). The combination of a Mach number of 0.185 and an angle of incidence of 6° or 13.1°, of which the latter is close to maximum lift conditions, yields subsonic flow. The Reynolds number based on the chord length of the airfoil is 2.51×10^6. In the viscous calculations the locations of the transition from laminar to turbulent flow are prescribed.

The C-type computational grids (either for inviscid or viscous flow) were constructed by J.J. Benton from British Aerospace, and are subdivided in 37 blocks (see figure 4). The grid lines are continuous over block boundaries. Two grids are used: one 'Euler' grid (inviscid) consisting of 16448 cells, and a 'Navier-Stokes' grid (viscous), which is refined in the boundary layers and wakes and consists of 28288 cells.

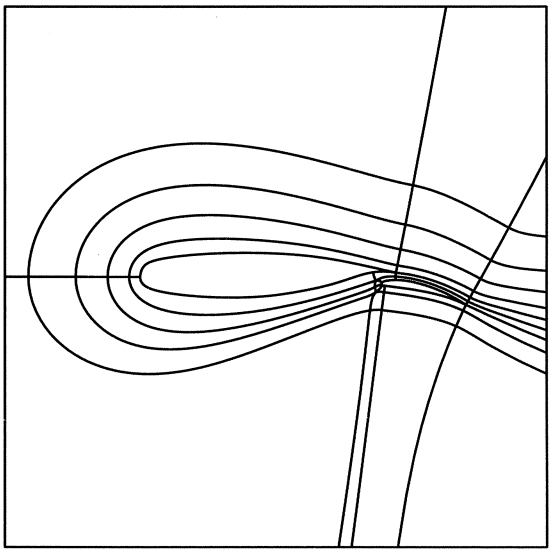


FIGURE 4. Block structure of the computational grid

For both angles of incidence results from wind-tunnel measurement by Van den Berg [7] are available, including velocity profiles in the boundary layers and the pressure coefficient on the profile. Since the flow is attached apart from a small laminar separation bubble near the leading edge of the wing, the adopted turbulence model should be adequate and yield a useful comparison between experiment

and calculation.

3.2 Inviscid Flow

In order to test the flow solver on the complicated block structure of the two-element airfoil geometry, we considered the relatively simple inviscid flow case, where in all blocks the Euler equations are solved. In this way problems related to the turbulence model are separated from possible algorithmic problems. The use of the Euler equations implies that the boundary conditions at the solid wall boundaries have to be changed. For inviscid flow there is only one physical boundary condition of zero mass flux through the wall. In the vertex based approach the density, the pressure and the tangential velocity at the wall are approximated by linear extrapolation.

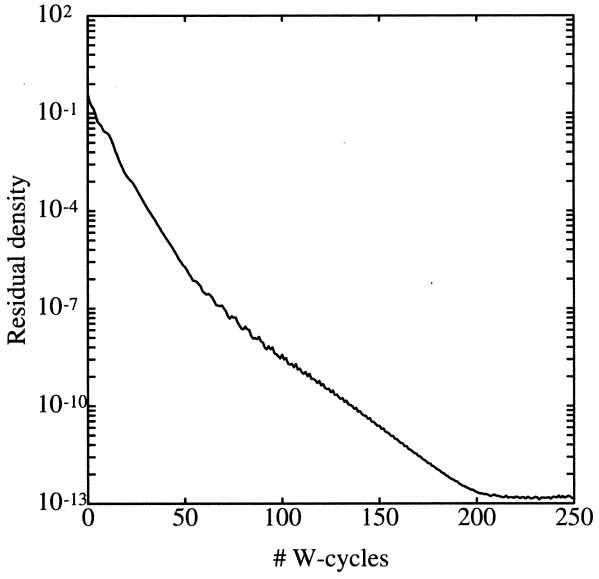


FIGURE 5. Convergence behavior for inviscid flow at an angle of incidence of 13.1°

In figure 5 the multigrid convergence behavior of the solver in the 13.1° case is shown. The discrete L_2-norm of the residual of the density is plotted as a function of the number of W-cycles. A converged solution is obtained within a much smaller calculation time when compared to the single grid approach even though only three different grid levels are available. Both for the single grid and the multigrid calculations machine accuracy was obtained. The specific block structure nor the treatment of the special points leads to any specific difficulties. For this inviscid test a comparison with experimental results is not meaningful and will

not be made.

3.3 Viscous Flow

We consider the simulations of turbulent, viscous flow and present results for the 6° case only. Single-grid calculations in which only local time-stepping is applied as a convergence acceleration technique yield a steady-state solution which is in good agreement with the experimental results. However, in contrast with a fully inviscid simulation, the rate of convergence is very small, and renders this method unacceptable for practical applications. Therefore, as a method to increase the convergence rate further, the multigrid technique and implicit residual averaging as described in section 2 are indispensable.

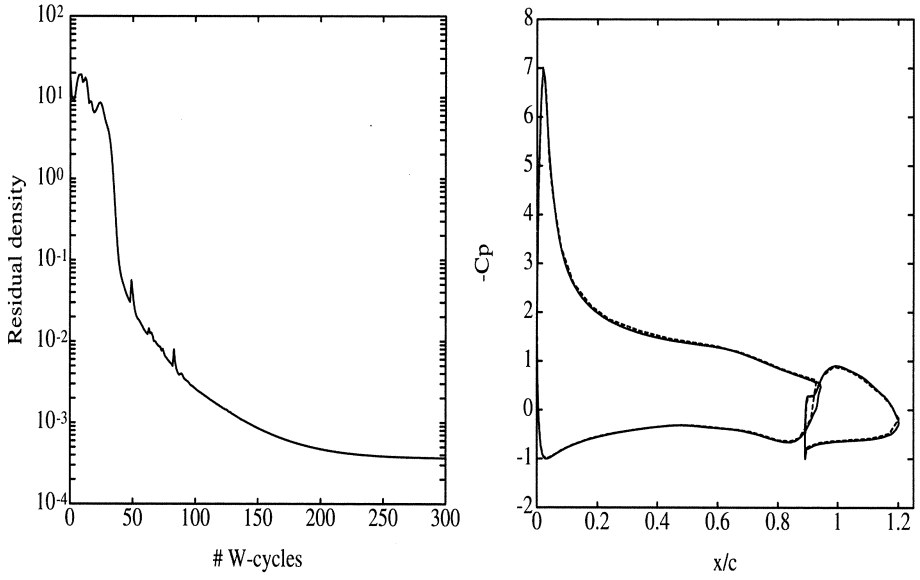


FIGURE 6. Viscous flow at an angle of incidence of 6.0°; left: convergence behavior; right: comparison of the pressure coefficient on the airfoil between calculation (solid) and experiment (dashed)

In a simulation of turbulent flow at high Reynolds number it is important that the effects related to the physical dissipation are not outweighed by those of the numerical or artificial dissipation. This requirement could give rise to difficulties in the present multigrid method, since the time-stepping method used requires a certain minimum amount of dissipation for sufficient smoothing of the large wave-number components of the error (see reference [5]). If the artificial dissipation in the boundary layer is reduced by scaling with the ratio of the local and

free-stream Mach number, i.e. decreasing the smoothing properties of the time-stepping method, a converged solution (engineering accuracy) could be obtained by increasing the number of pre- and post-relaxations. The convergence behavior of this calculation during the FAS stage is shown in figure 6, where the discrete L_2-norm of the residual of the density is plotted as a function of the number of W-cycles. In the blocks outside the boundary layers and wakes the Euler equations are solved instead of the Navier-Stokes equations. The good agreement with the

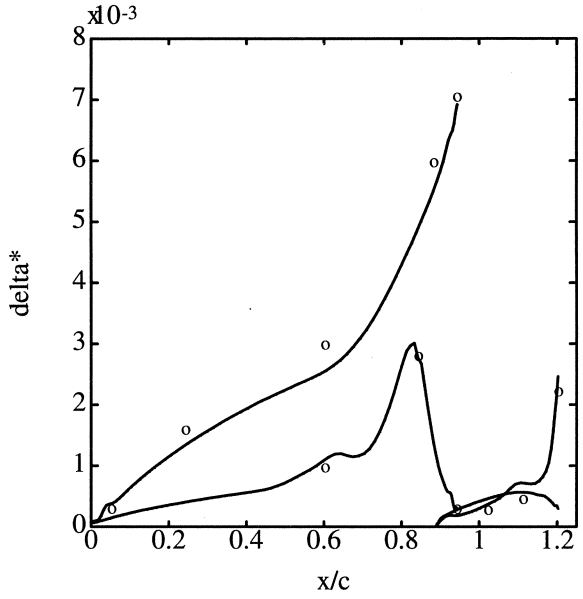


FIGURE 7. Comparison between the calculated (solid) and experimental (circles) boundary layer thickness for viscous flow at an angle of 6.0°.

wind-tunnel measurements can be inferred from figure 6 as well, where the experimental and numerically predicted pressure coefficient on the airfoil and flap are shown. The comparison between the measured and numerically predicted boundary layer thickness is shown in figure 7. The boundary layer thickness is a sensitive measure for the assertion that the artificial dissipation in the boundary layer does not outweigh the physical dissipation. The good agreement shown in figure 7 indicates that the scaling of the artificial dissipation with the local Mach number leads to an accurate representation of the flow in the regions close to the airfoil and flap.

This solution was obtained with the block loop inside the stage loop of the five-stage Runge-Kutta time-stepping method. Hence, the variables at the dummy vertices outside a block are updated after every stage, which implies that the effects of the multiblock structure on the convergence are kept to a minimum. The

frequency of data transfer between the blocks makes this method less efficient for parallel processing. However, with the block loop outside the stage loop, i.e. with an update of the dummy variables only after five flux evaluations, a converged solution could not be obtained. Apparently, the interval between two moments of data transfer between the blocks has to be sufficiently small in order to obtain a convergent multigrid method.

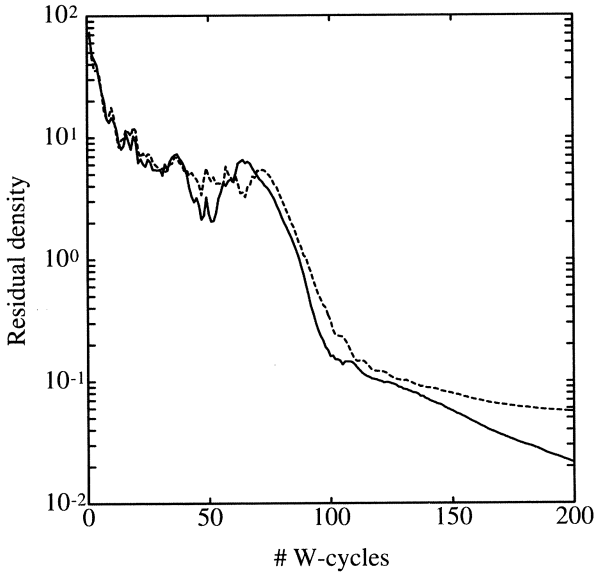


FIGURE 8. Convergence behavior of the three-stage Runge-Kutta scheme for turbulent flow; comparison between block loop inside (solid) and outside (dashed) stage loop

Further evidence for this statement is obtained from calculations with a three-stage instead of a five-stage Runge-Kutta time-stepping method. If the block loop is outside the stage loop, the dummy variables are updated after three flux evaluations. Although the rate of convergence is lower than in the case with the loops interchanged (see figure 8), the solution has converged within engineering accuracy after $\approx$ 200 W-cycles. A comparison of the three-stage and five-stage schemes with the block loop inside the stage loop shows that the five-stage scheme is more efficient: about 60 W-cycles suffice to get the residuals at the same level as with the three-stage scheme after 200 W-cycles. The five-stage scheme leads to a reduction in calculation time of approximately 60% in this instance.

4 Discussion

We presented simulation results obtained with a multigrid multiblock method for a two-element airfoil. Both viscous and inviscid calculations were performed using the same multigrid process and the same vertex-based spatial discretization method. Moreover, either a three- or a five-stage Runge-Kutta scheme was considered for the integration in time and the smoothing properties of this relaxation method were further enhanced through the introduction of local time-stepping, implicit residual averaging in which the residuals at the block boundaries were kept fixed to their non-smoothed values.

The inviscid calculations have shown that a solution which is converged up to machine accuracy can be obtained with this multigrid method. A comparison with the single grid simulation method shows that a considerable reduction in calculation time was obtained with the multigrid method, although the convergence of the single grid method for inviscid calculations was already quite acceptable. We also investigated two different numerical boundary conditions at the solid walls. It appeared that linear extrapolation of the pressure not only leads to a better convergence than constant extrapolation, but also gives rise to a much smaller entropy layer around the airfoil. The resulting drag coefficient, which theoretically should equal zero in this subsonic flow, is reduced by almost 60%.

In the viscous calculations the single grid method was found to yield a well converged result in the 6°-case, however, the convergence towards the steady state solution was extremely slow and makes the use of a multigrid approach essential. A comparison of the computational effort required in both methods shows that a total reduction with a factor of about 12.5 can be reached with the multigrid method. The calculation time is reduced by a factor of about 8. The difference in these two measures for the speed-up due to the multigrid method can be attributed to the strongly reduced vector-lengths associated with the coarser grid calculations reducing the effectiveness of processing the method on a vector computer. The numerical predictions obtained for the lift- and pressure coefficients compare well with experimental results and give confidence in the use of the Baldwin-Lomax model for this application. Moreover, the boundary layer thickness was found to agree well with the experiments. The convergence of the multigrid process was studied in detail, showing that the ordering of the various loops in the process has a considerable effect. Interchanging the block- and stage loops, keeping the grid loop as the outer loop yields an optimal convergence when the block loop is put inside the stage loop. If the stage loop is put inside the block loop then convergence of the multigrid process was absent when using the five-stage Runge-Kutta method as the relaxation method. Apparently, the smoothing of the relaxation method becomes less effective as the number of stages between two 'updates' of the dummy-variables increases. This result has some less favorable consequences in view of a possible parallel processing of the multigrid method. On the one hand parallel processing seems more efficient if the frequency of data transfer between the blocks can be reduced. On the other hand the reduction of this frequency results in a reduction

of the convergence rate of the multigrid process, and in some instances even to an absence of convergence. This suggests that in a possible parallel processing of this multigrid method, an optimal rate of data-exchange between the blocks should be determined.

ACKNOWLEDGEMENT

The authors are greatly indebted to Frans Brandsma for several useful and stimulating discussions and to Bart van Esch for invaluable support with the postprocessing of the results.

References

[1] F.J. Brandsma, M.E.S. Vogels, J. van der Vooren, D. Dijkstra and J.G.M. Kuerten, 'Pre-design document of the ISNaS compressible flow simulator', ISNaS 88.04.027 (1988)

[2] D.J. Mavriplis, 'Turbulent Flow Calculations using Unstructured and Adaptive Meshes', Proceedings of the 12th International Conference on Numerical Methods in Fluid Dynamics, Oxford 9-13 July (1990), pp. 228-232

[3] B.J. Geurts and H. Kuerten, 'Numerical Aspects of a Block Structured Compressible Flow Solver', to appear in J. Engg. Math. (1993)

[4] B. Baldwin and H. Lomax, 'Thin layer approximation and algebraic model for separated turbulent flow', AIAA-78-257 (1978)

[5] J.G.M. Kuerten, B.J. Geurts, J.W. van der Burg, A.W. Vreman and P.J. Zandbergen, 'Development and applications of a 3-D compressible Navier-Stokes solver', to appear in Proceedings of the 13th International Conference on Numerical methods in Fluid Dynamics, Rome 6-10 July (1992)

[6] R. Radespiel, 'A cell-vertex multigrid method for the Navier-Stokes equations', NASA Technical Memorandum 101557 (1989)

[7] B. van den Berg, 'Boundary layer measurements on a two-dimensional wing with flap', NLR TR 79009 U (1979)

12

A Study on Narrow Stencil Discretizations and Multigrid Solvers for the Generalized Stokes Equations

Yvonne Luh[1]

ABSTRACT In this study, the generalized Stokes equations (which can be considered as linearization of the steady state incompressible Navier Stokes equations) are chosen as model problem for fluid flow. The discretization of these equations is based on a flux difference splitting concept which, in this case, coincides with the Osher scheme. Depending on the definition of interior and exterior states at the interfaces of the control volumes, various discretizations of narrow-stencil type are derived. Ultimately aiming at a direction-independent smoother, pointwise Gauss-Seidel relaxation with lexicographic ordering of gridpoints and alternating line relaxation (with relaxation parameters $\omega = 1.0$ and $\omega = 0.75$) are used in a multigrid context. In case of 2nd order discretizations, we solve the resulting system of discrete equations via defect correction.
As a result we show that the convergence rates for the standard and for many narrow-stencil cases are very similar. With regard to accuracy, the discretization error of most narrow-stencil-discretizations proves to be of about at least a factor 2 better than in the standard case.

1 Equations under Study: the Generalized Stokes Equations

In the sequel, the generalized Stokes equations

$$\left\{\begin{array}{rcrcrcl} -\frac{1}{Re}\Delta u & + & a\,u_x & + & b\,u_y & + & p_x = f_1 \\ -\frac{1}{Re}\Delta v & + & a\,v_x & + & b\,v_y & + & p_y = f_2 \\ & & & & u_x & + & v_y = f_3 \end{array}\right\} \qquad (1)$$

(which can be considered as linearization of the steady state incompressible Navier Stokes equations) are chosen as a model problem for the systems of partial differ-

[1]Gesellschaft für Mathematik und Datenverarbeitung
Postfach 1316, 53731 Sankt Augustin, Germany

ential equations governing fluid flow.

Here, the velocities in x- and y- direction are denoted by u and v, the unknown p stands for pressure, and Re denotes the Reynolds number. In the following, we choose $a, b = const$ and $\frac{1}{Re} = 0^+$. The problem under consideration will be entering flow.

2 Discretization

Several discretizations and solvers for the generalized Stokes equations (1) are under consideration. The approach by Brandt/Yavneh [BY91] is characterized by the use of staggered grids, by a narrow-stencil-discretization of the advection operator (1st and 2nd order), by Gauss Seidel relaxation of the momentum equations *in streamwise direction* and by distributive relaxation of the continuity equation.

Several methods have been proposed based on non-staggered grids, e.g. flux difference splitting of Roe-type [DL89], the Osher scheme [SPE88] and approaches that are based on the addition of stabilizing terms to the continuity equation, e.g.

$$\partial_x^{h,c} u \; + \; \partial_y^{h,c} v \; - \; \omega h^2 \Delta^h p^h \; = \; 0.$$

As will be seen soon, in the case of the generalized Stokes equations, all these methods coincide.

The generalized Stokes equations (1) can be written in conservative formulation as

$$\frac{\partial \vec{f}}{\partial x} + \frac{\partial \vec{g}}{\partial y} = \frac{\partial \vec{f}_v}{\partial x} + \frac{\partial \vec{g}_v}{\partial y},$$

where $\vec{f}$ and $\vec{g}$ represent the convective fluxes, while $\vec{f}_v$ and $\vec{g}_v$ stand for the viscous parts.[2] Here,

$$\vec{q} = \begin{pmatrix} u \\ v \\ p \end{pmatrix}, \qquad \vec{f}(\vec{q}) = \begin{pmatrix} au + p \\ av \\ u \end{pmatrix}, \qquad \vec{g}(\vec{q}) = \begin{pmatrix} bu \\ bv + p \\ v \end{pmatrix}.$$

The convective flux in direction of (nx, ny) then is

$$\vec{F}(\vec{q}) \; := \; nx \cdot \vec{f}(\vec{q}) \; + \; ny \cdot \vec{g}(\vec{q}),$$

and the Jacobian reads

$$A(\vec{q}) \; := \; nx \cdot \frac{\partial \vec{f}}{\partial \vec{q}} \; + \; ny \cdot \frac{\partial \vec{g}}{\partial \vec{q}}.$$

[2] In the sequel, the viscous terms will be treated by a standard technique based on Peyret control volumes, and, as $\frac{1}{Re} = 0^+$, be neglected.

Let λ_i, $i = 1,2,3$ be the eigenvalues of matrix A, and R and L transformation matrices composed by the right and left eigenvectors such that

$$A = R\Lambda L \qquad \Lambda := diag(\lambda_i, i = 1,2,3).$$

Then, the matrix A can be split into matrices with positive or negative eigenvalues

$$A^+ := R\Lambda^+ L, \quad A^- := R\Lambda^- L, \quad A = A^+ + A^-,$$

where

$$\left\{ \begin{array}{lcl} \Lambda^+ & := & diag(\max(0,\lambda_i), \ i = 1,2,3) \\ \Lambda^- & := & diag(\min(0,\lambda_i), \ i = 1,2,3) \end{array} \right\}.$$

Analogously,

$$|A| := R|\Lambda|L, \qquad \text{for} \qquad |\Lambda| := \Lambda^+ - \Lambda^-.$$

By the above splitting of matrix A, a similar splitting of the flux $\vec{F}$ is induced:

$$\vec{F}(\vec{q}) = \vec{F}^+(\vec{q}) + \vec{F}^-(\vec{q})$$

where

$$\frac{\partial \vec{F}^+}{\partial \vec{q}} = A^+, \quad \frac{\partial \vec{F}^-}{\partial \vec{q}} = A^-.$$

In order to derive a (conservative) finite volume discretization, we consider the following general quadrilateral (i.e. logically rectangular) grid (cf. figure 1). Here $\Omega_{i,j}$ denotes the control volume at point (i,j), $\partial\Omega_{i,j} = \partial\Omega_{i+1/2} \cup \partial\Omega_{j+1/2} \cup \partial\Omega_{i-1/2} \cup \partial\Omega_{j-1/2}$ the boundary of the control volume where the fluxes have to be evaluated, $\partial\Omega_{i+1/2}$ etc. standing for the individual interfaces.

We assume that at each interface (e.g. $\partial\Omega_{i+1/2}$) the flux $\vec{F}$ is constant, depending only on a constant left (=inner) and right (=outer) state ($\vec{q}^{\,l}_{i+1/2}$ and $\vec{q}^{\,r}_{i+1/2}$ respectively). Thus, on each interface of the control volume, a locally one-dimensional Riemann problem (in direction of the outward unit normal (nx, ny)) depending on $\vec{q}^{\,l}$ and $\vec{q}^{\,r}$ has to be solved.

A natural approximate Riemann solver ("numerical flux function") is

$$\vec{F}_R(\vec{q}^{\,l}, \vec{q}^{\,r}) = \vec{F}^+(\vec{q}^{\,l}) + \vec{F}^-(\vec{q}^{\,r}),$$

from which

$$\vec{F}_R(\vec{q}^{\,l}, \vec{q}^{\,r}) = \frac{1}{2}\left(\vec{F}(\vec{q}^{\,l}) + \vec{F}(\vec{q}^{\,r}) - \int_{\vec{q}^{\,l}}^{\vec{q}^{\,r}} |A(\vec{q})| d\vec{q}\right)$$

can be derived.[3] In the case of the generalized Stokes equations, where the matrix A does not depend on $\vec{q}$, we obtain

$$\vec{F}_R(\vec{q}^{\,l}, \vec{q}^{\,r}) = \frac{1}{2}\left(\vec{F}(\vec{q}^{\,l}) + \vec{F}(\vec{q}^{\,r}) - |A|(\vec{q}^{\,l} - \vec{q}^{\,r})\right).$$

[3] e.g. [SPE88, p37ff].

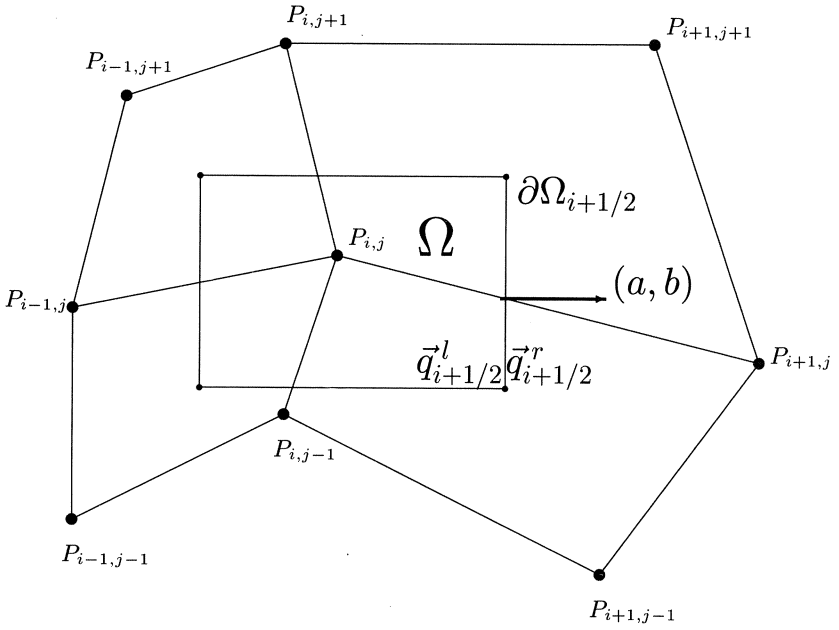


FIGURE 1. Underlying general grid ("quadrilaterals")

Balancing fluxes over all interfaces of the control volume, the homogeneous equations read

$$\begin{array}{rcccccl} & \frac{1}{2}\{ & \vec{F}(\vec{q}^{\,r}_{i+1/2}) & + & \vec{F}(\vec{q}^{\,l}_{i+1/2}) & - & |A|_{i+1/2}\left(\vec{q}^{\,r}_{i+1/2}-\vec{q}^{\,l}_{i+1/2}\right) \quad \} \cdot l_{i+1/2} \\ + & \frac{1}{2}\{ & \vec{F}(\vec{q}^{\,r}_{j+1/2}) & + & \vec{F}(\vec{q}^{\,l}_{j+1/2}) & - & |A|_{j+1/2}\left(\vec{q}^{\,r}_{j+1/2}-\vec{q}^{\,l}_{j+1/2}\right) \quad \} \cdot l_{j+1/2} \\ + & \frac{1}{2}\{ & \vec{F}(\vec{q}^{\,r}_{i-1/2}) & + & \vec{F}(\vec{q}^{\,l}_{i-1/2}) & - & |A|_{i-1/2}\left(\vec{q}^{\,r}_{i-1/2}-\vec{q}^{\,l}_{i-1/2}\right) \quad \} \cdot l_{i-1/2} \\ + & \frac{1}{2}\{ & \vec{F}(\vec{q}^{\,r}_{j-1/2}) & + & \vec{F}(\vec{q}^{\,l}_{j-1/2}) & - & |A|_{j-1/2}\left(\vec{q}^{\,r}_{j-1/2}-\vec{q}^{\,l}_{j-1/2}\right) \quad \} \cdot l_{j-1/2} \\ = & 0, & & & & & \end{array} \tag{2}$$

Here, $l_{i+1/2}$ denotes the length of the interface $\partial\Omega_{i+1/2}$.

In case of an equidistant Cartesian grid and constant a and b, equation (2)

leads to

$$
\begin{array}{lll}
 & \frac{1}{2}a & \left(u^r_{i+\frac{1}{2}} + u^l_{i+\frac{1}{2}} - u^r_{i-\frac{1}{2}} - u^l_{i-\frac{1}{2}}\right) - \frac{1}{2}\frac{a^2+2}{\sqrt{a^2+4}}\left(u^r_{i+\frac{1}{2}} - u^l_{i+\frac{1}{2}} + u^r_{i-\frac{1}{2}} - u^l_{i-\frac{1}{2}}\right) \\
+ & \frac{1}{2}b & \left(u^r_{j+\frac{1}{2}} + u^l_{j+\frac{1}{2}} - u^r_{j-\frac{1}{2}} - u^l_{j-\frac{1}{2}}\right) - \frac{1}{2}|b| \left(u^r_{j+\frac{1}{2}} - u^l_{j+\frac{1}{2}} + u^r_{j-\frac{1}{2}} - u^l_{j-\frac{1}{2}}\right) \\
+ & \frac{1}{2} & \left(p^r_{i+\frac{1}{2}} + p^l_{i+\frac{1}{2}} - p^r_{i-\frac{1}{2}} - p^l_{i-\frac{1}{2}}\right) - \frac{1}{2}\frac{a}{\sqrt{a^2+4}}\left(p^r_{i+\frac{1}{2}} - p^l_{i+\frac{1}{2}} + p^r_{i-\frac{1}{2}} - p^l_{i-\frac{1}{2}}\right) \\
= & 0 &
\end{array}
$$

$$
\begin{array}{lll}
 & \frac{1}{2}a & \left(v^r_{i+\frac{1}{2}} + v^l_{i+\frac{1}{2}} - v^r_{i-\frac{1}{2}} - v^l_{i-\frac{1}{2}}\right) - \frac{1}{2}|a| \left(v^r_{i+\frac{1}{2}} - v^l_{i+\frac{1}{2}} + v^r_{i-\frac{1}{2}} - v^l_{i-\frac{1}{2}}\right) \\
+ & \frac{1}{2}b & \left(v^r_{j+\frac{1}{2}} + v^l_{j+\frac{1}{2}} - v^r_{j-\frac{1}{2}} - v^l_{j-\frac{1}{2}}\right) - \frac{1}{2}\frac{b^2+2}{\sqrt{b^2+4}}\left(v^r_{j+\frac{1}{2}} - v^l_{j+\frac{1}{2}} + v^r_{j-\frac{1}{2}} - v^l_{j-\frac{1}{2}}\right) \\
+ & \frac{1}{2} & \left(p^r_{j+\frac{1}{2}} + p^l_{j+\frac{1}{2}} - p^r_{j-\frac{1}{2}} - p^l_{j-\frac{1}{2}}\right) - \frac{1}{2}\frac{b}{\sqrt{b^2+4}}\left(p^r_{j+\frac{1}{2}} - p^l_{j+\frac{1}{2}} + p^r_{j-\frac{1}{2}} - p^l_{j-\frac{1}{2}}\right) \\
= & 0 &
\end{array}
$$

$$
\begin{array}{lll}
 & \frac{1}{2} & \left(u^r_{i+\frac{1}{2}} + u^l_{i+\frac{1}{2}} - u^r_{i-\frac{1}{2}} - u^l_{i-\frac{1}{2}}\right) - \frac{1}{2}\frac{a}{\sqrt{a^2+4}}\left(u^r_{i+\frac{1}{2}} - u^l_{i+\frac{1}{2}} + u^r_{i-\frac{1}{2}} - u^l_{i-\frac{1}{2}}\right) \\
+ & \frac{1}{2} & \left(v^r_{j+\frac{1}{2}} + v^l_{j+\frac{1}{2}} - v^r_{j-\frac{1}{2}} - v^l_{j-\frac{1}{2}}\right) - \frac{1}{2}\frac{b}{\sqrt{b^2+4}}\left(v^r_{j+\frac{1}{2}} - v^l_{j+\frac{1}{2}} + v^r_{j-\frac{1}{2}} - v^l_{j-\frac{1}{2}}\right) \\
- & \frac{1}{\sqrt{a^2+4}}\left(p^r_{i+\frac{1}{2}} - p^l_{i+\frac{1}{2}} + p^r_{i-\frac{1}{2}} - p^l_{i-\frac{1}{2}}\right) - \frac{1}{\sqrt{b^2+4}}\left(p^r_{j+\frac{1}{2}} - p^l_{j+\frac{1}{2}} + p^r_{j-\frac{1}{2}} - p^l_{j-\frac{1}{2}}\right) \\
= & 0. &
\end{array}
$$

Assuming $a \geq b \geq 0$ and using the abbreviations

$$
\left\{
\begin{array}{lll}
\epsilon_a := \frac{a\sqrt{a^2+4}-(a^2+2)}{2\sqrt{a^2+4}} & & \delta_a := \frac{1}{\sqrt{a^2+4}} \\
 & \text{and} & \\
\epsilon_b := \frac{b\sqrt{b^2+4}-(b^2+2)}{2\sqrt{b^2+4}} & & \delta_b := \frac{1}{\sqrt{b^2+4}}
\end{array}
\right\},
$$

we obtain the general discretization formula displayed in figure 2. Here, the rows correspond to the different equations of the generalized Stokes system, and the columns are related to the unknowns u, v and p respectively. It can be seen that, by the underlying splitting approach, stabilizing terms with coefficients ϵ_a, ϵ_b, δ_a and δ_b have been introduced into the discretization.

In the sequel, the general discretization formula displayed in figure 2 will be the starting point from which, depending on the definition of left and right states at the interfaces, different discretizations (mostly based on narrow-stencil ideas) will be derived.

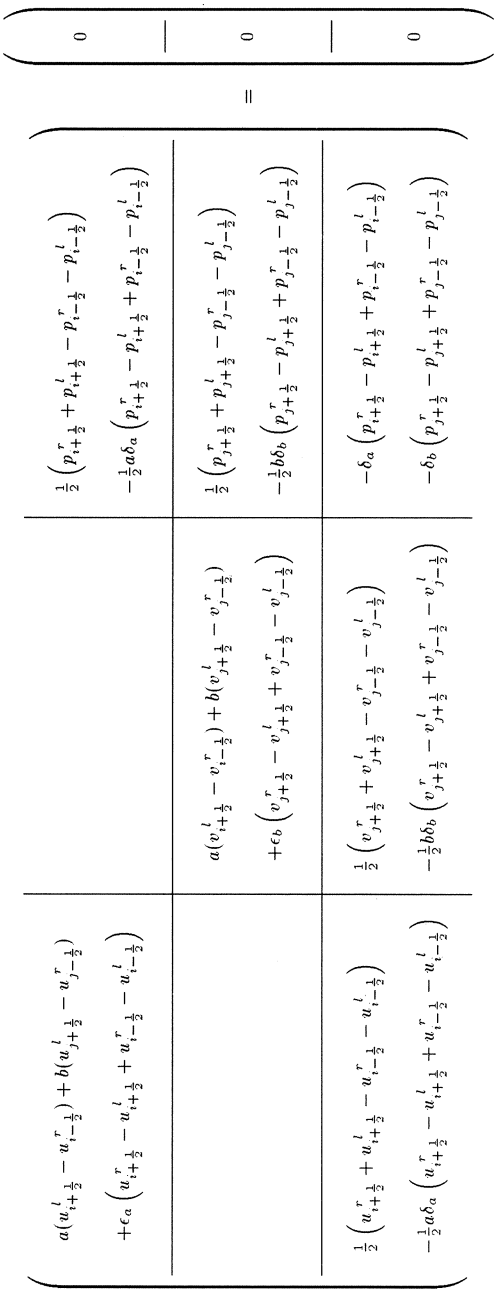


FIGURE 2. Starting point for splitting-type discretizations of the generalized Stokes equations

3 Various 1st order discretizations

3.1 Standard-Upstream-Discretization

The simplest definition of the inner and outer state at the interfaces, i.e. of $\vec{q}^{\,l}$ and $\vec{q}^{\,r}$, is based on the assumption of piecewise constant states in all control volumes:

$$\left\{ \begin{array}{lcl} \vec{q}^{\,r}_{i+\frac{1}{2}} &=& \vec{q}_{i+1,j} \\ \vec{q}^{\,r}_{j+\frac{1}{2}} &=& \vec{q}_{i,j+1} \\ \vec{q}^{\,r}_{i-\frac{1}{2}} &=& \vec{q}_{i-1,j} \\ \vec{q}^{\,r}_{j-\frac{1}{2}} &=& \vec{q}_{i,j-1} \end{array} \right. \quad \text{and} \quad \left. \begin{array}{lcl} \vec{q}^{\,l}_{i+\frac{1}{2}} &=& \vec{q}_{i,j} \\ \vec{q}^{\,l}_{j+\frac{1}{2}} &=& \vec{q}_{i,j} \\ \vec{q}^{\,l}_{i-\frac{1}{2}} &=& \vec{q}_{i,j} \\ \vec{q}^{\,l}_{j-\frac{1}{2}} &=& \vec{q}_{i,j} \end{array} \right\}.$$

This leads to the following discretization:

$$\left(\begin{array}{c|c|c} (a\partial_x + b\partial_y)|^{su} & 0 & \partial^c_x \\ +\epsilon_a \partial^c_{xx} & & -\frac{1}{2} a \delta_a \partial^c_{xx} \\ \hline 0 & (a\partial_x + b\partial_y)|^{su} & \partial^c_y \\ & +\epsilon_b \partial^c_{yy} & -\frac{1}{2} b \delta_b \partial^c_{yy} \\ \hline \partial^c_x & \partial^c_y & -\delta_a \partial^c_{xx} \\ -\frac{1}{2} a \delta_a \partial^c_{xx} & -\frac{1}{2} b \delta_b \partial^c_{yy} & -\delta_b \partial^c_{yy} \end{array} \right). \tag{3}$$

Again, the rows of the above matrix correspond to the three equations to be discretized, whereas the columns are related to the unknows u, v and p. Here,

$$\partial^c_x := \tfrac{1}{2}[-1 \ \underline{0} \ 1] \qquad \partial^c_{xx} := [1 \ \underline{-2} \ 1]$$

$$\partial^c_y := \tfrac{1}{2} \begin{bmatrix} 1 \\ \underline{0} \\ -1 \end{bmatrix} \qquad \partial^c_{yy} := \begin{bmatrix} 1 \\ \underline{-2} \\ 1 \end{bmatrix}$$

and

$$(a\partial_x + b\partial_y)|^{su} := \begin{bmatrix} 0 & 0 & 0 \\ -a & \underline{a+b} & 0 \\ 0 & -b & 0 \end{bmatrix}.$$

Note that for the first entry in (3)

$$\begin{array}{rcl} \text{Star(1,1)} &=& a\partial^c_x + b\partial^c_y - \frac{a^2+2}{\sqrt{a^2+4}} \partial^c_{xx} - \frac{1}{2}|b|\partial^c_{yy} \\ &=& \underbrace{(a\partial_x + b\partial_y)|^{su}}_{\substack{\text{advection} \\ \text{part}}} + \underbrace{\epsilon_a \partial^c_{xx}}_{\substack{\text{stabilizing} \\ \text{term}}}. \end{array}$$

3.2 Narrow-Stencil-Discretization

The standard definition of inner and outer states at the interfaces, i.e. of $\vec{q}^{\,l}$ and $\vec{q}^{\,r}$, does not take into account the direction (a, b) of the flow. In contrast to this,

narrow-stencil-discretizations can be derived by constructing the intersection point $Q^r_{i+1/2}$ (and thus $\vec{q}^r_{i+1/2}$) of the stream vector (a, b) outside the control volume $\Omega_{i,j}$ with the grid given by the points $P_{i,j}$, $P_{i,j-1}$, $P_{i+1,j-1}$, $P_{i+1,j}$, $P_{i+1,j+1}$ and $P_{i,j+1}$ (cf. Figure 3).

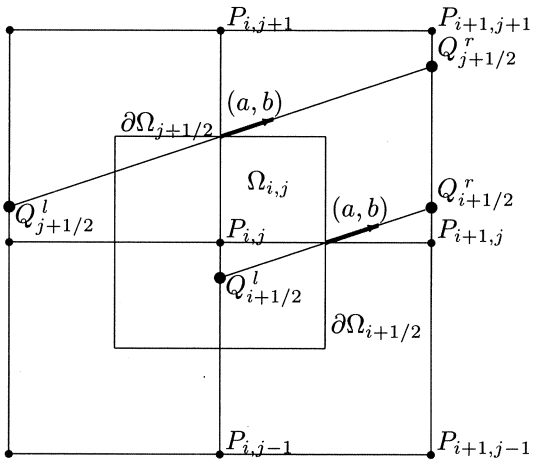


FIGURE 3. Derivation of narrow-stencil-discretization

Approximating $Q^r_{i+1/2}$ (and thus $\vec{q}^r_{i+1/2}$) by the nearest grid point leads to

$$\left\{ \begin{array}{lcl} \vec{q}^r_{i+\frac{1}{2}} = \vec{q}_{i+1,j} & & \vec{q}^l_{i+\frac{1}{2}} = \vec{q}_{i,j} \\ \vec{q}^r_{j+\frac{1}{2}} = \vec{q}_{i+1,j+1} & & \vec{q}^l_{j+\frac{1}{2}} = \vec{q}_{i-1,j} \\ \vec{q}^r_{i-\frac{1}{2}} = \vec{q}_{i-1,j} & \text{and} & \vec{q}^l_{i-\frac{1}{2}} = \vec{q}_{i,j} \\ \vec{q}^r_{j-\frac{1}{2}} = \vec{q}_{i-1,j-1} & & \vec{q}^l_{j-\frac{1}{2}} = \vec{q}_{i+1,j} \end{array} \right\}.$$

The resulting narrow-stencil-discretization of the generalized Stokes equations then reads

$$\left(\begin{array}{c|c|c} (a\partial_x + b\partial_y)|^{ns} & 0 & \partial^c_x \\ +\epsilon_a \partial^c_{xx} & & -\frac{1}{2} a\delta_a \partial^c_{xx} \\ \hline 0 & (a\partial_x + b\partial_y)|^{ns} & \partial^{ns}_y \\ & +\epsilon_b \partial^{ns}_{yy} & -\frac{1}{2} b\delta_b \partial^{ns}_{yy} \\ \hline \partial^c_x & \partial^{ns}_y & -\delta_a \partial^c_{xx} \\ -\frac{1}{2} a\delta_a \partial^c_{xx} & -\frac{1}{2} b\delta_b \partial^{ns}_{yy} & -\delta_b \partial^{ns}_{yy} \end{array} \right).$$

$$\text{Here, } \partial_y^{ns} := \frac{1}{2}\begin{bmatrix} & & 1 \\ 1 & \underline{0} & -1 \\ -1 & & \end{bmatrix}, \partial_{yy}^{ns} := \begin{bmatrix} & & 1 \\ -1 & \underline{0} & -1 \\ 1 & & \end{bmatrix} \text{ and}$$

$$(a\partial_x + \partial_y)|^{ns} := \begin{bmatrix} 0 & 0 & 0 \\ -a+b & \underline{a} & 0 \\ -b & 0 & 0 \end{bmatrix}.$$

3.3 Skew-Upwind-Discretization

Instead of approximating $Q^r_{i+1/2}$ by the nearest grid point, a linear interpolation involving both neighbouring grid points leads to

$$\left\{\begin{array}{lcl}
\vec{q}^{\,r}_{i+\frac{1}{2}} & = & \left(1-\frac{1}{2}\frac{b}{a}\right)\vec{q}_{i+1,j} + \frac{1}{2}\frac{b}{a}\vec{q}_{i+1,j+1} \\
\vec{q}^{\,r}_{j+\frac{1}{2}} & = & \left(\frac{1}{2}+\frac{b}{a}\right)\vec{q}_{i+1,j+1} + \left(\frac{1}{2}-\frac{b}{a}\right)\vec{q}_{i+1,j} \\
\vec{q}^{\,r}_{i-\frac{1}{2}} & = & \left(1-\frac{1}{2}\frac{b}{a}\right)\vec{q}_{i-1,j} + \frac{1}{2}\frac{b}{a}\vec{q}_{i-1,j-1} \\
\vec{q}^{\,r}_{j-\frac{1}{2}} & = & \left(\frac{1}{2}+\frac{b}{a}\right)\vec{q}_{i-1,j-1} + \left(\frac{1}{2}-\frac{b}{a}\right)\vec{q}_{i-1,j} \\
& & \text{and} \\
\vec{q}^{\,l}_{i+\frac{1}{2}} & = & \left(1-\frac{1}{2}\frac{b}{a}\right)\vec{q}_{i,j} + \frac{1}{2}\frac{b}{a}\vec{q}_{i,j-1} \\
\vec{q}^{\,l}_{j+\frac{1}{2}} & = & \left(\frac{1}{2}+\frac{b}{a}\right)\vec{q}_{i-1,j} + \left(\frac{1}{2}-\frac{b}{a}\right)\vec{q}_{i-1,j+1} \\
\vec{q}^{\,l}_{i-\frac{1}{2}} & = & \left(1-\frac{1}{2}\frac{b}{a}\right)\vec{q}_{i,j} + \frac{1}{2}\frac{b}{a}\vec{q}_{i,j+1} \\
\vec{q}^{\,l}_{j-\frac{1}{2}} & = & \left(\frac{1}{2}+\frac{b}{a}\right)\vec{q}_{i+1,j} + \left(\frac{1}{2}-\frac{b}{a}\right)\vec{q}_{i+1,j-1}
\end{array}\right\}.$$

The resulting discretization of the generalized Stokes equations is characterized by a skew-upwind-discretization [RAI76] of the advection part.

3.4 Variants

Several variants of narrow-stencil-discretizations (which, in the sequel will be called variant 1, variant 2 and variant 3)[4] can be derived in an analogous way.

3.5 Modifications

Starting from the standard-upstream-discretization of the generalized Stokes equations, another class of narrow-stencil-discretizations can be derived by substituting solely the discretization of the advection part, $(a\partial_x + b\partial_y)|^{su}$, by a narrow-stencil variant. These discretizations will be called narrow-stencil MOD1, skew-upwind

[4] see [LUH92] and [HK91, (3.10)]

MOD1 etc. Narrow-stencil MOD1 reads

$$\left(\begin{array}{c|c|c} (a\partial_x + b\partial_y)|^{ns} & 0 & \partial_x^c \\ +\epsilon_a \partial_{xx}^c & & -\frac{1}{2} a\delta_a \partial_{xx}^c \\ \hline 0 & (a\partial_x + b\partial_y)|^{ns} & \partial_y^c \\ & +\epsilon_b \partial_{yy}^c & -\frac{1}{2} b\delta_b \partial_{yy}^c \\ \hline \partial_x^c & \partial_y^c & -\delta_a \partial_{xx}^c \\ -\frac{1}{2} a\delta_a \partial_{xx}^c & -\frac{1}{2} b\delta_b \partial_{yy}^c & -\delta_b \partial_{yy}^c \end{array}\right).$$

Another class of modifications (MOD2) can be obtained by applying narrow-stencil ideas to the velocities u and v only, whereas the pressure p is treated in a standard way.

Thus, narrow-stencil MOD2 reads

$$\left(\begin{array}{c|c|c} (a\partial_x + b\partial_y)|^{ns} & 0 & \partial_x^c \\ +\epsilon_a \partial_{xx}^c & & -\frac{1}{2} a\delta_a \partial_{xx}^c \\ \hline 0 & (a\partial_x + b\partial_y)|^{ns} & \partial_y^c \\ & +\epsilon_b \partial_{yy}^{ns} & -\frac{1}{2} b\delta_b \partial_{yy}^c \\ \hline \partial_x^c & \partial_y^{ns} & -\delta_a \partial_{xx}^c \\ -\frac{1}{2} a\delta_a \partial_{xx}^c & -\frac{1}{2} b\delta_b \partial_{yy}^{ns} & -\delta_b \partial_{yy}^c \end{array}\right).$$

4 Computational example

Results are presented for an equidistant Cartesian grid (of step size h) on $\Omega = (0,1)^2$ with Dirichlet boundary conditions. The Reynolds number is $Re = 10000$.

The exact solution of the inviscid system is forced to be

$$\left\{\begin{array}{lcl} u & = & \sin\left[2\pi\left(y - \frac{b}{a}x\right)\right] \\ v & = & \sin\left[2\pi\left(y - \frac{b}{a}x\right)\right] \\ p & = & 10 \end{array}\right\},$$

the alignment being fixed at $a = 1$, $b = \tan(22.5°) \approx 0.4142$.

A multigrid iteration characterized by W(1,1)-cycles, full weighting and bilinear interpolation is used for solving the discrete systems. The calculation starts from zero initial values for all dependent unknowns on the finest grid. The coarsest grid consists of 5x5 points.[5]

As a smoother, pointwise Gauss-Seidel relaxation in lexicographic ordering and alternating line relaxation (with relaxation parameters $\omega = 1.00$ and $\omega = 0.75$) are tested.[6]

---

[5] The numerical experiments are based on the program LiSS [LS91].

[6] for standard discretizations and local mode analysis see [FS92]

5 Numerical experiments concerning 1st order discretizations

5.1 Accuracy

Table 1 displays the discretization error of various 1st order discretizations of the generalized Stokes equations for different step sizes h. As in some cases (Narrow-stencil, skew-upwind, var 2, skew-upwind MOD2, var 1 MOD2) all multigrid algorithms under consideration diverge, table 1 is not complete.

The accuracy of any narrow-stencil-discretization is better than in the standard case, at least by a factor of 2. The MOD1-variants in general are less accurate than MOD2.

5.2 Convergence

Tables 2, 3 and 4 show the convergence rates of the different multigrid algorithms for the different discretizations. Generally, alternating line relaxation with $\omega = 0.75$ leads to best results.

6 Various 2nd order discretizations

6.1 Standard-Upstream-Discretization

The simplest definition of inner and outer states at the interfaces leading to a 2nd order discretization is based on the assumption of piecewise linear states in the control volumes:

$$\left\{ \begin{array}{lll} \vec{q}^{\,r}_{i+\frac{1}{2}} = \frac{3}{2}\vec{q}_{i+1,j} - \frac{1}{2}\vec{q}_{i+2,j} & & \vec{q}^{\,l}_{i+\frac{1}{2}} = \frac{3}{2}\vec{q}_{i,j} - \frac{1}{2}\vec{q}_{i-1,j} \\ \vec{q}^{\,r}_{j+\frac{1}{2}} = \frac{3}{2}\vec{q}_{i,j+1} - \frac{1}{2}\vec{q}_{i,j+2} & & \vec{q}^{\,l}_{j+\frac{1}{2}} = \frac{3}{2}\vec{q}_{i,j} - \frac{1}{2}\vec{q}_{i,j-1} \\ \vec{q}^{\,r}_{i-\frac{1}{2}} = \frac{3}{2}\vec{q}_{i-1,j} - \frac{1}{2}\vec{q}_{i-2,j} & \text{and} & \vec{q}^{\,l}_{i-\frac{1}{2}} = \frac{3}{2}\vec{q}_{i,j} - \frac{1}{2}\vec{q}_{i+1,j} \\ \vec{q}^{\,r}_{j-\frac{1}{2}} = \frac{3}{2}\vec{q}_{i,j-1} - \frac{1}{2}\vec{q}_{i,j-2} & & \vec{q}^{\,l}_{j-\frac{1}{2}} = \frac{3}{2}\vec{q}_{i,j} - \frac{1}{2}\vec{q}_{i,j+1} \end{array} \right\}.$$

The discretization of the generalized Stokes equations then reads:

$$\left(\begin{array}{c|c|c} (a\partial_x + b\partial_y)|^{su,2} & 0 & \partial_x^{c,2} \\ +\epsilon_a \partial^c_{x^4} & & -\frac{1}{2}a\delta_a\partial^c_{x^4} \\ \hline 0 & (a\partial_x + b\partial_y)|^{su,2} & \partial_y^{c,2} \\ & +\epsilon_b\partial^c_{y^4} & -\frac{1}{2}b\delta_b\partial^c_{y^4} \\ \hline \partial_x^{c,2} & \partial_y^{c,2} & -\delta_a\partial^c_{x^4} \\ -\frac{1}{2}a\delta_a\partial^c_{x^4} & -\frac{1}{2}b\delta_b\partial^c_{y^4} & -\delta_b\partial^c_{y^4} \end{array} \right).$$

Here

$$\begin{array}{rcl} \partial_x^{c,2} & := & [\ 1/4 \quad -1 \quad \underline{0} \quad 1 \quad -1/4] \\ \partial^c_{x^4} & := & [-1/2 \quad 2 \quad \underline{-3} \quad 2 \quad -1/2] \end{array}$$

($\partial_y^{c,2}$ and $\partial_{y^4}^c$ analogously) and

$$(a\partial_x + b\partial_y)|^{su,2} := \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2}a & -2a & \frac{3}{2}(a+b) & 0 & 0 \\ 0 & 0 & -2b & 0 & 0 \\ 0 & 0 & \frac{1}{2}b & 0 & 0 \end{bmatrix}.$$

6.2 Central discretization

A central discretization can be derived by choosing identical inner and outer states at the interfaces, interpolating from the neighboring grid points

$$\left\{ \begin{array}{lll} \vec{q}^{\,r}_{i+\frac{1}{2}} = \vec{q}^{\,l}_{i+\frac{1}{2}} = \frac{1}{2}(\vec{q}_{i,j} + \vec{q}_{i+1,j}) \\ \vec{q}^{\,r}_{j+\frac{1}{2}} = \vec{q}^{\,l}_{j+\frac{1}{2}} = \frac{1}{2}(\vec{q}_{i,j} + \vec{q}_{i,j+1}) \\ \vec{q}^{\,r}_{i-\frac{1}{2}} = \vec{q}^{\,l}_{i-\frac{1}{2}} = \frac{1}{2}(\vec{q}_{i,j} + \vec{q}_{i-1,j}) \\ \vec{q}^{\,r}_{j-\frac{1}{2}} = \vec{q}^{\,l}_{j-\frac{1}{2}} = \frac{1}{2}(\vec{q}_{i,j} + \vec{q}_{i,j-1}) \end{array} \right\}.$$

In this case, the discretization reads:

$$\left(\begin{array}{c|c|c} (a\partial_x + b\partial_y)|^{c,2} & 0 & \partial_x^c \\ \hline 0 & (a\partial_x + b\partial_y)|^{c,2} & \partial_y^c \\ \hline \partial_x^c & \partial_y^c & 0 \end{array} \right),$$

where

$$(a\partial_x + b\partial_y)|^{c,2} := \begin{bmatrix} 0 & \frac{1}{2}b & 0 \\ -\frac{1}{2}a & 0 & \frac{1}{2}a \\ 0 & -\frac{1}{2}b & 0 \end{bmatrix}.$$

Here, no stabilizing terms occur.

6.3 Hybrid discretizations

An obvious combination of the 2nd order standard-upstream- ($\delta^{su,2}$) and the central scheme ($\delta^{c,2}$) is given by the van Leer scheme

$$\beta\,\delta^{su,2} + (1-\beta)\delta^{c,2}, \qquad \beta \in [0,1].$$

Narrow-stencil variants of the above and modifications MOD1 and MOD2 can also be derived, e.g. the 2nd order narrow-stencil hybrid discretization of the

advection part

$$
\begin{aligned}
(a\partial_x + b\partial_y)|^{\beta,ns} \;:=\; & \beta \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \frac{1}{2}(a-b) & -2(a-b) & \frac{3}{2}a & 0 & 0 \\ 0 & -2b & 0 & 0 & 0 \\ \frac{1}{2}b & 0 & 0 & 0 & 0 \end{bmatrix} \\
+ \; & (1-\beta) \begin{bmatrix} 0 & 0 & \frac{b}{2} \\ -\frac{a-b}{2} & 0 & \frac{a-b}{2} \\ -\frac{b}{2} & 0 & 0 \end{bmatrix}.
\end{aligned}
$$

7 Computational example

The same test case is being treated in the case of 2nd and 1st order discretizations of the generalized Stokes equations (see section 4).

As a solution procedure, defect correction is used here. The 2nd order operator L_h^2 is given by a hybrid scheme with $\beta = 1, 1/2, 1/3$, the 1st order operator L_h^1 by the standard-upstream- or narrow-stencil-discretization of 1st order.[7]

8 Numerical experiments concerning the 2nd order discretizations

8.1 Accuracy

As in the case of 1st order discretizations, the narrow-stencil variants are more accurate than the standard-upstream-discretizations (cf. tables 5 and 6).

8.2 Convergence

Defect correction for standard-upstream- and for narrow-stencil-discretizations displays similar behaviour, the convergence in the narrow-stencil case being slightly faster. Many results, obtained in the scalar advection case, can be generalized to the generalized Stokes equations (with the important exception of the $\beta = 1$ – schemes: here, convergence can be accelerated by introducing a relaxation parameter $\omega \approx 0.6$ to 0.8).

[7] Some variants concerning the boundary treatment were considered. As they did not affect the overall accuracy, they will not be discussed here.

References

[BY91] A. Brandt/I. Yavneh. *On Multigrid Solution of High-Reynolds Incompressible Entering Flow*, J. of Comp. Phys., Vol93, No1, 1991

[DL89] E. Dick/J. Linden. *A Multigrid Method for Solving the Incompressible Navier-Stokes Equations based on Flux-Difference Splitting*, Numerical Methods in Laminar and Turbulent Flow (Taylor, C. et al., eds.), 1989

[FS92] C. Frohn-Schauf. *Flux-Splitting-Methoden und Mehrgitterverfahren für hyperbolische Systeme mit Beispielen aus der Strömungsmechanik*, PhD thesis, Universität Düsseldorf, 1992

[HK91] P. W. Hemker/B. Koren. *Efficient multi-dimensional upwinding for the steady Euler equations*, CWI Amsterdam, Report NM-R9107, 1991

[LS91] G. Lonsdale/K. Stüben. *The Liss Package*, Gesellschaft für Mathematik und Datenverarbeitung, GMD-Arbeitspapier Nr. 524, St.Augustin, 1991

[LUH92] Y. Luh. *Diskretisierungen und Mehrgitteralgorithmen zur Lösung hyperbolischer Differentialgleichungen, am Beispiel der Wellengleichung, der Advektionsgleichung und der verallgemeinerten Stokes-Gleichungen*, PhD thesis, Universität Bonn, 1992

[RAI76] G. D. Raithby. *Skew upstream differencing schemes for problems involving fluid flow*, Comp.Meth.Appl.Mech.Eng., Vol. 9, 1976

[SPE88] S. P. Spekreijse. *Multigrid Solution of the Steady Euler Equations*, PhD thesis, CWI Amsterdam, 1988

discret. error		$h = 1/16$ max	$h = 1/16$ l2	$h = 1/32$ max	$h = 1/32$ l2	$h = 1/64$ max	$h = 1/64$ l2
Standard	u	5.4610(-1)	2.3171(-1)	3.3155(-1)	1.3922(-1)	1.8338(-1)	7.7042(-2)
	v	3.7468(-1)	1.3998(-1)	2.3754(-1)	7.9950(-2)	1.3572(-1)	4.3027(-2)
	p	2.2442(-1)	1.0625(-1)	1.4996(-1)	5.9114(-2)	8.9925(-2)	3.1604(-2)
Var 1	u	7.4498(-2)	3.4455(-2)	3.2797(-2)	1.5974(-2)	1.4839(-2)	7.7054(-3)
	v	8.3215(-2)	2.7560(-2)	4.1690(-2)	1.3131(-2)	2.0804(-2)	6.4427(-3)
	p	9.6263(-2)	4.5227(-2)	4.6684(-2)	2.3076(-2)	2.3327(-2)	1.1668(-2)
Var 3	u	3.7475(-1)	1.5759(-1)	2.1377(-1)	8.9539(-2)	1.1377(-1)	4.7900(-2)
	v	2.7462(-1)	9.5637(-2)	1.6079(-1)	5.2087(-2)	8.7556(-2)	2.7222(-2)
	p	1.5655(-1)	7.3938(-2)	9.8587(-2)	4.0051(-2)	5.6561(-2)	2.1022(-2)
Narrow	u	3.7759(-1)	1.5879(-1)	2.1118(-1)	8.9390(-2)	1.1197(-1)	4.7549(-2)
MOD1	v	2.7624(-1)	9.3385(-2)	1.6541(-1)	5.1263(-2)	9.1187(-2)	2.6919(-2)
	p	1.5842(-1)	8.4703(-2)	9.5957(-2)	4.5079(-2)	5.4836(-2)	2.3439(-2)
Skew	u	2.1579(-1)	9.5404(-2)	1.1307(-1)	5.0126(-2)	5.7818(-2)	2.5738(-2)
MOD1	v	1.7887(-1)	5.3509(-2)	1.0445(-1)	2.9073(-2)	5.6650(-2)	1.5205(-2)
	p	1.4422(-1)	7.0843(-2)	6.9884(-2)	3.5924(-2)	3.4501(-2)	1.8171(-2)
Var 1	u	2.2515(-1)	9 9270(-2)	1.1474(-1)	5.0986(-2)	5.8125(-2)	2.5937(-2)
MOD1	v	1.7884(-1)	5.2642(-2)	1.0446(-1)	2.8723(-2)	5.6700(-2)	1.5103(-2)
	p	1.4958(-1)	7.1555(-2)	7.0972(-2)	3.6019(-2)	3.4598(-2)	1.8160(-2)
Var 2	u	2.0611(-1)	8.8877(-2)	1.2021(-1)	4.8486(-2)	6.6150(-2)	2.5334(-2)
MOD1	v	1.8250(-1)	5.6734(-2)	1.0504(-1)	3.0018(-2)	5.6638(-2)	1.5455(-2)
	p	1.3368(-1)	6.9495(-2)	6.7864(-2)	3.5776(-2)	3.4367(-2)	1.8288(-2)
Var 3	u	4.3156(-1)	1.8174(-1)	2.4872(-1)	1.0466(-1)	1.3347(-1)	5.6389(-2)
MOD1	v	3.0840(-1)	1.0823(-1)	1.8794(-1)	6.0070(-2)	1.0475(-1)	3.1740(-2)
	p	1.7718(-1)	9.0779(-2)	1.1269(-1)	4.9108(-2)	6.5453(-2)	2.5784(-2)
Narrow	u	2.5655(-1)	1.0735(-1)	1.4258(-1)	5.9738(-2)	7.5239(-2)	3.1601(-2)
MOD2	v	1.9245(-1)	5.9951(-2)	1.0460(-1)	3.1581(-2)	5.4351(-2)	1.6286(-2)
	p	1.1244(-1)	3.4168(-2)	6.8786(-2)	1.8410(-2)	3.8449(-2)	9.6506(-3)
Var 2	u	7.0833(-2)	3.6195(-2)	2.6728(-2)	1.4153(-2)	1.1508(-2)	6.1380(-3)
MOD2	v	1.1160(-1)	3.3754(-2)	4.2803(-2)	1.3021(-2)	1.8033(-2)	5.5701(-3)
	p	6.1745(-2)	2.8546(-2)	3.2624(-2)	1.5762(-2)	1.6777(-2)	8.2397(-3)
Var 3	u	3.7056(-1)	1.5608(-1)	2.1220(-1)	8.9127(-2)	1.1337(-1)	4.7787(-2)
MOD2	v	2.8274(-1)	9.7108(-2)	1.6383(-1)	5.2411(-2)	8.8522(-2)	2.7292(-2)
	p	1.5221(-1)	7.2924(-2)	9.7235(-2)	3.9841(-2)	5.6290(-2)	2.0979(-2)

TABLE 1. Discretization error of different 1st order discretizations of the generalized Stokes equations

discretization		$h = 1/16$	$h = 1/32$	$h = 1/64$	$h = 1/128$
Standard		0.174	0.291	0.358	0.409
Narrow		*	*	*	*
Skew Upwind		*	*	*	*
Var 1		*	*	*	*
Var 2		*	*	*	*
Var 3		0.209	0.386	0.509	0.696
Narrow	MOD1	0.157	0.278	0.338	0.377
Skew Upwind	MOD1	0.161	0.278	0.384	0.400
Var 1	MOD1	0.144	0.187	0.315	0.983
Var 2	MOD1	0.252	*	*	*
Var 3	MOD1	0.161	0.279	0.342	0.399
Narrow	MOD2	0.373	0.655	0.563	0.498
Skew Upwind	MOD2	*	*	*	*
Var 1	MOD2	*	*	*	*
Var 2	MOD2	0.489	*	*	*
Var 3	MOD2	0.161	0.278	0.335	0.389

TABLE 2. Convergence rates of multigrid (alternating line relaxation with $\omega = 1.00$)

discretization		$h = 1/16$	$h = 1/32$	$h = 1/64$	$h = 1/128$
Standard		0.139	0.154	0.157	0.158
Narrow		*	*	*	*
Skew Upwind		*	*	*	*
Var 1		0.286	0.436	0.477	0.877
Var 2		*	*	*	*
Var 3		0.129	0.167	0.177	0.175
Narrow	MOD1	0.125	0.142	0.146	0.163
Skew Upwind	MOD1	0.119	0.135	0.204	0.266
Var 1	MOD1	0.118	0.146	0.282	0.391
Var 2	MOD1	0.120	0.145	0.135	0.251
Var 3	MOD1	0.127	0.145	0.149	0.151
Narrow	MOD2	0.698	0.823	0.728	0.604
Skew Upwind	MOD2	*	*	*	*
Var 1	MOD2	*	*	*	*
Var 2	MOD2	0.323	0.292	0.223	0.289
Var 3	MOD2	0.132	0.142	0.146	0.148

TABLE 3. Convergence rates of multigrid (alternating line relaxation with $\omega = 0.75$)

discretization		$h = 1/16$	$h = 1/32$	$h = 1/64$	$h = 1/128$
Standard		0.481	0.698	0.774	0.784
Narrow		*	*	*	*
Skew Upwind		*	*	*	*
Var 1		*	*	*	*
Var 2		*	*	*	*
Var 3		*	*	*	*
Narrow	MOD1	*	*	*	*
Skew Upwind	MOD1	*	*	*	*
Var 1	MOD1	*	*	*	*
Var 2	MOD1	0.793	0.833	0.928	0.859
Var 3	MOD1	0.671	0.734	0.767	0.779
Narrow	MOD2	*	*	*	*
Skew Upwind	MOD2	*	*	*	*
Var 1	MOD2	*	*	*	*
Var 2	MOD2	*	*	*	*
Var 3	MOD2	0.428	0.899	0.933	0.923

TABLE 4. Convergence rates of multigrid (pointwise lexicographic Gauss-Seidel)

		$h = 1/16$		$h = 1/32$		$h = 1/64$	
		max	l2	max	l2	max	l2
$\beta = 1$	u	1.8266(-1)	7.7941(-2)	5.1187(-2)	2.0720(-2)	1.2665(-2)	5.1095(-3)
	v	1.4499(-1)	5.8542(-2)	4.7550(-2)	1.6243(-2)	1.3118(-2)	4.1529(-3)
	p	9.1465(-2)	2.9426(-2)	2.5814(-2)	7.3893(-3)	6.7341(-3)	1.7782(-3)
		$\rho \approx 0.940$		$\rho \approx 0.966$		$\rho \approx 0.985$	
$\beta = 1/2$	u	1.5245(-1)	6.5046(-2)	4.2861(-2)	1.7128(-2)	1.0915(-2)	4.2881(-3)
	v	1.3416(-1)	4.5383(-2)	4.0638(-2)	1.2462(-2)	1.0820(-2)	3.2013(-3)
	p	8.7530(-2)	2.7342(-2)	2.4633(-2)	6.8552(-3)	6.4722(-3)	1.6742(-3)
		$\rho \approx 0.500$		$\rho \approx 0.600$		$\rho \approx 0.660$	
$\beta = 1/3$	u	1.4430(-1)	6.1645(-2)	4.0348(-2)	1.6364(-2)	1.0350(-2)	4.1445(-3)
	v	1.2868(-1)	4.1635(-2)	3.7654(-2)	1.1427(-2)	9.8945(-3)	2.9443(-3)
	p	8.4705(-2)	2.6473(-2)	2.3874(-2)	6.7218(-3)	6.2933(-3)	1.6643(-3)
		$\rho \approx 0.630$		$\rho \approx 0.680$		$\rho \approx 0.720$	

TABLE 5. Discretization error (and convergence rates of defect correction) for standard upstream discretisations

		$h = 1/16$		$h = 1/32$		$h = 1/64$	
		max	l2	max	l2	max	l2
$\beta = 1$	u	9.7860(-2)	4.1224(-2)	2.5847(-2)	1.0250(-2)	6.3444(-3)	2.4484(-3)
	v	1.0858(-1)	4.3587(-2)	3.3960(-2)	1.2337(-2)	9.2618(-3)	3.2060(-3)
	p	5.7690(-2)	1.9666(-2)	1.4666(-2)	4.4902(-3)	3.5122(-3)	9.9227(-4)
		$\rho \approx 0.906$		$\rho \approx 0.950$		$\rho \approx 0.974$	
$\beta = 1/2$	u	8.5495(-2)	3.8295(-2)	2.2892(-2)	9.7726(-3)	5.7929(-3)	2.4229(-3)
	v	9.9934(-2)	3.3115(-2)	2.9231(-2)	9.0816(-3)	7.7533(-3)	2.3415(-3)
	p	5.6648(-2)	1.9986(-2)	1.4878(-2)	4.9618(-3)	3.8252(-3)	1.2174(-3)
		$\rho \approx 0.474$		$\rho \approx 0.539$		$\rho \approx 0.638$	
$\beta = 1/3$	u	8.2121(-2)	3.7061(-2)	2.1889(-2)	9.5708(-3)	5.5808(-3)	2.4031(-3)
	v	9.5558(-2)	3.0793(-2)	2.7370(-2)	8.3843(-3)	7.1651(-3)	2.1585(-3)
	p	5.9911(-2)	2.0237(-2)	1.5992(-2)	5.1646(-3)	4.1625(-3)	1.2964(-3)
		$\rho \approx 0.528$		$\rho \approx 0.608$		$\rho \approx 0.712$	

TABLE 6. Discretization error (and convergence rates of defect correction) for narrow stencil discretisations

13

A Critical Analysis of Multigrid Methods on Massively Parallel Computers

Lesley R. Matheson[1] and Robert E. Tarjan[2]

ABSTRACT The hierarchical nature of multigrid algorithms leaves domain parallel strategies with a deficiency of parallelism as the computation moves to coarser and coarser grids. To introduce more parallelism several strategies have been designed to project the original problem space into non-interfering subspaces, allowing all grids to relax concurrently. Our objective is to understand the potential efficiency of standard and concurrent multigrid algorithms on existing and proposed massively parallel machines. We study model problems on simple domains discretized with finite difference techniques on block structured meshes in two and three dimensions with up to 10^6 and 10^9 points, respectively. Performance of the standard domain parallel V and F cycle schemes is compared to several proposed concurrent algorithms. The multigrid strategies are studied in several models of parallel computation, designed to reflect both the characteristics of existing machines and those of the proposed next generation of multicomputers. These models include a SIMD fine-grain model which contains a large number ($10^4 - 10^6$) of small (bit-serial) processors as well as a SPMD medium-grain model with a more modest number (256-16,384) of powerful (single chip) processors interconnected through a single stage or multistage communication network. Our analysis suggests that obtaining acceptable levels of performance requires optimization techniques which address the salient characteristics of each architectural class of massively parallel computers. With the appropriate optimization techniques, a comparison indicates that the F-cycle is potentially more efficient than the V-cycle despite the relative increase in coarse grid activity. In addition, the analysis suggests that subspace parallelism is too expensive to be practical, except under a very limited set of conditions, on either class of massively parallel computers.

---

[1]Department of Computer Science, Princeton University and NEC Research Institute

[2]Research at Princeton University partially supported by the National Science Foundation, Grant No. CCR-8920505, the Office of Naval Research, Contract No. N0014-91-J-1463, and by DIMACS (Center for Discrete Mathematics and Theoretical Computer Science), a National Science and Technology Center, Grant No. NSF-STC88-09648.

Introduction

Massively parallel computers can potentially deliver the large increase in processing speeds required for the solution of many multigrid applications. Yet the architectural characteristics of these machines present a set of well defined algorithm design incentives which need to be addressed if this potential is to be realized. We've studied a representative set of parallel multigrid strategies in several different models of computation. The purpose of this analysis was to try to gain some insight into how general strategies might compare and perform on different architectural classes of massively parallel machines. In addition, we hoped to gain insight into which implementation issues and strategies might have a substantial impact on the overall performance of multigrid algorithms on these machines.

1 Massively Parallel Computers

The evolution of massively parallel computers has produced at least two significant architectural classes. The machines of both classes are networks of physically distributed processing and memory elements but differ substantially in other architectural characteristics. The early generation of massively parallel computers were fine grained with from 1K to 64K processing elements. The processing elements were small, bit-serial processors commonly packed eight to a chip. The floating point rates of these processors were slow. The processors communicated through various permutation networks consisting of a wide range of topologies. This class of machines executes in either synchronous SIMD or asychronous MIMD mode. Though this generation consists of older machines such as the Intel Hypercube, several more recent machines include the Thinking Machines CM200 and the Maspar MP1 and MP2 series. The current generation of massively parallel computers has been motivated by the the proliferation and decreased cost of powerful, "workstation-size" microprocessors. These machines are multicomputers, interconnection networks of physically distributed processors and memory, linked in a variety of different topological configurations. The processors are generally "off the shelf" single-chip RISC microprocessors. They can perform integer and floating point computation significantly faster than the bit-serial processors which characterized many machines of the previous generation. The increased size, cost and speed of the individual processing elements has delineated a medium grain size for the current generation. Most of the machines are targeted for the range of 1K processors, with larger machines possibly ranging up to 16K processors. The current machines generally exhibit slow interprocessor communication speeds relative to on-chip events. This is frequently a result of handling the network communications processing in the software layer. Unlike the more rigid SIMD and asynchronous MIMD patterns of the previous generation most of the newer machines execute the same program on each processing element with different data, enforcing synchronization only as

required by interprocessor communication. The current generation includes: the CM5 by Thinking Machines, a network of Sun SPARC processor nodes potentially with vector accelerators, connected in a fat tree topology; the Touchstone Delta, developed by Intel and Caltech, a three dimensional mesh of two Intel i860s per node; the Paragon by Intel, a 3D mesh topology with one to four i860 processors per node; the Kendall Square Research machines, a hierarchy of concentric rings with shared virtual memory, with two custom designed chips per node; a Cray Research machine with DEC Alpha processors connected by a yet unrevealed topology.

2 Analysis Methodology

The emergence of these two architectural classes motivated us to study the potential performance of multigrid algorithms using practical models of computation which reflect the salient characteristics of each generation of massively parallel computers. The guiding philosophy behind the development of these models was to strike a reasonable balance between machine independence and practicality, simplicity and accuracy. The models used in this analysis are abstract, based not on a particular machine but on the substantive characteristics of a particular class. The models have parameters to facilitate analysis under different ratios of problem to machine size, and allow the incorporation of changes in technology, such as increases in on-chip computation speed or a decrease in network communication latency. On these models we implemented a representative set of parallel multigrid algorithms in order to try to compare the efficiency of several general strategies on each class of machine. These algorithms included several standard multigrid algorithms as well as more exotic concurrent methods. These implementations produced performance predictions in the form of analytic complexity measures. The complexity measures were interpreted with sensitivity analysis to provide insight into the impact of changes in parameters such as problem size, machine size, the latency of a network communication and on-chip computing rates. We examined these analytic and numerical results to try to understand their implications for algorithm design and implementation on each class of massively parallel computers.

3 Overview

The analysis presented in this paper implements four multigrid algorithms on three models of computation. The results are ordered by model. The first model is the abstract PRAM model. The last two are the practical models based on the early and current generation of massively parallel computers. The four multigrid algorithms include two standard cycling schemes, the V-cycle and the F-cycle, and two concurrent methods, the Gannon-Van Rosendale algorithm and the Chan-

Tuminaro algorithm. Following the results within each model overall conclusions are drawn.

4 Parallel Multigrid Algorithms

Four representative parallel multigrid algorithms, two standard and two concurrent strategies, were analyzed. We looked at simple problems on simple domains in order to provide an initial understanding of the interplay between algorithmic strategies and architectural characteristics. The algorithms solved model problems discretized onto block structured grids using second order finite difference techniques. The problem domains were square and cubic with up to 10^6 and 10^9 points respectively. The iteration schemes were simple explicit weighted Jacobi schemes. The grid hierarchy was generated using a coarsening ratio of two. Inter-grid transfer operators are simple, 5-9 full-weight linear operators. The implementations were parameterized to allow the analysis of more complicated, practical problems such as the Euler and Navier Stokes equations. In addition to considering the standard V-cycle and F-cycle algorithms [5], the analysis considered the performance of two concurrent algorithms. The first algorithm, the Chan-Tuminaro algorithm [4], decomposes the residual from several pre-relaxation sweeps on the finest grid into a set of approximately orthogonal subspaces, projects the subspaces onto the hierarchy of grids, relaxes concurrently on all grids and synthesizes the solution on the finest grid. This process is repeated until truncation error accuracy is obtained on the finest grid. In our analysis the Chan-Tuminaro algorithm decomposes the space over log N steps into log N subspaces by repeatedly bifurcating the residual with a product of the restriction and interpolation matrices. For each grid the residual is ideally split into high and low frequency error components with respect to the relaxation matrix. The high frequency component remains on the fine grid while the low frequency component is projected to the coarser grid. Concurrent relaxation proceeds for O(log N) steps, to amortize the O(log N) lower bound on decomposition, and then the solution is interpolated to the finest grid using standard linear operators. The second concurrent algorithm, the Gannon-Van Rosendale algorithm [1], generates subspace parallelism by decomposing the original problem space using the same frequency bifurcation techniques as those used in the Chan-Tuminaro algorithm but radically redirects the algorithm data flow. In this algorithm as the residual components are decomposed and projected to coarse grids, coarse grid correction information is computed and interpolated to fine grids. This bi-directional flow of information occurs after $O(\log \log N)$ concurrent relaxation sweeps. Thus the Gannon-Van Rosendale concurrent strategy differs fundamentally from the Chan-Tuminaro strategy in that it immediately propagates new approximation information. In addition, the Gannon-Van Rosendale strategy maintains the subspace decomposition by frequent local adjustments without propagating all information through the finest grid.

5 The Parallel Random Access Model

The Parallel Random Access Model of Computation (PRAM) is an ideal model which is somewhat impractical but can provide a fundamental measure of how much parallelism is available in an algorithm. The PRAM posits an arbitrarily large number of processors executing independently and a global shared memory. In the model there is a unit cost for an arithmetic operation. On a PRAM (CRCW) there are no inter-processor communication costs, no memory coherency costs and no synchronization costs. A clear definition of the PRAM can be found in [2].

6 Results on the PRAM

Standard Algorithms

Though the F-cycle is computationally more efficient that the V-cycle by a logarithmic factor, as the number of processors grows large relative to the size of the domain the performance of the two schemes becomes asymptotically equivalent on a PRAM. The asymptotic complexity of the V-cycle and the F-cycle on a PRAM are $O((N^d/P)\log N + \log^2 N)$ and $O(N^d/P + \log^2 N)$ respectively, on a domain with N^d points in d dimensions with P processors.

At or near full concurrency a comparison of the two standard schemes depends on constant factors such as the number of relaxation sweeps and the number of iterations of the F-cycle. If the F-cycle converges in one iteration, it is more efficient than the (2,1) V-cycle if the number of relaxation sweeps is less than 10 regardless of the number of available processors. If more than one F-cycle iteration is required, and more than a few relaxation sweeps are performed per iteration, the outcome depends on the number of processors [3]. With four iterations of the F-cycle and 64K processors (approximately 10 points/processor) processors, the V-cycle is always more efficient.

Concurrent Algorithms

On a PRAM the concurrent algorithms are asymptotically inferior to both of the standard algorithms except at full concurrency. At full concurrency the Gannon-Van Rosendale algorithm is asymptotically more efficient by a factor of $O(\log \log N/\log N)$, while the Chan-Tuminaro algorithm is asymptotically equivalent. Analysis on the PRAM predicts that any advantage of the Gannon-Van Rosendale algorithm degrades rapidly as the number of processors decreases. The number of concurrent iterations in the Gannon-Van Rosendale algorithm is bounded by O(log N) when $O(\log \log N)$ relaxation sweeps are performed each iteration. A Gannon-Van Rosendale iteration is somewhat less effective than a V-cycle iteration. When the number of concurrent iterations is more than a small constant

factor of the number of V-cycle iterations, the V-cycle is more efficient regardless of the number of processors. If the ratio of Gannon-Van Rosendale to V-cycle iterations is less than four, the Gannon-Van Rosendale algorithm can produce a small improvement with a very large number of processors. However, if the number of processors is small enough so that there are more than five points per processor, the Gannon-Van Rosendale algorithm is less efficient than the V-cycle regardless of the number of processors [3]. The Chan-Tuminaro algorithm must converge in a small constant number of concurrent iterations to be at least as efficient as the standard algorithms. The decomposition steps of the Chan-Tuminaro algorithm cannot be executed concurrently as in the Gannon-Van Rosendale algorithm and requires that information propagate through the entire hierarchy to the finest grid. A concurrent iteration is at least as effective as a V-cycle iteration. The analysis on a PRAM suggested that unless the algorithm can converge in a small fraction of the number of V-cycle iterations, the standard methods are more efficient. If the algorithm can converge in a small constant number of iterations, the strategy could be effective if there are a very large number of processors available [3]. The analysis of both algorithms suggests that in order for a concurrent method to be an efficient strategy the decomposition techniques must be fast and accurate and there must be $O(N^d)$ processors available, where N^d is the size of the domain of the finest grid. Without an accurate decomposition, interference inhibits converge and without a large number of processors costly subspace parallelism goes unexploited.

7 The Early Generation Model

The Early Generation Model attempts to capture the salient architectural characteristics of the previous generation of massively parallel computers. This class is characterized by fine grain networks of small processing elements operating in either SIMD or MIMD execution mode. The model is a two dimensional mesh of processors which communicate locally through single-stage nearest neighbor hops along the physical connections of the mesh (as well as along diagonal connections). A single stage communication is assigned a constant fixed cost and the cost of a multiple hop communications is approximately linear in the number of hops. The length of a message is a small constant number of bytes. The cost of an arithmetic operation is treated separately. An arithmetic operation is assigned a constant fixed cost which is based on an average of multiplication and addition floating point rates (without vector acceleration units) across this class of machines. The model executes all instructions in SIMD mode.

The model is a tool to try to better understand the potential of fine grain parallelism. Small (bit serial) processing elements allow these machines to potentially range up to tens or hundreds of thousands of processors. Yet small processing elements tend to have slow floating point rates. The ratio of the cost of an arith-

metic computation to a communicaiton is approximately 3:1. The fine granularity also suggests that local communication will be significantly more efficient than more distant communication. Because of the prohibitive cost of global or irregular communications on this class of machines, the model contains no over-arching multi-stage permutation network. Thus, distant communication incurs a high variable cost relative to the fixed cost of a single node communication. These defining characteristics are captured through constructs which are simple and abstract, but parameterized with data from working computers. Thus, the model provides abstract measures of performance but facilitates sensitivity analysis which can potentially clarify the practical implications of performance measures. A more detailed description of the model can be found in [3].

8 Results on the Early Generation Model

Standard Algorithms

Analysis on the Early Generation model suggests that the performance of multigrid algorithms on fine grain machines with high variable communication costs is highly dependent on the data mapping. With a simple static approach where the square domains are repeatedly quartered to create a 2D mesh of P subblocks and each subblock is assigned to its analog on the 2D mesh of processors, coarse grid communications become substantial. With a small number of processors fine grid computation dominates the execution time of both algorithms. As the number of processors grows, coarse grid communications dominate. With the static approach the asymptotic complexity of the coarse grid communications of the V-cycle and the F-cycle are equivalent. Thus, performance of the two standard algorithms depends on the number of processors available as well as the constant factors. The F-cycle is more efficient than the V-cycle only when there are a small number of processors, the algorithm converges quickly and executes only a small number of relaxation sweeps per iteration. With a dynamic mapping where the coarse grid point sets are contracted and expanded across the processor topology as grids coarsen and refine to maintain a compact data mapping, coarse grid communications can be significantly reduced. With this dynamic implementation the F-cycle is more efficient than the V-cycle even with a large number of processors and slow convergence. Figure 1 shows the total number of machine cycles required for both the V-cycle and the F-cycle for a two dimensional problem of 1,000,000 points when the number of processors varies from 10,000 to 250,000. The figure illustrates the greater efficiency of the F-cycle even with two iterations regardless of the number of processors available.

While the dynamic mapping significantly reduces coarse grid communication costs, the optimal number of processors for both standard multigrid algorithms is less than the number of points in the domain of the problem. As the number of processors grows, fine grid computation costs decrease as coarse grid communication

costs increase. At some point the marginal reduction in computation costs is less than the incremental increase in communications costs and performance degrades with the addition of more processors. This optimum can be approximated analytically as a function of a small number of model parameters [3]. Figure 2 illustrates the optimum for the V-cycle which is approximately four points per processor. Thus, the hierarchical nature of the algorithms coupled with the penalty for distant communications results in excess parallelism degrading the performance of standard multigrid algorithms.

Concurrent Algorithms

When the concurrent multigrid algorithms are implemented using simple, practical data mappings, inter-grid communication costs become significant. With simple mappings at the completion of each concurrent iteration, data must travel across the entire processor mesh. In the simplest strategy, the Chan-Tuminaro algorithm, this inter-grid communication takes place every log N relaxation sweeps. This improves performance relative to the standard V-cycle because the concurrent algorithm exhibits a lower ratio of communication to computation. Figure 3 shows the performance of the Chan-Tuminaro algorithm and the standard V-cycle on a two dimensional problem when the number of concurrent iterations is one-half the number of standard V-cycle iterations. The figure shows that the concurrent algorithm only becomes competitive, under favorable convergence assumptions, at full concurrency. The figure also shows the difference in optimal performance of the two algorithms. Because the inter-grid communications costs of the concurrent method are lower than the coarse grid communications costs of the V-cycle, the optimal number of processors in this range of processors $(0 - N^d)$ is equal to the number of points in the finest grid. The Gannon-Van Rosendale algorithm does not perform as well as the standard V-cycle regardless of the number of processors. The more frequent decomposition and inter-grid transfers of the Gannon-Van Rosendale algorithm make it more expensive than the V-cycle regardless of the number of processors available. Like the V-cycle the Gannon-Van Rosendale algorithm achieves optimal performance below full concurrency. Figure 4 shows the performance of the Gannon-Van Rosendale algorithm and the standard V-cycle in two dimensions with 1,000,000 points when the number of concurrent iterations is equal to the number of V-cycle iterations. The figure illustrates the greater efficiency of the V-cycle and the optimum number of processors for both algorithms below full concurrency. The asymptotic performance of the Gannon-Van Rosendale algorithm can be improved by a logarithmic factor if a more complicated data mapping is used. This mapping reduces the inter-grid communication distances by interspersing the coarse grids, while keeping intra-grid communication distances small. The details of this mapping can be found in [3]. Though the mapping provides an asymptotic improvement, it is substantially more complicated and therefore unlikely to be practical.

9 The Current Generation Models

The Current Generation Models are designed to reflect the architectural characteristics of the current generation of massively parallel computers. These models reflect the medium grain size of this generation with the number of processors ranging from 256 to 16K. The models also reflect the fast floating point rates of the powerful workstation-size processing elements. Motivated by the large disparity between the speeds of on-chip and network events, the models reflect the costs of a two level memory hierarchy. The cost of a local memory access is included in the cost of an arithmetic operation while the cost of a remote memory access is treated separately. The models assume the processors operate in a Single Program Multiple Data mode of execution. The characteristics of the models are similar, differing only in their treatment of communications costs. These different treatments include a simple topologically blind model which simply assigns a fixed cost to any network communication pattern, a model which adds to the fixed cost a variable cost based on a topological approximation of the communications distance and a topologically specific single stage model. Each of these treatments reflects the large fixed cost of a network communications which is characteristic of this generation of massively parallel machines. Conservative model parameters result in a ratio of the cost of a floating point operation to the cost of a communication of approximately 1:250-500. A more detailed description of these models can be found in [3].

10 Results on the Current Generation Models

Standard Algorithms

Analysis on the Current Generation models of the standard V-cycle and F-cycle algorithms indicates that for large problem sizes medium granularity increases fine grid communications costs to unacceptable levels. On moderate sized machines, those with 1K to 4K processors, with a 2500 fixed communications cost (approx. 75 microseconds with a 33 MHz clock speed), the models predicted speed-ups of only 130 times over the serial implementation with approximately 87% of execution time spent on communication. The table below shows the speed-ups and efficiency of the V-cycle, in three dimensions, for different machine sizes under different assumptions of fixed and variable communications costs. The problem size is 10^9 points or 1000 points along each dimension. The mapping used is a simple variant of the two dimensional partitioning scheme.

Interpreting the data is not straightforward. From a theoretical perspective these speed-ups are far from linear. On the other hand computing the wall-clock times associated with these predictions, then scaling these model problem times to reflect the increased complexity of actual applications, produces running times which are unacceptably slow. These discouraging predictions are a result of very

TABLE 1. Speed-Up and Efficiency

Three Dimensional V-cycle, $N^3 = 1,000,000,000$
Current Generation Models

Processors Fixed, Variable	256	1024	4096	16,384
2500, 200	49.1 19.19%	130.6 12.75%	338.1 8.25%	359.3 5.26%
500, 100	129.5 50.58%	389.2 38.01%	1102.2 26.92%	2969.2 18.12%
5000, 0	30.1 11.7%	79.2 7.7%	205.3 5.01%	526.7 3.22%
3600, 0	39.9 15.59%	106.8 10.42%	279.7 6.83%	722.5 4.4%

high communications latencies. In this range of processors, the fine grid communications costs dominate both the cost of the computation and the cost of coarse grid communications. The predictions of these models are in contrast to the asymptotic predictions of more abstract models of computation. Asymptotic analysis suggests the fine grid communications costs become negligible as the problem size gets large for a fixed range of machine sizes. These results suggest the huge imbalance between the cost of communication per word and the cost of a floating point computation causes communication time to dominate the time spent on computation, even with one billion points. The high fixed cost of a network communication coupled with a low spooling rate per word motivates trying to lower the average communication cost per word by transmitting large blocks of words per message. With large messages, the fixed cost of initiating a network communication can be amortized over a larger number of words, lowering the effective fixed cost per word. With the three dimensional V-cycle, the models suggest the ability to send variable length messages, up to 1000 words, produces a marked increase in solution speed in this range of processors, on problems up to one billion points. Table 2 below shows the speed-ups predicted for the three dimensional algorithm by Current Generation models.

With variable length messages, computation costs dominate the total execution time producing almost linear speed-ups in this range of problem to processor size. The corresponding efficiency levels are above 90%. These results suggest the average communications cost per word can be driven down far enough through the efficient transmission of large messages to effectively leverage the increased computational speeds of the current generation of microprocessors. Thus, the ability to package messages into large blocks, up to a 1000 word maximum, can potentially bring these machines closer to the goal of design tool performance on these prob-

TABLE 2. Speed-Up

Three Dimensional V-cycle
with Variable Length Messages
N^3 = 1,000,000,000

Processors Fixed, Variable	256	1024	4096	16,384
2500, 200	253.3	1006.1	3965.4	15,192.1
1000, 200	253.9	1010.3	3999.2	15,555.7
500, 100	254.2	1012.3	4016.9	15,773.9
100, 50	254.4	1013.7	4029.3	15,924.2
3600, 0	253.0	1004.2	3953.3	15,105.8

lems. The ability to bundle messages requires an optimized data partition which minimizes the maximum length of any subdomain border as well as minimizing the number of neighboring regions and load balancing the computation. With structured grids this type of optimized partition is straightforward. With unstructured meshes, however, more sophisticated partitioning techniques are required to produce well-shaped partitions. Several directions for unstructured multigrid partitioning strategies can be found in [3]. With both fixed, constant length and variable length message transmission, analysis on the Current Generation Models suggests that the F-cycle can be more efficient than the V-cycle. With fixed message lengths, this is due mainly to the reduced amount of fine grid communication of the F-cycle. With the ability to send large messages, the F-cycle outperformed the V-cycle in three dimensions because of the reduction in the amount of required computation.

Concurrent Algorithms

The medium granularity of the current generation leaves unexploited much of the subspace parallelism in the concurrent algorithms, making them far too expensive to be practical. With a machines in the range of 256-4096 processors producing more fine grained parallelism is inefficient for even moderate problems sizes. Figure 5 shows the performance of the standard V-cycle and the Chan-Tuminaro algorithm in three dimensions on a problem with 1,000,000 points. Within the medium grain range of machine sizes, the V-cycle is more efficient by at least a factor of two than the concurrent algorithm, even with favorable convergence assumptions.

11 Conclusions

The architectural characteristics of each class of massively parallel computers motivate different optimization strategies to facilitate the realization of the considerable processing power of these machines. Fine grain machines with a high variable cost component in communication costs motivate the optimization of the domain to processor topology mapping. The mapping needs to minimize the topological distance of the required communications. The medium grain machines of the current generation with their high fixed cost of a communication motivate an optimized domain partition. The partition needs to consist of "well-shaped" subdomains in which the maximum size of a subdomain border is minimized and the number of neighboring regions is small. This facilitates the transmission of large messages which allow the fixed cost to be amortized over a large number of words, significantly reducing the average fixed cost per word. With simple optimized strategies the F-cycle is a potentially more efficient than the V-cycle on massively parallel machines. The reduced fine grid activity potentially outweighs the additional coarse grid communications costs on many problems. Concurrent methods even with optimized strategies, require an accurate subspace decomposition and an unreasonably large number of processors to provide a practical alternative to the standard algorithms. This analysis suggests that this remains true regardless of the characterization of inter-processor communications costs.

References

[1] Gannon, D., and Van Rosendale, J., "On the Structure of Parallelism in a Highly Concurrrent PDE Solver", Journal of Parallel and Distributed Computing, vol. 3, pp. 106-135.

[2] JAJA J., "An Introduction to Parallel Algorithms", Addison-Wesley, Reading, MA., 1992.

[3] Matheson, L., "Multigrid Algorithms on Massively Parallel Computers", Phd. Dissertation, Princeton University, 1993.

[4] Tuminaro, R., "Multigrid Algorithms on Parallel Processing Systems", Phd. Dissertation, Stanford University, 1989.

[5] Briggs, W., Multigrid Tutorial, SIAM, Philadelphia, 1987

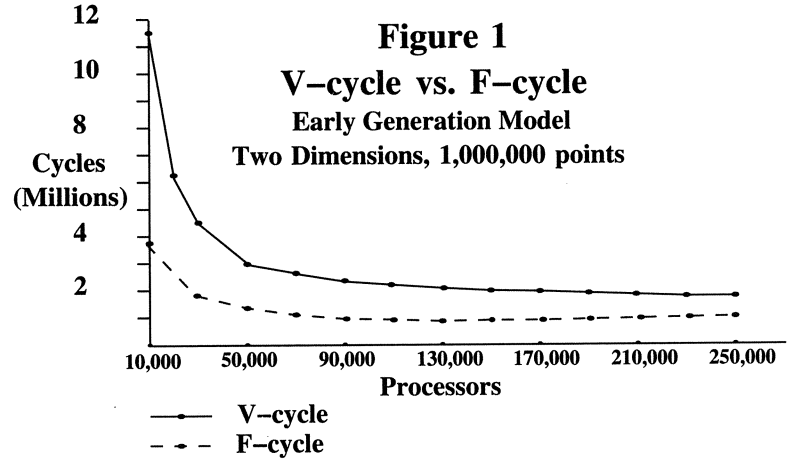


Figure 1
V–cycle vs. F–cycle
Early Generation Model
Two Dimensions, 1,000,000 points

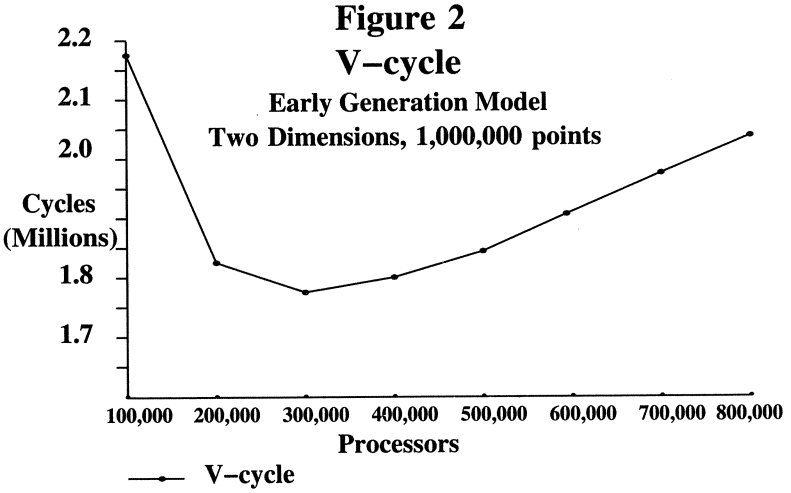


Figure 2
V–cycle
Early Generation Model
Two Dimensions, 1,000,000 points

Figure 3
Chan–Tuminaro versus V–cycle

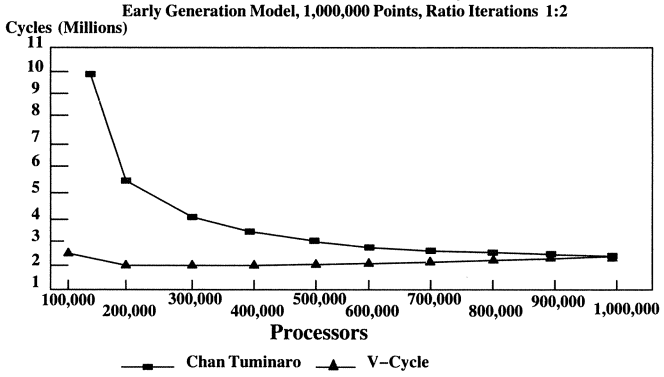


Figure 4
Gannon–Van Rosendale versus V–cycle

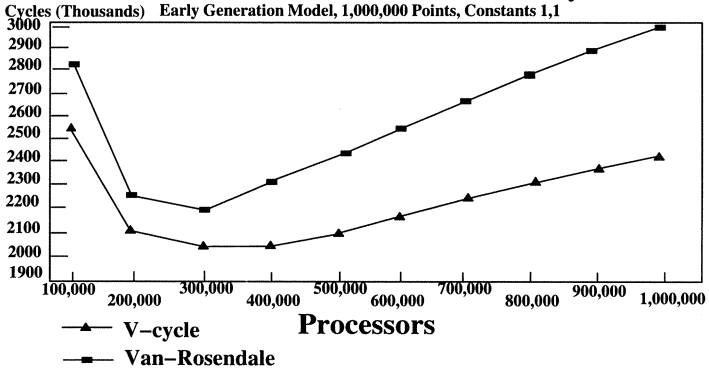
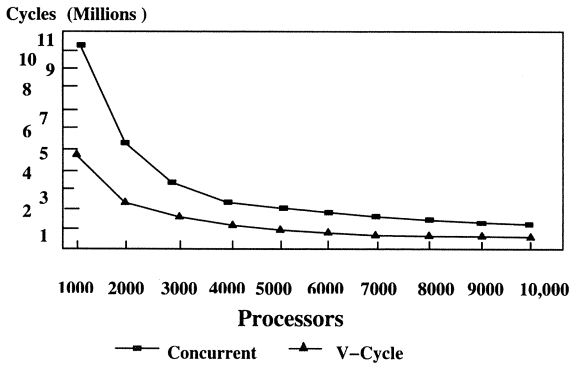


Figure 5
Standard versus Concurrent



14

A Multigrid Waveform Relaxation Method for Time Dependent Incompressible Navier-Stokes Equations

C.W. Oosterlee and P. Wesseling[1]

1 Introduction

There are a number of methods to solve time dependent equations. Different approaches have been suggested to treat the "time-marching" in this type of equations. A standard approach is to solve the equations one time step after another; this is called a time marching scheme. The next time step is tackled when a converged solution on the former time level is obtained. Each time step the equations are solved iteratively in space by, for example, a multigrid method. However, the sequential nature of time marching schemes does not lend itself well for implementation on parallel machines. Therefore new methods have been developed to solve time dependent equations more efficiently on these machines. In 1984 Hackbusch ([8]) introduced two multigrid approaches called parabolic multigrid methods. Here, the time direction can be seen as one of the axes in a space-time grid. The multigrid procedure updates all unknowns in this space-time grid. In these approaches the equations are not solved, but smoothed time step per time step. Some components of these multigrid methods, like prolongation and restriction, can be done in parallel. In one of the methods the smoothing algorithm is sequential. The smoothing procedure on a new time level uses updated values from the previous time levels. In the second multigrid method proposed several time steps can be smoothed simultaneously; "old" values from previous time levels are then used. This smoothing method can be efficiently implemented on parallel machines. The computation on each time level can be independently performed by a processor of a parallel computer. Results with these methods are described in [2] and [4] for the unsteady heat equation, in [5] and [10] for the unsteady incompressible Navier-Stokes equations in primitive variables, and in [11] for the unsteady incompressible Navier-Stokes equations in velocity-vorticity formulation. Parabolic multigrid

[1]Faculty of Technical Mathematics and Informatics, Delft University,
P.O. Box 5031, 2600 GA Delft, The Netherlands

methods or time-parallel multigrid methods for the incompressible Navier-Stokes equations are described in more detail in Section 3.
A different approach, based on the so-called waveform relaxation methods, is proposed by (amongst others) Vandewalle and described in detail in [18], [19]. In waveform relaxation methods, an approximation of an unknown in space is calculated along a time interval of interest consisting of a number of time steps. Instead of updating scalars time step by time step functions in time are updated. Waveform relaxation schemes can be accelerated by multigrid. This solution method also lends itself well for parallel implementation, as is shown in [18] for several initial value and time-periodic problems. In Section 4 a multigrid waveform relaxation algorithm is presented for the incompressible Navier-Stokes equations.
The multigrid methods discussed are investigated in this paper for the incompressible Navier-Stokes equations in general coordinates. All methods will be based on the same smoother, Symmetric Coupled Alternating Lines (SCAL), presented in [17]. SCAL has shown to be robust, showed good results solving the steady incompressible Navier-Stokes equations in general coordinates ([13]), and possesses possibilities for efficient implementation on parallel machines. The algorithms are not implemented as a code for parallel machines. We investigate the performance on a Convex 3840 computer on one processor, but efficiency on parallel machines is considered. We try to give a clear insight in and a comparison between these different multigrid schemes for an unsteady flow in a skewed cavity.

2 The discretization of the incompressible Navier-Stokes equations

The spatial discretization is described in more detail in [12], [13] and [15]. In general coordinates the unsteady incompressible Navier-Stokes equations are given in tensor formulation ([1]) by:

$$U^{\alpha}_{,\alpha} = 0 \tag{1}$$

$$\frac{\partial \rho U^{\alpha}}{\partial t} + (\rho U^{\alpha} U^{\beta})_{,\beta} + (g^{\alpha\beta} p)_{,\beta} - \tau^{\alpha\beta}_{,\beta} = \rho F^{\alpha} \tag{2}$$

where $\tau^{\alpha\beta}$ represents the deviatoric stress tensor given by:

$$\tau^{\alpha\beta} = \mu(g^{\alpha\gamma} U^{\beta}_{,\gamma} + g^{\gamma\beta} U^{\alpha}_{,\gamma}) \tag{3}$$

Here U^{α} are contravariant velocity components, ρ is density, p is pressure and μ is the viscosity coefficient. Unknowns $V^{\alpha} = \sqrt{g} U^{\alpha}$ are used as primary unknowns together with the pressure. The arbitrarily shaped flow domain Ω is mapped onto a rectangular block G, resulting in boundary-fitted coordinates. The coordinate transformation is given by $\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{\xi})$, with $\boldsymbol{x}$ Cartesian coordinates and $\boldsymbol{\xi}$ boundary conforming curvilinear coordinates. The covariant derivative formula used for the

continuity equation is:

$$U^{\alpha}_{,\alpha} = \frac{1}{\sqrt{g}} \frac{\partial \sqrt{g} U^{\alpha}}{\partial \xi^{\alpha}} \tag{4}$$

Terms in the momentum equations of the type $T^{\alpha\beta}_{,\beta}$ are given by:

$$T^{\alpha\beta}_{,\beta} = \frac{1}{\sqrt{g}} \frac{\partial \sqrt{g} T^{\alpha\beta}}{\partial \xi^{\beta}} + \{ \begin{smallmatrix} \alpha \\ \gamma\beta \end{smallmatrix} \} T^{\gamma\beta} \tag{5}$$

where $\sqrt{g}$ represents the Jacobian of the transformation and $\{ \begin{smallmatrix} \alpha \\ \gamma\beta \end{smallmatrix} \}$ the Christoffel symbol of the second kind.
The equations are discretized with a finite volume method on the uniform staggered grid in G. The convection tensor is linearized using a Picard iteration

$$(\rho U^{\alpha} U^{\beta})_{,\beta} \simeq (\rho U^{\alpha(n+1)} U^{\beta(n)})_{,\beta} \tag{6}$$

where the superscript n is an iteration index. The convection term is discretized with a so-called hybrid discretization scheme. Depending on the mesh-Reynolds-number $Re^{(i,j)}$ (i.e. the ratio between the absolute magnitudes of the flux part of the convection term and the viscous term in point (i,j)) the flux part of the convection term is discretized with a central difference scheme (when $Re^{(i,j)} < 1$) or with a first order upwind scheme (when $Re^{(i,j)} > 1$). There is a smooth switch between the two schemes using a smooth switching function $\omega(Re^{(i,j)})$. The total number of variables linked together in a momentum equation is 19.
The numerical solution of the unsteady incompressible Navier-Stokes equations involves the same problem as the solution for the steady equations: the absence of a pressure term in the continuity equation. For the solution of unsteady incompressible flow problems in the primitive formulation three different approaches have been developed. The uncoupled solution technique, in which the velocity components are calculated from the momentum equations separately from a "pressure-correction equation", a discretized form of the continuity equation combined with the momentum equations, is common use in practice. A second order accurate pressure-correction scheme is presented in [20] and is used for example in the ISNaS code, described in [15]. Pressure-correction methods are very efficient for time dependent problems, because one only needs to solve convection-diffusion type equations (usually with a large coefficient on the main diagonal coming from the time derivative) and a Poisson type equation for the pressure correction. In the steady case the advantage of uncoupled solution techniques is less pronounced than in the time dependent case, because they require assumptions about variables yet to be calculated and consequently need more iterations ([14]).
In coupled solution methods, the discretized momentum and continuity equations are solved simultaneously. The method adopted in this paper is a continuation of previous work ([13]). Therefore, a coupled solution method is used.

A third approach is adopted from compressible flow solvers, where for each unknown an explicit equation exists involving its time derivative, for velocity components the momentum equations, for density the continuity equation, and for pressure and temperature the equation of state and the enthalpy equation. Hence, time stepping is straightforward. This approach has been extended to incompressible flows by means of the so-called pseudo-compressibility method ([6]). An artificial time dependent pressure term is added to the continuity equation. A disadvantage of this method is the appearance of a parameter, which is placed in front of the additional term and determines the convergence rate. For the time dependent incompressible Navier-Stokes equations this approach has been used and investigated in [16] and in many other papers. Accurate solution of time dependent equations is difficult with pseudo-compressibility methods.
For the time discretization the second order accurate BDF(2)-scheme ([7]), also called B3-scheme, consisting of a second order approximation of the time derivative and a fully implicit treatment of all spatial derivatives is used, as follows,

$$U^{\alpha(n+1)}_{,\alpha} = 0 \tag{7}$$

$$\frac{3\rho U^{\alpha(n+1)} - 4\rho U^{\alpha(n)} + \rho U^{\alpha(n-1)}}{2\Delta t} + T^{\alpha\beta(n+1)}_{,\beta} = \rho f^{\alpha(n+1)} \tag{8}$$

The time marching scheme. Each time step the discretized equations are solved with the standard nonlinear multigrid method ([3], [9]). The algorithm can do multigrid V-, F- and W- iteration cycles. Prolongation and restriction operators are described in [12]. The smoothing method is the Symmetric Coupled Alternating Lines (SCAL) ([17], [13]). In the steady case new values $(V^{1(n+1)}, V^{2(n+1)}, p^{(n+1)})$ are found with underrelaxation. For the unsteady case the additional diagonal term $\frac{1}{\Delta t}$ acts as an underrelaxation term, therefore no additional relaxation is needed. SCAL is a zebra-type smoother: first all odd (white) rows are visited, then all even (black) rows are visited. With special ordering strategies acceleration can be obtained on parallel computers. Each time step several multigrid iterations are performed until a termination criterion is met. Then, the solution is considered accurate enough and a next time step is tackled. The termination criterion used is:

$$||res_i^{(n)}||_2 < 1 \times 10^{-3} ||rhs_0^{(n)}||_2 \tag{9}$$

That is, a next time step $(n+1)$ is started when the l_2-norm of the residual after i iterations is less than 10^{-3} times the norm of the initial right-hand side. This appeared to be a good termination criterion ([21]), while it is scaling invariant and independent of the initial estimate. An important aspect of time marching schemes is that every time step starts with a good initial approximation of the solution, for example:

$$\boldsymbol{u}_h^{(n+1)} = \boldsymbol{u}_h^{(n)} \tag{10}$$

Comparing the coupled time marching approach using multigrid to an uncoupled time marching approach (ISNaS) for the unsteady incompressible Navier-Stokes

equations ([15]) where a pressure-correction method ([20]) is used and the momentum equations and pressure equations are solved with a GMRESR method ([21]) and ILU preconditioner, it appeared that the coupled approach was slower than the uncoupled approach on a Convex 3840 using one processor. The pressure-correction technique can be incorporated in a parabolic multigrid method ([10]) for the unsteady case. However, to the author it is not clear at the moment how an uncoupled solution technique can be incorporated in a waveform algorithm. Finally it is to be noted that this coupled solution technique is robust; arbitrarily large time steps can be taken in arbitrary domains. Furthermore, all white rows as well as all black rows can be done in parallel, each on a single processor. Probably for many problems a white-black cell-by-cell smoother will be more efficient (well vectorizable), but certainly less robust !

3 The parabolic / time-parallel multigrid method

The sequential process of solving equations time step by time step with a time marching scheme makes algorithms less efficient on parallel machines. In the following multigrid schemes based on a paper by Hackbusch ([8]) the time-axis in a space-time grid is an axis along which solutions will be updated simultaneously for a number of time steps. A convergence criterion must be satisfied for all unknowns in this grid, so when the criterion is met all solutions on all time levels considered will be accurate enough. The algorithm will be sketched for the implicit Euler scheme, which can be summarized as

$$\frac{\mathbf{u}_h^{(n+1)} - \mathbf{u}_h^{(n)}}{\Delta t} + T_h(\mathbf{u}_h^{(n+1)}) + \boldsymbol{f}_h^{(n+1)} \tag{11}$$

The time steps n_1, $n_1 + 1$, ... , n_2 will be updated simultaneously. Two different smoothing algorithms (sequential and parallel) are now described. They are of the following type:

Sequential smoothing algorithm:

begin
for iteration number $\nu = 1$ step 1 until ν_{max} **do :**
for time levels $n = n_1$ step 1 until n_2 **do :**
for space indices $i \in I^o \bigcup \partial I_N$ **do:**

Solve $((\frac{I}{\Delta t} + M_h)\boldsymbol{u}^{(n+1),\nu})_i = (N_h \boldsymbol{u}^{(n+1),\nu-1})_i + (\frac{I}{\Delta t}\boldsymbol{u}^{(n),\nu})_i + \boldsymbol{f}_i^{(n+1)}$

enddo
end sequential smoothing.

Parallel smoothing algorithm:

begin
for $\nu = 1$ step 1 until ν_{max} **do :**
for $n = n_1$ step 1 until n_2 **do :**
for $i \in I^o \bigcup \partial I_N$ **do:**

Solve $((\frac{I}{\Delta t} + M_h)\boldsymbol{u}^{(n+1),\nu})_i = (N_h \boldsymbol{u}^{(n+1),\nu-1})_i + (\frac{I}{\Delta t}\boldsymbol{u}^{(n),\nu-1})_i + \boldsymbol{f}_i^{(n+1)}$

enddo
end parallel smoothing.

Note that in the latter case all time steps can be done in parallel. Indicating the coarse grid by a subscript H, the nonlinear time-parallel two-grid method is given by:

Nonlinear time-parallel two-grid algorithm:
begin algorithm

for number of iterations $\nu = 1$ step 1 until ν_{max} **do :**
for time levels $n = n_1$ step 1 until n_2 **do :**
for spatial indices $i \in I^o \bigcup \partial I_N$ **do:**
• Apply a pre-smoothing iteration (*sequential or parallel*)
enddo
for time levels $n = n_1$ step 1 until n_2 **do** *parallel*:
• Compute residual:

$$\mathbf{r}^{(n+1)} = (f_h^{(n+1)} + \frac{I}{\Delta t}u_h^{(n)}) - (\frac{I}{\Delta t} + T_h)u_h^{(n+1)} \tag{12}$$

enddo
for time levels $n = n_1$ step 1 until n_2 **do** *parallel*:
• Choose $\tilde{u}_H^{(n+1)}$.
• Apply Restriction $\tilde{R}^H : u_H^{(n)} = \tilde{R}^H u_h^{(n)}$, $\tilde{R}^H : \tilde{u}_H^{(n)} = \tilde{R}^H u_h^{(n)}$.
• Apply Restriction R^H as $\mathbf{r}^{(n+1)}$.
enddo
for $n = n_1$ step 1 until n_2 **do :**

...

...

• Solve the coarse grid equation for $\boldsymbol{u}_H^{(n+1)}$.

$$\frac{\boldsymbol{u}_H^{(n+1)} - \boldsymbol{u}_H^{(n)}}{\Delta t} + T_H \boldsymbol{u}_H^{(n+1)} = \boldsymbol{f}_H^{(n+1)} \tag{13}$$

$$= s_H R^H(\mathbf{r}^{(n+1)}) + (\frac{I}{\Delta t} + T_H)\tilde{\boldsymbol{u}}_H^{(n+1)} - \frac{I}{\Delta t}\tilde{\boldsymbol{u}}_H^{(n)} \tag{14}$$

enddo

for time levels $n = n_1$ step 1 until n_2 **do** *parallel*:

• Prolongation:

$$\boldsymbol{u}_h^{(n+1)} = \boldsymbol{u}_h^{(n+1)} + \frac{1}{s_H} P^h(\boldsymbol{u}_H^{(n+1)} - \tilde{\boldsymbol{u}}_H^{(n+1)}) \tag{15}$$

enddo

for number of iterations $\nu = 1$ step 1 until ν_{max} **do :**

for time levels $n = n_1$ step 1 until n_2 **do :**

for spatial indices $i \in I^o \bigcup \partial I_N$ **do:**

• Apply a post-smoothing iteration (*sequential or parallel*)

enddo

end nonlinear time-parallel two-grid algorithm.

Again the coarse grid equation can be solved in a similar way, i.e. by introducing a third coarser grid and by executing a number of two-grid cycles, etcetera. The introduction of a sequence of grids leads to the time-parallel multigrid algorithm, also called parabolic multigrid algorithm with different iteration cycles. Here again V-, F- and W-cycles are implemented. Note that different stages in the multigrid algorithm can be performed in parallel in time direction. The restriction and prolongation operators are only spatial operators. We do not apply grid coarsening in time.

In [2] results are obtained efficiently with a parabolic multigrid method on a transputer system. A sequential smoother of Gauss-Seidel type is compared to a parallel smoother of Jacobi type for a parabolic differential equation. It was found that a good speed-up was obtained for the sequential smoother for many processors in a model problem, while the good speed-up with the parallel smoothing method was limited to a small number of processors.

In [5] and [10] a time-parallel version of the SIMPLE algorithm is used to solve the incompressible Navier-Stokes equations on a transputer system. Good efficiency is obtained for an unsteady driven cavity problem. In [11] a vorticity-velocity formulation is applied to the incompressible Navier-Stokes equations. The smoother is a so-called Group Explicit Iterative method (GEI), and good convergence and CPU

time results for a small number of processors (varying from 1 to 4) are presented for the steady driven cavity problem solved with unsteady equations.
In this paper the incompressible Navier-Stokes equations in general coordinates will be smoothed in the time-parallel multigrid method with a sequential and a parallel variant of the SCAL smoother as smoothing method on all time levels. The parallel variant is implemented as in [8]. In these time-parallel methods (10) can not be used to obtain a starting solution. A good initial approximation on each time level will be produced with "nested iteration". Implemented are nested iteration V- and F-cycles.

4 The multigrid waveform relaxation method

A smoothing algorithm is the most time consuming part of a multigrid method, and it will be interesting to execute this part efficiently on a parallel machine. With a waveform relaxation method communication costs are probably lower than for the other relaxation schemes. Waveform relaxation methods update an unknown in a grid-point along a time interval consisting of a number of time steps. If an unknown in space is assigned to a processor, then during the smoothing of that unknown in time there is no need for a lot of communication with other processors, which can be costly. Originally, waveform methods were developed as relaxation schemes in simulation techniques of electrical network problems. Waveform relaxation methods are found to have qualitatively the same convergence behaviour as basic iterative methods. High frequency errors are smoothed quickly, while low frequency errors are damped slowly. Therefore waveform relaxation methods are also suited for a multigrid acceleration. The nonlinear multigrid waveform algorithm differs from the nonlinear time-parallel method, presented in the previous section, only in the smoothing algorithm. Again the restriction and prolongation operators, which are spatial operators (no coarsening is applied to time steps), can be done in parallel. In [18] the multigrid waveform algorithm is found to perform very well on several nonlinear initial boundary value and time-periodic parabolic partial differential equations on a parallel machine. The smoother used is a white/black Gauss-Seidel waveform smoother. A comparison between the standard time marching scheme and the multigrid waveform method even on a single processor shows a competitive performance for the multigrid waveform method in many cases considered ([19]). Here, a multigrid waveform relaxation method to solve the incompressible Navier-Stokes equations is presented with the SCAL waveform relaxation method as smoother:
First all horizontal "white" rows are updated (which can be done in parallel on a parallel machine) on all time levels t_{n_1},...,t_{n_2}. Then, all horizontal "black" rows are updated on all time levels. After a waveform sweep along horizontal rows a sweep along vertical rows will be applied.
Since many time levels are involved the horizontal and vertical rows (in space) are

in fact vertical time blocks.
For an implicit Euler scheme the waveform smoother looks like:

Waveform smoothing algorithm:

begin
for number of iterations $\nu = 1$ step 1 until ν_{max} **do :**
for spatial indices $i \in I^o \bigcup \partial I_N$ **do:**
for time indices $n = n_1$ step 1 until n_2 **do :**

$$\text{Solve } (\left(\frac{I}{\Delta t} + M_h\right) \boldsymbol{u}^{(n+1),\nu})_i = (N_h \boldsymbol{u}^{(n+1),\nu-1})_i + (\frac{I}{\Delta t}\boldsymbol{u}^{(n),\nu})_i + \mathbf{f}_i^{(n+1)}$$

enddo
end waveform smoothing.

A disadvantage of the waveform method chosen is that extra storage of operators and unknowns is required. A remedy to avoid too much storage is to use small values of $n_2 - n_1$.

5 Results

In the test example four methods are compared:
1) the time marching scheme with the initial solution from (10),
2) the time-parallel multigrid method with sequential SCAL,
3) time-parallel multigrid with parallel SCAL,
4) multigrid waveform relaxation with waveform SCAL,

An unsteady skewed driven cavity. The test example used here is the unsteady flow in a driven cavity, which has the shape of a parallelogram, with skewness angle 45^o. It is described and invesigated for the time-parallel method on a parallel machine in [10] for a square cavity. The topwall of the skewed cavity is moving with velocity $u = sin(t)$ from $t_0 = 0$ to $t_{end} = 1.5$ ($\cong \pi/2$). The number of time steps is 40; they are divided in time-windows $(n_2 - n_1)$ consisting of 5, 10, 20 and 40 time steps. The behaviour of an F-cycle with 0 pre- and 1 post-smoothing iterations (F(0,1)) for a higher Reynolds number is investigated: $Re = 1000$. We obtained the average number of iterations to satisfy termination criterion (9) and approximations of convergence factors $\nu_i^{(n)}$ from

$$\nu_i^{(n)} \equiv \| \frac{res_i^{(n)}}{res_{i-2}^{(n)}} \|^{1/2} \quad \text{for large i} \tag{16}$$

For n always the last time level is chosen. There the approximate convergence factor was generally not smaller than for earlier time steps. In many cases $\nu_i^{(n)}$ is found to be approximately constant (for $i \cong 20$), in which case we have found the asymptotic convergence factor ($n = 40$). As starting solution (10) is chosen for the F(0,1)-cycle for the time marching scheme and a zero starting solution for the other schemes. For these multigrid schemes also results for an FF(0,1)-cycle (an F-cycle with F-nested iteration) are presented. The results are presented in Table 1.

TABLE 1. Average number of iterations per time step to satisfy the termination criterion and approximations of the convergence factor for the skewed driven cavity problem with (FF(0,1)) and without (F(0,1)) nested iteration, on a 64×64- grid.

scheme	window T :	$1.\Delta t$
	non-nested	seq.
marching	F(0,1) # it.	2.0
	F(0,1) $\nu_i^{(n)}$	0.27

	window T :	$5.\Delta t$		$10.\Delta t$		$20.\Delta t$		$40.\Delta t$	
	(non)-nested	seq.	par.	seq.	par.	seq.	par.	seq.	par.
parabolic	F(0,1) # it.	3.9	5.7	4.2	8.0	4.4	12.2	4.5	20.4
	FF(0,1) # it.	2.9	4.5	3.0	6.5	3.0	10.7	3.1	18.1
waveform	F(0,1) # it.	4.1	-	4.6	-	5.0	-	5.2	-
	FF(0,1) # it.	3.0	-	3.2	-	3.3	-	3.5	-
parabolic	F(0,1) $\nu_i^{(n)}$	0.32	0.29	0.28	0.31	0.28	0.48	0.28	0.20
	FF(0,1) $\nu_i^{(n)}$	0.35	0.29	0.31	0.32	0.31	0.48	0.30	0.22
waveform	F(0,1) $\nu_i^{(n)}$	0.29	-	0.31	-	0.34	-	0.36	-
	FF(0,1) $\nu_i^{(n)}$	0.30	-	0.32	-	0.36	-	0.36	-

6 Conclusions

Three multigrid methods for the time dependent incompressible Navier-Stokes equations equations in general coordinates have been compared for several test problems, namely a time marching scheme, a parabolic multigrid method and a multigrid waveform relaxation method. For all methods an essential part of the algorithm, the smoother, was based on the same robust alternating zebra line smoothing method SCAL. Contrary to solving steady incompressible Navier-Stokes equations no additional underrelaxation is required to solve the unsteady

equations. A sequential and a parallel version of the smoothing method are compared in the time-parallel multigrid method. Approximate convergence factors and the average number of iterations per time step to satisfy a termination criterion are investigated. Satisfactory results were obtained for all three methods with the sequential smoother, especially with nested iteration. The number of iterations to satisfy the stopping criterium is approximately the same for the parabolic multigrid method and for the multigrid waveform relaxation method. Furthermore, this number did not differ much when larger time-windows were used. For the parallel smoother in the time-parallel method the efficiency is reduced, when larger time-windows are encountered. The sequential computational complexity of all three methods is about the same, but their differ markedly in their parallelization potential. It will be interesting to study the behaviour of the smoothers on a parallel machine.

References

[1] R. Aris, *Vectors, tensors and the basic equations of fluid mechanics.* Prentice-Hall, Englewood Cliffs, N.J. (1962).

[2] P. Bastian, J. Burmeister and G. Horton, Implementation of a parallel multigrid method for parabolic differential equations. In: W. Hackbusch (ed.), *Parallel algorithms for partial differential equations*, Proc. 6th GAMM seminar Kiel, pp. 18-27, Vieweg, Braunschweig, (1990).

[3] A. Brandt, Guide to multigrid development. In: W. Hackbusch and U. Trottenberg (eds.), *Multigrid Methods*, Lecture notes in Mathematics **960**, pp. 220–312, Springer Verlag, Berlin (1982).

[4] J. Burmeister, *Paralleles Lösen diskreter parabolischer Probleme mit Mehrgittertechniken* Master's Thesis, Kiel, (1985).

[5] J. Burmeister and G. Horton, Time-parallel multigrid solution of the Navier-Stokes equations. In: W. Hackbusch and U. Trottenberg (eds.), *Multigrid Methods III,* International Series of Numerical Mathematics **98**, pp. 155–166, Birkhäuser, Basel, (1991).

[6] A.J. Chorin, A numerical method for solving incompressible viscous flow problems. *J. Comp. Phys.* **2,** 12–26, (1967).

[7] C.W. Gear, *Numerical initial value problems for ordinary differential equations.* Prentice Hall, Englewood Cliffs, (1971).

[8] W. Hackbusch, Parabolic multigrid methods. In: R. Glowinski, J.R. Lions (eds.), *Computing methods in Applied Sciences and Engineering VI,* Proc. 6th Int. Symp. on Comp. Meth. in Appl. Sciences and Eng. pp. 20–45, North Holland, Amsterdam, (1984).

[9] W. Hackbusch, *Multi-grid methods and applications.* Springer Verlag, Berlin (1985).

[10] G. Horton, The time-parallel multigrid method. *Comm. Appl. Num. Methods* **8,** 585–596 (1992)

[11] S. Murata, N. Satofuka and T. Kushiyama, Parabolic multi-grid method for incompressible viscous flows using a group explicit relaxation scheme. *Comp. and Fluids*, **19**, 33–41, (1991).

[12] C.W. Oosterlee and P. Wesseling, A multigrid method for an invariant formulation of the incompressible Navier-Stokes equations in general coordinates. *Comm. Appl. Num. Methods* **8,** 721–734 (1992)

[13] C.W. Oosterlee and P. Wesseling, A robust multigrid method for a discretization of the incompressible Navier-Stokes equations in general coordinates. *Impact Comp. Science and Eng.* **5,** 128–151 (1993).

[14] W. Rodi, S. Majumdar and B. Schönung, Finite volume methods for two-dimensional incompressible flows with complex boundaries. *Comp. Meth. in Appl. Mech. and Eng.* **75,** 369–392 (1989).

[15] A. Segal, P. Wesseling, J. van Kan, C.W. Oosterlee and C.G.M. Kassels, Invariant discretization of the incompressible Navier-Stokes equations in boundary fitted co-ordinates. *Int. J. Num. Methods in Fluids* **15,** 411–426 (1992).

[16] W.Y. Soh and J.W. Goodrich, Unsteady solution of incompressible Navier-Stokes equations. *J. Comp. Phys.* **79,** 113–134, (1988).

[17] M.C. Thompson and J.H. Ferziger, An adaptive multigrid technique for the incompressible Navier- Stokes equations. *J. Comp. Phys.* **82,** 94–121 (1989).

[18] S. Vandewalle, *The parallel solution of parabolic partial differential equations by multigrid waveform relaxation methods.* Ph.D. Thesis, Leuven University, Leuven (1992).

[19] S. Vandewalle, R. Piessens, Efficient parallel algorithms for solving initial-boundary value and time-periodic parabolic partial differential equations. *SIAM J. Sci. Stat. Comp.*, **13,** 1330–1346, (1992).

[20] J.J.I.M. van Kan, A second-order accurate pressure correction method for viscous incompressible flow. *SIAM J. Sci. Stat. Comp.* **7,** 870–891, (1986).

[21] C. Vuik, Solution of the discretized incompressible Navier-Stokes equations with the GMRES method. *Int. J. Num. Methods in Fluids* **16,** 507–523 (1993).

[22] P. Wesseling, *An introduction to multigrid methods.* John Wiley, Chichester (1992).

15

Implementation Aspects of the Multigrid Formulation of Finite Volume Algorithms on Unstructured Grids

Kris Riemslagh and Erik Dick[1]

1 Introduction

Multigrid methods were originally formulated on structured grids. Seeking an efficient implementation of the multigrid on unstructured meshes involves reformulation of the different parts of the multigrid. The choice of grids and intergrid connections determines the amount of work needed for these operations. To facilitate the formulation of the intergrid functions, coarser meshes can be constructed with telescoping nodes or cell faces. Different possibilities are discussed.

The most cpu intensive part of a multigrid process is the spacial discretization operator. We considered the vertex centred formulation of a flux-difference finite volume algorithm. The data structure used to calculate the fluxes and flux balances determines the efficiency and the ease of the construction of the operator. Several alternatives, point-based, edge-based or cell-based data structures are possible. For every choice different possibilities exist to describe the connectivity information. An evaluation of these choices is given.

Finally for a given data structure type and a given algorithm, orderings can be imposed on the different lists of the data structure. These orderings can improve the speed of searching algorithms and the speed of execution by improving local use of data.

2 Data structure for spacial discretization

In the finite volume method, Euler or Navier-Stokes equations are discretized by writing the flux balance over a control volume. The vector of variables that belongs to the control volume is then updated with this flux balance. Therefore the

[1]Universiteit Gent, Sint-Pietersnieuwstraat 41, 9000 Gent, Belgium

calculation of the flux balances is very often one of the most calculation intensive parts of solving the problem. So, it is very important that these calculations are organised efficiently. For an unstructured mesh, the flux balance calculations can be ordered in several ways. The class of methods that can be used to update the vectors of variables is often determined by this. Also the way the flux balances are calculated has some influence on the data structure that can be used. The flux balance can be calculated per control volume, with a loop over the surrounding or forming edges. A Gauss-Seidel type update could then be used. It is also possible to calculate all the flux balances at once, with a loop over all the edges. For every edge, the calculated flux is added to one side and subtracted from the other side. Now, a Jacobi type update would be more suitable, since all the updates have to be done after the flux balance calculations. Other orderings for the calculations are possible. The different data structures that can be used to implement the calculations, are shortly discussed below. Given a certain mesh with cells (e.g. triangles, polygons), nodes and edges, the data structure can be cell-based, node-based or edge-based. This classification is made on the way the connectivity information is stored. In general, for any data structure there are lists of nodes (state-variables,...), lists of edges (length, outer normal,..), and lists of cells (area, ...). But the way information such as, which nodes form an edge or which edges form a triangle, is stored, classifies the data structure. A cell-based data structure stores for every cell a list of connectivities. These could be nodes or could be edges. The same is true for the other data structure types. Every type has some minimum form, in which the minimum information to describe the connectivity is stored. Extensions to a minimum form are always possible, and are very often used to facilitate the description of the algorithm. The extensions can always be constructed from the minimum form.

The classic cell-based data structure is illustrated in figure 1a, and consists of a list of node references for every cell. A typical extension is given in figure 1b, storing also references to the neighbouring cells. Figure 1c represents a minimum form of the edge-based data structure. Some variants of extensions are given in figure 1d, 1e, 1f and 1g. For a node-based data structure figure 1h and 1i give some suggestions. Figure 1i is a so-called out-degree structure. For more information about this data structure and others, see [1]. The examples given above are abstract data structures. They can all be implemented in different ways, depending on the programming language and inventivity of the programmer. For example the data structure of figure 1h implemented in Fortran could result in an array of nodes with for every node a list of connected nodes.

```
PARAMETER (MAXNODES=10000,MAXNEIGHBOURS=10)
INTEGER CONNECT(MAXNODES,MAXNEIGHBOURS)
```

Since the average number of neighbours for a triangulation is less then 6, a memory more efficient implementation would be to store for every node a reference to an array in which all the neighbours of all the nodes are stored.

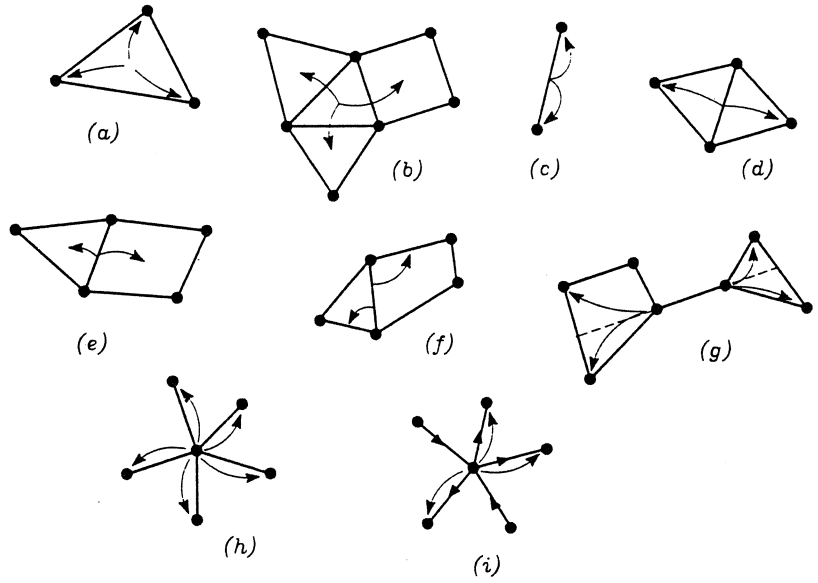


FIGURE 1. Geometrical interpretation of some abstract data structures.

```
PARAMETER (MAXNODES=10000)
INTEGER FIRST NEIGHBOUR(MAXNODES+1)
INTEGER NEIGHBOUR(6*MAXNODES)
```

A loop to calculate the flux balances could then look like

```
DO ND=1,NNODES
DO NB=FIRST NEIGHBOUR(ND),FIRST NEIGHBOUR(ND+1)-1
ND2 = NEIGHBOUR(NB)
calculate flux of edge with nodes ND and ND2
ENDDO
ENDDO
```

This second implementation may require more complex descriptions of grid changing algorithms. Certainly other alternatives are possible. The implementation in C-language could be done with pointers, with for every node a linked list of neighbouring nodes.

From the examples above, it is clear that a lot of different more or less efficient implementations of different variants of the same data structure type are possible. Which one to choose depends on the set of operations or algorithms that need to be described with it. The choice that was made by the authors is an edge-based data structure represented by figure 1c extended with the one of figure 1f, and

for the higher order flux-definition figure 1g. It consists of a list of edges with for every edge the two forming nodes. This is the minimal form. The extension to it, to facilitate mesh generation, exists of two references to neighbouring edges for every edge. These two neighbours are found by rotating around the two edge forming nodes counter-clockwise. An extra list of nodes with for every node one reference to a neighbouring edge, allows an easy description of a loop over all the nodes with an inner loop over the surrounding edges. The Fortran implementation of the data structure looks then like

```
PARAMETER (MAXNODES=10000)
PARAMETER (MAXEDGES=3*MAXNODES)
INTEGER NODE(MAXEDGES,2)
INTEGER NEXT ED(MAXEDGES,2)
INTEGER CONN ED(MAXNODES)
```

3 Intergrid operations

The amount of work needed for the intergrid operators is determined largely by the way the coarser meshes were generated. The first choice that should be considered is the case in which the meshes are totally unrelated. On the other hand, to facilitate intergrid functions, coarser meshes can be constructed with some telescoping nodes or cell faces. With telescoping is meant that the nodes or the cell faces of the coarse mesh appear in the finer meshes.

For two consecutive meshes from a multigrid cycle, the ratio of the number of nodes is roughly taken to be c^d with c normally around 2 and d the number of dimensions. This ratio should be met globally as well as locally. Most mesh generators have a way of defining the local mesh spacing to help meeting this ratio.

3.1 Totally unrelated meshes

In the case of totally unrelated meshes, no extra demands on the mesh generation are imposed. That this approach works well was illustrated in [2]. Since the meshes are generated independently, they have to be interconnected, preferably in a preprocessing stage. Whatever order of interpolation is used, for every fine grid node, the surrounding coarse grid cell has to be determined, and for every coarse grid node, the surrounding fine grid cell has to be found, see figure 2. With this information, interpolation coefficients can be calculated and stored for use during the flow calculation. How the interpolation formula looks like depends on the order of interpolation. For a triangular mesh, zero and first order are relatively simple in more dimensions, but higher order interpolation requires data from nodes located in a larger perimeter. For generally unstructured meshes these higher order interpolations are normally not used.

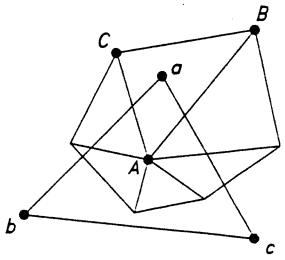


FIGURE 2. Intergrid connections for totally unrelated meshes.

To determine the surrounding cell for a particular node of a different mesh is not a trivial problem. A naive search over all the cells would require $O(N)$ operations, where N is the number of nodes, resulting in $O(N^2)$ operations for the connection of all the nodes. This would be prohibitively expensive, requiring more time than the flow solution itself. Hence an efficient search algorithm is needed. In [2] a grid-search algorithm is described with theoretical performance $O(N \log N)$ but with practical performance even $O(N)$. The algorithm for finding the surrounding cell of a node, starts from a given cell examining the neighbourhood of this cell. A larger perimeter is taken untill the surrounding cell is found. The closer the starting cell is, the faster the surrounding cell is found. Therefore, taking the next node as a neighbour of the previous node, with the starting cell the surrounding cell of the previous node, speeds up the algorithm considerably.

Using an unstructered mesh in a multigrid process, one would probably want to store the different meshes independently of each other. Therefore, an algorithm to connect two meshes is necessary anyhow, even when the meshes are generated with telescoping nodes or cell faces.

3.2 Telescoping cell faces

The coarse mesh is said to have telescoping cell faces if the coarse cell faces appear in the finer meshes. This is the case in the volume-agglomeration method for the control volumes [3, 4]. The cell faces are found in the dual mesh of the control volumes. The coarse levels are imbedded into the fine one by concatenation of control volumes defined around a mesh point. This results in coarse levels that are defined by a polygonal partition of the domain. These polygonal partitions are the control volumes for the finite volume discretisation. Since these partitions are not convex, the dual of this mesh does not necessarily exist, and this creates difficulties for the evaluation of higher order derivatives. The method works well if the stencil of the discretized operator is small enough, as for the central differencing or first order upwind flux differencing operator. The special form of the coarser levels has some important implication for the multigrid algorithm.

The restriction operator is similar to that used for structured grids, since coarse control volumes are built up of fine grid control volumes. The residuals are simply summed from the fine grid. For the restriction of the flow variables, an area-weighted interpolation can be used, as in [4]. For the prolongation of the correction, however, only piecewise constant interpolation can be applied. A linear interpolation would require the existence of a coarse triangulation. Of course, additional smoothing steps can be employed to smooth the correction.

The reason why the volume-agglomeration method was developed was to avoid the mesh generation of the coarser meshes, and to reuse the partitioning of the domain of the fine mesh. The aim was to place the agglomeration itself together with the construction of the restriction and the prolongation, in a separate unit of which the user does not need to know the details of implementation.

3.3 Telescoping nodes

In the mesh generation with telescoping nodes two different approaches are possible. Meshes can be generated from coarse to fine or from fine to coarse. Of course, given a fine mesh one can only generate the coarser ones, preferably in an automatic way independent of the geometry of the mesh. To do this, a set of nodes from the fine mesh has to be selected and reconnected. During the selection phase special attention must be paid to the boundary nodes. Typically half of the boundary nodes are selected. Some of the boundary nodes are crucial for the description of the geometry and have to be selected every time during the coarsening, e.g. leading and trailing edge of an airfoil. In [5] a coarsening algorithm is given which is based on the idea that all the neighbours of a coarse node can be removed from the list of selected nodes. Initially all nodes are selected. In [6] more or less the inverse is done, starting from a list of non-selected nodes, all neighbours of an non-selected node are selected. In the first algorithm, the special boundary nodes are processed first, then the other boundary nodes, and then the internal nodes. The algorithm reduces the number of nodes roughly by a factor of 2^d. In the second algorithm [6] special points cannot be removed, and the other boundary points are only processed the first time. Here the number of nodes is only reduced to $3/4$ of the initial number so that the algorithm must be applied a few times to get the required coarsening. Examples of both coarsening strategies are given in figure 3c and 3d. If we compare both strategies in terms of mesh quality, it is clear that the first strategy by [5] gives *smoother* meshes, although the multigrid convergence does not seem to differ much in both cases [6, 7].

After selection of the set of nodes that will form the coarser mesh, these nodes have to be connected. As for the initial mesh, all mesh generation algorithms can be used, starting from scratch. It is also possible to remove the non-selected nodes from the finer mesh. How this can be done will be discussed in the mesh generation section.

If the finest mesh is generated with a node incremental process starting from a very coarse mesh, always adding nodes, it is very easy to store some of the

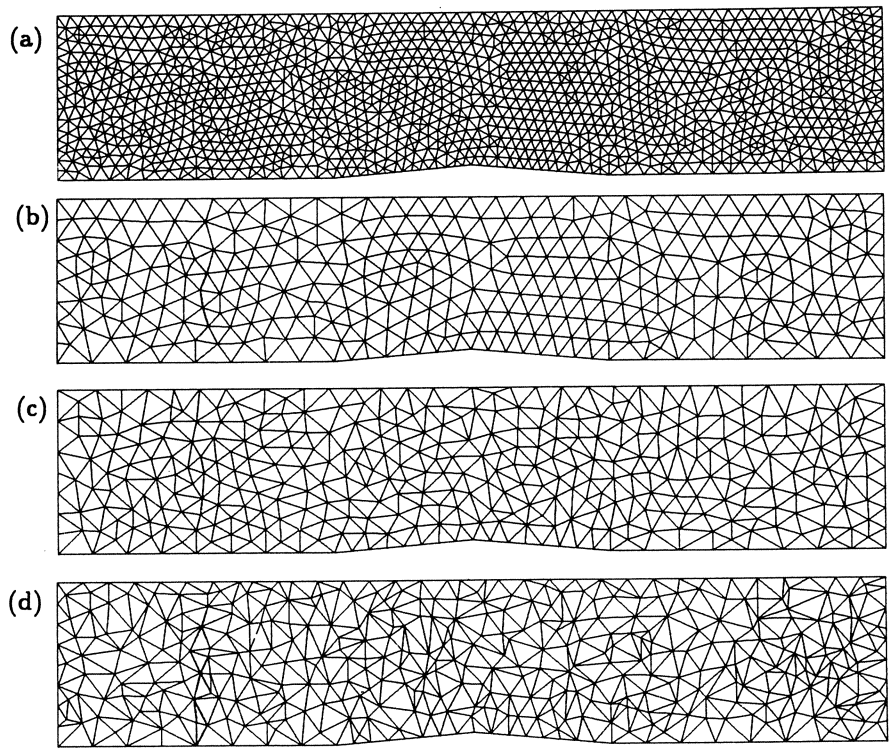


FIGURE 3. Double wedge triangular meshes. a: Final mesh (1333 nodes); b: Intermediate mesh in the generation process of mesh (a) (348 nodes); c: Mesh generated by coarsening mesh (a) [5] (365 nodes); d: Mesh generated by coarsening mesh (a) [6] (406 nodes)

intermediate meshes for use in the multigrid. When the quality of the fine mesh is improved by a smoother, the coarse grid nodes which are connected to the fine grid nodes, will also be moved. That this strategy also works, giving good quality coarse meshes, was already shown in [6, 8] and is illustrated in figure 3b.

Independent of the way the coarser meshes were generated, the property of telescoping nodes reduces the complexity of the intergrid operations. The injection of flow variables is straightforward, as is the prolongation of corrections in the telescoping nodes. For the fine grid nodes that do not appear in the coarser grid, corrections can be interpolated on the coarse grid, as in the totally unrelated meshes, or averaged on the fine grid, keeping the intergrid connectivity to a minimum. In the approach followed in this paper (figure 4), the correction in node i is the average of the corrections in nodes a, b, c, that appear on the coarser mesh.

For the restriction, more or less complex agglomeration and smoothing algorithms can be constructed. The one that is used here tries to keep the required

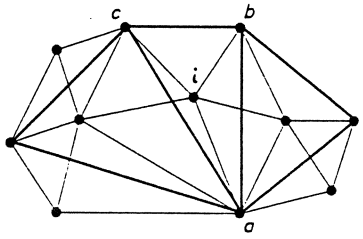


FIGURE 4. Intergrid operations on meshes with telescoping nodes.

intergrid connectivity information to a minimum. First, the residual is divided by the control volume area (2D). Then, these scaled residuals R_j are averaged or weighted on the fine grid. This averaging, which is done implicitly or explicitly has the function of a low-pass filter. The resulting quantity is then injected to the coarse grid, and multiplied with the coarse grid volume area. With $\tilde{R}_o$ the quantity in node o, calculated with R_o the residual in node o and R_j the residuals in the surrounding nodes, the explicit residual weighting is given by

$$(1 + n\epsilon)\tilde{R}_o = R_o + \epsilon \sum_{j=1}^{n} R_j, \tag{1}$$

while the implicit residual weighting is given by

$$(1 + n\epsilon)\tilde{R}_o - \epsilon \sum_{j=1}^{n} \tilde{R}_j = R_o. \tag{2}$$

Since the function of this weighting is to filter out the high frequencies, a few Jacobi sweeps (e.g. 3) are enough to calculate $\tilde{R}_o$. In the next section, the data structure is explained that allows easy implementation of the intergrid operators.

As described above, the residual restriction and correction prolongation were formulated as an operation on one grid, followed by an injection from that grid to an other grid, possibly followed by an operation on the other grid. In this way, the only intergrid operation that has to be implemented is the injection.

4 Data structure for the intergrid operations

As was indicated before, one of the design goals of the data structure is to allow easy and efficient implementation of the intergrid operations, in terms of memory and cpu. Since the code was written in Fortran, only some ideas of the object oriented philosophy could be used. Not a complete description of every object will be given here, only the relevant parts to understand the intergrid operations. One of the basic objects is the point, which is a location in space, with two coordinates.

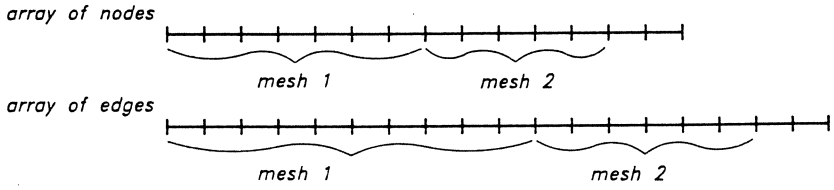


FIGURE 5. A list of nodes and a list of edges are part of the object mesh.

With the object point, the object node is built, which is a location in the computational space, i.e. flow variables will be part of a node. Two nodes can be connected by an edge. So the object edge consists of two nodes pointers. The object mesh is then formed by a set of edges together with a set of nodes. A pointer in Fortran is implemented as an index in an array. Since a lot of operations have to or can be done as a loop over edges (e.g. flux calculations, weighting functions), the edges that build up a mesh are stored consecutively in the array of edges. The same is true for the nodes that build up a mesh. So, a node and an edge can only belong to one mesh, figure 5.

This is not the case with the object point. A point, which stands for a unique location in space, can be referenced to by different nodes, and therefore belong to different meshes. The characteristic property of two meshes with telescoping nodes is that they have nodes with references to the same point. This property is used in the implementation of the injection. To do this, a vector of state variables, called trval, is added to the object point. One of the operations of the object mesh is now to copy the variables that need to be injected to trval for the points of all its nodes. If a point has received some values, a point flag trflag is set. Now the other mesh can pick up the values in trval. Of course, only flagged nodes are picked up. So, the two operations copy-to-points and copy-from-points can be performed on a set of variables that belong to the nodes of a mesh. The point flags trflag are flagged by setting their value to the mesh number. By using this type of flag no reset operation is required, only an initialization in the beginning of the program.

Of course, the suggested implementation is only one of a large number of possibilities. But it is one in which the object mesh contains no information of the other meshes. The pointers that build up the connection between two meshes are from node to point. This is a rather nice property, especially if one wants to optimize the orderings of the node and edge arrays. This is discussed in the section on orderings.

5 Mesh generation

The mesh generator that was used to construct the meshes of figure 3 and that is formulated with the edge-based data structure is built with two basic algorithms.

The first one is based on the advancing front method, developed by Tanemura et al. [9], simplified to work on a set of nodes which are already connected. The nodes have to lie on the boundary of a polygon. With this algorithm, a given, not necessarily convex, polygon can be triangulated. No extra nodes are added. The result is a constrained Delaunay triangulation of the given polygonal boundary. This algorithm was used for the generation of the initial triangulation of the domain. The domain was defined by discretizing the boundaries based on local curvature and local mesh spacing.

The second basic algorithm used in the mesh generation allows the addition of a node to an existing triangulation. The new node is connected to three or four existing nodes: three nodes if the new node lies inside a triangle, four nodes if the new node lies on an edge. The triangulation is made Delaunay by the use of a diagonal swapping algorithm [10].

To delete a node, the node (together with all the connected edges) is removed from the triangulation and then the remaining cavity is retriangulated using the first basic algorithm.

6 Orderings

In this section, orderings of the node and edge arrays of the previously described data structure are given. The orderings of the lists that are used to describe the mesh have an influence on the execution speed of algorithms. In many algorithms the bandwidth or profile of the matrix of the system to be solved, determines the amount of computation and memory required. The meshes obtained from most mesh generators have very poor natural orderings. The left side of figure 6 shows the non-zero entries associated with the *Laplacian* of the mesh of figure 3a. The Laplacian of a mesh represents the non-zero entries in the system matrix due to a discretization which involves only adjacent neighbours of the mesh.

Several algorithms exist that attempt to optimize the profile of the matrix [1]. The ordering technique that is used here is based on a Cuthill-McKee type algorithm. The heuristics behind the algorithm are very simple. The difference between the node numbers of an edge must be small, otherwise they will produce entries with a large bandwidth. The Cuthill-McKee algorithm splits up the nodes in sets. Starting from the first set (a node), the next set of nodes is defined as all the nodes connected to the nodes of the this set which have not been placed in earlier sets. The nodes of a set are ordered by increasing vertex degree. The sets are numbered consecutivly. Here, the ordering of the nodes of one set is modified compared with the original Cuthill-McKee algorithm. This is done to improve the profile with respect to the execution speed of the loop over edges, on a computer equiped with a data-cache. The algorithm is applied recursively, i.e. the sets are again divided into subsets with the same algorithm. The starting node for the first subset of a set is the one with the smallest neighbour number. The result of the

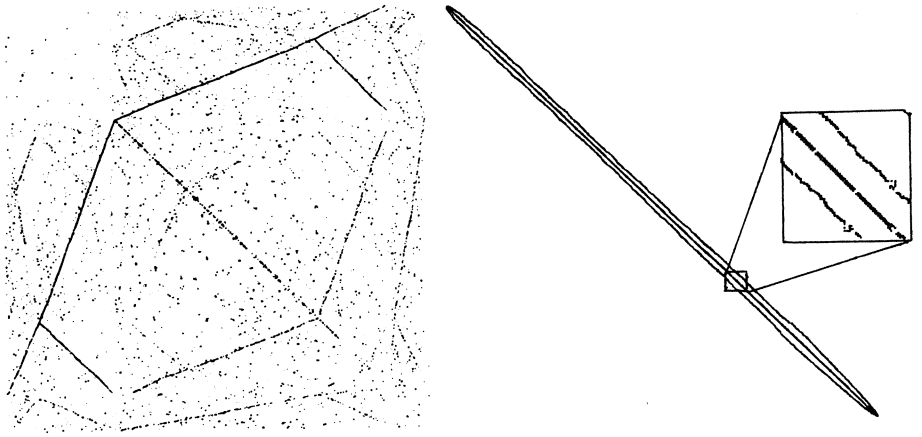


FIGURE 6. Non-zero entries in the system matrix associated with the Laplacian of the mesh, with no reordering (left) and with reordering (right).

reordering is shown in figure 6. Now the list of edges is ordered according to the minimum node number of an edge.

That this ordering improves the execution speed of a loop over the edges can be seen as follows. When the loop over the edges is done, the node numbers of an edge are close to the node numbers of the previous edge. Therefore it is likely that the data belonging to the nodes is found in the cache. The speed up due to this reordering depends on the amount of floating point calculations per edge. The speed up measured in cpu time on a IBM3090 with one processor, ranges from 5% for the first order flux-difference splitting scheme to 25% for the central scheme.

7 Conclusion

Implementation and formulation of a multigrid algorithm on unstructured meshes can be done in several ways. If meshes are constructed to minimize the overhead of intergrid operations, the mesh generation becomes more difficult. This is the case if the meshes have telescoping nodes. With the choice of a suitable data structure algorithms can be constructed to solve these problems.

In the formulation of the spacial discretization the data structure plays a leading role.

Acknowledgements

The research reported here was granted under contract IT/SC/13, as part of the Belgian National Programme for Large Scale Scientific Computing and under con-

tract IUAP/17 as part of the Belgian National Programme on Interuniversity Poles of Attraction, both initiated by the Belgian State, Prime Minister's Office, Science Policy Programming.

References

[1] T. J. Barth ,Aspects on unstructured grids and finite volume solvers for the Euler and Navier-Stokes equations, Unstructured Grid Methods for Advection Dominated FLows, *AGARD report 787*, 1992.

[2] D. J. Mavriplis ,Multigrid solution of the 2-D Euler equations on unstructured triangular meshes, *AIAA Journal*, Vol. 26, No. 7, pp. 824-831, 1988.

[3] M.-H. Lallemand, H. Steve, A. Dervieux, Unstructured multigridding by volume agglomeration: Current status, *Computers and Fluids*, Vol. 21, No. 3, pp. 397-433, 1992.

[4] V. Venkatakrishnan, D. J. Mavriplis, M. J. Berger, Unstructured multigrid through agglomeration, *Proceedings of the Sixth Copper Mountain Conference on Multigrid Methods*, to appear, 1993.

[5] H. Guillard, Node-nested multigrid with Delaunay coarsening, *INRIA Rapport de Recherche*, No. 1898, 1993.

[6] K. Riemslagh, E. Dick, A multigrid method for steady Euler equations on unstructured adaptive grids, *Proceedings of the Sixth Copper Mountain Conference on Multigrid Methods*, to appear, 1993.

[7] E. Morano, H. Guillard, A. Dervieux, M.-P. Leclerq, B. Stoufflet, Faster relaxation for non-structured multigrid with Voronoi coarsening, *Proceedings of the First European Computational Fluid Dynamics Conference*, Vol. 1, pp. 69-74, 1992.

[8] K. Riemslagh, E. Dick, A flux-difference finite volume method for steady Euler equations an adaptive unstructured grids. *Proc. of 18th ICAS Congress, Beijing*, AIAA, Washington, Vol. 1, pp. 991-999, 1992.

[9] Tanemura, M., Ogawa, T., Ogita, N (1983). A new algorithm for three-dimensional Voronoi tesselation. *Journal of Computational Physics*, *51*, 191-207.

[10] Lawson, C. L. (1972). Generation of a triangular grid with application to contour plotting. *California Institute of Technology, Jet Propulsion Laboratory*, Technical Memorandum, 299.

16

Solution of Full Potential Equation on an Airfoil by Multigrid Technique

C. Srinivasa[1]

ABSTRACT The multigrid technique is used to accelerate the convergence of flow algorithms. In this paper the technique has been used to solve the full potential equation in its non-conservative form for a symmetrical airfoil using a cartesian grid system. The solutions reported here used upto three grid levels for zero airfoil incidence and upto two levels for one degree of incidence. CPU times obtained for the two cases have been tabulated. They clearly show substantially reduced CPU times as against single grid solutions. The equivalent number of iterations for the converged solution decreases as the number of grid levels increase. The solution did not converge in some cases because the lowest level grid was very coarse. We are uncertain as to the effect of non-zero incidence on convergence, but it does seem to play a role. The computed pressure distributions are also presented.

1 Introduction

The multigrid technique is a useful tool for reducing computational times by accelerating solution convergence to flow problems[1]. This is a powerful method and experience is needed to implement the technique. Earlier, this technique was tried by the author on a one-dimensional polynomial equation[2], on the two dimensional Laplace equation[3] and later on the full potential equation for an airfoil at zero incidence upto two grid levels[4].

Deconinck and Hirsch[5] have combined the finite element approach with the multigrid technique to solve the transonic, conservative, full potential flow equation, using arbitrary body fitted coordinates. Among the examples, they considered a two-dimensional NACA 0012 airfoil at zero incidence for free stream Mach number of 0.85 and provided its convergence history and pressure distribution.

Boerstoel[6] adopted the multigrid technique for steady transonic potential flows around airfoils, using Newton iteration. The flow equations were discretized in

[1]Scientist, Computer Support and Services, National Aerospace Laboratories, BANGALORE 560017, INDIA

a fully conservative form on an approximately orthogonal grid of O-type. The grid was aligned along the airfoil. Numerical experiments were carried out on a NACA 0012 airfoil for different Mach numbers and incidences. Pressure distributions for the cases considered and the convergence analysis with reference to R_{max} (Residue Max) are reported in his paper.

Becker[7] has applied the multigrid technique to study the airfoils in two dimensional transonic flow using the full potential, non-conservative, flow equation with emphasis on convergence factors. Convergence factors for different Mach numbers and zero incidence for a NACA 0012 airfoil were tabulated.

This paper looks at the finite difference multigrid method using cartesian grids with column SLOR relaxations, to solve the full potential equation in non-conservative form at subsonic speed, around airfoils. Our emphasis was on CPU times obtained and the convergence history of the solution. The multigrid technique is based on the corrections applied to the solutions of the differential equation on a fine grid, by an approximated differential equation of similar type, on a coarse grid. We have used the Full Approximation Storage developed by Brandt[1]. The software for the flow problem was taken from NAL SOFFTS library[9].

The multigrid technique attempts to remove the high and low frequency errors during the computations to obtain the solution of the given equation. The high frequency is nothing but short wavelength errors, which can be effectively smoothened on a fine grid with the SLOR scheme. The low frequency or long wavelength errors are smoothened on a coarse grid more efficiently where the number of points are less.

To start the multigrid computations, some relaxation sweeps are performed on the finest grid and later the computations are moved to the coarser grid depending on the convergence rate. The computations are shifted to a still coarser grid, if there is no convergence on the higher level grid. After convergence is obtained on the coarse grid, computations are performed on the fine grid after updating all the grid point values. This continues till convergence is obtained on the finest grid.

The finite difference equations to be solved are defined on a cartesian grid[8]. The $x - y$ physical plane (Figure 1) has been stretched or transformed to $\xi - \eta$ computational plane. The tridiagonal equation systems used in line relaxation sweeps are solved by a tri-diagonal solver. Central difference scheme has been used for flow computations since only subsonic (elliptic) flows are being considered. The computer used was the Sperry 1100 at NAL.

2 Description

The full potential equation in non-conservative form is given by

$$(a^2 - u^2)\ \phi_{xx} - 2\,u\,v\ \ \phi_{xy} + (a^2 - v^2)\ \phi_{yy} = 0$$

$$\text{where} \quad a = \text{acoustic speed} = \left[1 - \frac{\gamma - 1}{2}\left(u^2 + v^2 - M_\infty^2\right)\right]^{1/2}$$

u = X − component of velocity = $M_\infty(\cos\alpha + \phi_x)$

v = Y − component of velocity = $M_\infty(\sin\alpha + \phi_y)$

ϕ = perturbation velocity potential

ν = specific heat ratio

M_∞ = free stream Mach number

α = incidence

θ = $\tan^{-1}(y/x)$

x, y = cartesian co-ordinates

The appropriate boundary conditions are

1. at the airfoil surface = $(dy/dx) = (v/u)$,
2. at the far field,

$$\phi = -\Gamma/2\pi \ \tan^{-1}\left(\sqrt{1 - M_\infty^2}\ \tan(\theta - \alpha)\right)$$

where Γ = circulation of the flow determined by the change in potential across the Kutta-Joukowski cut at the trailing edge of the airfoil, i.e.

$$\Gamma = (\phi_{y=0^+} - \phi_{y=0^-})_{\text{trailing edge}}$$

The finite difference equation of the full potential equation considered is

$$(a^2 - u^2)\, f\, \frac{f_{i+1/2}\ (\phi_{i+1,j} - \overline{\phi}_{ij}) - f_{i-1/2}\ (\overline{\phi}_{ij} - \phi_{i-1,j})}{\Delta\xi^2}$$

$$-u\, v\, f\, g\, \frac{\phi^+_{i-1,j-1} - \phi^+_{i-1,j+1} - \phi_{i+1,j-1} + \phi_{i+1,j+1}}{2(\Delta\xi)\ (\Delta\eta)}$$

$$+(a^2 - v^2)\, g\, \frac{g_{j+1/2}\ (\phi^+_{i,j+1} - \phi^+_{ij}) - g_{j-1/2}\ (\phi^+_{ij} - \phi^+_{i,j-1})}{\Delta\eta^2} = 0.0$$

$$\text{where} \quad \overline{\phi}_{ij} = \frac{1}{\omega}\,\phi^+_{ij} + \left(1 - \frac{1}{\omega}\right)\,\phi_{ij}$$

w = relaxation factor

superscript+ = current relaxation sweep values

f and g = coordinate stretching functions

The L2 norm of the residues of the above equation is taken as the measure to check for convergence against the tolerance assumed. The particular problem studied is the NACA 0012 airfoil, at a free stream Mach number of 0.63 and at

incidences 0 and 1 degree. The grid levels used were as follows;

level one	:	(97×49)	Finest grid
level two	:	(49×25)	Medium grid
level three	:	(25×13)	Coarse grid
level four	:	(13×7)	Coarsest grid

The airfoil at 0 degree used levels one to three, and at 1 degree, levels one and two.

3 Results and Discussions

The results obtained for the airfoil at 0 degree show that the technique reduces CPU time considerably when two- and three-levels of grids are used as compared to a single level, as expected. The CPU times obtained for the different levels are shown in Table 1. The plot of Log (ERRR) vs. equivalent number of iterations shows clearly that when the number of levels was increased, the equivalent number of iterations reduce correspondingly. The convergence history is shown in Figure 2. Here ERRR is nothing but the residue norm (L2 norm) as referred earlier. Equivalent number of iterations is defined as approximately the number of iterations on the coarse grid that is equal to one iteration on the fine grid, based on the grid size or the number of points.

When the number of grid levels was increased to four, we encountered non-convergence.This may have been due to the coarseness of the fourth level grid size (13×7).

The results for the airfoil at 1 degree showed that the CPU time is once again reduced considerably. The CPU time is tabulated in Table 2. The convergence history is shown in Figure 3. Here also it is evident that the equivalent number of iterations for two grid levels is less than that for single level grid.

At higher incidences, solutions failed to converge. We believe the reasons are the same as those mentioned in [6], i.e. large changes in the solution at the airfoil leading edge give rise to an increase of residue norms.

Also, keeping the incidence to one degree and increasing the number of grid levels to three failed to give a converged solution. The third level grid is perhaps unable to provide proper corrections to the next higher level and causes non-convergence.

The pressure distributions on the airfoil surface computed for zero incidence and one degree incidence agree at all levels. Figure 4 depicts the pressure distribution for zero incidence and Figure 5 for one degree incidence.

In our opinion, if the finite difference scheme is replaced by a finite element scheme,it may be possible to get convergence at coarser grid levels [5].

4 Conclusions

Multigrid method reduces the CPU time as compared to a single level solution. The equivalent number of iterations for a multigrid solution is less than that of a single level grid.

References

1. Brandt A. : 'Multi-level adaptive solutions to boundary value problems', Mathematics of Computation, Volume 31, Number 138, p.333-389, April 1977.

2. Mathur J.S., Srinivasa C. : 'Solution of a second order ODE by a multigrid method', NAL-PD-AE-8815, September 1988.

3. Srinivasa C. : 'Multigrid technique to solve the Laplace equation on the rectangle by point and line relaxations', NAL-PD- CF-9109, May 1991.

4. Srinivasa C. : 'Multigrid technique for solving the Full Potential Equation on an airfoil in cartesian grid', NAL-PD-CC- 9206, August 1992.

5. Deconinck H., Hirsch C. : 'A multigrid finite element method for the transonic potential equation', Lecture Notes in Mathematics, Multigrid Methods, Proceedings, Koln-Porz, pp.387-409, 1981.

6. Boerstoel J.W. : 'A multigrid algorithm for steady transonic potential flows around airfoils using Newton iteration', pp.151-172, Proceedings of the Symposium held at Ames Research Centre, Moffett field, California, October 21-22, 1981 (NASA Conf. Publication 2202).

7. Becker K. : 'A multigrid solver for two dimensional transonic full potential flow calculations', Proceedings of the Second European Conference on Multigrid Methods, pp.25-34, Cologne, October 1-4, 1985.

8. Leland A. Carlson : 'Transonic airfoil flow field analysis using cartesian coordinates', NASA-CR-2577, August 1975.

9. Rangarajan R., Desai S.S. and Ramaswamy M.A. : '2D-Transonic analysis and Design programme in cartesian Coordinates (Trade-1), NAL-AE-TM-6-83, September 1983.

TABLE 1. COMPARISON OF CPU TIMES

NACA 0012 Airfoil

ALPHA = 0.0 M = 0.63

Level	Grid	CPU time
one	97 × 49	9.08469 min
two	49 × 25	3.35482 min
three	25 × 13	2.34815 min

TABLE 2. COMPARISON OF CPU TIMES

NACA 0012 Airfoil

ALPHA = 1.0 M = 0.63

Level	Grid	CPU time
one	97 × 49	10.18599 min
two	49 × 25	5.45800 min

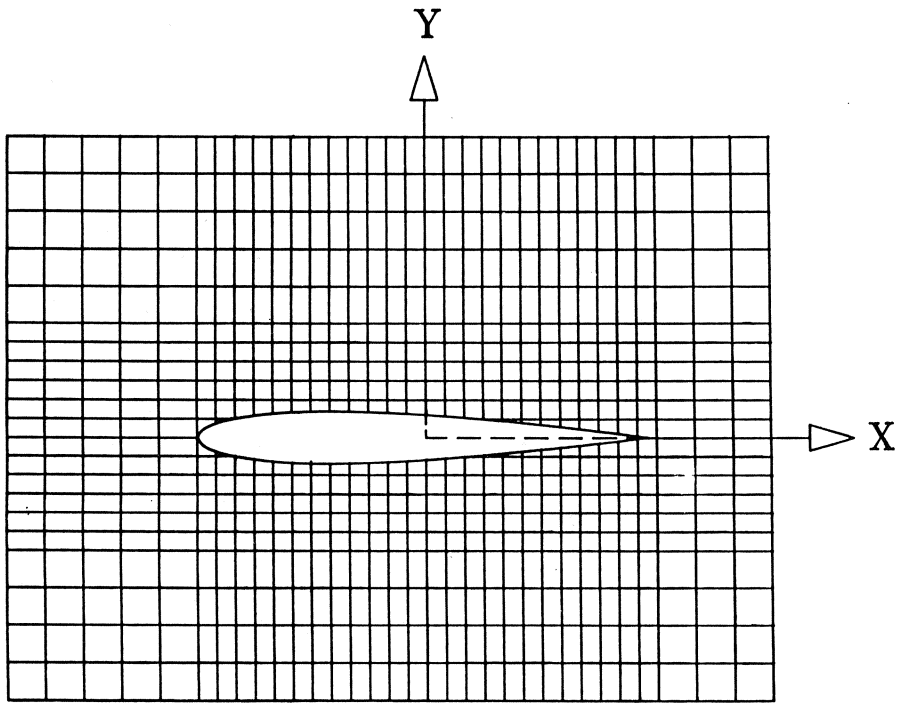


FIGURE 1. The Cartesian grid system

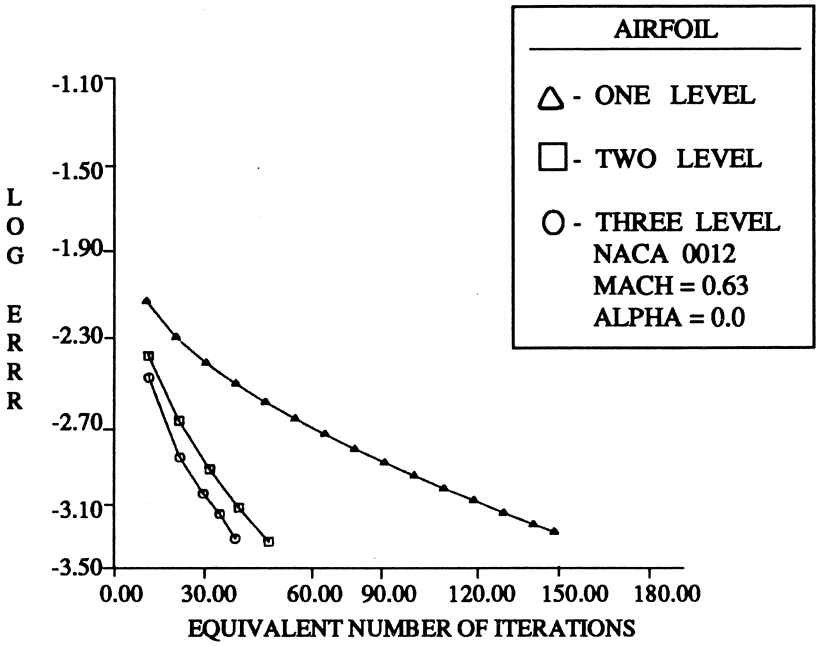


FIGURE 2. Convergence speed

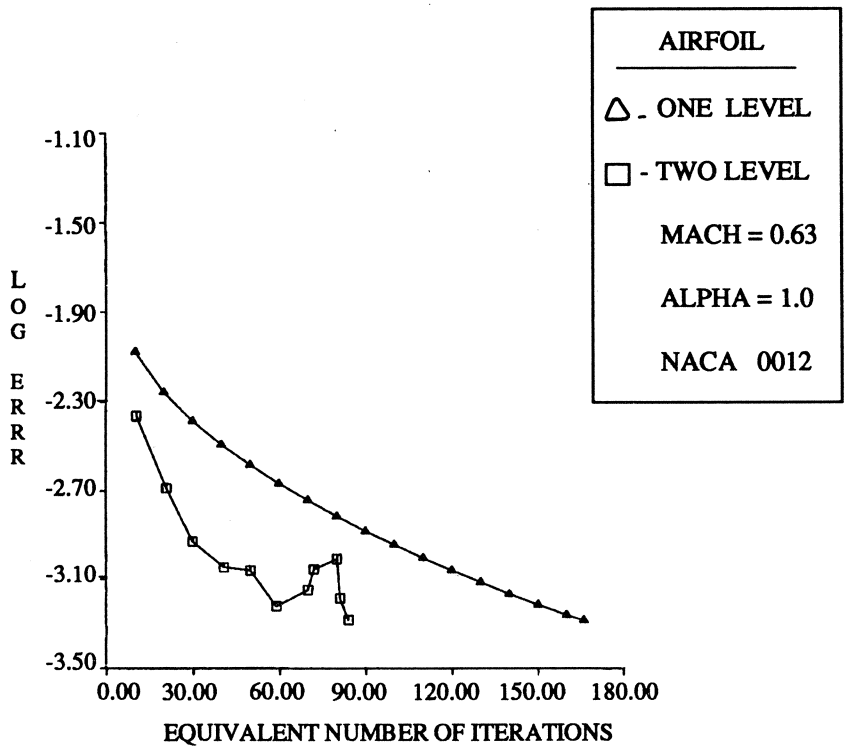


FIGURE 3. Convergence speed

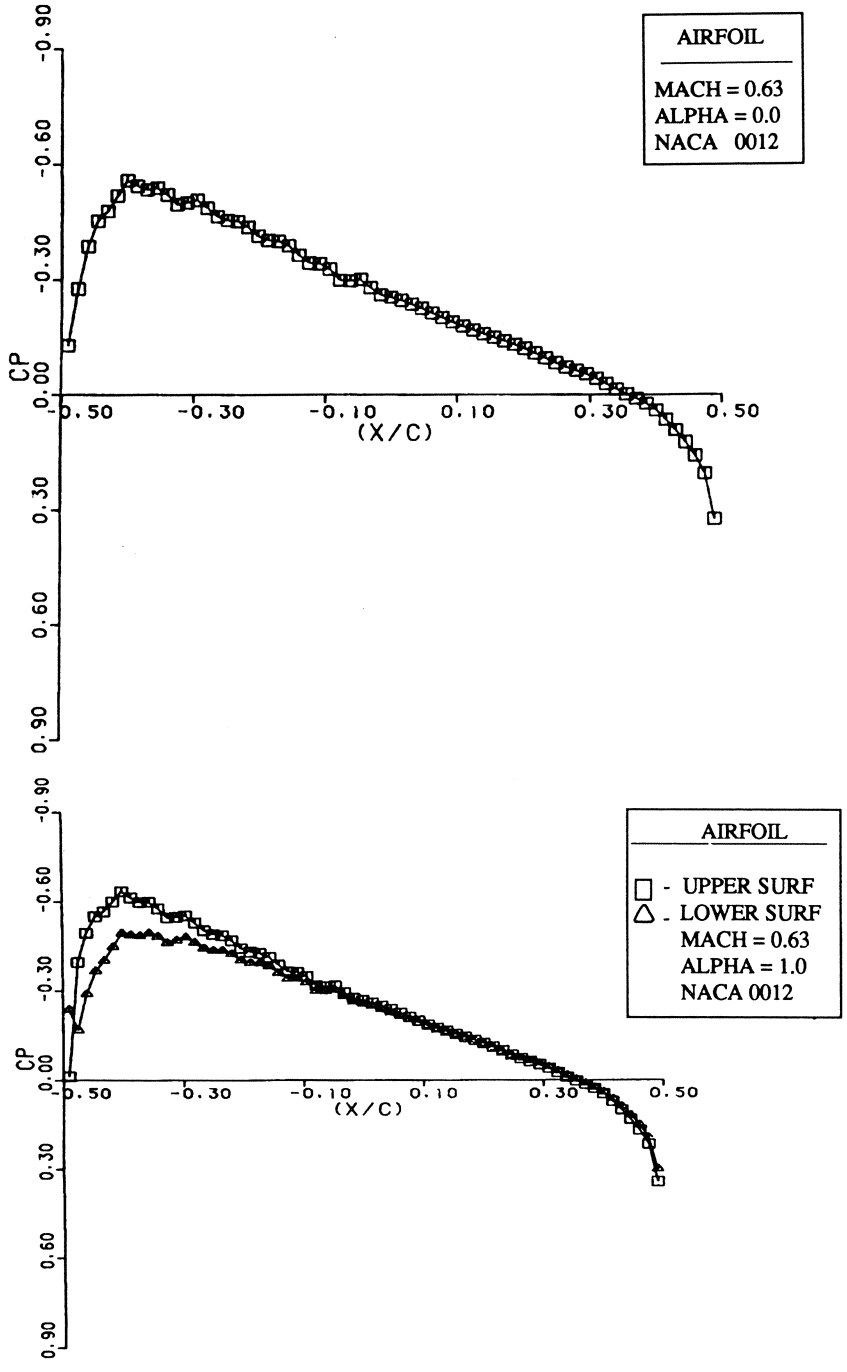


FIGURE 4. Pressure distribution

17

Multigrid for the Incompressible Navier-Stokes Equations on Staggered Grids in General Coordinates

S. Zeng and P. Wesseling[1]

ABSTRACT Galerkin coarse grid approximation (GCA) in multigrid methods is investigated for the incompressible Navier-Stokes equations in general coordinates. An efficient algorithm performing GCA is presented. The behavior of coarse grid matrices is studied under GCA with different transfer operators. For square and L-shaped driven cavity problems, the performance of the multigrid method using different combinations of transfer operators for the computation of coarse grid matrices and of coarse grid correction is investigated. Further computations are carried out in general coordinates for a channel flow problem with backward facing step in three dimensions.

1 Introduction

In multigrid methods for partial differential equations, a sequence of computational grids is used simultaneously. The finest grid discretization has to be approximated on the coarse grids. There are two ways of doing this: by discretization of the differential problem (discretization coarse grid approximation or DCA) or by algebraic manipulation of the finest grid discretization; one way of doing this called Galerkin coarse grid approximation or GCA. Which of the two is best depends on circumstances. GCA is practically limited to linear problems, so that iterations on nonlinearities have to be taken outside multigrid, which is used as a linear systems solver; more (but perhaps cheaper, especially on vector computers) iterations may be required than with the nonlinear multigrid method. With GCA the discretization and solution phases are clearly separated, which leads to better structured, maintainable and reusable software; on the other hand, the future may belong to intertwined discretization and solution in adaptive computing strategies. DCA may be inaccurate on the very coarse grids that normally occur in multigrid,

[1]Faculty of Technical Mathematics and Informatics, Delft University of Technology, P.O. Box 5031, 2600 GA Delft, The Netherlands

leading to bad multigrid convergence; an example is given in [15]. For equations with discontinuous coefficients GCA is the method of choice ([1], [5], [8], [17], [18]). For systems of equations efficient and correct implementation of GCA requires some care, and here this paper intends to contribute, for the particular case of the incompressible Navier-Stokes equations in general non-orthogonal boundary-fitted coordinates discretized with the finite volume method on a staggered grid. This is a particular case where incorporation of the discretization method inside multigrid causes practical problems, because discretization in general coordinates with staggered arrangement of unknowns is a complicated affair; for more on this see [10], [13], [19] and references quoted there. Improvements in discretization and boundary condition implementation lead to code changes in many places in a non-linear multigrid code employing DCA, and vectorization potential of the code is not great. Nevertheless, this approach is also pursued, see [11], but here we concentrate on GCA.

2 The Incompressible Navier-Stokes Equations in General Coordinates

Most of our considerations carry over to general systems of differential equations, but we will focus primarily on the incompressible Navier-Stokes equations, describing incompressible viscous flows. In order to be able to handle flows in complicated geometries, general boundary-fitted coordinates are used. In general coordinates, the incompressible Navier-Stokes equations are given in standard tensor notation and can be found in [10], [13] and [19]. With Euler backward time discretization and Newton linearization of the nonlinear terms, a discrete system on a staggered grid of the following form is obtained:

$$\begin{pmatrix} \mathbf{u}^1 \\ \mathbf{u}^2 \end{pmatrix}^{n+1} - \begin{pmatrix} \mathbf{u}^1 \\ \mathbf{u}^2 \end{pmatrix}^{n} = \begin{pmatrix} \mathbf{f}^1 \\ \mathbf{f}^2 \end{pmatrix} - \begin{pmatrix} \mathbf{A}^{11} & \mathbf{A}^{12} & \mathbf{A}^{13} \\ \mathbf{A}^{21} & \mathbf{A}^{22} & \mathbf{A}^{23} \end{pmatrix} \begin{pmatrix} \mathbf{u}^1 \\ \mathbf{u}^2 \\ \mathbf{u}^3 \end{pmatrix}^{n+1}, \quad (1)$$

$$\begin{pmatrix} \mathbf{A}^{31} & \mathbf{A}^{32} \end{pmatrix} \begin{pmatrix} \mathbf{u}^1 \\ \mathbf{u}^2 \end{pmatrix}^{n+1} = 0, \quad (2)$$

where $\mathbf{u}^1$, $\mathbf{u}^2$ and $\mathbf{u}^3$ approximate the variables related to the contravariant velocities and the pressure, respectively. In the instationary case, the spatial derivatives are approximated with central differences; in the stationary case, obtained by replacing the left-hand-side of (1) by zero, the convection terms are approximated by the hybrid scheme ([14]). Furthermore, in the stationary case we use Picard linearization instead of Newton. The structure of the discrete operator is given by

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}^{11} & \mathbf{A}^{12} & \mathbf{A}^{13} \\ \mathbf{A}^{21} & \mathbf{A}^{22} & \mathbf{A}^{23} \\ \mathbf{A}^{31} & \mathbf{A}^{32} & 0 \end{pmatrix}. \quad (3)$$

In stencil notation (for more on this, see [18]), the action of a linear operator $\mathbf{A}$ can be represented by

$$(\mathbf{Au})_i = \sum_j \mathbf{A}(i,j)\mathbf{u}_{i+j}, \tag{4}$$

where $i = (i_1, i_2, \cdots, i_d) \in \mathcal{G} \subset \mathcal{Z}^d$ are grid point indices; d is the number of space dimensions of $\mathcal{G}$; $\mathcal{G}$ is conveniently identified with its set of grid point indices. The operator $\mathbf{A}$ is fully specified by the numbers $\mathbf{A}(i,j)$. The structure $\mathcal{S}_A$ of $\mathbf{A}$ is defined by

$$\mathcal{S}_A = \{j \in \mathcal{Z}^d : \exists i \in \mathcal{G} | \mathbf{A}(i,j) \neq 0\}. \tag{5}$$

Graphically, $\mathbf{A}$ can be denoted as, for example for the approximation of the Laplace equation in two dimensions:

$$[\mathbf{A}] = h^{-2} \begin{bmatrix} & -1 & \\ -1 & \underline{4} & -1 \\ & -1 & \end{bmatrix} \quad \text{or} \quad [\mathbf{A}] = \begin{bmatrix} & * & \\ * & \underline{*} & * \\ & * & \end{bmatrix}. \tag{6}$$

where the elements with an underscore correspond to $j = 0$. The arrays in (6) and the numbers $\mathbf{A}(i,j)$ are all referred to as the stencil $[\mathbf{A}]$ of $\mathbf{A}$; this is not found to lead to confusion. Similarly, we have, for the Navier-Stokes equations

$$[\mathbf{A}^{11}] = \begin{bmatrix} * & * & * \\ * & \underline{*} & * \\ * & * & * \end{bmatrix}, \quad [\mathbf{A}^{12}] = \begin{bmatrix} * & * & * & * \\ * & * & \underline{*} & * \end{bmatrix}, \quad [\mathbf{A}^{13}] = \begin{bmatrix} * & * \\ * & \underline{*} \\ * & * \end{bmatrix},$$

$$[\mathbf{A}^{21}] = \begin{bmatrix} * & * \\ \underline{*} & * \\ * & * \\ * & * \end{bmatrix}, \quad [\mathbf{A}^{22}] = \begin{bmatrix} * & * & * \\ * & \underline{*} & * \\ * & * & * \end{bmatrix}, \quad [\mathbf{A}^{23}] = \begin{bmatrix} * & \underline{*} & * \\ * & * & * \end{bmatrix}, \tag{7}$$

$$[\mathbf{A}^{31}] = \begin{bmatrix} \underline{*} & * \end{bmatrix}, \quad [\mathbf{A}^{32}] = \begin{bmatrix} * \\ \underline{*} \end{bmatrix}, \tag{8}$$

Extension to three dimensions is straightforward.

Let us explain something about GCA. For our present purpose it suffices to consider two-grid methods. Let there be given a fine grid $\mathcal{G}$ and a coarse grid $\bar{\mathcal{G}}$. Let $\mathbf{U}$ be the space of fine grid functions, and $\bar{\mathbf{U}}$ the space of coarse grid functions:$\mathbf{U} = \{\mathbf{u} : \mathcal{G} \longmapsto \mathcal{R}\}$, $\bar{\mathbf{U}} = \{\bar{\mathbf{u}} : \bar{\mathcal{G}} \longmapsto \mathcal{R}\}$. After discretization of the partial differential equation under consideration and linearization a linear system of algebraic equations on the fine grid $\mathcal{G}$ is obtained, denoted by

$$\mathbf{Au} = \mathbf{b}. \tag{9}$$

This linear problem can be solved with a linear multigrid method. With GCA, $\bar{\mathbf{A}}$ is given by

$$\bar{\mathbf{A}} = \mathbf{RAP}, \tag{10}$$

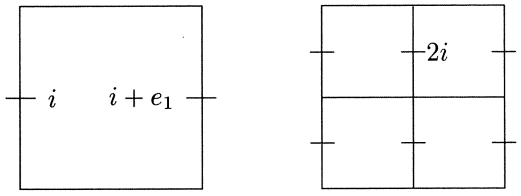


FIGURE 1. A cell of $\bar{\mathcal{G}}$ and the corresponding four cells of $\mathcal{G}$; the grid points for $\mathbf{u}^1$ and $\bar{\mathbf{u}}^1$ are indicated by —.

where $\mathbf{P} : \bar{\mathbf{U}} \longmapsto \mathbf{U}$ and $\mathbf{R} : \mathbf{U} \longmapsto \bar{\mathbf{U}}$ are a prolongation and a restriction operator. Stencil notation of $\mathbf{R}$ and $\mathbf{P}$ can be given as

$$(\mathbf{Ru})_i = \sum_j \mathbf{R}(i,j)\mathbf{u}_{2i+j}, \quad (\mathbf{P}\bar{\mathbf{u}})_i = \sum_j \mathbf{P}^*(j, i-2j)\bar{\mathbf{u}}_j \tag{11}$$

with $\mathbf{P}^*$ is the adjoint of $\mathbf{P}$.

3 Prolongation and Restriction

We construct coarse grids by taking unions of fine grid cells. In figure 1 we present a cell of the grid $\bar{\mathcal{G}}$ and the corresponding four cells of $\mathcal{G}$. Let $\mathbf{P}^1 : \bar{\mathbf{U}}^1 \longmapsto \mathbf{U}^1$ be defined by bilinear interpolation. This means, for example, that u^1_{2i} is obtained from $\bar{u}^1_i, \bar{u}^1_{i+e_1}, \bar{u}^1_{i+e_2}, \bar{u}^1_{i+e_1+e_2}$, with $e_1 = (1,0)$, $e_2 = (0,1)$. So one obtains:

$$[\mathbf{P}^{1*}] = \frac{1}{8}\begin{bmatrix} nw & 2n & ne \\ (4-n)w & 2(4-n) & (4-n)e \\ (4-s)w & 2(4-s) & (4-s)e \\ sw & 2s & se \end{bmatrix}, \tag{12}$$

$$[\mathbf{P}^{2*}] = \frac{1}{8}\begin{bmatrix} nw & (4-w)n & (4-e)n & ne \\ 2w & 2(4-w) & 2(4-e) & 2e \\ sw & (4-w)s & (4-e)s & se \end{bmatrix}. \tag{13}$$

Here $n = 0$ on the "north" boundary and $n = 1$ elsewhere, and similarly for s, e and w on the south etc. boundaries. Another possibility, to be called hybrid interpolation, is to use linear interpolation in ξ^1 but zeroth order interpolation in ξ^2 or vice versa, resulting in

$$\mathbf{P}^{1*} = \frac{1}{2}\begin{bmatrix} w & 2 & e \\ w & 2 & e \end{bmatrix}, \quad \mathbf{P}^{2*} = \frac{1}{2}\begin{bmatrix} n & n \\ 2 & 2 \\ s & s \end{bmatrix}. \tag{14}$$

For prolongation $\mathbf{P}^{3*}$ of $\bar{\mathbf{u}}^3$ we use zeroth order interpolation

$$[\mathbf{P}^{3*}] = \begin{bmatrix} 1 & \frac{1}{1} \\ 1 & 1 \end{bmatrix}. \tag{15}$$

Restriction is applied to residuals, and is performed by averaging the residuals in the fine grid staggered volumes that are (partly) contained in the coarse grid staggered finite volume under consideration, weighted with the part of the fine grid finite volume that is included. This leads to $\mathbf{R}^m = \mathbf{P}^{m*}$, $m = 1, 2$, with $\mathbf{P}^{m*}$ given by (14).

Prolongation and restriction should be sufficiently accurate. A well-known rule for this is as follows ([2], [6]). Let the order m_P of a prolongation be defined as the highest degree of polynomials that are interpolated exactly, plus one. The order m_R of a restriction is one plus the highest degree of polynomials that are interpolated exactly by $\mathbf{R}^*$ after suitable scaling. Then the rule for a single differential equation of order M is

$$m_P + m_R > M. \tag{16}$$

This is a sufficient condition. In [16] a numerical example is given of slow multigrid convergence in a case where

$$m_P + m_R = M. \tag{17}$$

What is really needed is that the coarse grid correction does not amplify short wavelength error components. In [7] it is shown that (17) is necessary and sufficient with a different definition of m_P and m_R, referring to short wavelength behaviour. Here we refrain from further analysis. The extension to systems would seem to be (cf. [6], section 11.1.1) that if $\mathbf{A}^{mn}$ is a discretization of a differential operator of order M^{mn}, then one should have at least

$$m_{P^n} + m_{R^m} \geq M^{mn} \tag{18}$$

or better still

$$m_{P^n} + m_{R^m} > M^{mn}. \tag{19}$$

In the Navier-Stokes case we have $M^{11} = M^{22} = 2$, $M^{12} \leq 2$, $M^{21} \leq 2$ (depending on circumstances: Cartesian or general coordinates), $M^{3n} = M^{n3} = 1$. For (12) and (13) we have $m_P = 2$; for (14) and (15) we have $m_P = 1$. By numerical experiments we have determined suitable choices for $\mathbf{P}$ and $\mathbf{R}$, that will be discussed later.

4 Efficient Programming of RAP

A generalization of the GCA of $\mathbf{A}^{mn}$ is

$$\bar{\mathbf{A}}^{mn} = \mathbf{R}^{mn}\mathbf{A}^{mn}\mathbf{P}^{mn}, \tag{20}$$

with $\mathbf{R}^{mn} : \mathbf{U}^m \longmapsto \bar{\mathbf{U}}^m$, $\mathbf{P}^{mn} : \bar{\mathbf{U}}^n \longmapsto \mathbf{U}^n$. We will now discuss algorithmic details of (20). For this purpose, the superscripts can be deleted.

An algorithm for determining $\mathcal{S}_{\bar{A}}$ proposed in [18] may be used. In stencil notation we have

$$\bar{\mathbf{A}}(i,n) = \sum_{m\in\mathcal{S}_R} \sum_{q\in\mathcal{S}_{P^*}} \mathbf{R}(i,m)\mathbf{A}(2i+m, q+2n-m)\mathbf{P}^*(i+n,q). \tag{21}$$

Equation (21) leads to nested do-loops over the d-tuples i, n, m and q. For high computing efficiency the longest loop, i.e. the loop over $\bar{\mathcal{G}}$, should be the innermost loop. Tests on whether $2i+m \in \mathcal{G}$ and $i+n \in \bar{\mathcal{G}}$ should not occur in the inner loop for vectorization. Therefore the correct set (called $\bar{\mathcal{G}}_2$) over which i varies is determined beforehand. Furthermore, $\mathbf{R}(i,m)$ and $\mathbf{P}^*(i,m)$ are not stored because they are constant in the interior and change only near boundaries. We therefore divide $\bar{\mathcal{G}}_2$ into parts $\bar{\mathcal{G}}_2(b)$, $b \in \mathcal{B}$ with $\mathcal{B}$ some index set, in each of which $\mathbf{R}(i,m)$ and $\mathbf{P}^*(i+n,q)$ are constant; their values will be denoted by $\mathbf{R}_b(m)$ and $\mathbf{P}^*_b(n,q)$. This leads to the following algorithm:

Algorithm **RAP**
comment Calculation of $\bar{\mathbf{A}} = \mathbf{RAP}$
begin $\bar{\mathbf{A}} = 0$
  for $m \in \mathcal{S}_R$ **do**
    $\bar{\mathcal{G}}_1 = \{i \in \bar{\mathcal{G}} : 2i+m \in \mathcal{G}\}$
    for $n \in \mathcal{S}_{\bar{A}}$ **do**
      $\bar{\mathcal{G}}_2 = \{i \in \bar{\mathcal{G}}_1 : i+n \in \bar{\mathcal{G}}\}$
      for $q \in \mathcal{S}_{P^*}$ **do**
        $k = q+2n-m$
        if $k \in \mathcal{S}_A$ **then**
          for $b \in \mathcal{B}$ **do**
            $\mu = \mathbf{R}_b(m)\mathbf{P}^*_b(n,q)$
            for $i \in \bar{\mathcal{G}}_2(b)$ **do**
              $\bar{\mathbf{A}}(i,n) = \bar{\mathbf{A}}(i,n) + \mu\mathbf{A}(2i+m,k)$
          od od
        end if
  od od od
end RAP

It is necessary to be more specific about the partitioning $\bar{\mathcal{G}}_2(b)$ of $\bar{\mathcal{G}}_2$ and about the determination of $\mathbf{R}_b(m)$ and $\mathbf{P}^*_b(n,q)$. Let

$$\bar{\mathcal{G}}_2 = \prod_{c=1}^{d} \{ib(c), ib(c)+1, \cdots, ie(c)\}. \tag{22}$$

Define

$$\bar{\mathcal{G}}_2(b) = \prod_{c=1}^{d} \bar{\mathcal{G}}_2(b_c, c), \quad b = (b_1, b_2, \cdots, b_d),\ b_c = -1, 0, 1 \tag{23}$$

with $\bar{\mathcal{G}}_2(-1,c) = ib(c)$, $\bar{\mathcal{G}}_2(1,c) = ie(c)$, $\bar{\mathcal{G}}_2(0,c) = \{ib(c)+1, ib(c)+2, \cdots, ie(c)-1\}$. It is easily seen that $\bar{\mathcal{G}}_2 = \bigcup_{b\in\mathcal{B}} \bar{\mathcal{G}}_2(b)$ with $\mathcal{B} = \prod^d \{-1,0,1\}$. Next we assume that

$\mathbf{R}$ and $\mathbf{P}^*$ are constant (but perhaps different) on the sides, corners and interior of $\bar{\mathcal{G}}$. All prolongations and restrictions discussed in section 3 satisfy this assumption. Since $\bar{\mathcal{G}}_2 \subset \bar{\mathcal{G}}$ the same is true for $\bar{\mathcal{G}}_2(b)$. For $\mathbf{R}_b(m)$ one simply checks whether $\bar{\mathcal{G}}_2(b)$ is in the interior or on a boundary of $\bar{\mathcal{G}}$, and selects the corresponding value of $\mathbf{R}(i,m)$ accordingly. With $i \in \bar{\mathcal{G}}_2(b)$, $i+n$ is in a part of $\bar{\mathcal{G}}$ where $\mathbf{P}^*(i+n,q)$ is constant, as is seen after some reflection; this is the value selected for $\mathbf{P}_b^*(n,q)$.

Extension to systems of differential equations is done block-by-block, according to (20); this needs no further discussion.

5 Properties of the Coarse Grid Operator

For efficient multigrid convergence the coarse grid should have the approximation property, as defined and studied in [6] for the case of a single differential equation; for an elementary introduction see [18]. For systems of equations the approximation property is analyzed in [20]. For the Stokes equations discretized on a staggered grid as described before it is shown in [20] that with GCA we have the approximation property if $\mathbf{P}$ corresponds to biquadratic interpolation, and if $\mathbf{R} = \mathbf{P}^*$. However, our numerical experience indicates that this condition (much stronger than (19)) is not necessary. This is fortunate, because with these transfer operators the structure of the stencil of the operator grows under GCA, leading to a superlinear dependence of computing work on number of unknowns.

The structure of $\bar{\mathbf{A}}^{mn}$ as given by (20) with $\mathbf{A}^{mn}$ given by (7) and (8) is as follows. We choose $\mathbf{R}^{mn} = \mathbf{R}^m = \mathbf{P}^{m*}$, $m, n = 1,2,3$ with $\mathbf{P}^{m*}$ given by (14) and (15). This means that $m_R = 1$. Next, $\mathbf{P}^{m3}$, $m = 1,2$ is given by (15), and $\mathbf{P}^{3m}$, $m = 1,2$ is given by (14). This gives $m_P = 1$. For $\mathbf{P}^{mn}$, $m, n = 1,2$ we choose $\mathbf{P}^{mn} = \mathbf{P}^n$, with two possibilities: bilinear or hybrid interpolation. In the bilinear case $\mathbf{P}^n$ is given by (12), (13), giving $m_P = 2$; in the hybrid case $\mathbf{P}^n$ is given by (14), giving $m_P = 1$. We find that with all of these choices the structure of $\bar{\mathbf{A}}^{mn}$ equals that of $\mathbf{A}^{mn}$.

In order to realize efficient smoothing it is desirable that the operators $\mathbf{A}$ and $\bar{\mathbf{A}}$ correspond to M-matrices. In the Navier-Stokes case the structure of the system to be solved is given by (7) and (8). It is desirable that $\mathbf{A}^{11}$ and $\mathbf{A}^{22}$ correspond to M-matrices. This can be realized by upwind or hybrid differencing. It is also desirable that the coarse grid approximations $\bar{\mathbf{A}}^{11}$ and $\bar{\mathbf{A}}^{22}$ correspond to M-matrices, or retain at least as much diagonal dominance as possible. With GCA according to (20) this depends on $\mathbf{R}^{mm}$ and $\mathbf{P}^{mm}$. We will now investigate this. Neglecting the viscous term (which brings the operator closer to the M-matrix property), typical upwind stencils for $\mathbf{A}^{11}$ are

$$[\mathbf{A}^{11}] = \begin{bmatrix} -1 & \underline{1} \end{bmatrix}, \quad [\mathbf{A}^{11}] = \begin{bmatrix} \underline{1} \\ -1 \end{bmatrix} \tag{24}$$

corresponding to a horizontal and a vertical flow direction, respectively. By nu-

merical experimentation we find that m-fold GCA according to

$$\mathbf{A}^{11(m)} = \mathbf{R}^{11}\mathbf{A}^{11(m-1)}\mathbf{P}^{11}, \ \mathbf{A}^{11(0)} = \mathbf{A}^{11} \tag{25}$$

with $\mathbf{P}^{11}$ given by the first equation of (14) and $\mathbf{R}^{11} = \mathbf{P}^{11*}$ results in (in the interior)

$$[\mathbf{A}^{11(m)}] = \frac{1}{2}\begin{bmatrix} -1 & \underline{2} & -1 \end{bmatrix} + \frac{2^m}{2}\begin{bmatrix} -1 & \underline{0} & 1 \end{bmatrix} \tag{26}$$

in the first case of (24) and in

$$[\mathbf{A}^{11(m)}] = \frac{2^m}{6}\begin{bmatrix} 0 & 0 & 0 \\ 1 & \underline{4} & 1 \\ -1 & -4 & -1 \end{bmatrix} + \frac{2^{-m}}{6}\begin{bmatrix} 0 & 0 & 0 \\ -1 & \underline{2} & -1 \\ 1 & -2 & 1 \end{bmatrix} \tag{27}$$

in the second case. With $\mathbf{P}^{11}$ given by (12) and $\mathbf{R}^{11}$ as before we obtain

$$\begin{aligned} [\mathbf{A}^{11(m)}] \ = \ & \frac{2^m}{12}\begin{bmatrix} -1 & 0 & 1 \\ -4 & \underline{0} & 4 \\ -1 & 0 & 1 \end{bmatrix} + \frac{1}{12}\begin{bmatrix} -1 & 2 & -1 \\ -4 & \underline{8} & -4 \\ -1 & 2 & -1 \end{bmatrix} \\ + \ & \frac{2^{-m}}{12}\begin{bmatrix} 1 & 0 & -1 \\ -2 & \underline{0} & 2 \\ 1 & 0 & -1 \end{bmatrix} + \frac{4^{-m}}{12}\begin{bmatrix} 1 & -2 & 1 \\ -2 & \underline{4} & -2 \\ 1 & -2 & 1 \end{bmatrix} \end{aligned} \tag{28}$$

in the first case of (24), whereas the second case results in a stencil that is obtained from (28) by a 90° rotation.

A finite difference operator $\mathbf{A}$ will be called a K-matrix if

$$\mathbf{A}(i,0) > 0, \quad \mathbf{A}(i,j) \leq 0 \ \text{ for } \ j \neq 0, \quad \textstyle\sum_j \mathbf{A}(i,j) \geq 0. \tag{29}$$

In [18] it is shown that a K-operator corresponds to an M-matrix. We see that all of the coarse grid operators (26)–(28) deviate more from the K-property as m increases. Accordingly, smoothing performance might deteriorate on coarser grids. This phenomenon is well-known for applications of GCA to convection-diffusion type equations ([3], [4]). Whether this is serious for our applications will be investigated by numerical experiments to be described later.

For completeness we also show how a typical viscous term, given by

$$[\mathbf{A}^{11}] = \begin{bmatrix} 0 & -1 & 0 \\ -1 & \underline{4} & -1 \\ 0 & -1 & 0 \end{bmatrix} \tag{30}$$

transforms under GCA. We find, with $\mathbf{P}^{11}$ given by the first equation of (14) and $\mathbf{R}^{11} = \mathbf{P}^{11*}$:

$$[\mathbf{A}^{11(m)}] = \frac{2^m}{6}\begin{bmatrix} -1 & -4 & -1 \\ 2 & \underline{8} & 2 \\ -1 & -4 & -1 \end{bmatrix} + \begin{bmatrix} -1 & \underline{2} & -1 \end{bmatrix} + \frac{2^{-m}}{6}\begin{bmatrix} 1 & -2 & 1 \\ -2 & \underline{4} & -2 \\ 1 & -2 & 1 \end{bmatrix}. \tag{31}$$

Note that the anisotropic nature of $\mathbf{P}^{11}$ and $\mathbf{R}^{11}$ causes the horizontal derivative to gradually disappear. With $\mathbf{P}^{11}$ given by (12) and $\mathbf{R}^{11}$ as before we find

$$[\mathbf{A}^{11(m)}] = \frac{1}{3}\begin{bmatrix} -1 & -1 & -1 \\ -1 & \underline{8} & -1 \\ -1 & -1 & -1 \end{bmatrix} + \frac{4^{-m}}{3}\begin{bmatrix} 1 & -2 & 1 \\ -2 & \underline{4} & -2 \\ 1 & -2 & 1 \end{bmatrix}. \tag{32}$$

Note that in (32) the K-property is conserved, but in (31) is lost.

Using central differences, we found that satisfactory smoothing could be obtained in the instationary case by choosing Δt small enough.

6 Numerical Experiments

In section 3 we have described two types of prolongation: bilinear and hybrid, denoted as type B and type H for short. Prolongation in GCA and prolongation of corrections may be chosen differently. The prolongation for GCA will be denoted by $\mathbf{P}_A$ and that for the correction by $\mathbf{P}_C$. We always use (15) for the pressure. The notation $\mathbf{P}_A(B)$ means that the bilinear prolongation was used for the coarse grid matrix, etc.. Our concern here is that $\mathbf{P}_A(H)$ violates (19); on the other hand, $\mathbf{P}_A(B)$ bring us farther from the K-matrix property on the coarse grids in some situations. We will describe some numerical experiments to find suitable combinations of $\mathbf{P}_A$ and $\mathbf{P}_C$.

Our first test problem is the two-dimensional square driven cavity flow problem, with a uniform computational grid. We use the stationary equations. The multigrid algorithm is as follows. The initial solution is zero. The W-cycle is used with one pre- and one post-smoothing. Since our main concern here is coarse grid approximation, it is not necessary to give details about the smoothing method, which was of distributive ILU type, not unlike the methods described in [20]. For details, see [21]. On the coarsest grid of size 2×2, exact solution takes place. The residual norm is measured by

$$r = \|\mathbf{b} - \mathbf{A}\mathbf{u}\| = \left(\frac{1}{M}\sum_{m=1}^{M}\left(\sum_{j\in\mathcal{G}^m}(\mathbf{b}^m - \mathbf{A}^m\mathbf{u})_j^2/N_g^m\right)\right)^{\frac{1}{2}} \tag{33}$$

with M the number of unknowns and N_g^m the number of grid points in $\mathcal{G}^m$; $\mathcal{G}^m$ is the part of the staggered grid where $\mathbf{u}^m$ is defined. Let r_0 and r_i be the norm of the initial residual and the norm after i multigrid iterations on the finest grid, respectively. The average reduction factor is $\bar{\rho} = (r_n/r_0)^{\frac{1}{n}}$. The reduction factor ρ_i is defined by $\rho_i = r_i/r_{i-1}$. First, two outer (Picard) iterations are carried out, each with one multigrid cycle as inner iteration. In the third outer iteration 20 multigrid cycles are performed. Table 1 presents results. The numbers accompanied by a '$*$' are not accurate because machine accuracy was reached before the 20-th cycle.

With $\mathbf{P}_C(H)$ and $\mathbf{P}_A(H)$, the algorithm works for both the low and the high Reynolds number case. The reduction factor grows with refinement of grids,

TABLE 1. Reduction factors for the square driven cavity problem

Finest grid	Re		1		10,000	
	$\mathbf{P}_C$ \ $\mathbf{P}_A$		H	B	H	B
32×32	H	$\bar{\rho}$	0.4093	0.2751	0.2788	0.2541
		ρ_{16}	0.4250	0.2760	0.3229	0.2699
		ρ_{17}	0.4253	0.2763	0.3239	0.2724
		ρ_{18}	0.4254	0.2763	0.3245	0.2750
		ρ_{19}	0.4255	0.2765	0.3247	0.2824*
		ρ_{20}	0.4256	0.2765	0.3241	0.3599*
	B	$\bar{\rho}$	0.3741	0.2461	0.2522	0.2543
		ρ_{16}	0.3961	0.2710	0.2731	0.2711
		ρ_{17}	0.3964	0.2713	0.2744	0.2735
		ρ_{18}	0.3966	0.2715	0.2764	0.2758
		ρ_{19}	0.3963	0.2714	0.2773	0.2837*
		ρ_{20}	0.3970	0.2714	0.2943*	0.3757*
64×64	H	$\bar{\rho}$	0.5064	0.3060	0.3236	0.2562
		ρ_{16}	0.5292	0.3051	0.3988	0.2635
		ρ_{17}	0.5299	0.3058	0.4031	0.2652
		ρ_{18}	0.5304	0.3058	0.4045	0.2698
		ρ_{19}	0.5308	0.3063	0.4041	0.2974
		ρ_{20}	0.5312	0.3064	0.4031	0.4996
	B	$\bar{\rho}$	0.4630	0.2411	0.2972	0.2645
		ρ_{16}	0.4971	0.2695	0.3534	0.2648
		ρ_{17}	0.4981	0.2706	0.3530	0.2677
		ρ_{18}	0.4990	0.2712	0.3520	0.2815*
		ρ_{19}	0.4998	0.2717	0.3507	0.4062*
		ρ_{20}	0.5005	0.2732	0.3493	0.7557*
128×128	H	$\bar{\rho}$	0.5798	0.3304	0.3637	
		ρ_{16}	0.6008	0.3293	0.4155	
		ρ_{17}	0.6019	0.3211	0.4211	div
		ρ_{18}	0.6028	0.3317	0.4245	
		ρ_{19}	0.6036	0.3329	0.4261	
		ρ_{20}	0.6043	0.3335	0.4263	
	B	$\bar{\rho}$	0.5313	0.2385	0.3329	
		ρ_{16}	0.5679	0.2522	0.3811	
		ρ_{17}	0.5695	0.2571	0.3855	div
		ρ_{18}	0.5709	0.2612	0.3866	
		ρ_{19}	0.5722	0.2646	0.3856	
		ρ_{20}	0.5733	0.2694	0.3835	

but the rate of convergence remains satisfactory. Apparently, the violation of (19) does not spoil convergence enough to make the method useless. With $\mathbf{P}_C(B)$ and $\mathbf{P}_A(B)$, the algorithm has very good convergence, except on the 128×128 grid for $Re = 10,000$, where it diverges. The failure is observed to be caused by bad smoothing on coarse grids. This is explained by (28); a large departure from the K-matrix property occurs. Other combinations of $\mathbf{P}_C$ and $\mathbf{P}_A$ give results lying between those from the above two extreme choices. Generally, for a fixed $\mathbf{P}_A$, $\mathbf{P}_C(B)$ gives better results; for a fixed $\mathbf{P}_C$, $\mathbf{P}_A(B)$ gives better results, if the algorithm works. $\mathbf{P}_A(H)$ is more suitable for high Reynolds number cases, while $\mathbf{P}_A(B)$ is more suitable for low Reynolds number cases.

Now we go to general coordinates and compute the flow in a two-dimensional L-shaped driven cavity. A benchmark solution is available in [12] for $Re = 100$ and $Re = 1000$. We now use central differencing for the convection terms and time-stepping. Due to time-stepping, we have a sequence of systems of equations at the corresponding time levels, which are to be solved by the linear multigrid method. With different combinations of $\mathbf{P}_C$ and $\mathbf{P}_A$, our purpose now is to see whether the multigrid method is also applicable in general coordinates. So we solve the problem until steady state. Because of central differencing, diagonal dominance is lost when the Reynolds number is sufficiently large, which could damage smoothing. Time-stepping increases the main diagonal, and so one may expect that a proper choice of the time step Δt could still maintain smoothing even if the Reynolds number is large. Based on some numerical experiments, $\Delta t = 0.2$ is used for $Re = 1000$. For $Re = 100$, Δt can be somewhat larger; here we choose $\Delta t = 1$. At each time step, one multigrid iteration is carried out. The final time t is fixed at 20. Denote the residual norm defined by (33) for the *stationary* Navier-Stokes equations at $t = 0$ as r^0, and that at $t = 20$ as r^{20}. In table 2, we give the reduction factor $\bar{\rho} = r_1/r_0$ of the one multigrid iteration at $t = 20$ on the 32×32, 64×64 and 128×128 grids and the ratio r^{20}/r^0 (measuring closeness to steady state), for all combinations of $\mathbf{P}_C$ and $\mathbf{P}_A$; the numbers indicated by a '$*$' are obtained with a heavier underrelaxation in the smoothing method (not discussed here). The reduction factor gradually approaches a stable value after several time levels.

The conclusions are as follows. When the Reynolds number is large, for a fixed $\mathbf{P}_A$, $\mathbf{P}_C(B)$ gives better results. But for a fixed $\mathbf{P}_C$, $\mathbf{P}_A(B)$ sometimes leads to slightly worse results. Whereas we have the same conclusion as for the square driven cavity problem when the Reynolds number is low. Again, $\mathbf{P}_A(H)$ is better for high Reynolds numbers and $\mathbf{P}_A(B)$ is better for low Reynolds numbers.

A further computation is carried out for a three-dimensional flow in a channel with a backward facing step, as shown in figure 2 The grid is uniform in direction 3, and is generated in plane 1-2 by using a biharmonic grid generator. Only $Re = 100$ is considered here. The combination of $\mathbf{P}_A(B)$ and $\mathbf{P}_C(B)$ is employed. Transfer operators are replaced by their corresponding three-dimensional versions, which are obtained from the two-dimensional versions in a straightforward way. The final time is again 20, and Δt is chosen to be 0.25 based on numerical experiments. Because this is found to be more efficient, the W-cycle used in the previous compu-

TABLE 2. Reduction factors $\bar{\rho}$ and r^{20}/r^0 for the L-shaped driven cavity problem on three grids with 2×2 coarsest grid

Grid	$\mathbf{P}_C$ \ $\mathbf{P}_A$	H	B
32×32		$Re = 100,\ \Delta t = 1$	
	H	$0.4120,\ 8.650 \times 10^{-8}$	$0.3401,\ 9.143 \times 10^{-9}$
	B	$0.3189,\ 5.542 \times 10^{-8}$	$0.3164,\ 6.758 \times 10^{-9}$
		$Re = 1000,\ \Delta t = 0.2$	
	H	$0.1437,\ 1.380 \times 10^{-3}$	$0.1366,\ 1.348 \times 10^{-3}$
	B	$0.1323,\ 1.361 \times 10^{-3}$	$0.1294,\ 1.352 \times 10^{-3}$
64×64		$Re = 100,\ \Delta t = 1$	
	H	$0.5043,\ 4.817 \times 10^{-7}$	$0.4035,\ 7.023 \times 10^{-9}$
	B	$0.4793,\ 2.978 \times 10^{-7}$	$0.3781,\ 2.974 \times 10^{-9}$
		$Re = 1000,\ \Delta t = 0.2$	
	H	$0.1979,\ 3.757 \times 10^{-4}$	$0.1802,\ 3.699 \times 10^{-4}$
	B	$0.1696,\ 3.682 \times 10^{-4}$	$0.1721,\ 3.698 \times 10^{-4}$
128×128		$Re = 100,\ \Delta t = 1$	
	H	$0.7992,\ 6.252 \times 10^{-4*}$	$0.4564,\ 6.437 \times 10^{-8}$
	B	$0.7252,\ 3.870 \times 10^{-5*}$	$0.4009,\ 1.904 \times 10^{-9}$
		$Re = 1000,\ \Delta t = 0.2$	
	H	$0.2071,\ 1.167 \times 10^{-4}$	$0.2270,\ 1.189 \times 10^{-4}$
	B	$0.1432,\ 1.153 \times 10^{-4}$	$0.1523,\ 1.172 \times 10^{-4}$

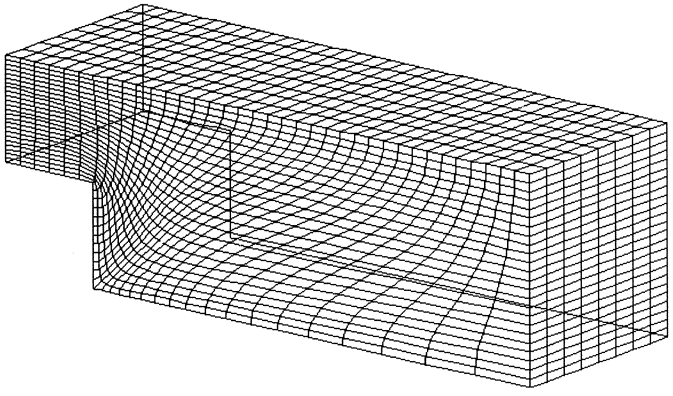


FIGURE 2. The channel with a backward facing step and a $24 \times 36 \times 8$ grid

TABLE 3. Reduction factors $\bar{\rho}$ and r^{20}/r^0 for the channel with backward facing step on two grids, $Re = 100$, $\Delta t = 0.25$

Grid	$\bar{\rho}$, r^{20}/r^0
$12 \times 18 \times 4$	$0.0489, 0.7457 \times 10^{-4}$
$24 \times 36 \times 8$	$0.1118, 0.9494 \times 10^{-5}$

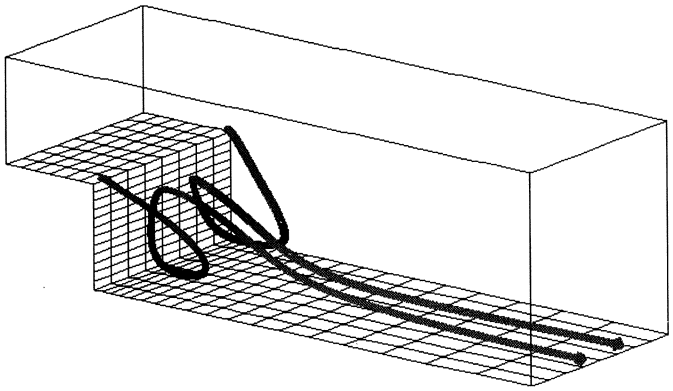


FIGURE 3. Two particle traces in the channel flow on the $24 \times 36 \times 8$ grid

tations is replaced by the F-cycle with one pre- and one post-smoothing. At each time step, one multigrid iteration is carried out. A smoothing sweep consists of one plane sweep marching in direction 3, and in a plane (parallel to plane 1-2) the ILU smoother used before is applied. The coarsest grid solver uses the smoother, performing 10 sweeps. Table 3 gives the reduction factor of the one multigrid iteration at the final time and the ratio r^{20}/r^0, on the finest grid. The reduction factor gradually approaches a constant after some time steps. Figure 3 presents two particle traces on the $24 \times 36 \times 8$ grid. Clearly, the flow pattern is very different from that obtained in two-dimensional cases (see, for instance, [9]), where the flow in the backward facing step forms a closed recirculation region.

7 Conclusions

Galerkin coarse grid approximation (GCA) for multigrid methods has been investigated for systems of equations. It is found that diagonal dominance of coarse grid matrices is lost or the derivative in a direction disappears for upwind differencing. For central differencing, the derivative in a direction vanishes with the

hybrid prolongation. Furthermore, with the hybrid prolongation, the well-known accuracy condition (19) is violated. However, all this seems not to have caused serious problems. Numerical experiments show that for a fixed $\mathbf{P}_A$, bilinear prolongation for $\mathbf{P}_C$ is more efficient, unless the method diverges, which may occur. Bilinear prolongation for $\mathbf{P}_A$ is more preferable for low Reynolds numbers, and hybrid prolongation for $\mathbf{P}_A$ is more preferable for high Reynolds numbers. With bilinear prolongation for $\mathbf{P}_A$ and $\mathbf{P}_C$, we apply the method to a three-dimensional flow in a channel with a backward facing step. Satisfactory results are obtained.

ACKNOWLEDGEMENTS

The authors are grateful to their colleagues E. Brakkee, C.G.M. Kassels, C.W. Oosterlee and A. Segal, and their student C. Kasbergen, for their assistance in various aspects of this work.

References

[1] R. E. ALCOUFFE, A. BRANDT, JR. J. E. DENDY, AND J. W. PAINTER, *The multigrid method for diffusion equations with strongly discontinuous coefficients*, SIAM J. Sci. Stat. Comput., 2 (1981), pp. 430–454.

[2] A. BRANDT, *Multi-level adaptive techniques (MLAT) for partial differential equations: ideas and software*, in Proc. Symposium on Mathematical Software, J. Rice, ed., Academic Press, New York, 1977, pp. 277–318.

[3] P. M. DE ZEEUW, *Matrix-dependent prolongations and restrictions in a block multigrid method solver*, J. Comput. Appl. Math., 3 (1990), pp. 1–27.

[4] P. M. DE ZEEUW AND E. J. VAN ASSELT, *The convergence rate of multilevel algorihms applied to the convection-diffusion equation*, SIAM J. Sci. Stat. Comput., 6 (1985), pp. 492–508.

[5] J. J. E. DENDY, *Black box multigrid*, J. Comp. Phys., 48 (1982), pp. 366–386.

[6] W. HACKBUSCH, *Multi-grid methods and applications*, Springer, Berlin, 1985.

[7] P. W. HEMKER, *On the order of prolongations and restrictions in multigrid procedures*, J. Comp. Appl. Math., 32 (1990), pp. 423–429.

[8] R. KETTLER, *Analysis and comparison of relaxation schemes in robust multigrid and conjugate gradient methods*, in Multigrid Methods. Lecture Notes in Mathematics, W. Hackbusch and U. Trottenberg, eds., vol. 960, Springer, Berlin, 1982, pp. 502–534.

[9] K. J. MORGAN, J. PÉRIAUX, AND F. THOMASSET, eds., *Analysis of laminar flow over a backward facing step*, Vieweg, Braunschweig, 1984. A GAMM-Workshop, Bievres, France, 1984.

[10] A. E. MYNETT, P. WESSELING, A. SEGAL, AND C. G. M. KASSELS, *The ISNaS incompressible Navier-Stokes solver: invariant discretization*, Applied Scientific Research, 48 (1991), pp. 175–191.

[11] C. W. OOSTERLEE AND P. WESSELING, *A robust multigrid method for a discretization of the incompressible Navier-Stokes equations in general coordinates*, in Computational Fluid Dynamics'92. Proc. First European Computational Fluid Dynamics Conference, Ch. Hirsch, J. Périaux, and W. Kordulla, eds., Elsevier, Amsterdam, 1992, pp. 101–107.

[12] C. W. OOSTERLEE, P. WESSELING, A. SEGAL, AND E. BRAKKEE, *Benchmark solutions of the incompressible Navier-Stokes equations in general coordinates on staggered grids*, To appear in Int. J. Numer. Methods in Fluids.

[13] A. SEGAL, P. WESSELING, J. VAN KAN, C. W. OOSTERLEE, AND C. G. M. KASSELS, *Invariant discretization of the incompressible Navier-Stokes equations in boundary fitted co-ordinates*, Int. J. Numer. Methods in Fluids, 15 (1992), pp. 411–426.

[14] D. B. SPALDING, *A novel finite difference formulation for differential expressions involving both first and second derivatives*, Int. J. Numer. Methods in Fluids, 4 (1972), pp. 551–559.

[15] P. WESSELING, *Theoretical and practical aspects of a multigrid method*, SIAM J. Sci. Stat. Comput., 3 (1982), pp. 387–407.

[16] ——, *Linear multigrid methods*, in Multigrid Methods, S. F. McCormick, ed., SIAM, Philadelphia, 1987, pp. 31–56. In R. E. Ewing, ed., Frontiers in Applied Mathematics, vol. 3.

[17] ——, *Cell-centered multigrid for interface problems*, J. Comput. Phys., 79 (1988), pp. 85–91.

[18] ——, *An introduction to multigrid methods*, John Wiley & Sons, Chichester, 1992.

[19] P. WESSELING, A. SEGAL, J. VAN KAN, C. W. OOSTERLEE, AND C. G. M. KASSELS, *Finite volume discretization of the incompressible Navier-Stokes equations in general coordinates on staggered grids*, Comp. Fluid Dyn. J., 1 (1992), pp. 27–33.

[20] G. WITTUM, *On the convergence of multi-grid methods with transforming smoothers—theory with applications to the Navier-Stokes equations*, Numer. Math., 57 (1990), pp. 15–38.

[21] S. ZENG AND P. WESSELING, *An ILU smoother for the incompressible Navier-Stokes equations in general coordinates*, Report 92-91, Faculty of Technical Mathematics and Informatics, TU Delft, The Netherlands, 1992.

Contents of "Multigrid Methods IV"

The book "Multigrid Methods IV", contains the Proceedings of the Fourth European Multigrid Conference, held in Amsterdam, June 5-9, 1993. It is published Birkhäuser Verlag, Basel, as Volume 116 in the International Series of Numerical Mathematics (368 pages), ISBN 3-7643-5030-X.

CWI TRACTS

1 D.H.J. Epema. *Surfaces with canonical hyperplane sections.* 1984.

2 J.J. Dijkstra. *Fake topological Hilbert spaces and characterizations of dimension in terms of negligibility.* 1984.

3 A.J. van der Schaft. *System theoretic descriptions of physical systems.* 1984.

4 J. Koene. *Minimal cost flow in processing networks, a primal approach.* 1984.

5 B. Hoogenboom. *Intertwining functions on compact Lie groups.* 1984.

6 A.P.W. Böhm. *Dataflow computation.* 1984.

7 A. Blokhuis. *Few-distance sets.* 1984.

8 M.H. van Hoorn. *Algorithms and approximations for queueing systems.* 1984.

9 C.P.J. Koymans. *Models of the lambda calculus.* 1984.

10 C.G. van der Laan, N.M. Temme. *Calculation of special functions: the gamma function, the exponential integrals and error-like functions.* 1984.

11 N.M. van Dijk. *Controlled Markov processes; time-discretization.* 1984.

12 W.H. Hundsdorfer. *The numerical solution of nonlinear stiff initial value problems: an analysis of one step methods.* 1985.

13 D. Grune. *On the design of ALEPH.* 1985.

14 J.G.F. Thiemann. *Analytic spaces and dynamic programming: a measure theoretic approach.* 1985.

15 F.J. van der Linden. *Euclidean rings with two infinite primes.* 1985.

16 R.J.P. Groothuizen. *Mixed elliptic-hyperbolic partial differential operators: a case-study in Fourier integral operators.* 1985.

17 H.M.M. ten Eikelder. *Symmetries for dynamical and Hamiltonian systems.* 1985.

18 A.D.M. Kester. *Some large deviation results in statistics.* 1985.

19 T.M.V. Janssen. *Foundations and applications of Montague grammar, part 1: Philosophy, framework, computer science.* 1986.

20 B.F. Schriever. *Order dependence.* 1986.

21 D.P. van der Vecht. *Inequalities for stopped Brownian motion.* 1986.

22 J.C.S.P. van der Woude. *Topological dynamix.* 1986.

23 A.F. Monna. *Methods, concepts and ideas in mathematics: aspects of an evolution.* 1986.

24 J.C.M. Baeten. *Filters and ultrafilters over definable subsets of admissible ordinals.* 1986.

25 A.W.J. Kolen. *Tree network and planar rectilinear location theory.* 1986.

26 A.H. Veen. *The misconstrued semicolon: Reconciling imperative languages and dataflow machines.* 1986.

27 A.J.M. van Engelen. *Homogeneous zero-dimensional absolute Borel sets.* 1986.

28 T.M.V. Janssen. *Foundations and applications of Montague grammar, part 2: Applications to natural language.* 1986.

29 H.L. Trentelman. *Almost invariant subspaces and high gain feedback.* 1986.

30 A.G. de Kok. *Production-inventory control models: approximations and algorithms.* 1987.

31 E.E.M. van Berkum. *Optimal paired comparison designs for factorial experiments.* 1987.

32 J.H.J. Einmahl. *Multivariate empirical processes.* 1987.

33 O.J. Vrieze. *Stochastic games with finite state and action spaces.* 1987.

34 P.H.M. Kersten. *Infinitesimal symmetries: a computational approach.* 1987.

35 M.L. Eaton. *Lectures on topics in probability inequalities.* 1987.

36 A.H.P. van der Burgh, R.M.M. Mattheij (eds.). *Proceedings of the first international conference on industrial and applied mathematics (ICIAM 87).* 1987.

37 L. Stougie. *Design and analysis of algorithms for stochastic integer programming.* 1987.

38 J.B.G. Frenk. *On Banach algebras, renewal measures and regenerative processes.* 1987.

39 H.J.M. Peters, O.J. Vrieze (eds.). *Surveys in game theory and related topics.* 1987.

40 J.L. Geluk, L. de Haan. *Regular variation, extensions and Tauberian theorems.* 1987.

41 Sape J. Mullender (ed.). *The Amoeba distributed operating system: Selected papers 1984-1987.* 1987.

42 P.R.J. Asveld, A. Nijholt (eds.). *Essays on concepts, formalisms, and tools.* 1987.

43 H.L. Bodlaender. *Distributed computing: structure and complexity.* 1987.

44 A.W. van der Vaart. *Statistical estimation in large parameter spaces.* 1988.

45 S.A. van de Geer. *Regression analysis and empirical processes.* 1988.

46 S.P. Spekreijse. *Multigrid solution of the steady Euler equations.* 1988.

47 J.B. Dijkstra. *Analysis of means in some non-standard situations.* 1988.

48 F.C. Drost. *Asymptotics for generalized chi-square goodness-of-fit tests.* 1988.

49 F.W. Wubs. *Numerical solution of the shallow-water equations.* 1988.

50 F. de Kerf. *Asymptotic analysis of a class of perturbed Korteweg-de Vries initial value problems.* 1988.

51 P.J.M. van Laarhoven. *Theoretical and computational aspects of simulated annealing.* 1988.

52 P.M. van Loon. *Continuous decoupling transformations for linear boundary value problems.* 1988.

53 K.C.P. Machielsen. *Numerical solution of optimal control problems with state constraints by sequential quadratic programming in function space.* 1988.

54 L.C.R.J. Willenborg. *Computational aspects of survey data processing.* 1988.

55 G.J. van der Steen. *A program generator for recognition, parsing and transduction with syntactic patterns.* 1988.

56 J.C. Ebergen. *Translating programs into delay-insensitive circuits.* 1989.

57 S.M. Verduyn Lunel. *Exponential type calculus for linear delay equations.* 1989.

58 M.C.M. de Gunst. *A random model for plant cell population growth.* 1989.

59 D. van Dulst. *Characterizations of Banach spaces not containing l^1.* 1989.

60 H.E. de Swart. *Vacillation and predictability properties of low-order atmospheric spectral models.* 1989.

61 P. de Jong. *Central limit theorems for generalized multilinear forms.* 1989.

62 V.J. de Jong. *A specification system for statistical software.* 1989.

63 B. Hanzon. *Identifiability, recursive identification and spaces of linear dynamical systems, part I.* 1989.

64 B. Hanzon. *Identifiability, recursive identification and spaces of linear dynamical systems, part II.* 1989.

65 B.M.M. de Weger. *Algorithms for diophantine equations.* 1989.

66 A. Jung. *Cartesian closed categories of domains.* 1989.

67 J.W. Polderman. *Adaptive control & identification: Conflict or conflux?.* 1989.

68 H.J. Woerdeman. *Matrix and operator extensions.* 1989.

69 B.G. Hansen. *Monotonicity properties of infinitely divisible distributions.* 1989.

70 J.K. Lenstra, H.C. Tijms, A. Volgenant (eds.). *Twenty-five years of operations research in the Netherlands: Papers dedicated to Gijs de Leve.* 1990.

71 P.J.C. Spreij. *Counting process systems. Identification and stochastic realization.* 1990.

72 J.F. Kaashoek. *Modeling one dimensional pattern formation by anti-diffusion.* 1990.

73 A.M.H. Gerards. *Graphs and polyhedra. Binary spaces and cutting planes.* 1990.

74 B. Koren. *Multigrid and defect correction for the steady Navier-Stokes equations. Application to aerodynamics.* 1991.

75 M.W.P. Savelsbergh. *Computer aided routing.* 1992.

76 O.E. Flippo. *Stability, duality and decomposition in general mathematical programming.* 1991.

77 A.J. van Es. *Aspects of nonparametric density estimation.* 1991.

78 G.A.P. Kindervater. *Exercises in parallel combinatorial computing.* 1992.

79 J.J. Lodder. *Towards a symmetrical theory of generalized functions.* 1991.

80 S.A. Smulders. *Control of freeway traffic flow.* 1993.

81 P.H.M. America, J.J.M.M. Rutten. *A parallel object-oriented language: design and semantic foundations.* 1992.

82 F. Thuijsman. *Optimality and equilibria in stochastic games.* 1992.

83 R.J. Kooman. *Convergence properties of recurrence sequences.* 1992.

84 A.M. Cohen (ed.). *Computational aspects of Lie group representations and related topics. Proceedings of the 1990 Computational Algebra Seminar at CWI, Amsterdam.* 1991.

85 V. de Valk. *One-dependent processes.* 1994.

86 J.A. Baars, J.A.M. de Groot. *On topological and linear equivalence of certain function spaces.* 1992.

87 A.F. Monna. *The way of mathematics and mathematicians.* 1992.

88 E.D. de Goede. *Numerical methods for the three-dimensional shallow water equations.* 1993.

89 M. Zwaan. *Moment problems in Hilbert space with applications to magnetic resonance imaging.* 1993.

90 C. Vuik. *The solution of a one-dimensional Stefan problem.* 1993.

91 E.R. Verheul. *Multimedians in metric and normed spaces.* 1993.

92 J.L.M. Maubach. *Iterative methods for non-linear partial differential equations.* 1993.

93 A.W. Ambergen. *Statistical uncertainties in posterior probabilities.* 1993.

94 P.A. Zegeling. *Moving-grid methods for time-dependent partial differential equations.* 1993.

95 M.J.C. van Pul. *Statistical analysis of software reliability models.* 1993.

96 J.K. Scholma. *A Lie algebraic study of some integrable systems associated with root systems.* 1993.

97 J.L. van den Berg. *Sojourn times in feedback and processor sharing queues.* 1993.

98 A.J. Koning. *Stochastic integrals and goodness-of-fit tests.* 1993.

99 B.P. Sommeijer. *Parallelism in the numerical integration of initial value problems.* 1993.

100 J. Molenaar. *Multigrid methods for semiconductor device simulation.* 1993.

101 H.J.C. Huijberts. *Dynamic feedback in nonlinear synthesis problems.* 1994.

102 J.A.M. van der Weide. *Stochastic processes and point processes of excursions.* 1994.

103 P.W. Hemker, P. Wesseling (eds.). *Contributions to multigrid.* 1994.

MATHEMATICAL CENTRE TRACTS

1 T. van der Walt. *Fixed and almost fixed points.* 1963.

2 A.R. Bloemena. *Sampling from a graph.* 1964.

3 G. de Leve. *Generalized Markovian decision processes, part I: model and method.* 1964.

4 G. de Leve. *Generalized Markovian decision processes, part II: probabilistic background.* 1964.

5 G. de Leve, H.C. Tijms, P.J. Weeda. *Generalized Markovian decision processes, applications.* 1970.

6 M.A. Maurice. *Compact ordered spaces.* 1964.

7 W.R. van Zwet. *Convex transformations of random variables.* 1964.

8 J.A. Zonneveld. *Automatic numerical integration.* 1964.

9 P.C. Baayen. *Universal morphisms.* 1964.

10 E.M. de Jager. *Applications of distributions in mathematical physics.* 1964.

11 A.B. Paalman-de Miranda. *Topological semigroups.* 1964.

12 J.A.Th.M. van Berckel, H. Brandt Corstius, R.J. Mokken, A. van Wijngaarden. *Formal properties of newspaper Dutch.* 1965.

13 H.A. Lauwerier. *Asymptotic expansions.* 1966, out of print; replaced by MCT 54.

14 H.A. Lauwerier. *Calculus of variations in mathematical physics.* 1966.

15 R. Doornbos. *Slippage tests.* 1966.

16 J.W. de Bakker. *Formal definition of programming languages with an application to the definition of ALGOL 60.* 1967.

17 R.P. van de Riet. *Formula manipulation in ALGOL 60, part 1.* 1968.

18 R.P. van de Riet. *Formula manipulation in ALGOL 60, part 2.* 1968.

19 J. van der Slot. *Some properties related to compactness.* 1968.

20 P.J. van der Houwen. *Finite difference methods for solving partial differential equations.* 1968.

21 E. Wattel. *The compactness operator in set theory and topology.* 1968.

22 T.J. Dekker. *ALGOL 60 procedures in numerical algebra, part 1.* 1968.

23 T.J. Dekker, W. Hoffmann. *ALGOL 60 procedures in numerical algebra, part 2.* 1968.

24 J.W. de Bakker. *Recursive procedures.* 1971.

25 E.R. Paërl. *Representations of the Lorentz group and projective geometry.* 1969.

26 European Meeting 1968. *Selected statistical papers, part I.* 1968.

27 European Meeting 1968. *Selected statistical papers, part II.* 1968.

28 J. Oosterhoff. *Combination of one-sided statistical tests.* 1969.

29 J. Verhoeff. *Error detecting decimal codes.* 1969.

30 H. Brandt Corstius. *Exercises in computational linguistics.* 1970.

31 W. Molenaar. *Approximations to the Poisson, binomial and hypergeometric distribution functions.* 1970.

32 L. de Haan. *On regular variation and its application to the weak convergence of sample extremes.* 1970.

33 F.W. Steutel. *Preservation of infinite divisibility under mixing and related topics.* 1970.

34 I. Juhász, A. Verbeek, N.S. Kroonenberg. *Cardinal functions in topology.* 1971.

35 M.H. van Emden. *An analysis of complexity.* 1971.

36 J. Grasman. *On the birth of boundary layers.* 1971.

37 J.W. de Bakker, G.A. Blaauw, A.J.W. Duijvestijn, E.W. Dijkstra, P.J. van der Houwen, G.A.M. Kamsteeg-Kemper, F.E.J. Kruseman Aretz, W.L. van der Poel, J.P. Schaap-Kruseman, M.V. Wilkes, G. Zoutendijk. *MC-25 Informatica Symposium.* 1971.

38 W.A. Verloren van Themaat. *Automatic analysis of Dutch compound words.* 1972.

39 H. Bavinck. *Jacobi series and approximation.* 1972.

40 H.C. Tijms. *Analysis of (s,S) inventory models.* 1972.

41 A. Verbeek. *Superextensions of topological spaces.* 1972.

42 W. Vervaat. *Success epochs in Bernoulli trials (with applications in number theory).* 1972.

43 F.H. Ruymgaart. *Asymptotic theory of rank tests for independence.* 1973.

44 H. Bart. *Meromorphic operator valued functions.* 1973.

45 A.A. Balkema. *Monotone transformations and limit laws.* 1973.

46 R.P. van de Riet. *ABC ALGOL, a portable language for formula manipulation systems, part 1: the language.* 1973.

47 R.P. van de Riet. *ABC ALGOL, a portable language for formula manipulation systems, part 2: the compiler.* 1973.

48 F.E.J. Kruseman Aretz, P.J.W. ten Hagen, H.L. Oudshoorn. *An ALGOL 60 compiler in ALGOL 60, text of the MC-compiler for the EL-X8.* 1973.

49 H. Kok. *Connected orderable spaces.* 1974.

50 A. van Wijngaarden, B.J. Mailloux, J.E.L. Peck, C.H.A. Koster, M. Sintzoff, C.H. Lindsey, L.G.L.T. Meertens, R.G. Fisker (eds.). *Revised report on the algorithmic language ALGOL 68.* 1976.

51 A. Hordijk. *Dynamic programming and Markov potential theory.* 1974.

52 P.C. Baayen (ed.). *Topological structures.* 1974.

53 M.J. Faber. *Metrizability in generalized ordered spaces.* 1974.

54 H.A. Lauwerier. *Asymptotic analysis, part 1.* 1974.

55 M. Hall, Jr., J.H. van Lint (eds.). *Combinatorics, part 1: theory of designs, finite geometry and coding theory.* 1974.

56 M. Hall, Jr., J.H. van Lint (eds.). *Combinatorics, part 2: graph theory, foundations, partitions and combinatorial geometry.* 1974.

57 M. Hall, Jr., J.H. van Lint (eds.). *Combinatorics, part 3: combinatorial group theory.* 1974.

58 W. Albers. *Asymptotic expansions and the deficiency concept in statistics.* 1975.

59 J.L. Mijnheer. *Sample path properties of stable processes.* 1975.

60 F. Göbel. *Queueing models involving buffers.* 1975.

63 J.W. de Bakker (ed.). *Foundations of computer science.* 1975.

64 W.J. de Schipper. *Symmetric closed categories.* 1975.

65 J. de Vries. *Topological transformation groups, 1: a categorical approach.* 1975.

66 H.G.J. Pijls. *Logically convex algebras in spectral theory and eigenfunction expansions.* 1976.

68 P.P.N. de Groen. *Singularly perturbed differential operators of second order.* 1976.

69 J.K. Lenstra. *Sequencing by enumerative methods.* 1977.

70 W.P. de Roever, Jr. *Recursive program schemes: semantics and proof theory.* 1976.

71 J.A.E.E. van Nunen. *Contracting Markov decision processes.* 1976.

72 J.K.M. Jansen. *Simple periodic and non-periodic Lamé functions and their applications in the theory of conical waveguides.* 1977.

73 D.M.R. Leivant. *Absoluteness of intuitionistic logic.* 1979.

74 H.J.J. te Riele. *A theoretical and computational study of generalized aliquot sequences.* 1976.

75 A.E. Brouwer. *Treelike spaces and related connected topological spaces.* 1977.

76 M. Rem. *Associons and the closure statement.* 1976.

77 W.C.M. Kallenberg. *Asymptotic optimality of likelihood ratio tests in exponential families.* 1978.

78 E. de Jonge, A.C.M. van Rooij. *Introduction to Riesz spaces.* 1977.

79 M.C.A. van Zuijlen. *Emperical distributions and rank statistics.* 1977.

80 P.W. Hemker. *A numerical study of stiff two-point boundary problems.* 1977.

81 K.R. Apt, J.W. de Bakker (eds.). *Foundations of computer science II, part 1.* 1976.

82 K.R. Apt, J.W. de Bakker (eds.). *Foundations of computer science II, part 2.* 1976.

83 L.S. van Benthem Jutting. *Checking Landau's "Grundlagen" in the AUTOMATH system.* 1979.

84 H.L.L. Busard. *The translation of the elements of Euclid from the Arabic into Latin by Hermann of Carinthia (?), books vii-xii.* 1977.

85 J. van Mill. *Supercompactness and Wallman spaces.* 1977.

86 S.G. van der Meulen, M. Veldhorst. *Torrix I, a programming system for operations on vectors and matrices over arbitrary fields and of variable size.* 1978.

88 A. Schrijver. *Matroids and linking systems.* 1977.

89 J.W. de Roever. *Complex Fourier transformation and analytic functionals with unbounded carriers.* 1978.

90 L.P.J. Groenewegen. *Characterization of optimal strategies in dynamic games.* 1981.

91 J.M. Geysel. *Transcendence in fields of positive characteristic.* 1979.

92 P.J. Weeda. *Finite generalized Markov programming.* 1979.

93 H.C. Tijms, J. Wessels (eds.). *Markov decision theory.* 1977.

94 A. Bijlsma. *Simultaneous approximations in transcendental number theory.* 1978.

95 K.M. van Hee. *Bayesian control of Markov chains.* 1978.

96 P.M.B. Vitányi. *Lindenmayer systems: structure, languages, and growth functions.* 1980.

97 A. Federgruen. *Markovian control problems; functional equations and algorithms.* 1984.

98 R. Geel. *Singular perturbations of hyperbolic type.* 1978.

99 J.K. Lenstra, A.H.G. Rinnooy Kan, P. van Emde Boas (eds.). *Interfaces between computer science and operations research.* 1978.

100 P.C. Baayen, D. van Dulst, J. Oosterhoff (eds.). *Proceedings bicentennial congress of the Wiskundig Genootschap, part 1.* 1979.

101 P.C. Baayen, D. van Dulst, J. Oosterhoff (eds.). *Proceedings bicentennial congress of the Wiskundig Genootschap, part 2.* 1979.

102 D. van Dulst. *Reflexive and superreflexive Banach spaces.* 1978.

103 K. van Harn. *Classifying infinitely divisible distributions by functional equations.* 1978.

104 J.M. van Wouwe. *Go-spaces and generalizations of metrizability.* 1979.

105 R. Helmers. *Edgeworth expansions for linear combinations of order statistics.* 1982.

106 A. Schrijver (ed.). *Packing and covering in combinatorics.* 1979.

107 C. den Heijer. *The numerical solution of nonlinear operator equations by imbedding methods.* 1979.

108 J.W. de Bakker, J. van Leeuwen (eds.). *Foundations of computer science III, part 1.* 1979.

109 J.W. de Bakker, J. van Leeuwen (eds.). *Foundations of computer science III, part 2.* 1979.

110 J.C. van Vliet. *ALGOL 68 transput, part I: historical review and discussion of the implementation model.* 1979.

111 J.C. van Vliet. *ALGOL 68 transput, part II: an implementation model.* 1979.

112 H.C.P. Berbee. *Random walks with stationary increments and renewal theory.* 1979.

113 T.A.B. Snijders. *Asymptotic optimality theory for testing problems with restricted alternatives.* 1979.

114 A.J.E.M. Janssen. *Application of the Wigner distribution to harmonic analysis of generalized stochastic processes.* 1979.

115 P.C. Baayen, J. van Mill (eds.). *Topological structures II, part 1.* 1979.

116 P.C. Baayen, J. van Mill (eds.). *Topological structures II, part 2.* 1979.

117 P.J.M. Kallenberg. *Branching processes with continuous state space.* 1979.

118 P. Groeneboom. *Large deviations and asymptotic efficiencies.* 1980.

119 F.J. Peters. *Sparse matrices and substructures, with a novel implementation of finite element algorithms.* 1980.

120 W.P.M. de Ruyter. *On the asymptotic analysis of large-scale ocean circulation.* 1980.

121 W.H. Haemers. *Eigenvalue techniques in design and graph theory.* 1980.

122 J.C.P. Bus. *Numerical solution of systems of nonlinear equations.* 1980.

123 I. Yuhász. *Cardinal functions in topology - ten years later.* 1980.

124 R.D. Gill. *Censoring and stochastic integrals.* 1980.

125 R. Eising. *2-D systems, an algebraic approach.* 1980.

126 G. van der Hoek. *Reduction methods in nonlinear programming.* 1980.

127 J.W. Klop. *Combinatory reduction systems.* 1980.

128 A.J.J. Talman. *Variable dimension fixed point algorithms and triangulations.* 1980.

129 G. van der Laan. *Simplicial fixed point algorithms.* 1980.

130 P.J.W. ten Hagen, T. Hagen, P. Klint, H. Noot, H.J. Sint, A.H. Veen. *ILP: intermediate language for pictures.* 1980.

131 R.J.R. Back. *Correctness preserving program refinements: proof theory and applications.* 1980.

132 H.M. Mulder. *The interval function of a graph.* 1980.

133 C.A.J. Klaassen. *Statistical performance of location estimators.* 1981.

134 J.C. van Vliet, H. Wupper (eds.). *Proceedings international conference on ALGOL 68.* 1981.

135 J.A.G. Groenendijk, T.M.V. Janssen, M.J.B. Stokhof (eds.). *Formal methods in the study of language, part I.* 1981.

136 J.A.G. Groenendijk, T.M.V. Janssen, M.J.B. Stokhof (eds.). *Formal methods in the study of language, part II.* 1981.

137 J. Telgen. *Redundancy and linear programs.* 1981.

138 H.A. Lauwerier. *Mathematical models of epidemics.* 1981.

139 J. van der Wal. *Stochastic dynamic programming, successive approximations and nearly optimal strategies for Markov decision processes and Markov games.* 1981.

140 J.H. van Geldrop. *A mathematical theory of pure exchange economies without the no-critical-point hypothesis.* 1981.

141 G.E. Welters. *Abel-Jacobi isogenies for certain types of Fano threefolds.* 1981.

142 H.R. Bennett, D.J. Lutzer (eds.). *Topology and order structures, part 1.* 1981.

143 J.M. Schumacher. *Dynamic feedback in finite- and infinite-dimensional linear systems.* 1981.

144 P. Eijgenraam. *The solution of initial value problems using interval arithmetic; formulation and analysis of an algorithm.* 1981.

145 A.J. Brentjes. *Multi-dimensional continued fraction algorithms.* 1981.

146 C.V.M. van der Mee. *Semigroup and factorization methods in transport theory.* 1981.

147 H.H. Tigelaar. *Identification and informative sample size.* 1982.

148 L.C.M. Kallenberg. *Linear programming and finite Markovian control problems.* 1983.

149 C.B. Huijsmans, M.A. Kaashoek, W.A.J. Luxemburg, W.K. Vietsch (eds.). *From A to Z, proceedings of a symposium in honour of A.C. Zaanen.* 1982.

150 M. Veldhorst. *An analysis of sparse matrix storage schemes.* 1982.

151 R.J.M.M. Does. *Higher order asymptotics for simple linear rank statistics.* 1982.

152 G.F. van der Hoeven. *Projections of lawless sequences.* 1982.

153 J.P.C. Blanc. *Application of the theory of boundary value problems in the analysis of a queueing model with paired services.* 1982.

154 H.W. Lenstra, Jr., R. Tijdeman (eds.). *Computational methods in number theory, part I.* 1982.

155 H.W. Lenstra, Jr., R. Tijdeman (eds.). *Computational methods in number theory, part II.* 1982.

156 P.M.G. Apers. *Query processing and data allocation in distributed database systems.* 1983.

157 H.A.W.M. Kneppers. *The covariant classification of two-dimensional smooth commutative formal groups over an algebraically closed field of positive characteristic.* 1983.

158 J.W. de Bakker, J. van Leeuwen (eds.). *Foundations of computer science IV, distributed systems, part 1.* 1983.

159 J.W. de Bakker, J. van Leeuwen (eds.). *Foundations of computer science IV, distributed systems, part 2.* 1983.

160 A. Rezus. *Abstract AUTOMATH.* 1983.

161 G.F. Helminck. *Eisenstein series on the metaplectic group, an algebraic approach.* 1983.

162 J.J. Dik. *Tests for preference.* 1983.

163 H. Schippers. *Multiple grid methods for equations of the second kind with applications in fluid mechanics.* 1983.

164 F.A. van der Duyn Schouten. *Markov decision processes with continuous time parameter.* 1983.

165 P.C.T. van der Hoeven. *On point processes.* 1983.

166 H.B.M. Jonkers. *Abstraction, specification and implementation techniques, with an application to garbage collection.* 1983.

167 W.H.M. Zijm. *Nonnegative matrices in dynamic programming.* 1983.

168 J.H. Evertse. *Upper bounds for the numbers of solutions of diophantine equations.* 1983.

169 H.R. Bennett, D.J. Lutzer (eds.). *Topology and order structures, part 2.* 1983.