Introduction to multigrid methods

Alfio Borzì

Institut für Mathematik und Wissenschaftliches Rechnen
Karl–Franzens–Universität Graz
Heinrichstraße 36, 8010 Graz, Austria

Phone:+43 316 380 5166
Fax: +43 316 380 9815
www: http://www.uni-graz.at/imawww/borzi/index.html
email: alfio.borzi@uni-graz.at

Contents

1 Introduction

In these short Lecture Notes we describe the modern class of algorithms named multigrid methods. While these methods are well defined and classified in a mathematical sense, they are still source of new insight and of unexpected new application fields. In this respect, to name the methodology underneath these methods, we use the term multigrid strategy. Before we start considering multigrid methods, for the purpose of intuition, we give a few examples of 'multigrid phenomena'. A first example is percolation (Orig. Latin: to filter through). Percolation deals with effects related to the varying of the richness of interconnections in a infinite network system. For example, consider the electric network of Figure 1. If the fraction of links is higher than some transition value p^*, the net is conducting. Below this value, the net is insulating.

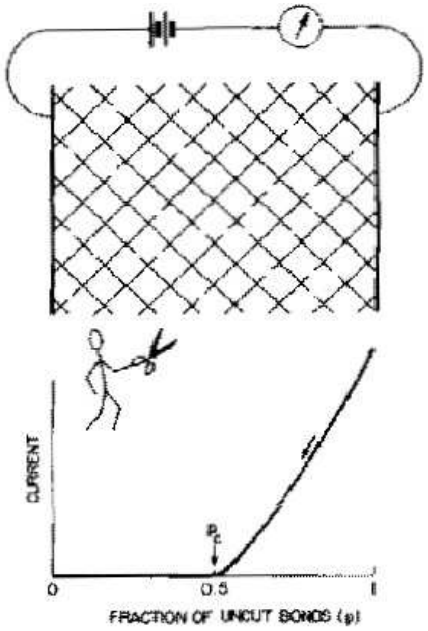

Figure 1: Electric network.

To have an insight on this transition phenomenon, consider the sim-

ple case of vertical percolation. In a 2×2 box there is vertical percolation in the cases depicted in Figure 2, which correspond to the occurrence of a continuous dark column. Assume that a single square in the box percolates (is dark) with probability p. Then the probability that the 2×2 cell percolates is

$$p' = R(p) = 2p^2(1-p^2) + 4p^3(1-p) + p^4$$

The mapping $p \rightarrow R(p)$ has a fixed point, $p^* = R(p^*)$, which is the

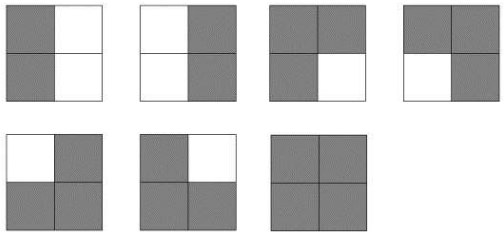

Figure 2: Vertical percolation in a 2×2 box.

critical value p^* to have percolation; in our case $p^* = 0.6180$. Now define a coarsening procedure, applied to a $n \times n$ (ideally infinite) box, which consists in replacing a 2×2 percolating box with a percolating square, otherwise a non percolating square. If the original box is characterized by $p < p^*$, the repeated application of the coarsening procedure results in a 'visible' non percolating box; see Figure 3. In the other case of $p > p^*$, a repeated application of the coarsening process reveals percolation as illustrated in Figure 4. At $p = p^*$ the distribution of connections is such that we have 'coarsening invariance'.

A second example that provides algebraic insight in the multigrid approach is cyclic reduction. Consider the finite-difference discretization of $-u'' = f$ on some interval and with homogeneous Dirichlet boundary conditions. On a uniform grid with 7 interior points the corresponding

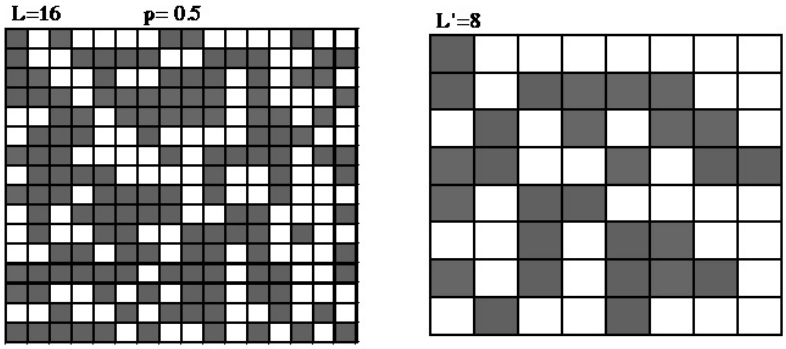

Figure 3: One coarsening step: No percolation ($p < p^*$).

algebraic problem is given by the following linear system

$$\begin{array}{c} E_1 \\ E_2 \\ E_3 \\ E_4 \\ E_5 \\ E_6 \\ E_7 \end{array} \begin{bmatrix} 2 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 2 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 2 \end{bmatrix} \cdot \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \end{bmatrix} = h^2 \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \end{bmatrix}$$

Let us call the equation corresponding to the row i as E_i. We define the following coarsening steps

$$E_i' = R(E_i) = E_{i-1} + 2\,E_i + E_{i+1}, \quad i = 2, 4, 6.$$

The result of one coarsening is

$$\begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix} \cdot \begin{bmatrix} u_2 \\ u_4 \\ u_6 \end{bmatrix} = H^2 \begin{bmatrix} \tilde{f}_2 \\ \tilde{f}_4 \\ \tilde{f}_6 \end{bmatrix} \quad (H = 2h),$$

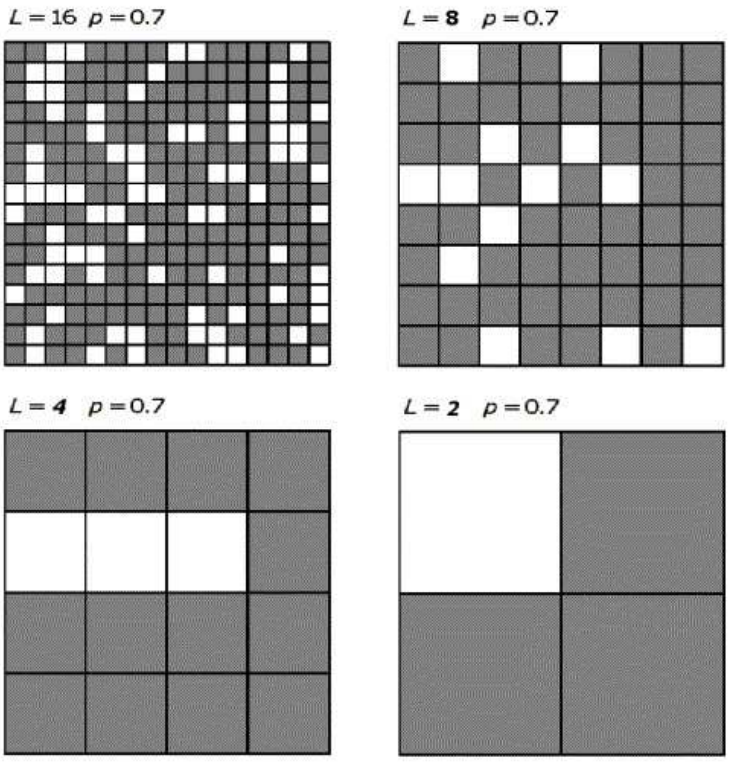

Figure 4: Four coarsening steps: Percolation ($p > p^*$).

thus obtaining a smaller algebraic system with the same structure. In this sense we have 'coarsening invariance'. We can start appreciate the advantage of a system possessing coarsening invariance. In this case in fact, we can re-solve properties of the system considering its coarsened version. In the first example, we can state the percolation property of the network by examining a coarsened version of the network. In the second example, we can compute part of the entire solution solving a system of 3 unknowns instead of a system of 7 unknowns. And this process can be repeated recursively.

Moving a step further towards the use of multigrid methods as solvers of physical models, we encounter the concept of computational complexity. Consider a system described by n unknowns defining the

solution of an algebraic system. The best possible (optimal) solver is one that provides this solution requiring a number of operations which is linearly proportional to n. In a multigrid context this concept is somewhat refined by the following two remarks [7]: *The amount of computational work should be proportional to the amount of real physical changes in the computed system*; and *the solution of many problems is made of several components with different scales, which interact with each other*. Therefore, in the discretization of a continuous problem, one should require that the number of unknowns representing the solution should be proportional to the number of the physical features to be described. So, for example, a smooth function can be represented by a few of its values, whereas high resolution is required for highly oscillating functions. As a consequence, to describe a solution of a given problem the use of many scales of discretization is appropriate to represent all components of the solution. This is one essential aspect of the multigrid strategy.

For the purpose of this introduction we now focus on multigrid (MG) methods for solving discretized partial differential equations and make some historical remarks. Already in the sixties R.P. Fedorenko [15, 16] developed the first multigrid scheme for the solution of the Poisson equation in a unit square. Since then, other mathematicians extended Fedorenko's idea to general elliptic boundary value problems with variable coefficients; see, e.g., [1]. However, the full efficiency of the multigrid approach was realized after the works of A. Brandt [3, 4] and W. Hackbusch [18]. These Authors also introduced multigrid methods for nonlinear problems like the multigrid *full approximation storage* (FAS) scheme [4, 21]. Another achievement in the formulation of multigrid methods was the *full multigrid* (FMG) scheme [4, 21], based on the combination of nested iteration techniques and multigrid methods. Multigrid algorithms are now applied to a wide range of problems, primarily to solve linear and nonlinear boundary value prob-

lems. Other examples of successful applications are eigenvalue problems [6, 19], bifurcation problems [33, 37], parabolic problems [10, 20, 41], hyperbolic problems [13, 34], and mixed elliptic/hyperbolic problems, optimization problems [38], etc.. For additional references see, e.g., [23, 24, 25, 27, 32, 36, 42] and references therein. Most recent developments of solvers based on the multigrid strategy are algebraic multigrid (AMG) methods [35] that resemble the geometric multigrid process utilizing only information contained in the algebraic system to be solved. In addition to partial differential equations (PDE), Fredholm's integral equations can also be efficiently solved by multigrid methods [9, 22, 26]. These schemes can be used to solve reformulated boundary value problems [21] or for the fast solution of N-body problems [2, 11]. Another example of applications are lattice field computations and quantum electrodynamics and chromodynamics simulations [12, 14, 17, 29, 31].

Also convergence analysis of multigrid methods has developed along with the multitude of applications. In these Lecture Notes we only provide introductory remarks. For results and references on MG convergence theory see the Readings given below.

An essential contribution to the development of the multigrid community is the MGNet of Craig C. Douglas. This is the communication platform on everything related to multigrid methods;
see `http://www.mgnet.org`.

Readings

1. J.H. Bramble, *Multigrid Methods*, Pitman research notes in mathematical series, Longman Scientific & Technical, 1993.

2. A. Brandt, *Multi-grid techniques: 1984 guide with applications to fluid dynamics*, GMD-Studien. no 85, St. Augustin, Germany, 1984.

3. W.L. Briggs, *A Multigrid Tutorial*, SIAM, Philadelphia, 1987.

4. W. Hackbusch, *Multi-Grid Methods and Applications*, Springer-Verlag, Heidelberg, 1985.

5. S.F. McCormick, *Multigrid Methods*, Frontiers in Applied Mathematics, vol. 3, SIAM Books, Philadelphia, 1987.

6. U. Trottenberg, C. Oosterlee, and A. Schüller, *Multigrid*, Academic Press, London, 2001.

7. P. Wesseling, *An introduction to multigrid methods*, John Wiley, Chichester, 1992.

2 Multigrid methods for linear problems

In this section, the basic components of a multigrid algorithm are presented. We start with the analysis of two standard iterative techniques: the Jacobi and Gauss-Seidel schemes. These two methods are characterized by global poor convergence rates, however, for errors whose length scales are comparable to the grid mesh size, they provide rapid damping, leaving behind smooth, longer wave-length errors. These smooth components are responsible for the slow global convergence. A multigrid algorithm, employing grids of different mesh sizes, allows to solve all wave-length components and provides rapid convergence rates. The multigrid strategy combines two complementary schemes. The high-frequency components of the error are reduced applying iterative methods like Jacobi or Gauss-Seidel schemes. For this reason these methods are called smoothers. On the other hand, low-frequency error components are effectively reduced by a coarse-grid correction procedure. Because the action of a smoothing iteration leaves only smooth error components, it is possible to represent them as the solution of an appropriate coarser system. Once this coarser problem is solved, its

solution is interpolated back to the fine grid to correct the fine grid approximation for its low-frequency errors.

2.1 Iterative methods and the smoothing property

Consider a large sparse linear system of equations $Au = f$, where A is a $n \times n$ matrix. Iterative methods for solving this problem are formulated as follows

$$u^{(\nu+1)} = Mu^{(\nu)} + Nf \ , \tag{1}$$

where M and N have to be constructed in such a way that given an arbitrary initial vector $u^{(0)}$, the sequence $u^{(\nu)}$, $\nu = 0, 1, \ldots$, converges to the solution $u = A^{-1}f$. Defining the solution error at the sweep ν as $e^{(\nu)} = u - u^{(\nu)}$, iteration (1) is equivalent to $e^{(\nu+1)} = Me^{(\nu)}$. M is called the *iteration matrix*. We can state the following convergence criterion based on the *spectral radius* $\rho(M)$ of the matrix M.

Theorem 1 *The method (1) converges if and only if* $\rho(M) < 1$.

A general framework to define iterative schemes of the type (1) is based on the concept of *splitting*. Assume the splitting $A = B - C$ where B is non singular. By setting $Bu^{(\nu+1)} - Cu^{(\nu)} = f$ and solving with respect to $u^{(\nu+1)}$ one obtains

$$u^{(\nu+1)} = B^{-1}Cu^{(\nu)} + B^{-1}f \ .$$

Thus $M = B^{-1}C$ and $N = B^{-1}$. Typically, one considers the regular splitting $A = D - L - U$ where $D = \mathrm{diag}(a_{11}, a_{22}, \ldots, a_{nn})$ denotes the diagonal part of the matrix A, and $-L$ and $-U$ are the strictly lower and upper parts of A, respectively. Based on this splitting many choices for B and C are possible leading to different iterative schemes. For example, the choice $B = \frac{1}{\omega}D$ and $C = \frac{1}{\omega}[(1-\omega)D + \omega(L+U)]$, $0 < \omega \leq 1$, leads to the *damped* Jacobi iteration

$$u^{(\nu+1)} = (I - \omega D^{-1}A)u^{(\nu)} + \omega D^{-1}f \ . \tag{2}$$

Choosing $B = D - L$ and $C = U$, one obtains the Gauss-Seidel iteration

$$u^{(\nu+1)} = (D-L)^{-1}Uu^{(\nu)} + (D-L)^{-1}f \ . \tag{3}$$

Later on we denote the iteration matrices corresponding to (2) and (3) with $M_J(\omega)$ and M_{GS}, respectively.

To define and analyze the smoothing property of these iterations we introduce a simple model problem. Consider the finite-difference approximation of a one-dimensional Dirichlet boundary value problem:

$$\begin{cases} -\frac{d^2u}{dx^2} = f(x), & \text{in} \quad \Omega = (0,1) \\ u(x) = g(x), & \text{on} \quad x \in \{0,1\} \ . \end{cases} \tag{4}$$

Let Ω be represented by a grid Ω_h with grid size $h = \frac{1}{n+1}$ and grid points $x_j = jh$, $j = 0, 1, \ldots, n+1$. A discretization scheme for the second derivative at x_j is $h^{-2}[u(x_{j-1}) - 2u(x_j) + u(x_{j+1})] = u''(x_j) + O(h^2)$. Set $f_j^h = f(jh)$, and $u_j^h = u(jh)$. We obtain the following tridiagonal system of n equations

$$\begin{aligned} \frac{2}{h^2}\,u_1^h - \frac{1}{h^2}\,u_2^h &= f_1^h + \frac{1}{h^2}\,g(0) \\ -\frac{1}{h^2}\,u_{j-1}^h + \frac{2}{h^2}\,u_j^h - \frac{1}{h^2}\,u_{j+1}^h &= f_j^h, \qquad j = 2, \ldots, n-1 \\ -\frac{1}{h^2}\,u_{n-1}^h + \frac{2}{h^2}\,u_n^h &= f_n^h + \frac{1}{h^2}\,g(1) \end{aligned} \tag{5}$$

Let us omit the discretization parameter h and denote (5) by $Au = f$. We discuss the solution to this problem by using the Jacobi iteration. Consider the eigenvalue problem for the damped Jacobi iteration, given by $M_J(\omega)v^k = \mu_k v^k$. The eigenvectors are given by

$$v^k = (\sin(k\pi hj))_{j=1,n} \ , \quad k = 1, \ldots, n \ , \tag{6}$$

and the corresponding eigenvalues are

$$\mu_k(\omega) = 1 - \omega(1 - \cos(k\pi h)) \ , \quad k = 1, \ldots, n \ . \tag{7}$$

We have that $\rho(M_J(\omega)) < 1$ for $0 < \omega \leq 1$, guaranteeing convergence. In particular, for the Jacobi iteration with $\omega = 1$ we have $\rho(M_J(1)) = 1 - \frac{1}{2}\pi h^2 + O(h^4)$, showing how the convergence of (2) deteriorates (i.e. ρ tends to 1) as $h \to 0$. However, the purpose of an iteration in a MG algorithm is primarily to be a smoothing operator. In order to define this property, we need to distinguish between low- and high-frequency eigenvectors. We define

- *low frequency* (LF) v^k with $1 \leq k < \frac{n}{2}$;
- *high frequency* (HF) v^k with $\frac{n}{2} \leq k \leq n$;

We now define the *smoothing factor* μ as the worst factor by which the amplitudes of HF components are damped per iteration. In the case of the Jacobi iteration we have

$$\begin{aligned} \mu &= \max\{|\mu_k|, \frac{n}{2} \leq k \leq n\} = \max\{1-\omega, \ |1-\omega(1-\cos(\pi)|\} \\ &\leq \max\{1-\omega, |1-2\omega|\}. \end{aligned}$$

Using this result we find that the optimal smoothing factor $\mu = 1/3$ is obtained choosing $\omega^* = 2/3$.

Now suppose to apply ν times the damped Jacobi iteration with $\omega = \omega^*$. Consider the following representation of the error $e^{(\nu)} = \sum_k e_k^{(\nu)} v^k$. The application of ν-times the damped Jacobi iteration on the initial error $e^{(0)}$ results in $e^{(\nu)} = M_J(\omega^*)^\nu e^{(0)}$. This implies for the expansion coefficients that $e_k^{(\nu)} = \mu_k(\omega^*)^\nu e_k^{(0)}$. Therefore a few sweeps of (2) give $|e_k^{(\nu)}| << |e_k^{(0)}|$ for high-frequency components. For this reason, although the global error decreases slowly by iteration, it is smoothed very quickly and this process does not depend on h.

Most often, instead of a Jacobi method other iterations are used that suppress the HF components of the error more efficiently. This is the case, for example, of the Gauss-Seidel iteration (3). The smoothing property of this scheme is conveniently analyzed by using *local mode analysis* (LMA) introduced by Brandt [4]. This is an effective tool for

analyzing the MG process even though it is based on certain idealized assumptions and simplifications: Boundary conditions are neglected and the problem is considered on infinite grids $G^h = \{jh,\, j \in \mathbb{Z}\}$. Notice that on G^h, only the components $e^{i\theta x/h}$ with $\theta \in (-\pi, \pi]$ are visible, i.e., there is no other component with frequency $\theta_0 \in (-\pi, \pi]$ with $|\theta_0| < \theta$ such that $e^{i\theta_0 x/h} = e^{i\theta x/h}$, $x \in G^h$.

In LMA the notion of low- and high-frequency components on the grid G^h is related to a coarser grid denoted by G^H. In this way $e^{i\theta x/h}$ on G^h is said to be an high-frequency component, with respect to the coarse grid G^H, if its restriction (projection) to G^H is not visible there. Usually $H = 2h$ and the high frequencies are those with $\frac{\pi}{2} \le |\theta| \le \pi$.

We give an example of local mode analysis considering the Gauss-Seidel scheme applied to our discretized model problem. A relaxation sweep starting from an initial approximation $u^{(\nu)}$ produces a new approximation $u^{(\nu+1)}$ such that the corresponding error satisfies

$$B_h e^{(\nu+1)}(x) - C_h e^{(\nu)}(x) = 0 \ , \quad x \in G^h \ , \tag{8}$$

where $B_h = D_h - L_h$ and $C_h = U_h$. To analyze this iteration we define the errors in terms of their θ components $e^{(\nu)} = \sum_\theta \mathcal{E}_\theta^{(\nu)} e^{i\theta x/h}$ and $e^{(\nu+1)} = \sum_\theta \mathcal{E}_\theta^{(\nu+1)} e^{i\theta x/h}$. Where $\mathcal{E}_\theta^{(\nu)}$ and $\mathcal{E}_\theta^{(\nu+1)}$ denote the error amplitudes of the θ component, before and after smoothing, respectively. Then, for a given θ, (8) reduces to

$$(2 - e^{-i\theta})\mathcal{E}_\theta^{(\nu+1)} - e^{i\theta}\mathcal{E}_\theta^{(\nu)} = 0 \ .$$

In the LMA framework, the smoothing factor is defined by

$$\mu = \max\{|\frac{\mathcal{E}_\theta^{(\nu+1)}}{\mathcal{E}_\theta^{(\nu)}}| \ , \ \frac{\pi}{2} \le |\theta| \le \pi\} \ , \tag{9}$$

For the present model problem we obtain

$$|\frac{\mathcal{E}_\theta^{(\nu+1)}}{\mathcal{E}_\theta^{(\nu)}}| = |\frac{e^{i\theta}}{2 - e^{-i\theta}}|,$$

and $\mu = 0.45$. Similar values are obtained by LMA for the Gauss-Seidel iteration applied to the two- and three-dimensional version of our model problem. For the two dimensional case the effect of smoothing can be seen in Figure 5.

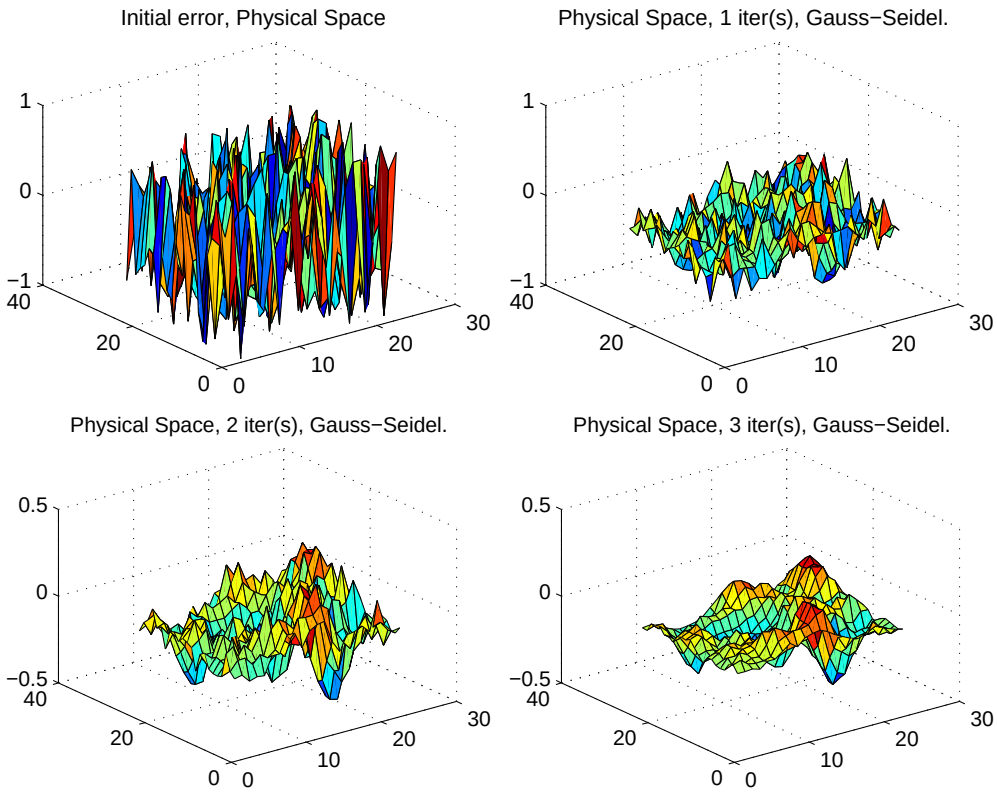

Figure 5: Smoothing by Gauss-Seidel iteration.

Another definition of smoothing property of an iterative scheme is due to Hackbusch [21]. Let M be the iteration matrix of a smoothing procedure and recall the relation $e^{(\nu)} = M^\nu e^{(0)}$. One can measure the smoothness of $e^{(\nu)}$ by a norm involving differences of the value of this error on different grid points. A natural choice is to take the second-order difference matrix A_h above. Then the following smoothing factor is defined

$$\mu(\nu) = \|A_h M^\nu\| / \|A_h\|. \tag{10}$$

The iteration defined by M is said to possess the smoothing property if there exists a function $\eta(\nu)$ such that, for sufficiently large ν, we have

$$\|A_h M^\nu\| \leq \eta(\nu)/h^\alpha,$$

where $\alpha > 0$ and $\eta(\nu) \to 0$ as $\nu \to \infty$. For our model problem using the damped Jacobi iteration, $\alpha = 2$ and $\eta(\nu) = \nu^\nu/(\nu+1)^{(\nu+1)}$. For the Gauss-Seidel iteration we have $\alpha = 2$ and $\eta(\nu) = c/\nu$ with c a positive constant.

2.2 The twogrid scheme and the approximation property

After the application of a few smoothing sweeps, we obtain an approximation $\tilde{u}_h$ whose error $\tilde{e}_h = u_h - \tilde{u}_h$ is smooth. Then $\tilde{e}_h$ can be approximated on a coarser space. We need to express this smooth error as solution of a coarse problem, whose matrix A_H and right-hand side have to be defined. For this purpose notice that in our model problem, A_h is a second-order difference operator and the *residual* $r_h = f_h - A_h\tilde{u}_h$ is a smooth function if $\tilde{e}_h$ is smooth. Obviously the original equation $A_h u_h = f_h$ and the residual equation $A_h\tilde{e}_h = r_h$ are equivalent. The difference is that $\tilde{e}_h$ and r_h are smooth, therefore we can think of representing them on a coarser grid with mesh size $H = 2h$. We define r_H as the restriction of the fine-grid residual to the coarse grid, that is, $r_H = I_h^H\, r_h$, where I_h^H is a suitable *restriction* operator (for example, injection). This defines the right-hand side of the coarse problem. Since $\tilde{e}_h$ is the solution of a difference operator which can be represented analogously on the coarse discretization level, we define the following coarse problem

$$A_H\tilde{e}_H = r_H \ , \tag{11}$$

with homogeneous Dirichlet boundary conditions as for $\tilde{e}_h$. Here A_H represents the same discrete operator but relative to the grid with mesh size H. Reasonably one expects that $\tilde{e}_H$ be an approximation to $\tilde{e}_h$ on

the coarse grid. Because of its smoothness, we can apply a *prolongation* operator I_H^h to transfer $\tilde{e}_H$ to the fine grid. Therefore, since by definition $u^h = \tilde{u}^h + \tilde{e}^h$, we update the function $\tilde{u}_h$ applying the following *coarse-grid correction* (CGC) step

$$\tilde{u}_h^{new} = \tilde{u}_h + I_H^h \tilde{e}_H.$$

Notice that $\tilde{e}_h$ was a smooth function and the last step has amended $\tilde{u}_h$ by its smooth error. In practice, also the interpolation procedure may introduce HF errors on the fine grid. Therefore it is convenient to complete the twogrid process by applying ν_2 post-smoothing sweeps after the coarse-grid correction.

We summarize the twogrid (TG) procedure with the following algorithm. To emphasize that the iteration $u_h^{(l)} = Mu_h^{(l-1)} + Nf_h$ is a smoothing procedure, we denote it by $u_h^{(l)} = S_h(u_h^{(l-1)}, f_h)$. When no confusion may arise, we also use S to denote the iteration matrix M.

Algorithm 1 (TG scheme)

- *Twogrid method for solving* $A_h u_h = f_h$.
  1. *Pre-smoothing steps on the fine grid:* $u_h^{(l)} = S(u_h^{(l-1)}, f_h)$, $l = 1, \ldots, \nu_1$;
  2. *Computation of the residual:* $r_h = f_h - A_h u_h^{(\nu_1)}$;
  3. *Restriction of the residual:* $r_H = I_h^H r_h$;
  4. *Solution of the coarse-grid problem* $e_H = (A_H)^{-1} r_H$;
  5. *Coarse-grid correction:* $u_h^{(\nu_1+1)} = u^{(\nu_1)} + I_H^h e_H$;
  6. *Post-smoothing steps on the fine grid:* $u_h^{(l)} = S(u_h^{(l-1)}, f_h)$, $l = \nu_1 + 2, \ldots, \nu_1 + \nu_2 + 1$;

A TG scheme starts at the fine level with pre-smoothing, performs a CGC correction solving a coarse-grid auxiliary problem, and ends with post-smoothing. A pictorial representation of this process where 'fine'

is a high level and 'coarse' is a low level looks like a 'V' workflow. This is called V cycle. To solve the problem to a given tolerance, we have to apply the TG V-cycle repeatedly (iteratively). Actually, the TG scheme can be written in the form (1) as stated by the following

Lemma 1 *The iteration matrix of the twogrid scheme is*

$$M_{TG} = S_h^{\nu_2}(I_h - I_H^h(A_H)^{-1}I_h^H A_h)S_h^{\nu_1} \quad , \tag{12}$$

where I_h is the identity and S_h is the smoothing iteration matrix.

For the model problem considered here, it is possible to estimate the spectral radius of M_{TG}. Consider the damped Jacobi smoother with $\omega = 1/2$, assume that I_H^h is piecewise linear interpolation [21], and I_h^H is restriction by weighting such that $r_H(x_j) = (r_h(x_{j-1}) + 2\,r_h(x_j) + r_h(x_{j+1}))/4$, $j = 2, 4, \ldots, n-1$. In this case the following theorem is proved [21] using discrete Fourier analysis

Theorem 2 *Let the TG scheme 1 with $\nu = \nu_1 + \nu_2 \geq 1$. The spectral radius of the iteration matrix M_{TG} given by (12) is bounded by*

$$\rho(M_{TG}) \leq \max\{\chi(1-\chi)^\nu + (1-\chi)\chi^\nu \;\; : \;\; 0 \leq \chi \leq 1/2\} =: \rho_\nu < 1 \quad ,$$

uniformly with respect to the mesh size h. Hence (12) is a convergent iteration.

In the framework of LMA, a simple and effective way to predict the convergence factor of the TG scheme, for usually moderate values of ν, is to assume that the coarse-grid correction step solves 'exactly' the LF error components, and there is no interaction between high- and low-frequency components. This can be considered an 'ideal' situation. Then the reduction of a suitable norm of the error (e.g., discrete L^2-norm) by one V cycle of the TG method is determined by the reduction of the HF components on the fine grid. For this reason the reduction (convergence) factor, denoted by ρ_{LMA}, can be estimated by

$$\rho_{LMA} = \mu^{\nu_1+\nu_2} \quad . \tag{13}$$

A sharper bound can be computed by *twogrid local mode analysis* [8]. In Table 1, we report estimates of ρ_{LMA} as given by (13) and estimates of ρ_ν by Theorem 2. Notice that the estimate ρ_{LMA} approximates well the bound ρ_ν provided that $\nu_1 + \nu_2$ is small. For large ν, ρ_{LMA} has an exponential behavior whereas $\rho_\nu \simeq 1/e\nu$.

ν	ρ_{LMA}	ρ_ν
1	0.5	0.5
2	0.25	0.25
3	0.125	0.125
4	0.0625	0.0832
5	0.03125	0.0671

Table 1: Comparison of error reduction factors as given by LMA and by Theorem 2.

Notice that measuring ρ requires the knowledge of the exact solution. Because this is usually not available, ρ is measured as the asymptotic value of the reduction of a suitable norm of the residuals after consecutive TG cycles.

Together with the definition of the smoothing property (10), Hackbusch introduces the *approximation property* to measure how good the coarse-grid solution approximates the fine grid solution. This is expressed by the following estimate

$$\|A_h^{-1} - I_H^h \, A_H^{-1} I_h^H\| \leq c_A \, h^\beta. \tag{14}$$

Notice that this estimate actually measures the accuracy between $u_h = A_h^{-1} \, f_h$ and $I_H^h u_H$ where $u_H = A_H^{-1} I_h^H \, f_h$.

In our model problem we have $\beta = 2$. Using the estimates of the smoothing and approximation properties, one can prove convergence of the TG scheme as follows. Consider, for simplicity, $\nu_2 = 0$, i.e. only

pre-smoothing is applied. Then for our model problem we have

$$\begin{aligned} \|M_{TG}\| &= \|(I_h - I_H^h (A_H)^{-1} I_h^H A_h) S_h^{\nu_1}\| \\ &= \|(A_h^{-1} - I_H^h (A_H)^{-1} I_h^H) A_h S_h^{\nu_1}\| \\ &\leq \|A_h^{-1} - I_H^h (A_H)^{-1} I_h^H\| \|A_h S_h^{\nu_1}\| \\ &\leq c_A \, \eta(\nu_1), \end{aligned} \tag{15}$$

where $c_A \, \eta(\nu_1) < 1$ for sufficiently large ν_1. Notice that the coarse-grid correction without pre- and post-smoothing is not a convergent iteration, in general. In fact, I_h^H maps from a fine- to a coarse-dimensional space and $I_H^h (A_H)^{-1} I_h^H$ is not invertible. We have $\rho(I_h - I_H^h (A_H)^{-1} I_h^H A_h) \geq 1$.

2.3 The multigrid scheme

In the TG scheme the size of the coarse grid is twice larger than the fine one, thus the coarse problem may be very large. However, the coarse problem has the same form as the residual problem on the fine level. Therefore one can use the TG method to determine $\tilde{e}_H$, i.e. equation (11) is itself solved by TG cycling introducing a further coarse-grid problem. This process can be repeated recursively until a coarsest grid is reached where the corresponding residual equation is inexpensive to solve. This is, roughly speaking, the qualitative description of the multigrid method.

For a more detailed description, let us introduce a sequence of grids with mesh size $h_1 > h_2 > \cdots > h_L > 0$, so that $h_{k-1} = 2\, h_k$. Here $k = 1, 2, \ldots, L$, is called the *level number*. With Ω_{h_k} we denote the set of grid points with grid spacing h_k. The number of interior grid points will be n_k.

On each level k we define the problem $A_k u_k = f_k$. Here A_k is a $n_k \times n_k$ matrix, and u_k, f_k are vectors of size n_k. The transfer among levels is performed by two linear mappings. The restriction I_k^{k-1} and

the prolongation I_{k-1}^k operators. With $u_k = S_k(u_k, f_k)$ we denote a smoothing iteration. The following defines the multigrid algorithm

Algorithm 2 (MG scheme)

- *Multigrid method for solving* $A_k u_k = f_k$.
  1. *If* $k = 1$ *solve* $A_k u_k = f_k$ *directly.*
  2. *Pre-smoothing steps on the fine grid:* $u_k^{(l)} = S(u_k^{(l-1)}, f_k)$, $l = 1, \ldots, \nu_1$;
  3. *Computation of the residual:* $r_k = f_k - A_k u_k^{(\nu_1)}$;
  4. *Restriction of the residual:* $r_{k-1} = I_k^{k-1} r_k$;
  5. *Set* $u_{k-1} = 0$;
  6. *Call* γ *times the MG scheme to solve* $A_{k-1} u_{k-1} = r_{k-1}$;
  7. *Coarse-grid correction:* $u_k^{(\nu_1+1)} = u_k^{(\nu_1)} + I_{k-1}^k u_{k-1}$;
  8. *Post-smoothing steps on the fine grid:* $u_k^{(l)} = S(u_k^{(l-1)}, f_k)$, $l = \nu_1 + 2, \ldots, \nu_1 + \nu_2 + 1$;

The multigrid algorithm involves a new parameter (cycle index) γ which is the number of times the MG procedure is applied to the coarse level problem. Since this procedure converges very fast, $\gamma = 1$ or $\gamma = 2$ are the typical values used. For $\gamma = 1$ the multigrid scheme is called V-cycle, whereas $\gamma = 2$ is named W-cycle. It turns out that with a reasonable γ, the coarse problem is solved almost exactly. Therefore in this case the convergence factor of a multigrid cycle equals that of the corresponding TG method, i.e., approximately $\rho = \mu^{\nu_1+\nu_2}$. Actually in many problems $\gamma = 2$ or even $\gamma = 1$ are sufficient to retain the twogrid convergence.

Also the multigrid scheme can be expressed in the form (1) as stated by the following lemma

Lemma 2 *The iteration matrix of the multigrid scheme is given in recursive form by the following.*

For $k = 1$ *let* $M_1 = 0$. *For* $k = 2, \ldots, L$:

$$M_k = S_k^{\nu_2}(I_k - I_{k-1}^k(I_{k-1} - M_{k-1}^{\gamma})A_{k-1}^{-1}I_k^{k-1}A_k)S_k^{\nu_1} \tag{16}$$

where I_k *is the identity,* S_k *is the smoothing iteration matrix, and* M_k *is the multigrid iteration matrix for the level* k.

To derive (16) consider an initial error $e_k^{(0)}$. The action of ν_1 pre-smoothing steps gives $e_k = S_k^{\nu_1} e_k^{(0)}$ and the corresponding residual $r_k = A_k e_k$. On the coarse grid, this error is given by $e_{k-1} = A_{k-1}^{-1} I_k^{k-1} r_k$. However, in the multigrid algorithm we do not invert A_{k-1} (unless on the coarsest grid) and we apply γ multigrid cycles instead. That is, denote with $v_{k-1}^{(\gamma)}$ the approximation to e_{k-1} obtained after γ application of M_{k-1}, we have

$$e_{k-1} - v_{k-1}^{(\gamma)} = M_{k-1}^{(\gamma)}(e_{k-1} - v_{k-1}^{(0)}).$$

Following the MG Algorithm 2, we set $v_{k-1}^{(0)} = 0$ (Step 5.). Therefore we have $e_{k-1} - v_{k-1}^{(\gamma)} = M_{k-1}^{(\gamma)} e_{k-1}$ which can be rewritten as $v_{k-1}^{(\gamma)} = (I_{k-1} - M_{k-1}^{(\gamma)})e_{k-1}$. It follows that

$$\begin{aligned} v_{k-1}^{(\gamma)} &= (I_{k-1} - M_{k-1}^{(\gamma)})e_{k-1} = (I_{k-1} - M_{k-1}^{(\gamma)})\, A_{k-1}^{-1} I_k^{k-1}\, r_k \\ &= (I_{k-1} - M_{k-1}^{(\gamma)})\, A_{k-1}^{-1} I_k^{k-1}\, A_k e_k. \end{aligned}$$

Correspondingly, the coarse-grid correction is $u_k^{(\nu_1+1)} = u_k^{(\nu_1)} + I_{k-1}^k v_{k-1}^{(\gamma)}$. In terms of error functions this means that

$$e_k^{(\nu_1+1)} = e_k - I_{k-1}^k v_{k-1}^{(\gamma)},$$

substituting the expression for $v_{k-1}^{(\gamma)}$ given above, we obtain

$$e_k^{(\nu_1+1)} = [I_k - I_{k-1}^k(I_{k-1} - M_{k-1}^{(\gamma)})\, A_{k-1}^{-1} I_k^{k-1}\, A_k] e_k.$$

Finally, consideration of the pre- and post-smoothing sweeps proves the Lemma.

A picture of the multigrid workflow is given in Figure 6.

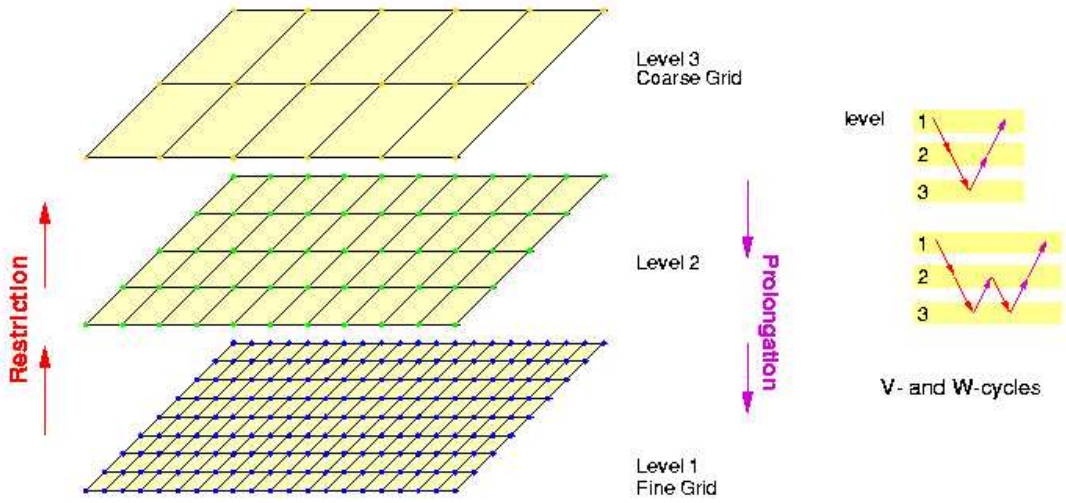

Figure 6: Multigrid setting.

3 Multigrid methods for nonlinear problems

Many problems of physical interest are nonlinear in character and for these problems the multigrid strategy provides new powerful algorithms. Roughly speaking there are two multigrid approaches to nonlinear problems. The first is based on the generalization of the MG scheme described above. The second is based on the Newton method and uses the multigrid scheme as inner solver of the linearized equations. In this respect the latter approach does not belong, conceptually, to the class of multigrid methods. The well known representative of the former approach is the *full approximation storage* (FAS) scheme [4].

An interesting comparison between the FAS and the MG-Newton schemes is presented in [28]. First, it is shown that, in terms of computing time, the exact Newton approach is not a viable method. In the case of multigrid-inexact-Newton scheme it is shown that this approach may have similar efficiency as the FAS scheme. However, it remains an open issue how many interior MG cycles are possibly needed in the MG-Newton method to match the FAS efficiency. In this sense the FAS scheme is more robust and we discuss this method in the following.

In the FAS scheme the global linearization in the Newton process is avoided and the only possible linearization involved is a local one in order to define the relaxation procedure, e.g., Gauss-Seidel-Newton. To illustrate the FAS method, consider the nonlinear problem $A(u) = f$, and denote its discretization by

$$A_k(u_k) = f_k \ , \tag{17}$$

where $A_k(\cdot)$ represents a nonlinear discrete operator on Ω_{h_k}.

The starting point for the FAS scheme is again to define a suitable smoothing iteration. We denote the corresponding smoothing process by $u = S(u, f)$. Now suppose to apply a few times this iterative method to (17) obtaining some approximate solution $\tilde{u}_k$. The desired exact correction e_k is defined by $A_k(\tilde{u}_k + e_k) = f_k$. Here the coarse residual equation (11) makes no sense (no superposition). Nevertheless the 'correction' equation can instead be written in the form

$$A_k(\tilde{u}_k + e_k) - A_k(\tilde{u}_k) = r_k \ , \tag{18}$$

if we define $r_k = f_k - A_k(\tilde{u}_k)$. Assume to represent $\tilde{u}_k + e_k$ on the coarse grid in terms of the coarse-grid variable

$$\hat{u}_{k-1} := \hat{I}_k^{k-1}\tilde{u}_k + e_{k-1} \ . \tag{19}$$

Since $\hat{I}_k^{k-1}\tilde{u}_k$ and $\tilde{u}_k$ represent the same function but on different grids, the standard choice of the fine-to-coarse linear operator $\hat{I}_k^{k-1}$ is the straight injection. Formulating (18) on the coarse level by replacing $A_k(\cdot)$ by $A_{k-1}(\cdot)$, $\tilde{u}_k$ by $\hat{I}_k^{k-1}\tilde{u}_k$, and r_k by $I_k^{k-1}r_k = I_k^{k-1}(f_k - A_k(\tilde{u}_k))$, we get the FAS equation

$$A_{k-1}(\hat{u}_{k-1}) = I_k^{k-1}(f_k - A_k(\tilde{u}_k)) + A_{k-1}(\hat{I}_k^{k-1}\tilde{u}_k) \ . \tag{20}$$

This equation is also written in the form

$$A_{k-1}(\hat{u}_{k-1}) = I_k^{k-1}f_k + \tau_k^{k-1} \ , \tag{21}$$

where

$$\tau_k^{k-1} = A_{k-1}(\hat{I}_k^{k-1}\tilde{u}_k) - I_k^{k-1}A_k(\tilde{u}_k).$$

Observe that (21) without the τ_k^{k-1} term is the original equation represented on the coarse grid. At convergence $\hat{u}_{k-1} = \hat{I}_k^{k-1}u_k$, because $f_k - A_k(u_k) = 0$ and $A_{k-1}(\hat{u}_{k-1}) = A_{k-1}(\hat{I}_k^{k-1}u_k)$. The term τ_k^{k-1} is the fine-to-coarse *defect or residual correction*. That is, the correction to (21) such that its solution coincides with the fine grid solution. This fact allows to reverse the point of view of the multigrid approach [7]. Instead of regarding the coarse level as a device for accelerating convergence on the fine grid, one can view the fine grid as a device for calculating the correction τ_k^{k-1} to the FAS equation. In this way most of the calculation may proceed on coarser spaces.

The direct use of $\hat{u}_{k-1}$ on fine grids, that is, the direct interpolation of this function by $I_{k-1}^k\hat{u}_{k-1}$ cannot be used, since it introduces the interpolation errors of the full (possibly oscillatory) solution, instead of the interpolation errors of only the correction e_{k-1}, which in principle is smooth because of the application of the smoothing iteration. For this reason the following coarse-grid correction is used

$$u_k = \tilde{u}_k + I_{k-1}^k(\hat{u}_{k-1} - \hat{I}_k^{k-1}\tilde{u}_k) \ . \tag{22}$$

The complete FAS scheme is summarized below

Algorithm 3 (FAS scheme)

- *Multigrid FAS method for solving* $A_k(u_k) = f_k$.
  1. *If* $k = 1$ *solve* $A_k(u_k) = f_k$ *directly.*
  2. *Pre-smoothing steps on the fine grid:* $u_k^{(l)} = S(u_k^{(l-1)}, f_k)$, $l = 1, \ldots, \nu_1$;
  3. *Computation of the residual:* $r_k = f_k - A_k u_k^{(\nu_1)}$;
  4. *Restriction of the residual:* $r_{k-1} = I_k^{k-1}r_k$;
  5. *Set* $u_{k-1} = \hat{I}_k^{k-1}u_k^{(\nu_1)}$;

6. *Set* $f_{k-1} = r_{k-1} + A_{k-1}(u_{k-1})$
7. *Call* γ *times the FAS scheme to solve* $A_{k-1}(u_{k-1}) = f_{k-1}$*;*
8. *Coarse-grid correction:* $u_k^{(\nu_1+1)} = u_k^{(\nu_1)} + I_{k-1}^k(u_{k-1} - \hat{I}_k^{k-1} u_k^{(\nu_1)})$*;*
9. *Post-smoothing steps on the fine grid:* $u_k^{(l)} = S(u_k^{(l-1)}, f_k)$, $l = \nu_1 + 2, \ldots, \nu_1 + \nu_2 + 1$*;*

4 The full multigrid method

When dealing with nonlinear problems it is sometimes important to start the iterative procedure from a good initial approximation. The multigrid setting suggests a natural way of how to get this approximation cheaply. Suppose to start the solution process from a coarse working level (present fine grid) $\ell < M$ where the discretized problem $A_\ell(u_\ell) = f_\ell$ is easily solved. We can usually interpolate this solution to the next finer working level as initial approximation for the iterative process to solve $A_{\ell+1}(u_{\ell+1}) = f_{\ell+1}$. We apply the interpolation

$$u_{\ell+1} = \tilde{I}_\ell^{\ell+1} u_\ell \ , \tag{23}$$

and then start the FAS (or MG) solution process at level $\ell+1$. The idea of using a coarse-grid approximation as a first guess for the solution process on a finer grid is known as *nested iteration*. The algorithm obtained by combining the multigrid scheme with nested iteration is called *full multigrid* (FMG) method; see Figure 7. The interpolation operator (23) used in the FMG scheme is called FMG interpolator. Because of the improvement on the initial solution at each starting level, the FMG scheme results to be cheaper than the iterative application of the multigrid cycle without FMG initialization.

Algorithm 4 (FMG scheme)

- *FMG method for solving* $A_L(u_L) = f_L$*.*

1. *Set (compute)* u_ℓ *on the working level* ℓ*;*
2. *If* $\ell < L$ *then FMG interpolate to the next finer working level:* $u_{\ell+1} = \tilde{I}_\ell^{\ell+1} u_\ell$*;*
3. *Apply FAS (or MG) scheme to* $A_{\ell+1}(u_{\ell+1}) = f_{\ell+1}$*, starting with* $u_{\ell+1}$*;*
4. *If* $\ell + 1 < L$ *set* $\ell := \ell + 1$ *go to* 2.

On each current working level one applies N FAS cycles and then the algorithm is called N-FMG scheme. With the N-FMG algorithm an estimate of the degree of accuracy can be obtained by comparison of solutions on different levels. Denote with u_ℓ the solution on the level ℓ after N FAS (or MG) cycles. Then in the FMG method this solution is interpolated to level $\ell + 1$ to serve as a first approximation for this working level. At the end of N cycles on this level, one obtains $u_{\ell+1}$. An estimate of the solution error on level ℓ can be defined as

$$E_\ell = \max_{\Omega_{h_\ell}} |u_\ell - \hat{I}_{\ell+1}^{\ell} u_{\ell+1}| \ , \tag{24}$$

and so on finer levels.

In order to show the efficiency of the full multigrid approach we consider a three dimensional Poisson equation with Dirichlet boundary conditions:

$$\begin{cases} -(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}) = 3\sin(x+y+z) & \text{in} \quad \Omega = (0,2)^3 \\ u(x,y,z) = \sin(x+y+z) & \text{for} \quad (x,y,z) \in \partial\Omega \end{cases} \tag{25}$$

Finite difference approximations to (25) are obtained in much the same way as was done in the one-space variable case. If on each level ℓ, we choose a uniform mesh with grid size $h_\ell = \frac{2}{n_\ell+1}$, $n_\ell = 2^\ell - 1$, a direct application of the discretization scheme used in the 1D case gives

$$-(\frac{u_{i+1jk}^\ell - 2u_{ijk}^\ell + u_{i-1jk}^\ell}{h_\ell^2} + \frac{u_{ij+1k}^\ell - 2u_{ijk}^\ell + u_{ij-1k}^\ell}{h_\ell^2} + \frac{u_{ijk+1}^\ell - 2u_{ijk}^\ell + u_{ijk-1}^\ell}{h_\ell^2})$$

$$= 3\sin(ih_\ell, jh_\ell, kh_\ell) \ , \ \ i,j,k = 1,\ldots,n_\ell \ . \tag{26}$$

This is the 7-point stencil approximation which is $O(h^2)$ accurate. To solve this problem we employ the FAS scheme with $\gamma = 1$ and $L = 7$. The smoothing iteration is the lexicographic Gauss-Seidel method, applied $\nu_1 = 2$ times for pre-smoothing and $\nu_2 = 1$ times as post-smoother. Its smoothing factor estimated by local mode analysis is $\mu = 0.567$. Hence, by this analysis, the expected reduction factor is $\rho^* = \mu^{\nu_1+\nu_2} = 0.18$. The observed reduction factor is ≈ 0.20.

Here $\hat{I}_k^{k-1}$ is simple injection and I_k^{k-1} and I_{k-1}^k are the full weighting and (tri-)linear interpolation, respectively. Finally, the FMG operator $\tilde{I}_\ell^{\ell+1}$ is cubic interpolation [7, 21]. Now let us discuss the optimality of the FMG algorithm constructed with these components.

To show that the FMG scheme is able to solve the given discrete problem at a minimal cost, we compute E_ℓ and show that it behaves like h_ℓ^2, demonstrating convergence. This means that the ratio $E_\ell/E_{\ell+1}$ at convergence should be a factor $h_\ell^2/h_{\ell+1}^2 = 4$. Results are reported in Table 2. In the 10-FMG column of we actually observe the h^2 behavior which can also be seen with smaller $N = 3$ and $N = 1$ FMG schemes.

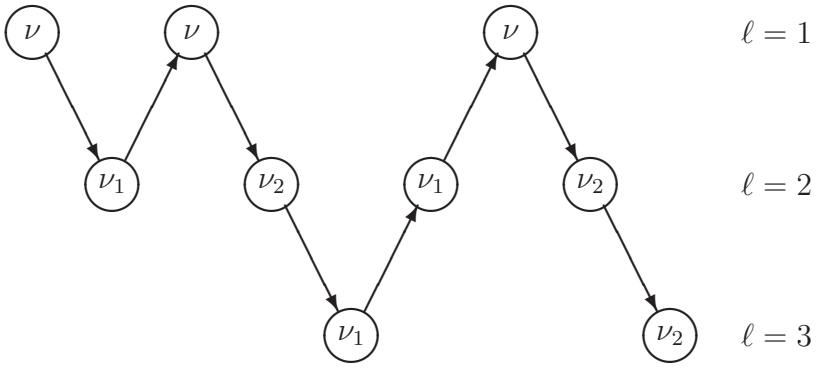

Figure 7: The FMG scheme.

In fact, as reported in Table 2, the 1-FMG scheme gives the same order of magnitude of errors as the 10-FMG scheme. Therefore the choice $N = 1$ in a FMG cycle is suitable to solve the problem to second-order accuracy.

level	10-FMG	3-FMG	1-FMG
3	$6.75\,10^{-4}$	$6.79\ \ 10^{-4}$	9.44×10^{-4}
4	$1.73\,10^{-4}$	$1.75\ \ 10^{-4}$	2.34×10^{-4}
5	$4.36\,10^{-5}$	$4.40\ \ 10^{-5}$	5.92×10^{-5}
6	$1.09\,10^{-5}$	$1.10\ \ 10^{-5}$	1.48×10^{-5}

Table 2: The estimated solution error for various N-FMG cycles.

To estimate the amount of work invested in the FMG method, let us define the work unit (WU) [4], i.e. the computational work of one smoothing sweep on the finest level M. The number of corresponding arithmetic operations is linearly proportional to the number of grid points. On the level $\ell \leq M$ the work involved is $(\frac{1}{2})^{3(M-\ell)}WU$, where the factor $\frac{1}{2}$ is given by the mesh size ratio $h_{\ell+1}/h_\ell$ and the exponent 3 is the number of spatial dimensions. Thus a multigrid cycle that uses $\nu = \nu_1 + \nu_2$ relaxation sweeps on each level requires

$$W_{cycle} = \nu \sum_{k=1}^{L} (\frac{1}{2})^{3(L-k)} WU < \frac{8}{7}\nu WU \ \ ,$$

ignoring transfer operations. Hence the computational work employed in a N-FMG method is roughly

$$W_{FMG} = N \sum_{\ell=2}^{L} (\frac{1}{2})^{3(L-\ell)} W_{cycle} \ \ ,$$

ignoring the FMG interpolation and work on the coarsest grid. This means that, using the 1-FMG method, we solve the discrete 3D Poisson problem to second-order accuracy with a number of computer operations which is proportional to the number of unknowns on the finest grid. In the present case we have FMG Work $\approx 4WU$.

5 Appendix: A multigrid code

The following is a FORTRAN 77 multigrid algorithm for the solution of the Poisson equation on a rectangle subject to Dirichlet boundary conditions; see [3, 4] for details.

```
      PROGRAM CYCLEV
C
C    MULTI-GRID ALGORITHM FOR THE SOLUTION OF THE POISSON PROBLEM:
C              DELTA(U)=F
C
C     EXPLANATIONS OF PARAMETERS:
C----------------------------
C
C  NX1-    NUMBER OF INTERVALS INX-DIRECTION ON THE COARSEST GRID
C  NY1-    NUMBER OF INTERVALS IN Y-DIRECTION ON THE COARSEST GRID
C  H1-     LENGTH OF EACH INTERVAL
C  M-      NUMBER OF LEVELS
C  NU1- NUMBER OF RELAXATION SWEEPS IN EACH CYCLE BEFORE TRAN-
C  SFER TO THE COARSER GRID
C  NU2-    NUMBER OF SWEEPS IN EACH CYCLE AFTER COMING BACK FROM
C  THE COARSER GRID
C  NCYC-   NUMBER OF CYCLES
C  IFAS-   IFAS=1 FAS SCHEME, IFAS=0 MG SCHEME
C
C  G(X,Y)- BOUNDARY VALUES AND INITIAL APPROXIMATION
C  FOR THE SOLUTION U(X,Y) ARE GIVEN BY THE C FUNCTION G(X,Y)
C
C CORRECTION SCHEME BEGINS FROM THE FINEST GRID TO COARSEST GRID.
C
C
      implicit real*8 (a-h,o-z)
      EXTERNAL G,F,Z
      COMMON Q(18000)

      DIMENSION IST(200)
      DATA NX1/2/,NY1/2/,H1/0.5/,M/5/,NU1/2/,NU2/2/,NCYC/10/
c
c-----set method IFAS=1 nonlinear method, IFAS=0 linear method c
      IFAS=1
```

```
c
c-----set up: the grid
c
      DO 1 K=1,M
      K2=2**(K-1)
      CALL GRDFN(K,NX1*K2+1,NY1*K2+1,H1/K2)
      CALL GRDFN(K+M,NX1*K2+1,NY1*K2+1,H1/K2)
  1   CONTINUE
      WU=0.
c
c-----set up: the data (initial approx, rhs, bc, etc.) c
      CALL PUTF(M,G,0)
      CALL PUTB(M,G)
      CALL PUTF(2*M,F,2)

      ERRMX=1.0
      IREL=0
c
c-----start cycling
c
      DO 5 IC=1,NCYC
c-----store the previous relative error
      ERROLD=ERRMX
c-----go up
      DO 3 KM=1,M
      K=1+M-KM
c-----pre-smoothing, NU1 times
      DO 2 IR=1,NU1
    2 CALL RELAX(K,K+M,WU,M,ERRM)
c-----store the relative error
      IF(K.EQ.M) ERRMX=ERRM
c-----set initial zero approx. on the coarse grid
      IF (K.NE.M.AND.IFAS.EQ.0) CALL PUTZ(K)
c-----compute residual res=b-Au (and transfer it to k-1)

c      H   H   h   h  h
c     r = I  (f - L  u )
c          h

      IF(K.GT.1) CALL RESCAL(K,K+M,K+M-1)
c
```

```
c-----set initial approx. on the coarse grid
      IF (K.NE.1.AND.IFAS.EQ.1) CALL PUTU(K,K-1)
c-----compute the right-hand side

c      H   H   h   h  h     H  H h
c     f = I  (f - L  u ) + L (I u)
c          h                  h

      IF(K.GT.1.AND.IFAS.EQ.1) CALL CRSRES(K-1,K+M-1)
    3 CONTINUE
c
c-----go down
      DO 5 K=1,M
      DO 4 IR=1,NU2
    4 CALL RELAX(K,K+M,WU,M,ERRM)
c-----interpolate the coarse solution (error function)

c     to the next finer grid and add to the existing approximation

c

c      h   h    h  H   H h
c     u = u  + I (u - I u )
c

      IF(IFAS.EQ.1.AND.K.LT.M) CALL SUBTRT(K+1,K)
c
      IF(K.LT.M) CALL INTADD(K,K+1)
c
c-----compute the convergence factor using the relative c error

      IF(K.EQ.M) write(*,*) 'rho ', errmx/errold
   5  CONTINUE
C
C  PRINT THE SOLUTION (FINEST GRID) C
 999  CONTINUE
      OPEN(UNIT=17,FILE='TEST.DAT',STATUS='UNKNOWN')
      CALL KEY(M,IST,II,JJ,H)
      JSTEP=17
      IF(JJ.GT.9) JSTEP=INT(JJ/9.)+1
      DO 90 I=1,II
```

```
   90 WRITE(17,100) (Q(IST(I)+J),J=1,JJ,JSTEP)
c
c     calculate the L00 error c
      difmx=0.0
      do 95 i=1,ii
         x=(i-1)*h
         do 95 j=1,jj
            y=(j-1)*h
            err=abs(q(ist(i)+j)-g(x,y))
            difmx=max(difmx,err)
   95 continue
      write(*,*) 'l00 norm of the error =',difmx
  100 FORMAT(1X,257(1X,E8.2))
      STOP
      END
C
      REAL*8 FUNCTION F(X,Y)
      implicit real*8 (a-h,o-z)
      PI=4.0D0 * DATAN(1.0D0)
      PI2=PI*PI
      F=-(2.0*PI2)*SIN(PI*X)*SIN(PI*Y)
      RETURN
      END
C
      REAL*8 FUNCTION G(X,Y)
      implicit real*8 (a-h,o-z)
      PI=4.0D0 * DATAN(1.0D0)
      PI2=PI*PI
      G=SIN(PI*X)*SIN(PI*Y)
      RETURN
      END
C
      SUBROUTINE GRDFN(K,M,N,HH)
      implicit real*8 (a-h,o-z)
      COMMON/GRD/NST(20),IMX(20),JMX(20),H(20)
      DATA IQ/1/
      NST(K)=IQ
      IMX(K)=M
      JMX(K)=N
      H(K)=HH
      IQ=IQ+M*N
```

```
      RETURN
      END
C
      SUBROUTINE KEY(K,IST,M,N,HH)
      implicit real*8 (a-h,o-z)
      COMMON/GRD/NST(20),IMX(20),JMX(20),H(20)
      DIMENSION IST(200)
      M=IMX(K)
      N=JMX(K)
      IS=NST(K)-N-1
      DO 1 I=1,M
      IS=IS + N
    1 IST(I)=IS
      HH=H(K)
      RETURN
      END
C
      SUBROUTINE PUTF(K,F,NH)
      implicit real*8 (a-h,o-z)
      COMMON Q(18000)
      DIMENSION IST(200)
      CALL KEY (K,IST,II,JJ,H)
      H2=H**NH
      DO 1 I=1,II
       DO 1 J=1,JJ
         X=(I-1)*H
         Y=(J-1)*H
    1 Q(IST(I)+J)=F(X,Y)*H2
      RETURN
      END
C
      SUBROUTINE PUTZ(K)
      implicit real*8 (a-h,o-z)
      COMMON Q(18000)
      DIMENSION IST(200)
      CALL KEY(K,IST,II,JJ,H)
      DO 1 I=1,II
      DO 1 J=1,JJ
    1 Q(IST(I)+J)=0.
      RETURN
      END
```

```
C
      SUBROUTINE PUTU(KF,KC)
      implicit real*8 (a-h,o-z)
      COMMON Q(18000)
      DIMENSION IUF(200), IUC(200)
      CALL KEY(KF,IUF,IIF,JJF,HF)
      CALL KEY(KC,IUC,IIC,JJC,HC)
      DO 1 IC=1,IIC
      IF=2*IC-1
      IFO=IUF(IF)
      ICO=IUC(IC)
      JF=-1
      DO 1 JC=1,JJC
      JF=JF+2
      Q(ICO+JC)=Q(IFO+JF)
    1 CONTINUE
      RETURN
      END
C
      SUBROUTINE PUTB(K,F)
      implicit real*8 (a-h,o-z)
      COMMON Q(18000)
      DIMENSION IST(200)
      CALL KEY (K,IST,II,JJ,H)
      DO 1 I=1,II
      X=(I-1)*H
      Y=0.0
      Q(IST(I)+1)=F(X,Y)
      Y=(JJ-1)*H
      Q(IST(I)+JJ)=F(X,Y)
    1 CONTINUE
      DO 2 J=1,JJ
      Y=(J-1)*H
      X=0.0
      Q(IST(1)+J)=F(X,Y)
      X=(II-1)*H
      Q(IST(II)+J)=F(X,Y)
    2 CONTINUE
      RETURN
      END
C
```

```
      SUBROUTINE SUBTRT(KF,KC)
      implicit real*8 (a-h,o-z)
      COMMON Q(18000)
      DIMENSION IUF(200),IUC(200)
      CALL KEY(KF,IUF,IIF,JJF,HF)
      CALL KEY(KC,IUC,IIC,JJC,HC)
      DO 1 IC=1,IIC
      IF=2*IC-1
      IFO=IUF(IF)
      ICO=IUC(IC)
      JF=-1
      DO 1 JC=1,JJC
      JF=JF+2
      Q(ICO+JC)=Q(ICO+JC)-Q(IFO+JF)
    1 CONTINUE
      RETURN
      END
C
      SUBROUTINE INTADD(KC,KF)
      implicit real*8 (a-h,o-z)
      COMMON Q(18000)
      DIMENSION ISTC(200),ISTF(200)
      CALL KEY(KC,ISTC,IIC,JJC,HC)
      CALL KEY(KF,ISTF,IIF,JJF,HF)
      HF2=HF*HF
      DO 1 IC=2,IIC
      IF=2*IC-1
      JF=1
      IFO=ISTF(IF)
      IFM=ISTF(IF-1)
      ICO=ISTC(IC)
      ICM=ISTC(IC-1)
      DO 1 JC=2,JJC
      JF=JF+2
      A=.5*(Q(ICO+JC)+Q(ICO+JC-1))
      AM=.5*(Q(ICM+JC)+Q(ICM+JC-1))
      Q(IFO+JF) = Q(IFO+JF)+Q(ICO+JC)
      Q(IFM+JF) = Q(IFM+JF)+.5*(Q(ICO+JC)+Q(ICM+JC))
      Q(IFO+JF-1)=Q(IFO+JF-1)+A
    1 Q(IFM+JF-1)=Q(IFM+JF-1)+.5*(A+AM)
      RETURN
```

```
      END
C
      SUBROUTINE RESCAL(KF,KRF,KRC)
      implicit real*8 (a-h,o-z)
      COMMON Q(18000)
      DIMENSION IUF(200),IRF(200),IRC(200)
      CALL KEY(KF,IUF,IIF,JJF,HF)
      CALL KEY(KRF,IRF,IIF,JJF,HF)
      CALL KEY(KRC,IRC,IIC,JJC,HC)
      IIC1=IIC-1
      JJC1=JJC-1
      HF2=HF*HF
      DO 1 IC=2,IIC1
      ICR=IRC(IC)
      IF=2*IC-1
      JF=1
      IFR=IRF(IF)
      IF0=IUF(IF)
      IFM=IUF(IF-1)
      IFP=IUF(IF+1)
      DO 1 JC=2,JJC1
      JF=JF+2
      S=Q(IF0+JF+1)+Q(IF0+JF-1)+Q(IFM+JF)+Q(IFP+JF)
    1 Q(ICR+JC)=4.*(Q(IFR+JF)-S+4.*Q(IF0+JF))
      RETURN
      END
C
      SUBROUTINE CRSRES(K,KRHS)
      implicit real*8 (a-h,o-z)
      COMMON Q(18000)
      DIMENSION IST(200),IRHS(200)
      CALL KEY(K,IST,II,JJ,H)
      CALL KEY(KRHS,IRHS,II,JJ,H)
      I1=II-1
      J1=JJ-1
      H2=H*H
      DO 1 I=2,I1
      IR=IRHS(I)
      IO=IST(I)
      IM=IST(I-1)
      IP=IST(I+1)
```

```
      DO 1 J=2,J1
      A=-Q(IR+J)-Q(IO+J+1)-Q(IO+J-1)-Q(IM+J)-Q(IP+J)
    1 Q(IR+J)=-A-4.*Q(IO+J)
      RETURN
      END
C
      SUBROUTINE RELAX(K,KRHS,WU,M,ERRM)
c-----Gauss-Seidel
      implicit real*8 (a-h,o-z)
      COMMON Q(18000)
      DIMENSION IST(200),IRHS(200)
      CALL KEY(K,IST,II,JJ,H)
      CALL KEY(KRHS,IRHS,II,JJ,H)
      I1=II-1
      J1=JJ-1
      ERR=0.
      ERRQ=0.
      ERRM=0.
      H2=H*H
      COEFF=4.

      DO 1 I=2,I1
      IR=IRHS(I)
      IQ=IST(I)
      IM=IST(I-1)
      IP=IST(I+1)
      DO 1 J=2,J1
      A=Q(IR+J)-Q(IQ+J+1)-Q(IQ+J-1)-Q(IM+J)-Q(IP+J)
c-----residual norm L2
      ERR=ERR+(A+COEFF*Q(IQ+J))**2
      QOLD=Q(IQ+J)
      Q(IQ+J)=-A/(COEFF)
      ERRQ=ERRQ+(QOLD-Q(IQ+J))**2
c-----relative 'dynamic' error norm max
      Z=abs(QOLD-Q(IQ+J))
      ERRM=MAX(ERRM,Z)
    1 CONTINUE
      ERR=SQRT(ERR)/H
      ERRQ=SQRT(ERRQ)
      WU=WU+4.**(K-M)
      write(*,2) K,ERRQ,WU
```

```
    2 FORMAT('  LEVEL',I2,' RESIDUAL NORM=',E10.3,'   WORK=',F7.3)
      RETURN
      END
C
```

References

[1] N.S. Bakhvalov, *On the convergence of a relaxation method with natural constraints on the elliptic operator*. *USSR Computational Math. and Math. Phys.* **6** (1966) 101.

[2] D.S. Balsara and A. Brandt, *Multilevel Methods for Fast Solution of N-Body and Hybrid Systems*. In: Hackbusch-Trottenberg [25].

[3] A. Brandt, *Multi-Level Adaptive Technique (MLAT) for Fast Numerical Solution to Boundary-Value Problems*. In: H. Cabannes and R. Temam (eds.), Proceedings of the Third International Conference on Numerical Methods in Fluid Mechanics, Paris 1972. *Lecture Notes in Physics* **18**, Springer-Verlag, Berlin, 1973.

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

[5] A. Brandt and N. Dinar, *Multigrid Solutions to Elliptic Flow Problems*. In: S.V. Parter (ed.), *Numerical Methods for Partial Differential Equations*, Academic Press, New York, 1979.

[6] A. Brandt, S. McCormick and J. Ruge, *Multigrid Methods for Differential Eigenproblems*. *SIAM J. Sci. Stat. Comput.* **4** (1983) 244-260.

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

[8] A. Brandt, *Rigorous Local Mode Analysis of Multigrid*. Lecture at the 2nd European Conference on Multigrid Methods, Cologne, Oct. 1985. Research Report, The Weizmann Institute of Science, Israel, Dec. 1987.

[9] A. Brandt and A.A. Lubrecht, *Multilevel Matrix Multiplication and Fast Solution of Integral Equations*. *J. Comp. Phys.* **90** (1990) 348-370.

[10] A. Brandt and J. Greenwald, *Parabolic Multigrid Revisited*. In: Hackbusch-Trottenberg [25].

[11] A. Brandt, *Multilevel computations of integral transforms and particle interactions with oscillatory kernels*. *Comp. Phys. Commun.* **65** (1991) 24-38.

[12] A. Brandt, D. Ron and D.J. Amit, *Multi-Level Approaches to Discrete-State and Stochastic Problems*. In: Hackbusch-Trottenberg [24].

[13] A. Brandt and D. Sidilkover , *Multigrid solution to steady-state two-dimensional conservation laws*, SIAM Journal on Numerical Analysis, 30 (1993), p.249-274.

[14] A. Brandt, *Multigrid Methods in Lattice Field Computations*. *Nucl. Phys.* (Proc.Suppl.) **B21** (1992) 1-45.

[15] R.P. Fedorenko, *A relaxation method for solving elliptic difference equations*. *USSR Computational Math. and Math. Phys.* **1** (1962) 1092.

[16] R.P. Fedorenko, *The rate of convergence of an iterative process*. *USSR Computational Math. and Math. Phys.* **4** (1964) 227.

[17] J. Goodman and A.D. Sokal, *Multigrid Monte Carlo for lattice field theories*. *Phys. Rev. Lett.* **56** (1986) 1015.

[18] W. Hackbusch, *A multi-grid method applied to a boundary problem with variable coefficients in a rectangle*. Report 77-17, Institut für Angewandte Mathematik, Universität Köln, 1977.

[19] W. Hackbusch, *On the computation of approximate eigenvalues and eigenfunctions of elliptic operators by means of a multi-grid method*. *SIAM J. Numer. Anal.* **16** (1979) 201-215.

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

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

[22] W. Hackbusch, *Multi-Grid Methods of the Second Kind*. In: D.J. Paddon and H. Holstein (eds.), *Multigrid Methods for Integral and Differential Equations*. Claredon Press, Oxford, 1985.

[23] W. Hackbusch and U. Trottenberg (eds.), *Multi-Grid Methods*, Proceedings, Köln-Porz, Nov. 1981, *Lecture Notes in Mathematics* **960**, Springer-Verlag, Berlin, 1982.

[24] W. Hackbusch and U. Trottenberg (eds.), *Multigrid Methods II*, II. Proc. of the European Conference on Multigrid Methods, Cologne, Oct. 1985, *Lecture Notes in Mathematics* **1228**, Springer-Verlag, Berlin, 1986.

[25] W. Hackbusch and U. Trottenberg (eds.), *Multigrid Methods III*, III. Proc. of the European Conference on Multigrid Methods, Bonn, Oct. 1990, Birkhäuser, Berlin, 1991.

[26] P.W. Hemker and H. Schippers, *Multiple grid methods for the solution of Fredholm integral equations of the second kind*. *Math. Comp. 36* (1981) 215-232.

[27] P.W. Hemker and P. Wesseling (eds.), *Multigrid Methods IV*, Proceedings of the IV European Multigrid Conference, Amsterdam, Jul. 1993, International Series on Numerical Mathematics, Vol. 116, Birkhauser Verlag, Basel, 1994.

[28] Van Emden Henson, Multigrid for Nonlinear Problems: an Overview, presented by Van Emden Henson at the SPIE 15th Annual Symposium on Electronic Imaging, Santa Clara, California, January 23, 2003.

[29] T. Kalkreuter, *Multigrid Methods for the Computation of Propagators in Gauge Fields*. DESY 92-158.

[30] L. Lapidus and G.E. Pinder, *Numerical Solution of Partial Differential Equations in Science and Engineering*. John Wiley & Sons, New York, 1982.

[31] G. Mack and S. Meyer, *The effective action from multigrid Monte Carlo. Nucl. Phys.* (Proc. Suppl.) **17** (1990) 293.

[32] J. Mandel et al., *Copper Mountain Conference on Multigrid Methods*. IV. Proc. of the Copper Mountain Conference on Multigrid Methods, Copper Mountain, April 1989, SIAM, Philadelphia, 1989.

[33] H.D. Mittelmann and H. Weber, *Multi-Grid Methods for Bifurcation Problems. SIAM J. Sci. Stat. Comput.* **6** (1985) 49.

[34] W.A. Mulder, *A new multigrid approach to convection problems*, Journal of Computational Physics, v.83 n.2, p.303-323, Aug. 1989

[35] K. Stüben, *Algebraic Multigrid (AMG): An Introduction with Applications*, GMD Report 53, March 1999.

[36] K. Stüben and U. Trottenberg, *Multigrid Methods: Fundamental Algorithms, Model Problem Analysis and Applications*. In: Hackbusch-Trottenberg [23].

[37] S. Ta'asan, *Multigrid Methods for Locating Singularities in Bifurcation Problems. SIAM J. Sci. Stat. Comput.* **11** (1990) 51-62.

[38] S. Ta'asan, *Introduction to shape design and control; Theoretical tools for problem setup; Infinite dimensional preconditioners for optimal design*. In: Inverse Design and Optimisation Methods, VKI LS 1997-05.

[39] C.-A. Thole and U. Trottenberg, *Basic smoothing procedures for the multigrid treatment of elliptic 3D-operators. Appl. Math. Comp.* **19** (1986) 333-345.

[40] U. Trottenberg, C. Oosterlee, and A. Schüller, *Multigrid*, Academic Press, London, 2001.

[41] S. Vandewalle and R. Piessens, *Efficient Parallel Algorithms for Solving Initial-Boundary Value and Time-Periodic Parabolic Partial Differential Equations. SIAM J. Sci. Stat. Comput.* **13** (1992) 1330-1346.

[42] P. Wesseling, *A survey of Fourier smoothing analysis results*. In: Hackbusch-Trottenberg [25].