

# **D 5    Multiscale and Multigrid Procedures**

Bernhard Steffen

Institut für Angewandte Mathematik

Forschungszentrum Jülich GmbH

## **Contents**

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Principles of multigrid</b>	<b>2</b>
2.1	The model problem: Poisson's equation in a rectangle . . . . .	2
2.2	Simple iterative solvers and smoothing . . . . .	3
2.3	Two grid procedure - simple restriction and interpolation . . . . .	4
2.4	A V-cycle full multigrid for the model problem . . . . .	6
2.5	More complicated problems . . . . .	7
<b>3</b>	<b>Treatment of multiscale problems</b>	<b>8</b>
3.1	Singularities and local grid refinement . . . . .	9
3.2	High frequency oscillations . . . . .	9
3.3	Irregular scale separation . . . . .	10
<b>4</b>	<b>Conclusion</b>	<b>10</b>

# 1 Introduction

Many problems of physics and engineering require the accurate solution of field equations for some complicated setup. The standard approach is to discretise the continuous equations and solve the resulting finite system of equations. If the equations are nonlinear (e.g. magnetic field equations in the presence of iron), they are linearised with an available approximation to the solution to give a linear equation (or a sequence of linear equations). Now to achieve high accuracy, the discretisation has to be fine, and  $10^6$  unknowns are not much for a 3D problem. Simple linear equation solvers are quite inefficient for such problem sizes. The number of arithmetic operations needed to solve such a problem (e.g.  $\Delta u = f$ ) with  $N$  unknowns to an accuracy compatible with the accuracy of the discretisation is about  $O(N^3)$  for full matrix direct elimination,  $O(N^{2.3})$  for band matrix direct elimination,  $O(N^2)$  for successive overrelaxation (SOR),  $O(N^{1.3})$  for a good preconditioned conjugate gradient (CG) method, and only  $O(N)$  for an optimal multigrid method (MG).

Other than CG or SOR, however, MG is not a linear equation solver, but a method for building efficient hierarchical solvers. The typical application for MG is in the numerical solution of elliptic partial differential equations in two or more dimensions, but MG is also applicable to more complicated, non-symmetric and nonlinear systems of equations like the (Navier-) Stokes equations and even to equations without a PDE background.

Many such problems have structures on a wide range of scales, so that an uniform resolution of the small scales is beyond reach. Localised small features can be treated by local grid refinement. As MG as a solver operates on many scales in parallel, it is especially suited for such problems. By necessity, only principles of MG and simple examples can be given here. For more details and tuning a MG solver to the problem at hand, we refer to the literature [2, 3, 1, 6, 9].

A different multiscale problem is posed by small scale oscillations or texture of materials. Here, two combined problems must be solved simultaneously: An averaged smooth problem, and a local oscillatory or textured problem. This situation allows a true separation of scales. The small scale calculation will need the large scale solution for boundary conditions, while the large scale calculation may need the small scale calculation for the definition of material coefficients.

## 2 Principles of multigrid

The basic requirement for MG to be applicable to a problem is that the problem can be solved on different scales, that coarser scales are much easier to treat than fine ones, and that there is an efficient way to connect solutions from different scales. All this is obviously true for linear elliptic PDE's, so we will explain MG by applying it to a specific PDE problem.

### 2.1 The model problem: Poisson's equation in a rectangle

As model problem P, we treat the following PDE:

$$Lu = u_{xx} + \epsilon \cdot u_{yy} = -f(x, y) \quad (x, y) \in \Omega = [0, \pi] \times [0, \pi] \quad (1)$$

$$u(x, y) = g(x, y) \quad (x, y) \in \delta\Omega \quad (2)$$

We discretise this problem using a Cartesian equidistant grid, setting

$$h = \pi/n, \quad x_i = i h, \quad i = 1, \dots, n, \quad y_j = j h, \quad j = 1, \dots, m \quad (3)$$

$$u_{ij} = u(x_i, y_j), \quad f_{ij} = f(x_i, y_j) \quad (4)$$

and replacing the differential operator in Eq. (1) by its discrete analogue  $L_h u = -f$ :

$$(L_h u)_{ij} = (u_{i-1,j} + u_{i+1,j})/h^2 + \epsilon \cdot (u_{i,j-1} + u_{i,j+1})/h^2 - (2/h^2 + 2\epsilon/h^2) u_{ij} = -f_{ij} \quad (5)$$

For the definitions of discrete operators on Cartesian grids, stencil notations are a convenient method. The stencil to the operator  $L_h$  is:

$$L_h = \frac{1}{h^2} \begin{bmatrix} 0 & \epsilon & 0 \\ 1 & -2(1+\epsilon) & 1 \\ 0 & \epsilon & 0 \end{bmatrix} \quad (6)$$

For a start, we will consider the isotropic problem ( $\epsilon = 1$ ) with  $n = 2^k$ , the anisotropic problem ( $\epsilon \ll 1$ ) will be discussed later.

The discretisation error always consists of two parts. The low frequency components will show an error which is roughly proportional to  $h^2$ , and components with a frequency higher than the grid can show are simply absent. With iterative solvers there is always a truncation error, too. This is the difference between the exact and the actually computed solution of the discrete equations. A reasonable requirement is that the truncation error should be less than half the discretisation error.

## 2.2 Simple iterative solvers and smoothing

With lexicographic ordering of points, the discrete problem is a linear equation  $Au = -f$  with a sparse regular system matrix  $A$  of bandwidth  $n$ . For such matrices, Jacobi relaxation is often used. It is the simplest (and easiest to analyse) solution method, though it is not very efficient. For notation, we define  $u$  as solution of the problem  $P_h : L_h u = -f$ ; with  $u^0$  as the start vector of the iteration,  $u^l$  the iterated vectors, and  $w^l = u^l - u$  the error, which is just the iteration vector of  $L_h w = 0$  with start  $w^0$ .

One step of the Jacobi relaxation is just one application of the relaxation operator

$$S_h = E_h - \frac{h^2}{4} L_h \quad E_h: \text{Identity matrix to grid } h \quad (7)$$

to the present error:  $w^{l+1} = S_h w^l$ . Now the eigenvectors of  $S_h$  (and  $L_h$ ) are  $\sin(kx_i) \sin(ly_j)$ , simply the sampled eigenfunctions of  $L$ , and the corresponding eigenvalues of  $S_h$  are

$$\lambda_{kl} = -(\cos(kh) + \cos(lh))/2 \quad k, l = 1, \dots, n-1 \quad (8)$$

Obviously, the asymptotic convergence is given by the convergence of the smoothest component,  $\sin(x_i) \sin(y_j)$ , and the least smooth component,  $\sin((n-1)x_i) \sin((n-1)y_j)$ , and the convergence rates are  $\cos(h)$  and  $-\cos(h)$ . For small  $h$  this is very slow convergence. The convergence of the least smooth components, which is most important in a multigrid context, may be enhanced by  $\omega$ -damping:  $S_{\omega,h} = (1-\omega)E_h + \omega S_h$ . This will for the least smooth component give a rate of  $(1-\omega) - \omega \cos(h)$ , which for  $\omega = 0.5$  is close to zero. The convergence of the smoothest component will get even worse by damping. Overrelaxation ( $\omega > 1$ ), on

the other hand, will speed the convergence of the smoothest component while the least smooth component will become divergent. Thus, if Jacobi relaxation is considered a solver,  $\omega = 1$  is optimal. If the aim is just to smooth out the error, as it will be in a multigrid context,  $\omega = 2/3$  is a good choice.

For other common relaxations, e.g. Gauss-Seidel relaxation or line relaxation, the eigenvectors of the iteration matrix allow no simple description, and all frequencies will be mixed in the process. However, it is observed that these relaxations smooth out the error much faster than reduce it, similar to damped Jacobi. However, overrelaxation is useful for these relaxations, indicating that the least smooth component is reduced much more rapidly than the smoothest one. A special case is the red/black (or chessboard) Gauss-Seidel (RBGS), where the 'red' half of points is updated first and the other half follows. One step of RBGS is on the 'black' points exactly equivalent to two steps of Jacobi relaxation, so for each Fourier component it has the squared convergence rate. However, as the initial values of the 'red' points do not contribute to the result at all, and the frequencies with  $k + l > n$  do not show up on the 'black' points due to aliasing, the smoothing is considerably better than for Jacobi relaxation. On the full grid, the staggered relaxation gives a good proportion of high frequency error as the 'red' points lag back by half an relaxation relative to the 'black' points, but this error is coupled to the errors of intermediate frequency and not propagated directly by the relaxation. Therefore, RBGS is an effective smoother and, for the problem given, the most effective of the simple solvers, especially when combined with overrelaxation.

These results carry over with minimal adjustment to the 3D case and to more complicated, but approximately isotropic problems. Problems with strong anisotropy will be considered later.

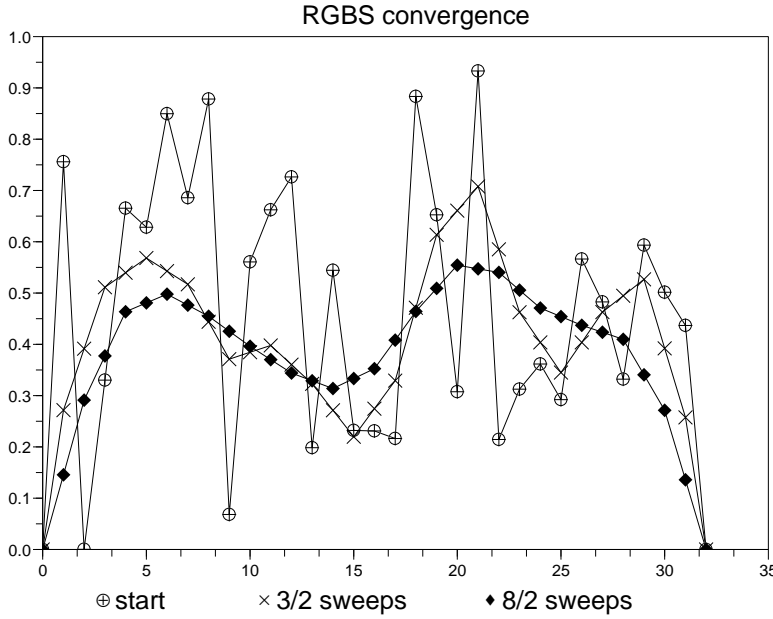
## 2.3 Two grid procedure - simple restriction and interpolation

Fig.1 shows the development of the solution to  $u''=0$  discretised with 33 points under RBGS relaxation with random start. Obviously, after 2 relaxations, the error can be represented quite well on a grid with double spacing. On this coarse grid, the relaxations are cheaper (by a factor of  $0.5^d$ , where  $d$  is the dimension of the problem), and the relaxation is going to converge faster, factor  $\cos(2h)$  instead of  $\cos(h)$ . After 7 relaxations, only the two lowest frequencies give a big contribution to the solution. The effect of RBGS in higher dimensions is similar, as can be seen by the eigenvalues of  $S_h$ . For the transfer of the problem to and from the coarse grid, two different approaches are possible. The coarse grid problem may either be a correction to the present fine grid approximation (correction scheme, CS), which is the preferred way for linear problems, or the full problem on the coarse grid augmented with some fine grid information, which is called 'full approximation scheme (FAS)'. FAS performs better for nonlinear systems, it is described in detail in [4].

May  $u_h^l$  be the present approximation to  $u$ . The correction problem on the fine grid reads: Solve  $L_h v_h = -f - L_h u_h^l$ . Then  $v_h + u_h^l$  is the exact solution of (5). This problem now has to be restricted to the coarse grid:

$$L_h v_h = -f - L_h u_h^l \implies L_{2h} v_{2h} = R_h^{2h}(-f - L_h u_h^l) \quad (9)$$

with  $L_{2h}$  the coarse discretisation of  $\Delta$ , and a restriction operator  $R_h^{2h}$  that transports the right hand side to the coarse grid. For the restriction, three different operators are common:



**Fig. 1:** RBGS at start, after 3/2 and after 8/2 relaxations for  $u''=0$ ; 33 points

Injection (IN), half weighting (HW), and full weighting (FW), with the 2D stencils

$$IN : \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad HW : \frac{1}{8} \begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad FW : \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \quad (10)$$

IN is the easiest, but lacks accuracy and robustness and is therefore possible only if both the error  $v_h^l$  and the residue  $L_h v_h^l - f$  are smooth. This would at best require additional smoothing steps, so IN is not very effective. HW is the restriction of choice for simple problems like our model problem, while for complicated problems, the added robustness of FW pays for the added effort. Other restrictions (e.g. anisotropic) are possible, but not common and usually give no advantage.

There is a special feature with RBGS: for the model problem,  $-f - L_h u_h^l$  is zero on the 'black' set of points. If the coarse grid point belong to 'red', HW is just half injection, and therefore very easy. IN gives a divergent coarse grid correction (CGS) in this case. For larger stencils of  $L_h$ ,  $-f - L_h u_h^l$  will not be zero on 'black', but quite small.

After the correction equation (9) is solved on the coarse grid,  $v_{2h}$  has to be interpolated to the fine grid ( $v_h \leftarrow I_{2h}^h(v_{2h})$ ) and added to  $u_h^l$ . Polynomial interpolation introduces additional high frequency errors, so a few smoothing steps afterwards are useful, producing  $u_h^k$ .

Let's look at the accuracy and cost of this procedure. The high frequency content of  $u_h^k$  is treated on the fine grid, and because of the rapid convergence of the iteration for high frequencies it will be quite accurate. The low frequency content is treated on the coarse grid, where it's discretisation error is about 4 times the discretisation error on the fine grid. The smoothing steps afterwards cannot improve the situation, the convergence of low frequency parts is too slow. However, with  $u_h^k$ , we can repeat the CGC process, only that this time the coarse grid correction is not most of the solution, but only the coarse grid discretisation error. It is enough to solve it up to a relative error of  $\approx 10\%$  to achieve a fine grid solution that is as accurate as

the fine grid discretisation error permits. The computational cost for all of this is the cost of the iterative coarse grid solution process plus a few fine grid relaxations plus interpolation and restriction, together about  $1/4$  (2D) of the cost of solving the problem by iteration on the fine grid only.

There are various options for the interpolation. Linear interpolation is simple, but causes a loss of accuracy of  $v_h$ . Second order interpolation gives asymmetric formulas, so cubic interpolation is generally preferred if linear interpolation is not sufficient. In the first correction step, where the function to be interpolated is a large part of the total result, a small relative interpolation error is needed to get all the accuracy available. In the second correction step, where the correction is small, the absolute interpolation error will be small enough even with linear interpolation.

## 2.4 A V-cycle full multigrid for the model problem

There is no reason why this process could not be applied to the coarser grid, also, as long as this grid has too many points for fast convergence. This idea leads to a proper MG scheme, where there is a hierarchy of grids, starting at a fairly coarse one and doubling the resolution till the finest grid gives the desired accuracy. In the model problem the coarsest grid may have just one inner point, but in most problems it will have some hundred points, but still allow a direct solver for solution. This needs some tuning, but when the solution needs only a few relaxations on all grids except the coarsest, the method really starts flying.

The second improvement is not starting from scratch. In every iteration, it pays to choose the best starting point that is readily available, and for a PDE this will, on all grids except the coarsest, most likely be the interpolated solution from the next coarser grid. The process is sketched in fig. 2. This is a simple variant of full multigrid (FMG), appropriate for the model problem. For an estimate of the work required, the accuracy achieved, and the parameters of

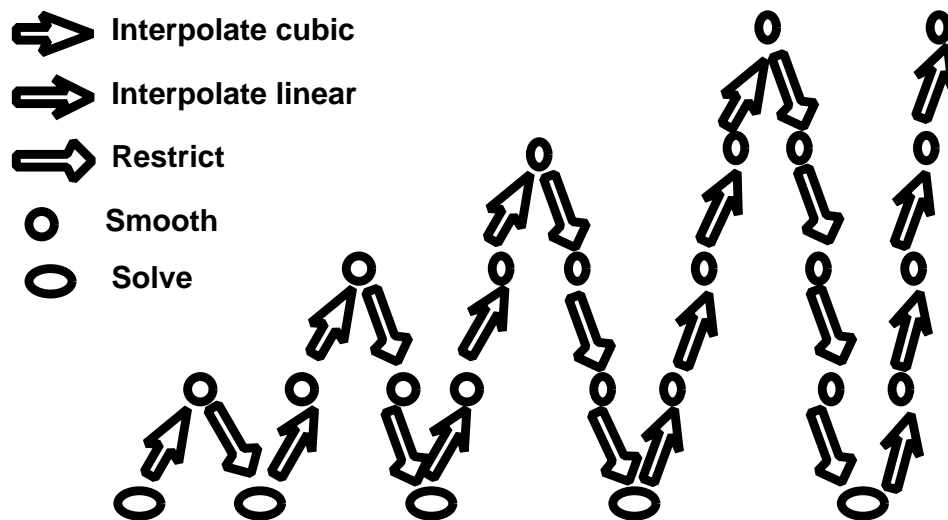


Fig. 2: FMG process using V-cycle

the procedure, let us assume that on all grids used the discretisation error is dominated by the low frequencies and below 10%. For convenience, let's number the grids from 1 (coarsest) to  $k$  (finest). Then on grid  $l$  the discretisation error is about a quarter of the discretisation error on grid  $l + 1$ . Assume, that the problem is solved on grid  $l$ . Interpolating this with cubic interpolation to grid  $l + 1$  gives an approximation  $u_{l+1}^0$  with an error of not more than 8 times

the discretisation error on grid  $l + 1$ , resulting from the sum of the discretisation error plus the truncation error on grid  $l$ , plus an allowance for a small error of the interpolation. Most of this will result from low frequency components which are not sufficiently accurate on grid  $l$ , a part from the high frequency components of the solution which are small, but not represented at all in  $u_{l+1}^0$ . So, on grid  $l + 1$ , the high frequency errors have to be smoothed by a factor of  $\approx 6$ , which asks for three relaxation steps, while a CGC reduces low frequency errors by a factor of  $\approx 8$ . This CGC requires only low accuracy, therefore three smoothing steps before restriction and after linear interpolation are sufficient to keep the absolute truncation error of the correction well below the absolute discretisation error on grid  $l + 1$ .

Adding up the computation, we find that on each of the grids  $k - m + 1$ ,  $m = 1 \dots k$ , we need  $6m$  smoothing steps,  $m$  restrictions,  $m$  linear interpolations, and 1 cubic interpolation. Additionally, we need  $k$  solutions on grid 1. Counting restrictions and interpolations like smoothing, and considering that the work for smoothing etc. is reduced by a factor of 4 per coarsening step, and defining  $SW$  as the work of one smoothing step on grid  $k$ , we get

$$TotalWork = SW \times \sum_{m=1}^k \frac{32m}{4^m} + k \times [Solution] \quad (11)$$

This is less than  $15 SW$  in 2D,  $11 SW$  in 3D. Optimized implementations [5] even need a little less work. Definitely, solving a linear equation will need more work than just touching every matrix element once, which would be  $SW$ , so MG is close to the lower limit.

## 2.5 More complicated problems

While the problem P is well suited to explain the principles of multigrid, most practical problems are much more complicated. Higher dimension is no complication, the higher the dimension, the more efficient is MG.

**Complicated domains** require some attention. While the eigenvalue analysis presented above is obviously not correct any more, on points away from the boundaries we can do a local Fourier analysis to find the smoothing properties of a relaxation method and the effects of grid transfers [1]. The results are the same as above, but they are not valid near the boundary. To get the required smoothing near the boundary, too, a small number of extra relaxations of boundary points is usually sufficient, but most implementations just add global relaxations. The coarse grids may give problems, however. With finite differences, a grid coarse enough to allow an efficient solution of the equation may not be able to represent the geometry accurately enough. With the more flexible finite elements, the problem is that there is no good method for the construction of a coarse grid when starting with a fine one. Either the discretisation has to be done by recursive refinement of the coarsest mesh, or coarse and fine mesh will not be properly aligned, and restriction and interpolation operators will become complicated and inefficient. This is the most serious obstacle to the combination of existing FEM codes with multigrid.

**Anisotropy** of the discrete operator, whether caused by an anisotropic problem or an anisotropic grid, requires special measures. With a stencil like eq. (6), the eigenvalues of  $S_h$  change to  $\lambda_{kl} = -(\cos(kh) + \epsilon \cdot \cos(lh)) / (1 + \epsilon)$ , and with small  $\epsilon$  the smoothing with simple relaxation will be efficient only in  $x$  direction. This means that either the smoother has to be modified - line relaxation or incomplete LU decomposition would help in this (simple) case - or the coarse grid has to contain all modes that are not smoothed out on the fine grid - e.g. coarsening in  $x$  direction only. Now in most cases the anisotropy is not aligned with the grid and may vary

in strength and direction. This asks for rather robust smoothing procedures, e.g. alternating direction line (in 2d) or plane (in 3D) [11] or transforming smoothers [10].

**Nonlinear problems** In nonlinear problems, the linear residual equation (9) does not apply. Instead, we can use the nonlinear residual equation  $L(u + v) - L(u) = r$  and transfer this to the coarse grid as

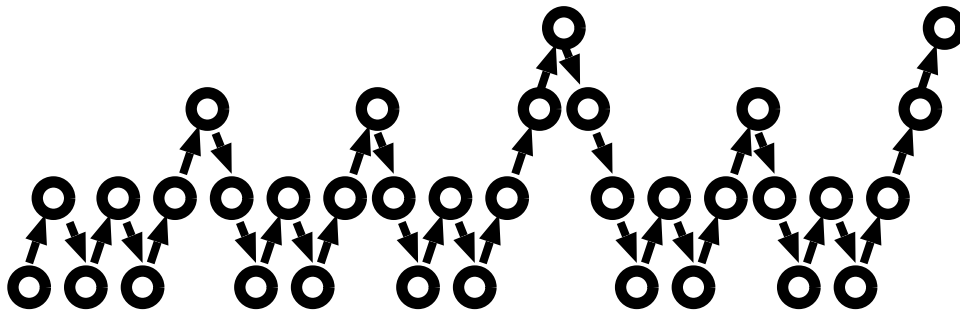
$$L_h(u_h + v_h) - L(u_h) = -f - L_h u_h^l \implies L_{2h}(u_{2h} + v_{2h}) = R_h^{2h}(-f - L_h u_h^l) \quad (12)$$

We solve for  $w_{2h} = u_{2h} + v_{2h}$  and update the fine grid approximation with the change only:

$$u_h \longleftarrow u_h + I_{2h}^h(w_{2h} - R_h^{2h} u_h) \quad (13)$$

**Multiphysics**, e.g. the modelling of the geodynamo, where thermal convection and conduction coupled with Maxwell's equation shape the earth's magnetic field, is an interesting field of application for multigrid. To utilize the full potential of MG, the coupling has to be incorporated on all grids, not just on the finest. This way, the iterations of the coupling needed for self-consistency of the equations are done on coarse grids, and the workload is not much more than just solving the equations for the different quantities independently. However, the different physical quantities may exhibit quite different features of the equations, and the optimal choice of MG components may differ e.g. between the equations for convective flow, temperature and magnetic induction.

**Caveat** For a perfectly designed and tuned multigrid, the total work should be not too large a multiple of the work for one relaxation sweep. For many practical problems perfection is out of reach, and we have to settle for just reasonable speed. This usually means more smoothing steps and frequently a more complicated cycle. The W-cycle [3], performing two coarse grid corrections before moving on to the next finer grid, is an example. It is only moderately more



**Fig. 3:** Full FMG process using W-cycle

programming effort to get an adaptive control of the MG process, doing smoothings as long as they are efficient, correction steps until the desired level of truncation is achieved, etc., and this effort is generally rewarded. One big benefit of monitoring the performance of the steps of MG is the boost it gives to debugging: Small errors in the algorithm design or programming are often hidden by the robustness of the entire MG process. They slow the convergence down, but do not change the results. Only when a monitoring of all steps is done do they reveal themselves.

### 3 Treatment of multiscale problems

Many practical problems operate on vastly different spatial or temporal scales, such that a resolution of the fine scales for the entire problem is completely out of reach. The character of these



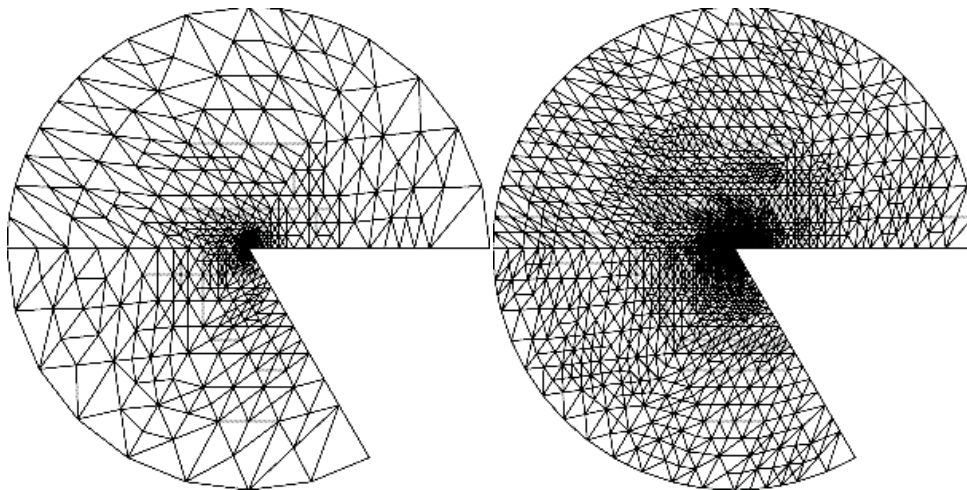
problems, as well as the ways to treat them, differ much for individual applications, so we can only look into a few types of multiscale behaviour and methods to treat them. The multigrid approach is obviously helpful here, as it operates naturally on scales differing by two or three magnitudes, but that is simply not enough for many problems. We will give some examples showing different kind of multiscale behaviour.

### 3.1 Singularities and local grid refinement

For a 2D flow around a corner, the analytical solution of potential is known:  $\phi = r^\alpha \cos(\alpha\theta)$ . For a 60 wedge,  $\alpha = 3/5$ . Thus the streamlines are circles around the origin. The velocity field is given by

$$(u, v) = [\alpha * r^{\alpha-1} \cos((\alpha - 1)\theta), -\alpha * r^{\alpha-1} \sin((\alpha - 1)\theta)] \quad (14)$$

Note that the velocities go as  $r^{\alpha-1}$ , and at the origin are infinite. Numerical calculations of the situation need a very fine grid near the origin. This can be achieved by local grid refinement. Of



**Fig. 4:** 9- and 13-level refined grids for flow around a corner [12]

course, the 2D problem is of little interest, but 3D flow around with wedges will appear e.g. in the calculation of air ducts for buildings. Even though potential flow is not a proper assumption here, it is a good guide for meshing.

At first look, it seems that the computational effort for the calculations with MG is quite small, as the number of points in the finest grid is moderate. However, the work done on intermediate grids is not smaller than on the finest, so the estimate of (11) is not valid any more. The actual effort depends on the number of points in all grids, and writing an optimal procedure with CGS for selected parts of the domain only is tricky.

### 3.2 High frequency oscillations

Many applications have spatial or temporal high frequency variations, such that there is a large gap in scales. One example is the magnetic field in a transformer, where one characteristic length scale is given by the dimensions of the iron (1 cm for a small device power supply, 1 m for a power station transformer) and the other by the lamination of the iron (0.5 mm). Another example is the calculation of the magnetisation of an magnetic resonance (MR) system [13] via

Bloch' equation

$$\frac{d\vec{M}}{dt} = \gamma(\vec{M} \times \vec{B}) - A_1(\vec{M} - \vec{M}_0) - A_2(\vec{M}) \quad (15)$$

$\vec{B}$  is composed of a strong ( $\approx 4T$ ) solenoidal field  $\vec{B}_0$ , inhomogeneities, a gradient field of shape  $\vec{B}_0(\vec{G} \cdot \vec{r})$ , and a radio frequency (RF) field approximately vertical to  $\vec{B}_0$ . All these are several orders of magnitude weaker than  $\vec{B}_0$ .  $A_1$  and  $A_2$  are tensors giving the spin-spin and the spin-lattice relaxations. Here three frequencies are present: the Lamour frequency  $\gamma \cdot |\vec{B}|$  which is in the order of 10 MHz, the frequency of the RF part of  $\vec{B}$  which is about 1000 Hz, and the relaxation frequencies close to 1 Hz.

In both cases, averaging over the high frequency is the method of choice. For the MR magnetisation, after going into cylindrical coordinates with z-axis parallel to  $\vec{M}_0$  we can average all effects over one period of the rotation of  $\vec{M}$ . This way, all phase information on  $\vec{M}$  is lost, but the phase information is not used and could by no means be calculated in MR, as  $\vec{B}_0$  is certainly not known to the accuracy necessary for this.  $M_z$  and  $M_r$  vary not faster than the RF timescales, thus allowing an efficient calculation.

For a transformer with lamination of 0.35mm iron and 0.15mm insulation, this means replacing the permeability  $\mu_{Fe}$  of iron and  $\mu_I = 1$  of insulation by an averaged permeability  $\mu_{av}^{\parallel} = 2(0.35\mu_{Fe} + 0.15 \cdot \mu_I)$  and  $\mu_{av}^{\perp} = 0.5 / (0.35/\mu_{Fe} + 0.15/\mu_I)$  in the direction parallel to the lamination and in the perpendicular direction. For high  $H$  applications, from the permeability curve  $\mu_{Fe}(H)$  there has to be calculated a curve  $\mu_{Fe}(H_{av})$  taking into account that  $H$  varies between iron and insulation. Of course, this data is available for all laminated iron in use. In other problems, such a separation of scales often leads to two coupled problems, where the small scale problem provides parameters for the large scale problem, which in turn defines the boundary conditions for the small scale problem.

### 3.3 Irregular scale separation

Frequently, the small scales have no apparent periodicity. There may be statistical information that allows averaging, but in general, the averaging itself is a formidable problem. A well investigated such multiscale problem is the calculation of the flow of air in weather forecast. Here again, the small scale process is parameterized and these parameters fed into the large scale calculation. E.g. kinetic energy is fed into the system at a scale of kilometres, through thermal convection. It is taken out of the system only by viscous forces, which operate at a scale of centimeters. Turbulence models [14, 15] are used to link the large scale motion to the energy dissipation.

In most multiscale processes with irregular small scales, the definition of the coupling of the large scale and the small scale model requires in depth analysis of the situation, and does not allow black box solutions.

## 4 Conclusion

Multigrid methods have the potential of being effective tools for the solution of multiscale and multiphysics problems, but they are not ready made solvers. Of course, there exist MG solutions for many practical problems (fluid mechanics, electrodynamics, mechanics etc.), but for others the full potential of MG has not been realised yet, and further research is needed.

## References

- [1] A. Brandt *Multilevel adaptive solutions to boundary value problems* Math. Comp. **31** 333-390 (1977)
- [2] U. Trottenberg, C. W. Oosterlee, A. Schüller, *Multigrid* Academic Press, San Diego, (2001).
- [3] W. Hackbusch, *Iterative Solution of Large Sparse Systems of Equations* Springer-Verlag, Berlin (1994)
- [4] A. Brandt *Guide to multigrid development* in W. Hackbusch, U. Trottenberg (Eds.), *Multigrid methods* (Springer Verlag, Berlin) 220-312, (1982)
- [5] H. Foerster, K. Witsch, *Multigrid software for the solution of elliptic problems on rectangular domains: MG00 (release 1)* *ibid.*, 427-460, (1982)
- [6] S. F. McCormick *Multigrid Methods* SIAM, Philadelphia, (1987)
- [7] W. L. Briggs, *A Multigrid Tutorial* SIAM, Philadelphia (1987),
- [8] <http://www.llnl.gov/casc/people/henson/mgtut/welcome.html>
- [9] C. G. Douglas <http://www.mgnet.org/>
- [10] G. Wittum *Multi-grid methods for Stokes and Navier-Stokes equations with transforming smoothers: algorithms and numerical results* Num. Math. **54** 543-563 (1989)
- [11] C. Thole, U. Trottenberg *Basic smoothing procedures for the multigrid treatment of elliptic 3D operators* Appl. Math. Comp. **19** 333-345 (1986)
- [12] Xing Cai, Klas Samuelsson <http://heim.ifi.uio.no/~xingca/pmg/>
- [13] B. Hargreaves <http://www-mrsl.stanford.edu/~brian/bloch/>
- [14] G. Lenderink and E. van Meijgaard *Impacts of cloud and turbulence schemes on integrated water vapor: comparison between model predictions and GPS measurements* Meteorology and Atmospheric Physics **77** 131-144 (2001)
- [15] U. Frisch, *Turbulence: the Legacy of A.N. Kolmogorov*. Cambridge University Press, Cambridge, 1995.