Chapter 14: Multigrid and Advection

Paul M. de Zeeuw
CWI
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

14.1 Introduction

In the previous chapter, Chapter 13, the virtues of multigrid methods were demonstrated through an elaborate example. That introduction enables us to regard a multigrid method that comes much closer to solving real-life problems. We consider a general linear 2nd order elliptic PDE in two dimensions

$$Lu \equiv -\nabla \cdot (D\nabla u) + b_1(x,y)\frac{\partial u}{\partial x} + b_2(x,y)\frac{\partial u}{\partial y} + c(x,y)u = f(x,y), \tag{14.1}$$

on a bounded domain $\Omega \subset \mathbb{R}^2$ with suitable boundary conditions. $D(x,y)$ is a positive definite 2×2 matrix function and $c(x,y) \geq 0$. We suppose that Ω is a rectangular domain. For the general concept of multigrid methods we refer to Chapter 13 and the extensive literature on this subject (for lots of exquisite theory see [6], for a more practical viewpoint see [14]). For the multigrid method to be discussed we consider a set of increasingly coarser grids (vertex-centered):

$$\Omega_l \supset \Omega_{l-1} \supset \ldots \Omega_k \supset \ldots \supset \Omega_0.$$

The grids are described as follows:

$$\Omega_k \equiv \{(x_i, y_i) \mid x_i = o_1 + (i-1)h_k, y_i = o_2 + (j-1)h_k\} \tag{14.2}$$

where (o_1, o_2) is the origin and $h_{k-1} = 2h_k$. The discretization on the finest grid Ω_l evokes the linear system

$$L_l u_l = f_l. \tag{14.3}$$

Neither uniformity nor rectangularity of the grids is essential to our approach, problem (14.1) may be discretized on a curvilinear grid.

Firstly, in section 14.2, we consider the coarse grid correction. It is explained why, with a standard choice for the prolongation and restriction a multigrid algorithm may fail to converge. We go on to show a remedy. Secondly, in section 14.3, we consider the other major part of the multigrid algorithm: the smoother. In this chapter we confine ourselves to structured grids and for that the Incomplete Line LU appears to be most appropriate as smoother. Thirdly, in section 14.4 we show numerical results for problem 4 (the Molenkamp problem) defined in Chapter 1 of this book. A multigrid solver is used, of which the major parts are described in the foregoing sections. This solver has proven to be able to solve many 9-point discretized advection-diffusion problems [16]. The Molenkamp problem, as it is specified, can be considered as the (difficult) limit-case of an advection-diffusion problem for which the diffusion vanishes. In section 14.4 our main concern is the efficiency of the iterative solver and not so much the accuracy of differencing schemes. In section 14.5 conclusions are summarized.

14.2 The Galerkin approximation

In this section we study one of the main parts of a multigrid algorithm: the coarse grid correction. It is demonstrated that the standard approach is not satisfactory; a full analysis of the occurring difficulty is given, followed by a remedy which consists of choosing the prolongation and restriction in an operator-dependent way.

Consider the coarse grid correction (CGC) within the multigrid correction scheme:

$$\begin{aligned} d_{k-1} &= R_{k-1}(f_k - L_k u_k); && \text{(14.4a)} \\ \text{solve } L_{k-1} e_{k-1} &= d_{k-1}; && \text{(14.4b)} \\ \tilde{u}_k &= u_k + P_k e_{k-1}. && \text{(14.4c)} \end{aligned}$$

R_{k-1} is the restriction operator that transfers the residual from the fine grid Ω_k onto the coarse grid Ω_{k-1}, P_k is the operator that transfers a correction for the solution from the coarse to the fine grid. Firstly we raise the question how to make an appropriate choice for L_{k-1}. One obvious and straightforward way is by discretization of the operator L on Ω_{k-1}. A disadvantage of this approach is that it may fail if L has (rapidly) varying coefficients and Ω_{k-1} is a very coarse grid, because then the sampling of the coefficients becomes faulty. Another approach is suggested by the diagram in Figure 14.1 ($S(\Omega_k)$ denotes the space of grid-functions defined on Ω_k).

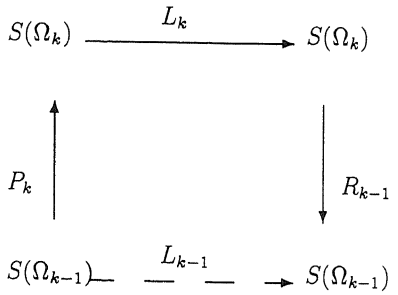

Figure 14.1: Diagram of Galerkin approximation

The operator L_{k-1} (corresponding to the dashed arrow) is defined by the sequence of operations

$$L_{k-1} = R_{k-1} L_k P_k. \tag{14.5}$$

Definition (14.5) is called the Galerkin coarse grid approximation [14]. Definition (14.5) is an essential ingredient for a black-box algorithm. By the latter we mean an algorithm with the linear system (14.3) as input, the solution as output and without parameters which have to be tuned. No interference of the user with the algorithm is required. In the context of multigrid methods this means that the user only needs to provide the system on the finest grid. The corresponding systems on the coarser grids are derived by an explicit construction according to (14.5). An advantage of (14.5) is that after the coarse grid correction the restriction of the residual vanishes

$$R_{k-1}(f_k - L_k \tilde{u}_k) = 0_{k-1}. \tag{14.6}$$

This means that at each coarse grid point a weighted average (with non-negative weights) of the fine-grid residual is zero, which implies that the residual consists of short wavelength components only. Such components can be reduced efficiently by a subsequent smoothing step.

We choose the restriction to be the transpose of the prolongation

$$R_{k-1} = P_k^T. \tag{14.7}$$

Hence, once P_k has been chosen, R_{k-1} and L_{k-1} follow automatically and the coarse grid correction is determined. Note that by (14.7) the possible (anti)symmetry of L_k is maintained on the coarser grid. Further, when L_k is a conservative discretization of L and P_k interpolates a constant function exactly, then the Galerkin approximation L_{k-1} is conservative just as well (see [16]).

So finally we end up by having to make a choice for the prolongation. Under the assumption of (14.7), the prolongation must satisfy an accuracy condition in order to obtain mesh-size independent rate of MG-convergence (see [2, 6, 7, 14])

$$2m_P > 2m, \tag{14.8}$$

where $2m$ is the order of the PDE, and m_P is the highest order plus one, of polynomials that are interpolated exactly by the prolongation. A standard choice for the prolongation is bilinear interpolation which satisfies the accuracy condition. This interpolation amounts to taking an equal average of solution-values at neighbouring coarse-grid points (see Figure 14.2).

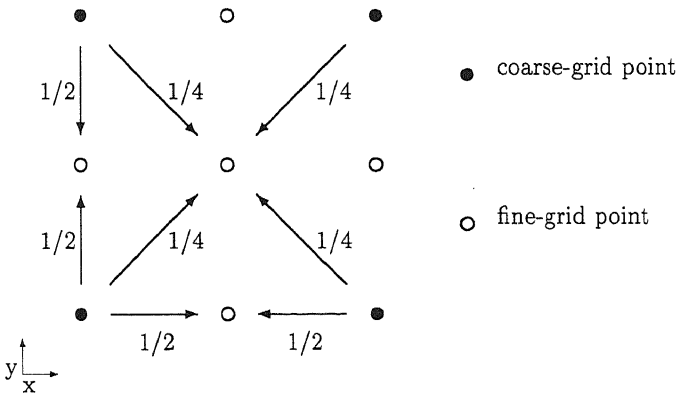

Figure 14.2: Bilinear prolongation

At the grid-points of the fine grid that coincide with the coarse grid we take identical values. The bilinear prolongation can also be denoted by the stencil

$$P_k \sim \begin{bmatrix} \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{2} & 1 & \frac{1}{2} \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \end{bmatrix}. \tag{14.9}$$

This stencil shows the non-zero values of the fine-grid function generated by the prolongation of a coarse-grid function which equals 1 at one point and 0 elsewhere. Because

of (14.7), the same stencil also represents the chosen restriction operator. Because of the foregoing remarks it looks as if an accordingly constructed MG-algorithm, furnished with an effective smoother, should do the job. Indeed, this is the case for a considerable class of problems. Yet, when a dominating advection term is present, divergence may occur. This is the topic of the next section.

14.2.1 Example of Galerkin approximation for an advection dominated problem

Consider the following linear operator:

$$Lu \equiv -\epsilon \Delta u + \frac{\partial u}{\partial x}. \tag{14.10}$$

For vanishing diffusion the following stencil is the result from some simple upwind discretization on a grid with mesh-size equal to 1:

$$L_l \sim \begin{bmatrix} 0 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{14.11}$$

By repeatedly applying (14.5) for the standard choice (14.7,14.9) we observe on the n times coarsened grid Ω_{l-n} the stencil:

$$L_{l-n} \sim \begin{bmatrix} -\frac{1}{12} & \frac{1}{6} & -\frac{1}{12} \\ -\frac{1}{3} & \frac{2}{3} & -\frac{1}{3} \\ -\frac{1}{12} & \frac{1}{6} & -\frac{1}{12} \end{bmatrix} + 2^n \begin{bmatrix} -\frac{1}{12} & 0 & +\frac{1}{12} \\ -\frac{1}{3} & 0 & +\frac{1}{3} \\ -\frac{1}{12} & 0 & +\frac{1}{12} \end{bmatrix} + \cdots \tag{14.12}$$

with a remainder that is rapidly decreasing with n. (This observation is verified in the next section.) The two stencils are the same ones as evoked by discretization with bilinear finite elements of a diffusion and advection term in the x-direction. Apparently the advection-stencil increases exponentially with n. Each time the grid is coarsened by the factor 2, the mesh Péclet numbers are multiplied by the same factor, which is reflected by the Galerkin coarse grid approximation. For increasing n the central advection-stencil will dominate and spurious high-frequent solutions will be created on the coarse grid and subsequently transferred to the fine grid. For small n it may be a feasible approach to require that the relaxation method on the fine grids is such that those components are sufficiently smoothed. However, when a really fine grid is employed, we need a substantial number of grid-levels to make use of the coarsest grid, where the components with the lowest frequency are reduced. In this case n can be so large that the spurious solutions created on the coarse grids affect severely the convergence of the multigrid algorithm as a whole. The equality (14.6) may still hold after the CGC, but we have to recognize that the amplitude of the short wavelength components comes to a very large magnitude. A first experimental result of divergence for an advection-diffusion problem, even when a robust smoother like ILU is in use, can be found in [17] (together with a Fourier local mode analysis).

14.2.2 Analysis of the Galerkin approach for constant coefficients

In this section we give an analysis of the behaviour of the Galerkin coarse grid approximation for the advection-diffusion equation. We confine ourselves to the case of constant

coefficients. The matrix L_k is represented by a single nine-point stencil only. With the choices (14.5),(14.7) and (14.9) we obtain a coarse grid matrix L_{k-1} which is also represented by a nine-point stencil. Because of the constant coefficients, the construction (14.5) can be seen as the linear transformation

$$G \begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \\ f_8 \\ f_9 \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \\ c_8 \\ c_9 \end{pmatrix}, \tag{14.13}$$

with G a 9×9-matrix. The vectors correspond to the stencils

$$f^* \sim \begin{bmatrix} f_7 & f_8 & f_9 \\ f_4 & f_5 & f_6 \\ f_1 & f_2 & f_3 \end{bmatrix}, \quad c^* \sim \begin{bmatrix} c_7 & c_8 & c_9 \\ c_4 & c_5 & c_6 \\ c_1 & c_2 & c_3 \end{bmatrix}. \tag{14.14}$$

The stencil f^* is defined on the fine grid, the stencil c^* is defined on the coarse grid. The matrix G describes what becomes of a stencil on the fine grid under the Galerkin coarse grid approximation. An eigenvalue decomposition of G exists and reads:

$$G = VDV^{-1}, G, V, D \in \mathbb{R}^9 \times \mathbb{R}^9, \tag{14.15}$$

where D is a diagonal matrix showing the eigenvalues of G and

$$V = \begin{pmatrix} -\frac{1}{6} & -\frac{1}{6} & -\frac{1}{4} & -\frac{1}{12} & -\frac{1}{12} & \frac{1}{36} & 1 & 1 & 1 \\ \frac{1}{3} & -\frac{2}{3} & 0 & 0 & -\frac{1}{3} & \frac{1}{9} & -2 & 0 & -2 \\ -\frac{1}{6} & -\frac{1}{6} & \frac{1}{4} & \frac{1}{12} & -\frac{1}{12} & \frac{1}{36} & 1 & -1 & 1 \\ -\frac{2}{3} & \frac{1}{3} & 0 & -\frac{1}{3} & 0 & \frac{1}{9} & 0 & -2 & -2 \\ \frac{4}{3} & \frac{4}{3} & 0 & 0 & 0 & \frac{4}{9} & 0 & 0 & 4 \\ -\frac{2}{3} & \frac{1}{3} & 0 & \frac{1}{3} & 0 & \frac{1}{9} & 0 & 2 & -2 \\ -\frac{1}{6} & -\frac{1}{6} & \frac{1}{4} & -\frac{1}{12} & \frac{1}{12} & \frac{1}{36} & -1 & 1 & 1 \\ \frac{1}{3} & -\frac{2}{3} & 0 & 0 & \frac{1}{3} & \frac{1}{9} & 2 & 0 & -2 \\ -\frac{1}{6} & -\frac{1}{6} & -\frac{1}{4} & \frac{1}{12} & \frac{1}{12} & \frac{1}{36} & -1 & -1 & 1 \end{pmatrix}, \tag{14.16a}$$

$$V^{-1} = \begin{pmatrix} -\frac{1}{3} & \frac{1}{6} & -\frac{1}{3} & -\frac{1}{3} & \frac{1}{6} & -\frac{1}{3} & -\frac{1}{3} & \frac{1}{6} & -\frac{1}{3} \\ -\frac{1}{3} & -\frac{1}{3} & -\frac{1}{3} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & -\frac{1}{3} & -\frac{1}{3} & -\frac{1}{3} \\ -1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & -1 \\ -1 & 0 & 1 & -1 & 0 & 1 & -1 & 0 & 1 \\ -1 & -1 & -1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \frac{1}{6} & -\frac{1}{12} & \frac{1}{6} & 0 & 0 & 0 & -\frac{1}{6} & \frac{1}{12} & -\frac{1}{6} \\ \frac{1}{6} & 0 & -\frac{1}{6} & -\frac{1}{12} & 0 & \frac{1}{12} & \frac{1}{6} & 0 & -\frac{1}{6} \\ \frac{1}{9} & -\frac{1}{18} & \frac{1}{9} & -\frac{1}{18} & \frac{1}{36} & -\frac{1}{18} & \frac{1}{9} & -\frac{1}{18} & \frac{1}{9} \end{pmatrix}, \tag{14.16b}$$

$$D = \operatorname{diag}\Big(1 \;\; 1 \;\; 1 \;\; 2 \;\; 2 \;\; 4 \;\; \tfrac{1}{2} \;\; \tfrac{1}{2} \;\; \tfrac{1}{4} \Big). \tag{14.16c}$$

The proof follows from a straightforward but tedious evaluation (see [16]).

The column-vectors of V are the right-eigenvectors of G, the row-vectors of V^{-1} are the left-eigenvectors of G. Below we depict these vectors as 9-point stencils, together with the corresponding eigenvalues. The stencils with the first six right-eigenvectors are the ones as evoked by discretization with bilinear finite elements of partial derivatives, ranging from zeroth to second order. The stencils read:

$$\begin{bmatrix} -\frac{1}{3} & \frac{1}{6} & -\frac{1}{3} \\ -\frac{1}{3} & \frac{1}{6} & -\frac{1}{3} \\ -\frac{1}{3} & \frac{1}{6} & -\frac{1}{3} \end{bmatrix}, \quad \begin{bmatrix} -\frac{1}{6} & \frac{1}{3} & -\frac{1}{6} \\ -\frac{2}{3} & \frac{4}{3} & -\frac{2}{3} \\ -\frac{1}{6} & \frac{1}{3} & -\frac{1}{6} \end{bmatrix} \quad \sim -h^2 \frac{\partial^2}{\partial x^2}, \qquad \lambda_1 = 1, \tag{14.17a}$$

$$\begin{bmatrix} -\frac{1}{3} & -\frac{1}{3} & -\frac{1}{3} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ -\frac{1}{3} & -\frac{1}{3} & -\frac{1}{3} \end{bmatrix}, \quad \begin{bmatrix} -\frac{1}{6} & -\frac{2}{3} & -\frac{1}{6} \\ \frac{1}{3} & \frac{4}{3} & \frac{1}{3} \\ -\frac{1}{6} & -\frac{2}{3} & -\frac{1}{6} \end{bmatrix} \quad \sim -h^2 \frac{\partial^2}{\partial y^2}, \qquad \lambda_2 = 1, \tag{14.17b}$$

$$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}, \quad \begin{bmatrix} \frac{1}{4} & 0 & -\frac{1}{4} \\ 0 & 0 & 0 \\ -\frac{1}{4} & 0 & \frac{1}{4} \end{bmatrix} \quad \sim -h^2 \frac{\partial^2}{\partial x \partial y}, \qquad \lambda_3 = 1, \tag{14.17c}$$

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}, \quad \begin{bmatrix} -\frac{1}{12} & 0 & \frac{1}{12} \\ -\frac{1}{3} & 0 & \frac{1}{3} \\ -\frac{1}{12} & 0 & \frac{1}{12} \end{bmatrix} \quad \sim h \frac{\partial}{\partial x}, \qquad \lambda_4 = 2, \tag{14.17d}$$

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}, \quad \begin{bmatrix} \frac{1}{12} & \frac{1}{3} & \frac{1}{12} \\ 0 & 0 & 0 \\ -\frac{1}{12} & -\frac{1}{3} & -\frac{1}{12} \end{bmatrix} \quad \sim h \frac{\partial}{\partial y}, \qquad \lambda_5 = 2, \tag{14.17e}$$

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \quad \begin{bmatrix} \frac{1}{36} & \frac{1}{9} & \frac{1}{36} \\ \frac{1}{9} & \frac{4}{9} & \frac{1}{9} \\ \frac{1}{36} & \frac{1}{9} & \frac{1}{36} \end{bmatrix} \quad \sim I, \qquad \lambda_6 = 4, \tag{14.17f}$$

$$\begin{bmatrix} -\frac{1}{6} & +\frac{1}{12} & -\frac{1}{6} \\ 0 & 0 & 0 \\ \frac{1}{6} & -\frac{1}{12} & \frac{1}{6} \end{bmatrix}, \quad \begin{bmatrix} -1 & 2 & -1 \\ 0 & 0 & 0 \\ 1 & -2 & 1 \end{bmatrix} \quad \sim -2h^3 \frac{\partial^3}{\partial y \partial x^2}, \lambda_7 = \tfrac{1}{2}, \tag{14.17g}$$

$$\begin{bmatrix} \frac{1}{6} & 0 & -\frac{1}{6} \\ -\frac{1}{12} & 0 & \frac{1}{12} \\ \frac{1}{6} & 0 & -\frac{1}{6} \end{bmatrix}, \quad \begin{bmatrix} 1 & 0 & -1 \\ -2 & 0 & 2 \\ 1 & 0 & -1 \end{bmatrix} \quad \sim -2h^3 \frac{\partial^3}{\partial x \partial y^2}, \lambda_8 = \tfrac{1}{2}, \tag{14.17h}$$

$$\begin{bmatrix} \frac{1}{9} & -\frac{1}{18} & \frac{1}{9} \\ -\frac{1}{18} & \frac{1}{36} & -\frac{1}{18} \\ \frac{1}{9} & -\frac{1}{18} & \frac{1}{9} \end{bmatrix}, \quad \begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix} \quad \sim h^4 \frac{\partial^4}{\partial x^2 \partial y^2}, \qquad \lambda_9 = \tfrac{1}{4}. \tag{14.17i}$$

We denote the set of right-eigenvectors by $\{v_j\}_{j=1,\cdots,9}$ and the set of left-eigenvectors by $\{w_j\}_{j=1,\cdots,9}$. The right-eigenvectors $\{v_j\}_{j=1,\cdots,9}$ form a linearly independent set of vectors that span $\mathbb{R}^9$. Because trivially $V^{-1}V = I$, we observe that the two sets of eigenvectors satisfy a *bi-orthonormality* relation. Therefore, if we have some stencil f^*, we can easily determine the coefficients of the unique linear combination of the v_j to

which it is equal

$$f = \sum_{j=1}^{9} (w_j, f) v_j. \tag{14.18}$$

For the advection-diffusion example described by (14.10) and (14.11) we find

$$L_l^\star = \tfrac{1}{2} v_1^\star + v_4^\star + \tfrac{1}{12} v_8^\star + \tfrac{1}{12} v_9^\star. \tag{14.19}$$

Because

$$G^n = V D^n V^{-1}, \tag{14.20}$$

we find after the n times subsequent application of the Galerkin coarse grid approximation the stencil

$$L_{l-n}^\star = \tfrac{1}{2} v_1^\star + 2^n v_4^\star + (\tfrac{1}{2})^n \tfrac{1}{12} v_8^\star + (\tfrac{1}{4})^n \tfrac{1}{12} v_9^\star \tag{14.21}$$

on the n times coarsened grid. We observe how, for increasing n, $L_{l-n}^\star$ is dominated by the central advection stencil $v_4^\star$.

Remark Analogous results hold for 7-point stencils of type

$$\begin{bmatrix} f_6 & f_7 & \\ f_3 & f_4 & f_5 \\ & f_1 & f_2 \end{bmatrix} \tag{14.22}$$

provided (14.7) holds and

$$P_k \sim \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & \\ \frac{1}{2} & 1 & \frac{1}{2} \\ & \frac{1}{2} & \frac{1}{2} \end{bmatrix}. \tag{14.23}$$

Here also exists an eigenvalue decomposition of G, which reads:

$$G = V D V^{-1}, G, V, D \in \mathbb{R}^7 \times \mathbb{R}^7. \tag{14.24}$$

D is a diagonal matrix representing the eigenvalues of G and

$$V = \begin{pmatrix} 0 & -1 & -\frac{1}{2} & -\frac{1}{6} & -\frac{1}{3} & \frac{1}{12} & 1 \\ 0 & 0 & \frac{1}{2} & \frac{1}{6} & -\frac{1}{6} & \frac{1}{12} & -1 \\ -1 & 0 & -\frac{1}{2} & -\frac{1}{3} & -\frac{1}{6} & \frac{1}{12} & -1 \\ 2 & 2 & 1 & 0 & 0 & \frac{1}{2} & 0 \\ -1 & 0 & -\frac{1}{2} & \frac{1}{3} & \frac{1}{6} & \frac{1}{12} & 1 \\ 0 & 0 & \frac{1}{2} & -\frac{1}{6} & \frac{1}{6} & \frac{1}{12} & 1 \\ 0 & -1 & -\frac{1}{2} & \frac{1}{6} & \frac{1}{3} & \frac{1}{12} & -1 \end{pmatrix}, \tag{14.25a}$$

$$V^{-1} = \begin{pmatrix} \frac{1}{6} & -\frac{1}{3} & -\frac{1}{3} & \frac{1}{6} & -\frac{1}{3} & -\frac{1}{3} & \frac{1}{6} \\ -\frac{1}{3} & -\frac{1}{3} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & -\frac{1}{3} & -\frac{1}{3} \\ -\frac{1}{6} & \frac{5}{6} & -\frac{1}{6} & -\frac{1}{6} & -\frac{1}{6} & \frac{5}{6} & -\frac{1}{6} \\ 0 & 1 & -1 & 0 & 1 & -1 & 0 \\ -1 & -1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ \frac{1}{6} & -\frac{1}{6} & -\frac{1}{6} & 0 & \frac{1}{6} & \frac{1}{6} & -\frac{1}{6} \end{pmatrix}, \tag{14.25b}$$

$$D = \operatorname{diag}\begin{pmatrix} 1 & 1 & 1 & 2 & 2 & 4 & \frac{1}{2} \end{pmatrix}. \tag{14.25c}$$

14.2.3 Upwind prolongation

The Galerkin coarse grid approximation (14.5) can be looked upon as discretizing L_k into L_{k-1} (the analogue of discretizing L into L_{l-1}). The type of this discretization is determined by the choice of P_k and R_{k-1}. When we choose (14.7) and the bilinear prolongation (14.9), the type of discretization is some central differencing scheme. This central differencing is the reason why the ratio of numerical advection and numerical diffusion is increasing at the same pace as the mesh Péclet number when the grid is coarsened. The remedy is to use an upwind prolongation, that is related to the method of characteristics. The restriction is again defined by (14.7), which hereby becomes of upwind-type as well. By this remedy, the Galerkin coarse grid approximation (14.5) becomes an upwind-type discretization of L_k into L_{k-1}. As an example, suppose we only accept information from the left-hand side, i.e. the solution upstream. This corresponds to a prolongation with biased weights as depicted in Figure 14.3.

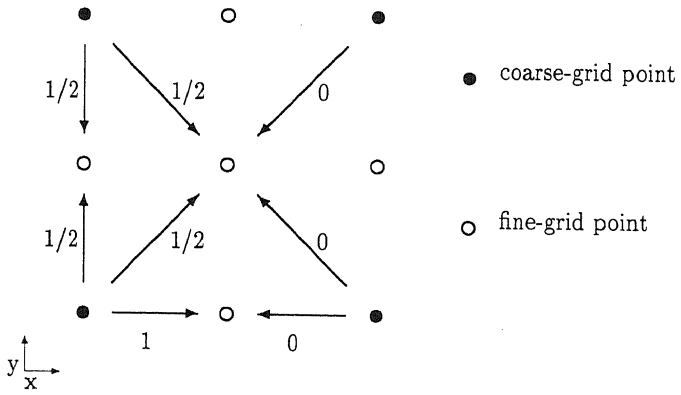

Figure 14.3: Example of upwind prolongation

In stencil notation the upwind prolongation and restriction look like

$$\begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ 1 & 1 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 \end{bmatrix}. \tag{14.26}$$

In this case the Galerkin coarse grid approximation is given by

$$G_{\text{up}} = V\tilde{D}V^{-1}, G_{\text{up}}, \tilde{D} \in \mathbb{R}^9 \times \mathbb{R}^9, \tag{14.27}$$

with V given by (14.16a) and

$$\tilde{D} = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 & \frac{1}{3} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{12} & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{12} & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} \end{pmatrix}. \tag{14.28}$$

We observe that now the amplification of the diffusion-stencil (14.17a) in the x-direction keeps pace with the amplification of the advection-stencil (14.17d). For the example from section 14.2.1 and by repeatedly applying (14.7) and (14.5) we obtain on the n times coarsened grid Ω_{l-n} (see [16]):

$$L^*_{l-n} = 2^n \tfrac{1}{2} v^*_1 + 2^n v^*_4 + (\tfrac{1}{2})^n \tfrac{1}{12} v^*_8 + (\tfrac{1}{2})^n \tfrac{1}{12} v^*_9, \tag{14.29}$$

that is

$$L^*_{l-n} = 2^n \begin{bmatrix} -\frac{1}{6} & \frac{1}{6} & 0 \\ -\frac{2}{3} & \frac{2}{3} & 0 \\ -\frac{1}{6} & \frac{1}{6} & 0 \end{bmatrix} + 2^{-n} \cdots \tag{14.30}$$

and we observe that the upwind approximation of the derivative is preserved.

Generally speaking, the point we like to make is as follows. We can use the 9-point discretization that we prefer on the finest grid (not necessarily of upwind-type). The coarse grid correction (14.4) can be viewed as accelerating method for the relaxation on the finest grid. When we use bilinear interpolation for prolongation, the ratio of convection over numerical diffusion may grow much larger on increasingly coarser grids and thereby divergence of the CGC may occur. When we use a prolongation of upwind-type, the ratio of convection over numerical diffusion remains roughly the same at all grids in use with the multigrid algorithm.

14.2.4 Matrix-dependent prolongations and restrictions

The bias in the upwind prolongation may have to be different in each grid-point because of the varying problem-coefficients. The only way to obtain information about these coefficients is by deriving them from the matrix (because we agreed to construct a black box which only needs to know the matrix and the right-hand side). Therefore, in this context, the upwind prolongation wished for is a matrix-dependent prolongation. Matrix-dependent prolongations were introduced in [1, 4]. In [16] a matrix-dependent prolongation operator has been proposed, able to handle both the case of dominant advection (in general directions) and interface problems at the same time. Here we give only an outline of the method. The grid Ω_k is split into four disjoint sub-grids as follows:

$$\Omega_{k,(0,0)} \equiv \Omega_{k-1},$$

$$\Omega_{k,(1,0)} \equiv \{(x + h_k, y) \in \Omega_k \mid (x, y) \in \Omega_{k-1}\},$$

$$\Omega_{k,(0,1)} \equiv \{(x, y + h_k) \in \Omega_k \mid (x, y) \in \Omega_{k-1}\},$$

$$\Omega_{k,(1,1)} \equiv \{(x + h_k, y + h_k) \in \Omega_k \mid (x, y) \in \Omega_{k-1}\},$$

where h_k is the mesh-size of grid Ω_k.

1. Let $\xi \in \Omega_{k,(1,0)}$ or $\xi \in \Omega_{k,(0,1)}$ be a point where we have to interpolate a coarse grid correction. Decompose the matrix L_k in its symmetric and antisymmetric part. The symmetric part S_k is presumed to correspond with diffusion and the zeroth order term, the antisymmetric part T_k with advection.
2. Reconstruct the diffusion and zeroth order coefficients at ξ from S_k, and the advection coefficients from T_k.
3. Use expressions that define an optimal choice for each sample of a set of degenerated cases for L_k.
4. At the fine-grid points in $\Omega_{k,(0,0)}$, adopt the values on Ω_{k-1}.
5. At the fine-grid points in $\Omega_{k,(1,1)}$, solve the homogeneous equation to obtain the correction.

Applying the derived formulae to the particular example of section 14.2.1 we obtain the upwind prolongation in Figure 14.3. Also here we adhere to (14.7) and (14.5), though the implementation of the latter is far from trivial. The actual computation of the coarse grid matrices takes less work than the Incomplete Line LU-decompositions described in the next section. The above is employed in the code MGD9V (this author). The code has a NAG-like outward appearance and is readily available. It uses the sawtooth multigrid correction scheme [13] (see Chapter 13) and Incomplete Line LU for smoother. For a more detailed motivation of the prolongation and a description of the code, together with numerical experiments to illustrate its good behaviour, see [16]. Reusken [10] proves uniform convergence for a multigrid method applied to a 1D singularly perturbed boundary value problem when matrix-dependent prolongations and restrictions are employed (see also Chapter 13, this book).

14.3 Smoothers for the advection-diffusion equation

A comprehensive survey of smoothers applicable to the advection-diffusion equation (14.1) can be found in [14, § 7]. For non-adaptive grids the Incomplete Line LU (ILLU) (or Incomplete Block LU) appears to be the most robust choice (see [8, 9, 11, 16]). By this method full advantage is taken of the matrix-structure. Here we repeat the general outline of this method, which has been originated by Underwood [12] (see also [3] for an overview on block-type methods). We want to solve the linear system

$$Ax = b \tag{14.31}$$

that we assume to have a block tridiagonal form, so

$$A = \begin{pmatrix} D_1 & U_1 & & & \\ L_2 & D_2 & U_2 & & \\ & L_3 & D_3 & \cdot & \\ & & \cdot & \cdot & \cdot \\ & & & \cdot & D_{n_y} \end{pmatrix} \tag{14.32}$$

where n_y is the number of grid-points in the y-direction. The block D_j has the tridiagonal form:

$$D_j = \begin{pmatrix} d_{1j} & u_{1j} & & & & \\ l_{2j} & d_{2j} & u_{2j} & & & \\ & l_{3j} & d_{3j} & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & \cdot \\ & & & & \cdot & d_{n_x j} \end{pmatrix} \tag{14.33}$$

where n_x is the number of grid-points in the x-direction. In case of a discretization with nine-point stencils the blocks L_j and U_j have a similar form. The ILLU-decomposition is defined by

$$L_j(j = 2, \ldots, n_y), \overline{D}_j(j = 1, \ldots, n_y), U_j(j = 1, \ldots, n_y - 1)$$

with

$$\overline{D}_1 = D_1, \tag{14.34a}$$

$$\overline{D}_j = D_j - \mathbf{tridiag}(L_j \overline{D}_{j-1}^{-1} U_{j-1}), \quad j = 2(1)n_y. \tag{14.34b}$$

The operator **tridiag**() forces a block (by clipping) into the sparsity pattern of the D_j. Without this particular operator, the factorization of A would be a complete one (see [5, § 8.5]). Performing one ILLU-relaxation sweep requires the following steps

ILLU-sweep:

$r = b - Ax$;
$z_1 = r_1$;
$z_j = r_j - L_j \overline{D}_{j-1}^{-1} z_{j-1}, \quad j = 2(1)n_y$;
$c_{n_y} = \overline{D}_{n_y}^{-1} z_{n_y}$;
$c_j = \overline{D}_j^{-1}(z_j - U_j c_{j+1}), \quad j = n_y - 1(-1)1$;
$x_{new} = x + c$.

In [15] the ILLU-relaxation has been generalized from the case of a discretized scalar PDE to a set of coupled PDEs.

There exist several degenerated cases for which ILLU becomes a complete decomposition, e.g. when all L_j or all U_j are zero. Another case of interest is the following

Proposition 14.3.1 *If $L_j = \alpha_j D_1, U_{j-1} = \beta_{j-1} D_1, D_j = \gamma_j D_1, j = 2(1)n_y$ where $\alpha_j, \beta_{j-1}, \gamma_j$ are arbitrary scalars, then the ILLU-decomposition, if existent, is a complete factorization.*

Proof. If

$$\mathbf{tridiag}(L_j \overline{D}_{j-1}^{-1} U_{j-1}) = L_j \overline{D}_{j-1}^{-1} U_{j-1}, j = 2(1)n_y,$$

then the factorization is complete. We prove that scalars $\mu_j \neq 0$ exist such that

$$\overline{D}_j = \mu_j D_1, j = 1(1)n_y - 1.$$

The equality holds trivially for $j = 1$ with $\mu_1 = 1$. For $j > 1$ we obtain (by induction)

$$\overline{D}_j = D_j - \mathbf{tridiag}(L_j \overline{D}_{j-1}^{-1} U_{j-1}) \Rightarrow \overline{D}_j = D_j - \mathbf{tridiag}(L_j \mu_{j-1}^{-1} D_1^{-1} U_{j-1}) \Rightarrow$$

$$D_j = \mu_j D_1$$

with

$$\mu_j = \gamma_j - \alpha_j \beta_{j-1} \mu_{j-1}^{-1}.$$

From the assumption that the decomposition exists, it follows that $\mu_j \neq 0$. □

Note that (but for existence of the ILLU-decomposition) this proposition applies to all the stencils on the right-hand side of (14.17), which correspond to the various partial derivatives.

14.4 The Molenkamp test-problem

In Chapter 2 of this book, Vreugdenhil applies several difference schemes to the Molenkamp problem. Those schemes are generated by central differencing in space and implicit time-stepping. At each time-step a large, sparse linear system has to be solved. In Chapter 2 this is done by using a standard banded-matrix routine from the NAG library. The corresponding LU-decomposition needs to be done only once, yet this method puts too high demands concerning CPU-time and memory allocation. Typically, on an $n \times n$-grid such a direct method takes

$$n^4 \text{ flops}$$

for the decomposition, and at least

$$2n^3 \text{ flops}$$

for each subsequent solution step (one flop is the amount of work associated with a multiplication joined with an addition). The storage requirements amount to

$$2(n+1)n^2 \text{ reals.}$$

Here we report on the performance of the iterative multigrid solver MGD9V (section 14.2.4) for the solution of the said large, sparse linear systems. Of course, also here the automatic construction of the coarse grid matrices and the ILLU-decompositions need to be done only once. The storage requirements of MGD9V amount to

$$\frac{68}{3} n^2 \text{ reals.}$$

The total amount of work is not a fixed function of n because for an iterative method it also depends on its convergence rate and the desired tolerance. However, if a multigrid method is well constructed, it is known from both theory and practice that the work necessary to reach some tolerance is ideally proportional to as little as n^2, i.e. the number of grid-points. In this section we check by experiment whether this statement holds for MGD9V.

In scheme (2.28) of Chapter 2, two parameters do occur: θ and α. The θ determines the degree of time-implicitness: $\theta = 1$ corresponds to backward Euler, $\theta = 0$ corresponds to forward Euler. Only if $\theta = \frac{1}{2}$ the accuracy of the scheme is of second order in time. The parameter α determines the space-discretization: $\alpha = 0$ corresponds to the classical five-point scheme with central differences, $\alpha = \frac{1}{6}$ corresponds to the bilinear finite element method and $\alpha = \frac{1}{4}$ to the box-scheme. In principle, the convergence rate of MGD9V may depend on these parameters.

14.4.1 Complexity and the Courant number

We solve the Molenkamp problem with the 2-d implicit schemes of Chapter 2, and perform - as specified - a full rotation of the cone. When we solve the linear system at each point of time by means of complete LU-factorization, this is called the *direct approach*; when at each point of time we use the multigrid solver MGD9V instead, this is called the *multigrid approach*. We use the solution at the previous point of time as initial solution for MGD9V, and we solve the linear system up to a certain (fixed) tolerance. Suppose we perform numerical experiments with varying n, but with fixed Courant number σ. When n is increased by a factor 2, the direct approach takes twice the number of time steps. Because of the complexity of the decomposition and the subsequent time-stepping, the work for one rotation then increases by a factor 2^4. With the multigrid approach the work for one rotation increases by a factor 2^3, a factor 2 is explained by the number of time-steps and the remaining factor 2^2 is explained by the number of grid-points. This statement is valid under the condition that MGD9V presents mesh-independent convergence rate. The experimental verification of the statement follows from Figure 14.4. Horizontally we put the 2-logarithm of the Courant number σ, vertically we show the number of multigrid cycles needed for a full rotation of the cone. Note that a multigrid cycle is a fixed amount of work for a given grid, and proportional to n^2. The experiments are performed with $\alpha = \frac{1}{6}$ and $\theta = \frac{1}{2}$, which gives the highest accuracy of the solution. For $\sigma = \frac{1}{2}$ the number of time-steps is 250, 500, 1000 for a 41×41, 81×81 and 161×161 grid respectively. We observe that for Courant numbers in the range of interest ($\sigma \leq 1$),

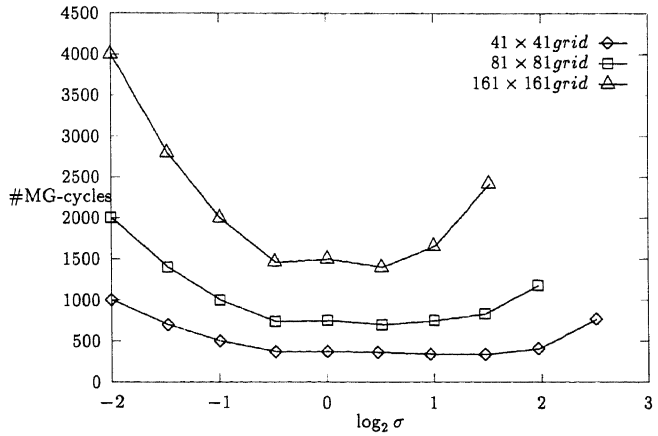

Figure 14.4: MG-cycles consumed versus $\log_2$ of Courant number, $\alpha = \frac{1}{6}$ and $\theta = \frac{1}{2}$

the number of multigrid cycles is multiplied by a factor 2 when the grid-dimensions are doubled. This factor is due to the doubled number of time-steps. Apparently the number of multigrid cycles per time-step remains unchanged, which demonstrates the mesh-independent convergence rate of the multigrid algorithm. Indeed, MGD9V turns out to be very efficient; an average of only two multigrid cycles per time-step proves to be quite common for this problem. This is due to the high convergence rate (typically a reduction factor of 10^{-4} per multigrid cycle), and the advantage that is taken by using the solution at the previous point of time as initial one (the smaller the time-step, the better). Further we observe in Figure 14.4 that for each grid a range of Courant numbers

exists for which the amount of work is constant (we call this a plateau). Of course, the lower Courant numbers yield the more accurate results because of the smaller time-step. Apparently, when increasing the time accuracy, the computational cost of the multigrid approach does not increase (provided σ remains within the range of the plateau). This is obviously contrary to the direct approach. For very small Courant numbers (at the left-hand side of the plateau) the amount of work increases in a linear way with σ^{-1}. That is because at each time step, at least one multigrid cycle has to be performed. (A mere fraction of a multigrid cycle cannot be performed, of course). For very high Courant numbers (at the right-hand side of the plateau) the amount of work increases (and even divergence may occur) because of the deteriorating convergence rate and the worsening initial solutions. For $\alpha = 0$ we find similar results, see Figure 14.5.

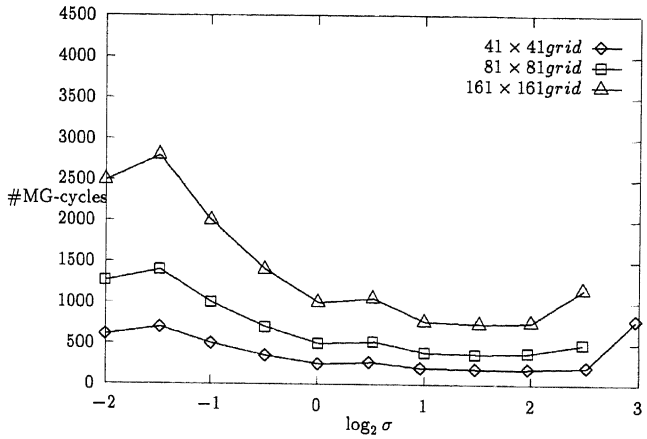

Figure 14.5: MG-cycles consumed versus $\log_2$ of Courant number, $\alpha = 0$ and $\theta = \frac{1}{2}$

14.4.2 Miscellaneous results and remarks

In section 14.4.1 we reported on results of MGD9V for $\alpha = \frac{1}{6}$ and $\alpha = 0$, but not on results for $\alpha = \frac{1}{4}$. In fact, MGD9V fails to convergence for this value of α even for extremely small Courant numbers. The explanation is as follows. Let $\sigma = 0$, then the difference scheme involves the following stencil for $\alpha = \frac{1}{4}$ (see (2.28) in Chapter 2 of this book)

$$\begin{bmatrix} \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \\ \frac{1}{8} & \frac{1}{4} & \frac{1}{8} \\ \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \end{bmatrix}.$$

When we apply the corresponding matrix to the chess-board grid-function

$$c(ih_l, jh_l) = (-1)^{i+j}$$

defined on Ω_l, we obtain the zero grid-function. Also when Dirichlet boundary conditions are taken into account, we obtain a similar result as we will see below. Consider the unit square with Dirichlet boundary conditions all around. We take a uniform $[0:n] \times [0:n]$-grid on this area. Consider the matrix for $\alpha = \frac{1}{4}$, $\sigma = 0$. We substitute the discretized

Dirichlet boundary conditions. Define the near chess-board grid-function

$$b(ih, jh) = (-1)^{i+j} \sin(\pi \frac{i}{n}) \sin(\pi \frac{j}{n}), \quad i = 1(1)n-1, \quad j = 1(1)n-1, \tag{14.35}$$

with $h = \frac{1}{n}$. (This grid-function vanishes on the boundaries of the unit square.) It can be easily verified that b corresponds to an eigenvector of the matrix, with eigenvalue

$$\lambda = \sin^4(\frac{\pi h}{2}).$$

Hence, for $h \downarrow 0$, this eigenvalue goes very rapidly to zero:

$$\lambda = \tfrac{1}{16}\pi^4 h^4.$$

It follows that a small perturbation of the right-hand side of the linear system to be solved, may produce a huge perturbation of the solution, an effect that is not present for the continuous problem. In this sense we speak of an unstable discretization. The foregoing explains why we find a highly oscillating (spurious) solution when we solve, or try to solve, the linear system for $\alpha = \frac{1}{4}$ and $\sigma = 0$.

A similar argument holds when $\sigma > 0$, as we will see below. Consider the matrix $C(h)$ that corresponds to the convection-stencil (in the x-direction, $\alpha = \frac{1}{4}$)

$$\frac{1}{h} \begin{bmatrix} -\frac{1}{4} & 0 & +\frac{1}{4} \\ -\frac{1}{2} & 0 & +\frac{1}{2} \\ -\frac{1}{4} & 0 & +\frac{1}{4} \end{bmatrix}$$

and with substituted Dirichlet boundary conditions. When the matrix $C(h)$ is applied to the grid-function (14.35), then

$$\|C(h)b\| \leq \frac{4}{h} \sin^3(\frac{\pi h}{2}),$$

in the maximum-norm. Apparently, here too the near chess-board grid-function is annihilated for $h \downarrow 0$.

In Table 14.1 we observe how the accuracy and the amount of work depends on θ. The results are for a 161×161-grid, $\alpha = \frac{1}{6}$, Courant number $\sigma = \frac{1}{2}$, and for a full rotation of the cone. Apparently, the results for $\theta = \frac{1}{2}$ are both the most accurate and

Table 14.1: Accuracy and amount of work depending on θ

θ	# MG-cycles	$c_{\min}$	$1 - c_{\max}$
$\frac{1}{2}$	2000	-2.9010^{-6}	4.3310^{-7}
$\frac{3}{4}$	2052	-7.2710^{-5}	7.9810^{-2}
1	3000	-1.3110^{-4}	1.4210^{-1}

the cheapest to obtain.

In Table 14.2 we measure the CPU-seconds consumed on a Silicon Graphics workstation (R3000 Processor), by the multigrid approach for various grids. Again, the results are for a full rotation of the cone, *fixed* Courant number $\sigma = \frac{1}{2}$ and $\theta = \frac{1}{2}$, $\alpha = \frac{1}{6}$. The benchmark-problem (see Chapter 15 of this book) took 2.40 CPU-seconds. By means of the timing of this benchmark-problem, we can provide Table 14.2 also with the (fictitious) numbers of CPU-seconds consumed by the direct approach on the R3000 Processor (courtesy of Vreugdenhil). We observe how the results match the predictions as made at the outset of section 14.4.1.

Table 14.2: CPU-seconds consumed for a full rotation

	41×41	81×81	161×161
direct approach	58	721	
multigrid approach	28	216	1644

14.5 Conclusions

In this chapter we considered the multigrid solution of advection-diffusion problems. The use of implicit methods, as some of them are described in Chapter 2 of this book, requires the solution of large, sparse linear systems. We studied the feasibility of a black-box multigrid solver for the solution of such systems. We have shown that a standard choice for the prolongation and restriction is not satisfactory and that an upwind approach for these operators leads to an important improvement.

The implementation of this approach, together with a robust relaxation method, resulted in the multigrid-code MGD9V. This code proves to be a highly efficient iterative solver. Hence it is feasible to solve problems with both small mesh-size and small time-steps. In this way, implicit methods become competitive again. It is noteworthy that smaller time-steps do not necessarily increase the amount of work, because of a better convergence rate and a better initial solution.

The code performs only for the scalar case and within the constraints of a regular domain and a structured grid. In [16] various results are reported for some hard advection-diffusion problems (with stagnation points) and for problems with discontinuous diffusion-coefficients (among which Kershaw's problem). In [15] a result is reported for Van der Vorst's aquifer-problem that is marked by both dominating convection and discontinuous diffusion-coefficients. The code (written in standard FORTRAN 77) is available from this author (electronic mail address: *pauldz@cwi.nl*).

References

[1] R.E. Alcouffe, A. Brandt, J.E. Dendy Jr. and J.W. Painter: The multigrid method for the diffusion equation with strongly discontinuous coefficients, *SIAM J. Sci. Stat. Comput.*, **2**(4), 1981, pp. 430–454.

[2] A. Brandt: Multi-level adaptive techniques (MLAT) for partial differential equations: ideas and software, in: J. Rice (Ed.) *Proc. Symposium on Mathematical Software*, (Academic, New York, 1977), pp. 277–318.

[3] P. Concus, G.H. Golub and G. Meurant: Block preconditioning for the conjugate gradient method, *SIAM J. Sci. Stat. Comput.*, **6** (1), 1985, pp. 220–252.

[4] J.E. Dendy Jr.: Blackbox multigrid for nonsymmetric problems, *Appl. Math. Comput.* **13**, 1983, pp. 261–283.

[5] I.S. Duff, A.M. Erisman, J.K. Reid: *Direct Methods for Sparse Matrices* (Monographs on Numerical Analysis, Clarendon Press, Oxford, 1986).

[6] W. Hackbusch: *Multi-Grid Methods and Applications* (Springer Ser. Comput. Math. **4**, Berlin, 1985).

[7] P.W. HEMKER: On the order of prolongations and restrictions in multigrid procedures, *J. Comput. Appl. Math.*, **32**, 1990, pp. 423–429.

[8] P.W. HEMKER AND P.M. DE ZEEUW: Some implementations of multigrid linear system solvers, in: D.J. Paddon and H. Holstein (Eds.) *Multigrid methods for integral and differential equations* (Inst. Math. Appl. Conf. Ser. New Ser., Oxford Univ. Press, New York, 1985), pp. 85–116.

[9] R. KETTLER: Analysis and comparison of relaxation schemes in robust multigrid and preconditioned conjugate gradient methods, in: W. Hackbusch and U. Trottenberg (Eds.), *Lecture Notes in Mathematics*, **960** (Springer, Berlin, 1981), pp. 502–534.

[10] A. REUSKEN: Multigrid with matrix-dependent transfer operators for a singular perturbation problem, *Report RANA 92-09* (Dept. of Mathematics and Computing Science, Eindhoven University of Technology, The Netherlands, 1992).

[11] P. SONNEVELD, P. WESSELING AND P.M. DE ZEEUW: Multigrid and conjugate gradient methods as convergence acceleration techniques, in: D.J. Paddon and H. Holstein (Eds.) *Multigrid methods for integral and differential equations* (Inst. Math. Appl. Conf. Ser. New Ser., Oxford Univ. Press, New York, 1985), pp. 117–167.

[12] R.R. UNDERWOOD: An approximate factorization procedure based on the block Cholesky decomposition and its use with the conjugate gradient method, *Report NEDO-11386* (General Electric Co., Nuclear Energy Div., San Jose, California, 1976).

[13] P. WESSELING: A robust and efficient multigrid method, in: W. Hackbusch and U. Trottenberg (Eds.), *Lecture Notes in Mathematics*, **960** (Springer, Berlin, 1981), pp. 614–630.

[14] P. WESSELING: *An Introduction to Multigrid Methods* (John Wiley & Sons Ltd., Chichester, England, 1991).

[15] P.M. DE ZEEUW: Incomplete line LU for discretized coupled PDEs as preconditioner in Bi-CGSTAB, *Report NM-R9213* (CWI, Amsterdam, 1992).

[16] P.M. DE ZEEUW: Matrix-dependent prolongations and restrictions in a blackbox multigrid solver, *J. Comput. Appl. Math.*, **33**, 1990, pp. 1–27.

[17] P.M. DE ZEEUW AND E.J. VAN ASSELT: The convergence rate of multi-level algorithms applied to the convection-diffusion equation, *SIAM J. Sci. Stat. Comput.*, **6** (2), 1985, pp. 492–503.