Chapter 1

Literature

Remark 1.1 *Literature.* There are several text books about multigrid methods, e.g.,

- Briggs et al. (2000), easy to read introduction,
- Hackbusch (1985), the classical book, sometimes rather hard to read,
- Shaidurov (1995),
- Wesseling (1992),
- Trottenberg et al. (2001).

$\square$

Chapter 2

Model Problems

Remark 2.1 *Motivation.* The basic ideas and properties of multigrid methods will be explained in this course on two model problems. $\square$

Example 2.2 *A two-point boundary value problem.* Consider the boundary value problem

$$\begin{aligned} -u'' + au &= f \quad \text{in } \Omega = (0,1), \\ u(0) = u(1) &= 0, \end{aligned} \tag{2.1}$$

with $a \geq 0$ for all $x \in (0,1)$. Often, this problem can be solved analytically. However, multigrid methods are solvers for linear system of equations that arise, e.g., in the discretization of partial differential equations. For this reason, discretizations of (2.1) will be considered: a finite difference method and a finite element method. These discretizations are described in detail in the lecture notes of Numerical Mathematics III.

Consider an equidistant triangulation of Ω with the nodes $0 = x_0 < x_1 < \ldots < x_N = 1$ with the distance $h = 1/N$ between two neighboring nodes.

The application of the second order finite difference scheme leads to a linear system of equations

$$A\mathbf{u} = \mathbf{f} \tag{2.2}$$

with the tridiagonal matrix $A \in \mathbb{R}^{(N-1)\times(N-1)}$ with

$$a_{ij} = \frac{1}{h^2}\begin{cases} 2 & \text{if } i = j, i = 1, \ldots, N-1, \\ -1 & \text{if } i = j-1, i = 2, \ldots, N-1, \text{ or } i = j+1, i = 1, \ldots, N-2, \\ 0 & \text{else,} \end{cases}$$

and the right-hand side

$$(\mathbf{f})_i = f_i = f(x_i), \quad i = 1, \ldots, N-1.$$

Using the P_1 finite element method leads to a linear system of equations (2.2) with the tridiagonal matrix

$$a_{ij} = \frac{1}{h}\begin{cases} 2 & \text{if } i = j, i = 1, \ldots, N-1, \\ -1 & \text{if } i = j-1, i = 2, \ldots, N-1, \text{ or } i = j+1, i = 1, \ldots, N-2, \\ 0 & \text{else,} \end{cases}$$

and the right-hand side

$$f_i = \int_{x_{i-1}}^{x_{i+1}} f(x)\varphi_i(x)\, dx, \quad i = 1, \ldots, N-1,$$

where $\varphi_i(x)$ is the function from the local basis that does not vanish in the node x_i. Note that there is a different scaling in the matrices of the finite difference and the finite element method. □

Example 2.3 *Poisson equation in two dimension.* The Poisson equation in two dimensions with homogeneous boundary conditions has the form

$$\begin{array}{rcll} -\Delta u &=& f & \text{in } \Omega = (0,1)^2, \\ u &=& 0, & \text{on } \partial\Omega. \end{array} \tag{2.3}$$

Again, an equidistant grid is considered for the discretization of (2.3) with mesh width $h_x = h_y = h = 1/N$. The nodes are numbered lexicographically.

The application of the finite difference method with the five point stencil leads to a linear system of equations of dimension $(N-1) \times (N-1)$ with the matrix entries

$$a_{ij} = \frac{1}{h^2} \begin{cases} 4 & \text{if } i = j, \\ -1 & \text{if } i = j-1, i = j+1, i = j-(N+1), i = j+(N+1), \\ 0 & \text{else,} \end{cases}$$

with obvious modifications for the nodes near the boundary of the domain.

For applying the P_1 finite element method, the grid has to be decomposed into triangles. Using a decomposition where the edges are either parallel to the axes or parallel to the line $y = x$, one obtains the matrix

$$a_{ij} = \begin{cases} 4 & \text{if } i = j, \\ -1 & \text{if } i = j-1, i = j+1, i = j-(N+1), i = j+(N+1), \\ 0 & \text{else,} \end{cases}$$

again with obvious modifications for the degrees of freedom near the boundary. □

Remark 2.4 *Properties of the matrices.*

- The matrix A is sparse. In one dimensions, there are not more than three non-zero entries per row and column, the matrix is even tridiagonal. In the two-dimensional case, there are not more than five non-zero entries per row and column.
- The matrix A is symmetric. It follows that all eigenvalues are real.
- The matrix A is positive definite, i.e.,

  $$\mathbf{x}^T A \mathbf{x} > 0 \quad \forall\, \mathbf{x} \setminus \{\mathbf{0}\},$$

  where the dimension of the vector $\mathbf{x}$ corresponds to the dimension of the matrix A. It follows that all eigenvalues are positive.
- The matrix A is diagonally dominant, i.e., it is

  $$|a_{ii}| \geq \sum_{j \neq i} |a_{ij}| \quad \forall\, i,$$

  and there is at least one index for which the equal sign is not true. For the considered problems, the upper sign applies for all nodes or degrees of freedom which are close to the boundary.

It is well known from the course on iterative methods for sparse large linear systems of equations, Numerical Mathematics II, that these properties are favorable. In fact, also for multigrid methods, the state of the art is that most of the analysis is known for systems with symmetric positive definite matrices, or matrices which are only slight perturbations of such matrices. However, in practice, multigrid methods often work very well also for the solution of systems with other matrices.

Even if the properties given above are favorable, the condition of the matrices might be large. A direct calculation reveals (this was an exercise problem in Numerical Mathematics II) that in one dimension, the eigenvalues of the finite element matrix A are

$$\lambda_i = \frac{4}{h}\sin^2\left(\frac{i\pi}{2N}\right), \quad i = 1,\ldots,N-1,$$

and the corresponding eigenvectors $\mathbf{v}_i = (v_{i,1},\ldots,v_{i,N-1})^T$ with

$$v_{ij} = \sin\left(\frac{ij\pi}{N}\right), \quad i,j = 1,\ldots,N-1.$$

Then, a direct calculation, using a theorem for trigonometric functions and a Taylor expansion, shows for the spectral condition number

$$\begin{aligned}\kappa_2(A) &= \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)} = \frac{\sin^2\left(\frac{(N-1)\pi}{2N}\right)}{\sin^2\left(\frac{\pi}{2N}\right)} = \frac{\sin^2\left((1-h)\frac{\pi}{2}\right)}{\sin^2\left(h\frac{\pi}{2}\right)}\\ &= \left(\frac{\sin\left(\frac{\pi}{2}\right)\cos\left(h\frac{\pi}{2}\right)-\cos\left(\frac{\pi}{2}\right)\sin\left(h\frac{\pi}{2}\right)}{\sin\left(h\frac{\pi}{2}\right)}\right)^2 = \left(\frac{\cos\left(h\frac{\pi}{2}\right)}{\sin\left(h\frac{\pi}{2}\right)}\right)^2\\ &= \cot^2\left(h\frac{\pi}{2}\right) = \left(\frac{2\pi}{h}-\mathcal{O}(h)\right)^2 = \mathcal{O}\left(h^{-2}\right).\end{aligned}$$

Also in higher dimensions, the condition number is $\kappa_2(A) = \mathcal{O}\left(h^{-2}\right)$. □

Example 2.5 *Behavior of iterative methods for the Poisson equation.* Consider the Poisson equation (2.3) with $f = 1$ for all $\mathbf{x} \in \Omega$ and the P_1 finite element discretization of this problem on meshes with different fineness.

Table 2.1 gives the number of iterations and the computing times for different solvers applied to the solution of this problem. The simulations were performed with the research code MooNMD, John and Matthies (2004). The SSOR method and the conjugate gradient method (CG) are already known from Numerical Mathematics II. For these methods, not the system $A\mathbf{u} = \mathbf{f}$ was solved, but the system

$$D^{-1}A\mathbf{u} = D^{-1}\mathbf{f},$$

where D is the diagonal of A. It is known from Numerical Mathematics II that the number of iterations for SSOR can be estimated to be proportional to the condition number of the matrix and the number of iterations for CG to be proportional to the square root of the condition number. If $\kappa_2(D^{-1}A) < \kappa_2(A)$, then the estimates become better. As comparison, the number of iterations with a multigrid method as solver and with a multigrid method as preconditioner within a flexible general minimized residual (GMRES) method are presented. Finally, the computing times for the application of the sparse direct solver UMFPACK, Davis (2004), are given. UMFPACK is the solver behind the backslash command in MATLAB.

Table 2.1: Example 2.5. Number of iterations and computing times (13/10/11 on a HP BL460c Gen8 2xXeon, Eight-Core 2700MHz). The number of degrees of freedom (d.o.f.) includes the Dirichlet values.

level	h	d.o.f.	SSOR		PCG		MG		FGMRES+MG		UMFPACK	
			ite	time	ite	time	ite	time	ite	time	ite	time
1	1/4	25	49	0	3	0	11	0	6	0	1	0
2	1/8	81	164	0	9	0	13	0	8	0	1	0
3	1/16	289	543	0	31	0	13	0	8	0	1	0
4	1/32	1089	2065	0.07	66	0.01	14	0.03	8	0.01	1	0.01
5	1/64	4225	7998	0.92	132	0.02	14	0.11	8	0.03	1	0.03
6	1/128	16641	31054	14.61	263	0.16	13	0.35	8	0.21	1	0.12
7	1/256	66049	> 100000		524	1.79	13	1.55	8	1.06	1	0.75
8	1/512	263169			1038	16.55	12	6.09	8	3.90	1	5.40
9	1/1024	1050625			1986	127.76	12	27.46	7	18.32	1	46.46
10	1/2048	4198401			3944	1041.68	12	111.03	7	68.38		
factor $\approx$		4	4	16	2	8	1	4	1	4	1	

The number of floating point operations per iteration is for all iterative methods proportional to the number of degrees of freedom. One gets for the complete number of operations the values from Table 2.2. One can observe that the estimate for the number of iterations is sharp for PCG. For the multigrid approaches, the total number of operations is proportional to the number of unknowns. Since in the solution of a linear system of equations, each unknown has to be considered at least once, the total number of operations is asymptotically optimal for multigrid methods.

Table 2.2: Example 2.5. Number of floating point operations, where n is the number of degrees of freedom.

method	op./iter.	no. of iterations	total no. of operations
SSOR	$\mathcal{O}(n)$	$\mathcal{O}(\kappa_2(A)) = \mathcal{O}(h^{-2}) = \mathcal{O}(n^2)$	$\mathcal{O}(n^3)$
PCG	$\mathcal{O}(n)$	$\mathcal{O}\left(\sqrt{\kappa_2(A)}\right) = \mathcal{O}(h^{-1}) = \mathcal{O}(n)$	$\mathcal{O}(n^2)$
MG	$\mathcal{O}(n)$	$\mathcal{O}(1)$	$\mathcal{O}(n)$

In addition, it can be seen that it is even more efficient to use the multigrid method as a preconditioner in a Krylov subspace method than as a solver. One has to use here the flexible GMRES method since the preconditioner is not a fixed matrix but a method. That means, the preconditioner might change slightly from iteration to iteration. The flexible GMRES method can cope with this difficulty.

The development of sparse direct solvers has shown remarkable progress in the last couple of years. One can observe that for the model problem, the direct solver is best for small and medium sized problems, up to about 100000 degrees of freedom. But for large problems, good iterative methods are still better. On the fine grid, UMFPACK is not able to solve the problem because there is an internal memory limitation in this program. □