APPENDIX A

Numerical linear algebra

In this part we want to give a quick review on solution methods for systems of linear equations. Of course, our attention is focussed on systems which arise from the discretization of linear elliptic PDEs. The existing algorithms for the solution of

$$(A.1)\qquad Ax = b, \qquad A \in \mathbb{R}^{m\times m}, \quad x, b \in \mathbb{R}^m, \quad m \in \mathbb{N}$$

mainly fall into two categories: *direct methods* which compute the exact answer to (A.1) after a finite number of steps (if we neglect roundoff errors) and *iterative methods* which do not produce the exact answer after a finite number of steps (in general) but decrease the error by some fraction in each iteration.

A.1. Direct methods

A typical example in this category is the *Gaussian elimination*. The main idea of the algorithm is to split the matrix A into the product of a lower triangular matrix L and an upper triangular matrix U (therefore the algorithm is also called LU factorization). After this splitting has been accomplished, the original problem (A.1) is easily solved in two steps: first, $L^{-1}b$ is calculated which is simple since L is a lower triangular matrix. Then U^{-1} is applied to the result of step one which is again simple due to the triangular structure. The numerical cost for the final steps is $\mathcal{O}(m^2)$ and thus is dominated by the cost for the splitting which is $\mathcal{O}(m^3)$. In general, the algorithm can be applied to A only after a suitable reorganization of rows and/or columns (so called *pivoting*). This extra effort is not necessary if A is symmetric and positive definite (spd). In this special case, Gaussian elimination is essentially equivalent to *Cholesky decomposition* where A factors into LL^T with a lower triangular matrix L. In standard form, both Gaussian elimination and Cholesky decomposition do not take care of a possible sparse structure of A. (However, there are modifications, so called *direct sparse solvers*, which try to do that. See [**7**] for references.) If (A.1) originates in the discretization of a PDE, we easily get $m > 10^6$ with only very few non zero elements per line (depending on the size of the stencil). Hence, there are several arguments why direct solvers are not the methods of choice:

- standard solvers need memory of order m^2,
- the computation time is in general of order m^3,
- we get an exact solution to a wrong (or rather discretized) problem.

A.2. Iterative methods

With iterative methods, we compute the first members of a sequence $(x_k)_{k\in\mathbb{N}}$ of approximations to the solution of (A.1). Formally, we write

$$x_k = \Phi^{(k)}(x_{k-1}, \dots, x_0)$$

and require

a) $(x_k)_{k\in\mathbb{N}}$ should converge quickly,
b) $\Phi^{(k)}$ should be simple to evaluate (i.e. not more expensive than matrix vector multiplication which are of the order m for sparse matrices).

The basic idea of classical iterative methods is to transform the original problem (A.1) into a fixed point problem. One way to do this is

$$\text{(A.2)} \qquad Ax = b \qquad \Longleftrightarrow \qquad x = (I - A)x + b = \Phi(x).$$

Then, the fixed point iteration $x_{k+1} = \Phi(x_k)$ converges under certain conditions on the matrix A (if $I - A$ is a contraction). Certainly, requirement b) is satisfied by this approach. As far as property a) is concerned, we have the following fundamental

Theorem A.1. *The fixed point iteration $x_{k+1} = Gx_k + c$ converges for every initial vector x_0 if and only if $\rho(G) < 1$, where $\rho(G)$ is the spectral radius of G, i.e. $\rho(g) = \max\{|\lambda_i| : \lambda_i$ eigenvalue of $G\}$.*

Proof: (for symmetric G)
There are rotation matrices $R \in \mathbb{R}^{m\times m}$ such that

$$RGR^T = \Lambda, \qquad \Lambda = \operatorname{diag}(\lambda_1, \dots, \lambda_m), \qquad |\lambda_i| < 1.$$

The sequence $y_k = Rx_k$ then satisfies

$$y_{k+1} = \Lambda y_k + d, \qquad d = Rc$$

or equivalently,

$$y_{k+1} = \Lambda^{k+1} y_0 + \sum_{i=0}^{k} \Lambda^i d.$$

Since $|\lambda_i| < 1$ we thus get

$$y_k \xrightarrow[k\to\infty]{} \operatorname{diag}\left(\frac{1}{1-\lambda_1}, \dots, \frac{1}{1-\lambda_m}\right) d = (I - \Lambda)^{-1} Rc.$$

multiplying by R^T from the right finally yields

$$x_k \xrightarrow[k\to\infty]{} (I - R^T \Lambda R)^{-1} c = (I - G)^{-1} c.$$

■

To check that $\rho(G) < 1$, it suffices to have $\|G\| < 1$ in some matrix norm $\|\cdot\|$ which is related to a vector norm. In such a norm it is then easy to see that

$$\|x_k - x\| \leq \|G\|^k \|x_0 - x\|.$$

Thus, the smaller $\|G\|$, the faster the iterative scheme converges which gives some result concerning requirement a).

Applying these considerations to (A.2) (which leads to the so called *Richardson iteration*) we see that the scheme works only with special matrices A. If, for example, A is spd, the spectrum of A is located on the positive real line in $\mathbb{C}$. Consequently, $G = I - A$ has its spectrum to the left of 1 so that $\rho(G) < 1$ is only satisfied if $\rho(A) < 2$. This is of course a severe restriction (just think of the matrix $\bar{\mathcal{L}}_h$ obtained from the 1D Dirichlet Poisson problem in Section 2.7.1 where the maximal eigenvalue grows like $\mathcal{O}(m^2)$ (see Exercise 2.11)). Hence, (A.2) is not very helpful in our case and we have to think of a different fixed point formulation. The basic idea is to modify the original problem (A.1) by multiplying with the inverse of a matrix Q (so called *preconditioning*)

$$Ax = b \qquad \Longleftrightarrow \qquad Q^{-1}Ax = Q^{-1}b.$$

With the same idea as in (A.2) this leads to the fixed point formulation

$$Ax = b \qquad \Longleftrightarrow \qquad x = \underbrace{(I - Q^{-1}A)}_{G}\, x + \underbrace{Q^{-1}b}_{c}. \tag{A.3}$$

Compared to $Q = I$, which is the original formulation (A.2), the case $Q = A$ is at the other extreme. Indeed, if $Q = A$ then $G = 0$ and the first step of the iterative scheme yields the solution. However, to set up the scheme, A^{-1} has to be known already so that requirement b) is certainly violated: the evaluation of Φ is as hard as solving the problem itself. As so often, the truth lies somewhere in the middle which means in this case that we have to choose $Q \approx A$, so that $\rho(G) = \rho(I - Q^{-1}A) \approx 0$ but with a simple rule how to get Q^{-1}, or more precisely, how to solve (the simplified) problems of the form $Qx = b$. In other words, preconditioning amounts to having already a good guess about the solution of the problem. In the following, we list some ideas how to derive preconditioners (taken from [**16**]):

- *Diagonal scaling or Jacobi.* A very important preconditioner is given by $Q = \mathrm{diag}(A)$ provided this matrix is invertible. Similarly, one can take any invertible diagonal matrix but the entries should be related to the problem to get an improvement of the convergence behavior.
- *Incomplete Cholesky or LU factorization.* The idea is to perform similar transformations on A as in Cholesky or, for the non symmetric case, LU factorization, but to keep only those entries in the triangular factors which have nonzero counterparts in the matrix A.
- *Coarse–grid approximation.* While the first two examples work independent of the underlying problem, the coarse–grid approximation is designed for matrices which result from the discretization of a partial differential or integral equation. If the discretization which leads to A is very fine, the analogous discretization on a coarser grid may lead to a small system that is comparably easy to solve. Thus, a powerful

preconditioner may be obtained of the following schematic form:

$$Q = \langle \text{transfer to fine grid} \rangle \circ A_{\text{coarse}} \circ \langle \text{transfer to coarse grid} \rangle \, .$$

Applying this approach to a hierarchy of grids is one of the ideas in *multigrid schemes.*

- *Local approximation.* In a coarse–grid approximation, large scale structure of the problem is taken into account while ignoring some of the finer structure. If A contains coupling between near as well as far elements it can also be reasonable to consider the operator Q analogous to A but with the longer–range interactions omitted. In the simples case of this kind, Q may consist simply of a few of the diagonals of A near the main diagonal.
- *Block preconditioners and domain decomposition.* If a matrix has a block structure (say block tridiagonal), the Jacobi preconditioner can be generalized to block–diagonal or block–Jacobi form. This is another kind of local approximation, in that local effects within certain components are considered while connections to other components are ignored. Ideas of that kind have lately been generalized in the field of domain decomposition, in which solvers for certain subdomains of a problem are composed in flexible ways to form preconditioners for the global problem.
- *Low–order discretization.* If a differential or integral method is discretized by a higher order method one typically finds a gain in accuracy but stencils become bigger and thus A is less sparse. A lower approximation of the same problem, with its sparser matrix, may be an effective preconditioner.
- *Constant coefficient or symmetric approximation.* For a problem with variable coefficients, a constant–coefficient approximation implemented by a fast solver may be a good preconditioner. Analogously, if a differential equation is not self–adjoint but is close in some sense to a self–adjoint equation that can be solved more easily, then the latter may sometimes serve as preconditioner.
- *Splitting of a multi–term operator.* Many applications involve combinations of physical effects which can manifest itself in the discretization as a splitting of A into natural contributions $A = A_1 + \cdots + A_m$. If a partial sum of the matrices A_i is easily invertible, it may serve as a good preconditioner.
- *Dimensional splitting or ADI.* Another kind of splitting takes advantage of the fact that an operator such as the Laplacian in two or three dimensions is composed of analogous operators in each of the directions separately. This idea may form the basis for a preconditioner, and in one form goes by the name *ADI* or *alternating direction implicit* methods.
- *One step of a classical iterative method.* This is another idea which is used in multigrid schemes. Some of the classical iterative methods are presented below.

We now briefly describe some classical iterative methods. A detailed investigation can be found in [**3**]. For the case of a discretized Poisson problem on a square, the methods are presented in [**6**]. In the *Richardson iteration*, no preconditioning is performed. If A is spd, then the eigenvalues of $G = I - A$ are located left of $\lambda = 1$. The requirement on the spectral radius of G given in Theorem A.1, leads to the restriction $|1 - \rho(A)| < 1$. In other words, the Richardson iteration converges for spd matrices A provided $\rho(A) < 2$. For the next two methods, we split the matrix A into a lower triangle L and an upper triangle U (both with zeros in the diagonal) and the diagonal part $D = \operatorname{diag}(A)$ of A

$$A = L + D + U.$$

If we take D as preconditioner, we find

$$G = I - D^{-1}A = I - D^{-1}(L + D + U) = -D^{-1}(L + U)$$

giving rise to the *Jacobi iteration* respectively *single step method*. Under the assumption of strictly diagonal dominance,

$$|A_{ii}| > \sum_{j \neq i} |A_{ij}|, \qquad i = 1, \ldots, m$$

the Jacobi iteration converges. (Note that for strictly diagonal dominant matrices A we have $\|G\|_\infty = \|D^{-1}(L + U)\|_\infty < 1$.) Finally, if we take as preconditioner the lower triangle $Q = L + D$ (which is easy to invert due to the triangular structure) we obtain the so called *Gauß–Seidel* or *multi step method*. The matrix G is now of the form

$$G = I - (D + L)^{-1}(L + D + U) = -(D + L)^{-1}U$$

It can be shown that the Gauß–Seidel method converges for any symmetric and positive definite matrix A.

A general approach to construct a one–parameter family of iteration methods based on a given method is *relaxation*. Given an iteration method described by G and c, we take as new value x_{k+1} a linear combination of the old step x_k and original correction $Gx_k + c$

$$x_{k+1} = \omega(Gx_k + c) + (1 - \omega)x_k = \underbrace{(\omega G + (1 - \omega)I)}_{G_\omega} x_k + \omega c.$$

The advantage of this approach is that we can now determine the optimal scheme in the family by picking ω in such a way that $\rho(G_\omega)$ becomes minimal. If the corresponding ω is less than one, we speak of *under–relaxation*, if it is larger than one the notion is *over–relaxation*. Typically, the original scheme can be improved by relaxation, i.e. the optimal parameter ω will not be one. For an spd matrix A, the optimal relaxation of the Richardson iteration is obtained with

$$\omega = \frac{2}{\lambda_{\max}(A) + \lambda_{\min}(A)}$$

which yields the spectral radius

$$\rho(G_\omega) = \frac{\lambda_{\max}(A) - \lambda_{\min}(A)}{\lambda_{\max}(A) + \lambda_{\min}(A)} < 1.$$

Consequently, the relaxed scheme converges also in cases $\lambda_{\max}(A) \geq 2$ in which the original Richardson iteration fails.

For the special case $A = \mathcal{L}_h$, where A results from the discretization of an elliptic operator $\mathcal{L}$, we can also use positivity properties to show the convergence of iterative methods. As in Exercise 2.12, we introduce the notion *M–matrix* for matrices with a sign structure like $\mathcal{L}_h$.

Definition A.2. *$A \in \mathbb{R}^{m \times m}$ is called M–matrix, if $A_{ij} \leq 0$ for $i \neq j$, if A is regular and if A^{-1} is positive.*

It can be shown (see [**3**]) that both Jacobi and Gauß–Seidel iterations and their relaxed versions with $\omega \in (0,1)$ converge for any start vector. However, under– relaxation is not useful, because convergence is slowed down.

Generally the most efficient linear, first order iteration scheme is SOR (which expands to *successive over–relaxation*). It is nothing else than Gauß–Seidel plus relaxation (for $\omega < 1$, the method is still called successive over–relaxation although $\omega < 1$ is actually under–relaxation). For general spd matrices A, we have the result that SOR converges for any $\omega \in (0,2)$.