Chapter IV
The Conjugate Gradient Method

The discretization of boundary-value problems leads to very large systems of equations which often involve several thousand unknowns. The systems are particularly large for three-dimensional problems and for problems of higher order. Often the bandwidth of the matrices is so large that the classical Gauss elimination algorithm and its modern variants are not efficient methods. This suggests that even for linear problems, we should use iterative methods.

Iterative methods first became popular at the end of the fifties, primarily as a means for solving large problems using computers with a small memory. The methods developed then are no longer competitive, but they still provide useful ingredients for modern iterative methods, and so we review them in §1. The bulk of this chapter is devoted to the conjugate gradient method which is particularly useful for the solution of variational problems and saddle point problems. Since the CG methods discussed here can be applied to a broad spectrum of problems, they are competitive with the still faster multigrid methods to be discussed later (whose implementation is generally more complicated and requires more individual programming).

We begin by classifying problems according to the number n of unknowns:

1. *Small problems*: For linear problems we can use a direct method. For nonlinear problems (e.g., using the Newton method), all elements of the Jacobi matrices should be computed (at least approximately).
2. *Midsized problems*: If the matrices are sparse, we should make use of this fact. For nonlinear problems (e.g., for quasi-Newton methods), the Jacobi matrices should be approximated. Iterative methods can still be used even when the number of steps in the iteration exceeds n.
3. *Very large problems*: Here the only choice is to use iterative methods which require fewer than n steps to compute a solution.

For very large problems, we have to deal with completely different aspects of the method of conjugate gradients as compared with midsized problems. For example, the fact that in exact arithmetic CG methods produce a solution in n steps plays no role for very large problems. In this case it is more important that the accuracy of the approximate solution depends on the condition number of the matrix.

§ 1. Classical Iterative Methods for Solving Linear Systems

For a small step size h, the finite element method leads to very large systems of equations. Although the associated matrices are sparse, unfortunately their structure is such that after a few (approximately $\sqrt{n}$) steps of Gauss elimination, the computational effort will grow significantly because many zero elements will be replaced by nonzero ones.

This is the reason why iterative methods were studied in the fifties. The methods introduced then, which we shall call *classical iterative methods*, converge very slowly. Nowadays they are rarely used as stand-alone iterative methods. Nevertheless, they have not lost their importance. Indeed they are often used in conjunction with modern iterative methods, for example in the CG method for preconditioning, and in the multigrid method for smoothing.

We now review classical iterative methods. For more detailed treatments, see the monographs of Varga [1962], Young [1971], and Hackbusch [1991].

Stationary Linear Processes

Many iterative methods for the solution of the system $Ax = b$ are based on a decomposition of the matrix

$$A = M - N.$$

Here M is assumed to be an easily invertible matrix, and the given system can be rewritten in the form

$$Mx = Nx + b.$$

This leads to the iteration

$$Mx^{k+1} = Nx^k + b,$$

or equivalently $x^{k+1} = M^{-1}(Nx^k + b)$, or

$$x^{k+1} = x^k + M^{-1}(b - Ax^k), \quad k = 0, 1, 2, \ldots . \tag{1.1}$$

Clearly, (1.1) is an iteration of the form

$$x^{k+1} = Gx^k + d \tag{1.2}$$

with $G = M^{-1}N = I - M^{-1}A$, $d = M^{-1}b$.

The solution x^* of the equation $Ax = b$ is a *fixed point* of the process (1.2), i.e.,

$$x^* = Gx^* + d.$$

Subtracting the last two equations gives

$$x^{k+1} - x^* = G(x^k - x^*),$$

and by induction we see that

$$x^k - x^* = G^k(x^0 - x^*). \tag{1.3}$$

In view of (1.1) we may consider M^{-1} as an *approximate inverse* of A. The associated error reduction per step is described by the matrix $G = I - M^{-1}A$.

The iteration (1.2) is called *convergent* if for every arbitrary $x^0 \in \mathbb{R}^n$, we have $\lim_{k\to\infty} x^k = x^*$. By (1.3), this is equivalent to

$$\lim_{k\to\infty} G^k = 0. \tag{1.4}$$

1.1 Definition. Let A be a (real or complex) $n \times n$ matrix with eigenvalues λ_i, $i = 1, 2, \ldots, m$, $m \le n$. Then

$$\rho(A) = \max_{1\le i\le m} |\lambda_i|$$

is called the *spectral radius* of A.

The following characterization of (1.4) via the spectral radius is well known in linear algebra (cf. Varga [1962], p. 64, or Hackbusch [1991]).

1.2 Theorem. *Let G be an $n \times n$ matrix. Then the following assertions are equivalent:*

(i) *The iteration (1.2) converges for every $x^0 \in \mathbb{R}^n$.*
(ii) $\lim_{k\to\infty} G^k = 0$.
(iii) $\rho(G) < 1$.

The spectral radius also provides a quantitative measure of the rate of convergence. Note that $\|x^k - x^*\| \le \|G^k\|\,\|x^0 - x^*\|$, and thus

$$\frac{\|x^k - x^*\|}{\|x^0 - x^*\|} \le \|G^k\|. \tag{1.5}$$

1.3 Theorem. *For every $n \times n$ matrix G,*

$$\lim_{k\to\infty} \|G^k\|^{1/k} = \rho(G). \tag{1.6}$$

Proof. Let $\varepsilon > 0$, $\rho = \rho(G)$, and

$$B = \frac{1}{\rho + \varepsilon} G.$$

Then $\rho(B) = \rho/(\rho+\varepsilon) < 1$, and $\lim_{k\to\infty} B^k = 0$. Hence, $\sup_{k\geq 0} \|B^k\| \leq a < \infty$. This implies $\|G^k\| \leq a(\rho+\varepsilon)^k$ and $\lim_{k\to\infty} \|G^k\|^{1/k} \leq \rho+\varepsilon$. Thus, the left-hand side of (1.6) is at most $\rho(G)$.

On the other hand, the left-hand side of (1.6) cannot be smaller than $\rho(G)$ since G has an eigenvalue of modulus ρ. □

Here we have made use of the usual notation. In particular, M^{-1} can be thought of as an approximate inverse of A. In the framework of the CG method, the matrix M is called a *preconditioner*, and is usually denoted by the symbol C.

The Jacobi and Gauss–Seidel Methods

One way to get an iterative method is to start with the decomposition

$$A = D - L - U.$$

Here D is diagonal, L lower-triangular, and U upper-triangular:

$$D_{ik} = \begin{cases} a_{ik} & \text{if } i = k, \\ 0 & \text{otherwise,} \end{cases}$$

$$L_{ik} = \begin{cases} -a_{ik} & \text{if } i > k, \\ 0 & \text{otherwise,} \end{cases} \qquad U_{ik} = \begin{cases} -a_{ik} & \text{if } i < k, \\ 0 & \text{otherwise.} \end{cases}$$

The *Jacobi method* corresponds to the iteration

$$Dx^{k+1} = (L + U)x^k + b, \tag{1.7}$$

which can be written componentwise as

$$a_{ii}x_i^{k+1} = -\sum_{j\neq i} a_{ij}x_j^k + b_i, \quad i = 1, 2, \ldots, n. \tag{1.8}$$

The associated iteration matrix is $G = G_J = D^{-1}(L + U)$, and thus

$$G_{ik} = \begin{cases} -\dfrac{a_{ik}}{a_{ii}} & \text{for } k \neq i, \\ 0 & \text{otherwise.} \end{cases}$$

Normally we compute the components of the new vector x^{k+1} sequentially, i.e., we compute $x_1^{k+1}, x_2^{k+1}, \ldots, x_n^{k+1}$. During the computation of x_i^{k+1}, we already have the components of the new vector up to the index $j = i - 1$. If we use this information, we are led to *the method of successive relaxation*, also called the *Gauss–Seidel method*:

$$a_{ii}x_i^{k+1} = -\sum_{j<i} a_{ij}x_j^{k+1} - \sum_{j>i} a_{ij}x_j^k + b_i. \tag{1.9}$$

Here the decomposition in matrix-vector form reads

$$Dx^{k+1} = Lx^{k+1} + Ux^k + b,$$

and the associated iteration matrix is $G = G_{GS} = (D - L)^{-1}U$.

1.4 Example. Consider the matrix

$$A = \begin{pmatrix} 1 & 0.1 \\ 4 & 1 \end{pmatrix}.$$

Then the iteration matrix for the Jacobi method is

$$G_J = \begin{pmatrix} 0 & -0.1 \\ -4 & 0 \end{pmatrix},$$

and the matrix for the Gauss–Seidel method is

$$G_{GS} = -\begin{pmatrix} 1 & 0 \\ 4 & 1 \end{pmatrix}^{-1}\begin{pmatrix} 0 & 0.1 \\ 0 & 0 \end{pmatrix} = -\begin{pmatrix} 1 & 0 \\ -4 & 1 \end{pmatrix}\begin{pmatrix} 0 & 0.1 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & -0.1 \\ 0 & 0.4 \end{pmatrix}.$$

A simple calculation shows that $\rho(G_{GS}) = \rho^2(G_J) = 0.4$, and hence both iterations converge.

The convergence of the above methods can often be established with the help of the following simple concept.

1.5 Definition. An $n \times n$ matrix A is said to be *strictly diagonally dominant* provided that

$$|a_{ii}| > \sum_{j\neq i} |a_{ij}|, \quad 1 \le i \le n.$$

A is said to be *diagonally dominant* if

$$|a_{ii}| \ge \sum_{j\neq i} |a_{ij}|, \quad 1 \le i \le n,$$

and strict inequality holds for at least one i.

1.6 Definition. An $n \times n$ matrix A is called *reducible* if there exists a proper subset $J \subset \{1, 2, \ldots, n\}$ such that

$$a_{ij} = 0 \quad \text{for } i \in J, \quad j \notin J.$$

After reordering, a reducible matrix can be written in block form as

$$\begin{pmatrix} A_{11} & 0 \\ A_{21} & A_{22} \end{pmatrix}.$$

In this case the linear system $Ax = b$ splits into two smaller ones. Thus, there is no loss of generality in restricting our attention to irreducible matrices.

1.7 Diagonal Dominance. *Suppose the $n \times n$ matrix A is diagonally dominant and irreducible. Then both the Jacobi and Gauss–Seidel methods converge.*

Supplement: If A is strictly diagonally dominant, then the hypothesis that A be irreducible is not needed, and the spectral radius of the iteration matrix is

$$\rho \le \max_{1 \le i \le n} \frac{1}{|a_{ii}|} \sum_{j \ne i} |a_{ik}| < 1.$$

Proof. Let $x \ne 0$. For the Jacobi method the iteration matrix is $G = D^{-1}(L + U)$, and with the maximum norm we have

$$\begin{aligned} |(Gx)_i| &\le \frac{1}{|a_{ii}|} \sum_{j \ne i} |a_{ij}||x_j| \\ &\le \frac{1}{|a_{ii}|} \sum_{j \ne i} |a_{ij}| \cdot \|x\|_\infty \le \|x\|_\infty. \end{aligned} \tag{1.10}$$

In particular, $\|Gx\|_\infty \le \beta \|x\|_\infty$ with $\beta < 1$ if the stronger condition holds. Thus, $\|G^k x\|_\infty \le \beta^k \|x\|_\infty$, and we have established the convergence.

The diagonal dominance implies $\|Gx\|_\infty \le \|x\|_\infty$, and so $\rho(G) \le 1$. Suppose that $\rho = 1$. Then there exists a vector x with $\|x\|_\infty = \|Gx\|_\infty = 1$, and equality holds in all components of (1.10). Set

$$J = \{j \in \mathbb{N};\ 1 \le j \le n,\ |x_j| = 1\}.$$

In order for equality to hold in each component of (1.10), we must have

$$a_{ij} = 0 \quad \text{for } i \in J,\ j \notin J.$$

Moreover, by hypothesis $|a_{ii}| > \sum_{j \ne i} |a_{ij}|$ for some $i = i_0$, and it follows that $|(Gx)_{i_0}| < 1$. By definition, $i_0 \notin J$, and J is a proper subset of $\{1, 2 \ldots, n\}$, contradicting the assumption that A is irreducible.

For the Gauss–Seidel method, the iteration matrix G is given implicitly by

$$Gx = D^{-1}(LGx + Ux).$$

First we prove that $|(Gx)_i| \le \|x\|_\infty$ for $i = 1, 2, \ldots, n$. By induction on i we can verify that

$$\begin{aligned} |(Gx)_i| &\le \frac{1}{|a_{ii}|}\{\sum_{j<i} |a_{ij}|\ |(Gx)_j| + \sum_{j>i} |a_{ij}|\ |x_j|\} \\ &\le \frac{1}{|a_{ii}|} \sum_{j\neq i} |a_{ij}|\ \|x\|_\infty \le \|x\|_\infty. \end{aligned}$$

The rest of the proof is similar to that for the Jacobi method. □

The Model Problem

The system of equations which arises in using the five-point stencil (II.4.9) to discretize the Poisson equation on a square of lenth L has been frequently studied. With $h = L/m$ and $(m-1) \times (m-1)$ interior points, we have

$$4x_{i,j} - x_{i+1,j} - x_{i-1,j} - x_{i,j+1} - x_{i,j-1} = b_{i,j}, \quad 1 \le i, j \le m-1, \tag{1.11}$$

where variables with index 0 or m are assumed to be zero. Clearly, the associated matrix is diagonally dominant, and since the system is irreducible, this suffices for the convergence.

Here we have the special case where the diagonal D is a multiple of the unit matrix. Hence, the iteration matrix for the Jacobi method can be obtained immediately from the original matrix:

$$G = I - \frac{1}{4}A.$$

In particular, A and G have the same eigenvectors $z^{k,\ell}$, as can be seen by simple substitution using elementary properties of trigonometric functions:

$$\left.\begin{aligned} (z^{k,\ell})_{i,j} &= \sin\frac{ik\pi}{m}\sin\frac{j\ell\pi}{m}, \\ Az^{k,\ell} &= (4 - 2\cos\frac{k\pi}{m} - 2\cos\frac{\ell\pi}{m})\, z^{k,\ell}, \\ Gz^{k,\ell} &= (\frac{1}{2}\cos\frac{k\pi}{m} + \frac{1}{2}\cos\frac{\ell\pi}{m})\, z^{k,\ell}, \end{aligned}\right\} \quad 1 \le k, \ell \le m-1. \tag{1.12}$$

The eigenvalue of G with largest modulus corresponds to $k = \ell = 1$, and we have

$$\rho(G) = \frac{1}{2}(1 + \cos\frac{\pi}{m}) = 1 - \frac{\pi^2}{4m^2} + \mathcal{O}(m^{-4}). \tag{1.13}$$

For large m, the quantity $\rho(G)$ tends to 1, and so the rate of convergence is very poor for grids with small mesh size h.

The situation is slightly better for the Gauss–Seidel method, but even then we have $\rho(G) = 1 - \mathcal{O}(m^{-2})$.

Overrelaxation

Overrelaxation was the first acceleration technique. In updating via (1.9), the change in the value of x_i is immediately multiplied by a factor ω:

$$a_{ii}x_i^{k+1} = \omega[-\sum_{j<i} a_{ij}x_j^{k+1} - \sum_{j>i} a_{ij}x_j^k + b_i] - (\omega - 1)a_{ii}x_i^k, \tag{1.14}$$

or in matrix form

$$Dx^{k+1} = \omega[Lx^{k+1} + Ux^k + b] - (\omega - 1)Dx^k. \tag{1.15}$$

If the factor ω is larger than 1, this is called *overrelaxation*, otherwise it is called *underrelaxation.* The process of *successive overrelaxation* ($\omega > 1$) is referred to as the SOR method.

Let A be a symmetric positive definite matrix. Then the Gauss–Seidel method (with or without relaxation) can be viewed as the process of minimizing the expression

$$f(x) = \frac{1}{2}x'Ax - b'x. \tag{1.16}$$

Clearly,

$$\frac{\partial f}{\partial x_i} = (Ax - b)_i.$$

If we fix $x_1, x_2, \ldots, x_{i-1}, x_{i+1} \ldots, x_n$ and vary only x_i, then a minimum of $f(x)$ as a function of x_i is characterized by the equation $\partial f/\partial x_i = 0$, i.e., by $(Ax-b)_i = 0$. This gives the formula (1.9). In particular, the improvement in the value of f after one step is given by $\frac{1}{2}a_{ii}(x_i^{k+1} - x_i^k)^2$. The value of f can be reduced using relaxation for any choice of $\omega \in (0, 2)$. The improvement is then $\frac{1}{2}\omega(2 - \omega)$ $[(x_i^{k+1} - x_i^k)/\omega]^2$. These quantities also arise in the proof of the following theorem.

1.8 Ostrowski–Reich Theorem. *Suppose A is a symmetric $n \times n$ matrix with positive diagonal elements, and let $0 < \omega < 2$. Then the SOR method (1.14) converges if and only if A is positive definite.*

Proof. After an elementary manipulation, it follows from (1.15) that

$$(1 - \frac{\omega}{2})D(x^{k+1} - x^k) = \omega[(L - \frac{1}{2}D)x^{k+1} + (U - \frac{1}{2}D)x^k + b].$$

Multiplying on the left by $(x^{k+1} - x^k)$ and taking into account $y'Lz = z'Uy$, we have

$$\begin{aligned}(1 - \frac{\omega}{2})(x^{k+1} - x^k)'D(x^{k+1} - x^k) \\ &= \frac{\omega}{2}[-x^{k+1'}Ax^{k+1} + x^{k'}Ax^k + 2b'(x^{k+1} - x^k)] \\ &= \omega[f(x^k) - f(x^{k+1})].\end{aligned} \tag{1.17}$$

Since the convergence only depends on the spectral radius of the iteration matrix, without loss of generality we can assume that $b = 0$. Moreover, observe that $x^{k+1} \neq x^k$ since $Ax^k - b \neq 0$.

(1) Let A be positive definite. For every x^0 in the unit sphere

$$S^{n-1} := \{x \in \mathbb{R}^n;\ \|x\| = 1\},$$

(1.17) implies

$$\frac{x^{1\prime} A x^1}{x^{0\prime} A x^0} < 1.$$

In view of the continuity of the mapping $x^0 \mapsto x^1$ and the compactness of the sphere S^{n-1}, there exists $\beta < 1$ with

$$\frac{x^{1\prime} A x^1}{x^{0\prime} A x^0} \leq \beta < 1, \tag{1.18}$$

for all $x^0 \in S^{n-1}$. Since the quotient on the left-hand side of (1.18) does not change if x^0 is multiplied by a factor $\neq 0$, (1.18) holds for all $x^0 \neq 0$. It follows by induction that

$$x^{k\prime} A x^k \leq \beta^k x^{0\prime} A x^0,$$

and thus $x^{k\prime} A x^k \to 0$. By the definiteness of A, we have $\lim_{k\to\infty} x^k = 0$.

(2) Let A be indefinite. Without loss of generality, consider the iteration (1.9) for the homogeneous equation with $b = 0$. Then there exists $x^0 \neq 0$ with $\alpha := f(x^0) < 0$. Now (1.17) implies $f(x^k) \leq f(x^{k-1})$ and

$$f(x^k) \leq \alpha < 0, \quad k = 0, 1 \ldots,$$

and we conclude that $x^k \not\to 0$. □

In carrying out the relaxation methods (1.9) and (1.14), the components of the vectors are recomputed in the order from $i = 1$ to $i = n$. Obviously, we can also run through the indices in the reverse order. In this case, the roles of the submatrices L and U are reversed. In the *symmetric* SOR *method*, SSOR method for short, the iteration is performed alternately in the forward and backward directions. Thus, each iteration step consists of two half steps:

$$\begin{aligned} Dx^{k+1/2} &= \omega[Lx^{k+1/2} + Ux^k + b] - (\omega - 1)Dx^k, \\ Dx^{k+1} &= \omega[Lx^{k+1/2} + Ux^{k+1} + b] - (\omega - 1)Dx^{k+1/2}. \end{aligned} \tag{1.19}$$

By the remark following (1.16), we conclude that the assertion of the theorem also holds for the symmetric iteration process.

The SSOR method has two advantages. The computational effort in performing k SSOR cycles is not the same as that for $2k$ cycles of the SOR method, but only to $k + 1/2$ cycles. Moreover, the associated iteration matrix

$$M^{-1} = \omega(2 - \omega)(D - \omega U)^{-1} D (D - \omega L)^{-1} \tag{1.20}$$

in the sense of (1.1) is symmetric; cf. Problem 1.10.

Problems

1.9 Write the SSOR method (1.19) in componentwise form similar to (1.14).

1.10 Verify (1.20). – Hint: Under iteration the matrix A corresponds to the approximate inverse M^{-1}. In particular, $x^1 = M^{-1}b$ for $x^0 = 0$.

1.11 Consider the matrices

$$A_1 = \begin{pmatrix} 1 & 2 & -2 \\ 1 & 1 & 1 \\ 2 & 2 & 1 \end{pmatrix} \qquad \text{and} \qquad A_2 = \frac{1}{2}\begin{pmatrix} 2 & -1 & 1 \\ 2 & 2 & 2 \\ -1 & -1 & 2 \end{pmatrix}.$$

For which of the matrices do the Jacobi method and Gauss–Seidel methods converge?

1.12 Let G be an $n \times n$ matrix with $\lim_{k\to\infty} G^k = 0$. In addition, suppose $\|\cdot\|$ is an arbitrary vector norm on $\mathbb{R}^n$. Show that

$$|||x||| := \sum_{k=0}^{\infty} \|G^k x\|$$

defines a norm on $\mathbb{R}^n$, and that $|||G||| < 1$ for the associated matrix norm. – Give an example to show that $\rho(G) < 1$ does not imply $\|G\| < 1$ for every arbitrary norm.

1.13 If we choose $\omega = 2$ in the SSOR method, then $x^{k+1} = x^k$. How can this be shown without using any formulas?

1.14 Suppose the Jacobi and Gauss–Seidel methods converge for the equation $Ax = b$. In addition, suppose D is a nonsingular diagonal matrix. Do we still get convergence if AD (or DA) is substituted for A?

1.15 A matrix B is called *nonnegative*, written as $B \geq 0$, if all matrix elements are nonnegative. Let $D,\ L,\ U \geq 0$, and suppose the Jacobi method converges for $A = D - L - U$. Show that this implies $A^{-1} \geq 0$.
What is the connection with the discrete maximum principle?

1.16 Let M^{-1} be an approximate inverse of A in the sense of (1.1). Determine the approximate inverse that corresponds to k steps of the iteration.

§ 2. Gradient Methods

In developing iterative methods for the solution of systems of equations associated with a positive definite matrix A, it is very useful to observe that the solution of $Ax = b$ is also the minimum of

$$f(x) = \frac{1}{2}x'Ax - b'x. \tag{2.1}$$

The simplest method for finding a minimum is the *gradient method*. In its classical form it is a stable method, but converges very slowly if the condition number $\kappa(A)$ is large. Unfortunately, this is usually the case for the systems which arise in the discretization of elliptic boundary-value problems. For problems of order $2m$, the condition number typically grows like h^{-2m}.

The so-called *PCG method* (see §§3 and 4) was developed from the gradient method by making two modifications to it, and is one of the most effective system solvers.

The General Gradient Method

For completeness, we formulate the gradient method in a more general form as a method for the minimization of a C^1 function f defined on an open set $M \subset \mathbb{R}^n$.

2.1 Gradient Method with Complete Line Search.
Choose $x_0 \in M$. For $k = 0, 1, 2, \ldots$, perform the following calculations:
1. Determine the direction: Compute the negative gradient

$$d_k = -\nabla f(x_k). \tag{2.2}$$

2. Line search: Find a point $t = \alpha_k$ along the line $\{x_k + td_k : t \geq 0\} \cap M$ where f attains a (local) minimum. Set

$$x_{k+1} = x_k + \alpha_k d_k. \tag{2.3}$$

□

Clearly, the method generates a sequence (x_k) with

$$f(x_0) \geq f(x_1) \geq f(x_2) \geq \cdots,$$

where equality holds only at points where the gradient vanishes.

For the special case of the quadratic function (2.1), we have

$$d_k = b - Ax_k, \tag{2.4}$$

$$\alpha_k = \frac{d_k'd_k}{d_k'Ad_k}. \tag{2.5}$$

2.2 Remark. In practice the line search is done only approximately. There are two possibilities:

(1) Find a point for which the directional derivative is small, say

$$|d_k' \nabla f(x_k + t d_k)| < \frac{1}{4} |d_k' \nabla f(x_k)|.$$

(2) First choose a reasonable step size t, and continue halving it until the corresponding improvement in the function value is at least one quarter of what we would get by linearization, i.e., until

$$f(x_k + t d_k) \leq f(x_k) - \frac{t}{4} |d_k' \nabla f(x_k)|.$$

This version of the gradient method leads to global convergence: *either there is some subsequence which converges to a point $x \in M$ with $\nabla f(x) = 0$, or the sequence tends to the boundary of M*. The latter cannot happen if f is greater than $f(x_0)$ as we approach ∂M (or as $x \to \infty$, respectively).

Gradient Methods and Quadratic Functions

Rather than presenting a general proof of convergence, we instead focus on a more detailed examination of the case of quadratic functions. In this case the rate of convergence is determined by the size of $\kappa(A)$.

As a measure of distance, we use the *energy norm*

$$\|x\|_A := \sqrt{x' A x}. \tag{2.6}$$

If x^* is a solution of the equation $Ax = b$, then as in II.2.4

$$f(x) = f(x^*) + \frac{1}{2} \|x - x^*\|_A^2. \tag{2.7}$$

Now (2.4) and (2.5) imply

$$\begin{aligned} f(x_{k+1}) &= f(x_k + \alpha_k d_k) \\ &= \frac{1}{2} (x_k + \alpha_k d_k)' A (x_k + \alpha_k d_k) - b'(x_k + \alpha_k d_k) \\ &= f(x_k) + \alpha_k d_k' (A x_k - b) + \frac{1}{2} \alpha_k^2 \, d_k' A d_k \\ &= f(x_k) - \frac{1}{2} \frac{(d_k' d_k)^2}{d_k' A d_k}. \end{aligned}$$

Combining this with (2.7), we have

$$\|x_{k+1} - x^*\|_A^2 = \|x_k - x^*\|_A^2 - \frac{(d_k' d_k)^2}{d_k' A d_k}.$$

Since $d_k = -A(x_k - x^*)$, we have $\|x_k - x^*\|_A^2 = (A^{-1}d_k)'A(A^{-1}d_k) = d_k'A^{-1}d_k$ and

$$\|x_{k+1} - x^*\|_A^2 = \|x_k - x^*\|_A^2 \left[1 - \frac{(d_k'd_k)^2}{d_k'Ad_k \; d_k'A^{-1}d_k}\right]. \tag{2.8}$$

We now estimate the quantity in the square brackets. If we compute the condition number of a matrix with respect to the Euclidean vector norm, then for positive definite matrices, it coincides with the *spectral condition number*

$$\kappa(A) = \frac{\lambda_{\max}(A)}{\lambda_{\min}(A)}.$$

2.3 The Kantorovich Inequality. *Let A be a symmetric, positive definite matrix with spectral condition number κ. Then*

$$\frac{(x'Ax)(x'A^{-1}x)}{(x'x)^2} \le (\frac{1}{2}\sqrt{\kappa} + \frac{1}{2}\sqrt{\kappa^{-1}})^2 \tag{2.9}$$

for every vector $x \neq 0$; cf. Kantorovich [1948].

Proof. Set $\mu := [\lambda_{\max}(A)\,\lambda_{\min}(A)]^{1/2}$, and let λ_i be an eigenvalue of A. It follows that $\kappa^{-1/2} \le \lambda_i/\mu \le \kappa^{1/2}$. Since the function $z \mapsto z + z^{-1}$ is monotone on the interval $(0, 1)$ and for $z > 1$, we know that

$$\lambda_i/\mu + \mu/\lambda_i \le \kappa^{1/2} + \kappa^{-1/2}.$$

The eigenvectors of A are also eigenvectors of the matrix $\frac{1}{\mu}A + \mu A^{-1}$, and the eigenvalues of the latter are bounded by $\kappa^{1/2} + \kappa^{-1/2}$. Hence, it follows from Courant's maximum principle that

$$\frac{1}{\mu}(x'Ax) + \mu(x'A^{-1}x) \le (\kappa^{1/2} + \kappa^{-1/2})\,(x'x)$$

holds for all $x \in \mathbb{R}^n$. The variant of Young's inequality $ab \le \frac{1}{4}(|a| + |b|)^2$ now yields

$$(x'Ax)(x'A^{-1}x) \le \frac{1}{4}\big[\frac{1}{\mu}(x'Ax) + \mu(x'A^{-1}x)\big]^2 \le \frac{1}{4}(\kappa^{1/2} + \kappa^{-1/2})^2\,(x'x)^2,$$

and the proof is complete. □

The left-hand side of (2.9) attains its maximum when the vector x contains only components from the eigenvectors that are associated to the largest and the smallest eigenvalue and if the two contributions have the same size. Furthermore, Example 2.5 below shows that the bound is sharp.

Now combining (2.8), (2.9), and the identity $1 - 4/(\sqrt{\kappa} + \sqrt{\kappa^{-1}})^2 = (\kappa - 1)^2/(\kappa + 1)^2$, we get

2.4 Theorem. *Suppose A is a symmetric positive definite matrix with spectral condition number κ. Then applying the gradient method to the function (2.1) generates a sequence with*

$$\|x_k - x^*\|_A \leq \left(\frac{\kappa - 1}{\kappa + 1}\right)^k \|x_0 - x^*\|_A. \tag{2.11}$$

Convergence Behavior in the Case of Large Condition Numbers

If we solve a system of equations with a very large condition number κ, the convergence rate

$$\frac{\kappa - 1}{\kappa + 1} \approx 1 - \frac{2}{\kappa}$$

is very close to 1. We can show that this unfavorable rate dominates the iteration with a simple example involving two unknowns.

2.5 Example. Let $a \gg 1$. We seek the minimum of

$$f(x, y) = \frac{1}{2}(x^2 + ay^2), \quad \text{i.e.,} \quad A = \begin{pmatrix} 1 & \\ & a \end{pmatrix}. \tag{2.12}$$

Here $\kappa(A) = a$. Suppose we choose

$$(x_0, y_0) = (a, 1)$$

as the starting vector. Then the direction of steepest descent is given by $(-1, -1)$. We claim that

$$x_{k+1} = \rho x_k, \quad y_{k+1} = -\rho y_k, \quad k = 0, 1, \ldots, \tag{2.13}$$

where $\rho = (a - 1)/(a + 1)$. This is easily checked for $k = 0$, and it follows for all k by symmetry arguments. Thus, the convergence rate is exactly as described in Theorem 2.4.

The contour lines of the function (2.12) are strongly elongated ellipses (see Fig. 46), and the angle between the gradient and the direction leading to a minimum can be very large (cf. Problem 2.8).

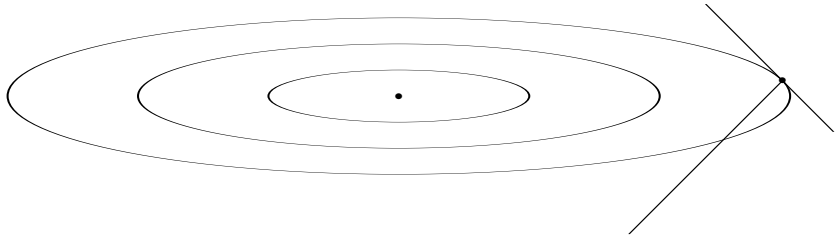


Fig. 46. Contour lines and the gradient for Example 2.5

For two-dimensional problems, there is one exceptional case. Since $d_k' \nabla f(x_{k+1}) = 0$, we have $d_k' d_{k+1} = 0$. Thus, d_k and d_{k+2} are parallel in $\mathbb{R}^2$. This cycling does not occur in n-space for $n \geq 3$. If we start with an arbitrary vector in $\mathbb{R}^n$ $(n \geq 3)$, after a few iterations we get a vector which lies in a position similar to that in Example 2.5. This agrees with the observation that the iteration nearly stops after a few steps, and any further approach to the solution will be extremely slow.

Problems

2.6 By (2.5), $\alpha_k \geq \alpha^* := 1/\lambda_{\max}(A)$. Show that convergence is guaranteed for every fixed step size α with $0 < \alpha < 2\alpha^*$.

2.7 Establish the recurrence (2.13) in Example 2.5.

2.8 (a) In Example 2.5 the directions d_k and d_{k+1} are orthogonal w.r.t. the Euclidean metric. Show that they are nearly parallel in the metric defined by (2.6). Here the angle between two vectors x and y is given by

$$\cos\varphi = \frac{x'Ay}{\|x\|_A \|y\|_A}. \tag{2.14}$$

(b) Show that $|\cos\varphi| \leq \frac{\kappa-1}{\kappa+1}$ for the angle defined in (2.14), provided $x'y = 0$.
Hint: For the unit vector,

$$\frac{1-\cos\varphi}{1+\cos\varphi} = \frac{\|x-y\|_A}{\|x+y\|_A}.$$

2.9 Consider the unbounded quadratic form

$$f(x) = \sum_{j=1}^{\infty} j|x^{(j)}|^2$$

in ℓ_2. Let $x_0 = (1, 1/8, 1/27, \ldots)$, or more generally let x_0 be any element with $f(x_0) < \infty$, $\nabla f(x_0) \in \ell_2$, such that the rate of decay of the components is polynomial. Show that the gradient method stops after finitely many steps.

2.10 Use the Kantorovitch inequality to give upper and lower bounds for the quotients of

$$(x'Ax)^2 \quad \text{and} \quad (x'A^2x)(x'x)$$

and also for those of

$$(x'Ax)^n \quad \text{and} \quad (x'A^nx)(x'x)^{n-1}.$$

§ 3. Conjugate Gradients and the Minimal Residual Method

The conjugate gradient method was developed in 1952 by Hestenes and Stiefel, and first became of importance in 1971 when it was combined with a simple method for preconditioning by Reid [1971]. The big advantage of the conjugate gradient method became apparent when faster computers became available and larger problems were attacked.

For the systems of equations which arise in the discretization of two-dimensional second order elliptic boundary-value problems, the method is more efficient than Gauss elimination as soon as 400 to 800 unknowns are involved, and also uses much less memory. For three-dimensional problems, the advantages are even more significant than for two dimensions. The advantages are less clear for problems of fourth order.

In this section we restrict our discussion to linear problems. We discuss further details in the next section, where we also present a generalization to nonlinear minimization problems.

The basic idea behind the *conjugate gradient method* (CG method) is to make sure that successive directions are not nearly parallel (in the sense of Problem 2.8). The idea is to use orthogonal directions, where orthogonality is measured in a metric more suited to problem (2.1) than the *Euclidean metric*.

3.1 Definition. Let A be a symmetric nonsingular matrix. Then two vectors x and y are called *conjugate* or *A-orthogonal* provided $x'Ay = 0$.

In the following we will assume that A is positive definite. In this case, any set of k pairwise conjugate vectors $x_1, x_2, \ldots, x_k$ are linearly independent provided that none of them is the zero vector.

In particular, suppose $d_0, d_1, \ldots, d_{n-1}$ are conjugate directions, and that the desired solution $x^* = A^{-1}b$ has the expansion

$$x^* = \sum_{k=0}^{n-1} \alpha_k d_k$$

in terms of this basis. Then in view of the orthogonality relations, we have $d_i' A x^* = \sum_k d_i' A \alpha_k d_k = \alpha_i d_i' A d_i$, and

$$\alpha_i = \frac{d_i' A x^*}{d_i' A d_i} = \frac{d_i' b}{d_i' A d_i}. \tag{3.1}$$

Thus, when using conjugate vectors – in contrast to using an arbitrary basis – we can compute the coefficients α_i in the expansion of x^* directly from the given vector b.

Let x_0 be an arbitrary vector in $\mathbb{R}^n$. Then an expansion of the desired correction vector $x^* - x_0$ in terms of conjugate directions can be computed recursively directly from the gradients $g_k := \nabla f(x_k)$.

3.2 Lemma on Conjugate Directions. *Let $d_0, d_1, \ldots, d_{n-1}$ be conjugate directions, and let $x_0 \in \mathbb{R}^n$. Then the sequence generated by the recursion*

$$x_{k+1} = x_k + \alpha_k d_k$$

with

$$\alpha_k = -\frac{d_k' g_k}{d_k' A d_k}, \quad g_k := Ax_k - b,$$

gives a solution $x_n = A^{-1}b$ after (at most) n steps.

Proof. Writing $x^* - x_0 = \sum_i \alpha_i d_i$, from (3.1) we immediately have

$$\alpha_k = \frac{d_k' A(x^* - x_0)}{d_k' A d_k} = -\frac{d_k'(Ax_0 - b)}{d_k' A d_k}.$$

Since the vector d_k is assumed to be conjugate to the other directions, we have $d_k' A(x_k - x_0) = d_k' A \sum_i^{k-1} \alpha_i d_i = 0$. Hence,

$$\alpha_k = -\frac{d_k'(Ax_k - b)}{d_k' A d_k} = -\frac{d_k' g_k}{d_k' A d_k}. \qquad \square$$

3.3 Corollary. *Under the hypotheses of 3.2, x_k minimizes the function f not only on the line $\{x_{k-1} + \alpha d_{k-1};\ \alpha \in \mathbb{R}\}$, but also over $x_0 + V_k$, where $V_k := \operatorname{span}[d_0, d_1, \ldots, d_{k-1}]$. In particular,*

$$d_i' g_k = 0 \quad \text{for } i < k. \tag{3.2}$$

Proof. It suffices to establish the relation (3.2). The choice of α_i ensures that

$$d_i' g_{i+1} = 0. \tag{3.3}$$

Thus, (3.2) holds for $k = 1$. Assume that the assertion has been proved for $k - 1$. From $x_k - x_{k-1} = \alpha_{k-1} d_{k-1}$ it follows that $g_k - g_{k-1} = A(x_k - x_{k-1}) = \alpha_{k-1} A d_{k-1}$. Since the directions $d_0, d_1, \ldots, d_{k-1}$ are conjugate, we have $d_i'(g_k - g_{k-1}) = 0$ for $i < k - 1$. Combining this with the induction hypothesis, i.e. (3.2) for $k - 1$, we conclude that the formula is also correct for k and $i \leq k - 2$. The remaining equation for k and $i = k - 1$ is a direct consequence of (3.3). $\square$

The CG Algorithm

In the conjugate gradient method, the directions $d_0, d_1, \ldots, d_{n-1}$ are not selected in advance, but are computed from the current gradients g_k by addition of a correction factor. From an algorithmic point of view, this means that we do not need a complicated orthogonalization process, but instead can use a simple three-term recurrence formula.[9] We show later that this approach makes sense from an analytic point of view.

3.4 The Conjugate Gradient Method.
Let $x_0 \in \mathbb{R}^n$, and set $d_0 = -g_0 = b - Ax_0$.
For $k = 0, 1, 2, \ldots$, compute

$$\begin{aligned} \alpha_k &= \frac{g_k' g_k}{d_k' A d_k}, \\ x_{k+1} &= x_k + \alpha_k d_k, \\ g_{k+1} &= g_k + \alpha_k A d_k, \\ \beta_k &= \frac{g_{k+1}' g_{k+1}}{g_k' g_k}, \\ d_{k+1} &= -g_{k+1} + \beta_k d_k, \end{aligned} \tag{3.4}$$

while $g_k \neq 0$. □

We note that in the derivation of the above, we initially get

$$\alpha_k = -\frac{g_k' d_k}{d_k' A d_k}, \quad \beta_k = \frac{g_{k+1}' A d_k}{d_k' A d_k}. \tag{3.5}$$

For quadratic problems, the expressions in (3.4) are equivalent, but are more stable from a numerical point of view, and require less memory.

3.5 Properties of the CG Method. *While $g_{k-1} \neq 0$, we have the following:*
(1) $d_{k-1} \neq 0$.
(2)

$$\begin{aligned} V_k &:= \operatorname{span}[g_0, Ag_0, \ldots, A^{k-1} g_0] \\ &= \operatorname{span}[g_0, g_1, \ldots, g_{k-1}] = \operatorname{span}[d_0, d_1, \ldots, d_{k-1}]. \end{aligned}$$

(3) The vectors $d_0, d_1, \ldots, d_{k-1}$ are pairwise conjugate.
(4)

$$f(x_k) = \min_{z \in V_k} f(x_0 + z). \tag{3.6}$$

[9] Three-term recurrence relations are well known for orthogonal polynomials. The connection is explored in Problem 3.9.

Proof. The assertions are obvious for $k = 1$. Suppose that they hold for some $k \geq 1$. Then

$$g_k = g_{k-1} + A(x_k - x_{k-1}) = g_{k-1} + \alpha_{k-1} A d_{k-1},$$

and thus $g_k \in V_{k+1}$ and $\operatorname{span}[g_i]_{i=0}^{k} \subset V_{k+1}$. By the induction hypothesis, d_0, d_1, ..., d_{k-1} are conjugate, and by the optimality of x_k, we have

$$d_i' g_k = 0 \quad \text{for } i < k. \tag{3.7}$$

Thus, g_k is linearly dependent on $d_0, d_1, \ldots, d_{k-1}$ only if $g_k = 0$, and so $g_k \neq 0$ implies $g_k \notin V_k$. It follows that $\operatorname{span}[g_i]_{i=0}^{k}$ is a $(k+1)$-dimensional space, and cannot be a proper subspace of V_{k+1}. This establishes the first equality in (2) for $k+1$. Moreover, V_{k+1} coincides with $\operatorname{span}[d_i]_{i=0}^{k}$, since in view of $g_k + d_k \in V_k$, we could just as well have included d_k.

Now $d_k \neq 0$ immediately implies $g_k + d_k \in V_k$ provided $g_k \neq 0$, and thus (1) holds.

For the proof of (3), we compute

$$d_i' A d_k = -d_i' A g_k + \beta_{k-1} d_i' A d_{k-1}. \tag{3.8}$$

For $i \leq k-2$ the first term on the right-hand side vanishes because $A d_i \in A V_{k-1} \subset V_k$ and (3.7) holds. Moreover, by assumption the second term is zero. Finally, with β_k as in (3.5), the right-hand side of (3.8) vanishes for $i = k - 1$.

The last assertion follows from Corollary 3.3, and the inductive proof is complete. □

The use of conjugate directions prevents the iteration from proceeding in nearly parallel directions (cf. Problem 2.8). In order to get conjugate directions from nearly parallel directions, we need to use very large factors β_k. Thus in principle, we have to keep in mind the possibility that small denominators will be encountered due to roundoff errors, and the iteration will have to be restarted. This doesn't happen as often as might be expected, however, since by Problem 3.13, the denominator can be reduced by at most a factor κ; cf. Powell [1977].

In this connection we should note that the accumulated roundoff errors do not lead to a loss of stability. Since d_k is a linear combination of g_k and d_{k-1}, x_{k+1} provides a minimum for f over the two-dimensional manifold

$$x_k + \operatorname{span}[g_k, d_{k-1}].$$

This problem is still well posed if the variables with roundoff errors are used in the calculation instead of the exact vectors g_k and d_{k-1}. The choice of the coefficients α_k and β_k according to (3.4) ensures that the *perturbed* two-dimensional minimum problem is well posed; see 3.13 and 4.11. This is the reason for the numerical stability of the CG method.

Analysis of the CG method as an Optimal Method

The CG method finds a minimum of a quadratic function on $\mathbb{R}^n$ in (at most) n steps. However, at the time of its discovery, the fact that this iterative method finds a solution in n steps was overemphasized. Indeed, for problems with 1000 or more unknowns, this property is of little significance. What is much more important is the fact that a very good approximation can already be found with many fewer than n steps.

First we note that for all iterative methods of the form

$$x_{k+1} = x_k + \alpha_k(b - Ax_k), \qquad \text{with } \alpha_k \in \mathbb{R}$$

– and thus also for the simplest gradient method – the approximation x_k lies in $x_0 + V_k$, where V_k is defined as in 3.5(2). It follows from 3.5(2) that among all of these methods, the CG method is the one which gives the smallest error $\|x_k - x^*\|_A$.

As usual, the *spectrum* $\sigma(A)$ of a matrix A is the set of its eigenvalues.

3.6 Lemma. *Suppose there exists a polynomial $p \in \mathcal{P}_k$ with*

$$p(0) = 1 \quad \textit{and} \quad |p(z)| \le r \textit{ for all } z \in \sigma(A). \tag{3.9}$$

Then for arbitrary $x_0 \in \mathbb{R}^n$, the CG method satisfies

$$\|x_k - x^*\|_A \le r\|x_0 - x^*\|_A. \tag{3.10}$$

Proof. Set $q(z) = (p(z) - 1)/z$. Using the same notation as in 3.5, we have $y := x_0 + q(A)g_0 \in x_0 + V_k$, and $g_0 = A(x_0 - x^*)$ implies

$$\begin{aligned} y - x^* &= x_0 - x^* + y - x_0 = x_0 - x^* + q(A)g_0 \\ &= p(A)(x_0 - x^*). \end{aligned}$$

Now let $\{z_j\}_{j=1}^n$ be a complete system of orthonormal eigenvectors with $Az_j = \lambda_j z_j$ and $x_0 - x^* = \sum_j c_j z_j$. Then

$$y - x^* = \sum_j c_j\, p(A) z_j = \sum_j c_j\, p(\lambda_j) z_j. \tag{3.11}$$

The orthogonality of the eigenvectors implies

$$\|x_0 - x^*\|_A^2 = \sum_j \lambda_j |c_j|^2 \tag{3.12}$$

and

$$\|y - x^*\|_A^2 = \sum_j \lambda_j |c_j\, p(\lambda_j)|^2 \le r^2 \sum_j \lambda_j |c_j|^2.$$

Thus, $\|y - x^*\|_A \le r\|x_0 - x^*\|_A$. Combining this with $y \in x_0 + V_k$ and the minimal property 3.5(4) for x_k, we get (3.10). □

If we only know that the spectrum lies in the interval $[a, b]$, where $b/a = \kappa$, then we get optimal estimates using the so-called *Chebyshev polynomials*[10]

$$T_k(x) := \frac{1}{2}[(x + \sqrt{x^2 - 1})^k + (x - \sqrt{x^2 - 1})^k], \quad k = 0, 1, \ldots \tag{3.13}$$

The formulas (3.13) define polynomials with real coefficients since after multiplying out and using the binomial formula, the terms with odd powers of the roots cancel each other. Moreover, $|x+\sqrt{x^2-1}| = |x+i\sqrt{1-x^2}| = 1$ for real $|x| \le 1$, and thus

$$T_k(1) = 1 \quad \text{and} \quad |T_k(x)| \le 1 \quad \text{for } -1 \le x \le 1. \tag{3.14}$$

The special polynomial $p(z) := T([b + a - 2z]/[b - a])/T([b + a]/[b - a])$ satisfies $p(0) = 1$, and so Lemma 3.6 implies the main result:

3.7 Theorem. *For any starting vector $x_0 \in \mathbb{R}^n$, the CG method generates a sequence x_k satisfying*

$$\begin{aligned} \|x_k - x^*\|_A &\le \frac{1}{T_k\left(\dfrac{\kappa+1}{\kappa-1}\right)} \|x_0 - x^*\|_A \\ &\le 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^k \|x_0 - x^*\|_A . \end{aligned} \tag{3.15}$$

Proof. Since $\sigma(A) \subset [\lambda_{\min}, \lambda_{\max}]$ and $\kappa = \lambda_{\max}/\lambda_{\min}$, the first inequality follows with the special polynomial above. Recalling (3.13) we clearly have $T_k(z) \ge \frac{1}{2}(z+\sqrt{z^2-1})^k$ for $z \in [1, \infty)$. We evaluate $z+\sqrt{z^2-1}$ for $z := (\kappa+1)/(\kappa-1)$ and use $\kappa - 1 = (\sqrt{\kappa}+1)(\sqrt{\kappa}-1)$:

$$\frac{\kappa+1}{\kappa-1} + \sqrt{\left(\frac{\kappa+1}{\kappa-1}\right)^2 - 1} = \frac{\kappa+1+\sqrt{4\kappa}}{\kappa-1} = \frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1}.$$

This establishes the second assertion. □

A comparison with Theorem 2.4 and Example 2.5 shows that the computation of conjugate directions has the same positive effect as replacing the condition number by its square root. In practice, the improvement in each iteration is usually even better than theoretically predicted by Theorem 3.7. The inequality (3.15) also covers the pessimistic case where the eigenvalues are uniformly distributed between $\lambda_{\min}$ and $\lambda_{\max}$. Frequently, the eigenvalues appear in groups, and because of the gaps in the spectrum, Theorem 3.7 is too pessimistic.

Here we should point out the connection with the so-called *semi-iterative methods* [Varga 1962]. They are based on modifying the relaxation method

[10] The usual definition $T_k(x) := \cos(k \arccos x)$ is equivalent to (3.13).

so that using the comparison polynomial q_k constructed in Theorem 3.7, we actually get $x_k = x_0 + q_k(A)(x_0 - x^*)$ for certain k. The process is optimal if $\lambda_{\min}$ and $\lambda_{\max}$ are known. – The situation is better for the conjugate gradient method since for it, even this information is not needed, and since the gaps in the spectrum automatically lead to an acceleration in the convergence.

In practice, the errors decrease nonuniformly in the course of the iteration. After a clear reduction of the error in the first few steps, there is often a phase with only small improvements. Then the convergence speeds up again. Often, in terms of computational time, it is not important whether we choose a relative accuracy of 10^{-4} or 10^{-5}, see the numerical example in §4. – Here we note that the factors α_k and β_k in (3.4) vary from step to step, and the iteration is *not* a stationary linear process.

The Minimal Residual Method

There is an easy modification of Method 3.4 for which x_k in the linear manifold $x_0 + V_k$ minimizes the error in the norm

$$\|x_k - x^*\|_{A^\mu}$$

for some $\mu \geq 1$, rather than in the energy norm. To achieve this, we replace the scalar products $u'v$ in the quotients in (3.4) by $u'A^{\mu-1}v$. The Euclidean norm $\|x_k - x^*\| = \|x_k - x^*\|_{A^0}$ can also be easily minimized by using the space $x_0 + AV_k$ instead of $x_0 + V_k$.

The case $\mu = 2$ is of some practical importance. Since

$$\|x - x^*\|_{A^2}^2 = \|Ax - b\|^2 = x'A^2x - 2b'Ax + const, \tag{3.16}$$

it is called the *minimal residual method*. The method is also applicable for indefinite or unsymmetric matrices. We shall use this method to illustrate that the strength of the CG method is due more to its *analytic* properties than to its simple *algebraic* properties.

For $\mu > 1$, Lemma 3.6 and Theorem 3.7 hold for positive definite matrices. It follows from (3.11) that the vector y constructed in the proof of the corollary satisfies

$$\|y - x^*\|_{A^\mu} \leq r\|x_0 - x^*\|_{A^\mu}. \tag{3.17}$$

Although the leading term in the quadratic form (3.16) is determined by the matrix A^2, the rate of convergence depends on $\kappa(A)$, rather than $\kappa(A^2)$.

Indefinite and Unsymmetric Matrices

We now turn to indefinite problems. First, we recall that an indefinite system $Ax = b$ cannot be converted into a system $A^2x = Ab$ with positive definite matrix simply by multiplication by A. Indeed, since $\kappa(A^2) = [\kappa(A)]^2$, the condition number increases significantly under the transformation. This raises the question of whether we can avoid this shortcoming by applying the minimal residual method.

This is in fact the case for problems with only a few negative eigenvalues, and also for unsymmetric spectra. However, if the spectrum is symmetric w.r.t. zero, then unfortunately we are faced with the same effect as squaring the matrix.

3.8 Example. Suppose we double the system $Ay = b$,

$$\begin{pmatrix} A & \\ & -A \end{pmatrix} \begin{pmatrix} y \\ z \end{pmatrix} = \begin{pmatrix} b \\ -b \end{pmatrix},$$

where A is positive definite. Let $y_0 = z_0$. Then the residual has the form $(h_0, -h_0)'$. Since the expression $\|A(y_0 + \alpha h_0) - b\|^2 + \| - A(z_0 - \alpha h_0) + b\|^2$ assumes its minimum at $\alpha = 0$, it follows that $y_1 = y_0 = z_1 = z_0$. This shows that in general, we get an improvement only for an even number of steps, and the minimum for $x_0 + \text{span}[Ag_0, A^3g_0, A^5g_0, \ldots]$ will be computed. Unfortunately, this again corresponds to the calculation with the squared matrix. □

Since in contrast to 3.5(2), here the gradients $g_1, g_2, g_3, \ldots$ are not linearly independent, the formal extension of the minimal residual algorithm breaks down. More importantly, as we shall see later in Remark 4.3, for minimal residuals, the preconditioning generally can no longer be built into a three-term recurrence.

To treat problems with indefinite or unsymmetric matrices, we need to make modifications; cf. Paige and Saunders [1975], Stoer [1983], Golub and van Loan [1983]. They are still relatively simple for symmetric indefinite matrices. The Cholesky decomposition hidden in the CG method is replaced by a QR decomposition; cf. Paige and Saunders [1975]. Otherwise, we have to distinguish between an incomplete minimization and a very short recurrence with stabilization. QMRES and its variants belong to the first group; see Saad and Schultz [1985] and Saad [1993]. It generates directions which are conjugate only to the last few difference vectors. The other methods use two systems of biorthogonal vectors; see van der Vorst [1992]. In order to avoid degeneracies as in Example 3.8, several steps are carried out together. This is called the "look ahead strategy"; cf. Freund, Gutknecht, and Nachtigal [1993]. Various studies have shown that no optimal algorithm exists for indefinite and unsymmetric problems.

Because of this phenomenon, completely different methods based on the Uzawa algorithm have been developed for the class of indefinite problems involving saddle point problems. They are described in §5 below.

Problems

3.9 Suppose $z \in \mathbb{R}^n$ and $k \geq 1$. In addition, let A, B and C be positive definite $n \times n$ matrices. Suppose that the matrices A and B commute. Using as few arithmetic operations as possible, compute A-orthogonal directions $d_0, d_1, \ldots, d_k$, which span the same space as

(a) $z, Az, A^2z, \ldots, A^kz$,
(b) $z, CAz, (CA)^2z, \ldots, (CA)^kz$,
(c) $z, Bz, B^2z, \ldots, B^kz$.

How many matrix-vector multiplications, and how many scalar products are needed? When can A^2-orthogonal directions be computed simply?

3.10 Let $S = \{a_0\} \cup [a, b]$ with $0 < a_0 < a < b$ and $\kappa = b/a$. Show that there exists a polynomial p of degree k such that

$$p(0) = 1 \quad \text{and} \quad |p(x)| \leq \frac{2b}{a_0} \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^{k-1} \quad \text{for } x \in S.$$

3.11 How does the iteration in Problem 2.9 perform using conjugate directions? Does the difficulty discussed in Problem 2.9 disappear if we use conjugate directions?

3.12 Let $\kappa(A) = 1000$. How many iteration steps are needed in the gradient and the conjugate gradient methods in order to reduce the error by 0.01 in the worst case?

3.13 The choice of α_k guarantees that $d_k' g_{k+1} = 0$, independent of the roundoff errors in the previous steps. Thus, $\|d_{k+1}\|_A$ is just the distance of the vector g_{k+1} from the one-dimensional linear space span$[d_k]$. Show that

$$\|d_{k+1}\|_A \geq \frac{1}{\kappa(A)^{1/2}} \|g_{k+1}\|_A .$$

Hint: First compare the Euclidean norm of the vectors d_{k+1} and g_{k+1}.

§ 4. Preconditioning

The conjugate gradient method becomes an especially efficient method when it is coupled with preconditioning. The combination is called the *preconditioned conjugate gradient method* or, for short, the *PCG method*. We describe two standard preconditioning methods which suffice for the solution of many systems of equations arising from second order elliptic problems. These methods do not need to be tailored to the problem at hand, and can even be built into a black box.

Given the equation $Ax = b$, suppose we have an easily invertible positive definite matrix C which approximates the matrix A. We discuss later how to measure the quality of the approximation. Given $x_0 \in \mathbb{R}^n$, consider

$$x_1 = x_0 - \alpha C^{-1} g_0, \tag{4.1}$$

where $g_0 = Ax_0 - b$. If $C = A$, then we already get the solution in the first step. Thus, it is to be expected that choosing C to be any (reasonable) approximation to A will get us to the solution faster than the trivial choice $C = I$.

This idea leads to the following algorithm:

4.1 The Conjugate Gradient Method with Preconditioning.
Let $x_0 \in \mathbb{R}^n$. Set $g_0 = Ax_0 - b$, $d_0 = -h_0 = -C^{-1}g_0$, and compute

$$\begin{aligned} x_{k+1} &= x_k + \alpha_k d_k, \\ \alpha_k &= \frac{g_k' h_k}{d_k' A d_k}, \\ g_{k+1} &= g_k + \alpha_k A d_k, \\ h_{k+1} &= C^{-1} g_{k+1}, \\ d_{k+1} &= -h_{k+1} + \beta_k d_k, \\ \beta_k &= \frac{g_{k+1}' h_{k+1}}{g_k' h_k}, \end{aligned} \tag{4.2}$$

for $k \geq 0$. □

If C is positive definite, then in analogy with 3.5 we have

4.2 Properties of the PCG Method. *As long as $g_{k-1} \neq 0$, we have*

(1) $d_{k-1} \neq 0$.

(2) $V_k := \operatorname{span}[g_0, AC^{-1}g_0, \ldots, (AC^{-1})^{k-1}g_0] = \operatorname{span}[g_0, g_1, \ldots, g_{k-1}]$
and $\operatorname{span}[d_0, d_1, \ldots, d_{k-1}] = C^{-1} \operatorname{span}[g_0, g_1, \ldots, g_{k-1}]$.

(3) The vectors $d_0, d_1, \ldots, d_{k-1}$ are pairwise conjugate.
(4)

$$f(x_k) = \min_{z \in V_k} f(x_0 + C^{-1}z). \tag{4.3}$$

The proof of these algebraic properties proceeds in exactly the same way as the proof of 3.5, and can be left to the reader.

4.3 Remark. The matrices C and A do not have to commute for the method to work, see Problem 3.9. Indeed, we only need to compute scalar products of the form

$$((AC^{-1})^j u)' A (C^{-1}A)^k v,$$

and the matrix $((AC^{-1})^j)^t A (C^{-1}A)^k$ depends only on the sum $k + j$. Unfortunately, for $((AC^{-1})^j)^t A^2 (C^{-1}A)^k$ this holds only in exceptional cases, and combining preconditioning with the minimal residual method is not so simple; see Axelsson [1980], Young and Kang [1980], Saad and Schultz [1985]. Conjugate directions can no longer be determined by three-term recurrence relations. Therefore, we do not attempt to find a complete orthogonalization, and instead make sure that the new direction is conjugate to the last five directions, say.

The convergence theory for the CG method can be generalized as follows.

4.4 Theorem. *(1) Suppose there exists a polynomial $p \in \mathcal{P}_k$ with*

$$p(0) = 1 \quad \textit{and} \quad |p(z)| \le r \quad \textit{for all } z \in \sigma(C^{-1}A).$$

Then for arbitrary $x_0 \in \mathbb{R}^n$, the PCG method satisfies

$$\|x_k - x^*\|_A \le r \|x_0 - x^*\|_A.$$

(2) With $\kappa = \kappa(C^{-1}A)$,

$$\|x_k - x^*\|_A \le 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k \|x_0 - x^*\|_A.$$

Proof. Consider $q(z) := (p(z) - 1)/z$. Then $y := x_0 + q(C^{-1}A)C^{-1}g_0 \in x_0 + C^{-1}V_k$, and so $y - x^* = p(C^{-1}A)(x_0 - x^*)$. Now let $\{z_j\}_{j=1}^n$ be a complete system of eigenvectors for the problem

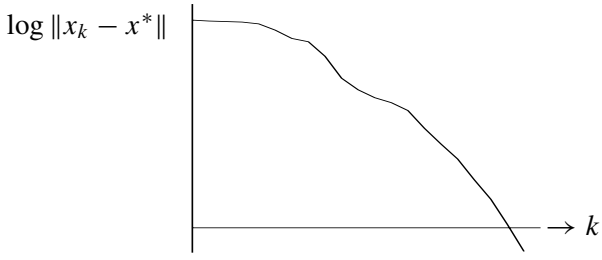
$$Az_j = \lambda_j C z_j, \quad j = 1, 2, \ldots, n. \tag{4.4}$$

In particular, suppose the vectors are normalized so that

$$z_i' C z_j = \delta_{ij} \quad \text{for } i, j = 1, 2, \ldots, n.$$

0	12.95	14	3.93	28	3.13_{-2}
2	12.31	16	1.76	30	1.33_{-2}
4	11.99	18	0.519	32	5.79_{-3}
6	11.64	20	0.273	34	1.82_{-3}
8	10.55	22	0.175	36	6.21_{-4}
10	7.47	24	0.130	38	1.51_{-4}
12	4.76	26	0.086	40	3.35_{-5}

Table 6 and Fig. 47. Reduction in the energy norm of the error when the PCG method is applied to a cantilever problem with 544 unknowns. The slow decrease at the beginning and again in the middle is typical for the CG method



Then $z_i' A z_j = \lambda_j \delta_{ij}$, and (3.12) again follows. Moreover, $(C^{-1}A)^\ell z_j = \lambda_j^\ell z_j$ for all ℓ. The rest of the proof of (1) follows as in Lemma 3.5.

Since the numbers λ_j in (4.4) are actually the eigenvalues of $C^{-1}A$, the assertion (2) follows by the arguments used in Theorem 3.7. □

Preconditioning also helps to reduce the effect of the following difficulty. In principle, in using gradient methods we want to choose the *direction of steepest descent*. Which direction gives the steepest descent depends on the metric of the space. For the simple gradient method, we implicitly use the Euclidean metric. But if $\|x\|_C := \sqrt{x'Cx}$ is a better approximation to the metric $\|x\|_A$ than the Euclidean metric $\|x\| = \sqrt{x'x}$, then C is a good choice for preconditioning. By Theorem 4.4, the *oscillation* of the quotient

$$\frac{x'Ax}{x'Cx} \tag{4.5}$$

is the main determining factor for the rate of convergence. We shall make use of similar arguments in the following section.

Although the widely applicable methods described below can be used for the solution of second order boundary-value problems, for large problems of fourth order, we usually have to tailor the preconditioning to the problem. This is due to the strong growth of order h^{-4} of κ. There are three common approaches to constructing special methods:

1. Subdivide the domain. The solution of the much smaller systems corresponding to the partial domains serves as a preconditioning; cf. Widlund [1988].

2. Alter the boundary conditions to give a simpler problem. (For example, a modification of the boundary conditions for the biharmonic equation leads to two decoupled Laplace equations, see Braess and Peisker [1986].) The approximate solution so obtained is then used for the preconditioning.

3. Use so-called *hierarchical bases*, i.e., choose basis functions consisting of low and high frequency functions, respectively; see Yserentant [1986], Xu [1992], or Bramble, Pasciak, and Xu [1990]. The condition number can be significantly reduced by a suitable scaling of the different parts.

Preconditioning by SSOR

A simple but effective preconditioning can be obtained from the Gauss–Seidel method, despite its slow convergence when it is used as stand-alone iteration. We decompose the given symmetric matrix A as

$$A = D - L - L^t,$$

where L is a lower triangular matrix and D is a diagonal matrix. Then for $1 < \omega < 2$,

$$x \mapsto (D - \omega L)^{-1}(\omega b + \omega L^t x - (\omega - 1)Dx)$$

defines an iteration step in the forward direction; cf. (1.19). Similarly, the relaxation in the backwards direction is defined by

$$x \mapsto (D - \omega L^t)^{-1}(\omega b + \omega L x - (\omega - 1)Dx).$$

Then the first half step gives

$$x^{1/2} = \omega(D - \omega L)^{-1} g_k,$$

where $x = 0$ and $b = g_k$, and the second half step gives

$$h_k = \omega(2 - \omega)(D - \omega L^t)^{-1} D (D - \omega L)^{-1} g_k,$$

since $\omega g_k + \omega L x^{1/2} - D x^{1/2} = 0$. In particular, $h_k = C^{-1} g_k$ with $C := [\omega(2 - \omega)]^{-1}(D - \omega L)D^{-1}(D - \omega L^t)$. Clearly, the matrix C is symmetric and positive definite.

We point out that multiplying the preconditioning matrix C by a positive factor has no influence on the iteration. Thus, the factor $\omega(2 - \omega)$ can be ignored in the calculation.

Experience shows that the quality of the preconditioning is not very sensitive to the choice of the parameter ω. Calculation with the fixed value $\omega = 1.3$ is only slightly worse than using the corresponding optimal values, which in practice lie between 1.2 and 1.6 [Axelsson and Barker 1984].

On the other hand, the numbering of the variables has a major influence on the performance of the method. The differences are very evident for the equations arising from five-point stencils on a rectangular mesh. We recommend that the *lexicographical* ordering $x_{11}, x_{12}, \ldots, x_{1n}, x_{21}, x_{22}, \ldots, x_{nn}$ be used. The *checkerboard ordering*, where all variables x_{ij} with $i + j$ even appear first, followed by all those with $i + j$ odd (or conversely), reduces the efficiency of the SSOR preconditioner dramatically. Thus it cannot be recommended. The disadvantages of this numbering are so great that they cannot even be compensated by vectorization or parallelization.

Preconditioning by ILU

Another preconditioning method can be developed from a variant of the Cholesky decomposition. For symmetric matrices of the type which appear in the finite element method, the Cholesky decomposition $A = LDL^t$ or $A = LL^t$ leads to a triangular matrix L which is significantly less sparse than A. Using an approximate inverse leads to the so-called *incomplete Cholesky decomposition (ICC)* or *incomplete LU decomposition (ILU)*; see Varga [1960]. In the simplest case, we simply avoid calculation with all matrix elements which vanish in the given matrix. This leads to a decomposition

$$A = LL^t + R \tag{4.6}$$

with an error matrix in which $R_{ij} \neq 0$ only appears if $A_{ij} = 0$.

This preconditioning method is often faster than the one using SSOR relaxation. However, there does not seem to be a general rule for deciding in which cases SSOR or ICC is more effective.

There are many variants of the method, and often filling in of elements in the neighborhood of the diagonal is allowed. In the so-called *modified incomplete decomposition* due to Meijerink and van Vorst [1977], instead of suppressing matrix elements, they are moved onto the main diagonal.

Gustafsson [1978] developed a preconditioning method for the standard five-point stencil for the Laplace equation. While in general there is only empirical evidence for the improvement of the conditioning, in this case he proved that the condition number is reduced from $\mathcal{O}(h^{-2})$ to $\mathcal{O}(h^{-1})$.

It is crucial for the proof that the diagonal elements can be increased by a small amount. Let $\zeta > 0$. In view of Friedrichs' inequality, we can estimate the quadratic forms $a(u, u) = |u|_1^2$ and $|u|_1^2 + \zeta \|u\|_0^2$ in terms of each other. The discretization

$$\begin{bmatrix} & 0 & \\ b_{i-1} & a_i & 0 \\ & c_{i-m} & \end{bmatrix}_* \qquad \begin{bmatrix} & c_i & \\ 0 & a_i & b_i \\ & 0 & \end{bmatrix}_*$$

$$\begin{bmatrix} b_{i-1}c_{i-1} & a_i c_i & \\ a_{i-1}b_{i-1} & a_i^2 + b_{i-1}^2 + c_{i-m}^2 & a_i b_i \\ & c_{i-m}a_{i-m} & b_{i-m}c_{i-m} \end{bmatrix}_*$$

$$\begin{bmatrix} & -\gamma_i & \\ -\beta_{i-1} & \alpha_i & -\beta_i \\ & -\gamma_{i-m} & \end{bmatrix}_* \qquad \begin{bmatrix} -r_i & & \\ & r_i + r_{i-m+1} & \\ & & -r_{i-m+1} \end{bmatrix}_*$$

Difference stencils for L, L^t (top), LL^t (middle) and A, R (bottom) for the incomplete Cholesky decomposition

of $\|u\|_0^2$ leads to the so-called *mass matrix*. Since its condition number is bounded independent of h, $\|u\|_0$ and $h^2\|u\|_{\ell_2}$ are equivalent norms. Thus, for the design of a preconditioning matrix, instead of the standard five-point stencil we can consider the following modified stencil:

$$\begin{bmatrix} & -1 & \\ -1 & 4 + \zeta_i h^2 & -1 \\ & -1 & \end{bmatrix}_* \tag{4.7}$$

where $0 < \zeta_i < \zeta$.

For simplicity, we now assume that the same number m of nodes lie on each horizontal grid line. Suppose that the neighbors of the node i to the South and West have the indices $i - m$ and $i - 1$, respectively.

The incomplete Cholesky decomposition leads to triangular matrices with at most three nonzero elements in every row. The general form of the difference stencils can be seen in the schemes above. The error matrix (4.6) can only have nonzero elements on the diagonal and at positions to the Northwest and Southeast. Combining this with $\sum_j R_{ij} = 0$, we see that the matrix R must have the form shown in the scheme. The coefficients a_i, b_i and c_i can be found recursively by

$$\begin{aligned} a_i^2 &= \alpha_i - b_{i-1}^2 - c_{i-m}^2 - r_i - r_{i-m+1}, \\ b_i &= -\beta_i / a_i, \\ c_i &= -\gamma_i / a_i, \\ r_i &= b_{i-1}c_{i-1}. \end{aligned} \tag{4.8}$$

Recalling (4.7) we set $\alpha_i := 4 + 8h^2$ and $\beta_i := \gamma_i := 1$ with the usual convention that β_i and γ_i are set to 0 at points next to the boundary. It follows by induction

that

$$a_i \geq \sqrt{2}(1+h), \quad 0 < b_i, c_i \leq 1/a_i, \quad r_i \leq \frac{1}{2}(1+h)^{-2}.$$

We can now estimate $x'Rx$ with the help of the formula $(x+y)^2 \leq 2(x^2+y^2)$:

$$\begin{aligned} 0 \leq x'Rx &= \sum_i r_i (x_i - x_{i+m-1})^2 \\ &\leq \sum_i \frac{1}{(1+h)^2}\{(x_i - x_{i-1})^2 + (x_{i-1} - x_{i+m-1})^2\} \\ &\leq \frac{1}{1+h} x'Ax. \end{aligned}$$

Combining this with (4.6), we have $x'LL^tx \geq h/(1+h)x'Ax$ and

$$x'LL^tx \leq x'Ax \leq \frac{1+h}{h} x'LL^tx$$

and thus

$$\kappa([LL^t]^{-1}A) \leq \frac{1+h}{h} = \mathcal{O}(h^{-1}). \tag{4.9}$$

Since $\kappa(A) = \mathcal{O}(h^{-2})$, the preconditioning has the effect that the effective condition number κ is reduced by one power of h.

The equation (4.6) also clearly shows that multiplication by $(LL^t)^{-1}$ would be equivalent to an SSOR step if the overrelaxation factors were point-independent and approximately of size $2 - \mathcal{O}(h^{-2})$. Thus, with small modifications in the argument, it is possible to show that in applying preconditioning with the SSOR method using a (fixed) factor $\omega = 2 - \mathcal{O}(h^{-2})$, the condition number is also reduced by one power of h, see Axelsson and Barker [1984].

Remarks on Parallelization

SSOR relaxation and multiplication by L^{-1} and R^{-1}, where L and R are associated with an ILU decomposition, are recursive processes. Nevertheless, both parallelization and vectorization are possible. [We recall that we should not choose a checkerboard order for simple parallelization.] The implementation depends heavily on the computer architecture. There is intense activity surrounding the use of parallel and vector machines, and so we would like to give a first impression of how to treat the kinds of banded matrices which arise for finite element problems.

We restrict our considerations to the equations arising from the use of the five-point stencil on a square domain with m^2 unknowns. We write the unknowns with double indices. Then in the first phase (the preconditioning), to determine the current variable x_{ij}, we need to know the values x_{kj} for $k < i$ and $x_{i\ell}$ for $\ell < j$.

In a vector machine we can collect the calculations of all x_{ij} with $i + j = const$. The calculation then proceeds in $2m$ groups. It is well known that we can save time in a vector machine by overlapping about eight arithmetic operations. The time saved is proportional to m^2, and is worthwhile if it exceeds the time required for the initialization of the $2m$ groups. Normally, this is the case when $m >$ approx. 40.

A different approach can be used on a parallel machine. We sketch the case of two processors [Wittum 1989a]. First we divide the domain into two parts. The first processor takes care of the nodes (i, j) with $j \leq n/2$, and the second one takes care of those with $j > n/2$. Once the first processor has dealt with $(1, 1), (1, 2), \ldots, (1, n/2)$, the second one is signaled. While the second processor works on its assigned values in the row $i = 1$, the first can do row $i + 1 = 2$. The two processors continue to work in parallel on succeeding rows.

We have to provide memory with access by two processors only at the boundaries, i.e. for $j = n/2$ and $j = n/2 + 1$. It is clear how memory can be freed for access by the other processor.

Without considering the memory restriction, there is also another approach which we could take. The first processor works on the entire first row. With the delay of one node, it signals the second processor to begin work on the second row. The remaining rows are then dealt with alternately by the two processors. In particular, with several processors, we get a complete parallelization after a short initialization time.

For more on parallelization, see, e.g., Hughes, Ferencz and Hallquist [1987], Meier and Sameh [1988], Ortega and Voigt [1985] and Ortega [1988].

Nonlinear Problems

The CG method can be carried over to nonlinear problems for which the function f to be minimized is not necessarily a quadratic function. This avoids iterating with the Newton method, where the solution of the corresponding linear system of equations would again require an iterative method.

As in §2, let f be a C^1 function defined on an open set $M \subset \mathbb{R}^n$. Very often f has the form

$$f(x) = \frac{1}{2}x'Ax - \sum_{i=1}^{n} d_i \phi(x_i) - b'x$$

with $\phi \in C^1(\mathbb{R})$. The first term has a more significant effect on poor conditioning than the second [Glowinski 1984]. Suppose we have a matrix C which is appropriate for preconditioning A (otherwise we choose $C = I$).

The minimization of

$$\frac{x'Ax}{x'x}$$

for the determination of the smallest eigenvalue of A also involves a non-quadratic problem.

4.5 The Conjugate Direction Method for Nonlinear Problems *following Fletcher and Reeves.*
Let $x_0 \in M$. Set $g_0 = \nabla f(x_0)$ and $d_0 = -h_0 = -C^{-1}g_0$.
For $k = 0, 1, 2, \ldots$, perform the following calculations:

1. Line search: Find the minimum of f on the line $\{x_k + td_k : t \geq 0\} \cap M$. Suppose the minimum (or a local minimum) is assumed at $t = \alpha_k$. Set

$$x_{k+1} = x_k + \alpha_k d_k.$$

2. Determination of the direction:

$$\begin{aligned} g_{k+1} &= \nabla f(x_{k+1}), \\ h_{k+1} &= C^{-1}g_{k+1}, \\ d_{k+1} &= -h_{k+1} + \beta_k d_k, \\ \beta_k &= \frac{g'_{k+1}h_{k+1}}{g'_k h_k}. \end{aligned} \tag{4.10}$$

□

4.6 Remark. In the method of Polak and Ribière, which is a variant of the Fletcher and Reeves method, β_k is not chosen as in (4.10), but instead we compute

$$\beta_k = \frac{g'_{k+1}(h_{k+1} - h_k)}{g'_k h_k}. \tag{4.11}$$

Problems

The following three exercises deal with the inversion of the so-called mass matrix.

4.7 Let $A_1, A_2, \ldots, A_k, C_1, C_2, \ldots, C_k$ be positive semidefinite matrices with

$$a\, x'C_i x \leq x'A_i x \leq b\, x'C_i x \quad \text{for } x \in \mathbb{R}^n \text{ and } i = 1, 2, \ldots, k.$$

In addition, let $0 < a \leq b$. Suppose that the matrices $A = \sum_i A_i$ and $C = \sum_i C_i$ are positive definite. Show that $\kappa(C^{-1}A) \leq b/a$.

4.8 Show that the matrix

$$A = \begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{pmatrix}$$

is positive definite, and that its condition number is 4.

Hint: The quadratic form associated with the matrix A is $x^2+y^2+z^2+(x+y+z)^2$.

4.9 The computation of the mass matrix $\int \psi_i \psi_j dx$ for linear triangular elements on the element level leads to the matrix in Problem 4.8 (w.r.t. the nodes which are involved). Show that we can get $\kappa \leq 4$ using preconditioning with an easily computable diagonal matrix. How much is the error reduced after three steps of the PCG method?

4.10 Consider Problem 3.12. How does the answer change if we replace κ by $2\sqrt{\kappa}$?

4.11 The equation $Ax = b$ implies

$$By = c \quad \text{with } B = C^{-1/2}AC^{-1/2},\ c = C^{-1/2}b, \tag{4.12}$$

where $y = C^{1/2}x$, since $C^{-1/2}AC^{-1/2}C^{1/2}x = C^{-1/2}b$. Show that applying the PCG method 4.1 (with preconditioning by the matrix C) to the original equation is equivalent to applying the CG method 3.4 to (4.12). Use this to derive the properties 4.2.

4.12 For preconditioning we often use a change of basis, e.g., in the method of hierarchical bases; see Yserentant [1986]. Let

$$x = Sy,$$

where S is a nonsingular matrix. Show that carrying out Algorithm 3.4 with the variables y is equivalent to Algorithm 4.1 with preconditioning based on

$$C^{-1} = SS^t.$$

4.13 For preconditioning we often use a matrix C which is not exactly symmetric. (In particular this is the case if multiplication by C^{-1} is only done approximately.) This means that we are not requiring that all of the d_k be pairwise conjugate. But we still want x_{k+1} to be the minimum of the function (2.1) over the two-dimensional manifold $x_k + \text{span}[h_k, d_{k-1}]$. Hence d_{k+1} and d_k (and d_k and d_{k-1}, respectively) should be conjugate. Which of the following formulas for β_k can be used for unsymmetric C?

$$(1)\quad \beta_k = \frac{g'_{k+1}h_{k+1}}{g'_k h_k}, \qquad (2)\quad \beta_k = \frac{(g'_{k+1} - g_k)h_{k+1}}{g'_k h_k}, \qquad (3)\quad \beta_k = \frac{h'_{k+1}Ad_k}{d'_k Ad_k}.$$

The following problems are useful not only for constructing preconditioners but also as preparation for a multigrid theory.

4.14 Let $A \leq B$ denote that $B - A$ is positive semidefinite. Show that $A \leq B$ implies $B^{-1} \leq A^{-1}$, but it does not imply $A^2 \leq B^2$.
To prove the first part note that $(x, B^{-1}x) = (A^{-1/2}x, A^{1/2}B^{-1}x)$ and apply Cauchy's inequality. Next consider the matrices

$$A := \begin{pmatrix} 1 & a \\ a & 2a^2 \end{pmatrix} \quad \text{and} \quad B := \begin{pmatrix} 2 & 0 \\ 0 & 3a^2 \end{pmatrix}$$

for establishing the negative result. From the latter it follows that we cannot derive good preconditioners for the biharmonic equation by applying those for the Poisson equation twice.

Note: The converse is more favorable, i.e., $A^2 \leq B^2$ implies $A \leq B$. Indeed, the Rayleigh quotient $\lambda = \max\{(x, Ax)/(x, Bx)$ is an eigenvalue, and the maximum is attained at an eigenvector, i.e., $Ax = \lambda Bx$. On the other hand, by assumption

$$0 \leq (x, B^2x) - (x, A^2x) = (1 - \lambda^2)\, \|Bx\|^2.$$

Hence, $\lambda \leq 1$ and the proof is complete.

4.15 Show that $A \leq B$ implies $B^{-1}AB^{-1} \leq B^{-1}$.

4.16 Let A and B be symmetric positive definite matrices with $A \leq B$. Show that

$$(I - B^{-1}A)^m B^{-1}$$

is positive definite for $m = 1, 2, \ldots$. To this end note that

$$q(XY)X = Xq(YX)$$

holds for any matrices X and Y if q is a polynomial. Which assumption may be relaxed if m is even?

4.17 Let B^{-1} be an approximate inverse of A. Moreover, assume that A and B are symmetric positive definite matrices and that

$$A \leq B.$$

Let B_m^{-1} be the approximate inverse for m steps of the iteration (1.1); cf. Problem 1.16. Show that

$$A \leq B_{m+1} \leq B_m \leq B \qquad \text{for } m \geq 1$$

by making use of the preceding problems.

§ 5. Saddle Point Problems

The determination of a minimum of

$$J(u) = \tfrac{1}{2}u'Au - f'u$$

with the constraint

$$Bu = g \tag{5.1}$$

leads to an indefinite system of equations of the form

$$\begin{aligned} Au + B^t\lambda &= f, \\ Bu \qquad\quad &= g. \end{aligned} \tag{5.2}$$

If B is an $m \times n$ matrix, then the Lagrange multiplier λ is an m-dimensional vector. Clearly, we can restrict our attention to the case where the restrictions are linearly independent.

In most cases, A is invertible. After multiplying the first equation in (5.2) by A^{-1}, we can eliminate u from the second equation:

$$BA^{-1}B^t\lambda = BA^{-1}f - g. \tag{5.3}$$

The matrix $BA^{-1}B^t$ for this so-called *reduced equation* is positive definite, although it is given only implicitly. In a paper by I. Schur [1917, p. 217] we find those submatrices that are now termed *Schur complement* of A; cf. Problem 5.9.

The Uzawa Algorithm and its Variants

A widely known iterative method for saddle point problems is connected with the name Uzawa.

5.1 The Uzawa Algorithm. Let $\lambda_0 \in \mathbb{R}^m$. Find u_k and λ_k so that

$$\left.\begin{aligned} Au_k &= f - B^t\lambda_{k-1}, \\ \lambda_k &= \lambda_{k-1} + \alpha(Bu_k - g), \end{aligned}\right\} \quad k = 1, 2, \ldots \tag{5.4}$$

Here we assume that the step size parameter α is sufficiently small. □

For the analysis of the Uzawa algorithm, we define the *residue*

$$q_k := g - Bu_k. \tag{5.5}$$

In addition, suppose the solution of the saddle point problem is denoted by (u^*, λ^*). Now substituting the iteration formula for u_k into (5.5) and using (5.3), we get

$$q_k = g - BA^{-1}(f - B^t\lambda_{k-1}) = BA^{-1}B^t(\lambda_{k-1} - \lambda^*).$$

This means that

$$\lambda_k - \lambda_{k-1} = -\alpha q_k = \alpha BA^{-1}B^t(\lambda^* - \lambda_{k-1}).$$

Thus the Uzawa algorithm is equivalent to applying the gradient method to the reduced equation using a fixed step size (cf. Problem 2.6). In particular, the iteration converges for

$$\alpha < 2\|BA^{-1}B^t\|^{-1}.$$

The convergence results of §§2 and 3 can be carried over directly. We need to use a little trick in order to get an efficient algorithm. The formula (2.5) gives the step size

$$\alpha_k = \frac{q_k' q_k}{(B^t q_k)' A^{-1} B^t q_k}.$$

However, if we were to use this rule formally, we would need an additional multiplication by A^{-1} in every step of the iteration. This can be avoided by storing an auxiliary vector. – Here we have to pay attention to the differences in the sign.

5.2 Uzawa Algorithm *(the variant equivalent to the gradient method).*
Let $\lambda_0 \in \mathbb{R}^m$ and $Au_1 = f - B^t\lambda_0$.
For $k = 1, 2, \ldots$, compute

$$\begin{aligned} q_k &= g - Bu_k, \\ p_k &= B^t q_k, \\ h_k &= A^{-1} p_k, \\ \lambda_k &= \lambda_{k-1} - \alpha_k q_k, \qquad \alpha_k = \frac{q_k' q_k}{p_k' h_k}, \\ u_{k+1} &= u_k + \alpha_k h_k. \end{aligned}$$

□

Because of the size of the condition number $\kappa(BA^{-1}B^t)$, it is often more effective to use conjugate directions. Since the corresponding factor β_k in (3.4) is already independent of the matrix of the system, the extension is immediately possible.

5.3 Uzawa Algorithm with Conjugate Directions.
Let $\lambda_0 \in \mathbb{R}^m$ and $Au_1 = f - B^t\lambda_0$. Set $d_1 = -q_1 = Bu_1 - g$.
For $k = 1, 2, \ldots$, find

$$\begin{aligned} p_k &= B^t d_k, \\ h_k &= A^{-1} p_k, \\ \lambda_k &= \lambda_{k-1} + \alpha_k d_k, \qquad \alpha_k = \frac{q_k' q_k}{p_k' h_k}, \\ u_{k+1} &= u_k - \alpha_k h_k, \\ q_{k+1} &= g - Bu_{k+1}, \\ d_{k+1} &= -q_{k+1} + \beta_k d_k, \qquad \beta_k = \frac{q_{k+1}' q_{k+1}}{q_k' q_k}. \end{aligned}$$

An Alternative

In performing k steps of Algorithms 5.2 and 5.3, $k+1$ multiplications by A^{-1} are required. Thus, there are $k+1$ equations to be solved. In practice, we do this only approximately. In particular, we approximate A^{-1} by C^{-1}, where C is considered as a preconditioner for A and is again assumed to be a symmetric positive definite matrix.

We can go one step further and replace the matrix A in the initial problem (5.1) by a matrix C which is understood to be a preconditioner. This leads to the modified minimum problem

$$\begin{gathered} \frac{1}{2} u'Cu - f'u \to \min! \\ Bu = g. \end{gathered} \tag{5.6}$$

As can be seen by carrying over (5.2), the matrix corresponding to this problem is

$$\begin{pmatrix} C & B^t \\ B & \end{pmatrix}.$$

Inserting this matrix in (1.1) in place of the matrix M, we get the iteration

$$\begin{pmatrix} u_{k+1} \\ \lambda_{k+1} \end{pmatrix} = \begin{pmatrix} u_k \\ \lambda_k \end{pmatrix} + \begin{pmatrix} C & B^t \\ B & \end{pmatrix}^{-1} \begin{pmatrix} f - Au_k - B^t\lambda_k \\ g - Bu_k \end{pmatrix}. \tag{5.7}$$

The rate of convergence of this iteration is determined by the gap between the upper and lower bounds on the quotients $\frac{u'Au}{u'Cu}$, $u \neq 0$; cf. (4.5). In fact, it suffices to examine the bounds for the subspace $V = \{u \in \mathbb{R}^n;\ Bu = 0\}$. (They can of course be estimated by the coarser bounds for $\mathbb{R}^n$.)

In view of the following (cf. Braess and Sarazin [1997], Zulehner [2000]), the iteration (5.4) of Uzawa and the iteration (5.7) are extreme cases. If the iteration (5.7) is built into a cg-iteration, then u-variables and the Lagrange multipliers have to be treated in a different way; see Braess, Deuflhard, and Lipnikov [2002].

5.4 Remark. In the Uzawa algorithm (5.4), u_{k+1} and λ_{k+1} are independent of u_k. In the iteration (5.7), u_{k+1} and λ_{k+1} are independent of λ_k.

The assertion about the Uzawa algorithm follows directly from the definition (5.4) of the algorithm. The other assertion is a consequence of the following formula which is equivalent to (5.7):

$$\begin{pmatrix} u_{k+1} \\ \lambda_{k+1} \end{pmatrix} = \begin{pmatrix} C & B^t \\ B & \end{pmatrix}^{-1} \begin{pmatrix} f - (A - C)u_k \\ g \end{pmatrix}. \tag{5.8}$$

Bramble and Pasciak [1988] took a completely different approach. By employing a different metric for the indefinite problem, they were able to get a preconditioning in almost the same way as in the positive definite case.

Problems

5.5 Consider the special case $A = I$, and compare the condition number of $BA^{-1}B^t$ with that of the squared matrix. In particular, show that the Uzawa algorithm is better than the gradient method for the squared matrix.

5.6 For the case $m \ll n$, the restriction can be eliminated indirectly. Let F be an $m \times m$ matrix with $FF^t = BB^t$, e.g., say F stems from the Cholesky decomposition of BB^t. In the special case $A = I$, we have the triangular decomposition:

$$\begin{pmatrix} I & \\ B & F \end{pmatrix} \begin{pmatrix} I & B^t \\ & -F^t \end{pmatrix} = \begin{pmatrix} I & B^t \\ B & \end{pmatrix}.$$

How can we construct a corresponding triangular decomposition for the matrix in (5.2) if a decomposition $A = L^t L$ is known?

5.7 Show that $\kappa(BA^{-1}B^t) \leq \kappa(A)\kappa(BB^t)$.

5.8 For the saddle point problem (5.2), the norm $\|\cdot\|_A$ is obviously the natural norm for the u components. Show that the norm $\|\cdot\|_{BA^{-1}B^t}$ is then the natural one for the λ components in the following sense: the inf-sup condition holds for the mapping $B^t : \mathbb{R}^m \mapsto \mathbb{R}^n$ with the constant $\beta = 1$.

5.9 Verify the block Cholesky decomposition for the matrix

$$\begin{pmatrix} A & B^t \\ B & 0 \end{pmatrix} = \begin{pmatrix} A & 0 \\ B & I \end{pmatrix} \begin{pmatrix} A^{-1} & 0 \\ 0 & -BA^{-1}B^t \end{pmatrix} \begin{pmatrix} A & B^t \\ 0 & I \end{pmatrix}$$

appearing in the saddle point problem. What is the connection between this factorization and the computation of the reduced equation (5.3)? In addition, prove that the inverse has the following decomposition:

$$\begin{pmatrix} A & B^t \\ B & 0 \end{pmatrix}^{-1} = \begin{pmatrix} A^{-1} - A^{-1}B^t S^{-1} B A^{-1} & A^{-1}B^t S^{-1} \\ S^{-1}BA^{-1} & -S^{-1} \end{pmatrix},$$

where $S = BA^{-1}B^t$ is the Schur complement.