Iterative Methods

Summer semester 2008

Part II of "Software for Numerical Linear Algebra"

Martin H. Gutknecht

ETH Zurich

© Martin H. Gutknecht, 2006

Contents

Chapter 1

Some Basic Ideas

1.1 Sparse Matrices

Large **sparse linear systems of equations** [*dünn besetzte lineare Gleichungssysteme*] or **sparse matrix eigenvalue problems** [*dünn besetzte Matrix-Eigenwertprobleme*] appear in most applications of scientific computing. In particular, the discretization of partial differential equations with the finite element method (FEM) or with the (older) finite difference method (FDM) leads to such problems:

$$\boxed{\text{PDEs}} \quad \overset{\text{discretization/linearization}}{\Longrightarrow} \quad \boxed{\begin{array}{c}\mathbf{A}\mathbf{x}=\mathbf{b}\\ \text{or}\\ \mathbf{A}\mathbf{x}=\mathbf{x}\lambda\end{array}}$$

"Sparse" refers to $\mathbf{A}$ being a **sparse matrix** [*dünn besetzte Matrix*] and means that most of the elements of $\mathbf{A}$ are 0. In a simple, physically one-dimensional problem, there may be as few as three nonzeros per row; in a complex, physically three-dimensional problem, there may be as many as a hundred. Often, when PDEs are solved, most computer time is spent for repeatedly solving a linear system or an eigenvalue problem. In addition to PDE based problem, more and more from other applications are encountered.

While in 1950 "large" meant that the $N \times N$ matrix $\mathbf{A}$ had, say, $30 < N < 100$, nowadays it means something like $1000 \leq N \leq 100'000'000$, and the upper bound keeps growing.

Variations of the two problems $\mathbf{A}\mathbf{x} = \mathbf{b}$ and $\mathbf{A}\mathbf{x} = \mathbf{x}\lambda$ appear too: *e.g.*, sparse least squares or generalized eigenvalue problems.

Sparse matrices are stored in appropriate data formats, which avoid to store the zero elements. There are many different such formats; see, *e.g.*, Saad (1996), Sect. 3.4. Three examples are:

- Matlab format: row-column index pairs, values of elements.
- Harwell-Boeing format (`*.rsa`)
- MatrixMarket format (`*.mtx`)

Examples of sparse matrices have been collected for numerical experiments. The two best known collections are:

- *Matrix Market:*
  `http://math.nist.gov/MatrixMarket`
- *University of Florida Sparse Matrix Collection (Tim Davis):*
  `http://www.cise.ufl.edu/research/sparse/matrices`

Matlab provides commands that extend many operations and functions to sparse matrices.

- Help command: `help sparfun`
- Directory of m-files: `matlab/toolbox/matlab/sparfun`
- Examples of "sparse Matlab" commands:

`sparse`	converts full to sparse matrix
`full`	converts sparse to full matrix
`speye`	sparse unit matrix
`spy`	visualizes the sparsity pattern of a sparse matrix
`+, -, *`	also applicable to sparse matrices
`\ , /`	also for solving a linear system with sparse matrix

In the iterative methods discussed here $\mathbf{A}$ is only needed to compute $\mathbf{Ay}$ for any $\mathbf{y} \in \mathbb{R}^N$ (or any $\mathbf{y} \in \mathbb{C}^N$ if the matrix is complex). Thus, $\mathbf{A}$ may be given as a procedure/function

$$\mathbf{A} : \mathbf{y} \mapsto \mathbf{Ay}\,.$$

We will refer to this operation as a **matrix-vector multiplication (MV)** [*Matrix-Vektor-Multiplikation*], although in practice the required computation may be much more complicated than multiplying a sparse matrix with a vector; in general, $\mathbf{Ay}$ is just an abbreviation for finding the result of a possibly lengthy calculation (which, *e.g.*, may include solving certain other linear systems).

Example 1.1. One of the simplest examples of a boundary value problem is

$$\begin{aligned} u'' &= f \quad \text{on} \quad (0,1), \\ u(0) &= u(1) = 0\,. \end{aligned} \tag{1.1}$$

In one interpretation u is the temperature in a homogeneous rod under the condition of a steady state distribution of heat. The ends of the rod are kept at temperature 0, and there is at position x an external heat source $-f(x)$; see, *e.g.*, Strang (1986), p. 160.

The finite difference method assumes a (one-dimensional) uniform grid $\{x_j :\equiv j * h \mid j = 0, \dots, M+1\}$ of grid size $h :\equiv 1/(M+1)$ and makes use of the approximation

$$u''(x_j) = \frac{1}{h^2}\,(u_{j+1} - 2u_j + u_{j-1}) + O(h^2)\,, \tag{1.2}$$

where u_j is the approximate value of $u(x_j)$. After setting additionally $f_j :\equiv f(x_j)$ we are lead to the linear system

$$\underbrace{\begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & -1 & 2 & \ddots & \\ & & \ddots & \ddots & -1 \\ & & & -1 & 2 \end{pmatrix}}_{\equiv:\,\mathbf{T}} \underbrace{\begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ \vdots \\ u_M \end{pmatrix}}_{\equiv:\,\mathbf{u}} = -\,h^2 \underbrace{\begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ \vdots \\ f_M \end{pmatrix}}_{\equiv:\,\mathbf{f}}\,. \tag{1.3}$$

Here the $M \times M$ system matrix $\mathbf{T}$ is symmetric tridiagonal and can be seen to be positive definite. (It is weakly diagonally dominant.) The system could be solved easily by Gaussian elimination, that is by LU factorization, by LDL$^\mathsf{T}$ factorization, or by Cholesky factorization. The cost of such a factorization is only $\mathcal{O}(M)$ since the factors are bidiagonal. There exist yet other efficient recursive algorithms for solving such a system, for example, cyclic reduction or fast Toeplitz solvers, since $\mathbf{T}$ has also Toeplitz structure (*i.e.*, t_{ij} only depends on $i-j$).

The corresponding two-dimensional example is the **Poisson problem**

$$\begin{aligned} \Delta u &= f \qquad \text{on} \quad S :\equiv (0,1)\times(0,1)\,, \\ u(x,y) &= 0 \qquad \text{on} \quad \partial S\,, \end{aligned} \tag{1.4}$$

where

$$\Delta u :\equiv \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\,.$$

Here ∂S denotes the boundary of the unit square S. In analogy to (1.2) we apply the approximation

$$\Delta u(x_j, y_j) = \frac{1}{h^2}\left[(u_{j+1} - 2u_j + u_{j-1}) + (u_{j+M} - 2u_j + u_{j-M})\right] + O(h^2)\,, \tag{1.5}$$

where we assume that the $N :\equiv M^2$ interior points are numbered row by row from the upper left corner to the lower right one. This transforms (1.4) into the linear system

$$\mathbf{A}\mathbf{u} = -\,h^2\mathbf{f} \tag{1.6}$$

with the block matrix $\mathbf{A}$ of size $N \times N = M^2 \times M^2$ defined by

$$\mathbf{A} :\equiv \begin{pmatrix} \mathbf{T} & -\mathbf{I} & \mathbf{O} & \cdots & \cdots & \mathbf{O} \\ -\mathbf{I} & \mathbf{T} & -\mathbf{I} & \ddots & & \vdots \\ \mathbf{O} & -\mathbf{I} & \mathbf{T} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & -\mathbf{I} & \mathbf{O} \\ \vdots & & \ddots & -\mathbf{I} & \mathbf{T} & -\mathbf{I} \\ \mathbf{O} & \cdots & \cdots & \mathbf{O} & -\mathbf{I} & \mathbf{T} \end{pmatrix}. \tag{1.7}$$

Here, $\mathbf{I}$ is the $M \times M$ unit matrix and

$$\mathbf{T} :\equiv \begin{pmatrix} 4 & -1 & & & \\ -1 & 4 & -1 & & \\ & -1 & 4 & \ddots & \\ & & \ddots & \ddots & -1 \\ & & & -1 & 4 \end{pmatrix}$$

is also of size $M \times M$. The system matrix $\mathbf{A}$ is still symmetric positive definite and banded, but the total bandwidth is now $2M+1$. Clearly, in three dimensions we would have $N = M^3$ and obtain a matrix with total bandwidth $2M^2+1$ and with at most seven nonzeros per row, namely one number six and at most six minus ones.

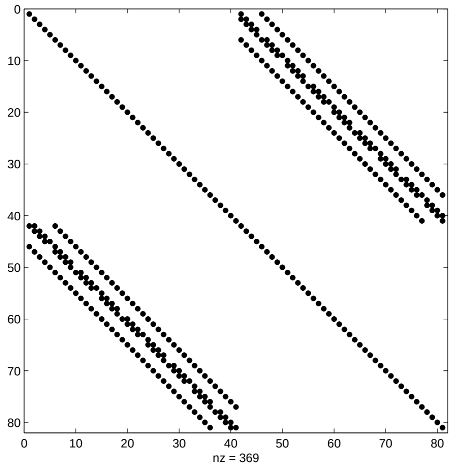
Note that the matrices depend on the numbering or order of the points. For example, if in the first problem (1.3) we take all the currently odd numbered points first, followed by all even numbered points — a one-dimensional version of what is called **red-black ordering** — the matrix $\mathbf{T}$ is transformed into

$$\widetilde{\mathbf{T}} :\equiv \begin{pmatrix} 2\,\mathbf{I} & -\mathbf{B}_1 \\ -\mathbf{B}_1^{\mathsf{T}} & 2\,\mathbf{I} \end{pmatrix}, \tag{1.8}$$

where $\mathbf{B}_1$ is a lower bidiagonal matrix of ones of size $\frac{1}{2}M$ if M is even. Likewise, by using red-black ordering in the two-dimensional problem (1.6) and assuming M odd we can replace $\mathbf{A}$ by

$$\widetilde{\mathbf{A}} :\equiv \begin{pmatrix} 4\mathbf{I}_1 & -\mathbf{B}_2 \\ -\mathbf{B}_2^{\mathsf{T}} & 4\mathbf{I}_2 \end{pmatrix}, \tag{1.9}$$

where $\mathbf{I}_1$ is the unit matrix of size[1] $\lceil \frac{1}{2}M^2 \rceil$ and $\mathbf{I}_2$ is the unit matrix of size $\lfloor \frac{1}{2}M^2 \rfloor$, while $\mathbf{B}_2$ is a (non-square) Toeplitz matrices with four co-diagonals of ones (but not all next to each other). The nonzeros of $\widetilde{\mathbf{A}}$ are shown in the following "spy plot" (with $M = 9$, $N = 81$):



It was generated with the following Matlab commands:

$M = 9;\ N = M^2;$
$A = gallery('poisson', M);$
$p = [1:2:N, 2:2:N];$
$Atilde = A(p,p);$
$spy(Atilde)$ ♦

In most applications the data are real, *i.e.*, $A \in \mathbb{R}^{N\times N}$ and $b \in \mathbb{R}^N$. But, *e.g.*, in electrical engineering there are also problems where $A \in \mathbb{C}^{N\times N}$ and $b \in \mathbb{C}^N$. Most of what we say is true for both cases. To handle them at once we will write $A \in \mathbb{E}^{N\times N}$, $b \in \mathbb{E}^N$ with $\mathbb{E} :\equiv \mathbb{R}$ or $\mathbb{C}$, and use $\mathbf{x}^\star :\equiv \mathbf{x}^{\mathsf{T}}$ if $\mathbb{E} = \mathbb{R}$ and $\mathbf{x}^\star :\equiv \mathbf{x}^{\mathsf{H}}$ if $\mathbb{E} = \mathbb{C}$.

Except in the sections on least squares we will assume that $\mathbf{A}$ is nonsingular. In many applications $\mathbf{A}$ is real symmetric or Hermitian and positive definite (or, briefer: **spd** or **Hpd**) — in which case all the eigenvalues are positive, and there is an orthonormal basis of eigenvectors. We will see that solving the linear system $\mathbf{Ax} = \mathbf{b}$ is then much easier than when $\mathbf{A}$ is indefinite or even nonsymmetric.

[1] $\lceil y \rceil$ denotes the smallest integer larger or equal to y, and $\lfloor y \rfloor$ denotes the largest integer smaller or equal to y.

1.2 Sparse Direct Methods

In connection with the solution of sparse linear systems the term "**direct method**" is used for all methods that are in a wide sense variations of Gaussian elimination or — what is essentially the same — **LU decomposition** (or **LU factorization**) :

$$\boxed{\mathbf{PA} = \mathbf{LU}\,.}$$

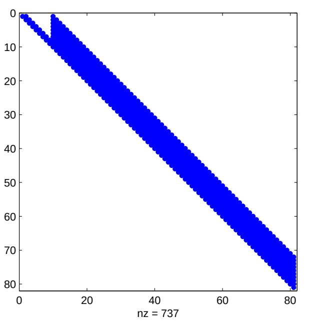
Here, $\mathbf{P}$ is a permutation matrix representing possible row interchanges, and the matrices $\mathbf{L}$ and $\mathbf{U}$ are lower and upper triangular, respectively.

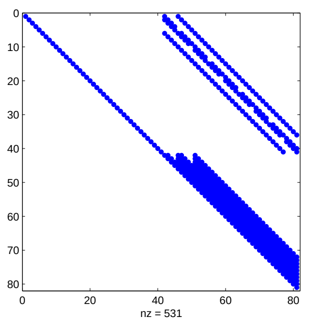
The cost of such a decomposition is roughly

$\sim \frac{2}{3}N^3$	FLOPS	in general
$\sim 2b^2N$	FLOPS	if $\mathbf{A}$ banded, bandwidth $b \ll N$
reduced by $\sim 50\%$		if $\mathbf{A}$ sym. pos. def.

These variations of Gauss elimination for sparse linear systems are called **sparse direct methods** or **sparse direct solvers**. But the more $\mathbf{A}$ differs from a narrowly banded matrix, the less effective they are due to **fill-in**: the matrix $\mathbf{L} + \mathbf{R}$ may contain many more nonzeros than $\mathbf{A}$.

If the original physical problem is 2- or 3-dimensional, $\mathbf{A}$ will be far from being narrowly banded. For example, the Cholesky factors of the matrices $\mathbf{A}$ and $\widetilde{\mathbf{A}}$ of the previous section have the following nonzero patterns:





The ordering of equations and unknowns effects the fill-in, and can be used to reduce it. There exist several well established algorithms to achieve this. But pivoting complicates matters further.

In the last twenty years enormous effort has been spent for vectorizing and parallelizing sparse direct solvers; see, *e.g.*, the program PARDISO by Schenk (2000).

Also in Matlab a sparse direct solver is implemented. It is called whenever a linear system $\mathbf{Ax} = \mathbf{b}$ or $\mathbf{w}^\mathsf{T}\mathbf{A} = \mathbf{c}^\mathsf{T}$ with a sparse matrix $\mathbf{A}$ is solved by writing `x = A\b` or `w' = c'/A`, respectively.

Freely available software for sparse direct solvers and for some other linear algebra tasks can be found on

`http://www.netlib.org/utk/people/JackDongarra/la-sw.html`

1.3 Fixed Point and Jacobi Iteration

Pure mathematicians often write nonlinear equations in fixed point form as $\mathbf{x} = \mathbf{\Phi}(\mathbf{x})$, and solve them under the assumption that $\mathbf{\Phi}$ is a contraction by the so-called **Picard iteration** or **fixed point iteration** [*Fixpunkt-Iteration*] $\mathbf{x}_{n+1} := \mathbf{\Phi}(\mathbf{x}_n)$. This idea is also applicable to a linear system $\mathbf{Ax} = \mathbf{b}$: we rewrite it in fixed point form as

$$\boxed{\mathbf{x} = \mathbf{Bx} + \mathbf{b}} \qquad \text{with} \qquad \boxed{\mathbf{B} :\equiv \mathbf{I} - \mathbf{A}} \tag{1.10}$$

and apply fixed point iteration: starting with some $\mathbf{x}_0$, we compute for $n = 0, 1, 2, \dots$

$$\boxed{\mathbf{x}_{n+1} := \mathbf{Bx}_n + \mathbf{b}\,.} \tag{1.11}$$

Here, $\mathbf{x}_n$ is the nth approximation of the fixed point $\mathbf{x}_\star$ satisfying $\mathbf{x}_\star = \mathbf{Bx}_\star + \mathbf{b}$, *i.e.*, $\mathbf{Ax}_\star = \mathbf{b}$; it is also called nth **iterate** [*Iterierte*].

If $a_{k,k} = 1$ $(\forall k)$, *i.e.*, $b_{k,k} = 0$ $(\forall k)$, then iteration (1.11) is called **Jacobi iteration** [*Gesamtschritt-Verfahren*]. Note that $a_{k,k} = 1$ can be achieved whenever $a_{k,k} \neq 0$ by scaling the kth equation.

In the linear case, $\mathbf{\Phi}\,:\ \mathbf{x} \mapsto \mathbf{Bx}$ is called a **contraction** [*Kontraktion*] if $\|\mathbf{B}\| < 1$ in some norm. Then the linear fixed point iteration converges globally, that is, for every $\mathbf{x}_0$. The proof is left as an easy exercise. But we can establish a sharper result because the underlying space is finite dimensional:

THEOREM 1.3.1 *For the (linear) fixed point iteration* (1.11) *holds*

$$\boxed{\mathbf{x}_n \to \mathbf{x}_\star \ \textit{for any}\ \mathbf{x}_0 \quad \Longleftrightarrow \quad \rho(\mathbf{B}) < 1\,,} \tag{1.12}$$

where $\rho(\mathbf{B}) :\equiv \max\{|\lambda| \bigm| \lambda \ \textit{eigenvalue of}\ \mathbf{B}\}$ *is the* ***spectral radius*** *of* $\mathbf{B}$.

PROOF. Let $\mathbf{BV} = \mathbf{V\Lambda}$ be an eigenvalue decomposition of $\mathbf{B}$, so that $\mathbf{\Lambda}$ is diagonal or, in general, the **Jordan canonical form** [*Jordansche Normalform*] of $\mathbf{B}$. Clearly,

$$\mathbf{x}_n - \mathbf{x}_\star = \mathbf{B}(\mathbf{x}_{n-1} - \mathbf{x}_\star) = \mathbf{B}^n(\mathbf{x}_0 - \mathbf{x}_\star)\,,$$

so convergence occurs if and only if $\mathbf{B}^n \to \mathbf{O}$ as $n \to \infty$. This, in turn, happens if and only if $\mathbf{\Lambda}^n \to \mathbf{O}$ as $n \to \infty$.

If $\mathbf{\Lambda}$ is diagonal, the equivalence (1.12) clearly follows. But in general, $\mathbf{\Lambda}$ is block diagonal and may contain bidiagonal **Jordan blocks** [*Jordansche Blöcke*] like

$$\mathbf{J}_k :\equiv \begin{pmatrix} \lambda_k & 1 & 0 & \cdots & 0 \\ 0 & \lambda_k & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & \lambda_k & 1 \\ 0 & \cdots & \cdots & 0 & \lambda_k \end{pmatrix}. \tag{1.13}$$

It remains to show that $\mathbf{J}_k^n \to \mathbf{O}$ as $n \to \infty$. Assume $\mathbf{J}_k$ is of size $m_k \times m_k$. We write it as $\mathbf{J}_k = \lambda_k \mathbf{I} + \mathbf{S}$ and note that $\mathbf{S}^\ell$ is zero except for ones on the ℓth upper codiagonal. In particular, $\mathbf{S}^\ell = \mathbf{O}$ when $\ell \geq m_k$. Therefore, if $n \geq m_k - 1$,

$$\mathbf{J}_k^n = (\lambda_k \mathbf{I} + \mathbf{S})^n = \sum_{i=0}^{n} \binom{n}{i} \lambda_k^{n-i} \mathbf{S}^i = \sum_{i=0}^{m_k-1} \binom{n}{i} \lambda_k^{n-i} \mathbf{S}^i \,. \tag{1.14}$$

In this finite sum, in view of $|\lambda_k| < 1$, in each term we have, as $n \to \infty$,

$$\binom{n}{i} \lambda_k^{n-i} = \frac{n(n-1)\cdot(n-i+1)}{i!}\, \lambda_k^{n-i} \to 0\,.$$

Consequently, $\mathbf{J}_k^n \to \mathbf{O}$ as $n \to \infty$. □

REMARK. A matrix norm is called **compatible** [*kompatibel*] with a given vector norm if $\|\mathbf{By}\| \leq \|\mathbf{B}\|\,\|\mathbf{y}\|$ for any square matrix $\mathbf{B}$ and any vector $\mathbf{y}$ of matching size. For a compatible matrix norm we have $\|\mathbf{B}\| \geq \rho(\mathbf{B})$ as is seen by choosing for $\mathbf{y}$ an eigenvector corresponding to an eigenvalue of largest absolute value, so that $\|\mathbf{By}\| = \rho(\mathbf{B})\,\|\mathbf{y}\|$. So, $\rho(\mathbf{B}) < 1$ if $\|\mathbf{B}\| < 1$ for some norm. ▼

Theorem 1.3.1 can be recast as follows: *linear fixed point iteration in $\mathbb{E}^N$ is globally convergent if and only if* $\rho(\mathbf{B}) < 1$. By a careful analysis of its proof we can also specify the speed of convergence[2].

What can we say about the rate of convergence?

THEOREM 1.3.2 *If $\rho(\mathbf{B}) < 1$ then, for any $\mathbf{x}_0$, the iterates of the linear fixed point iteration* (1.11) *satisfy*

$$\boxed{\limsup_{n\to\infty} \|\mathbf{x}_n - \mathbf{x}_\star\|^{1/n} \leq \rho(\mathbf{B})\,,} \tag{1.15}$$

and for some $\mathbf{x}_0$ equality holds.
So, if $0 < \rho(\mathbf{B}) < 1$, linear fixed point iteration converges linearly with a root-convergence factor of at most $\rho(\mathbf{B})$, while if $\rho(\mathbf{B}) = 0$, it converges superlinearly.

PROOF. We have, as in the proof of Theorem 1.3.1,

$$\|\mathbf{x}_n - \mathbf{x}_\star\|^{1/n} = \|\mathbf{V}\boldsymbol{\Lambda}^n\mathbf{V}^{-1}(\mathbf{x}_0 - \mathbf{x}_\star)\|^{1/n} \leq \|\mathbf{V}\|^{1/n} \|\boldsymbol{\Lambda}^n \mathbf{w}_0\|^{1/n}\,,$$

where $\mathbf{w}_0 :\equiv \mathbf{V}^{-1}(\mathbf{x}_0 - \mathbf{x}_\star)$. Since $\|\mathbf{V}\|^{1/n} \to 1$, all depends on $\boldsymbol{\Lambda}^n$, whose diagonal blocks are eigenvalues or Jordan blocks $\mathbf{J}_k$. The powers

[2] A sequence $\{\mathbf{x}_n\} \subset \mathbb{C}^N$ is said to converge R-linearly to $\mathbf{x}_\star$ if the **R-factor** or **root-convergence factor** [*Wurzel-Konvergenzfaktor*]

$$\kappa_R\{\mathbf{x}_n\} :\equiv \limsup_{n\to\infty} \|\mathbf{x}_n - \mathbf{x}_\star\|^{1/n}$$

lies in $(0, 1)$. An iterative process $\mathcal{J}$ that generates a nonempty set $\mathcal{C}(\mathcal{J}, \mathbf{x}_\star)$ of sequences that converge R-linearly to the limit point $\mathbf{x}_\star$ has R-factor

$$\kappa_R(\mathcal{J}) :\equiv \sup\{R\{\mathbf{x}_n\} \mid \{\mathbf{x}_n\} \in \mathcal{C}(\mathcal{J}, \mathbf{x}_\star)\}\,.$$

of the latter were considered in (1.14). If we write, with $\rho :\equiv \rho(\mathbf{B})$,

$$\mathbf{J}_k^n = \rho^n \sum_{i=0}^{m_k-1} \binom{n}{i} \lambda_k^{-i} \left(\frac{\lambda_k}{\rho}\right)^n \mathbf{S}^i \,, \tag{1.16}$$

it is evident that only eigenvalues of maximum absolute value count, because for the others the nth power of the fraction λ_k/ρ tends to zero. If $|\lambda_k| = \rho$, the dominant term in the sum (1.16) is

$$\binom{n}{m_k - 1} \lambda_k^{n-m_k+1} \mathbf{S}^{m_k-1} \,.$$

There may be several such terms. In any case, $\|\mathbf{\Lambda}^n \mathbf{w}_0\|^{1/n}$ behaves asymptotically at worst like the nth root of the absolute value of such a term, hence like $|\lambda_k| = \rho$. So this is asymptotically a bound for $\|\mathbf{x}_n - \mathbf{x}_\star\|^{1/n}$. To see that this bound is sharp we just have to choose $\mathbf{x}_0$ so that $\mathbf{w}_0$ is an eigenvector of $\mathbf{\Lambda}$ for a dominant eigenvalue λ_k. $\square$

The applicability of fixed point or Jacobi iteration in practice is quite limited since, if $\rho(\mathbf{B}) < 1$ holds at all, then typically only with $\rho(\mathbf{B})$ nearly 1, so that convergence is very slow. But we need a good approximate solution in $n \ll N$ steps.

Example 1.2. We continue Example 1.1: it can be seen that the $M \times M$ matrix $\mathbf{T}$ in (1.3) has the eigenvalues

$$\lambda_j = 2 - 2\,\cos \frac{j\pi}{M+1} = 4\,\sin^2 \frac{j\pi}{2(M+1)} = 4\,\sin^2 \frac{hj\pi}{2} \,.$$

Thus, when we divide every equation by 2, so that $a_{k,k} = \frac{1}{2}\, t_{k,k} = 1$ and $b_{k,k} = 1 - a_{k,k} = 0$, then

$$\rho(\mathbf{B}) = \rho(\mathbf{I} - \tfrac{1}{2}\mathbf{T}) = \cos \frac{\pi}{M+1} = \cos h\pi \,. \tag{1.17}$$

For example, when $M = 100$, then $\rho(\mathbf{B}) = 0.9995\,1628...$

In order to reduce the error by a factor of 10, we need 4760 iterations!

It helps to use a courser grid: for $M = 10$ we have $\rho(\mathbf{B}) = 0.9594\,9297..$, and "only" 56 iterations are needed to reduce the error by a factor of 10, but of course this is still excessive for a 10×10 system.

To verify (1.17) with Matlab, $\mathbf{B}$ and its spectral radius can be found by

```
B = gallery('tridiag', M, 0.5, 0, 0.5);
rhoB = norm(full(B))
```

or better by

```
B = gallery('tridiag', M, 0.5, 0, 0.5);
svd6 = svds(B); rhoB = svd6(1)
```

♦

Since we cannot compute the nth **error (vector)** [*Fehler(vektor)*]

$$\boxed{\mathbf{d}_n :\equiv \mathbf{x}_n - \mathbf{x}_\star} \tag{1.18}$$

for checking the convergence of an iteration, *i.e.*, for checking the "quality" of the approximate solution (or, iterate) $\mathbf{x}_n$, we use the nth **residual (vector)** [*Residuum, Residuenvektor*]

$$\boxed{\mathbf{r}_n :\equiv \mathbf{b} - \mathbf{A}\mathbf{x}_n \, .} \tag{1.19}$$

Note that

$$\mathbf{r}_n = -\mathbf{A}(\mathbf{x}_n - \mathbf{x}_\star) = -\mathbf{A}\mathbf{d}_n \, . \tag{1.20}$$

For the linear fixed point iteration we have

$$\mathbf{r}_n = \mathbf{b} - \mathbf{A}\mathbf{x}_n = \mathbf{B}\mathbf{x}_n + \mathbf{b} - \mathbf{x}_n = \mathbf{x}_{n+1} - \mathbf{x}_n \, ,$$

so we can rewrite it as

$$\boxed{\mathbf{x}_{n+1} := \mathbf{x}_n + \mathbf{r}_n \, ,} \tag{1.21}$$

and, by multiplying it by $-\mathbf{A}$, we obtain a recursion for the residual,

$$\boxed{\mathbf{r}_{n+1} := \mathbf{r}_n - \mathbf{A}\mathbf{r}_n = \mathbf{B}\mathbf{r}_n \, .} \tag{1.22}$$

We see that we can compute the residual $\mathbf{r}_n$ either according to the definition (1.19) or by using the recursion (1.22). In either case, we need one matrix-vector multiplication. Once, $\mathbf{r}_n$ is known, the new iterate $\mathbf{x}_{n+1}$ is obtained without any matrix-vector multiplication from (1.21). Mathematically both ways of computing $\mathbf{r}_n$ are equivalent, but roundoff errors may cause the results to differ.

Assignment (1.21) is a typical update formula for the iterate: the new approximation of the solution is obtained by adding a correction, here $\mathbf{r}_n$, to the old one.

From (1.22) it follows by induction that

$$\mathbf{r}_n = p_n(\mathbf{A})\mathbf{r}_0 \in \mathsf{span}\ \{\mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \dots, \mathbf{A}^n\mathbf{r}_0\} \, , \tag{1.23}$$

where p_n is a polynomial of exact degree n, actually $p_n(\zeta) = (1-\zeta)^n$. Moreover, from (1.21) we conclude that

$$\begin{aligned} \mathbf{x}_n &= \mathbf{x}_0 + \mathbf{r}_0 + \cdots + \mathbf{r}_{n-1} && \text{(1.24a)} \\ &= \mathbf{x}_0 + q_{n-1}(\mathbf{A})\mathbf{r}_0 && \text{(1.24b)} \\ &\in \mathbf{x}_0 + \mathsf{span}\ \left\{\mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \dots, \mathbf{A}^{n-1}\mathbf{r}_0\right\} && \text{(1.24c)} \end{aligned}$$

with a polynomial q_{n-1} of exact degree $n-1$. We note that here $\mathbf{x}_0 + \mathsf{span}\ \{\mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \dots, \mathbf{A}^{n-1}\mathbf{r}_0\}$ is an affine space, *i.e.*, a linear subspace shifted by the translation $\mathbf{x}_0$.

Building up $\mathbf{x}_n = \mathbf{x}_0 + q_{n-1}(\mathbf{A})\mathbf{r}_0$ and $\mathbf{r}_n = p_n(\mathbf{A})\mathbf{r}_0$ requires in particular a total of $n+1$ matrix-vector multiplications and this is the main work of the whole iteration process (unless $\mathbf{A}$ is extremely sparse). With roughly the same work we can construct any other vector $\mathbf{x}_n$ in the same affine space and its corresponding residual. We may hope that by making a better choice in this space we will find a sequence $(\mathbf{x}_n)$ that converges faster.

1.4 Iterations Based on Matrix Splittings

We introduced the Jacobi iteration as fixed point iteration (1.11) with a matrix $\mathbf{B}$ whose diagonal elements are zero. If we start from an arbitrary nonsingular system $\mathbf{Ax} = \mathbf{b}$ and let $\mathbf{D}$ be the diagonal matrix with the diagonal of $\mathbf{A}$, we thus replace the system by

$$\boxed{\mathbf{x} = \widehat{\mathbf{B}}\mathbf{x} + \widehat{\mathbf{b}}} \quad \text{with} \quad \boxed{\widehat{\mathbf{B}} :\equiv \mathbf{I} - \mathbf{D}^{-1}\mathbf{A}\,, \quad \widehat{\mathbf{b}} :\equiv \mathbf{D}^{-1}\mathbf{b}} \qquad (1.25)$$

and apply the fixed point iteration $\mathbf{x}_{n+1} := \widehat{\mathbf{B}}\mathbf{x}_n + \widehat{\mathbf{b}}$.

Written in terms of the components of $\mathbf{x}_n \equiv: \begin{pmatrix} x_1^{(n)} & \dots & x_N^{(n)} \end{pmatrix}^{\mathsf{T}}$ one step — or, as it is often called, one **sweep** — of Jacobi iteration becomes

$$x_j^{(n+1)} := \frac{1}{a_{jj}} \left(b_j - \sum_{k=1}^{j-1} a_{jk}\, x_k^{(n)} - \sum_{k=j+1}^{N} a_{jk}\, x_k^{(n)} \right) , \quad j = 1, \dots, N\,. \qquad (1.26)$$

It seems that Gauss was the one who discovered that the convergence is usually improved if we replace in the first sum of (1.26) the old values $x_n^{(k)}$ by the already computed new values $x_{n+1}^{(k)}$:

$$x_j^{(n+1)} := \frac{1}{a_{jj}} \left(b_j - \sum_{k=1}^{j-1} a_{jk}\, x_k^{(n+1)} - \sum_{k=j+1}^{N} a_{jk}\, x_k^{(n)} \right) , \quad j = 1, \dots, N\,. \qquad (1.27)$$

This is called **Gauss-Seidel method** [*Gauss-Seidel-Verfahren oder Einzelschrittverfahren*]. To recast it in matrix notation we write $\mathbf{A}$ as

$$\mathbf{A} = \mathbf{D} - \mathbf{E} - \mathbf{F}\,, \qquad (1.28)$$

where $\mathbf{E}$ and $\mathbf{F}$ are strictly lower and upper triangular, respectively. Then (1.27) becomes

$$\mathbf{D}\mathbf{x}_{n+1} := \mathbf{E}\mathbf{x}_{n+1} + \mathbf{F}\mathbf{x}_n + \mathbf{b}$$

or

$$\boxed{(\mathbf{D} - \mathbf{E})\mathbf{x}_{n+1} := \mathbf{F}\mathbf{x}_n + \mathbf{b}\,,} \qquad (1.29)$$

which can be brought into the form of fixed point iteration with

$$\boxed{\widehat{\mathbf{B}} :\equiv (\mathbf{D} - \mathbf{E})^{-1}\mathbf{F}\,, \qquad \widehat{\mathbf{b}} :\equiv (\mathbf{D} - \mathbf{E})^{-1}\mathbf{b}\,.} \qquad (1.30)$$

Of course, we never intend to compute the lower triangular matrix $(\mathbf{D} - \mathbf{E})^{-1}$, but implement this recursion as the sweep (1.27). The representation as fixed point iteration tells us that for a convergence analysis we have to determine the spectral radius of $\widehat{\mathbf{B}}$.

In the notation (1.28) Jacobi iteration takes the form

$$\boxed{\mathbf{x}_{n+1} := \mathbf{D}^{-1}(\mathbf{E} + \mathbf{F})\mathbf{x}_n + \mathbf{D}^{-1}\mathbf{b} = \mathbf{x}_n + \mathbf{D}^{-1}\mathbf{r}_n} \qquad (1.31)$$

and has the iteration matrix

$$\boxed{\widehat{\mathbf{B}} :\equiv \mathbf{D}^{-1}(\mathbf{E}+\mathbf{F})\,.} \tag{1.32}$$

A classical idea is to apply here is that of **relaxation** [*Relaxation*]: we multiply the correction $\mathbf{D}^{-1}\mathbf{r}_n$ in the Jacobi iteration (1.31) by a **relaxation factor** [*Relaxationsfaktor*] ω, which typically satisfies $0 < \omega < 1$:

$$\boxed{\mathbf{x}_{n+1} := \mathbf{x}_n + \mathbf{D}^{-1}\mathbf{r}_n\omega\,.} \tag{1.33}$$

Naturally this method is called the **damped Jacobi iteration** [*gedämpftes Gesamtschritt-Verfahren*]. It has also been referred to as **Jacobi overrelaxation (JOR) method** or as **stationary Richardson iteration**. (In general, Richardson iteration is nonstationary, that is, $\omega :\equiv \omega_n$ depends on n.)

Note that

$$\begin{aligned}\mathbf{x}_{n+1} &= \mathbf{x}_n + \mathbf{D}^{-1}\left[\mathbf{b} - \mathbf{A}\mathbf{x}_n\right]\omega \\ &= \mathbf{x}_n(1-\omega) + \mathbf{D}^{-1}\left[(\mathbf{E}+\mathbf{F})\mathbf{x}_n + \mathbf{b}\right]\omega \\ &= \mathbf{x}_n(1-\omega) + \mathbf{x}_{n+1}^{\mathrm{Jac}}\omega\,.\end{aligned} \tag{1.34}$$

So the new iterate $\mathbf{x}_{n+1}$ is the weighted mean of the old iterate $\mathbf{x}_n$ and one step of Jacobi, (1.31), starting from $\mathbf{x}_n$.

The iteration matrix is now

$$\boxed{\widehat{\mathbf{B}} :\equiv (1-\omega)\mathbf{I} + \omega\mathbf{D}^{-1}(\mathbf{E}+\mathbf{F}) = \mathbf{I} - \omega\mathbf{D}^{-1}\mathbf{A}\,.} \tag{1.35}$$

If the spectrum of $\mathbf{D}^{-1}\mathbf{A}$ lies, *e.g.*, on the interval $[\alpha, \beta]$ of the positive real axis, the one of $\widehat{\mathbf{B}}$ lies on $[1-\omega\beta, 1-\omega\alpha]$. Therefore, if $\omega > 0$ is chosen such that $\omega\beta < 2$, we will have convergence (according to Theorem 1.3.1). Moreover, ω could be chosen easily such that $\max\{|1-\omega\beta|, |1-\omega\alpha|\}$ is minimal, and thus ω is optimal.

It was Young (1950) who realized in his dissertation that the idea of relaxation is particularly effective in connection with the Gauss-Seidel method: we take *componentwise* a weighted mean of the old iterate $x_n^{(j)}$ and a step of Gauss-Seidel for that component, as given by (1.27):

$$\begin{aligned} x_j^{(n+1)} &:= x_j^{(n)}(1-\omega) + x_j^{(n+1,\mathrm{GS})}\omega \qquad (1.36)\\ &= x_j^{(n)}(1-\omega) + \frac{\omega}{a_{jj}}\left(b_j - \sum_{k=1}^{j-1} a_{jk}\, x_k^{(n+1)} - \sum_{k=j+1}^{N} a_{jk}\, x_k^{(n)}\right), \\ &\qquad j = 1, \dots, N\,. \end{aligned}$$

This translates into

$$\boxed{(\mathbf{D} - \omega\mathbf{E})\mathbf{x}_{n+1} := [(1-\omega)\mathbf{D} + \omega\mathbf{F}]\mathbf{x}_n + \omega\mathbf{b}} \tag{1.37}$$

or the fixed point iteration with

$$\boxed{\widehat{\mathbf{B}} :\equiv \left(\tfrac{1}{\omega}\mathbf{D} - \mathbf{E}\right)^{-1} \left[\left(\tfrac{1}{\omega} - 1\right)\mathbf{D} + \mathbf{F}\right] , \qquad \widehat{\mathbf{b}} :\equiv \left(\tfrac{1}{\omega}\mathbf{D} - \mathbf{E}\right)^{-1} \mathbf{b}} \tag{1.38}$$

and is called **successive overrelaxation (SOR) method** [*Verfahren der sukzessiven Überrelaxation*]. It can be shown that for fixed ω in the interval $0 < \omega < 2$ the method converges for any system with Hpd matrix $\mathbf{A}$. (For a proof, see, e.g., p. 56 of Schwarz, Rutishauser and Stiefel (1968).)

There remains the question on how to choose ω so that convergence is fastest. Young (1950) derived a beautiful theory leading to the optimal ω for matrices that have the so-called **Property A** and are **consistently ordered** [*konsistent geordnet*]. The class of matrices with Property A includes many examples obtained by discretizing partial differential equations with the finite difference method. The property is (by definition) invariant under simultaneous column and row permutations of the matrix, but for optimal convergence we need rows and columns so-called consistently ordered.

For example, the matrix $\mathbf{A}$ of (1.7) has Property A, but only $\widetilde{\mathbf{A}}$ of (1.9) is also consistently ordered. Although we do not exactly define Property A and consistent ordering here, we give a statement of Young's convergence result. For partial proofs under slightly varying assumptions see, *e.g.*, pages 59–65 and 208–211 of Schwarz et al. (1968) (for $\mathbf{A}$ spd), pages 112–116 of Saad (1996) (discussion of consistent ordering, but no proof of optimal ω), and pages 149–154 of Greenbaum (1997) (no discussion of consistent ordering).

Theorem 1.4.1 *For a consistently ordered matrix $\mathbf{A}$ with Property A and real eigenvalues of $\mathbf{D}^{-1}\mathbf{A}$, the optimal relaxation factor ω_{opt} of the SOR method is*

$$\boxed{\omega_{\text{opt}} :\equiv \frac{2}{1 + \sqrt{1 - \lambda_{\max}^2}} ,} \tag{1.39}$$

where, in terms of notation (1.28), *$\lambda_{\max}$ is the largest eigenvalue of the matrix $\mathbf{D}^{-1}(\mathbf{E} + \mathbf{F})$, which is the iteration matrix of the Jacobi method. The spectral radius of the optimal SOR iteration matrix $\widehat{\mathbf{B}}_{\text{opt}}$ is then*

$$\boxed{\rho(\widehat{\mathbf{B}}_{\text{opt}}) = \omega_{\text{opt}} - 1 ,} \tag{1.40}$$

An interesting aspect is that, under the assumptions of the theorem, the real eigenvalues of $\mathbf{D}^{-1}\mathbf{A}$ are mapped by SOR with optimal ω onto the circle with radius $\rho(\widehat{\mathbf{B}}_{\text{opt}})$. So all the eigenvalues of $\widehat{\mathbf{B}}_{\text{opt}}$, although in general complex, have the same absolute value.

A further related method is the **symmetric SOR (SSOR) method**, where each step is a double step consisting of an SOR step followed by a backward SOR step; in short form:

$$\boxed{\begin{aligned}(\mathbf{D}-\omega\mathbf{E})\mathbf{x}_{n+\frac{1}{2}} &:= [(1-\omega)\mathbf{D}+\omega\mathbf{F}]\mathbf{x}_n+\omega\mathbf{b}\,,\\ (\mathbf{D}-\omega\mathbf{F})\mathbf{x}_{n+1} &:= [(1-\omega)\mathbf{D}+\omega\mathbf{E}]\mathbf{x}_{n+\frac{1}{2}}+\omega\mathbf{b}\,.\end{aligned}} \tag{1.41}$$

The convergence analysis is even more complicated than for SOR.

There are also block versions of all these schemes. For example, one may consider the equations and the unknowns that belong to points on a line of the discretized Poisson equation (1.6) as a group defining a diagonal block of **A**. Then, in (1.28), **D** will be a block diagonal matrix and **E**, **F** the corresponding block triangular parts. With this new meaning for **D**, **E**, and **F**, the various methods discussed in this section can be redefined.

Except for SSOR all the methods discussed in this section can be viewed as applications of the general principle of improving fixed point iteration by **matrix splitting**: we write

$$\boxed{\mathbf{A}=\mathbf{M}-\mathbf{N}\,,} \tag{1.42}$$

where **M** is chosen so that, for any **y**, the system $\mathbf{M}\mathbf{x}=\mathbf{y}$ is easy to solve for **x**. Then an iterative method can be defined by

$$\boxed{\mathbf{M}\mathbf{x}_{n+1}:=\mathbf{N}\mathbf{x}_n+\mathbf{b}=\mathbf{M}\mathbf{x}_n+\mathbf{r}_n\,.} \tag{1.43}$$

In every step a linear system with the matrix **M** has to be solved. Formally, the method is equivalent with

$$\boxed{\mathbf{x}_{n+1}:=\mathbf{M}^{-1}\mathbf{N}\mathbf{x}_n+\mathbf{M}^{-1}\mathbf{b}=\mathbf{x}_n+\mathbf{M}^{-1}\mathbf{r}_n\,.} \tag{1.44}$$

This is just a linear fixed point iteration with

$$\boxed{\widehat{\mathbf{B}}:\equiv\mathbf{M}^{-1}\mathbf{N}\,,\qquad \widehat{\mathbf{b}}:\equiv\mathbf{M}^{-1}\mathbf{b}\,.} \tag{1.45}$$

Therefore, the iteration converges if and only if $\rho(\mathbf{M}^{-1}\mathbf{N})<1$. So, the aim must be to find splittings (1.42) where this spectral radius is small. Unfortunately, in general it is difficult to conclude from the spectrum of **A** on that of $\widehat{\mathbf{B}}=\mathbf{M}^{-1}\mathbf{N}$. But there is the important class of so-called **M-matrices** [*M-Matrizen*] and corresponding so-called **regular splittings** [*reguläre Splittings*] where at least convergence can be proved.

Table 1.1 summarizes the meanings of **M** and **N** in our above treated classical methods. These methods, in particular SOR, were very popular in the 1950ies and 1960ies, but they are hardly used nowadays for the original purpose. We will come back to them in Section 1.8 on basic preconditioning techniques, however.

Table 1.1 Some matrix splittings based on $\mathbf{A} = \mathbf{D} - \mathbf{E} - \mathbf{F}$.

method	$\mathbf{M}$	$\mathbf{N}$
Jacobi	$\mathbf{D}$	$\mathbf{E} + \mathbf{F}$
Gauss-Seidel	$\mathbf{D} - \mathbf{E}$	$\mathbf{F}$
damped Jacobi	$\frac{1}{\omega}\mathbf{D}$	$\left(\frac{1}{\omega} - 1\right)\mathbf{D} + \mathbf{E} + \mathbf{F}$
SOR	$\frac{1}{\omega}\mathbf{D} - \mathbf{E}$	$\left(\frac{1}{\omega} - 1\right)\mathbf{D} + \mathbf{F}$
backward SOR	$\frac{1}{\omega}\mathbf{D} - \mathbf{F}$	$\left(\frac{1}{\omega} - 1\right)\mathbf{D} + \mathbf{E}$

1.5 Krylov Subspaces and Krylov Space Solvers

The discussion of the Jacobi method in Section 1.3 suggests to look for better approximate solutions lying in the following affine space:

$$\mathbf{x}_n \in \mathbf{x}_0 + \mathsf{span}\left\{\mathbf{r}_0, \mathbf{A}\mathbf{r}_0, \dots, \mathbf{A}^{n-1}\mathbf{r}_0\right\},$$

see (1.24c). We first investigate the subspace that appears in this formula.

Definition. Given a nonsingular $N \times N$ matrix $\mathbf{A}$ and an N-vector $\mathbf{y} \neq \mathbf{o}$, the nth **Krylov (sub)space** $\mathcal{K}_n(\mathbf{A}, \mathbf{y})$ [*Krylov–Raum*] generated by $\mathbf{A}$ from $\mathbf{y}$ is

$$\boxed{\mathcal{K}_n :\equiv \mathcal{K}_n(\mathbf{A}, \mathbf{y}) :\equiv \mathsf{span}\,(\mathbf{y}, \mathbf{A}\mathbf{y}, \dots, \mathbf{A}^{n-1}\mathbf{y}).} \qquad (1.46)$$

▲

Clearly, by this definition, whenever $\mathbf{z} \in \mathcal{K}_n(\mathbf{A}, \mathbf{y})$, there is a polynomial p of degree at most $n - 1$ such that $\mathbf{z} = p(\mathbf{A})\mathbf{y}$. In general, this polynomial may be not unique since the spanning set in (1.46) may be linearly dependent. We can say more about this in a moment.

Definition (1.46) associates with a matrix $\mathbf{A}$ and a starting vector $\mathbf{y}$ a whole nested sequence of Krylov subspaces:

$$\mathcal{K}_1 \subseteq \mathcal{K}_2 \subseteq \mathcal{K}_3 \subseteq \dots .$$

The following lemma answers the question of the equality signs.

Lemma 1.5.1 *There is a positive integer $\bar{\nu} :\equiv \bar{\nu}(\mathbf{y}, \mathbf{A})$ such that*

$$\boxed{\mathsf{dim}\;\mathcal{K}_n(\mathbf{A}, \mathbf{y}) = \begin{cases} n & \textit{if} \quad n \leq \bar{\nu}\,, \\ \bar{\nu} & \textit{if} \quad n \geq \bar{\nu}\,. \end{cases}}$$

The inequalities $1 \leq \bar{\nu} \leq N$ hold, and $\bar{\nu} < N$ is possible if $N > 1$.

Definition. The positive integer $\bar{\nu} :\equiv \bar{\nu}(\mathbf{y}, \mathbf{A})$ of Lemma 1.5.1 is called **grade of y with respect to A** [*Grad von* $\mathbf{y}$ *bezüglich* $\mathbf{A}$]. ▲

Proof of Lemma 1.5.1. By definition (1.46),

$$\mathcal{K}_n :\equiv \mathcal{K}_n(\mathbf{A}, \mathbf{y}) :\equiv \mathsf{span}\,(\mathbf{y}, \mathbf{A}\mathbf{y}, \dots, \mathbf{A}^{n-1}\mathbf{y})\,,$$

with some $\mathbf{y} \neq \mathbf{o}$. So, $\dim \mathcal{K}_1 = 1$, and if $\mathbf{y}, \mathbf{A}\mathbf{y}, \dots, \mathbf{A}^{n-1}\mathbf{y}$ are linearly independent, then $\dim \mathcal{K}_n = n$. Let $n = \bar{\nu}$ be the first n for which $\mathbf{y}, \mathbf{A}\mathbf{y}, \dots, \mathbf{A}^{n-1}\mathbf{y}, \mathbf{A}^n\mathbf{y}$ are linearly dependent. Then $\mathbf{A}^n\mathbf{y} = \mathbf{A}^{\bar{\nu}}\mathbf{y}$ is a linear combination of the linearly independent vectors $\mathbf{y}, \mathbf{A}\mathbf{y}, \dots, \mathbf{A}^{\bar{\nu}-1}\mathbf{y}$:

$$\mathbf{A}^{\bar{\nu}}\mathbf{y} = \mathbf{y}\gamma_0 + \mathbf{A}\mathbf{y}\gamma_1 + \cdots + \mathbf{A}^{\bar{\nu}-1}\mathbf{y}\gamma_{\bar{\nu}-1} \,. \tag{1.47}$$

Here, $\gamma_0 \neq 0$, because otherwise, after multiplication by $\mathbf{A}^{-1}$, we would have

$$\mathbf{A}^{\bar{\nu}-1}\mathbf{y} = \mathbf{y}\gamma_1 + \mathbf{A}\mathbf{y}\gamma_2 + \cdots + \mathbf{A}^{\bar{\nu}-2}\mathbf{y}\gamma_{\bar{\nu}-1} \,,$$

in contrast to the assumption that the vectors on the right-hand side are linearly independent.

If we consider the definition (1.46) for some $n > \bar{\nu}$, then, by using (1.47), all terms $\mathbf{A}^k\mathbf{y}$ with $k \geq \bar{\nu}$ in the span can be recursively replaced by sums of terms with $k < \bar{\nu}$. So, the dimension cannot be larger than $\bar{\nu}$; and of course, it cannot be smaller since $\mathcal{K}_n \supseteq \mathcal{K}_{\bar{\nu}}$ if $n \geq \bar{\nu}$.

If we choose $\mathbf{y}$ as an eigenvector of $\mathbf{A}$, then $\bar{\nu} = 1$, so $\bar{\nu}$ can be smaller than N if $N > 1$. $\square$

We can say more and understand the structure of the Krylov subspaces better if we consider the Jordan canonical form of $\mathbf{A}$ (as in the proof if Theorem 1.3.1) and recall the notion of the minimal polynomial of $\mathbf{A}$. So we let $\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^{-1}$ be the Jordan decomposition of $\mathbf{A}$, *i.e.,* $\mathbf{\Lambda}$ is a block-diagonal matrix of the form

$$\mathbf{\Lambda} = \begin{pmatrix} \mathbf{J}_1 & & \\ & \ddots & \\ & & \mathbf{J}_\mu \end{pmatrix}, \quad \text{where } \mathbf{J}_k = \begin{pmatrix} \lambda_k & 1 & & \\ & \lambda_k & \ddots & \\ & & \ddots & 1 \\ & & & \lambda_k \end{pmatrix} \in \mathbb{C}^{m_k \times m_k} \tag{1.48}$$

are either bidiagonal Jordan blocks or 1×1 blocks (λ_k). Some of the eigenvalues λ_k in the different blocks may coincide, and we choose to rearrange the blocks so that a full set of Jordan blocks of maximum size m_k corresponding to all the $\bar{\mu}$ distinct eigenvalues $\lambda_1, \dots, \lambda_{\bar{\mu}}$ is at the top. So, if $\bar{\mu} < \mu$, then $\lambda_{\bar{\mu}+1}, \dots, \lambda_\mu$ all coincide with some eigenvalue among the first $\bar{\mu}$, and the block sizes m_k for $k > \bar{\mu}$ are at most as large as the corresponding ones for the same eigenvalue with $k \leq \bar{\mu}$. Then the **minimal polynomial** [*Minimalpolynom*] $\widehat{\chi}_{\mathbf{A}}$ of $\mathbf{A}$ is defined by

$$\boxed{\widehat{\chi}_{\mathbf{A}}(t) :\equiv \prod_{k=1}^{\bar{\mu}} (t - \lambda_k)^{m_k} \,.} \tag{1.49}$$

It is a divisor of the characteristic polynomial

$$\boxed{\chi_{\mathbf{A}}(t) :\equiv \prod_{k=1}^{\mu} (t - \lambda_k)^{m_k}} \tag{1.50}$$

(where the product runs now over all μ diagonal blocks $\mathbf{J}_k$ of the Jordan decomposition (1.48)). Its degree $M :\equiv \partial\widehat{\chi}_{\mathbf{A}}$ satisfies

$$M :\equiv \partial\widehat{\chi}_{\mathbf{A}} = \sum_{k=1}^{\bar{\mu}} m_k \le \sum_{k=1}^{\mu} m_k = \partial\psi = N\,.$$

We claim that if we insert $t := \mathbf{A}$ into the minimal polynomial (so that it becomes a matrix polynomial), we get the zero matrix.

Theorem 1.5.2 *If $\widehat{\chi}_{\mathbf{A}}$ denotes the minimal polynomial of $\mathbf{A}$, then*

$$\widehat{\chi}_{\mathbf{A}}(\mathbf{A}) = \prod_{k=1}^{\bar{\mu}} (\mathbf{A} - \lambda_k \mathbf{I})^{m_k} = \mathbf{O}\,. \tag{1.51}$$

Proof. We have

$$\widehat{\chi}_{\mathbf{A}}(\mathbf{A}) = \mathbf{V}\,\widehat{\chi}_{\mathbf{A}}(\mathbf{\Lambda})\,\mathbf{V}^{-1} = \mathbf{V}\left(\prod_{k=1}^{\bar{\mu}} (\mathbf{\Lambda} - \lambda_k \mathbf{I})^{m_k}\right)\mathbf{V}^{-1},$$

and here all the factors of the product, and hence the product itself, is block diagonal. Consider, the block $(\mathbf{J}_k - \lambda_k \mathbf{I})^{m_k}$ of the kth factor. In the notation of (1.14) it equals $\mathbf{S}^{m_k}$, where, in this factor, $\mathbf{S}$ is the $m_k \times m_k$ matrix with ones on the upper bidiagonal and zeros elsewhere. So, $\mathbf{S}^{m_k} = \mathbf{O}$. Hence, in the product in question, there is for each k a factor where the kth block is zero. Consequently, the whole product is a zero matrix, and thus also $\widehat{\chi}_{\mathbf{A}}(\mathbf{\Lambda})$ and $\widehat{\chi}_{\mathbf{A}}(\mathbf{A})$. $\square$

Since the minimal polynomial is a divisor of the characteristic polynomial, the following famous **Caley-Hamilton theorem** follows immediately.

Corollary 1.5.3 *If $\chi_{\mathbf{A}}$ denotes the characteristic polynomial of $\mathbf{A}$, then $\chi_{\mathbf{A}}(\mathbf{A}) = \mathbf{O}$.*

Now we are ready to prove the following result on the grade $\bar{\nu}$ that appeared in Lemma 1.5.1.

Lemma 1.5.4 *The nonnegative integer $\bar{\nu}$ of Lemma 1.5.1 satisfies*

$$\boxed{\bar{\nu} = \min\left\{n \;\middle|\; \mathbf{A}^{-1}\mathbf{y} \in \mathcal{K}_n(\mathbf{A},\mathbf{y})\right\} \le \partial\widehat{\chi}_{\mathbf{A}},}$$

where $\partial\widehat{\chi}_{\mathbf{A}}$ denotes the degree of the minimal polynomial of $\mathbf{A}$.

Proof. Multiplying (1.47) by $\mathbf{A}^{-1}$ we see that

$$\mathbf{A}^{-1}\mathbf{y} = \left(\mathbf{A}^{\bar{\nu}-1}\mathbf{y} - \mathbf{A}^{\bar{\nu}-2}\mathbf{y}\gamma_{\bar{\nu}-1} - \cdots - \mathbf{A}\mathbf{y}\gamma_2 - \mathbf{y}\gamma_1\right)\frac{1}{\gamma_0}\,, \tag{1.52}$$

so $\mathbf{A}^{-1}\mathbf{y} \in \mathcal{K}_{\bar{\nu}}(\mathbf{A},\mathbf{y})$. We cannot replace $\bar{\nu}$ by some $n < \bar{\nu}$ here, because this would lead to a contradiction to the minimality of $\bar{\nu}$ in (1.47).

It remains to show that $\bar{\nu} \le \partial\widehat{\chi}_{\mathbf{A}}$. The product formula for $\widehat{\chi}_{\mathbf{A}}(\mathbf{A}) = \mathbf{O}$ in (1.51) could be written as a linear combination of $\mathbf{I}$, $\mathbf{A}$, …, $\mathbf{A}^M$ that is zero, but the coefficient of $\mathbf{A}^M$ is 1 (here, again, $M :\equiv \partial\widehat{\chi}_{\mathbf{A}}$). This remains valid if we post-multiply each term by any $\mathbf{y}$. So, for any $\mathbf{y}$, the Krylov subspace $\mathcal{K}_{M+1}(\mathbf{A}, \mathbf{y})$ has dimension at most M, and in view of Lemma 1.5.1 the same is true for $\mathcal{K}_n(\mathbf{A}, \mathbf{y})$ for any n. □

Of course, the actual dimension may be smaller for some $\mathbf{y}$. In fact, it is, if in the representation of $\mathbf{y}$ in terms of the basis associated with the Jordan decomposition of $\mathbf{A}$ some of the relevant coordinates vanish.

We can conclude that as long as $n \leq \bar{\nu}(\mathbf{y}, \mathbf{A})$, the vectors $\mathbf{y}$, $\mathbf{Ay}$, ..., $\mathbf{A}^{n-1}\mathbf{y}$ in (1.46) are linearly independent, and thus the polynomial p representing some $\mathbf{z} = p(\mathbf{A})\mathbf{y} \in \mathcal{K}_n(\mathbf{A}, \mathbf{y})$ is uniquely determined. In other words, as long as $n \leq \bar{\nu}$, there is a natural one-to-one correspondence (actually, an isomorphism) between the linear space $\mathcal{K}_n$ and the linear space $\mathcal{P}_{n-1}$ of polynomials of degree at most $n-1$.

Unfortunately, even for $n \leq \bar{\nu}$ the vectors $\mathbf{y}$, $\mathbf{Ay}$, ..., $\mathbf{A}^{n-1}\mathbf{y}$ form typically a very ill-conditioned basis for $\mathcal{K}_n$, since they tend to be nearly linearly dependent. In fact, one can easily show that if $\mathbf{A}$ has a unique eigenvalue of largest absolute value and of algebraic multiplicity one, and if $\mathbf{y}$ is not orthogonal to a corresponding normalized eigenvector $\mathbf{v}$, then $\mathbf{A}^k\mathbf{y}/\|\mathbf{A}^k\mathbf{y}\| \to \pm\mathbf{v}$ as $k \to \infty$. Therefore, in practice, we will never make use of this so-called **Krylov basis** [*Krylovbasis*].

In connection with the iterative solution of a linear system Lemma 1.5.4 yields a most welcome corollary.

COROLLARY 1.5.5 *Let $\mathbf{x}_\star$ be the solution of $\mathbf{Ax} = \mathbf{b}$ and let $\mathbf{x}_0$ be any initial approximation of it and $\mathbf{r}_0 :\equiv \mathbf{b} - \mathbf{Ax}_0$ the corresponding residual. Moreover, let $\bar{\nu} :\equiv \bar{\nu}(\mathbf{r}_0, \mathbf{A})$. Then*

$$\boxed{\mathbf{x}_\star \in \mathbf{x}_0 + \mathcal{K}_{\bar{\nu}}(\mathbf{A}, \mathbf{r}_0)\,.}$$

PROOF. By (1.18), (1.20), and by Lemma 1.5.4,

$$\mathbf{x}_\star - \mathbf{x}_0 = \mathbf{d}_0 = -\mathbf{A}^{-1}\mathbf{r}_0 \in \mathcal{K}_{\bar{\nu}}(\mathbf{A}, \mathbf{r}_0)\,.$$

□

This corollary shows that if we choose $\mathbf{x}_n$ from the affine space $\mathbf{x}_0 + \mathcal{K}_n(\mathbf{A}, \mathbf{r}_0)$ there is a chance that we find the exact solution within at most $\bar{\nu}$ steps. We say then that our method has the **finite termination property**. We will see that it is easy to deduce methods that have this property. In fact, it suffices to insure that the residuals $\mathbf{r}_n$ are linearly independent as long as they are nonzero. Of course, once $\mathbf{r}_n = \mathbf{o}$ for some n, the linear system is solved.

However, in practice the finite termination property is normally not important, since $\bar{\nu}$ is typically much larger than the maximum number of iterations we are willing to execute. We rather want an approximation with sufficiently small residual quickly.

The corollary and the analogy to the relation (1.24c) now motivate the following definition.

Definition. A **(standard) Krylov space method for solving a linear system** [*(Standard-)Krylov-Raum-Methode*] $\mathbf{Ax} = \mathbf{b}$ or, briefly, a **(standard) Krylov space solver** [3], is an iterative method starting from some initial approximation $\mathbf{x}_0$ and the corresponding residual $\mathbf{r}_0 :\equiv \mathbf{b} - \mathbf{Ax}_0$ and generating for all, or at least most n, iterates $\mathbf{x}_n$ such that

$$\boxed{\mathbf{x}_n - \mathbf{x}_0 = q_{n-1}(\mathbf{A})\mathbf{r}_0 \in \mathcal{K}_n(\mathbf{A}, \mathbf{r}_0)} \tag{1.53}$$

with a polynomial q_{n-1} of exact degree $n-1$. ▲

We first note that (1.53) implies that

$$\mathbf{d}_n - \mathbf{d}_0 = q_{n-1}(\mathbf{A})\mathbf{r}_0 \in \mathcal{K}_n(\mathbf{A}, \mathbf{r}_0)\,, \tag{1.54}$$
$$\mathbf{r}_n - \mathbf{r}_0 = -\mathbf{A}q_{n-1}(\mathbf{A})\mathbf{r}_0 \in \mathbf{A}\mathcal{K}_n(\mathbf{A}, \mathbf{r}_0)\,. \tag{1.55}$$

From the second equation we find a result that shows that these methods generalize the Jacobi iteration; *cf.* (1.23).

Lemma 1.5.6 *The residuals of a Krylov space solver satisfy*

$$\boxed{\mathbf{r}_n = p_n(\mathbf{A})\mathbf{r}_0 \in \mathbf{r}_0 + \mathbf{A}\mathcal{K}_n(\mathbf{A}, \mathbf{r}_0) \subseteq \mathcal{K}_{n+1}(\mathbf{A}, \mathbf{r}_0)\,,} \tag{1.56}$$

where p_n is a polynomial of degree n, which is related to the polynomial q_{n-1} of (1.53) by

$$\boxed{p_n(\zeta) = 1 - \zeta q_{n-1}(\zeta)\,.} \tag{1.57}$$

In particular,

$$\boxed{p_n(0) = 1\,.} \tag{1.58}$$

Definition. The polynomials $p_n \in \mathcal{P}_n$ in (1.56) are the **residual polynomials** [*Residualpolynome*] of the Krylov space solver. Eq. (1.58) is their **consistency condition** [*Konsistenz-Bedingung*]. ▲

In certain Krylov space solvers there may exist exceptional situations where for some n the iterate $\mathbf{x}_n$ and the residual $\mathbf{r}_n$ are not defined. There are also **nonstandard Krylov space methods** [*Nicht-Standard-Krylov-Raum-Methode*] where the approximation space for $\mathbf{x}_n - \mathbf{x}_0$ is still a Krylov space, but one that differs from $\mathcal{K}_n(\mathbf{A}, \mathbf{r}_0)$.

Krylov space methods are a very important class of numerical methods. With respect to the "influence on the development and practice of science and engineering in the 20th century", they are considered as one of the ten most important classes (Dongarra and Sullivan, 2000; van der Vorst, 2000).

[3]Many authors use the term **Krylov subspace method** instead, but, of course, any subspace of a linear space is itself a linear space. We certainly want to avoid the German "Krylov-Unterraum-Methode".

Krylov space methods have been the most important topic of sparse matrix analysis through the whole second part of the last century, although there exist other approaches for solving sparse linear systems that do not fit into this class. Moreover, the Krylov space approach is also applicable to eigenvalue problems.

As we will see there are various ways to derive suitable Krylov space solvers. Depending on the way they were used and the time period, various names have been given to the class of methods called Krylov space solvers here: **gradient methods** (Rutishauser, 1959), **semi-iterative methods** (Varga, 1962; Young, 1971), **polynomial acceleration methods**, **polynomial preconditioners**, **Krylov subspace iterations** (van der Vorst, 2000).

In view of $\mathbf{r}_n = -\,\mathbf{A}\mathbf{d}_n$ (see (1.20) and (1.54)–(1.55)) Lemma 1.5.6 implies an analogous result on the error vectors.

LEMMA 1.5.7 *The error vectors of a Krylov space solver satisfy*

$$\mathbf{d}_n = p_n(\mathbf{A})\mathbf{d}_0 \in \mathbf{d}_0 + \mathbf{A}\mathcal{K}_n(\mathbf{A}, \mathbf{d}_0) \subseteq \mathcal{K}_{n+1}(\mathbf{A}, \mathbf{d}_0)\,, \qquad (1.59)$$

where p_n is the nth residual polynomial.

Note, however, that the Krylov space $\mathcal{K}_n(\mathbf{A}, \mathbf{d}_0)$ that appears here, is different from the one we normally consider, $\mathcal{K}_n(\mathbf{A}, \mathbf{r}_0)$.

1.6 Chebyshev Iteration

As we have mentioned beforehand, the simplest way — though not always the best way — to check the convergence of a Krylov space solver is to evaluate a norm of the residual vector. A natural approach to designing a Krylov space solver is therefore to try to minimize a residual norm. The representation (1.56), $\mathbf{r}_n = p_n(\mathbf{A})\mathbf{r}_0$, of the residual and the spectral decomposition $\mathbf{AU} = \mathbf{U\Lambda}$ of the matrix $\mathbf{A}$ yield $\mathbf{r}_n = \mathbf{U}p_n(\mathbf{\Lambda})\mathbf{U}^{-1}\mathbf{r}_0$. Therefore, in the 2-norm of the vectors and the induced spectral norm for the matrices,

$$\boxed{\|\mathbf{r}_n\|/\|\mathbf{r}_0\| \le \kappa(\mathbf{U})\|p_n(\mathbf{\Lambda})\|} \qquad (1.60)$$

with $\kappa(\mathbf{U}) :\equiv \|\mathbf{U}\|\;\|\mathbf{U}^{-1}\|$ the (spectral) condition number of $\mathbf{U}$. If $\mathbf{\Lambda}$ is diagonal, $\mathbf{\Lambda} \equiv: \mathsf{diag}\,\{\lambda_1, \dots, \lambda_N\}$,

$$\|p_n(\mathbf{\Lambda})\| = \max_{i=1,\dots,N} |p_n(\lambda_i)|\,, \qquad (1.61)$$

so the problem of finding a good Krylov space solver for a particular $\mathbf{A}$ can be reduced to the approximation problem

$$\boxed{\max_{i=1,\dots,N} |p_n(\lambda_i)| = \min! \quad \text{subject to} \quad p_n \in \mathcal{P}_n, \quad p_n(0) = 1\,.} \qquad (1.62)$$

There are some problems with this approach, however. First, in general we cannot assume that the eigenvalues of $\mathbf{A}$ are known; their computation is normally much more costly than solving a linear system with the same matrix. Second, the condition number $\kappa(\mathbf{U})$ may be very large, and the inequality in (1.60) may be far from sharp. Third, the approximation problem (1.62) is very difficult to solve if there are complex eigenvalues.

But let us assume here that $\mathbf{A}$ is real symmetric or Hermitian and positive definite (*i.e.,* spd or Hpd), so that $\mathbf{\Lambda}$ is diagonal, the eigenvalues are positive, and $\mathbf{U}$ is unitary and thus $\kappa(\mathbf{U}) = 1$. There is still the difficulty that the individual eigenvalues are not known. Any norm of $\mathbf{A}$ yields an upper bound for the eigenvalues, and often a positive lower bound can be found somehow. So, assume

$$\lambda_i \in \mathcal{I} :\equiv [\alpha - \delta, \alpha + \delta] \qquad (i = 1, \dots, N) \qquad (1.63)$$

with $\alpha > 0$ and $0 < \delta < \alpha$. Then

$$\boxed{\|\mathbf{r}_n\|/\|\mathbf{r}_0\| \le \|p_n(\mathbf{\Lambda})\| = \max_{i=1,\dots,N} |p_n(\lambda_i)| \le \max_{\tau\in\mathcal{I}} |p_n(\tau)|\,.} \qquad (1.64)$$

This suggests to replace the approximation problem (1.62) by

$$\boxed{\max_{\tau\in\mathcal{I}} |p_n(\tau)| = \min! \quad \text{subject to} \quad p_n \in \mathcal{P}_n, \quad p_n(0) = 1\,.} \qquad (1.65)$$

We claim that this real polynomial approximation problem can be solved analytically and that the optimal polynomial is just a shifted and scaled **Chebyshev polynomial** [*Tschebyscheff–Polynom*] of degree n, which on $\mathbb{R}$ is defined by

$$T_n(\xi) :\equiv \begin{cases} \cos(n \arccos(\xi)) & \text{if} \quad |\xi| \le 1 \\ \left((\text{sign}\,(\xi)\right)^n \cosh(n \,\text{arcosh}(|\xi|)) & \text{if} \quad |\xi| \ge 1 \end{cases}$$

and satisfies the three-term recursion

$$\boxed{T_{n+1}(\xi) := 2\xi T_n(\xi) - T_{n-1}(\xi) \qquad (n > 1)} \tag{1.66}$$

with initial values $T_0(\xi) := 1$, $T_1(\xi) := \xi$. The recursion is just a translation of the identity $2cos\phi \cos n\phi = \cos(n+1)\phi + \cos(n-1)\phi$ under the substitution $\xi = \cos\phi$. Clearly, $T_n(\xi) = \cos(n \arccos(\xi))$ oscillates on the interval $[-1, 1]$ between its minima and maxima ± 1, of which there are a total of $n+1$, including two at the endpoints ± 1; see Figure 1.1 for two clippings of the graph of T_{11}.

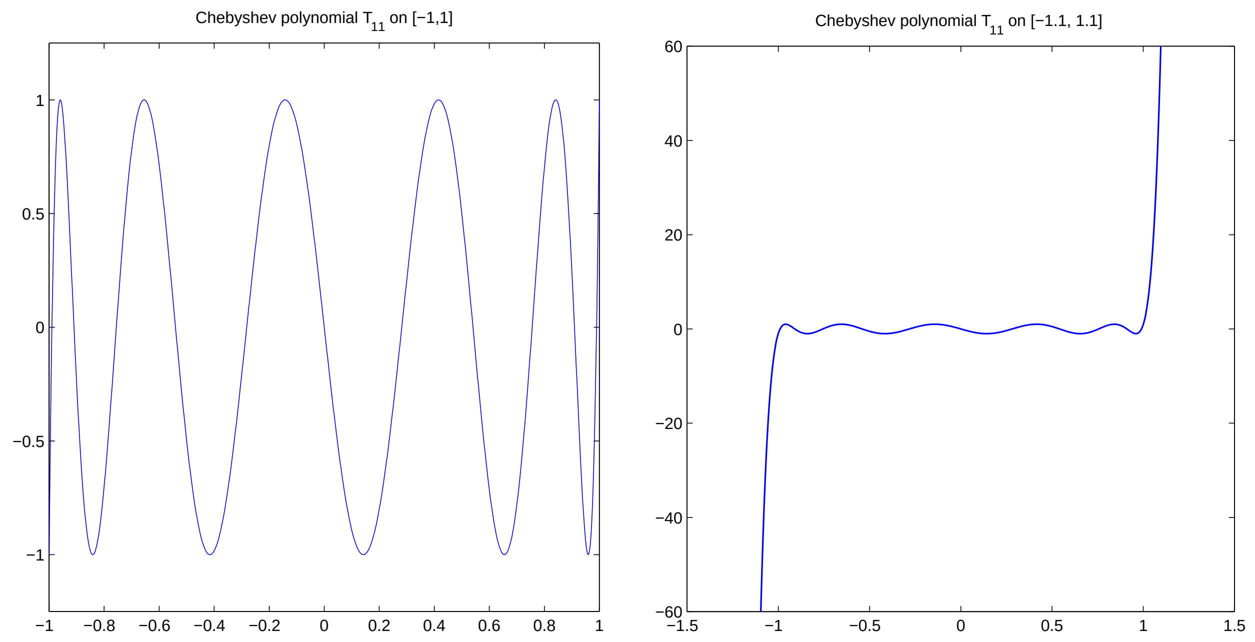


Figure 1.1: The Chebyshev polynomial T_{11} equioscillating on the interval $[-1, 1]$ (at left), and its steep increase of the absolute value outside the interval $[-1, 1]$ (at right).

We capitalize upon this behavior by using the additional substitution $\tau \mapsto \xi := (\tau - \alpha)/\delta$, which maps the given interval $\mathcal{I}$ onto $[-1, 1]$ and by scaling the function so that $p(0) = 1$; see Figure 1.2.

THEOREM 1.6.1 *The optimal solution of the approximation problem* (1.65) *with* $\mathcal{I} :\equiv [\alpha - \delta, \alpha + \delta]$ *is the* ***shifted and scaled Chebyshev polynomial***

$$\boxed{p_n(\tau) = \frac{T_n\left(\frac{\tau-\alpha}{\delta}\right)}{T_n\left(-\frac{\alpha}{\delta}\right)}}\,. \tag{1.67}$$

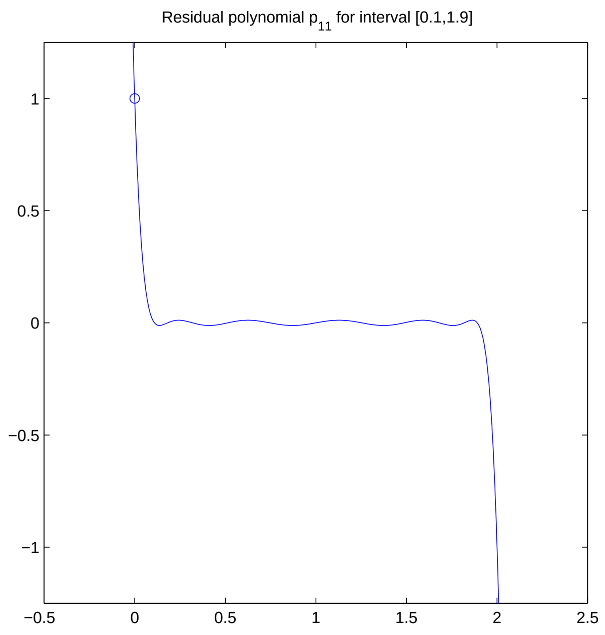


Figure 1.2: The residual polynomial p_{11} of the Chebyshev iteration for the interval $[0.1, 1.9]$ — a shifted and scaled version of T_{11}.

Proof. Problem (1.65) is a variation of a classical approximation problem where p_n is required to be monic (*i.e.*, to have leading coefficient 1) instead of having constant coefficient 1 and the interval is $[-1, 1]$. For that problem, the solution is T_n, and the proof is a special case of the one for the equioscillation theorem from the theory of real polynomial uniform (or Chebyshev) approximation.

The polynomial p_n has on $\mathcal{I}$ the maximum and minimum values $\pm 1/T_n\left(-\frac{\alpha}{\delta}\right)$. The maximum is taken at $\lceil \frac{1}{2}(n+1) \rceil$ points and the minimum at interlacing $\lceil \frac{1}{2}n \rceil$ points. Assume there is a polynomial $s \in \mathcal{P}_n$ with $s(0) = 1$, which yields a smaller maximum in (1.65). Then the difference $p_n - s \in \mathcal{P}_n$ will have alternating sign at the $n + 1$ maxima and minima of p_n, so it will have n zeros between these points. Moreover, it has another zero at $\tau = 0$ where $p_n(0) = s(0) = 1$. That makes a total of $n + 1$ zeros, which contradicts to the limit of n zeros for a nonzero polynomial of degree n. □

The recursions (1.66) for the Chebyshev polynomials lead to recursions for the residual polynomials p_n and, thus, for the residuals $\mathbf{r}_n$ and the corresponding iterates $\mathbf{x}_n$. The resulting Krylov space solver is called **Chebyshev iteration** [*Tschebyscheff-Iteration*]. In earlier times it was the **Chebyshev semi-iterative method**. The method was, *e.g.*, investigated by Rutishauser (1959), Golub and Varga (1961), and Varga (1962).

Chebyshev iteration is optimal for the set of matrices $\mathbf{A}$ whose spectrum is confined to the interval $\mathcal{I} :\equiv [\alpha - \delta, \alpha + \delta]$. Of course, for an individual such matrix there are still faster methods. The method can be seen to be asymptotically optimal (for $n \to \infty$) also for the set of matrices $\mathbf{A}$ whose spectrum is confined to an elliptical domain $\mathcal{E}$ with foci $\alpha-\delta$ and $\alpha+\delta$ that does not contain the origin. But, in general, it is not optimal in this case, see Fischer and Freund (1990) and Fischer and Freund (1991): exceptions exists even with $\alpha, \delta \in \mathbb{R}$. Moreover, in the case where $\mathbf{A}$ is not real symmetric or Hermitian, the factor $\kappa(\mathbf{U})$ in (1.60) will normally not be 1.

Although our derivation is limited to the case of eigenvalues in a real interval $\mathcal{I}$, we formulate the method here so that the case of complex eigenvalues is included.

Algorithm 1.1 (Chebyshev Iteration) .
For solving $\mathbf{A}\mathbf{x} = \mathbf{b}$ *choose* $\mathbf{x}_0 \in \mathbb{E}^N$ *and let* $\mathbf{r}_0 := \mathbf{b} - \mathbf{A}\mathbf{x}_0$. *Set* $\mathbf{r}_{-1} := \mathbf{x}_{-1} := \mathbf{o}$.
Choose the parameters α *and* δ *so that the spectrum of* $\mathbf{A}$ *lies on the interval* $\mathcal{I} :\equiv [\alpha-\delta, \alpha+\delta]$ *or on an elliptical domain* $\mathcal{E}$ *with foci* $\alpha \pm \delta$, *but so that* $0 \notin \mathcal{I}$ *or* $0 \notin \mathcal{E}$, *respectively. Then let* $\eta :\equiv -\alpha/\delta$,

$$\beta_{-1} := 0\,, \qquad \beta_0 := \frac{\delta}{2}\frac{1}{\eta} = -\frac{\delta^2}{2\alpha}\,, \qquad \gamma_0 := -\alpha\,, \tag{1.68a}$$

and compute, for $n = 0, 1, \ldots$ *until convergence,*

$$\beta_{n-1} :\equiv \frac{\delta}{2}\frac{T_{n-1}(\eta)}{T_n(\eta)} := \left(\frac{\delta}{2}\right)^2 \frac{1}{\gamma_{n-1}} \qquad \text{if} \quad n \ge 2, \tag{1.68b}$$

$$\gamma_n :\equiv \frac{\delta}{2}\frac{T_{n+1}(\eta)}{T_n(\eta)} := -(\alpha + \beta_{n-1}) \qquad \text{if} \quad n \ge 1, \tag{1.68c}$$

$$\mathbf{x}_{n+1} := -\big(\mathbf{r}_n + \mathbf{x}_n\alpha + \mathbf{x}_{n-1}\beta_{n-1}\big)/\gamma_n\,, \tag{1.68d}$$

$$\mathbf{r}_{n+1} := \big(\mathbf{A}\mathbf{r}_n - \mathbf{r}_n\alpha - \mathbf{r}_{n-1}\beta_{n-1}\big)/\gamma_n\,. \tag{1.68e}$$

Note that in the case of real foci, $\eta :\equiv -\alpha/\delta < -1$, so that in (1.68b) and (1.68c) the formula $T_n(\eta) = (-1)^n \cosh(n \operatorname{arcosh}(-\eta))$ should be used.

For the Chebyshev iteration we can specify a bound for the residual norm reduction and thus estimate the speed of convergence. This requires a little bit more classical analysis; additionally, complex analysis is also helpful for understanding the background of the method.

Let us again assume that $\mathbf{A}$ is Hpd and that its spectrum is known to be in $\mathcal{I} = [\alpha-\delta, \alpha+\delta]$, where $\alpha > \delta > 0$. Moreover, let us set

$$\kappa_{\mathcal{I}} :\equiv \frac{\alpha+\delta}{\alpha-\delta}\,. \tag{1.69}$$

Note that $\kappa_{\mathcal{I}}$ is a bound for the spectral condition number $\kappa(\mathbf{A})$, and that $\kappa_{\mathcal{I}} = \kappa(\mathbf{A})$ if $\mathbf{I}$ is chosen smallest possible, that is so that $\alpha \pm \delta$ are the smallest and the largest eigenvalues of $\mathbf{A}$. Recall that the two bounds in (1.64) hold. The one in the middle,

$$\|\mathbf{r}_n\|/\|\mathbf{r}_0\| \le \max_{i=1,\ldots,N} |p_n(\lambda_i)|\,, \tag{1.70}$$

is sharp in the sense that for any n we can specify an $\mathbf{x}_0$ so that equality holds. In fact, if m is the index for which the maximum is taken in (1.70), and if $\mathbf{u}_m$ is the eigenvector corresponding to λ_m, then we just need to choose $\mathbf{x}_0 := \mathbf{x}_\star - \mathbf{u}_m$. Then $\mathbf{r}_0 = \mathbf{A}(\mathbf{x}_\star - \mathbf{x}_0) = \mathbf{u}_m \lambda_m$ and $\mathbf{r}_n = \mathbf{u}_m \lambda_m p_n(\lambda_m) = \mathbf{r}_0 p_n(\lambda_m)$.

Since $(\tau - \alpha)/\delta \in [-1, 1]$ if $\tau \in \mathcal{I}$ we conclude further that

$$\max_{i=1,\dots,N} |p_n(\lambda_i)| = |p_n(\lambda_m)| \le \frac{1}{\left|T_n\left(-\frac{\alpha}{\delta}\right)\right|} = \frac{1}{|T_n(\eta)|}\,. \tag{1.71}$$

Here, since $\eta < -1$, we have $|T_n(\eta)|^{-1} < 1$, which means that $\|\mathbf{r}_n\| < \|\mathbf{r}_0\|$ if $n > 0$. The bound in (1.71) is sharp in the sense that there are matrices with spectrum contained in $\mathcal{I}$ for which equality holds.

As is easy to verify, $\vartheta :\equiv \exp\bigl(\operatorname{arcosh}(-\eta)\bigr) > 1$ satisfies

$$\frac{1}{2}\left(\vartheta + \frac{1}{\vartheta}\right) = -\eta\,. \tag{1.72}$$

In terms of ϑ the value $T_n(\eta)$ can be written as

$$T_n(\eta) = \frac{(-1)^n}{2}\left(\vartheta^n + \frac{1}{\vartheta^n}\right). \tag{1.73}$$

Relation (1.72), which, up to the minus sign, describes the **Joukowski transformation** (Henrici, 1974), means that $\vartheta^2 + 2\eta\vartheta + 1 = 0$ and yields

$$\vartheta = -\eta \pm \sqrt{\eta^2 - 1}. \tag{1.74}$$

In terms of $\kappa_\mathcal{I}$ we have from (1.69)

$$\eta = -\frac{\kappa_\mathcal{I} + 1}{\kappa_\mathcal{I} - 1}\,, \tag{1.75}$$

and, after some manipulation, we find the two reciprocal solutions

$$\vartheta = \frac{\sqrt{\kappa_\mathcal{I}} + 1}{\sqrt{\kappa_\mathcal{I}} - 1} \qquad \text{or} \qquad \vartheta = \frac{\sqrt{\kappa_\mathcal{I}} - 1}{\sqrt{\kappa_\mathcal{I}} + 1}\,, \tag{1.76}$$

which both yield

$$|T_n(\eta)| = \frac{1}{2}\left[\left(\frac{\sqrt{\kappa_\mathcal{I}} + 1}{\sqrt{\kappa_\mathcal{I}} - 1}\right)^n + \left(\frac{\sqrt{\kappa_\mathcal{I}} - 1}{\sqrt{\kappa_\mathcal{I}} + 1}\right)^n\right]. \tag{1.77}$$

Actually, only the first solution, the one with $\vartheta > 1$ is consistent with our definition of ϑ based on the principal branch of arcosh. In summary, (1.70), (1.71), and (1.77) yield the following estimate.

Theorem 1.6.2 *The residual norm reduction of the Chebyshev iteration, when applied to an Hpd system whose condition number is bounded by $\kappa_\mathcal{I}$, is bounded according to*

$$\frac{\|\mathbf{r}_n\|}{\|\mathbf{r}_0\|} \le 2\left[\left(\frac{\sqrt{\kappa_\mathcal{I}} + 1}{\sqrt{\kappa_\mathcal{I}} - 1}\right)^n + \left(\frac{\sqrt{\kappa_\mathcal{I}} - 1}{\sqrt{\kappa_\mathcal{I}} + 1}\right)^n\right]^{-1} \le 2\left(\frac{\sqrt{\kappa_\mathcal{I}} - 1}{\sqrt{\kappa_\mathcal{I}} + 1}\right)^n. \tag{1.78}$$

The first bound is sharp in the sense that there are matrices $\mathbf{A}$ *with spectrum in* $\mathcal{I}$ *and suitable initial vectors* $\mathbf{x}_0$ *such that the bound is attained.*

The theorem means that, in general, the residuals in the Chebyshev iteration converge linearly. The **asymptotic (root-)convergence factor** is bounded according to

$$\left(\frac{\|\mathbf{r}_n\|}{\|\mathbf{r}_0\|}\right)^{1/n} \leq \frac{\sqrt{\kappa_{\mathcal{I}}}-1}{\sqrt{\kappa_{\mathcal{I}}}+1} = \frac{1}{\vartheta} = \mathrm{e}^{-\operatorname{arcosh}(-\eta)} = |\eta| - \sqrt{\eta^2-1}\,. \tag{1.79}$$

Using some of the formulas given above, it is easy to show that the coefficients β_n and γ_n of the Chebyshev iteration converge as $n \to \infty$, that is $\beta_n \to \beta$ and $\gamma_n \to \gamma$. In fact, from (1.73) we have

$$\frac{T_{n-1}(\eta)}{T_n(\eta)} = -\frac{\vartheta^{n-1}+\vartheta^{-(n-1)}}{\vartheta^n+\vartheta^{-n}} \to -\frac{1}{\vartheta} \qquad \text{as} \quad n \to \infty\,.$$

Therefore, by (1.68b) and (1.68c), we have

$$\beta_{n-1} \to \beta :\equiv -\frac{\delta}{2\vartheta} = -\frac{\delta}{2}\left(|\eta| - \sqrt{\eta^2-1}\right), \tag{1.80}$$

$$\gamma_n \to \gamma :\equiv -\frac{\delta\vartheta}{2} = -\frac{\delta}{2}\left(|\eta| + \sqrt{\eta^2-1}\right), \tag{1.81}$$

and thus

$$\beta + \gamma = -\frac{\delta}{2}\left(\vartheta + \vartheta^{-1}\right) = \delta\eta = -\alpha\,, \tag{1.82}$$

as required by the limit of (1.68c).

Redefining in the recursions (1.68d) and (1.68e) $\gamma_0 := -\alpha$ and $\beta_{n-1} := \beta$, $\gamma_n := \gamma$ if $n > 0$, we obtain another Krylov space solver, which is also asymptotically optimal for the same interval and the same set of confocal ellipses. It is called **second-order Richardson iteration**. In contrast to the Chebyshev iteration it uses a stationary three-term recursion for $\mathbf{r}_n$, that is, the coefficients do not depend on n. Actually, the recursion involves again four terms, but the underlying recursion for the residual polynomials has only three terms if we write $\tau p_n(\tau) - \alpha p_n(\tau)$ as $(\tau - \alpha)p_n(\tau)$.

Like for SOR, the main disadvantage of these two methods is that some knowledge of the spectrum of $\mathbf{A}$ is required, in particular a lower bound for the distance of the smallest eigenvalue from the origin.

1.7 Preconditioning

When applied to large real-world problems Krylov space solvers often converge very slowly — if at all. In practice, Krylov space solvers are therefore nearly always applied with **preconditioning** [*Vorkonditionierung, Präkonditionierung*]. The basic idea behind it is to replace the given linear system $\mathbf{Ax} = \mathbf{b}$ by an equivalent one whose matrix is more suitable for a treatment by the chosen Krylov space method. In particular, it is normally expected to have much better condition. Ideal are matrices whose eigenvalues are clustered around one point except for a few outliers, and such that the cluster is well separated from the origin. Other properties, like the degree of nonnormality, also play a role, since highly nonnormal matrices often cause a delay of the convergence. Minor perturbations of such matrices can cause the spectrum to change much. Note that again, this new matrix needs not be available explicitly.

There are several ways of preconditioning. In the simplest case, called **left preconditioning** [*linke Vorkonditionierung*], the system is just multiplied from the left by some matrix $\mathbf{C}$ that is in some sense an approximation of the inverse of $\mathbf{A}$:

$$\boxed{\underbrace{\mathbf{CA}}_{\widehat{\mathbf{A}}}\,\mathbf{x} = \underbrace{\mathbf{Cb}}_{\widehat{\mathbf{b}}}\,.} \tag{1.83}$$

$\mathbf{C}$ is then called a **left preconditioner** [*linker Vorkonditionierer*] or, more appropriately, an **approximate inverse** [*approximative Inverse*] applied on the left-hand side. Of course, $\mathbf{C}$ should be sparse or specially structured, so that the matrix-vector product $\mathbf{Cy}$ can be calculated quickly for any $\mathbf{y}$.

Often, given is not $\mathbf{C}$ but its inverse $\mathbf{M} :\equiv \mathbf{C}^{-1}$, which is also called **left preconditioner**. In this case, we need to be able to solve $\mathbf{Mz} = \mathbf{y}$ quickly for any $\mathbf{y}$.

An alterative is to substitute $\mathbf{x}$ by $\widehat{\mathbf{x}} :\equiv \mathbf{C}^{-1}\mathbf{x}$, so that $\mathbf{Ax} = \mathbf{b}$ is replaced by

$$\boxed{\underbrace{\mathbf{AC}}_{\widehat{\mathbf{A}}}\,\underbrace{\mathbf{C}^{-1}\mathbf{x}}_{\widehat{\mathbf{x}}} = \mathbf{b}\,.} \tag{1.84}$$

This is **right preconditioning** [*rechte Vorkonditionierung*]. Here, logically, $\mathbf{C}$ is called a **right preconditioner** [*rechter Vorkonditionierer*] or, an approximate inverse applied on the right-hand side. Again, we may not have $\mathbf{C}$ available but its inverse $\mathbf{M}$. Note that in the situation of (1.84) we do not need $\mathbf{C}^{-1}$ because we will first obtain $\widehat{\mathbf{x}}$ and so have to compute $\mathbf{x} = \mathbf{C}\widehat{\mathbf{x}}$.

Sometimes it is most appropriate to combine these approaches by so-called **split preconditioning** [*gesplittete Vorkonditionierung*].

In general, the effect of preconditioning on the formulation and convergence of a Krylov space solver can be understood as the replacement of the system $\mathbf{A}\mathbf{x} = \mathbf{b}$ by

$$\boxed{\underbrace{\mathbf{C}_L\mathbf{A}\mathbf{C}_R}_{\widehat{\mathbf{A}}}\,\underbrace{\mathbf{C}_R^{-1}\mathbf{x}}_{\widehat{\mathbf{x}}} = \underbrace{\mathbf{C}_L\mathbf{b}}_{\widehat{\mathbf{b}}}\,,} \tag{1.85}$$

where we allow either $\mathbf{C}_L$ or $\mathbf{C}_R$ to be the identity, or by

$$\boxed{\underbrace{\mathbf{M}_L^{-1}\mathbf{A}\mathbf{M}_R^{-1}}_{\widehat{\mathbf{A}}}\,\underbrace{\mathbf{M}_R\mathbf{x}}_{\widehat{\mathbf{x}}} = \underbrace{\mathbf{M}_L^{-1}\mathbf{b}}_{\widehat{\mathbf{b}}}\,,} \tag{1.86}$$

where now at most either $\mathbf{M}_L$ or $\mathbf{M}_R$ is the identity. The product $\mathbf{C} :\equiv \mathbf{C}_L\,\mathbf{C}_R$ or the product $\mathbf{M} :\equiv \mathbf{M}_R\,\mathbf{M}_L$, respectively, is called **split preconditioner** [*gesplitteter Vorkonditionierer*].

Split preconditioning is particularly useful if $\mathbf{A}$ is real symmetric (or Hermitian) and we choose as the right preconditioner the transpose (or, in the complex case, the Hermitian transpose) of the left one, so that $\widehat{\mathbf{A}}$ is still symmetric. In particular, if $\mathbf{M}_R$ and $\mathbf{C}_L$ are lower triangular matrices $\mathbf{L}$ and $\mathbf{K}$, respectively, we have

$$\boxed{\mathbf{M} = \mathbf{L}\,\mathbf{L}^\star\,,} \qquad \boxed{\mathbf{C} = \mathbf{K}\,\mathbf{K}^\star\,,} \tag{1.87}$$

and these can be viewed as the **Cholesky decompositions** [*Cholesky-Zerlegungen*] of $\mathbf{M}$ and $\mathbf{C}$, respectively.

Left preconditioning effects the residuals: it involves the **preconditioned residual vectors**

$$\boxed{\widehat{\mathbf{r}}_n :\equiv \mathbf{C}_L\,\mathbf{r}_n = \mathbf{M}_L^{-1}\,\mathbf{r}_n = \mathbf{M}_L^{-1}\,(\mathbf{b}_n - \mathbf{A}\mathbf{x}_n)\,.} \tag{1.88}$$

On the other hand, right preconditioning effects the error vectors: it involves the **preconditioned error vectors**

$$\boxed{\widehat{\mathbf{x}}_n - \widehat{\mathbf{x}}_\star :\equiv \mathbf{M}_R\,(\mathbf{x}_n - \mathbf{x}_\star) = \mathbf{M}_R\,\mathbf{d}_n = \mathbf{C}_R^{-1}\,(\mathbf{x}_n - \mathbf{x}_\star)\,.} \tag{1.89}$$

Of course, the difficult part in preconditioning is to find the preconditioner, be it $\mathbf{C}_L$, $\mathbf{C}_R$, $\mathbf{M}_L$, $\mathbf{M}_R$, $\mathbf{L}$, or $\mathbf{K}$. Yet another question is how to efficiently combine the Krylov space solver with the preconditioner. Of course, one could just replace $\mathbf{A}$, $\mathbf{b}$, and $\mathbf{x}$ by $\widehat{\mathbf{A}}$, $\widehat{\mathbf{b}}$, and $\widehat{\mathbf{x}}$, but there are sometimes more efficient ways to build a preconditioner into a Krylov space solver. We will treat such cases in Section 2.7, where, in various ways, preconditioning is built into conjugate gradient algorithms.

1.8 Some Basic Preconditioning Techniques

1.8.1 Preconditioning based on classical matrix splittings

Recall from Section 1.4 that a matrix splitting $\mathbf{A} = \mathbf{M} - \mathbf{N}$ with appropriately chosen $\mathbf{N}$ may be used to speed up the linear fixed point iteration

$$\boxed{\mathbf{x}_{n+1} := \mathbf{B}\mathbf{x}_n + \mathbf{b}} \qquad \text{with} \quad \boxed{\mathbf{B} :\equiv \mathbf{I} - \mathbf{A}} \tag{1.90}$$

by replacing it by

$$\boxed{\mathbf{x}_{n+1} := \widehat{\mathbf{B}}\mathbf{x}_n + \widehat{\mathbf{b}}} \tag{1.91}$$

with

$$\boxed{\widehat{\mathbf{B}} :\equiv \mathbf{M}^{-1}\mathbf{N} = \mathbf{M}^{-1}(\mathbf{M} - \mathbf{A})\,, \qquad \widehat{\mathbf{b}} :\equiv \mathbf{M}^{-1}\mathbf{b}\,.} \tag{1.92}$$

While (1.90) is the straightforward fixed point iteration for the linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$, the modified iteration (1.91)–(1.92) is the fixed point iteration for the left-preconditioned system $\mathbf{M}^{-1}\mathbf{A}\mathbf{x} = \mathbf{M}^{-1}\mathbf{b}$.

So the methods based on matrix splittings that we discussed in Section 1.4 can all be understood as linear fixed point iterations for preconditioned systems. There the aim of the preconditioning was to make the spectral radius $\rho(\widehat{\mathbf{B}})$ as small as possible, since this spectral radius equals the asymptotic rate of convergence. In any case it has to be smaller than 1, since this is a necessary and sufficient condition for convergence for all $\mathbf{x}_0$. This condition also guarantees that $\widehat{\mathbf{A}} :\equiv \mathbf{M}^{-1}\mathbf{A} = \mathbf{I} - \widehat{\mathbf{B}}$ is nonsingular.

The same preconditioners $\mathbf{M}$ can also be used with other Krylov space solvers. For example, we could use the matrix $\mathbf{M}$ from a block Jacobi splitting or the one from an SSOR splitting as a left preconditioner in the Chebyshev method or in many of the Krylov space solvers we will discuss later. This simple approach to preconditioning is quite popular. Note that here the preconditioning has the effect that all eigenvalues of $\widehat{\mathbf{A}}$ lie in a circle of radius $\rho(\widehat{\mathbf{B}}) < 1$ around the point 1. We do not iterate with, say block Jacobi or SSOR, but we only use the underlying splitting to obtain a better conditioned matrix $\widehat{\mathbf{A}}$. In some sense, we combine each step of the Krylov space solver with one step of the iteration based on splitting.

The SOR splitting is not appropriate for preconditioning. In this case, as we mentioned, if $\mathbf{A}$ has Property A and is consistently ordered, and if $\mathbf{D}^{-1}\mathbf{A}$ has real eigenvalues, then the eigenvalues of $\widehat{\mathbf{B}}$ lie for the optimal ω on the circle with radius $\rho(\widehat{\mathbf{B}}) < 1$ and center 0, those of $\widehat{\mathbf{A}}$ lie on a circle with the same radius but center 1. This kind of spectrum is not very suitable for Krylov space solvers unless $\rho(\widehat{\mathbf{B}})$ is really small.

1.8.2 Incomplete LU and Cholesky factorizations

If we knew a Gaussian LU factorization of $\mathbf{A}$, say $\mathbf{A} = \mathbf{P}^\mathsf{T}\mathbf{L}\mathbf{U}$ with a permutation matrix $\mathbf{P}$, a lower triangular matrix $\mathbf{L}$, and an upper triangular matrix $\mathbf{U}$, we could choose

$$\mathbf{M} :\equiv \mathbf{P}^\mathsf{T}\mathbf{L}\mathbf{U} \tag{1.93}$$

as a left preconditioner, which means that $\widehat{\mathbf{A}} = \mathbf{M}^{-1}\mathbf{A} = \mathbf{I}$ would be optimally preconditioned. Application of this preconditioner would require to forward substitute with $\mathbf{U}$, back substitute with $\mathbf{L}$, and to permute components according to $\mathbf{P}$. But application of $\mathbf{M}^{-1}$ to $\mathbf{r}_0$ would yield in one step

$$\mathbf{x}_\star = \mathbf{x}_0 - \mathbf{d}_0 = \mathbf{x}_0 + \mathbf{M}^{-1}\mathbf{r}_0\,. \tag{1.94}$$

Of course, when we choose $\mathbf{x}_0 := \mathbf{o}$, so that $\mathbf{r}_0 = \mathbf{b}$, then this is essentially just the application of Gauss elimination.

As we mentioned in Section 1.2 the problem with the LU factorization is that for most large sparse matrices (except for banded ones with a very dense band) this approach is inefficient because the LU factors are much denser than $\mathbf{A}$ and thus their computation is costly and their memory requirement is large. The set of additional nonzero elements in $\mathbf{L}$ and $\mathbf{U}$ in positions of zero elements in $\mathbf{A}$ is called **fill-in**.

An often very effective alternative is to compute an approximate LU factorization that is as sparse as $\mathbf{A}$ or at least nearly as sparse: we choose sparse matrices $\mathbf{L}$ and $\mathbf{U}$ and a permutation matrix $\mathbf{P}$ so that the difference $\mathbf{L}\mathbf{U} - \mathbf{P}\mathbf{A}$ is small. The product $\mathbf{M} :\equiv \mathbf{P}^\mathsf{T}\mathbf{L}\mathbf{U} \approx \mathbf{A}$ is then called an **incomplete LU (ILU) factorization** [*unvollständige LU-Zerlegung*]. For the case of spd and Hpd matrices there are analogous products $\mathbf{M} :\equiv \mathbf{L}\mathbf{L}^\mathsf{T} \approx \mathbf{A}$ called **incomplete Cholesky (IC) factorization** [*unvollständige Cholesky-Zerlegung*].

There are many variants of ILU and IC factorizations. Often, no pivoting is used even in the unsymmetric ILU decomposition; that is, $\mathbf{P} = \mathbf{I}$. In the simplest variant we compute an LU or a Cholesky factorization, but where $\mathbf{A}$ has a zero element we replace any nonzero element of $\mathbf{L}$ or $\mathbf{U}$ by a zero. That is, zero fill-in is enforced.

The next step of sophistication is to prescribe some **pattern** [*Muster*] P of forced zeros:

$$\boxed{P \subseteq \big\{(i,j) \bigm| i \neq j,\ a_{i,j} = 0,\ 1 \leq i \leq N,\ 1 \leq j \leq N\big\}\ .} \tag{1.95}$$

One version of the corresponding ILU factorization algorithm without pivoting looks as follows:

Algorithm 1.2 (ILU factorization with fixed pattern P)

```
for $k = 1, \dots, n-1$ do
  for $i = k+1, \dots, n$ do
    if $(i,k) \notin P$ ,
      $a_{ik} := a_{ik}/a_{kk}$ ;
      for $j = k+1, \dots, n$ do
        if $(i,j) \notin P$ ,
          $a_{ij} := a_{ij} - a_{ik} * a_{kj}$ ;
        endif
      endfor
    endif
  endfor
endfor
```

Of course, there is no guarantee that this decomposition exists, even when $\mathbf{A}$ is nonsingular, since we do not include pivoting here. But worse, the ILU decomposition may not exist even when the full LU decomposition of $\mathbf{A}$ exists. For example, even when $\mathbf{A}$ is spd, the ILU decomposition or the corresponding IC decomposition need not exist. But there are classes of matrices for which it can be shown that the ILU decomposition exists, at least in exact arithmetic.

It is easy to show that after the ILU decomposition — if it can be completed — holds

$$\boxed{\mathbf{A} = \mathbf{LU} - \mathbf{R},} \tag{1.96}$$

where

$$\boxed{\begin{array}{lll} l_{ij} = 0 & \text{if} \quad i < j \quad \text{or} \quad (i,j) \in P\,, \\ u_{ij} = 0 & \text{if} \quad i < j \quad \text{or} \quad (i,j) \in P\,, \\ r_{ij} = 0 & \text{if} \quad (i,j) \notin P\,. \end{array}} \tag{1.97}$$

If one wants to avoid the assumption $(i,j) \in P \implies a_{i,j} = 0$ in the definition of P, then one needs the following assignment before executing the algorithm

$$\forall (i,j) \in P \text{ with } a_{ij} \neq 0 : \quad r_{ij} := -a_{ij}\,, \quad a_{ij} := 0\,. \tag{1.98}$$

But this is hardly ever required in practice.

In other versions of ILU algorithms the pattern P is not fixed in advance, but depends on the sizes of the matrix elements that are constructed: any constructed small element of $\mathbf{L}$ or $\mathbf{U}$ is deleted on the spot by comparison with a threshold (drop tolerance) T. This version is called **ILUT**. A detailed treatment of various ILU algorithms is given in Saad (1996).

Matlab provides for example:

`[L,U,P] = luinc(A,'0')`: ILU with pivoting, no fill-in
`[L,U,P] = luinc(A,droptol)`: ILUT with pivoting
`[L,U,P] = cholinc(A,'0')`: IC with no fill-in
`[L,U,P] = cholinc(A,droptol)`: IC with drop tolerance

1.8.3 Polynomial preconditioning

Given $\mathbf{A}$ and $\mathbf{c}$, a Krylov space solver applied to $\mathbf{Aw} = \mathbf{c}$ delivers approximations $\mathbf{w}_\ell$ of $\mathbf{w}_\star :\equiv \mathbf{A}^{-1}\mathbf{c}$, and according to (1.53) we have, when choosing $\mathbf{w}_0 := \mathbf{o}$,

$$\mathbf{w}_\ell = q_{\ell-1}(\mathbf{A})\mathbf{c} \in \mathcal{K}_\ell(\mathbf{A}, \mathbf{c})\,. \tag{1.99}$$

In some Krylov space solvers, the polynomial $q_{\ell-1} \in \mathcal{P}_{\ell-1}$ depends on the right-hand side $\mathbf{c}$, but in others, like Jacobi or Chebyshev iteration, it does not. Let us assume the latter case. Then $q_{\ell-1}(\mathbf{A})$ can be viewed as an operator that maps any $\mathbf{c}$ into an approximation of $\mathbf{A}^{-1}\mathbf{c}$. In particular, if $\mathbf{c} := \mathbf{Aw}$ is considered as an image point of $\mathbf{A}$ (and if $\mathbf{A}$ is nonsingular, it always can be considered so), then $q_{\ell-1}(\mathbf{A})\mathbf{A}$ can be viewed as an approximation of the identity. So, $\mathbf{C} := q_{\ell-1}(\mathbf{A})$ is an approximate inverse of $\mathbf{A}$. It is often called **polynomial preconditioner** [*polynomialer Präkonditionierer*].

In summary: *given any Krylov space solver whose recurrence coefficients only depend on* $\mathbf{A}$ *but not on* $\mathbf{c}$*, and given any fixed iteration number* ℓ*, if* $q_{\ell-1} \in \mathcal{P}_{\ell-1}$ *is the polynomial representing the* ℓ*th iterate* $\mathbf{w}_\ell$ *when solving* $\mathbf{Aw} = \mathbf{c}$*, then* $\mathbf{C} := q_{\ell-1}(\mathbf{A})$ *is an approximate inverse of* $\mathbf{A}$.

To implement this preconditioner for computing, say, $\mathbf{CAz}$, we first compute $\mathbf{c} := \mathbf{Az}$ and then perform ℓ steps of the Krylov space solver applied to $\mathbf{Aw} = \mathbf{c}$; so that $\mathbf{w}_\ell = \mathbf{CAz}$.

As early as 1959, it was suggested by Rutishauser (1959) to use Chebyshev iteration in this way as a preconditioner (although the notion of "preconditioning" did not yet exist at that time).

1.8.4 Inner-outer iteration

A natural generalization of polynomial preconditioning is to allow any Krylov space solver, even one where $q_{\ell-1}$ depends on $\mathbf{c}$, and also to allow ℓ to vary from one application of $\mathbf{C}$ to the next; so we should write $\mathbf{C}_n$ when applying the **inner iteration** [*innere Iteration*] the nth time, that is, in the nth step of the **outer iteration** [*äussere Iteration*]. Instead of choosing ℓ in advance, we may then terminate each inner iteration when the inner residual

$$\mathbf{c}_n - \mathbf{A}\mathbf{w}_{n,\ell_n} = \mathbf{c}_n - \mathbf{A}\underbrace{q_{n,\ell_n-1}(\mathbf{A})}_{\equiv:\,\mathbf{C}_n}\mathbf{c}_n = \underbrace{(\mathbf{I} - \mathbf{A}q_{n,\ell_n-1}(\mathbf{A}))}_{\equiv:\,p_{n,\ell_n}(\mathbf{A})}\mathbf{c}_n$$

is considered small enough. It need not be very small. The combination of such a **flexible preconditioning** [*flexible Vorkonditionierung*] with a Krylov space solver as outer iteration is called **inner-outer iteration** and has become very fashionable.

Chapter 2

The Conjugate Gradient Method

2.1 Energy Norm Minimization

In many areas of science and technology stable states are characterized by minimum energy. Discretization then leads in the first approximation to the minimization of a quadratic function in several variables,

$$\boxed{\Psi(\mathbf{x}) :\equiv \tfrac{1}{2}\,\mathbf{x}^{\mathsf{T}}\mathbf{A}\mathbf{x} - \mathbf{b}^{\mathsf{T}}\mathbf{x} + \gamma} \tag{2.1}$$

with an spd matrix $\mathbf{A}$. (We assume real data in Sections 2.1–2.6.) Such a function Ψ is well known to be convex since its second derivative is the matrix $\mathbf{A}$. So it has a unique minimum, which can be found by setting the first derivative, the gradient, equal to $\mathbf{o}$. The gradient can be seen to be

$$\boxed{\nabla\Psi(\mathbf{x}) = \mathbf{A}\mathbf{x} - \mathbf{b} = -\mathbf{r}\,,} \tag{2.2}$$

where $\mathbf{r}$ is the residual corresponding to $\mathbf{x}$. Hence,

$$\boxed{\mathbf{x} \text{ minimizer of } \Psi \quad \Longleftrightarrow \quad \nabla\Psi(\mathbf{x}) = \mathbf{o} \quad \Longleftrightarrow \quad \mathbf{A}\mathbf{x} = \mathbf{b}\,.} \tag{2.3}$$

As before, we let $\mathbf{x}_\star$ be the minimizer, *i.e.*, the solution of $\mathbf{A}\mathbf{x} = \mathbf{b}$, and $\mathbf{d} :\equiv \mathbf{x} - \mathbf{x}_\star$ be the error of $\mathbf{x}$. If we define the $\mathbf{A}$-norm as usual by

$$\|\mathbf{y}\|_{\mathbf{A}} :\equiv \sqrt{\mathbf{y}^{\mathsf{T}}\mathbf{A}\mathbf{y}}\,, \tag{2.4}$$

it is easily seen that

$$\boxed{\|\mathbf{d}\|_{\mathbf{A}}^2 = \|\mathbf{x} - \mathbf{x}_\star\|_{\mathbf{A}}^2 = \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_{\mathbf{A}^{-1}}^2 = \|\mathbf{r}\|_{\mathbf{A}^{-1}}^2 = 2\,\Psi(\mathbf{x})} \tag{2.5}$$

if in (2.1)

$$\gamma :\equiv \tfrac{1}{2}\,\mathbf{b}^{\mathsf{T}}\mathbf{A}^{-1}\mathbf{b}\,. \tag{2.6}$$

Of course, the value of the constant γ has no influence on the solution, so we can assume from now on that we have made this choice; thus, in other words, we are minimizing the $\mathbf{A}$-norm of the error, which is often referred to as the **energy norm** [*Energienorm*].

In summary: If $\mathbf{A}$ *is spd, to minimize the quadratic function* Ψ *means to minimize the energy norm of the error vector of the linear system* $\mathbf{A}\mathbf{x} = \mathbf{b}$.

Since $\mathbf{A}$ is spd, the level curves $\Psi(\mathbf{x}) = \mathsf{const}$ are ellipses if $N = 2$ and ellipsoids if $N = 3$.

2.2 Steepest Descent

The interpretation of the solution of a linear system as the minimizer of convex quadratic function Ψ suggests to find this minimizer by moving down the surface representing Ψ in the direction of steepest descent given by minus the gradient.

However, to follow approximately the so defined curve would require to make many small steps and to evaluate the gradient at every step. We prefer to follow a piecewise straight line with rather long pieces, that is, to make a long step whenever the gradient and thus the direction of steepest descent has been computed. So, in the nth step, we proceed from $(\mathbf{x}_n, \Psi(\mathbf{x}_n))$ on a straight line in the opposite direction of the gradient $\nabla\Psi(\mathbf{x})$, that is, in the direction of $\mathbf{r}_n$. A natural choice is to go to the point with the minimum value of Ψ on that line. In general, finding the minimum value of a functional on a line is called **line search**, but here, since Ψ is quadratic, the length of the step is easy to compute. On the line

$$\omega \mapsto \mathbf{x}_n + \mathbf{r}_n\omega \tag{2.7}$$

the minimum

$$\boxed{\mathbf{x}_{n+1} := \mathbf{x}_n + \mathbf{r}_n\omega_n} \tag{2.8}$$

of $\|\mathbf{x}_{n+1} - \mathbf{x}_\star\|^2_{\mathbf{A}} = \|\mathbf{r}_{n+1}\|^2_{\mathbf{A}^{-1}}$ is readily found: $\mathbf{x}_{n+1} = \mathbf{x}_n + \mathbf{r}_n\omega_n$ implies that $\mathbf{r}_{n+1} = \mathbf{r}_n - \mathbf{A}\mathbf{r}_n\omega_n$, so that

$$\begin{aligned}
\|\mathbf{r}_{n+1}\|^2_{\mathbf{A}^{-1}} &= \|\mathbf{r}_n - \mathbf{A}\mathbf{r}_n\omega_n\|^2_{\mathbf{A}^{-1}} \\
&= \|\mathbf{r}_n\|^2_{\mathbf{A}^{-1}} - 2\,\langle \mathbf{r}_n, \mathbf{A}\mathbf{r}_n\rangle_{\mathbf{A}^{-1}}\,\omega_n + \|\mathbf{A}\mathbf{r}_n\|^2_{\mathbf{A}^{-1}}\,\omega_n^2 \\
&= \|\mathbf{r}_n\|^2_{\mathbf{A}^{-1}} - 2\,\langle \mathbf{r}_n, \mathbf{r}_n\rangle\,\omega_n + \langle \mathbf{r}_n, \mathbf{A}\mathbf{r}_n\rangle\,\omega_n^2\,.
\end{aligned}$$

As a function of ω_n, this expression has a vanishing derivative (and is thus minimal) when

$$\boxed{\omega_n :\equiv \frac{\langle \mathbf{r}_n, \mathbf{r}_n\rangle}{\langle \mathbf{r}_n, \mathbf{A}\mathbf{r}_n\rangle}\,.} \tag{2.9}$$

This is the **method of steepest descent** [*Methode des steilsten Abstiegs*]. Comparing its formulas (2.8) and (2.9) with (1.21) we see that it differs from the Jacobi iteration only in the (locally optimal) choice of the step length.

However, this method may converge very slowly even if N is only 2. This is easily seen if we assume that $N = 2$ and that the two (positive) eigenvalues of $\mathbf{A}$ differ very much in size. Then the level curves of Ψ are concentric ellipses with a large axis ratio of λ_1/λ_2, and the search directions are orthogonal to these ellipses. Obviously, there are situations, where it takes many steps to come close to the center of the ellipses; see Figure 2.1. It is also clear that, in general, this method does not converge in at most N steps.

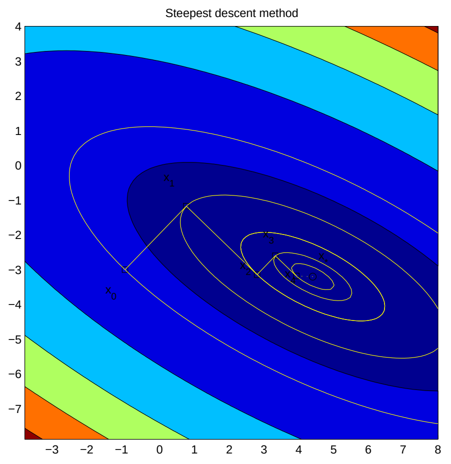


Figure 2.1: The steepest descent method for $N = 2$.

2.3 Conjugate Direction Methods

Figure 2.1 manifests where the slow convergence of the steepest descent method comes from: while the direction of steepest descent is optimal on an infinitesimally small scale, it may be far from optimal when we take long steps. For an optimal step in the two-dimensional situation depicted we would have to choose as the second direction one that leads directly to the center of the ellipse. In elementary geometry we have learned that this second direction must be **conjugate** [*konjugiert*] to the first one, which at $\mathbf{x}_1 := \mathbf{x}_0 + \mathbf{v}_0\omega_0$ is tangential to the ellipse; in other words, we need $\mathbf{v}_1^\mathsf{T}\mathbf{A}\mathbf{v}_0 = 0$, see Figure 2.2.

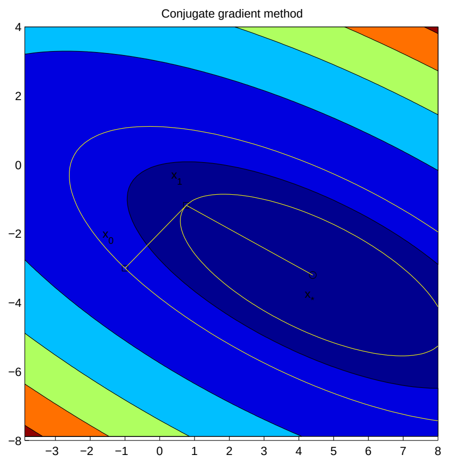


Figure 2.2: Conjugate directions — the CG method for $N = 2$.

How does this generalize to N dimensions? We choose nonzero **search directions** [*Suchrichtungen*] or **direction vectors** [*Richtungsvektoren*] $\mathbf{v}_n$ $(n = 0, 1, 2, \dots)$ that are conjugate or $\mathbf{A}$–orthogonal to each other, that is

$$\boxed{\mathbf{v}_n^\mathsf{T}\mathbf{A}\mathbf{v}_k = 0\,, \qquad k = 0, \dots, n-1,} \tag{2.10}$$

and define

$$\boxed{\mathbf{x}_{n+1} := \mathbf{x}_n + \mathbf{v}_n \omega_n\,,} \tag{2.11}$$

so that

$$\boxed{\mathbf{r}_{n+1} = \mathbf{r}_n - \mathbf{A}\mathbf{v}_n \omega_n\,.} \tag{2.12}$$

We choose the step length ω_n again such that the $\mathbf{A}$-norm of the error (that is, the $\mathbf{A}^{-1}$-norm of the residual) is minimized on the straight line

$$\omega \mapsto \mathbf{x}_n + \mathbf{v}_n \omega\,. \tag{2.13}$$

More generally: for any spd matrix $\mathbf{C}$, the minimum of

$$\|\mathbf{r}_n - \mathbf{A}\mathbf{v}_n\omega\|^2_{\mathbf{C}} = \|\mathbf{r}_n\|^2_{\mathbf{C}} - 2\,\langle \mathbf{r}_n, \mathbf{A}\mathbf{v}_n\rangle_{\mathbf{C}}\;\omega + \|\mathbf{A}\mathbf{v}_n\|^2_{\mathbf{C}}\,\omega^2 \tag{2.14}$$

on this line is at

$$\boxed{\omega_n :\equiv \frac{\langle \mathbf{r}_n, \mathbf{A}\mathbf{v}_n\rangle_{\mathbf{C}}}{\|\mathbf{A}\mathbf{v}_n\|^2_{\mathbf{C}}}\,.} \tag{2.15}$$

To minimize the $\mathbf{A}^{-1}$-norm of the residual, we choose $\mathbf{C} = \mathbf{A}^{-1}$, so

$$\boxed{\omega_n :\equiv \frac{\langle \mathbf{r}_n, \mathbf{v}_n\rangle}{\langle \mathbf{v}_n, \mathbf{A}\mathbf{v}_n\rangle}\,.} \tag{2.16}$$

Definition. Any iterative method satisfying (2.10), (2.11), and (2.16) is called a **conjugate direction method** [*Methode der konjugierten Richtungen*]. ▲

By definition, such a method chooses the step length ω_n so that $\mathbf{x}_{n+1}$ is locally optimal on the straight line (2.13). But does it also yield the best

$$\mathbf{x}_{n+1} \in \mathbf{x}_0 + \mathsf{span}\,\{\mathbf{v}_0, \dots, \mathbf{v}_n\} \tag{2.17}$$

with respect to the $\mathbf{A}$–norm of the error? We show next that due to choosing conjugate directions this is indeed true.

Theorem 2.3.1 *For a conjugate direction method the problem of minimizing the energy norm of the error of an approximate solution of the form* (2.17) *decouples into $n+1$ one-dimensional minimization problems on the lines $\omega \mapsto \mathbf{x}_k + \mathbf{v}_k\omega$, $k = 0, 1, \dots, n$. A conjugate direction method yields after $n+1$ steps the approximate solution of the form* (2.17) *that minimizes the energy norm* ($\mathbf{A}$*–norm*) *of the error in this affine space.*

Proof. By (2.5),

$$\begin{aligned}
\Psi(\mathbf{x}_{n+1}) &= \tfrac{1}{2}\,\|\mathbf{x}_{n+1} - \mathbf{x}_\star\|^2_{\mathbf{A}} \\
&= \tfrac{1}{2}\,\|\mathbf{x}_n + \mathbf{v}_n\omega_n - \mathbf{x}_\star\|^2_{\mathbf{A}} \\
&= \tfrac{1}{2}\,\|(\mathbf{x}_n - \mathbf{x}_\star) + \mathbf{v}_n\omega_n\|^2_{\mathbf{A}} \\
&= \Psi(\mathbf{x}_n) + \omega_n \mathbf{v}_n^{\mathsf{T}}\mathbf{A}(\mathbf{x}_n - \mathbf{x}_\star) + \tfrac{1}{2}\,\omega_n^2 \mathbf{v}_n^{\mathsf{T}}\mathbf{A}\mathbf{v}_n \\
&= \Psi(\mathbf{x}_n) - \omega_n \mathbf{v}_n^{\mathsf{T}}\mathbf{r}_n + \tfrac{1}{2}\,\omega_n^2 \mathbf{v}_n^{\mathsf{T}}\mathbf{A}\mathbf{v}_n\,.
\end{aligned}$$

As a consequence of (2.10) we get

$$\begin{aligned}\mathbf{v}_n^{\mathsf{T}}\mathbf{r}_n &= \mathbf{v}_n^{\mathsf{T}}\left(\mathbf{r}_0 + \mathbf{A}(\mathbf{x}_0 - \mathbf{x}_n)\right) \\ &= \mathbf{v}_n^{\mathsf{T}}\mathbf{r}_0 - \mathbf{v}_n^{\mathsf{T}}\mathbf{A}(\mathbf{v}_0\omega_0 + \cdots + \mathbf{v}_{n-1}\omega_{n-1}) \\ &= \mathbf{v}_n^{\mathsf{T}}\mathbf{r}_0\,.\end{aligned}$$

Therefore,

$$\Psi(\mathbf{x}_{n+1}) = \Psi(\mathbf{x}_n) - \omega_n\mathbf{v}_n^{\mathsf{T}}\mathbf{r}_0 + \tfrac{1}{2}\,\omega_n^2\mathbf{v}_n^{\mathsf{T}}\mathbf{A}\mathbf{v}_n\,. \qquad (2.18)$$

Here, the last two terms on the right-hand side are independent of $\mathbf{v}_0$, …, $\mathbf{v}_{n-1}$ and ω_0, …, ω_{n-1}. So from (2.18) we conclude that the problem of finding the global minimum with respect to the search directions $\mathbf{v}_0, \dots, \mathbf{v}_n$ decouples into the one of minimizing with respect to $\mathbf{v}_0, \dots, \mathbf{v}_{n-1}$ (which by induction we may assume to yield $\mathbf{x}_n$) and the one-dimensional minimization (line search) with respect to $\mathbf{v}_n$. The optimal ω_n in (2.18) is given by $\omega_n = (\mathbf{v}_n^{\mathsf{T}}\mathbf{r}_0)/(\mathbf{v}_n^{\mathsf{T}}\mathbf{A}\mathbf{v}_n)$, but in view of (2.10) this yields the same as (2.16). By induction, finding the global minimum of $\Psi(\mathbf{x}_n)$ decouples further into n one-dimensional minimum problems. $\square$

Conjugate direction methods in this general sense, as well as the special case of the conjugate gradient method treated next have been introduced by Hestenes and Stiefel (1952).

2.4 The Conjugate Gradient (CG) method

In general, conjugate direction methods are not Krylov space solvers, but if we choose the search directions in a suitable Krylov subspace, we obtain one. From the relation

$$\mathbf{x}_{n+1} = \mathbf{x}_0 + \mathbf{v}_0\omega_0 + \cdots + \mathbf{v}_n\omega_n \in \mathbf{x}_0 + \mathsf{span}\,\{\mathbf{v}_0, \mathbf{v}_1, \dots, \mathbf{v}_n\} \qquad (2.19)$$

we see that we need to choose search directions $\mathbf{v}_0, \dots, \mathbf{v}_n$ so that

$$\mathsf{span}\,\{\mathbf{v}_0, \dots, \mathbf{v}_n\} = \mathcal{K}_{n+1}(\mathbf{A}, \mathbf{r}_0)\,, \qquad n = 0, 1, 2, \dots\,. \qquad (2.20)$$

Actually, we need primarily that $\mathsf{span}\,\{\mathbf{v}_0, \dots, \mathbf{v}_n\} \subseteq \mathcal{K}_{n+1}$, but the conjugacy condition (2.10) and the requirement $\mathbf{v}_n \neq 0$ imply that equality must hold.

DEFINITION. The **conjugate gradient (CG) method** [*Methode der konjugierten Gradienten*] is the conjugate direction method with the choice (2.20). ▲

Theorem 2.3.1 immediately leads to the main result on this method:

THEOREM 2.4.1 *The* CG *method yields approximate solutions $\mathbf{x}_n \in \mathbf{x}_0 + \mathcal{K}_n(\mathbf{A}, \mathbf{r}_0)$ that are optimal in the sense that they minimize the energy norm ($\mathbf{A}$–norm) of the error vector for $\mathbf{x}_n$ from this affine space, or, equivalently, it minimizes the $\mathbf{A}^{-1}$–norm of the residual.*

By the two conditions (2.10) and (2.20) the search directions $\mathbf{v}_n$ have to satisfy, they are uniquely determined up to a scalar factor. But how can we construct them efficiently?

We note that if $\mathbf{r}_n \in \mathcal{K}_{n+1} \backslash \mathcal{K}_n$, we may normalize $\mathbf{v}_n$ so that

$$\mathbf{v}_n - \mathbf{r}_n \in \mathcal{K}_n \,. \tag{2.21}$$

Before we can proceed we need

Lemma 2.4.2 *The* CG *method produces mutually orthogonal residuals:*

$$\boxed{\mathbf{r}_n^\mathsf{T} \mathbf{r}_k = 0 \,, \qquad k = 0, \dots, n-1 \,.} \tag{2.22}$$

So, if $\mathbf{r}_0, \dots, \mathbf{r}_{n-1} \neq \mathbf{o}$,

$$\mathsf{span}\, \{\mathbf{r}_0, \dots, \mathbf{r}_{n-1}\} = \mathcal{K}_n(\mathbf{A}, \mathbf{r}_0) \perp \mathbf{r}_n \,. \tag{2.23}$$

Proof. Let us insert the optimal ω_n of (2.16) into the update formula (2.12) for $\mathbf{r}_n$:

$$\begin{aligned} \mathbf{r}_{n+1} &= \mathbf{r}_n - \mathbf{A}\mathbf{v}_n \omega_n \\ &= \mathbf{r}_n - \mathbf{A}\mathbf{v}_n \frac{\langle \mathbf{r}_n, \mathbf{v}_n \rangle}{\langle \mathbf{v}_n, \mathbf{A}\mathbf{v}_n \rangle} \,. \end{aligned} \tag{2.24}$$

By induction, assuming that $\mathbf{o} \neq \mathbf{r}_k \perp \mathcal{K}_k$ for $k \leq n$, which implies that (2.23) holds, we see using (2.21) and (2.10), respectively, that

$$\langle \mathbf{r}_n, \mathbf{r}_n \rangle = \langle \mathbf{r}_n, \mathbf{v}_n \rangle \,, \qquad \langle \mathbf{r}_n, \mathbf{A}\mathbf{v}_n \rangle = \langle \mathbf{v}_n, \mathbf{A}\mathbf{v}_n \rangle \,. \tag{2.25}$$

Multiplying (2.24) from the left by $\mathbf{r}_n^\mathsf{T}$ we conclude that $\mathbf{r}_{n+1} \perp \mathbf{r}_n$. And multiplying it by $\mathbf{r}_k^\mathsf{T}$ with $k < n$ shows likewise that $\mathbf{r}_{n+1} \perp \mathbf{r}_k$. So, (2.22) and (2.23) hold with n replaced by $n+1$. $\square$

Note that in the formula (2.16) for ω_n we can replace $\langle \mathbf{r}_n, \mathbf{v}_n \rangle$ by $\langle \mathbf{r}_n, \mathbf{r}_n \rangle$; see (2.25). So,

$$\boxed{\omega_n = \frac{\langle \mathbf{r}_n, \mathbf{r}_n \rangle}{\langle \mathbf{v}_n, \mathbf{A}\mathbf{v}_n \rangle} = \frac{\delta_n}{\delta'_n} \,,} \tag{2.26}$$

where $\delta_n :\equiv \langle \mathbf{r}_n, \mathbf{r}_n \rangle$, $\delta'_n :\equiv \langle \mathbf{v}_n, \mathbf{A}\mathbf{v}_n \rangle$.

In view of the above we can choose

$$\mathbf{v}_{n+1} := \mathbf{r}_{n+1} - \text{linear combination of} \quad \mathbf{v}_0, \mathbf{v}_1, \dots, \mathbf{v}_n \,.$$

But we may try the simpler ansatz

$$\boxed{\mathbf{v}_{n+1} := \mathbf{r}_{n+1} - \mathbf{v}_n \psi_n \,.} \tag{2.27}$$

Since $\mathbf{A}\mathbf{v}_k \in \mathcal{K}_{n+1}$ for $k < n$, we find from Lemma 2.4.2 that $(\mathbf{A}\mathbf{v}_k)^\mathsf{T} \mathbf{r}_{n+1} = 0$ for $k < n$. Therefore, by multiplying (2.27) from the left by $(\mathbf{A}\mathbf{v}_k)^\mathsf{T}$ with $k < n$ we see that $\langle \mathbf{v}_{n+1}, \mathbf{A}\mathbf{v}_k \rangle = 0$ $(k < n)$ as required. Moreover, we also get $\langle \mathbf{v}_{n+1}, \mathbf{A}\mathbf{v}_n \rangle = 0$ if we choose

$\psi_n = \langle \mathbf{r}_{n+1}, \mathbf{A}\mathbf{v}_n \rangle / \langle \mathbf{v}_n, \mathbf{A}\mathbf{v}_n \rangle$. Substituting in numerator and denominator $\mathbf{A}\mathbf{v}_n = (\mathbf{r}_n - \mathbf{r}_{n+1})\frac{1}{\omega_n}$ (from the update formula (2.21) for $\mathbf{r}_n$) and making use of $\langle \mathbf{v}_n, \mathbf{r}_n \rangle = \langle \mathbf{r}_n, \mathbf{r}_n \rangle$ we finally obtain

$$\boxed{\psi_n :\equiv \frac{\langle \mathbf{r}_{n+1}, \mathbf{A}\mathbf{v}_n \rangle}{\langle \mathbf{v}_n, \mathbf{A}\mathbf{v}_n \rangle} = -\frac{\langle \mathbf{r}_{n+1}, \mathbf{r}_{n+1} \rangle}{\langle \mathbf{r}_n, \mathbf{r}_n \rangle} = -\frac{\delta_{n+1}}{\delta_n}\,.} \tag{2.28}$$

Putting things together we now get a detailed algorithm for the CG method. We will see that there exist other algorithms using different recursions, but here we get the (standard) **Hestenes-Stiefel algorithm**, which is also called **coupled two-term version** or OMIN **version** of the method. The terminology "coupled two-term" refers to the fact that for both the residuals and the search directions two-term recursions, namely (2.12) and (2.27), are used for updating. The similar recursion for the iterates $\mathbf{x}_n$ follows from the one for the residuals. Again, these recursions actually contain three terms, but only two belong to the same sequence. The name "OMIN" refers to orthogonality and minimality and will be put in a broader context later.

Algorithm 2.1 (OMIN FORM OF THE CG METHOD) .
For solving $\mathbf{A}\mathbf{x} = \mathbf{b}$ *choose an initial approximation* $\mathbf{x}_0$, *and let* $\mathbf{v}_0 := \mathbf{r}_0 := \mathbf{b} - \mathbf{A}\mathbf{x}_0$ *and* $\delta_0 := \|\mathbf{r}_0\|^2$. *Then, for* $n = 0, 1, 2, \dots,$ *compute*

$$\begin{aligned}
\delta'_n &:= \|\mathbf{v}_n\|^2_{\mathbf{A}}\,, && (2.29a)\\
\omega_n &:= \delta_n/\delta'_n\,, && (2.29b)\\
\mathbf{x}_{n+1} &:= \mathbf{x}_n + \mathbf{v}_n\omega_n\,, && (2.29c)\\
\mathbf{r}_{n+1} &:= \mathbf{r}_n - \mathbf{A}\mathbf{v}_n\omega_n\,, && (2.29d)\\
\delta_{n+1} &:= \|\mathbf{r}_{n+1}\|^2, && (2.29e)\\
\psi_n &:= -\delta_{n+1}/\delta_n\,, && (2.29f)\\
\mathbf{v}_{n+1} &:= \mathbf{r}_{n+1} - \mathbf{v}_n\psi_n\,. && (2.29g)
\end{aligned}$$

If $\|\mathbf{r}_{n+1}\| \leq \mathrm{tol}$, *the algorithm terminates and* $\mathbf{x}_{n+1}$ *is a sufficiently accurate approximation of the solution.*

In MATLAB: `x = pcg(A,b)` , but there are many options:

```
[X,FLAG,RELRES,ITER,RESVEC] = PCG(A,B,TOL,MAXIT,M1,M2,X0)
```

These options include: multiple right-hand sides, tolerance, maximum number of iterations, preconditioning, stopping flag, residual norm, iteration number, residual history.

Since the CG method produces optimal approximations from the affine spaces $\mathbf{x}_0 + \mathcal{K}_n$ it is clear that it finds the solution in the minimum number of steps, and from Corollary 1.5.5 we know that $\bar{\nu}(\mathbf{r}_0, \mathbf{A})$ steps are needed:

Theorem 2.4.3 *The conjugate gradient method yields (in exact arithmetic) the solution of* $\mathbf{Ax} = \mathbf{b}$ *in the minimum number* $\bar{\nu}(\mathbf{r}_0, \mathbf{A})$ *of steps.*

However, it is important to point out that this is a theoretical result, which does not hold in practice due to rounding errors. These often have a very strong effect on the method. Nevertheless, in most cases the method still converges, and in fact delivers very small residuals in far less than $\bar{\nu}(\mathbf{r}_0, \mathbf{A})$ steps. We stop when some measure of the error is small enough, and $\|\mathbf{r}_{n+1}\|$ is a simple such measure. However, since the CG method minimizes the energy norm within each Krylov subspace, it might be more appropriate to check the size of $\|\mathbf{d}_{n+1}\|_{\mathbf{A}} = \|\mathbf{r}_{n+1}\|_{\mathbf{A}^{-1}}$. Hestenes and Stiefel (1952) also showed how this can be done efficiently.

Next we want to illustrate how the recursions Krylov space solvers are based on can be described by matrix equations. We note that the recursions (2.29c), (2.29d), and (2.29g) mean that

$$\begin{aligned} \mathbf{v}_n &= -\,(\mathbf{x}_n - \mathbf{x}_{n+1})\,\tfrac{1}{\omega_n}\,, \\ \mathbf{Av}_n &= (\mathbf{r}_n - \mathbf{r}_{n+1})\,\tfrac{1}{\omega_n}\,, \\ \mathbf{r}_{n+1} &= \mathbf{v}_{n+1} + \mathbf{v}_n \psi_n\,. \end{aligned}$$

If we let

$$\mathbf{R}_m :\equiv \begin{pmatrix} \mathbf{r}_0 & \mathbf{r}_1 & \cdots & \mathbf{r}_{m-1} \end{pmatrix}, \tag{2.30a}$$

$$\mathbf{V}_m :\equiv \begin{pmatrix} \mathbf{v}_0 & \mathbf{v}_1 & \cdots & \mathbf{v}_{m-1} \end{pmatrix}, \tag{2.30b}$$

$$\mathbf{X}_m :\equiv \begin{pmatrix} \mathbf{x}_0 & \mathbf{x}_1 & \cdots & \mathbf{x}_{m-1} \end{pmatrix}, \tag{2.30c}$$

and assume $m > n$, we can write them as

$$\mathbf{v}_n = -\underbrace{\begin{pmatrix} \mathbf{x}_0 & \cdots & \mathbf{x}_{n-1} & \mathbf{x}_n & \mathbf{x}_{n+1} & \mathbf{x}_{n+2} & \dots & \mathbf{x}_m \end{pmatrix}}_{\mathbf{X}_{m+1}} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \omega_n^{-1} \\ -\omega_n^{-1} \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

$$\mathbf{Av}_n = \underbrace{\begin{pmatrix} \mathbf{r}_0 & \cdots & \mathbf{r}_{n-1} & \mathbf{r}_n & \mathbf{r}_{n+1} & \mathbf{r}_{n+2} & \dots & \mathbf{r}_m \end{pmatrix}}_{\mathbf{R}_{m+1}} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \omega_n^{-1} \\ -\omega_n^{-1} \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

and

$$\mathbf{r}_{n+1} = \underbrace{\begin{pmatrix} \mathbf{v}_0 & \cdots & \mathbf{v}_{n-1} & \mathbf{v}_n & \mathbf{v}_{n+1} & \mathbf{v}_{n+2} & \dots & \mathbf{v}_{m-1} \end{pmatrix}}_{\mathbf{V}_m} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \psi_n \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} .$$

Let us further introduce the matrices

$$\mathbf{U}_m :\equiv \begin{pmatrix} 1 & \psi_0 & & & \\ & 1 & \psi_1 & & \\ & & 1 & \ddots & \\ & & & \ddots & \psi_{m-2} \\ & & & & 1 \end{pmatrix} , \tag{2.31}$$

$$\underline{\mathbf{L}}^{\circ}_m :\equiv \begin{pmatrix} \omega_0^{-1} & & & & \\ -\omega_0^{-1} & \omega_1^{-1} & & & \\ & -\omega_1^{-1} & \omega_2^{-1} & & \\ & & \ddots & \ddots & \\ & & & -\omega_{m-2}^{-1} & \omega_{m-1}^{-1} \\ & & & & -\omega_{m-1}^{-1} \end{pmatrix} , \tag{2.32}$$

$$\underline{\mathbf{E}}_m :\equiv \begin{pmatrix} 1 & & & & \\ -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \\ & & & & -1 \end{pmatrix} , \tag{2.33}$$

and

$$\mathbf{D}_{\omega;\, m} :\equiv \mathsf{diag}\left\{\omega_0, \omega_1, \dots, \omega_{m-1}\right\} , \tag{2.34}$$

so that

$$\underline{\mathbf{L}}^{\circ}_m = \underline{\mathbf{E}}_m \, \mathbf{D}^{-1}_{\omega;\, m} . \tag{2.35}$$

$\underline{\mathbf{L}}^{\circ}_m$ and $\underline{\mathbf{E}}_m$ are "extended" lower bidiagonal matrices of the size $(m+1) \times m$ whose column sums vanish[1].

In terms of all these matrices the coupled two-term CG recursions (2.29d) for $n = -1, \dots, m-2$ and (2.29g) for $n = 0, \dots, m-1$ can be summarized as

$$\boxed{\mathbf{R}_m = \mathbf{V}_m \mathbf{U}_m , \qquad \mathbf{A}\mathbf{V}_m = \mathbf{R}_{m+1} \underline{\mathbf{L}}^{\circ}_m \quad (m \leq \bar{\nu}) ,} \tag{2.36}$$

and the additional recursion (2.29c) for the iterates turns into

$$\boxed{\mathbf{V}_m = -\, \mathbf{X}_{m+1} \underline{\mathbf{L}}^{\circ}_m \quad (m \leq \bar{\nu}) .} \tag{2.37}$$

[1]By underlining $\mathbf{E}_m$ we want to indicate that we augment this matrix by an additional row. We suggest reading $\underline{\mathbf{E}}_m$ as "E sub m extended". The same notation will be used on other occasions.

Eliminating the direction vectors contained in $\mathbf{V}_m$ yields

$$\boxed{\mathbf{A}\mathbf{R}_m = \mathbf{R}_{m+1}\underline{\mathbf{T}}_m^\circ\,, \qquad \mathbf{R}_m = -\,\mathbf{X}_{m+1}\underline{\mathbf{T}}_m^\circ \quad (m \leq \bar{\nu})\,,} \tag{2.38}$$

where

$$\boxed{\underline{\mathbf{T}}_m^\circ :\equiv \underline{\mathbf{L}}_m^\circ \mathbf{U}_m} \tag{2.39}$$

is an 'extended' tridiagonal matrix:

$$\underline{\mathbf{T}}_m^\circ \equiv: \left(\begin{array}{c} \mathbf{T}_m^\circ \\ \hline \gamma_{m-1}\mathbf{l}_m^\mathsf{T} \end{array}\right) \equiv: \begin{pmatrix} \alpha_0 & \beta_0 & & & \\ \gamma_0 & \alpha_1 & \beta_1 & & \\ & \gamma_1 & \alpha_2 & \ddots & \\ & & \ddots & \ddots & \beta_{m-2} \\ & & & \gamma_{m-2} & \alpha_{m-1} \\ & & & & \gamma_{m-1} \end{pmatrix}, \tag{2.40}$$

where $\mathbf{l}_m^\mathsf{T}$ is the last row of the unit matrix of size m.

The characteristic property of a matrix $\mathbf{Z}$ with column sums 0 is that $\mathbf{e}^\mathsf{T}\mathbf{Z} = \mathbf{o}^\mathsf{T}$ for $\mathbf{e} :\equiv \begin{pmatrix} 1 & 1 & \dots 1 \end{pmatrix}^\mathsf{T}$ of appropriate size. So, $\mathbf{e}^\mathsf{T}\underline{\mathbf{L}}_m^\circ = \mathbf{o}^\mathsf{T}$, and therefore also

$$\mathbf{e}^\mathsf{T}\underline{\mathbf{T}}_m^\circ = \mathbf{e}^\mathsf{T}\underline{\mathbf{L}}_m^\circ\mathbf{U}_m = \mathbf{o}^\mathsf{T}\,. \tag{2.41}$$

Thus, $\underline{\mathbf{T}}_m^\circ$ has column sums zero too: if we let $\beta_{-1} :\equiv 0$, then

$$\boxed{\gamma_n = -\alpha_n - \beta_{n-1}\,, \qquad n = 0, 1, \dots, m-1 \leq \bar{\nu} - 1\,.} \tag{2.42}$$

According to (2.38) the CG residuals and iterates satisfy three-term recursions:

$$\mathbf{r}_{n+1}\gamma_n = (\mathbf{A}\mathbf{r}_n - \mathbf{r}_n\alpha_n - \mathbf{r}_{n-1}\beta_{n-1})\,, \tag{2.43a}$$

$$\mathbf{x}_{n+1}\gamma_n = -(\mathbf{r}_n + \mathbf{x}_n\alpha_n + \mathbf{x}_{n-1}\beta_{n-1})\,, \tag{2.43b}$$

but what remains to find are formulas for α_n and β_{n-1}. In Lemma 2.4.2 we have seen that the residuals are orthogonal to each other, and, when multiplying (2.43a) from the left by $\mathbf{r}_n^\mathsf{T}$ and $\mathbf{r}_{n-1}^\mathsf{T}$ this property gives

$$\alpha_n = \frac{\mathbf{r}_n^\mathsf{T}\mathbf{A}\mathbf{r}_n}{\mathbf{r}_n^\mathsf{T}\mathbf{r}_n}\,, \qquad \beta_{n-1} = \frac{\mathbf{r}_{n-1}^\mathsf{T}\mathbf{A}\mathbf{r}_n}{\mathbf{r}_{n-1}^\mathsf{T}\mathbf{r}_{n-1}}\,.$$

Here, since $\mathbf{A}$ is symmetric and since the three-term recursion and the orthogonality property of Lemma 2.4.2 hold,

$$\begin{aligned} \mathbf{r}_{n-1}^\mathsf{T}\mathbf{A}\mathbf{r}_n &= (\mathbf{A}\mathbf{r}_{n-1})^\mathsf{T}\mathbf{r}_n \\ &= (\mathbf{r}_n\gamma_{n-1} + \mathbf{r}_{n-1}\alpha_{n-1} + \mathbf{r}_{n-2}\beta_{n-2})^\mathsf{T}\mathbf{r}_n \\ &= \mathbf{r}_n^\mathsf{T}\mathbf{r}_n\gamma_{n-1}\,, \end{aligned}$$

so, we only need to compute two inner products, $\langle \mathbf{r}_n, \mathbf{A}\mathbf{r}_n \rangle$ and $\langle \mathbf{r}_n, \mathbf{r}_n \rangle$ per step since $\langle \mathbf{r}_{n-1}, \mathbf{r}_{n-1} \rangle$ and γ_{n-1} will be known from the previous step, while γ_n is computed from (2.42).

Summarizing everything we obtain a new algorithm for the CG method, which is called **three-term version** or ORES **version** of the method, the latter because it makes direct use of the orthogonality of the residuals.

Algorithm 2.2 (ORES FORM OF THE CG METHOD) .
For solving $\mathbf{Ax} = \mathbf{b}$ *with an Hpd matrix* $\mathbf{A}$*, choose an initial approximation* $\mathbf{x}_0$*, and let* $\mathbf{r}_0 := \mathbf{b} - \mathbf{Ax}_0$*,* $\beta_{-1} := 0$*, and* $\delta_0 := \|\mathbf{r}_0\|^2$*. Then, for* $n = 0, 1, \ldots,$ *compute*

$$\begin{aligned}
\alpha_n &:= \|\mathbf{r}_n\|^2_{\mathbf{A}} \,/\, \delta_n \,, && (2.44a)\\
\beta_{n-1} &:= \gamma_{n-1}\delta_n/\delta_{n-1} \qquad (\textit{if} \quad n > 0), && (2.44b)\\
\gamma_n &:= -\alpha_n - \beta_{n-1}\,, && (2.44c)\\
\mathbf{x}_{n+1} &:= -(\mathbf{r}_n + \mathbf{x}_n\alpha_n + \mathbf{x}_{n-1}\beta_{n-1})/\gamma_n\,, && (2.44d)\\
\mathbf{r}_{n+1} &:= (\mathbf{Ar}_n - \mathbf{r}_n\alpha_n - \mathbf{r}_{n-1}\beta_{n-1})/\gamma_n\,, && (2.44e)\\
\delta_{n+1} &:= \|\mathbf{r}_{n+1}\|^2\,. && (2.44f)
\end{aligned}$$

If $\|\mathbf{r}_{n+1}\| \leq \mathrm{tol}$*, the algorithm terminates and* $\mathbf{x}_{n+1}$ *is a sufficiently accurate approximation of the solution.*

Of course, $\mathbf{Ar}_n$, which is needed in (2.44a) and (2.44e), is computed only once and stored in a temporary vector.

There is yet another version of the CG method. Instead of $\mathbf{V}_m$ one can eliminate $\mathbf{R}_m$ in (2.36) to obtain

$$\boxed{\mathbf{AV}_m = \mathbf{V}_{m+1}\underline{\mathbf{T}}'_m \quad (m \leq \bar{\nu})\,,} \qquad (2.45)$$

where

$$\boxed{\underline{\mathbf{T}}'_m :\equiv \mathbf{U}_{m+1}\underline{\mathbf{L}}^\circ_m} \qquad (2.46)$$

is again an 'extended' tridiagonal matrix. Here, (2.45) describes a 3-term recursion for the direction vectors:

$$\mathbf{v}_{n+1}\gamma'_n = \mathbf{Av}_n - \mathbf{v}_n\alpha'_n - \mathbf{v}_{n-1}\beta'_{n-1}\,, \qquad (2.47)$$

where we are free to choose $\gamma'_n \neq 0$, while α'_n and β'_{n-1} have to be determined such that

$$\mathbf{v}_n^{\mathsf{T}}\mathbf{Av}_{n+1} = 0\,, \qquad \mathbf{v}_{n-1}^{\mathsf{T}}\mathbf{Av}_{n+1} = 0\,. \qquad (2.48)$$

It is easy to verify that the search direction are then all conjugate to each other, as required by (2.10). In general, $\underline{\mathbf{T}}'_m$ does not have column sums zero.

To update the residuals and iterates we can use (2.29d) and (2.29c) from the OMIN version. Altogether we obtain yet another algorithm, which is called ODIR version of the CG method, as it makes explicit usage of the $\mathbf{A}$–orthogonality of the search directions.

Algorithm 2.3 (ODir form of the CG method) .
For solving $\mathbf{A}\mathbf{x} = \mathbf{b}$, *choose an initial approximation* $\mathbf{x}_0$, *and let* $\mathbf{v}_0 := \mathbf{r}_0 := \mathbf{b} - \mathbf{A}\mathbf{x}_0$, $\beta'_{-1} := 0$.
Then, for $n = 0, 1, \ldots,$ *compute*

$$\delta'_n := \|\mathbf{v}_n\|^2_{\mathbf{A}}\,, \tag{2.49a}$$

$$\omega'_n := \langle \mathbf{r}_n, \mathbf{v}_n \rangle / \delta'_n\,, \tag{2.49b}$$

$$\mathbf{x}_{n+1} := \mathbf{x}_n + \mathbf{v}_n \omega'_n\,, \tag{2.49c}$$

$$\mathbf{r}_{n+1} := \mathbf{r}_n - \mathbf{A}\mathbf{v}_n \omega'_n\,, \tag{2.49d}$$

$$\alpha'_n := \|\mathbf{A}\mathbf{v}_n\|^2 / \delta'_n\,, \tag{2.49e}$$

$$\beta'_{n-1} := \gamma'_{n-1}\delta'_n / \delta'_{n-1} \qquad (\textit{if} \quad n > 0), \tag{2.49f}$$

$$\mathbf{v}_{n+1} := (\mathbf{A}\mathbf{v}_n - \mathbf{v}_n\alpha'_n - \mathbf{v}_{n-1}\beta'_{n-1}) / \gamma'_n\,, \tag{2.49g}$$

where $\gamma'_n \neq 0$ *can be chosen* (*e.g., to normalize* $\mathbf{v}_{n+1}$).
If $\|\mathbf{r}_{n+1}\| \leq \text{tol}$, *the algorithm terminates and* $\mathbf{x}_{n+1}$ *is a sufficiently accurate approximation of the solution.*

Remark. Both in (2.44e) and (2.49g) it is preferable to use the modified Gram-Schmidt algorithm, which also means to modify the formulas (2.44a) for α_n and (2.49e) for α'_n. But, since there are only two terms to subtract, the difference will be marginal. ▼

Remark. Each step in the sequence

$$\underline{\mathbf{T}}^{\circ}_m \quad \rightsquigarrow \quad (\,\underline{\mathbf{L}}^{\circ}_m,\ \mathbf{U}_m\,) \quad \rightsquigarrow \quad \underline{\mathbf{T}}'_m \rightsquigarrow \quad (\,\underline{\mathbf{L}}'_m,\ \mathbf{U}'_m\,) \tag{2.50}$$

is essentially half a step of Rutihauser's quotient-difference (qd) algorithm (Rutishauser, 1957), except that here the matrices are not square. ▼

2.5 The Conjugate Residual (CR) Method

We can replace the aim of minimizing the energy norm considered in Section 2.1 by the aim of minimizing the 2-norm of the residual. Let us still assume that $\mathbf{A}$ is symmetric positive definite (spd), so that this is the same as minimizing the $\mathbf{A}^2$–norm of the error:

$$\boxed{\|\mathbf{x} - \mathbf{x}_\star\|^2_{\mathbf{A}^2} = \|\mathbf{d}\|^2_{\mathbf{A}^2} = \|\mathbf{r}\|^2 = \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 = 2\,\widehat{\Psi}(\mathbf{x})} \tag{2.51}$$

if

$$\boxed{\widehat{\Psi}(\mathbf{x}) :\equiv \tfrac{1}{2}\,\mathbf{x}^{\mathsf{T}}\mathbf{A}^2\mathbf{x} - \mathbf{b}^{\mathsf{T}}\mathbf{A}\mathbf{x} + \tfrac{1}{2}\,\mathbf{b}^{\mathsf{T}}\mathbf{b}\,.} \tag{2.52}$$

Now we have

$$\boxed{\nabla\widehat{\Psi}(\mathbf{x}) = \mathbf{A}^2\mathbf{x} - \mathbf{A}\mathbf{b} = -\mathbf{A}\mathbf{r}} \tag{2.53}$$

and

$$\boxed{\mathbf{x} \text{ minimizer of } \widehat{\Psi} \quad \Longleftrightarrow \quad \nabla\widehat{\Psi}(\mathbf{x}) = \mathbf{o} \quad \Longleftrightarrow \quad \mathbf{A}\mathbf{r} = \mathbf{o}\,.} \tag{2.54}$$

It should be no surprise that we can develop a variation of the conjugate gradient method adapted to this error norm by replacing in the derivations of Sections 2.3 and 2.4 the occurrences of the **A**–inner product by those of the $\mathbf{A}^2$–inner product, and the occurrences of the standard inner product by those of the **A**–inner product. In particular, the search directions are chosen from a growing sequence of Krylov subspaces as in (2.20), but are now $\mathbf{A}^2$–orthogonal:

$$\boxed{\mathbf{v}_n^{\mathsf{T}} \mathbf{A}^2 \mathbf{v}_k = 0\,, \qquad k = 0, \dots, n-1\,.} \tag{2.55}$$

The new iterates are again found by line search,

$$\boxed{\mathbf{x}_{n+1} := \mathbf{x}_n + \mathbf{v}_n \omega_n\,,} \tag{2.56}$$

where now the step length ω_n is chosen such that the 2-norm of the residual is minimized on the line $\omega \mapsto \mathbf{x}_n + \mathbf{v}_n \omega$. In (2.15) this requires that $\mathbf{C} = \mathbf{I}$, hence

$$\boxed{\omega_n :\equiv \frac{\langle \mathbf{r}_n, \mathbf{A}\mathbf{v}_n \rangle}{\|\mathbf{A}\mathbf{v}_n\|^2}\,.} \tag{2.57}$$

DEFINITION. The Krylov space solver satisfying (2.20), (2.55), (2.56), and (2.57) is called the **conjugate residual (CR) method** [*Methode der konjugierten Residuen*]. ▲

The CR method comes again in three basic version, OMIN, ORES, and ODIR, analogous to those for CG. A straightforward adaptation leads to two matrix-vector products (MVs) per step, but it is possible to replace one by a recursion. (This causes additional roundoff error propagation, however.) Again some of the formulas for the coefficients can be simplified.

Below is an OMIN version that requires only one MV per iteration, namely for computing $\mathbf{A}\mathbf{r}_{n+1}$, which is needed in (2.58e) and is then also used in (2.58h). The algorithm requires to store the actual $\mathbf{w}_n :\equiv \mathbf{A}\mathbf{v}_n$, but this is only one extra vector in addition to the three that are needed in the OMIN version: $\mathbf{x}_n$, $\mathbf{r}_n$, and $\mathbf{v}_n$. The ORES version also needs four: $\mathbf{x}_n$, $\mathbf{x}_{n-1}$, $\mathbf{r}_n$, and $\mathbf{r}_{n-1}$.

Note that $\mathbf{w}_n = \mathbf{o}$ implies $n = \bar{\nu}$, hence, $\delta_n' :\equiv \|\mathbf{w}_n\|^2$ cannot vanish before the Krylov space is exhausted and we have found the solution of the linear system.

Algorithm 2.4 (OMin form of the CR method) .
For solving $\mathbf{Ax} = \mathbf{b}$ *choose an initial approximation* $\mathbf{x}_0$*, and let* $\mathbf{v}_0 := \mathbf{r}_0 := \mathbf{b} - \mathbf{Ax}_0$*,* $\mathbf{w}_0 := \mathbf{Av}_0$*, and* $\delta_0 := \|\mathbf{r}_0\|^2_{\mathbf{A}}$*. Then, for* $n = 0, 1, 2, \dots,$ *compute*

$$\delta'_n := \|\mathbf{w}_n\|^2 \,, \tag{2.58a}$$

$$\omega_n := \delta_n / \delta'_n \,, \tag{2.58b}$$

$$\mathbf{x}_{n+1} := \mathbf{x}_n + \mathbf{v}_n \omega_n \,, \tag{2.58c}$$

$$\mathbf{r}_{n+1} := \mathbf{r}_n - \mathbf{w}_n \omega_n \,, \tag{2.58d}$$

$$\delta_{n+1} := \|\mathbf{r}_{n+1}\|^2_{\mathbf{A}}, \tag{2.58e}$$

$$\psi_n := -\delta_{n+1} / \delta_n \,, \tag{2.58f}$$

$$\mathbf{v}_{n+1} := \mathbf{r}_{n+1} - \mathbf{v}_n \psi_n \,, \tag{2.58g}$$

$$\mathbf{w}_{n+1} := \mathbf{Ar}_{n+1} - \mathbf{w}_n \psi_n \,. \tag{2.58h}$$

If $\|\mathbf{r}_{n+1}\| \leq \mathrm{tol}$*, the algorithm terminates and* $\mathbf{x}_{n+1}$ *is a sufficiently accurate approximation of the solution.*

The conjugate residual method was introduces by Stiefel (1955), who gave formulas for an ORes version, but for one that differs from the analogue of our ORes version of CG. Stiefel used update formulas for the differences of two successive approximations and two successive residuals, respectively.

As in the various forms of the CG method, here there are again several theoretically equivalent ways to compute the quantities δ_n and δ'_n as inner products.

An interesting aspect of the ODir version is that it also applies to indefinite symmetric (or Hermitian) matrices. In fact, $\mathbf{A}^2$ is still spd (or Hpd), and thus the quantities $\|\mathbf{v}_n\|_{\mathbf{A}^2}$ that are analogous to δ'_n in Algorithm 2.3 cannot vanish as long as $\mathbf{v}_n \neq \mathbf{o}$.

Nevertheless, the ODir version of the CR method is no longer used much, since the mathematically equivalent MinRes algorithm has been said to be more stable. Only recently, it was pointed out by Sleijpen, van der Vorst and Modersitzki (2000) that with MinRes the attainable accuracy of the approximate solutions may be rather low for ill-conditioned matrices.

We will later treat a generalization of the CR method to nonsymmetric and non-Hermitian matrices.

2.6 A Bound for the Convergence

For the moment, let us assume that $\mathbf{A}$ is symmetric and consider the $\mathbf{C}$-norm of the error vector $\mathbf{d}_n$, where $\mathbf{C} = \mathbf{A}^\ell$ is a positive power of $\mathbf{A}$. If $\mathbf{A}$ is indefinite, we need an even power to make $\mathbf{C}$ Hpd. Then, since $\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{-1}$ with $\mathbf{U}$ orthogonal and $\mathbf{\Lambda}$ diagonal, and since, for any Krylov space solver, $\mathbf{d}_n = p_n(\mathbf{A})\mathbf{d}_0$ according to (1.59), we have

$$\begin{aligned}\|\mathbf{d}_n\|_{\mathbf{C}} &= \|\mathbf{A}^{\ell/2} p_n(\mathbf{A})\mathbf{d}_0\| = \|\mathbf{U} p_n(\mathbf{\Lambda})\mathbf{U}^{-1}\mathbf{A}^{\ell/2}\mathbf{d}_0\| \\ &\le \|p_n(\mathbf{\Lambda})\|\ \|\mathbf{A}^{\ell/2}\mathbf{d}_0\| = \|p_n(\mathbf{\Lambda})\|\ \|\mathbf{d}_0\|_{\mathbf{C}} \\ &= \max_{i=1,\dots,N} |p_n(\lambda_i)|\ \|\mathbf{d}_0\|_{\mathbf{C}} \,. \end{aligned} \tag{2.59}$$

Further, we assume that the spectrum of $\mathbf{A}$ lies on the interval $\mathcal{I} = [\alpha - \delta, \alpha + \delta]$ of the positive real axis, and we consider the Chebyshev iteration for that interval. We denote its nth error vector and residual polynomial by $\mathbf{d}_n^{\text{Cheb}}$ and p_n^{Cheb}, respectively, while $\mathbf{d}_n$ is assumed to be the error for a Krylov space solver that is optimal in the $\|.\|_{\mathbf{C}}$–norm of the error when applied to $\mathbf{A}\mathbf{x} = \mathbf{b}$ with a matrix whose spectrum lies on $\mathcal{I}$. Then we have

$$\frac{\|\mathbf{d}_n\|_{\mathbf{C}}}{\|\mathbf{d}_0\|_{\mathbf{C}}} \le \frac{\|\mathbf{d}_n^{\text{Cheb}}\|_{\mathbf{C}}}{\|\mathbf{d}_0\|_{\mathbf{C}}} \le \max_{i=1,\dots,N} |p_n^{\text{Cheb}}(\lambda_i)| \le \min_{\substack{p\in\mathcal{P}_n \\ p(0)=1}} \max_{\tau\in\mathcal{I}} |p(\tau)| \,. \tag{2.60}$$

Note that in this estimate the dependence of the residual polynomial on the initial residual and on the norm used is lost: we estimate the error of any optimal method by the error of the Chebyshev method, and in the estimate of the latter, the norm used has no effect on the bound for the error reduction.

In particular, the estimate is applicable for the CG method, which is optimal in the $\mathbf{A}$–norm of the error, and for the CR method, which is optimal in the $\mathbf{A}^2$–norm of the error, that is, in the 2–norm of the residual.

The bound on the right-hand side of (2.60) was estimated in the derivation of Theorem 1.6.2, and we can conclude that the bounds obtained in that theorem hold here too.

Theorem 2.6.1 *The residual norm reduction of the* CR *method and the energy norm reduction of the* CG *method, when applied to an spd system whose condition number is bounded by $\kappa_{\mathcal{I}}$, have the bounds given in* (1.78) *for the residual norm reduction of the Chebyshev iteration.*

2.7 Preconditioned CG Algorithms

The conjugate gradient method is a typical example where preconditioning must be done so that the symmetry of $\mathbf{A}$ is not lost. So it seems to be a case for split preconditioning with $\mathbf{C}_L = \mathbf{C}_R^{\mathsf{T}}$ or $\mathbf{M}_L = \mathbf{M}_R^{\mathsf{T}}$, respectively. In principle, we could then apply any CG algorithm with $\mathbf{A}$, $\mathbf{b}$, and $\mathbf{x}$ replaced by $\widehat{\mathbf{A}}$, $\widehat{\mathbf{b}}$, and $\widehat{\mathbf{x}}$ to get a **preconditioned CG algorithm**, or, as often referred to, a **PCG algorithm**. However, we will see that there are more refined ways to incorporate the preconditioning into the algorithms. There are even ways to use one-sided preconditioning destroying the symmetry of $\mathbf{A}$ if we combine it with a variation of CG using an alternative inner product.

We start with the details of the integration of symmetrically split preconditioning. For simplicity, we write the preconditioner as in (1.87): $\mathbf{M} = \mathbf{L}\mathbf{L}^{\mathsf{T}}$ or $\mathbf{C} = \mathbf{K}\mathbf{K}^{\mathsf{T}}$, even if $\mathbf{L}$ or $\mathbf{K}$, respectively, is not lower triangular. Although the former notation is more common, we prefer the latter, because we can avoid the usage of $\mathbf{L}^{-1}$ in the formulation of the preconditioned algorithm. If $\mathbf{L}$ is known instead of $\mathbf{K}$, we will just have to solve a linear system — assumed to be easy due to the structure of $\mathbf{L}$ — whenever a matrix-vector product with $\mathbf{K}$ or $\mathbf{K}^{\mathsf{T}}$ comes up.

If $\mathbf{A}$ is spd and $\mathbf{K}$ is nonsingular, then $\widehat{\mathbf{A}} :\equiv \mathbf{K}\mathbf{A}\mathbf{K}^{\mathsf{T}}$ is spd too, so we can replace the spd system $\mathbf{A}\mathbf{x} = \mathbf{b}$ by the spd system

$$\boxed{\underbrace{\mathbf{K}\mathbf{A}\mathbf{K}^{\mathsf{T}}}_{\widehat{\mathbf{A}}}\,\widehat{\mathbf{x}} = \underbrace{\mathbf{K}\mathbf{b}}_{\widehat{\mathbf{b}}}\,, \qquad \mathbf{x} = \mathbf{K}^{\mathsf{T}}\widehat{\mathbf{x}}\,.} \tag{2.61}$$

In the following OMin form of CG with split preconditioning $\widehat{\mathbf{x}}_n$ does not appear. The iterates $\mathbf{x}_n$ are computed directly.

Algorithm 2.5 (PCG–OMin with split preconditioning) *For solving* $\mathbf{A}\mathbf{x} = \mathbf{b}$ *with spd* $\mathbf{A}$ *in the split preconditioned form* (2.61) *choose an initial approximation* $\mathbf{x}_0$, *and let* $\mathbf{r}_0 := \mathbf{b} - \mathbf{A}\mathbf{x}_0$, $\widehat{\mathbf{r}}_0 := \mathbf{K}\mathbf{r}_0$, $\mathbf{v}_0 := \mathbf{K}^{\mathsf{T}}\widehat{\mathbf{r}}_0$, *and* $\delta_0 := \|\widehat{\mathbf{r}}_0\|^2$.
Then, for $n = 0, 1, 2, \ldots,$ *compute*

$$\delta'_n := \|\mathbf{v}_n\|^2_{\mathbf{A}}\,, \tag{2.62a}$$
$$\omega_n := \delta_n/\delta'_n\,, \tag{2.62b}$$
$$\mathbf{x}_{n+1} := \mathbf{x}_n + \mathbf{v}_n\omega_n\,, \tag{2.62c}$$
$$\widehat{\mathbf{r}}_{n+1} := \widehat{\mathbf{r}}_n - \mathbf{K}\mathbf{A}\mathbf{v}_n\omega_n\,, \tag{2.62d}$$
$$\delta_{n+1} := \|\widehat{\mathbf{r}}_{n+1}\|^2\,, \tag{2.62e}$$
$$\psi_n := -\delta_{n+1}/\delta_n\,, \tag{2.62f}$$
$$\mathbf{v}_{n+1} := \mathbf{K}^{\mathsf{T}}\widehat{\mathbf{r}}_{n+1} - \mathbf{v}_n\psi_n\,. \tag{2.62g}$$

If $\delta_{n+1} \le \mathrm{tol}$, *the algorithm terminates and* $\mathbf{x}_{n+1}$ *is a sufficiently accurate approximation of the solution.*

The algorithm makes use of the fact that for the preconditioned direction vectors $\widehat{\mathbf{v}}_n :\equiv \mathbf{K}^{-\mathsf{T}}\mathbf{v}_n$, which do not appear, we have

$$\|\widehat{\mathbf{v}}_n\|^2_{\widehat{\mathbf{A}}} = \left\langle \widehat{\mathbf{v}}_n, \widehat{\mathbf{A}}\widehat{\mathbf{v}}_n \right\rangle = \left\langle \mathbf{K}^{-\mathsf{T}}\mathbf{v}_n, \widehat{\mathbf{A}}\mathbf{K}^{-\mathsf{T}}\mathbf{v}_n \right\rangle = \langle \mathbf{v}_n, \mathbf{A}\mathbf{v}_n \rangle = \|\mathbf{v}_n\|^2_{\mathbf{A}} .$$

Let us now turn to a widely used alternative based on preconditioning $\mathbf{Ax} = \mathbf{b}$ on the left with the inverse $\mathbf{M}^{-1}$ of an spd matrix $\mathbf{M}$ that is in some sense an approximation of $\mathbf{A}$. So, we solve (as in (1.83), but in different notation),

$$\boxed{\underbrace{\mathbf{M}^{-1}\mathbf{A}}_{\widehat{\mathbf{A}}}\,\mathbf{x} = \underbrace{\mathbf{M}^{-1}\mathbf{b}}_{\widehat{\mathbf{b}}} .} \tag{2.63}$$

The CG method can be defined for any inner product space, in particular for $\mathbb{R}^N$ with the inner product induced by $\mathbf{M}$, the so-called **M–inner product** [**M**–*Skalarprodukt*],

$$\boxed{\langle \mathbf{x}, \mathbf{y} \rangle_{\mathbf{M}} :\equiv \langle \mathbf{x}, \mathbf{My} \rangle = \langle \mathbf{Mx}, \mathbf{y} \rangle \ .}$$

The matrix $\widehat{\mathbf{A}} :\equiv \mathbf{M}^{-1}\mathbf{A}$ in (2.63) is **M–self-adjoint** [**M**–*selbstadjungiert*], that is, it is self-adjoint (or, symmetric) with respect to the **M**–inner product:

$$\left\langle \mathbf{M}^{-1}\mathbf{Ax}, \mathbf{y} \right\rangle_{\mathbf{M}} = \langle \mathbf{Ax}, \mathbf{y} \rangle = \langle \mathbf{x}, \mathbf{Ay} \rangle = \left\langle \mathbf{x}, \mathbf{M}^{-1}\mathbf{Ay} \right\rangle_{\mathbf{M}} . \tag{2.64}$$

Noting that

$$\langle \widehat{\mathbf{r}}_n, \widehat{\mathbf{r}}_n \rangle_{\mathbf{M}} = \langle \mathbf{r}_n, \widehat{\mathbf{r}}_n \rangle \ , \qquad \left\langle \mathbf{v}_n, \mathbf{M}^{-1}\mathbf{A}\mathbf{v}_n \right\rangle_{\mathbf{M}} = \langle \mathbf{v}_n, \mathbf{A}\mathbf{v}_n \rangle$$

we obtain the following version of a PCG algorithm suggested by Meijerink and van der Vorst (1977):

Algorithm 2.6 (PCG–OMIN WITH M–INNER PRODUCT) *For solving* $\mathbf{Ax} = \mathbf{b}$ *with spd* $\mathbf{A}$ *and an spd left preconditioner* $\mathbf{M}$ *as in* (2.63) *choose an initial approximation* $\mathbf{x}_0$*, and let* $\mathbf{r}_0 := \mathbf{b} - \mathbf{Ax}_0$*,* $\mathbf{v}_0 := \widehat{\mathbf{r}}_0 := \mathbf{M}^{-1}\mathbf{r}_0$*, and* $\delta_0 := \langle \mathbf{r}_0, \widehat{\mathbf{r}}_0 \rangle$*.*
Then, for $n = 0, 1, 2, \dots,$ *compute*

$$\delta'_n := \|\mathbf{v}_n\|^2_{\mathbf{A}} , \tag{2.65a}$$
$$\omega_n := \delta_n / \delta'_n , \tag{2.65b}$$
$$\mathbf{x}_{n+1} := \mathbf{x}_n + \mathbf{v}_n\omega_n , \tag{2.65c}$$
$$\mathbf{r}_{n+1} := \mathbf{r}_n - \mathbf{A}\mathbf{v}_n\omega_n , \tag{2.65d}$$
$$\widehat{\mathbf{r}}_{n+1} := \mathbf{M}^{-1}\mathbf{r}_{n+1} , \tag{2.65e}$$
$$\delta_{n+1} := \langle \mathbf{r}_{n+1}, \widehat{\mathbf{r}}_{n+1} \rangle , \tag{2.65f}$$
$$\psi_n := -\delta_{n+1}/\delta_n , \tag{2.65g}$$
$$\mathbf{v}_{n+1} := \widehat{\mathbf{r}}_{n+1} - \mathbf{v}_n\psi_n . \tag{2.65h}$$

If $\delta_{n+1} \le \mathrm{tol}$ *(or, if* $\|\mathbf{r}_{n+1}\|^2 \le \mathrm{tol}$*), the algorithm terminates and* $\mathbf{x}_{n+1}$ *is a sufficiently accurate approximation of the solution.*

2.8 CG and CR for Complex Systems

The CG and CR methods can also be applied to linear systems $\mathbf{Ax} = \mathbf{b}$ with complex data $\mathbf{A}$ and $\mathbf{b}$, but again, for CG the matrix must be Hermitian positive definite (Hpd), and for CR it must be Hermitian. Of course, in the formulas $\mathbf{A}^\mathsf{T}$ has to be replaced by $\mathbf{A}^\mathsf{H}$, and $\langle \mathbf{x}, \mathbf{y} \rangle :\equiv \mathbf{x}^\mathsf{T}\mathbf{y}$ becomes[2] $\langle \mathbf{x}, \mathbf{y} \rangle :\equiv \mathbf{x}^\mathsf{H}\mathbf{y}$.

Some adaptations are necessary in the treatment of error functional minimization: In particular, in (2.1) $\mathbf{b}^\mathsf{T}\mathbf{x}$ is replaced by $\mathsf{Re}\,(\mathbf{b}^\mathsf{H}\mathbf{x}) = \mathsf{Re}\,\langle \mathbf{b}, \mathbf{x} \rangle$, and in (2.52) we have $\mathsf{Re}\,(\mathbf{b}^\mathsf{H}\mathbf{Ax}) = \mathsf{Re}\,\langle \mathbf{b}, \mathbf{Ax} \rangle$.

With these changes, virtually everything remains correct.

In the following we normally allow that the data are complex, although in nearly all applications they are real. Recall that we let $\mathbb{E} :\equiv \mathbb{R}$ or $\mathbb{C}$ to combine the real and the complex cases, and that we denote the adjoint of $\mathbf{A}$ by $\mathbf{A}^\star$ — so this is the transpose $\mathbf{A}^\mathsf{T}$ if $\mathbf{A}$ is real, and the Hermitian transpose $\mathbf{A}^\mathsf{H}$ if $\mathbf{A}$ is complex.

2.9 CG for Least Squares Problems

When parameters of a linear model are determined by measurements, the influence of measurement errors can be reduced by making many more measurements than the number of parameters. This leads to an **overdetermined linear system $\mathbf{Ax} \approx \mathbf{b}$** of M equations in N unknowns, where $M > N$.

The classical approach to this problem, introduced by Gauss, is to determine the approximate solution for which the (Euclidean) 2-norm of the residual is minimized, or, what amounts to the same, where the square of this 2-norm is minimized:

$$\boxed{\|\mathbf{r}\|^2 = \|\mathbf{b} - \mathbf{Ax}\|^2 = \min!} \tag{2.66}$$

Since $\|\mathbf{r}\|^2 = \sum r_i^2$, Gauss called this approximation the **least squares solution**. The geometric properties of the Euclidean space imply that the residual of the least squares solution is orthogonal to the column space of $\mathbf{A}$, so $\mathbf{A}^\star\mathbf{r} = \mathbf{o}$ or

$$\boxed{\mathbf{A}^\star\mathbf{Ax} = \mathbf{A}^\star\mathbf{b}\,.} \tag{2.67}$$

These are the **normal equations**. We assume that $\mathbf{A}$ has full column rank (*i.e.*, $\mathsf{rank}\,\mathbf{A} = N$). Then $\mathbf{A}^\star\mathbf{A}$ is Hpd and the CG method can be applied to the normal equations (2.67). This approach is known as the CGNR **method**; the "N" and "R" refer to the normal equations and the residual, respectively. It means that the approximate solutions are chosen such that

$$\boxed{\mathbf{x}_n - \mathbf{x}_0 \in \mathcal{K}_n(\mathbf{A}^\star\mathbf{A}, \mathbf{A}^\star\mathbf{r}_0)\,,} \quad \text{where} \quad \mathbf{r}_0 :\equiv \mathbf{b} - \mathbf{Ax}_0\,. \tag{2.68}$$

[2]Attention: many mathematicians define $\langle \mathbf{x}, \mathbf{y} \rangle :\equiv \mathbf{y}^\mathsf{H}\mathbf{x}$ instead.

Hence, with $\mathbf{B} :\equiv \mathbf{A}\mathbf{A}^\star$, the residuals $\mathbf{r}_n :\equiv \mathbf{b} - \mathbf{A}\mathbf{x}_n$ satisfy

$$\boxed{\mathbf{r}_n - \mathbf{r}_0 \in \mathbf{A}\,\mathcal{K}_n(\mathbf{A}^\star\mathbf{A}, \mathbf{A}^\star\mathbf{r}_0) = \mathbf{A}\mathbf{A}^\star\,\mathcal{K}_n(\mathbf{A}\mathbf{A}^\star, \mathbf{r}_0) = \mathbf{B}\mathcal{K}_n(\mathbf{B}, \mathbf{r}_0)\,.} \tag{2.69}$$

Applying CG to the normal equations means that the residuals $\mathbf{s}_n :\equiv \mathbf{A}^\star\mathbf{r}_n$ of the normal equations are orthogonal, so they satisfy

$$\boxed{\mathbf{s}_n - \mathbf{s}_0 \in \mathbf{A}^\star\mathbf{A}\,\mathcal{K}_n(\mathbf{A}^\star\mathbf{A}, \mathbf{s}_0)\,, \qquad \mathbf{s}_n \perp \mathcal{K}_n(\mathbf{A}^\star\mathbf{A}, \mathbf{s}_0)\,.} \tag{2.70}$$

It follows that for the residuals $\mathbf{r}_n = \mathbf{b} - \mathbf{A}\mathbf{x}_n$ of the given system holds the Galerkin condition

$$\boxed{\mathbf{r}_n \perp \mathbf{A}\mathcal{K}_n(\mathbf{A}^\star\mathbf{A}, \mathbf{A}^\star\mathbf{r}_0) = \mathbf{B}\mathcal{K}_n(\mathbf{B}, \mathbf{r}_0)\,.} \tag{2.71}$$

Let us summarize some of these results:

THEOREM 2.9.1 *Let a full rank overdetermined rectangular system* $\mathbf{A}\mathbf{x} = \mathbf{b}$ *be given; so* $\mathbf{A}$ *is* $M \times N$, $\mathsf{rank}\,\mathbf{A} = N \leq M$. *Its least squares solution minimizing* (2.66) *is the unique solution of the normal equations* (2.67).
The CGNR *method consisting of applying the* CG *method to these normal equations has the following properties:*

- *The iterates* $\mathbf{x}_n$ *minimize the 2-norm of the residuals* $\mathbf{r}_n :\equiv \mathbf{b} - \mathbf{A}\mathbf{x}_n$ *of the original system subject to the condition* (2.68), *and in this sense they are least squares solution of the given system restricted further by* (2.68).
- *The residuals* $\mathbf{r}_n$ *satisfy* (2.69) *and* (2.71). *For the residuals* $\mathbf{s}_n$ *of the normal equations holds* (2.70).

On the other hand, if the given system $\mathbf{A}\mathbf{x} = \mathbf{b}$ is an **underdetermined linear system** of M equations in N unknowns, where $M < N$, then we can restrict $\mathbf{x}$ to the image of $\mathbf{A}^\star$ in order to define a unique solution if $\mathbf{A}$ has full rank M, which we assume again. We define $\mathbf{z}$ by

$$\mathbf{x} \equiv: \mathbf{A}^\star\mathbf{z} \tag{2.72}$$

and write the given system as

$$\boxed{\mathbf{A}\mathbf{A}^\star\mathbf{z} = \mathbf{b}\,,} \tag{2.73}$$

which is again referred to as **normal equations**. We claim that (2.72) restricts $\mathbf{x}$ to be orthogonal to the nullspace (or, kernel) of $\mathbf{A}$. In fact, we can write any $\mathbf{x}' \in \mathbb{E}^N$ as

$$\mathbf{x}' = \mathbf{x}^\perp + \mathbf{A}^\star\widetilde{\mathbf{z}}\,, \qquad \text{where} \quad \mathbf{x}^\perp \perp \mathsf{im}\,\mathbf{A}^\star\,, \quad \widetilde{\mathbf{z}} \in \mathbb{E}^M\,. \tag{2.74}$$

The latter means that $\langle \mathbf{x}^\perp, \mathbf{A}^\star\mathbf{w}\rangle = 0$ $(\forall \mathbf{w} \in \mathbb{E}^M)$ or $\mathbf{A}\mathbf{x}^\perp \perp \mathbb{E}^M$, which implies that $\mathbf{A}\mathbf{x}^\perp = \mathbf{o}$. In other words, $\mathbf{x}^\perp$ lies in the kernel of $\mathbf{A}$. Actually, it is a well-known fact from matrix theory that the kernel (or null space) of $\mathbf{A}$ and the image (or range) of $\mathbf{A}^\star$ are orthogonal complementary subspaces. We conclude that $\mathbf{A}\mathbf{x}' = \mathbf{A}\mathbf{A}^\star\widetilde{\mathbf{z}}$. So any solution of the given system $\mathbf{A}\mathbf{x}' = \mathbf{b}$ has the form (2.74) with $\widetilde{\mathbf{z}} = \mathbf{z}$ being a solution of (2.73) and $\mathbf{x}^\perp \in \ker \mathbf{A}$. From Pythagoras' theorem we can conclude further that

$$\|\mathbf{x}'\|^2 = \|\mathbf{x}^\perp\|^2 + \|\mathbf{A}^\star\mathbf{z}\|^2 .$$

Hence, $\mathbf{x} = \mathbf{A}^\star\mathbf{z}$ is the shortest solution. In other words, if $\mathbf{A}$ has full rank M $(< N)$, then under the substitution $\mathbf{x} = \mathbf{A}^\star\mathbf{z}$ the system (2.73) is equivalent with

$$\boxed{\mathbf{A}\mathbf{x} = \mathbf{b}\,, \qquad \|\mathbf{x}\| = \min!} \tag{2.75}$$

Since $\mathbf{A}\mathbf{A}^\star$ is Hpd, we can apply CG to the normal equations (2.73). This is the CGNE **method**, where "E" now refers to the error. It is also called **Craig's method**. If $\mathbf{z}_n$ is any approximate solution, the corresponding residual is

$$\|\mathbf{r}_n\| = \|\mathbf{b} - \mathbf{A}\mathbf{A}^\star\mathbf{z}_n\| = \|\mathbf{b} - \mathbf{A}\mathbf{x}_n\| \tag{2.76}$$

and is also the residual of the original system. So, implicitly, CGNE yields approximate solutions $\mathbf{z}_n$ of the normal equations (2.73) with

$$\boxed{\mathbf{z}_n - \mathbf{z}_0 \in \mathcal{K}_n(\mathbf{A}\mathbf{A}^\star, \mathbf{r}_0) = \mathcal{K}_n(\mathbf{B}, \mathbf{r}_0)\,,} \tag{2.77}$$

corresponding approximate solutions of the original system with

$$\boxed{\mathbf{x}_n - \mathbf{x}_0 \in \mathbf{A}^\star\mathcal{K}_n(\mathbf{A}\mathbf{A}^\star, \mathbf{r}_0) = \mathcal{K}_n(\mathbf{A}^\star\mathbf{A}, \mathbf{A}^\star\mathbf{r}_0)\,,} \tag{2.78}$$

as well as residuals that satisfy again (2.69) and, as can be seen,

$$\boxed{\mathbf{r}_n \perp \mathcal{K}_n(\mathbf{A}\mathbf{A}^\star, \mathbf{r}_0) = \mathcal{K}_n(\mathbf{B}, \mathbf{r}_0)\,.} \tag{2.79}$$

Consequently, the relevant Krylov space is the same as for CGNR, but the Galerkin condition is different.

In the nth step, the $\mathbf{A}\mathbf{A}^\star$-norm or $\mathbf{B}$-norm of the error $\mathbf{z}_n - (\mathbf{A}\mathbf{A}^\star)^{-1}\mathbf{b}$ of (2.73) is minimized subject to the condition (2.77). This is the same as minimizing the 2-norm of the error $\mathbf{x}_n - \mathbf{x}_\star$ of the original system subject to the condition (2.78).

THEOREM 2.9.2 *Let a full rank underdetermined rectangular system* $\mathbf{Ax} = \mathbf{b}$ *be given; so* $\mathbf{A}$ *is* $M \times N$, $\mathsf{rank}\,\mathbf{A} = M < N$. *Its least squares solution minimizing* (2.75) *is the unique solution of the normal equations* (2.73). *Any other solution is of the form* $\mathbf{x} = \mathbf{x}^{\perp} + \mathbf{A}^{\star}\mathbf{z}$, *where* $\mathbf{x}^{\perp} \perp \mathsf{im}\,\mathbf{A}^{\star}$, *which is equivalent to* $\mathbf{x}^{\perp} \in \mathsf{ker}\,\mathbf{A}$.
The CGNE *algorithm consisting of applying the* CG *method to these normal equations has the following properties:*

- $\mathbf{z}_n$ *minimizes the* $\mathbf{B}$*-norm of the error of the normal equations* (2.73) *subject to* (2.77), *where* $\mathbf{B} :\equiv \mathbf{AA}^{\star}$.
- *The corresponding original approximations* $\mathbf{x}_n = \mathbf{A}^{\star}\mathbf{z}_n$ *minimize the 2-norm of the errors* $\mathbf{x}_n - \mathbf{x}_{\star}$ *of the original system subject to the condition* (2.78).
- *The residuals* $\mathbf{r}_n$, *which are both the residuals of the original system and the normal equations, satisfy* (2.69) (*as in* CGNR) *and the Galerkin condition* (2.79).

The theorem remains correct for $M = N$ if we let $\mathbf{x}^{\perp} := \mathbf{o}$. The condition $\|\mathbf{x}\| = \min!$ in (2.75) can be dropped because $\mathbf{Ax} = \mathbf{b}$ has then a unique solution.

Let us now look at the normally used OMIN algorithms of the CGNR and CGNE methods. We have to adapt Algorithm 2.1 of Section 2.4 so that it is applied to the normal equations (2.67) and (2.73), respectively.

For CGNR the relevant residuals that are orthogonal to each other are the residuals $\mathbf{s}_n (= \mathbf{A}^{\star}\mathbf{r}_0)$ of the normal equations that satisfy (2.70). They also determine δ_n. The search directions $\mathbf{v}_n$ are now $\mathbf{A}^{\star}\mathbf{A}$-orthogonal, and thus $\delta'_n := \|\mathbf{v}_n\|^2_{\mathbf{A}^{\star}\mathbf{A}} = \|\mathbf{A}\mathbf{v}_n\|^2$.

Algorithm 2.7 (OMIN FORM OF THE CGNR METHOD) .
For solving the least squares problem (2.66) *via the normal equations* (2.67) *choose an initial approximation* $\mathbf{x}_0$, *and let* $\mathbf{r}_0 := \mathbf{b} - \mathbf{A}\mathbf{x}_0$, $\mathbf{v}_0 := \mathbf{s}_0 := \mathbf{A}^{\star}\mathbf{r}_0$, $\delta_0 := \|\mathbf{s}_0\|^2$, *and* $\psi_{-1} := 0$. *Then, for* $n = 0, 1, 2, \ldots,$ *compute*

$$\delta'_n := \|\mathbf{A}\mathbf{v}_n\|^2 , \tag{2.80a}$$
$$\omega_n := \delta_n / \delta'_n , \tag{2.80b}$$
$$\mathbf{x}_{n+1} := \mathbf{x}_n + \mathbf{v}_n \omega_n , \tag{2.80c}$$
$$\mathbf{r}_{n+1} := \mathbf{r}_n - \mathbf{A}\mathbf{v}_n \omega_n , \tag{2.80d}$$
$$\mathbf{s}_{n+1} := \mathbf{A}^{\star}\mathbf{r}_{n+1} , \tag{2.80e}$$
$$\delta_{n+1} := \|\mathbf{s}_{n+1}\|^2 , \tag{2.80f}$$
$$\psi_n := -\delta_{n+1}/\delta_n , \tag{2.80g}$$
$$\mathbf{v}_{n+1} := \mathbf{s}_{n+1} - \mathbf{v}_n \psi_n . \tag{2.80h}$$

If $\|\mathbf{r}_{n+1}\| \leq \mathrm{tol}$, *the algorithm terminates and* $\mathbf{x}_{n+1}$ *is a sufficiently accurate approximation of the solution.*

We have chosen to update in (2.80d) also $\mathbf{r}_n$, which allows us to rely on $\|\mathbf{r}_{n+1}\|$ for termination. One could instead update directly $\mathbf{s}_n$ according to

$$\mathbf{s}_{n+1} := \mathbf{s}_n - \mathbf{A}^\star \mathbf{A} \mathbf{v}_n \omega_n \,, \tag{2.81}$$

but then termination must rely on $\|\mathbf{s}_{n+1}\|$.

For CGNE the residuals $\mathbf{r}_n$ are the same for the original and the normal equations (2.73), but the iterates seem to change to $\mathbf{z}_n$. However, it turns out that instead everything can be formulated in terms of the approximate solutions $\mathbf{x}_n = \mathbf{A}^\star \mathbf{z}_n$ of the original system and the corresponding search directions.

Algorithm 2.8 (OMIN FORM OF THE CGNE METHOD) .
For solving the minimum solution least squares problem (2.75) *via the normal equations* (2.73) *choose an initial approximation* $\mathbf{x}_0$, *and let* $\mathbf{v}_0 := \mathbf{r}_0 := \mathbf{b} - \mathbf{A}\mathbf{x}_0$, $\delta_0 := \|\mathbf{r}_0\|^2$, *and* $\psi_{-1} := 0$.
Then, for $n = 0, 1, 2, \ldots$, *compute*

$$\delta'_n := \|\mathbf{v}_n\|^2 \,, \tag{2.82a}$$
$$\omega_n := \delta_n / \delta'_n \,, \tag{2.82b}$$
$$\mathbf{x}_{n+1} := \mathbf{x}_n + \mathbf{v}_n \omega_n \,, \tag{2.82c}$$
$$\mathbf{r}_{n+1} := \mathbf{r}_n - \mathbf{A}\mathbf{v}_n \omega_n \,, \tag{2.82d}$$
$$\delta_{n+1} := \|\mathbf{r}_{n+1}\|^2 \,, \tag{2.82e}$$
$$\psi_n := -\delta_{n+1} / \delta_n \,, \tag{2.82f}$$
$$\mathbf{v}_{n+1} := \mathbf{A}^\star \mathbf{r}_{n+1} - \mathbf{v}_n \psi_n \,. \tag{2.82g}$$

If $\|\mathbf{r}_{n+1}\| \le \mathrm{tol}$, *the algorithm terminates and* $\mathbf{x}_{n+1}$ *is a sufficiently accurate approximation of the solution.*

A potential disadvantage of using the normal equations (2.67) and (2.73) is that the condition number of $\mathbf{A}^\star\mathbf{A}$ and $\mathbf{A}\mathbf{A}^\star$ may be large compared to the one of the triangular matrix $\mathbf{R}$ in a QR decomposition of $\mathbf{A}$. In particular, if $\mathbf{A}$ is square and nonsingular,

$$\kappa_2(\mathbf{A}^\star\mathbf{A}) = \kappa_2(\mathbf{A}\mathbf{A}^\star) = (\kappa_2(\mathbf{A}))^2 = (\kappa_2(\mathbf{R}))^2 \,.$$

Bibliography

W. E. Arnoldi (1951), 'The principle of minimized iterations in the solution of the matrix eigenvalue problem', *Quart. Appl. Math.* **9**, 17–29.

S. F. Ashby, T. A. Manteuffel and P. E. Saylor (1990), 'A taxonomy for conjugate gradient methods', *SIAM J. Numer. Anal.* **27**, 1542–1568.

J. K. Cullum and R. A. Willoughby (1985), *Lanczos Algorithms for Large Symmetric Eigenvalue Computations (2 Vols.)*, Birkhäuser, Boston-Basel-Stuttgart.

J. Dongarra and F. Sullivan (2000), 'Guest editors' introduction to the top 10 algorithms', *Computing in Science and Engineering* **2**(1), 22–23.

B. Fischer and R. W. Freund (1990), 'On the constrained Chebyshev approximation problem on ellipses', *Journal of Approximation Theory* **62**, 297–315.

B. Fischer and R. W. Freund (1991), 'Chebyshev polynomials are not always optimal', *Journal of Approximation Theory* **65**, 261–272.

R. Fletcher (1976), Conjugate gradient methods for indefinite systems, in *Numerical Analysis, Dundee, 1975* (G. A. Watson, ed.), Vol. 506 of *Lecture Notes in Mathematics*, Springer, Berlin, pp. 73–89.

R. W. Freund and N. M. Nachtigal (1991), 'QMR: a quasi-minimal residual method for non-Hermitian linear systems', *Numer. Math.* **60**, 315–339. Received Feb. 19, 1991.

G. H. Golub and R. S. Varga (1961), 'Chebyshev semiiterative methods, successive overrelaxation iterative methods, and second order Richardson iterative methods', *Numer. Math.* **3**, 147–168.

A. Greenbaum (1997), *Iterative Methods for Solving Linear Systems*, SIAM, Philadelphia, PA.

P. Henrici (1974), *Applied and Computational Complex Analysis, Vol. 1*, Wiley, New York.

M. R. Hestenes and E. Stiefel (1952), 'Methods of conjugate gradients for solving linear systems', *J. Res. Nat. Bureau Standards* **49**, 409–435.

R. A. Horn and C. R. Johnson (1985), *Matrix Analysis*, Cambridge University Press, New York.

C. Lanczos (1950), ‘An iteration method for the solution of the eigenvalue problem of linear differential and integral operators’, *J. Res. Nat. Bureau Standards* **45**, 255–281.

C. Lanczos (1952), ‘Solution of systems of linear equations by minimized iterations’, *J. Res. Nat. Bureau Standards* **49**, 33–53.

J. A. Meijerink and H. A. van der Vorst (1977), ‘An iterative solution method for linear equations systems of which the coefficient matrix is a symmetric M-matrix’, *Math. Comp.* **31**, 148–162.

C. C. Paige (1971), The computations of eigenvalues and eigenvectors of very large sparse matrices, PhD thesis, University of London.

C. C. Paige and M. A. Saunders (1975), ‘Solution of sparse indefinite systems of linear equations’, *SIAM J. Numer. Anal.* **12**, 617–629.

B. N. Parlett (1980), *The Symmetric Eigenvalue Problem*, Prentice-Hall, Englewood Cliffs, N.J.

H. Rutishauser (1957), *Der Quotienten-Differenzen-Algorithmus*, Mitt. Inst. angew. Math. ETH, Nr. 7, Birkhäuser, Basel.

H. Rutishauser (1959), Theory of gradient methods, in *Refined Iterative Methods for Computation of the Solution and the Eigenvalues of Self-Adjoint Boundary Value Problems*, Mitt. Inst. angew. Math. ETH Zürich, Nr. 8, Birkhäuser, Basel, pp. 24–49.

Y. Saad (1996), *Iterative Methods for Sparse Linear Systems*, PWS Publishing, Boston.

Y. Saad and M. H. Schultz (1985), ‘Conjugate gradient-like algorithms for solving nonsymmetric linear systems’, *Math. Comp.* **44**, 417–424.

O. Schenk (2000), Scalable Parallel Sparse LU Factorization Methods on Shared Memory Multiprocessors, PhD thesis, Diss. No. 13515, ETH Zurich, Zurich, Switzerland.

H. R. Schwarz, H. Rutishauser and E. Stiefel (1968), *Numerik symmetrischer Matrizen*, Teubner, Stuttgart.

G. L. G. Sleijpen, H. A. van der Vorst and J. Modersitzki (2000), ‘Differences in the effects of rounding errors in Krylov solvers for symmetric indefinite linear systems’, *SIAM J. Matrix Anal. Appl.* **22**(3), 726–751.

E. Stiefel (1955), ‘Relaxationsmethoden bester Strategie zur Lösung linearer Gleichungssysteme’, *Comm. Math. Helv.* **29**, 157–179.

G. Strang (1986), *Introduction to Applied Mathematics*, Wellesley–Cambridge Press, Wellesley, MA, U.S.A.

H. A. van der Vorst (2000), ‘Krylov subspace iteration’, *Computing in Science and Engineering* **2**, 32–37.

R. S. Varga (1962), *Matrix Iterative Analysis*, Prentice-Hall, Englewood Cliffs, N.J. Rev. 2nd ed., Springer-Verlag, 1999.

H. F. Walker and L. Zhou (1994), ‘A simpler GMRES’, *Numer. Linear Algebra Appl.* **1**(6), 571–581.

D. M. Young (1950), Iterative Methods for Solving Partial Difference Equations of Elliptic Type, PhD thesis, Harvard University. http://www.cs.utexas.edu/users/young/david_young_thesis.pdf.

D. M. Young (1971), *Iterative Solution of Large Linear Systems*, Academic Press, Orlando.

Index