Advanced Numerical Analysis

Lecture Notes for the Sommersemester 2023

by

Lukas **Exl**, Norbert J. **Mauser**, Hans Peter **Stimming**

Research platform MMM "Mathematics-Magnetism-Materials"
and
Fak. f. Mathematik, Univ. Wien

This course is designed for the Master of Mathematics at Univ. Wien, it fits also for the Master Computational Sciences and the Master Data Science, as well as for master and PhD students in all MINT fields with a focus on numerics, like Computational Physics, Computational Astrophysics / Geology / Meteorology, Informatics, etc.
The prerequisite for this course is a sound understanding of analyis and linear algebra, and a basic numerics course containing standard subjects like numerical linear algebra etc. Good students can learn that on the fly in this course using e.g. the introductory lecture notes provided by the teachers.
The exercise classes in parallel to the lecture are strongly recommended since understanding numerical methods is hard without practical application.

1 Iterative methods for large linear systems

Iterative methods try to find an approximate solution $x \in \mathbb{R}^n$ to the linear equation $Ax = b$, where $A \in \mathbb{R}^{n\times n}$ is typically a *sparse matrix* and n a very large integer. In these cases, so-called iterative methods might be preferred over direct methods (which are based on matrix factorization) because their iterations mainly rely on efficient (sparse) matrix-vector evaluation. Large and sparse matrices appear frequently in the applied sciences e.g. as finite difference or stiffness matrices in discretizations of partial differential equations (PDEs).

An extensive discourse of the topic is given in the book of Y. Saad [1]. We will cover the most important aspects of this topic including the most insightful proofs.

With increasing computational power larger numerical problems can be tackled. Many of these problems ultimately reduce to or rely on the task of solving (many) systems of linear equations of large sizes, e.g. $n > 10^7$. Direct solution of such large systems are almost never efficient or even intractable. However, if special structure or, specifically, sparsity (many matrix entries are zero) can be imposed, iteration methods can tackle such tasks. An easy model example for the occurrence of large and sparse linear systems is the finite difference or finite element discretization of PDEs such as the Poisson equation. We will show these examples in the next introductory section of this chapter. They already involve important features like sparsity but also an typically ill-conditioned system matrix. Also many other (spatial) discretizations of continuous problems in mathematical sciences ultimately reduce to subproblems which are large linear systems. Often linear systems occur as local approximations to nonlinear problems (like in nonlinear equations, optimization or differential equations), where linearizations often yield approximate subproblems whose successive solution yields an approximation to the original problem. The key subproblem is often a large (sparse) linear system which occurs each iteration, and hence, has to be solved many times for the approximate solution of the main problem. It is therefore very important to be able to efficiently solve the linear systems.
Depending on features of the system matrix A different methods and versions of methods yield appropriate efficiency. Such features can be the sparsity pattern, condition number, matrix properties like symmetry or spectrum (e.g. positive eigenvalues). An important consideration in a concrete application case might also be the desired error of the approximate solution and its definition or measurement, e.g. via residual (minimum residual methods) or absolute/relative error.
However, the unprepared use of a certain iterative method for solving linear systems might not be effective or even be practically impossible because of a too large condition number of the system matrix. Therefore, so-called preconditioning has to be considered. We will introduce basic techniques for preconditioning linear systems in the end of this chapter.

1.1 Introductory examples

We briefly show the occurrence of large sparse linear systems in a model problem, which here shall be the Poisson problem in two dimensions with Dirichlet boundary conditions on $\Omega = (0,1)^2$. This is the following equation for the scalar function $u = u(x,y)$ induced by the density $f = f(x,y)$:

$$\begin{aligned} -\Delta u &= f, \\ u_{|\partial\Omega} &= 0. \end{aligned} \tag{1}$$

A finite difference method would try to solve the above differential equation on an equidistant grid defined by the grid points (nodes) $x_{ij} = (x_i, y_j) = (ih, jh),\ i,j = 0\dots n+1$ with the grid spacing $h = 1/(n+1)$. The Laplace operator $\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$ is approximated by a (second order) finite difference formula. Incorporating the given values on the boundary (zeros) this yields the approximations u_{ij} to the values $u(x_i, x_j)$ according to the solution of

$$4\,u_{ij} - u_{i-1,j} - u_{i+1,j} - u_{i,j-1} - u_{i,j+1} = h^2\, f_{ij} \quad i,j = 1\dots n, \tag{2}$$

where $f_{ij} = f(x_i, x_j)$.
The equations (2) can be written in matrix form $A\widetilde{u} = b$ by (e.g.) 'lexicographic' re-ordering of the

involved variables: make 'long vectors' by setting $\widetilde{u}_{i+(j-1)n} = u_{ij}$ and $b_{i+(j-1)n} = h^2 f_{ij}$. The matrix $A \in \mathbb{R}^{N\times N}$ with $N = n^2$ (K_{2d} matrix) is sparse (see Figure 1) and can be generated by so-called Kronecker products $A = K_{2d} := K_{1d} \otimes I + I \otimes K_{1d} \in \mathbb{R}^{N\times N}$, where $K_{1d} \in \mathbb{R}^{n\times n}$ is the tridiagonal matrix arising from the central difference quotient, i.e., $K_{1d} = \mathrm{tridiag}(-1, 2, -1)$ and $I \in \mathbb{R}^{n\times n}$ the identity (diag(1)). The matrix A has 4's in the diagonal, -1's in off diagonals and is sparse (many zeros). It is also symmetric and positive definite (positive eigenvalues), commonly abbreviated by SPD. The eigenvalues satisfy $\lambda_{min} = 8\,\sin^2(\frac{\pi}{2}h) = O(h^2)$ and $\lambda_{max} = 8\cos^2(\frac{\pi}{2}\,h) = O(1)$ and hence, the condition number is $\kappa_2 = O(h^{-2})$.

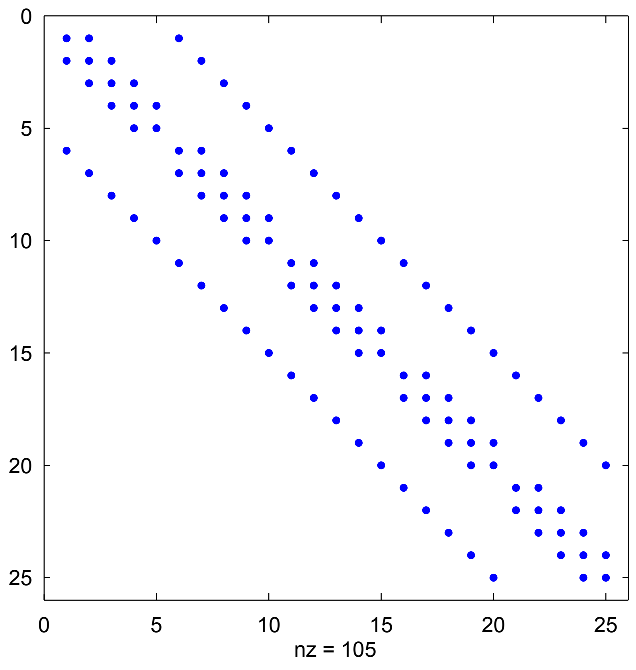


Figure 1: Sparsity pattern of K_{2d} matrix

It is a typical example of a matrix (arising from a (spatial) discretization) which gets increasingly ill-conditioned as the original continuous problem gets more and more resolved (h smaller). We remark that A is a so-called Toeplitz matrix, that is a (band) matrices with constant entries along the (off-) diagonal. These matrices have eigenvalues which can be expressed by trigonometric functions and systems which involve Toeplitz matrices can be efficiently solved by FFT methods. Anyhow, the special structure normally gets lost as soon as more general geometries (and/or boundary conditions) or variable coefficients in the differential equation are considered.

Sparse systems also arise in so-called finite element methods (FEM) which are often used e.g. in engineering to numerically compute problems formulated as differential equations on more general geometries. The arising sparsity of the systems is the key consideration of the choice of FE basis functions as opposed to the more global Ritz/Galerkin methods of which FEM is a special version. We briefly illustrate the so-called variational procedure of FEM by a simple example in one dimension. The key derivation is similar in higher dimensions and general geometries. Consider the boundary value problem

$$-u'' = f, \quad u(0) = u(1) = 0.$$

The variational scheme now consists of multiplying both sides of the equation with functions which incorporate the boundary conditions and followed by integration (by parts):

$$a(u,v) := \int_0^1 u'(x)v'(x)\,dx = \int_0^1 f(x)v(x)\,dx =: b(v)$$
$$v:\ v(0) = v(1) = 0$$

Now the solution is approximated by an ansatz in a finite dimensional space X_n, e.g. piecewise linear, $u^*(x) \approx \sum_{j=1}^n c_j\,\varphi_j(x)$ ($\varphi_j \in X_n$). Inserting the ansatz into the variational form and varying $v = \varphi_i$, $i = 1,\dots,n$ in X_n yields a linear system $Ac = b$ for the coefficients

$$\sum_{j=1}^n \underbrace{a(\varphi_i,\varphi_j)}_{=:a_{ij}\hookrightarrow A\in\mathbb{R}^{n\times n}} \underbrace{c_j}_{\hookrightarrow c\in\mathbb{R}^n} = \underbrace{b(\varphi_i)}_{=:b_i\hookrightarrow b\in\mathbb{R}^n}, \quad i = 1,\dots,n. \tag{3}$$

In the FEM the approximation and ansatz space X_n is chosen in a way such that the values of the bilinear form $a(\varphi_i,\varphi_j)$, and hence the matrix A, vanish most of the time. It is usually called the *stiffness matrix.*

1.2 Sparse formats

Here we consider a few of the straightforward formats for sparse matrix storage. The reason is twofold: (i) the sparse storage decreases memory requirements, (ii) the frequently occurring matrix-vector multiplication can be treated in an effective way.

1.2.1 Coordinate format

The coordinate format consists of three arrays (see figure 2):

- AA: an array of nonzero elements of the matrix A,
- IR: row indices array (integers) of nonzero elements,
- IC: column indices array (integers) of nonzero elements.

$$A = \begin{pmatrix} 2 & -1 & 0 & 0 & 0 \\ -1 & 4 & -2 & 0 & 0 \\ 0 & -2 & 4 & -3 & 0 \\ 0 & 0 & -3 & 6 & -2 \\ 0 & 0 & 0 & -2 & 4 \end{pmatrix}$$

$$\begin{array}{lll} \text{AA} & = & [2\ \ -1\ \ -1\ \ 4\ \ -2\ \ -2\ \ 4\ \ -3\ \ -3\ \ 6\ \ -2\ \ -2\ \ 4] \\ \text{IR} & = & [1\ \ 1\ \ 2\ \ 2\ \ 2\ \ 3\ \ 3\ \ 3\ \ 4\ \ 4\ \ 4\ \ 5\ \ 5] \\ \text{IC} & = & [1\ \ 2\ \ 1\ \ 2\ \ 3\ \ 2\ \ 3\ \ 4\ \ 3\ \ 4\ \ 5\ \ 4\ \ 5] \end{array}$$

Figure 2: Example for coordinate format taken from [3].

1.2.2 Compressed Sparse Row (CSR) and Column (CSC) formats

More compression than in the coordinate form is achieved by only storing integer pointers for the positions where a row (resp. column) starts, since the nonzeros can be arranged in the nonzero-array row-by-row (resp. column-by-column).
The CSR format consists of the following three arrays (CSC is analogue), compare with figure 3 :

- AA: an array of nonzero elements of the matrix $A \in \mathbb{R}^{n\times n}$,

$$A = \begin{pmatrix} 2 & -1 & 0 & 0 & 9 \\ -1 & 4 & -2 & 0 & 0 \\ 0 & -2 & 4 & -3 & 0 \\ 0 & 0 & -3 & 6 & -2 \\ 0 & 0 & 0 & -2 & 4 \end{pmatrix}$$

$$\begin{array}{lll} \text{AA} & = & [2 \;\; -1 \;\; 9 \;\; -1 \;\; 4 \;\; -2 \;\; -2 \;\; 4 \;\; -3 \;\; -3 \;\; 6 \;\; -2 \;\; -2 \;\; 4] \\ \text{PR} & = & [1 \quad\quad\quad 4 \quad\quad\quad 7 \quad\quad\quad 10 \quad\quad\quad 13 \quad\quad 15] \\ \text{IC} & = & [1 \;\; 2 \;\; 5 \;\; 1 \;\; 2 \;\; 3 \;\; 2 \;\; 3 \;\; 4 \;\; 3 \;\; 4 \;\; 5 \;\; 4 \;\; 5] \end{array}$$

Figure 3: Example for compressed sparse row (CSR) format taken from [3].

- IC: column indices array (integers) of nonzero elements,
- PR: array of $n+1$ integers. Entry $PR(i)$ points to positions in AA and IC where the i-th row of A starts. Nonzeros of i-th row are $AA(PR(i) : PR(i+1)-1)$ and their column indices are $IC(PR(i) : PR(i+1)-1)$. The last entry $PR(n+1) = nz+1$, where nz is the number of nonzeros.

A description of matrix-vector multiplication, which efficiently uses the CSR format is given in Alg. 1.

Algorithm 1: Matrix vector multiplication in the compressed sparse row format.

```
Data: CSR for $A \in \mathbb{R}^{n\times n}$: $AA, IC, PR$, vector $x \in \mathbb{R}^n$
Result: vector $y \in \mathbb{R}^n$
for $i = 1 : n$ do
    $k_1 = PR(i)$
    $k_2 = PR(i+1) - 1$
    $y(i) = AA(k_1 : k_2) * x(IC(k_1 : k_2))^T$
end
```

1.3 Basic iterative methods

In the following we will describe basic methods, which aim at solving a given linear system $Ax = b$ with invertible matrix $A \in \mathbb{R}^{n\times n}$ (can also be $\mathbb{C}$) iteratively. The matrices can be considered to be large, e.g. $n > 10^7$, which often occurs in applications. The iteration methods will take the following fixed point form:

$$x_{k+1} = \Phi(x_k, b) \quad \text{starting at some } x_0. \tag{4}$$

Iterative methods are designed to be faster than direct methods in the case that iterations are cheaply computable. This is the case if the involved matrix-vector products are cheap, since these are the main operations in iterative methods for linear systems.
In practice, one would first choose a certain available numerical method, ideally based on mathematical analysis or other understanding of the problem, and then prescribe an approximation accuracy, which should be reached for some error measure (stopping criterion). The 'converged solution' is only an approximation to the true solution of the nonsingular system. Practitioners still might be satisfied with this situation since direct solution (exact within computer accuracy) simply might be too expensive.
We emphasize that although the requirements for convergence might be perfectly fulfilled theoretically in a concrete problem case, the practical performance still depends on e.g. condition number, preconditioning of the problem, the chosen numerical method, starting point, stability of the algorithm and its computer realization, error measurement, size of the problem and other factors.
We will start with a very brief theoretical background on so-called linear fixed point iterations, which are the theoretical basis for the splitting methods described afterwards.

1.3.1 Linear fixed point iteration and convergence

We assume here a linear system

$$Ax = b, \tag{5}$$

with invertible matrix $A \in \mathbb{R}^{n\times n}$ or $\mathbb{C}^{n\times n}$. Many iterative methods for solving (5) have the form of a so-called *fixed point iteration*

$$x_{k+1} = \Phi(x_k, b), \tag{6}$$

for some function Φ (iteration mapping), see Alg. 2.

Algorithm 2: (Stationary) iterative methods

```
Data: Matrix $A \in \mathbb{R}^{n\times n}$ nonsing., r.h.s. $b \in \mathbb{R}^n$, initial vector $x_0 \in \mathbb{R}^n$
Result: $x \in \mathbb{R}^n$ approximate solution
Initialization: Set $x = x_0$
while Convergence criterion not satisfied do
    $x \leftarrow \Phi(x, b)$   with $\Phi$ according to chosen method
end
```

Definition 1.1 (fixed point iteration). *Solutions to the equation $x = \Phi(x, b)$ are called fixed points of* (6)*. The fixed point iteration is*

- ***consistent**, if for every b the solution $x_* = A^{-1}b$ is a fixed point,*
- ***(globally) convergent**, if for every b there is a x_* such that for every starting vector x_0 the sequence $(x_k)_{k\in\mathbb{N}}$ defined by the iteration* (6) *converges to x_*,*
- ***linear**, if the iteration mapping has the special form*

  $$\Phi(x, b) = Mx + Nb \tag{7}$$

  the matrix M is called iteration matrix,
- ***symmetric**, if it is linear, the matrix A symmetric and N is SPD.*

We will consider linear fixed point iterations in the following.

Theorem 1.2. *The linear fixed point iteration* (7) *is consistent if and only if there holds*

$$M = I - NA. \tag{8}$$

Proof. Exercise. □

The convergence of the linear fixed point iteration (7) depends on the iteration matrix M. This can be seen by the sufficient contraction condition for unique convergence of the *Banach fixed point theorem* which is $\|\Phi(x) - \Phi(y)\| \le q\|x - y\|$ for $q < 1$ for all $x, y \in \mathbb{R}^n$. In the present case we have $\|M(x-y)\| \le \|M\|\|x - y\| \le q\|x - y\|$, which shows that $\|M\| < 1$ guarantees convergence.
In fact this statement can be strengthened [1] to

Theorem 1.3. *Let $\rho(M) < 1$. Then there exists a unique fixed point x_* of* (6) *(with Φ from* (7)*) and the iteration* (6) *converges to x_* for any starting value x_0.*

Lemma 1.4. *Let $A \in \mathbb{K}^{n\times n}$, where $\mathbb{K} = \mathbb{R}$ or $\mathbb{C}$. Then, there holds:*

(i) $\rho(A^m) = \rho(A)^m$ for all $m \in \mathbb{N}$,

[1] $\rho(M) \le \|M\|$, only for normal M we have $\rho(M) = \|M\|_2$.

(ii) For any norm $\|.\|$ on $\mathbb{K}^n$ we have $\rho(A) \leq \|A\|$,

(iii) For every $\varepsilon > 0$ there exists a norm $\|.\|_\varepsilon$ on the vector space $\mathbb{K}^n$ such that

$$\rho(A) \leq \|A\|_\varepsilon \leq \rho(A) + \varepsilon,$$

(iv) For any norm $\|.\|$ on $\mathbb{K}^n$ we have $\rho(A) = \lim_{m\to\infty} \|A^m\|^{1/m}$. □

Proof Th. 1.3. Since $\rho(M) < 1$, we conclude that $I - M$ is regular ($1 \notin \sigma(M)$), and so the fixed point equation $x = Mx + Nb$ has a unique solution x_*. When we now subtract the fixed point equation and the linear iteration $x_{k+1} = Mx_k + Nb$, denoting the error of the k-th iterate with $\varepsilon_k = x_* - x_k$, we get

$$\varepsilon_{k+1} = M\varepsilon_k = M^2\varepsilon_{k-1} = \ldots = M^{k+1}\varepsilon_0. \tag{9}$$

By Lemma (1.4) we can define a norm $\|.\|_\epsilon$ such that $\|M\|_\epsilon \leq \rho(M) + \epsilon < 1$, where we get

$$\|\varepsilon_{k+1}\|_\epsilon = \|M^{k+1}\varepsilon_0\|_\epsilon \leq \|M^{k+1}\|_\epsilon \|\varepsilon_0\|_\epsilon \leq \|M\|_\epsilon^{k+1}\|\varepsilon_0\|_\epsilon \to 0, \quad \text{as } k \to \infty.$$

This convergence is true in any norm, since every norm is equivalent in $\mathbb{K}^n$ ($\mathbb{K} = \mathbb{C}$ or $\mathbb{R}$). □

Corollary 1.5. *Let the linear fixed point iteration* (6) *(with Φ from* (7)*) be consistent with the invertible matrix A and $\rho(M) < 1$. Then for every starting point the iteration converges to the solution of $Ax = b$.* □

We remark that the invertibility of N is necessary for convergence, because otherwise (by consistency) the matrix $I - M = NA$ has an eigenvalue $\lambda = 0$, which contradicts $\rho(M) < 1$.

We also have that $\rho(M) < 1$ is necessary for convergence:

Theorem 1.6 (Proof [1, 3]). *Consider the linear fixed point iteration* (6) *with Φ from* (7)*. Let further $\rho(M) \geq 1$. Then the iteration* (6) *does not converge, i.e., either there are starting vectors x_0 and r.h.s. b such that the sequence (x_k) is not convergent or the iterates corresponding to different starting values converge to different fixed points.* □

Ad Proof. We can choose $b = 0$ and pick $\lambda \in \sigma(M)$ with $|\lambda| = \rho(M) \geq 1$ with associated eigenvector $x_0 \neq 0$. Then we have $x_k = M^k x_0 = \lambda^k x_0$. If $|\lambda| > 1$ then $|x_k| \to \infty$. In the case $|\lambda| = 1$ with $\lambda \neq 1$ we have $\lambda = e^{i\varphi}$ with $\varphi \in (0, 2\pi)$ and $x_k = e^{ik\varphi}x_0$, which is not converging (no Cauchy sequence). In the final case of $\lambda = 1$ we get the trivial iterates $x_k \equiv x_0$ which 'converge' to different limits as the starting value would change. □

The contraction property, used in convergence studies of fixed point iterations, indicates a reduction of the error by a factor $q \in (0, 1)$ each iteration. Since the norm of the error of the k-th iterate $\varepsilon_k = x_* - x_k$ is $\|\varepsilon_k\| = \|M^k\varepsilon_0\| \leq \|M^k\|\|\varepsilon_0\|$, the quantity $q := \limsup_{k\to\infty}(\|\varepsilon_k\|/\|\varepsilon_0\|)^{1/k} = \limsup_{k\to\infty}\|M^k\|^{1/k} = \rho(M)$ determines the (asymptotic) error reduction each step (convergence factor).
It should be mentioned here that the size of $\rho(M)$ only determines the asymptotic behavior of convergence [2], whereas the empirically observed convergence rate might be very different. For instance, in the case of non-normal M the norm $\|M\|_2$ [3] might be greater than 1.

1.3.2 Splitting methods: Fixed point and residual form

Consider the splitting of the matrix A by

$$A = G - H. \tag{10}$$

To construct a consistent fixed point iteration we start with

$$Ax_* = b \quad \to \quad Gx_* = Hx_* + b. \tag{11}$$

[2] $\rho(M)$ is also called *convergence factor*, while $-\ln\rho(M)$ is often referred to as *convergence rate*.
[3] Compare with Banach fixed point criterion, where $\|M\|$ is connected with the 'rate of contraction'.

References

[1] Y. Saad. *Iterative Methods for sparse linear systems - Second Edition.* SIAM 2003.

[2] Y. Saad. *Numerical Methods for Large Eigenvalue Problems - Second Edition.* SIAM 2011.

[3] W. Auzinger and J.M. Melenk. *Iterative Solution of Large Linear Systems..* Lecture notes, TU Wien, 2009.

[4] O. Scherzer. *CO-MAT1.* Lecture notes, Uni Wien, 2014/15.

[5] L. Exl. *Numerical Mathematics II.* Lecture notes in Physics, Uni Wien, 2016-2023.

[6] L. Exl. *Iterative Methoden für große dünnbesetzte lineare Gleichungssysteme.* Lecture notes, Universität Hamburg, 2012.

[7] L. Exl. *Mathematik 1 für Industrial Simulation* Lecture notes, FH St. Pölten, 2011.

[8] L. Exl. *Mathematik 2 für Industrial Simulation* Lecture notes, FH St. Pölten, 2012.

[9] N. Trefethen and D. Bau III. *Numerical Linear Algebra.* SIAM, Philadelphia, 1997.

[10] A. Quarteroni, R. Sacco, and F. Saleri *Numerical Mathematics.* Springer Verlag, Berlin, 2000.

[11] J. Nocedal and S. Wright. *Numerical Optimization - Second Edition.* Springer Verlag, Science & Business Media, 2006.

Now consider G being invertible and denote

$$N = G^{-1}, \tag{12}$$

and (left-)multiply the r.h.s. of (11) by N. This motivates the definition of the consistent *linear fixed point iteration*

$$x_{k+1} = \underbrace{N\,H}_{=:M} x_k + Nb = Mx_k + Nb. \tag{13}$$

Lemma 1.7. *The fixed point iteration* (13) *is consistent.*

Proof. This holds true by construction. Indeed, we have for any r.h.s. b that for a fixed point of (13) there holds $x_* = G^{-1}Hx_* + G^{-1}b$. After left multiplication by G, we arrive at $Gx_* = Hx_* + b$, and hence, $(G - H)x_* = Ax_* = b$. □

Note that N is not computed, only the action $y \mapsto G^{-1}y$ or the approximate solution of $Gz = y$ needs to be computationally realized efficiently. The matrix N is often considered to be an approximate inverse to A. [4] This can be motivated by the following *residual (or correction) form* of the linear fixed point iteration (13): After construction (13) is consistent and we have (8)

$$M = I - NA, \tag{14}$$

and hence,

$$x_{k+1} = (I - NA)x_k + Nb = x_k + N(b - Ax_k) = x_k + Nr_k, \tag{15}$$

where $r_k := b - Ax_k$ is the *residual* of the k-th iterate. The iteration given in Eqn. (15) is called the *residual form.* If N is considered to be a good approximation to A^{-1} we indeed have

$$x_k + N(b - Ax_k) = x_k + \underbrace{Nb}_{\approx x_*} - \underbrace{NA}_{\approx I} x_k \approx x_k + x_* - x_k = x_*, \tag{16}$$

which motivates the choice of x_{k+1} in (15).

1.3.3 (Damped) Richardson, (damped) Jacobi, Gauss-Seidel, SOR, SSOR

(Damped) Richardson iteration: The easiest case is the splitting $A = G - H = I - (I - A)$, i.e., the approximate inverse is the identity, $(G^{-1} =)N = I$, and the iteration matrix is $M = I - A$. This leads to the *Richardson iteration*

$$x_{k+1} = x_k + (b - Ax_k) = x_k + r_k. \tag{17}$$

We see from the residual form (15) that every linear iteration can be interpreted as a Richardson iteration applied to the transformed system

$$NAx = Nb. \tag{18}$$

The matrix N is called the *preconditioner* and considered to be a reasonable approximation of the inverse A^{-1}. We emphasize that by choosing the approximate inverse $N = G^{-1}$, different iterative methods can be constructed based on the splitting $A = G - H$.
By choosing a real *relaxation or damping parameter* $\omega \in \mathbb{R}$ we get the damped version of an iteration by setting

$$x_{k+1} = x_k + \omega N(b - Ax_k) = x_k + \omega N r_k. \tag{19}$$

The damped Richardson iteration corresponds to the choice

$$x_{k+1} = x_k + \omega(b - Ax_k). \tag{20}$$

[4]The inverse of N, an approximation of A itself, is called the preconditioner. See Ch. 1.7 for more details.

One can show that $\sigma(M^{dRich}) = \{1 - \omega\lambda : \lambda \in \sigma(A)\}$ (exercise). Convergence can be achieved by choosing ω in accordance with Th. 1.3.

The splitting methods described below will use the following decomposition of the matrix A:

$$A = D + L + U, \tag{21}$$

where D is the diagonal of A, and L and U the lower and upper part of A, respectively. If A is symmetric, then $L = U^T$.

(Damped) Jacobi iteration: Here $A = G - H = D - (-L - U)$. Thus, we have $N = D^{-1}$ (nonzero diagonal entries are assumed here) and the iteration

$$x_{k+1} = x_k + D^{-1}(b - Ax_k) = x_k + D^{-1}r_k. \tag{22}$$

Since the inversion of the diagonal matrix is trivial we have the explicit component form

$$x_{k+1,i} = \frac{1}{a_{ii}}\left(b_i - \sum_{j\neq i} a_{ij}x_{k,j}\right), \quad i = 1, \ldots, n. \tag{23}$$

The *damped Jacobi iteration* takes the form

$$x_{k+1} = x_k + \omega D^{-1}(b - Ax_k) = x_k + \omega D^{-1}r_k. \tag{24}$$

If A is SPD, then the damped Jacobi iteration converges if and only if $\frac{2}{\omega}D - A$ is positive definite (proof [3]). This is a property in the sense of 'diagonal dominance'. However, the regime of damping parameters which lead to a convergent Jacobi iteration depend on the problem even in the SPD case. This is in contrast to other methods like the so-called *Successive overrelaxation* (SOR), which converges for $\omega \in (0, 2)$ in the SPD case. However, convergence of the Jacobi method is ensured in the strictly diagonally dominant case, that is $|a_{ii}| > \sum_{j\neq i} |a_{ij}|, \quad i = 1, \ldots, n$.

Gauss-Seidel: Choose $A = G - H = (D + L) - (-U)$, which leads to

$$x_{k+1} = x_k + (D + L)^{-1}(b - Ax_k). \tag{25}$$

This is called the *forward Gauss-Seidel* methods; a *backward* version is obtained by choosing $G = D+U$, instead of $G = D + L$. Note that, if the diagonal of A contains no zeros, the triangular matrices $D + L$ or $D + U$ are easily inverted by 'forward substitution' or 'back substitution', respectively. In component form forward Gauss-Seidel reads

$$x_{k+1,i} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_{k+1,j} - \sum_{j=i+1}^{n} a_{ij}x_{k,j}\right), \quad i = 1, \ldots, n, \tag{26}$$

where the first sum on the r.h.s. contains the already computed components $x_{k+1,j}$ for $j < i$. The component form translates to matrix form again:[5]

$$x_{k+1} = (D + L)^{-1}(b - Ux_k). \tag{27}$$

The convergence property of Gauss-Seidel is contained in that of the next method (SOR), since it is a more general version of Gauss-Seidel. However, convergence of the Gauss-Seidel method is ensured in the strictly diagonal dominant case.

Successive OverRelaxation (SOR): The SOR is a variant of the Gauss-Seidel method that improves convergence rate by using a linear combination of the last iterate x_k and the Gauss-Seidel iterate x_{k+1}

[5]In the backward version L and U interchange.

(Eqn. 26). The method uses the relaxation parameter $\omega \in (0,2)$, which is often chosen heuristically in dependence of the specific problem. The iteration is given by

$$x_{k+1,i} = x^{k,i}(1-\omega) + \frac{\omega}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_{k+1,j} - \sum_{j=i+1}^{n} a_{ij}x_{k,j}\right), \quad i = 1,\dots,n. \tag{28}$$

This can be recast in matrix notation in residual form as

$$x_{k+1} = x_k + \omega(D+\omega L)^{-1}(b - Ax_k), \tag{29}$$

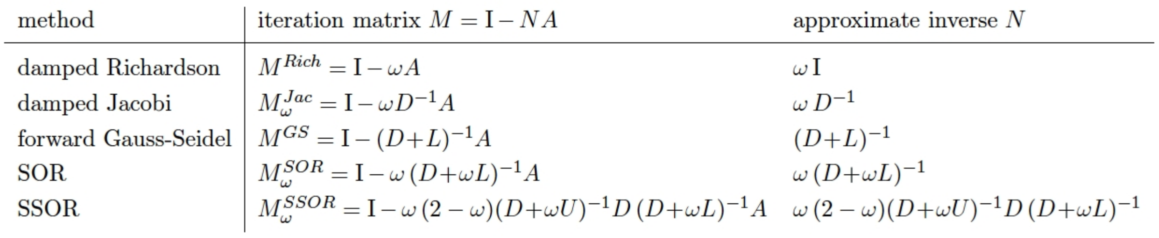
which gives (forward) Gauss-Seidel for $\omega = 1$.
It can be shown that SOR converges in the SPD case for $\omega \in (0,2)$, [3].

Symmetric SOR (SSOR): Both, Gauss-Seidel and SOR depend on the numbering of the unknowns, in contrast to the Richardson and Jacobi method. Also, the iteration matrix and the approximate inverse is not symmetric even if the original matrix is symmetric. This can be overcome by the symmetric version of SOR, namely SSOR. It is constructed by two half-steps of SOR, one based on forward Gauss-Seidel, the other based on backward Gauss-Seidel. This leads to the iteration

$$x_{k+1} = x_k + \omega(2-\omega)(D+\omega U)^{-1}D(D+\omega L)^{-1}(b - Ax_k). \tag{30}$$

It can be shown that SSOR converges in the SPD case for $\omega \in (0,2)$, [3].

A collection of the different iteration matrices M and approximate inverses N are listed in Fig. 4.



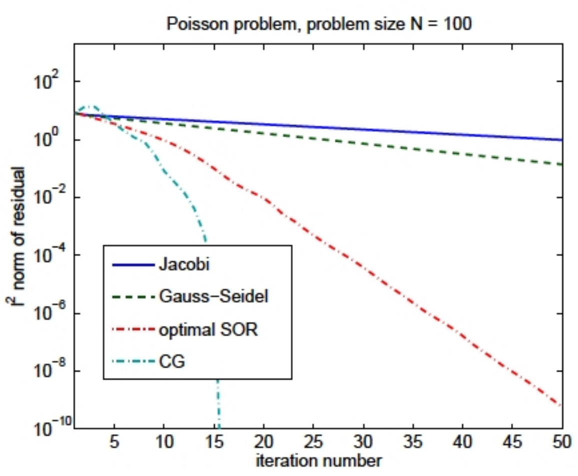
method	iteration matrix $M = \mathrm{I} - NA$	approximate inverse N
damped Richardson	$M^{Rich} = \mathrm{I} - \omega A$	$\omega\,\mathrm{I}$
damped Jacobi	$M_\omega^{Jac} = \mathrm{I} - \omega D^{-1}A$	$\omega\, D^{-1}$
forward Gauss-Seidel	$M^{GS} = \mathrm{I} - (D+L)^{-1}A$	$(D+L)^{-1}$
SOR	$M_\omega^{SOR} = \mathrm{I} - \omega\,(D+\omega L)^{-1}A$	$\omega\,(D+\omega L)^{-1}$
SSOR	$M_\omega^{SSOR} = \mathrm{I} - \omega\,(2-\omega)(D+\omega U)^{-1}D\,(D+\omega L)^{-1}A$	$\omega\,(2-\omega)(D+\omega U)^{-1}D\,(D+\omega L)^{-1}$

Figure 4: Summary iterative methods, taken from [3].

1.3.4 Remarks on optimal damping parameter

In general it is not possible to determine the optimal damping parameter analytically. However, in the case of SPD matrices the optimal damping parameter for the Richardson and Jacobi method takes the form $\omega_{opt} = 2/(\lambda_{min} + \lambda_{max})$, where $\lambda_{min/max}$ is the minimum and maximum eigenvalue of A, respectively. For so-called *consistently ordered* SPD matrices [1] (e.g. K_{2d} Poisson matrix Fig. 1), it is possible to determine the optimal damping parameter for the SOR method analytically by exploiting the so-called *Young theorem*, [1, 3]: $\omega_{opt} = 2/(1+\sqrt{1-\beta^2})$, $\quad \beta := \rho(M^{Jac})$. In the case of the K_{2d} matrix of the uniformly discretized Laplace operator in two dimensions (cf. Fig. 1) the spectrum of M^{Jac} can be computed analytically, leading to $\beta = \rho(M^{Jac}) = 1 - ch^2 + \mathcal{O}(h^3)$, $c > 0$, where $h = 1/(n+1)$ is the mesh size. In this case the optimal damping parameters for Jacobi and SOR can be computed, as well as the convergence factor (rate) of the optimally damped SOR method, which turns out to be $\rho(M^{SOR,\omega_{opt}}) = 1 - \widetilde{c}h + \mathcal{O}(h^2)$. Note, that the latter spectral radius indicates a significant improvement for the convergence rate compared to Jacobi.
Fig. 5 shows a comparison of the convergence for the $2d$ Poisson problem for different methods. The left figure shows convergence for all methods, whereas in the right figure Jacobi and Gauss-Seidel do not converge practically speaking. However, from the left figure one can observe that Gauss-Seidel converges at twice the convergence rate of Jacobi. [6] The optimally damped SOR is faster in both cases, whereas the CG method is obviously superior here. Convergence gets slower as problem size increases.

[6]For consistently ordered matrices there holds $\rho(M^{GS}) = \rho(M^{Jac})^2$.



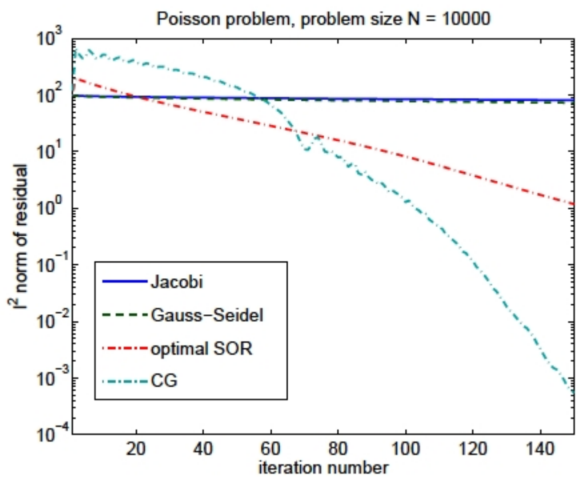


Figure 5: Comparison for the $2d$ Poisson problem for Jacobi, Gauss-Seidel, optimally damped SOR and CG (Conjugate Gradient, see Sec.1.5), taken from [3].

1.4 Gradient methods and steepest descent variants

1.4.1 Gradient methods

We consider here the case of a SPD matrix $A \in \mathbb{R}^{n\times n}$. Gradient methods are based on the observation that critical points of a function correspond to solutions of the linear system.

Theorem 1.8. *Let $A \in \mathbb{R}^{n\times n}$ be SPD. The function*

$$\Phi(x) := \frac{1}{2}x^T Ax - b^T x \tag{31}$$

fulfills[7]

$$\nabla\Phi(x) = Ax - b. \tag{32}$$

Further, the (global) minimizer of Φ, which is given through $\nabla\Phi(x) = 0$, solves the linear system $Ax = b$.

Proof. Looking at the linear parts in the perturbation $\delta \in \mathbb{R}^n$ in the expression $\Phi(x+\delta) - \Phi(x)$ and using the symmetry of A reveals $\nabla\Phi(x) = Ax - b$. Further, since the Hessian of (31) is A and positive definite (by assumption), the function (31) is strictly convex and we conclude that the solution of $Ax = b$ is the unique minimizer of (31). □

The following lemma identifies the minimization of the function Φ as the minimization of the energy error.

Lemma 1.9. *Minimizing (31) is equivalent to minimizing the error $x - x_*$ in the energy norm induced by A, that is $\|x - x_*\|_A^2$.*

Proof. We have

$$\begin{aligned}\Phi(x) - \Phi(x_*) &= \frac{1}{2}x^T Ax - x^T b - \frac{1}{2}x_*^T Ax_* + x_*^T b \\ &= \frac{1}{2}(x-x_*)^T A(x-x_*) + \underbrace{\frac{1}{2}\left(x^T Ax_* + x_*^T Ax\right) - x_*^T Ax_* - x^T b + x_*^T b}_{=0, \text{ since } Ax_*=b} \\ &= \frac{1}{2}(x-x_*)^T A(x-x_*) \\ &= \frac{1}{2}\|x-x_*\|_A^2.\end{aligned} \tag{33}$$

[7]If A is not symmetric, there holds $\nabla\Phi(x) = \frac{1}{2}(A + A^T)x - b$.

□

One class of numerical methods for minimization are so-called *line search methods* which take the form

$$x_{k+1} = x_k + \alpha_k d_k, \tag{34}$$

where the update on the r.h.s. consists of the *search direction* $d_k \in \mathbb{R}^n$ and the *step length* $\alpha_k \in \mathbb{R}$. Once a search direction is chosen, the step length is determined by a one-dimensional minimization problem (*line search*)

$$\alpha_k = \arg\min_{\alpha\in\mathbb{R}} \Phi(x_k + \alpha d_k). \tag{35}$$

Lemma 1.10. *In the present case of the quadratic function* Φ *in* (31) *it is possible to solve* (35) *explicitly as*

$$\alpha_k = \frac{(b - Ax_k)^T d_k}{d_k^T A d_k} = \frac{r_k^T d_k}{d_k^T A d_k}. \tag{36}$$

Proof. Exercise. □

Many different search directions could be considered. For instance, if we choose the k-th coordinate unit vectors $d_k = e_k = (0, \dots 0, \underbrace{1}_{k-th}, 0, \dots, 0)^T$ for the k-th update, we have

$$x_{k+1} = x_k + \frac{(r_k)_k}{a_{kk}} d_k, \tag{37}$$

which exactly corresponds to the k-th update in the inner loop of a Gauss-Seidel step, thus, n of these updates, successively applied and with cyclic choice of unit vectors as search directions, result in a single Gauss-Seidel step.
If we choose $d_k = r_k$ but $\alpha_k \equiv 1$ (or ω) we arrive at the (damped) Richardson iteration.

1.4.2 The steepest descent method

In the *steepest descent (SD) method* the step length is locally optimized corresponding to (36) for the choice $d_k = r_k = -\nabla\Phi(x_k)$, i.e.,

$$\alpha_k = \frac{r_k^T r_k}{r_k^T A r_k} = \frac{\|r_k\|_2^2}{\|r_k\|_A^2}. \tag{38}$$

Algorithm 3: Steepest descent for solving $Ax = b,\ A$ SPD.

```
Data: Matrix $A \in \mathbb{R}^{n\times n}$ SPD, r.h.s. $b \in \mathbb{R}^n$, initial vector $x_0 \in \mathbb{R}^n$
Result: $x \in \mathbb{R}^n$ approximate solution
Initialization: Set $k \leftarrow 0$
while Convergence criterion not satisfied do
    $r_k \leftarrow b - Ax_k$
    $\alpha_k \leftarrow \frac{r_k^T r_k}{r_k^T A r_k}$
    $x_{k+1} \leftarrow x_k + \alpha_k r_k$
    $k \leftarrow k + 1$
end
$x \leftarrow x_k$
```

The choice $d_k = r_k = -\nabla\Phi(x_k)$, namely the negative gradient direction, yields the fastest rate of decrease *locally* (direction of steepest descent).[8] The steepest descent algorithm is given in Alg. 3. We note, that the number of matrix-vector multiplications as given in the pseudo-algorithm Alg. 3 is not optimal, in fact, it can be reduced to just one matrix-vector multiplication inside the loop by precomputing $p_k := Ar_k$ and using the fact $r_{k+1} = r_k - \alpha_k p_k$. *(exercise)*
As *convergence criterion* one can measure the (relative) residual length, i.e., $\|r_k\|$ (or $\|r_k\|/\|b\|$ resp.) and terminate if the error is below a prescribed threshold $0 < \epsilon \ll 1$. The length of successive iterates, i.e., $\|x_{k+1} - x_k\|$ or $\|x_{k+1} - x_k\|_A$, would yield a similar criterion. Furthermore, a maximum iteration number should be prescribed.
It is notable that in the steepest descent method *consecutive search directions are orthogonal* to each other w.r.t. the Euclidean inner product.

Lemma 1.11. *Consecutive search directions in the SD method are orthogonal, that is, $d_{k+1}^T d_k = 0$, see also Fig. 6.*

Proof. This can be seen by observing that $d_{k+1} = b - Ax_{k+1} = b - A(x_k + \alpha_k r_k) = r_k - \alpha_k Ar_k$. Inserting the step length (38) yields

$$d_{k+1}^T d_k = (r_k - \alpha_k Ar_k)^T r_k = r_k^T r_k - \alpha_k r_k^T Ar_k = r_k^T r_k - \frac{r_k^T r_k}{r_k^T Ar_k} r_k^T Ar_k = 0. \tag{39}$$

□

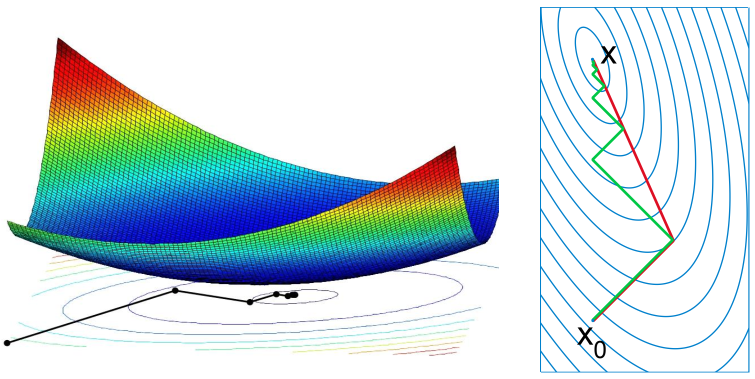


Figure 6: Two dimensional illustration of the convergence process for the minimization of the functional Φ from (31) corr. to a SPD system $Ax = b$. Consecutive search directions (right figure, green lines) in the SD method are orthogonal.

1.4.3 Convergence of the SD method

In the SPD case the SD method converges and the speed of convergence depends on the spectrum of A, more precisely, on the condition number $\kappa_2(A) = \lambda_{max}/\lambda_{min}$. The following theorem estimates the error in the energy norm.

Theorem 1.12. *The SD method applied to the SPD system $Ax = b$ yields the following estimate for the error $\varepsilon_k := x_* - x_k$ in the energy norm*

$$\|\varepsilon_k\|_A \leq \left(\frac{\kappa_2(A) - 1}{\kappa_2(A) + 1}\right)^k \|\varepsilon_0\|_A . \tag{40}$$

Hence, $\varepsilon_k \to 0$ for $k \to \infty$. □

[8] $\nabla\Phi(x_k)$ would yield the direction of steepest ascent.

Ad proof. The proof uses the *Kantorovich inequality* as a first ingredient: Let B be SPD and $n \times n$ with extreme eigenvalues $\lambda_{min}, \lambda_{max} > 0$. Then

$$\frac{\|x\|_B^2 \|x\|_{B^{-1}}^2}{\|x\|^4} \leq \frac{(\lambda_{min} + \lambda_{max})^2}{4\lambda_{max}\lambda_{min}}. \tag{41}$$

Secondly, we have by Lemma 1.9 that $\frac{1}{2}\|\varepsilon_k\|_A^2 = \Phi(x_k) - \Phi(x_*)$. Further, by elementary calculation, one gets for $x_{k+1} = x_k + \alpha_k r_k$ with optimal α_k that $\Phi(x_{k+1}) - \Phi(x_k) = -\frac{1}{2}\frac{\|r_k\|^4}{\|r_k\|_A^2}$. All together this leads to

$$\tfrac{1}{2}\|\varepsilon_{k+1}\|_A^2 = \Phi(x_{k+1}) - \Phi(x_*) = \Big(\Phi(x_{k+1}) - \Phi(x_k)\Big) + \Big(\Phi(x_k) - \Phi(x_*)\Big) = -\tfrac{1}{2}\frac{\|r_k\|^4}{\|r_k\|_A^2} + \tfrac{1}{2}\|\varepsilon_k\|_A^2. \tag{42}$$

By the Kantorovich inequality (41) we have

$$\frac{\|r_k\|^4}{\|r_k\|_A^2} \geq \frac{4\lambda_{max}\lambda_{min}}{(\lambda_{min} + \lambda_{max})^2}\|r_k\|_{A^{-1}}^2 = \frac{4\lambda_{max}\lambda_{min}}{(\lambda_{min} + \lambda_{max})^2}\|\varepsilon_k\|_A^2. \tag{43}$$

We can therefore simplify (42) to

$$\|\varepsilon_{k+1}\|_A^2 \leq \Big(1 - \frac{4\lambda_{max}\lambda_{min}}{(\lambda_{min} + \lambda_{max})^2}\Big)\|\varepsilon_k\|_A^2 = \Big(\frac{\lambda_{max} - \lambda_{min}}{\lambda_{min} + \lambda_{max}}\Big)^2\|\varepsilon_k\|_A^2 = \Big(\frac{\kappa_2(A) - 1}{\kappa_2(A) + 1}\Big)^2\|\varepsilon_k\|_A^2. \tag{44}$$

□

If the condition number is large, the contour lines of the quadratic functional Φ in (31) are elongated ellipses and the orthogonality of the consecutive search directions yields an inefficient 'zig-zag' path, see Fig. 6 where a two-dimensional ($n = 2$) example is illustrated.

1.4.4 Variants of the SD method

Residual Norm Steepest Descent. The SD method described above requires A to be SPD. If $A \in \mathbb{R}^{n\times n}$ is nonsingular but not symmetric one can rewrite the problem to the (in the present case) mathematically equivalent *normal equations* [9]

$$A^T A x = A^T b, \tag{45}$$

and apply the steepest descent method (Alg. 3) to the system (45). It is called the *residual norm steepest descent method* [1] because it minimizes the residual norm $\|b - Ax\|_2^2$. In fact, if one defines for nonsingular square matrix A the quadratic functional

$$\widetilde{\Phi}(x) = \frac{1}{2}\|b - Ax\|_2^2, \tag{46}$$

its gradient is $\nabla\widetilde{\Phi}(x) = A^T A x - A^T b$, thus critical points solve (45). Since, $A^T A$ is SPD iff A is nonsingular, the minimizer is unique and coincides with the solution of $Ax = b$.
Often the condition number of $A^T A$ is much larger[10] than that of A, which makes methods via the normal equation often numerically inefficient. We will overcome this by the *generalized minimal residual* (GMRES) method in chapter 1.6.

Minimal Residual (MR) Iteration. One variant is to use the steepest descent direction for the functional (31) but the step length is chosen in a line search to minimize the squared norm of the new residual $\|b - Ax_{k+1}\|_2^2 = \|b - A(x_k + \alpha_k r_k)\|_2^2 = \|r_k - \alpha_k A r_k\|_2^2$, resulting in $\alpha_k = r_k^T A r_k / \|A r_k\|_2^2$. One can show that this iteration converges if the matrix is not necessarily symmetric [1]. A general version *GMRES* for only nonsingular requirement will be discussed in Ch. 1.6.

[9] Known from linear least squares problems for overdetermined systems (regression).
[10] Typically near the squared value $\kappa_2(A)^2$.

1.5 The conjugate gradient method

1.5.1 Motivation and Krylov subspaces

The *conjugate gradient (CG) method* is a very effective algorithm for solving the SPD system $Ax = b$. It is a perfect example of a *Krylov subspace method*, which will be discussed for general systems (not necessarily SPD) in Ch. 1.6. The inverse of the matrix A can be described as a polynomial in A of degree less than n by using the *Cayley-Hamilton theorem*:

Theorem 1.12. *Let $A \in \mathbb{C}^{n\times n}$ be a matrix with characteristic polynomial $\chi(z)$. Then, the matrix A satisifies its own characteristic equation, i.e. $\chi(A) = 0$.* □

In $\mathbb{C}$ we may write χ as

$$\chi(z) = c_n z^n + c_{n-1}z^{n-1} + \ldots + c_1 z + c_0. \tag{47}$$

Theorem 1.12 states that

$$\chi(A) = c_n A^n + c_{n-1}A^{n-1} + \ldots + c_1 A + c_0 I = 0 \tag{48}$$

is satisified as a matrix equation. A direct consequence of (48) for a nonsingular matrix $A \in \mathbb{R}^{n\times n}$ is a representation of the inverse A^{-1} as a matrix polynomial in A of degree less than n, i.e. by multiplying (48) with A^{-1} and dividing by $c_0 = \det(A) \neq 0$ (A is nonsingular) we get

$$A^{-1} = -\frac{c_n}{c_0}A^{n-1} - \frac{c_{n-1}}{c_0}A^{n-2} - \ldots - \frac{c_1}{c_0}I. \tag{49}$$

Now note, that the solution x_* of $Ax = b$ can be written in the form

$$x_* = x_0 + A^{-1}r_0 \tag{50}$$

with the residual $r_0 = b - Ax_0$. The term $A^{-1}r_0$ can be constructed by using the Cayley-Hamilton theorem

$$A^{-1}r_0 = (d_{n-1}A^{n-1} + d_{n-2}A^{n-2} + \ldots + d_1 I)r_0, \tag{51}$$

where the coefficients are determined by the characteristic polynomial. This means that

$$A^{-1}r_0 \in \mathcal{K}_n := \operatorname{span}\{r_0, Ar_0, \ldots, A^{n-1}r_0\}, \tag{52}$$

and hence, the solution x_* satisfies

$$x_* \in x_0 + \mathcal{K}_n. \tag{53}$$

Definition 1.13 (Krylov subspace). *In general, we define for $m \geq 1$ the m-th Krylov space of A w.r.t. the initial residual r_0 as (compare with [1])*

$$\mathcal{K}_m = \mathcal{K}_m(A, r_0) = \mathit{span}\{r_0, Ar_0, A^2r_0, \ldots, A^{m-1}r_0\}, \tag{54}$$

which has dimension less or equal m.[11]

As will be discussed in Ch. 1.6, Krylov subspace methods try to approximate the solution x_* by

$$x_* \approx x_m \in x_0 + \mathcal{K}_m, \tag{55}$$

with $m \ll n$.

The *conjugate gradient method* is the historically earliest example of such a method, developed for SPD systems by Hestenes, Stiefel and Lanczos in the $50's$. It minimizes the error $\varepsilon_k = x_* - x_k$ in the energy norm while constructing the iterates $x_k \in x_0 + \mathcal{K}_k(A, r_0)$, i.e.

$$x_k = \underset{x\in x_0+\mathcal{K}_k(A,r_0)}{\arg\min} \|x - x_*\|_A = \underset{x\in x_0+\mathcal{K}_k(A,r_0)}{\arg\min} \Phi(x), \tag{56}$$

see (33) for the latter equality.

[11]Note, that $x_* = A^{-1}b \in \mathcal{K}_n(A, b)$ is independent of the starting value. Therefore, some authors define the m-th Krylov space as $\mathcal{K}_m(A, b)$.

1.5.2 Conjugate direction methods

The conjugate gradient method is a so-called *conjugate direction method* for minimizing the quadratic functional (31).

Definition 1.14 (Conjugate direction methods). *In a conjugate direction method the new iterates are determined by*

$$x_{k+1} = x_k + \alpha_k p_k, \tag{57}$$

where the set of search directions $\{p_0, p_1, \dots, p_k\}$ are conjugate w.r.t. the SPD matrix A, that is, orthogonal w.r.t. the energy product, i.e., [12]

$$p_i^T A p_j = 0, \quad \textit{for all } i \neq j. \tag{58}$$

The step length α_k in (57) *is the minimizer along p_k (line search - compare with* (36)*) given by*

$$\alpha_k = \frac{r_k^T p_k}{p_k^T A p_k}. \tag{59}$$

The property (58) implies linear independence of the directions.
We can motivate the sense in using conjugate directions in the two dimensional case by the following observation:

Remark 1.15 (Motivation for conjugate directions (in 2d)). Consider the convex minimization problem (31) in $\mathbb{R}^2$. Let $x_0 \in \mathbb{R}^2$ be any initial guess. We choose any descent direction $p_0 \in \mathbb{R}^2$, that is a direction p_0 with $p_0^T \nabla\Phi(x_0) < 0$ [13] Let us define the new approximation as $x_1 = x_0 + \alpha_0 p_0$ with the optimal α_0 as in (59). Then the directions $x_* - x_1$ and p_0 are orthogonal w.r.t. the A-inner product:

$$\langle p_0, x_* - x_1 \rangle_A = p_0^T A(x_* - x_0 - \alpha_0 p_0) = p_0^T r_0 - \alpha_0\, p_0^T A p_0 = p_0^T r_0 - r_0^T p_0 = 0. \tag{60}$$

This means, convergence in two steps (in the two dimensional case) requires the directions p_0 and p_1 to be conjugate. □

In fact, there holds the following remarkable theorem:

Theorem 1.16. *For any starting vector x_0 the sequence generated by a conjugate direction method* (57) *converges to the solution x_* of the system $Ax = b$, where A is $n \times n$ and SPD, in at most n steps.*

Proof. From the conjugacy of a set $\{p_i\}_{i=0,\dots,n-1}$ one gets linear independence of this set. In particular, there holds $\mathbb{R}^n = \operatorname{span}\{p_0, p_1, \dots, p_{n-1}\}$. This means one can represent the initial error as

$$x_* - x_0 = \sigma_0 p_0 + \sigma_1 p_1 + \dots + \sigma_{n-1} p_{n-1} \quad \text{for } \sigma_j \in \mathbb{R},\ j = 0, \dots, n-1. \tag{61}$$

Multiplying (61) with $p_k^T A$ for $k \in \{0, \dots, n-1\}$ yields, owing to conjugacy,

$$\sigma_k = \frac{p_k^T A(x_* - x_0)}{p_k^T A p_k}. \tag{62}$$

The proof is done when we show that $\sigma_k = \alpha_k$, the optimally chosen step length from line search (59): A conjugate direction algorithm generates the iterate

$$x_k = x_0 + \alpha_0 p_0 + \alpha_1 p_1 + \dots + \alpha_{k-1} p_{k-1}.$$

Multiplication with $p_k^T A$ yields (analogue to above) $p_k^T A(x_k - x_0) = 0$ and hence

$$p_k^T A(x_* - x_0) = p_k^T A\big((x_* - x_k) + (x_k - x_0)\big) = p_k^T A(x_* - x_k) = p_k^T(b - Ax_k) = p_k^T r_k. \tag{63}$$

[12]The eigenvectors of A symmetric are an example for conjugate directions.

[13]The angle between the steepest descent direction $-\nabla\Phi(x_0)$ at x_0 and the descent direction p_0 is between $-90°$ and $90°$.

Now compare (62) with (59) to see $\sigma_k = \alpha_k$. $\square$

In two dimensions one can also illustrate the special role of conjugate directions by the following [11] interpretation. Choosing conjugate directions corresponds to a coordinate transform where the contour lines of the quadratic function (31) are ellipses which are aligned along the coordinate directions in the transformed system. A solution can therefore be found by successive minimization along the coordinate directions (these are the conjugate directions) in at most two steps, see Fig. 7. In contrast, successive minimization along the coordinate directions in the original system does not give the exact solution in two steps because in general the contour ellipses are no longer aligned along the coordinate directions.

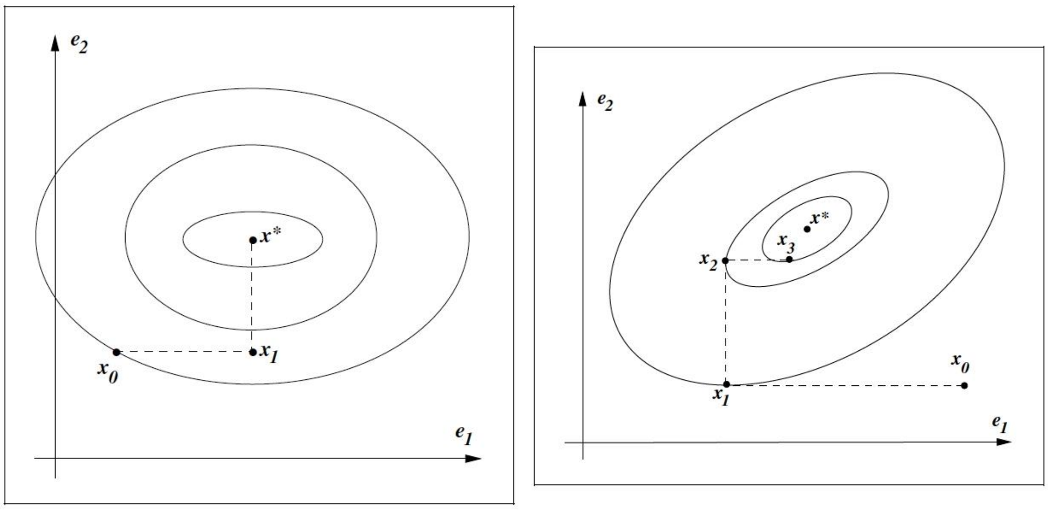


Figure 7: Two dimensional illustration of the convergence process for conjugate direction method (left) and simple successive coordinate direction minimization (right). Taken from [11].

There could be many choices for the set of conjugate directions, e.g. the eigenvectors of a symmetric matrix are pairwise orthogonal and conjugate. For large-scale problems the computation of (all) eigenvectors is too expensive. An alternative would be to modify the Gram-Schmidt orthogonalization process to orthogonality w.r.t. the energy product. But this is also very expensive since it would require to store all previous directions. In contrast, the conjugate gradient method is special and ultimately efficient because it generates the conjugate direction set by only using the previous direction. More precisely, the next direction is chosen as a linear combination of the steepest descent direction and the previous search direction.

1.5.3 The CG method

Definition 1.17 (Conjugate direction and step length in conjugate gradient (CG) method). *The CG method uses the linear combination*

$$p_k = r_k + \beta_k p_{k-1}, \tag{64}$$

to enforce conjugacy of p_k. Hence, the scalar β_k is determined by the requirement that p_k and p_{k-1} are conjugate, which yields

$$\beta_k = -\frac{p_{k-1}^T A r_k}{p_{k-1}^T A p_{k-1}} = -\frac{r_k^T A p_{k-1}}{p_{k-1}^T A p_{k-1}}. \tag{65}$$

The step length α_k is determined by one dimensional minimization along the new direction, see (59). The first direction in the CG method is the steepest descent direction.

There holds the following theorem which shows that the search directions are indeed mutually conjugate and that CG is a Krylov subspace method.

Theorem 1.18. *Let A be SPD. Suppose the k-th iterate of the CG method is not the solution x_* of the SPD system $Ax = b$. Then, there holds*

$$p_k^T A p_i = 0 \quad \text{for } i = 0, 1, \ldots, k-1, \tag{66}$$

and

$$\text{span}\{p_0, p_1, \ldots, p_{k-1}\} = \text{span}\{r_0, Ar_0, \ldots, A^{k-1}r_0\} = \mathcal{K}_k(A, r_0). \tag{67}$$

The CG method converges to x_* in at most n steps.

Ad proof. The proof is by induction with no special technical difficulties. It can be found e.g. in [11]. □

1.5.4 The practical algorithm

There are two more orthogonality relations involved here: It can be shown [11] that for conjugate direction methods there holds

$$r_k^T p_i = 0 \quad \text{for } i = 0, 1, \ldots, k-1 \tag{68}$$

and for the residuals in the CG method [14]

$$r_k^T r_i = 0 \quad \text{for } i = 0, 1, \ldots, k-1. \tag{69}$$

This leads to simplifications for the computation of the coefficients α_k and β_k:

Lemma 1.19. *The coefficients involved in the CG method as in Def. 1.17 can be reformulated in a practically more convenient way as*

$$\beta_{k+1} = -\frac{r_{k+1}^T A p_k}{p_k^T A p_k} = \frac{\|r_{k+1}\|_2^2}{\|r_k\|_2^2} \tag{70}$$

and

$$\alpha_k = \frac{r_k^T p_k}{p_k^T A p_k} = \frac{\|r_k\|_2^2}{\|p_k\|_A^2}. \tag{71}$$

Proof. From the construction rule $p_k = r_k + \beta_k p_{k-1}$ (64) of the conjugate direction we get by left-multiplying with r_k^T and using Eqn. (68) that

$$r_k^T p_k = r_k^T r_k = \|r_k\|_2^2, \tag{72}$$

and hence

$$\alpha_k = \frac{r_k^T p_k}{p_k^T A p_k} = \frac{\|r_k\|_2^2}{\|p_k\|_A^2}. \tag{73}$$

To simplify the expression (65) for β_k, note that from the update rule (57) one gets

$$r_{k+1} - r_k = -\alpha_k A p_k. \tag{74}$$

By left-multiplying (74) with p_k^T and using (68) and relation (72) we get

$$p_k^T A p_k = \frac{1}{\alpha_k} p_k^T r_k = \frac{1}{\alpha_k} \|r_k\|_2^2. \tag{75}$$

Now, left-multiplication of (74) with r_{k+1}^T and using (69) yields

$$r_{k+1}^T A p_k = -\frac{1}{\alpha_k} \|r_{k+1}\|_2^2. \tag{76}$$

[14] Use (68) and $p_i = r_i + \beta_i p_{i-1}$, $i > 1$ and $p_0 = r_0$.

Finally, we get from (75) and (76)

$$\beta_{k+1} = -\frac{r_{k+1}^T A p_k}{p_k^T A p_k} = \frac{\|r_{k+1}\|_2^2}{\|r_k\|_2^2}. \tag{77}$$

□

These simplifications allow to reduce the necessary matrix vector multiplications to only the product Ap_k in each iteration in the CG method. The previous residual is stored and the update formula (74) is used for the new residual. The CG method is summarized in Alg. 4.

```
Algorithm 4: CG method for solving $Ax = b$, $A$ SPD.
  Data: Matrix $A \in \mathbb{R}^{n\times n}$ SPD, r.h.s. $b \in \mathbb{R}^n$, initial vector $x_0 \in \mathbb{R}^n$
  Result: $x \in \mathbb{R}^n$ approximate solution
  Initialization: Set $r_0 \leftarrow b - Ax_0$, $p_0 \leftarrow r_0$, $k \leftarrow 0$.
  while Convergence criterion not satisfied do
      $\alpha_k \leftarrow \frac{r_k^T r_k}{p_k^T A p_k}$   see (71)
      $x_{k+1} \leftarrow x_k + \alpha_k p_k$   see (57)
      $r_{k+1} \leftarrow r_k - \alpha_k A p_k$   see (74)
      $\beta_{k+1} \leftarrow \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k}$   see (70)
      $p_{k+1} \leftarrow r_{k+1} + \beta_{k+1} p_k$   see (64)
      $k \leftarrow k + 1$
  end
  $x \leftarrow x_k$
```

1.5.5 Convergence of the CG method

For conjugate direction methods there holds the following property.

Theorem 1.20 (Expanding subspace minimization; Proof [11]). *Let $x_0 \in \mathbb{R}^n$ and x_k generated by a conjugate direction method (Def 1.14) with conjugate directions p_i, $i = 0, \ldots, k-1$. Then there holds the expanding subspace minimization property, that is*

$$x_k = \underset{\{x \mid x \in x_0 + span\{p_0,\ldots,p_{k-1}\}\}}{\arg\min} \; \tfrac{1}{2} x^T A x - b^T x. \tag{78}$$

□

In addition we know from (67) that for the CG method the expanding subspaces coincide with the Krylov subspaces, i.e.,

$$\operatorname{span}\{p_0, p_1, \ldots, p_{k-1}\} = \operatorname{span}\{r_0, Ar_0, \ldots, A^{k-1} r_0\} = \mathcal{K}_k(A, r_0). \tag{79}$$

The rate of convergence of the CG method can be estimated with the following lemma, which uses the expanding subspace minimization property.

Lemma 1.21. *For the error of the iterates x_k generated by the CG method holds*

$$\|x_* - x_{k+1}\|_A^2 \le \big(\min_{P_k \in \mathcal{P}_k} \max_{1 \le i \le n} [1 - \lambda_i P_k(\lambda_i)]^2 \big) \|x_* - x_0\|_A^2, \tag{80}$$

where $\mathcal{P}_k$ denotes polynomials of degree smaller or equal k and $\lambda_i > 0$, $i = 1, \ldots, n$ the eigenvalues of the SPD matrix A.

Proof. From (79) we have (with some constants γ_i, $i = 0, \dots, k$)

$$x_{k+1} = x_0 + \gamma_0 r_0 + \gamma_1 A r_0 + \dots + \gamma_k A^k r_0 = x_0 + P_k^*(A) r_0, \tag{81}$$

for a polynomial $P_k^* \in \mathcal{P}_k$. Now recall from (33) the property

$$\frac{1}{2}\|x_* - x\|_A^2 = \Phi(x) - \Phi(x_*). \tag{82}$$

This, together with the expanding subspace minimization property, shows that the polynomial P_k^* solves the problem

$$\min_{P_k \in \mathcal{P}_k} \|x_* - (x_0 + P_k(A) r_0)\|_A, \tag{83}$$

where, due to $r_0 = A(x_* - x_0)$, further holds

$$\|x_* - x_{k+1}\|_A = \|x_* - (x_0 + P_k(A) r_0)\|_A = \|\left[I - P_k(A)A\right](x_* - x_0)\|_A. \tag{84}$$

Since A is SPD, we have a orthonormal spectral decomposition $A = V \Lambda V^T$, where we denote the eigenvalues with $0 < \lambda_1 \le \dots \le \lambda_n$ and associated orthonormal eigenvectors with v_i, $i = 1, \dots, n$, which form a basis of $\mathbb{R}^n$. Specifically, we can write the initial error as a linear combination of the eigenbasis, that is $x_* - x_0 = \sum_{i=1}^n \xi_i v_i$ with some coefficients ξ_i, $i = 1, \dots, n$. Eigenvectors of A are also eigenvectors of $P_k(A)$ and we have $P_k(A) v_i = P_k(\lambda_i) v_i$, $i = 1, \dots, n$. We therefore have

$$x_* - x_{k+1} = \sum_{i=1}^n \left[1 - \lambda_i P_k^*(\lambda_i)\right] \xi_i v_i, \tag{85}$$

and due to the orthonormality of the v_i, i.e. $v_i^T v_j = \delta_{ij}$, we get for the respective energy norm of the error

$$\|x_* - x_{k+1}\|_A^2 = \sum_{i=1}^n \lambda_i \left[1 - \lambda_i P_k^*(\lambda_i)\right]^2 \xi_i^2 = \min_{P_k \in \mathcal{P}_k} \sum_{i=1}^n \lambda_i \left[1 - \lambda_i P_k(\lambda_i)\right]^2 \xi_i^2. \tag{86}$$

Extracting the polynomial expression finally leads to

$$\|x_* - x_{k+1}\|_A^2 \le \min_{P_k \in \mathcal{P}_k} \max_{1 \le i \le n} \left[1 - \lambda_i P_k(\lambda_i)\right]^2 \left(\sum_{j=1}^n \lambda_j \xi_j^2\right) = \min_{P_k \in \mathcal{P}_k} \max_{1 \le i \le n} \left[1 - \lambda_i P_k(\lambda_i)\right]^2 \|x_* - x_0\|_A^2, \tag{87}$$

which concludes the proof. □

In some cases it is possible to estimate the nonnegative factor $\min_{P_k \in \mathcal{P}_k} \max_{1 \le i \le n} \left[1 - \lambda_i P_k(\lambda_i)\right]^2$ to quantify the rate of convergence. An interesting case occurs if the spectrum of A consists of less than n distinct eigenvalues.

Theorem 1.22. *Let A be SPD with $r \le n$ distinct eigenvalues. Then the CG method produces the exact solution in at most r iterations.*

Proof. Let us denote the eigenvalues of A with $\lambda_1, \dots, \lambda_n$ taking on the r distinct values $0 < \tau_1 < \dots < \tau_r$. Consider the polynomial $Q_r(\lambda) \in \mathcal{P}_r$ defined as

$$Q_r(\lambda) = \frac{(-1)^r}{\tau_1 \tau_2 \cdots \tau_r} (\lambda - \tau_1)(\lambda - \tau_2) \cdots (\lambda - \tau_r). \tag{88}$$

We have $Q_r(\lambda_i) = 0$, $i = 1, \dots, n$ and $Q_r(0) = 1$, and hence $1 - Q_r(\lambda) \in \mathcal{P}_r$ with root at $\lambda = 0$. By polynomial division we can define the polynomial with degree $r - 1$

$$\widetilde{P}_{r-1} = \frac{1 - Q_r(\lambda)}{\lambda} \in \mathcal{P}_{r-1}. \tag{89}$$

For the convergence factor in Lemma 1.21 we get with $k = r - 1$

$$0 \leq \min_{P_{r-1} \in \mathcal{P}_{r-1}} \max_{1 \leq i \leq n} \left[1 - \lambda_i P_{r-1}(\lambda_i)\right]^2 \leq \max_{1 \leq i \leq n} \left[1 - \lambda_i \widetilde{P}_{r-1}(\lambda_i)\right]^2 = \max_{1 \leq i \leq n} Q_r(\lambda_i)^2 = 0. \tag{90}$$

Thus, from Lemma 1.21 we get $\|x_* - x_r\|_A^2 = 0$. □

A similar behavior can be expected if A has clustered eigenvalues, see Fig. 8.

There also holds a similar theorem as Th. 1.11 for the SD method, but only the square root of the condition number occurs in the upper bound:

Theorem 1.23 (Proof [1]). *The CG method applied to the SPD system $Ax = b$ yields the following estimate for the error $\varepsilon_k := x_* - x_k$ in the energy norm*

$$\|\varepsilon_k\|_A \leq 2\Big(\frac{\sqrt{\kappa_2(A)} - 1}{\sqrt{\kappa_2(A)} + 1}\Big)^k \|\varepsilon_0\|_A \,. \tag{91}$$

□

Although this is an improvement over Th. 1.11 for the steepest descent method, Th. 1.23 might be still too pessimistic. A major difference over SD is the fact that convergence of CG depends on the whole spectrum of A (not only the extreme eigenvalues). In fact, as Th. 1.22 shows, the CG method converges in at most r steps if A has $r < n$ distinct eigenvalues. Similar behavior occurs if there are clusters of eigenvalues, see Fig. 8.

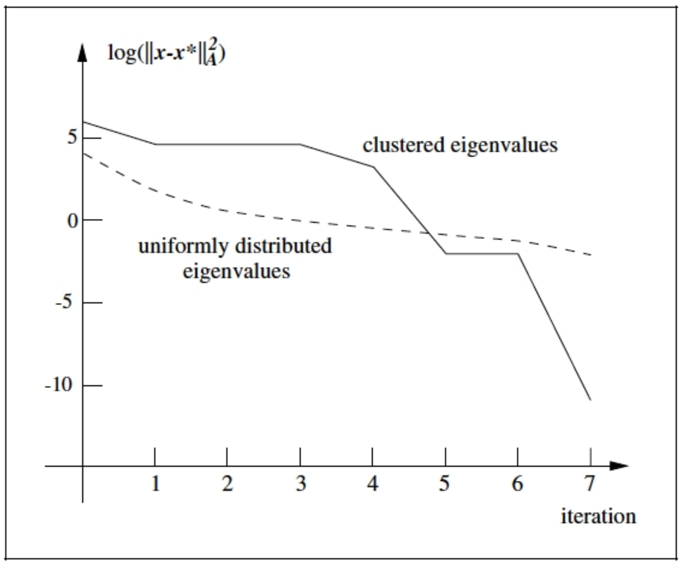


Figure 8: Convergence example of CG for uniformly distributed and clustered spectrum, taken from [11].

1.6 Krylov subspace methods

1.6.1 Krylov subspace projection

We now turn to the general case where $A \in \mathbb{R}^{n \times n}$ is an arbitrary nonsingular matrix.

The basic concept that we will use is a projection property called *Petrov-Galerkin orthogonality*:
Assume we have two subspaces of $\mathbb{R}^n$ that we denote with $\mathcal{K}$ and $\mathcal{L}$. These linear subspaces might be of reduced dimensionality, e.g. $m < n$. Starting from an initial vector $x_0 \in \mathbb{R}^n$ we want to define an

For the convergence factor in Lemma 1.21 we get with $k = r - 1$

$$0 \leq \min_{P_{r-1} \in \mathcal{P}_{r-1}} \max_{1 \leq i \leq n} \left[1 - \lambda_i P_{r-1}(\lambda_i)\right]^2 \leq \max_{1 \leq i \leq n} \left[1 - \lambda_i \widetilde{P}_{r-1}(\lambda_i)\right]^2 = \max_{1 \leq i \leq n} Q_r(\lambda_i)^2 = 0. \tag{90}$$

Thus, from Lemma 1.21 we get $\|x_* - x_r\|_A^2 = 0$. □

A similar behavior can be expected if A has clustered eigenvalues, see Fig. 8.

There also holds a similar theorem as Th. 1.11 for the SD method, but only the square root of the condition number occurs in the upper bound:

Theorem 1.23 (Proof [1]). *The CG method applied to the SPD system $Ax = b$ yields the following estimate for the error $\varepsilon_k := x_* - x_k$ in the energy norm*

$$\|\varepsilon_k\|_A \leq 2\Big(\frac{\sqrt{\kappa_2(A)} - 1}{\sqrt{\kappa_2(A)} + 1}\Big)^k \|\varepsilon_0\|_A \,. \tag{91}$$

□

Although this is an improvement over Th. 1.11 for the steepest descent method, Th. 1.23 might be still too pessimistic. A major difference over SD is the fact that convergence of CG depends on the whole spectrum of A (not only the extreme eigenvalues). In fact, as Th. 1.22 shows, the CG method converges in at most r steps if A has $r < n$ distinct eigenvalues. Similar behavior occurs if there are clusters of eigenvalues, see Fig. 8.

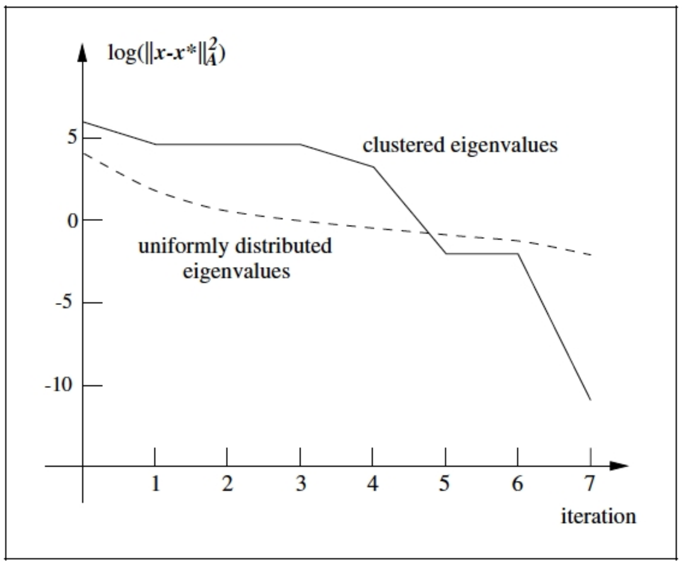


Figure 8: Convergence example of CG for uniformly distributed and clustered spectrum, taken from [11].

1.6 Krylov subspace methods

1.6.1 Krylov subspace projection

We now turn to the general case where $A \in \mathbb{R}^{n \times n}$ is an arbitrary nonsingular matrix.

The basic concept that we will use is a projection property called *Petrov-Galerkin orthogonality*:
Assume we have two subspaces of $\mathbb{R}^n$ that we denote with $\mathcal{K}$ and $\mathcal{L}$. These linear subspaces might be of reduced dimensionality, e.g. $m < n$. Starting from an initial vector $x_0 \in \mathbb{R}^n$ we want to define an

updated approximation $\widetilde{x} = x_0 + \delta$ with update δ in the space $\mathcal{K}$ in such a way that the new residual is perpendicular to the (test) space $\mathcal{L}$, that is:

$$\text{Find} \quad \widetilde{x} = x_0 + \delta,\ \delta \in \mathcal{K} \quad \text{such that } \ b - A\widetilde{x} = r_0 - A\delta \perp \mathcal{L}. \tag{92}$$

Fig. 9 shows this process of subspace projection.

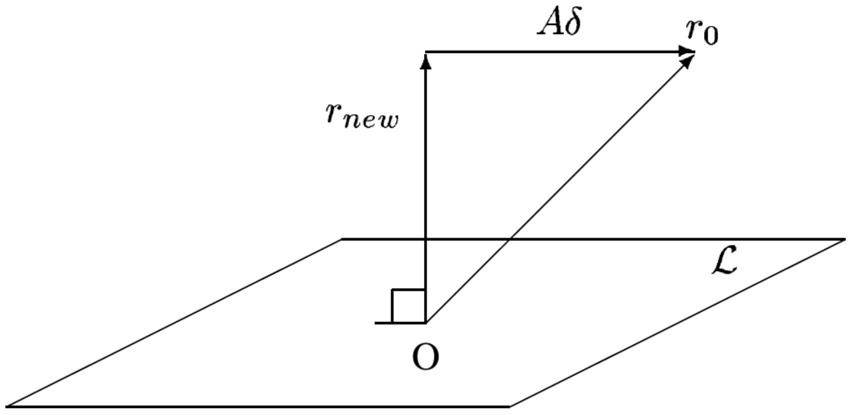


Figure 9: Subspace projection, taken from [1].

Krylov subspace methods aim at finding a good approximation x_m of the solution of the linear system $Ax = b$ in the Krylov subspace $\mathcal{K}_m$ ($m \ll n$), i.e., $x_m \in x_0 + \mathcal{K}_m$ which satisfies the orthogonality

$$v^T(b - Ax_m) = 0 \quad \text{for all } v \in \mathcal{L}_m, \tag{93}$$

where $\mathcal{L}_m$ is called the *test space*, while $\mathcal{K}_m = \mathcal{K}_m(A, r_0)$ is the *ansatz* or *solution space*. Krylov subspace methods are linear model reduction methods for the problem $Ax = b$ that aim at reducing computational effort by utilizing the lower dimensionality of the involved subspaces. In fact, we will see that the computational effort of one projection step (iteration) is reduced to the solution of a system of smaller size.

The different Krylov subspace methods differ in the choice of the test space $\mathcal{L}_m$. There are three typical choices:

1. $\mathcal{L}_m = \mathcal{K}_m$ (*'Full orthogonalization methods' e.g. CG, D-Lanczos*)

$$r_m \perp \mathcal{K}_m \quad (\Leftrightarrow A\varepsilon_m \perp \mathcal{K}_m). \tag{94}$$

   For symmetric A

$$\varepsilon_m \perp A\mathcal{K}_m \quad (\varepsilon_m \perp_A \mathcal{K}_m \quad \text{if } A \text{ SPD}). \tag{95}$$

   In the SPD case the orthogonality requirement has a unique solution satisfying [1]

$$\|x_m - x_*\|_A = \min_{x \in x_0 + \mathcal{K}_m(A, r_0)} \|x - x_*\|_A. \tag{96}$$

2. $\mathcal{L}_m = A\mathcal{K}_m$ (*'Generalized minimal residual methods': GMRES*)

$$r_m \perp A\mathcal{K}_m \quad (\Leftrightarrow A\varepsilon_m \perp A\mathcal{K}_m). \tag{97}$$

3. $\mathcal{L}_m = \mathcal{K}_m(A^T, \widetilde{r}_0)$ for some $\widetilde{r}_0$ (*Biorthogonal methods, e.g. BiCG*).

Our focus here shall be on the GMRES method ($\mathcal{L}_m = A\mathcal{K}_m$).

Lemma 1.24. *The orthogonality* (97) *is equivalent to*

$$\|r_m\|_2 = \|b - Ax_m\|_2 = \min_{x \in x_0 + \mathcal{K}_m(A, r_0)} \|b - Ax\|_2, \tag{98}$$

which motivates the term minimal residual method.

Proof. To make (98) plausible imagine an arbitrary $x \in x_0 + \mathcal{K}_m$, where x_0 and hence r_0 are fixed. The norm of a related (generic) residual vector $r \in r_0 + A\mathcal{K}_m = r_0 + \mathcal{L}_m$ is indeed minimized iff r is perpendicular to $\mathcal{L}_m = A\mathcal{K}_m$: Write $r = r_0 + \ell = (r_0^{\|_{\mathcal{L}_m}} + \ell) + r_0^{\perp_{\mathcal{L}_m}}$, $\ell \in \mathcal{L}_m$ (the superscript $\|_{\mathcal{L}_m}$ and $\perp_{\mathcal{L}_m}$ denote parallel and perpendicular component with respect to $\mathcal{L}_m$, respectively) then $\|r\|_2^2 = \|r_0^{\|_{\mathcal{L}_m}} + \ell\|_2^2 + \|r_0^{\perp_{\mathcal{L}_m}}\|_2^2$ is minimized for $\ell = -r_0^{\|_{\mathcal{L}_m}}$. Hence, a minimal residual yields $r = r_0 - r_0^{\|_{\mathcal{L}_m}} = r_0^{\perp_{\mathcal{L}_m}} \in \mathcal{L}_m^{\perp}$ and also conversely, r perpendicular to the test space $\mathcal{L}_m$ yields a minimal residual norm. □

The characterization (98) as minimization problem would suggest to calculate the iterate x_m with the ansatz

$$x_m = x_0 + K_m c, \tag{99}$$

where $K_m = [r_0|Ar_0|\cdots|A^{m-1}r_0] \in \mathbb{R}^{n\times m}$ is the Krylov matrix and $c \in \mathbb{R}^m$ the corresponding coefficient vector. From the ansatz (99) one gets $r_m = r_0 - AK_m c$ and hence, the minimum condition (98) yields

$$c = \arg\min_{d \in \mathbb{R}^m} \|r_0 - AK_m d\|_2. \tag{100}$$

In general, this least squares problem will be ill-conditioned (nearly singular Krylov matrix), and hence the method will be numerically unstable. This is the reason why orthonormalization procedures for the Krylov spaces have to be incorporated into the methods.

Definition 1.25 (Steps in a Krylov subspace method). *A Krylov subspace algorithm consists of the following steps:*

- *Construct the (Arnoldi) matrix $V_m \in \mathbb{R}^{n\times m}$ whose columns form an orthonormal basis (i.e. $V_m^T V_m = I_m$) of the Krylov space $\mathcal{K}_m$. Compare with the forthcoming Ch. 1.6.2.*
- *Construct the matrix $W_m \in \mathbb{R}^{n\times m}$ whose columns form a basis of the test space $\mathcal{L}_m$. For $\mathcal{L}_m = A\mathcal{K}_m$ this is simply $W_m = AV_m$, which is then computed in parallel in step one.*
- *Make the ansatz $x_m = x_0 + V_m y_m$, where $y_m \in \mathbb{R}^m$ is the vector of weights to be determined.*
- *Require the Petrov-Galerkin orthogonality*

$$W_m^T(Ax_m - b) = 0 \Leftrightarrow W_m^T A V_m y_m = W_m^T r_0. \tag{101}$$

The iterate takes the form

$$x_m = x_0 + V_m(W_m^T A V_m)^{-1} W_m^T r_0 \tag{102}$$

Note that the matrix $W_m^T A V_m$ is only of size $m \times m$, which is OK for $m \ll n$.

Remark 1.26. Also note that for $W_m = AV_m$ (GMRES) the linear system (101) has the form $W_m^T W_m y_m = W_m^T r_0$, which are the normal equations of the linear least squares problem

$$y_m = \arg\min_{y \in \mathbb{R}^m} \|r_0 - AV_m y\|_2. \tag{103}$$

Note that from the ansatz $x_m = x_0 + V_m y_m$ one gets $Ax_m = Ax_0 + AV_m y_m \approx Ax_0 + r_0 = b$. □

In the following we will describe the involved orthogonalization procedure (Arnoldi/Lanczos) and the computational details of the generalized minimal residual (GMRES) method in a little more detail.

1.6.2 The Arnoldi and Lanczos Procedure: Orthogonalization of $\mathcal{K}_m$

We consider an arbitrary matrix $A \in \mathbb{R}^{n\times n}$ and a given residual vector $r_0 \in \mathbb{R}^n$ with the corresponding Krylov subspace ($m \le n$)

$$\mathcal{K}_m = \mathcal{K}_m(A, r_0) = \operatorname{span}\{r_0, Ar_0, \dots, A^{m-1}r_0\}. \tag{104}$$

Let the matrix

$$K_m = [r_0|Ar_0|\cdots|A^{m-1}r_0] \in \mathbb{R}^{n\times m} \tag{105}$$

the corresponding *Krylov matrix.*
The Arnoldi procedure is based on the Gram-Schmidt algorithm to construct an orthonormal set/basis of the column vectors of K_m. The basic idea is to start with $v_1 = r_0/\|r_0\|_2$, then in the j-th step multiply the current Arnoldi vector v_j with A and orthonormalize Av_j against all previous Arnoldi vectors. If the plain Gram-Schmidt procedure is used for orthonormalization, this leads to the following steps

$$\begin{aligned}
& & v_1 &= r_0/\|r_0\|_2\\
w_1 &= Av_1 - (v_1^T Av_1)v_1, & v_2 &= w_1/\|w_1\|_2\\
w_2 &= Av_2 - (v_1^T Av_2)v_1 - (v_2^T Av_2)v_2, & v_3 &= w_2/\|w_2\|_2\\
& & &\vdots\\
w_m &= Av_m - (v_1^T Av_m)v_1 - \ldots - (v_m^T Av_m)v_m, & v_{m+1} &= w_m/\|w_m\|_2
\end{aligned} \tag{106}$$

This generates the Arnoldi matrix

$$V_m = [v_1|v_2|\cdots|v_m] \in \mathbb{R}^{n\times m} \tag{107}$$

and the upper Hessenberg matrix $\hat{H}_m \in \mathbb{R}^{(m+1)\times m}$

$$\hat{H}_m = \begin{bmatrix} h_{11} & h_{12} & h_{13} & \dots & h_{1m}\\ h_{21} & h_{22} & h_{23} & \dots & h_{2m}\\ & h_{32} & h_{33} & & h_{3m}\\ & & \ddots & \ddots & \vdots\\ & & & h_{m(m-1)} & h_{mm}\\ & & & & h_{(m+1)m}\end{bmatrix} = \begin{bmatrix} v_1^TAv_1 & v_1^TAv_2 & v_1^TAv_3 & \dots & v_1^TAv_m\\ v_2^TAv_1 & v_2^TAv_2 & v_2^TAv_3 & \dots & v_2^TAv_m\\ & v_3^TAv_2 & v_3^TAv_3 & & v_3^TAv_m\\ & & \ddots & \ddots & \vdots\\ & & & v_m^TAv_{m-1} & v_m^TAv_m\\ & & & & v_{m+1}^TAv_m\end{bmatrix}.$$

Note that the off-diagonal elements of $\hat{H}_m$ are calculated in the normalization step of (106) by calculating $\|w_m\|_2 = v_{m+1}^T w_m = v_{m+1}^T Av_m$ using the mutual orthogonality of the Arnoldi vectors, i.e., $v_{m+1}^T v_j$, $j = 1, \dots, m$.
If a breakdown occurs in the m-th step, $w_m = 0$ is defined but not v_{m+1} and so the algorithm stops; in this case the element $h_{m+1,m} = 0$.

Lemma 1.27. *If the Arnoldi iteration* (106) *does not terminate prematurely, the vectors $v_1, \dots, v_m$ form an orthonormal basis of the Krylov space $\mathcal{K}_m$, i.e., $V_m^T V_m = I_m$. Further, the operator $P_m = V_m V_m^T$ is the orthogonal projection onto $\mathcal{K}_m$.* □

In any case, we have from (106) that

$$Av_j = w_j + \sum_{i=1}^{j} h_{ij}v_i, \tag{108}$$

which means in matrix notation [15]

[15]Note for the last identity that $V_{m+1}\hat{H}_m = [V_m|v_{m+1}]\begin{bmatrix} H_m \\ (v_{m+1}^T Av_m)e_m^T\end{bmatrix} = V_m H_m + (v_{m+1}^T Av_m)v_{m+1}e_m^T$ and due to orthogonality $v_{m+1}^T Av_m = \|w_m\|_2$ (use (106)).

$$AV_m = V_m H_m + w_m e_m^T \ \ (= V_{m+1}\hat{H}_m), \tag{109}$$

with $H_m \in \mathbb{R}^{m\times m}$ is the square Hessenberg matrix obtained from $\hat{H}_m$ by removing the last row and e_m denotes the m-th unit vector.

Lemma 1.28 (Arnoldi identity). *From* (109) *we conclude that*

$$V_m^T A V_m = H_m + \underbrace{V_m^T w_m}_{=0} e_m^T = H_m. \tag{110}$$

Proof. We have $V_m^T A V_m = H_m + \underbrace{V_m^T w_m}_{=0} e_m^T = H_m$. □

Theorem 1.29 ([3]). *The Arnoldi procedure generates a reduced QR-decomposition of the Krylov matrix K_m in the form*

$$K_m = V_m R_m \tag{111}$$

with the Arnoldi matrix V_m from (107) *and a triangular matrix $R_m \in \mathbb{R}^{m\times m}$. Further, the Hessenberg matrix H_m is an projection of A onto the Krylov subspace spanned by the columns of K_m, i.e.,*

$$H_m = V_m^T A V_m. \tag{112}$$

□

Remark 1.30. "H_m is a projected version of A": If we consider $x \in \mathcal{K}_m$ expressed in the basis V_m, i.e., $x = V_m y$, then the orthogonal projection $V_m V_m^T$ onto $\mathcal{K}_m$ of Ax is

$$V_m V_m^T A x = V_m V_m^T A V_m y = V_m H_m y, \tag{113}$$

that is, the coefficients (or coordinates w.r.t. the basis V_m) of the projected image is given by $H_m y$.
If x is not yet in $\mathcal{K}_m$, we can simply consider the projected version $A_m := P_m A P_m = V_m H_m V_m^T$ with the projection $P_m = V_m V_m^T$: $A_m x = V_m H_m (V_m^T x) = V_m H_m y$, where $y = V_m^T x$ are the coordinates of the projection image of x, i.e., $P_m x = V_m(V_m^T x) = V_m y$, w.r.t. the basis V_m. □

Remark 1.31. Since H_m is a projection of A the eigenvalues of H_m are good estimates for (some of) those of A. They are called Arnoldi estimates or Ritz values, see Ch. **??**. □

Remark 1.32. The Arnoldi procedure is connected with polynomial approximation. The problem of finding a monic polynomial p_m of degree m [16] such that the norm $\|p_m(A)r_0\|_2$ becomes minimal, has a unique solution in the case where K_m has full rank, namely the characteristic polynomial of H_m, i.e., $\chi(H_m)$. Further, for $x \in \mathcal{K}_m$ one can show

$$V_m \chi(H_m) V_m^T x = \chi(A)x = 0. \tag{114}$$

This shows that $V_m \chi(H_m) V_m^T$ behaves like the characteristic polynomial of A when restricted to $\mathcal{K}_m$. Therefore the eigenvalues of H_m (Ritz values) can be expected to be good approximations to some of the eigenvalues of A. In fact, one can show that each eigenvalue of H_m is also an eigenvalue of $A_m := V_m H_m V_m^T = P_m A P_m$ with the projection $P_m = V_m V_m^T$ and all other eigenvalues of A_m are zero. □

The Arnoldi procedure is usually implemented in a numerically stable way known as *Modified Gram-Schmidt algorithm* (MGS).
The Arnoldi iteration in the modified Gram-Schmidt variant is summarized in Alg. 5.

[16] A polynomial where the highest coefficient is one, i.e., polynomials of the form $p_m(z) = c_0 + c_1 z + \ldots + c_{m-1}z^{m-1} + z^m$.

Algorithm 5: Arnoldi iteration (Modified Gram-Schmidt variant)

```
Data: Matrix $A$, vector $r_0$, number $m$
Result: Arnoldi vectors $v_1, \dots, v_m$, Upper Hessenberg matrix $\hat{H}_m = (h_{ij})$
Initialization: $v_1 \leftarrow r_0/\|r_0\|_2$
for $j = 1 \dots m$ do
    $w_j \leftarrow Av_j$
    for $i = 1 \dots j$ do
        $h_{ij} \leftarrow v_i^T w_j$
        $w_j \leftarrow w_j - h_{ij} v_i$
    end
    $h_{(j+1),j} \leftarrow \|w_j\|_2$
    if $h_{j+1,j} = 0$ then
        Stop
    end
    $v_{j+1} \leftarrow w_j / h_{(j+1),j}$
end
```

In the case where A is symmetric, also the upper Hessenberg matrix H_m is symmetric ($H_m = V_m^T A V_m = V_m^T A^T V_m = H_m^T$) and therefore a symmetric tridiagonal matrix

$$T_m = \begin{bmatrix} \alpha_1 & \beta_2 & & & \\ \beta_2 & \alpha_2 & \beta_3 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & \beta_m \\ & & & \beta_m & \alpha_m \end{bmatrix}.$$

As a consequence, the inner loop in Alg. 5 simplifies, see Alg. 6. This is called the *Lanczos iteration.*

Algorithm 6: Lanczos iteration

```
Data: Matrix $A$ symmetric, vector $r_0$, number $m$
Result: Arnoldi vectors $v_1, \dots, v_m$, tridiagonal matrix $T_m$
Initialization: $\beta_1 \leftarrow 0$, $v_0 \leftarrow 0, v_1 \leftarrow r_0/\|r_0\|_2$
for $j = 1 \dots m$ do
    $w_j \leftarrow Av_j - \beta_j v_{j-1}$
    $\alpha_j \leftarrow v_j^T w_j$
    $w_j \leftarrow w_j - \alpha_j v_j$
    $\beta_{j+1} \leftarrow \|w_j\|_2$
    if $\beta_{j+1} = 0$ then
        Stop
    end
    $v_{j+1} \leftarrow w_j / \beta_{j+1}$
end
```

1.6.3 Generalized minimal residual method (GMRES)

Here $A \in \mathbb{R}^{n \times n}$ is a (nonsingular) square matrix. As described in the beginning of this chapter the GMRES method requires the Petrov-Galerkin orthogonality $r_m \perp A\mathcal{K}_m$. This leads to the normal equations (see (101))

$$W_m^T W_m y_m = W_m^T r_0, \tag{115}$$

for $W_m = AV_m$ and the ansatz for the approximate solution $x_m = x_0 + V_m y_m$. The normal equations (115) can be simplified in a practically more convenient way:

Lemma 1.33. *The normal equations* (115) *in the computation of a GMRES iterate can be reformulated as*

$$\hat{H}_m^T \hat{H}_m y_m = \hat{H}_m^T \, \|r_0\|_2 \, e_1, \tag{116}$$

where e_1 is the unit vector $e_1 = (1, 0, 0, \dots, 0)^T$. Eqn. (116) *has a unique solution under the "no-break-down assumption": $h_{j+1,j} \neq 0, \ j = 1, \dots, m-1$.*

Proof. We recall the identity (109) which yields

$$W_m = AV_m = V_m H_m + w_m e_m^T = V_{m+1} \hat{H}_m. \tag{117}$$

Let us assume that the Arnoldi iteration does not break down, that is $h_{j+1,j} \neq 0, \ j = 1, \dots, m-1$, which implies that $\hat{H}_m$ has full column rank m. Now note that

$$W_m^T W_m = \hat{H}_m^T \underbrace{V_{m+1}^T V_{m+1}}_{I_{m+1}} \hat{H}_m = \hat{H}_m^T \hat{H}_m, \tag{118}$$

is non-singular due to our "no-break-down assumption". Also the r.h.s. of (115) simplifies due to the orthogonality of $v_1, \dots, v_{m+1}$ to

$$W_m^T r_0 = \hat{H}_m^T V_{m+1}^T r_0 = \hat{H}_m^T \underbrace{(v_1^T r_0)}_{=\|r_0\|_2} e_1 = \hat{H}_m^T \, \|r_0\|_2 \, e_1. \tag{119}$$

Thus, the resulting normal equations in the GMRES method take the form

$$\hat{H}_m^T \hat{H}_m y_m = \hat{H}_m^T \, \|r_0\|_2 \, e_1. \tag{120}$$

They are uniquely solvable [17] under the "no-break-down assumption"[18]. □

The normal equations (116) are equivalent to the linear least squares problem

$$y_m = \arg\min_{y \in \mathbb{R}^m} \left\| e_1 \, \|r_0\|_2 - \hat{H}_m y \right\|_2. \tag{121}$$

The GMRES algorithm is given in Alg. 7; it consists of the Arnoldi iteration followed by the solution of the linear least squares problem (121).

Algorithm 7: GMRES algorithm

```
Data: Matrix $A \in \mathbb{R}^{n \times n}$, r.h.s. $b \in \mathbb{R}^n$, initial vector $x_0 \in \mathbb{R}^n$
Result: $x \in \mathbb{R}^n$ approximate solution
Initialization: $m \leftarrow 1, r_0 \leftarrow b - Ax_0$
while convergence criterion not satisfied do
        Compute $V_m, \hat{H}_m$ from Arnoldi iteration Alg. 5
        $y_m \leftarrow \arg\min_{y \in \mathbb{R}^m} \|e_1 \|r_0\|_2 - \hat{H}_m y\|_2$        (Linear least squares problem)
        $x_m \leftarrow x_0 + V_m y_m$
        $m \leftarrow m + 1$
end
$x \leftarrow x_{m-1}$
```

Remark 1.34. (Efficient Updates) Note that in Alg. 7 the new Arnoldi vector v_m is obtained from the previously computed $v_1, \dots, v_{m-1}$. Also the matrix $\hat{H}_m$ is only updated form the previously computed $\hat{H}_{m-1}$ by adding a column and the element $h_{m+1,m}$. Similarly, the QR-decomposition of $\hat{H}_m$, which is used for solving the linear least squares problem, can be efficiently updated. □

[17]Because of the Hessenberg structure of $\hat{H}_m$ the QR-decomposition is a good choice to solve the underlying linear least squares problem, rather than applying the more expensive Cholesky-decomposition to the normal equations.

[18]The system is solvable until a break-down occurs.

Remark 1.35. **(Restarts)** To control the required memory the GMRES algorithm is usually restarted after a maximum dimension $m = m_{max}$ is reached. In this *restarted version* the approximation $x_{m_{max}}$ is used as the new starting vector while all the previously computed information is deleted. □

Remark 1.36. **(Truncated (Quasi) GMRES)** An incomplete version of GMRES is derived by incomplete orthogonalization. Only the k previous Arnoldi vectors are kept in memory (similar idea as in *limited memory* quasi Newton methods in numerical optimization). For further details see [1, 11]. □

Example 1.37. Consider the Poisson equation in three dimensions on $\Omega = (-a, a)^3$

$$\begin{aligned} -\Delta u &= f \equiv 1 \\ u_{|\partial\Omega} &= 0, \end{aligned} \tag{122}$$

and a finite difference discretization with uniform mesh size $h = 2a/30$ with resulting sparse system matrix

$$A = K_{1d} \otimes I \otimes I + I \otimes K_{1d} \otimes I + I \otimes I \otimes K_{1d} \in \mathbb{R}^{30^3 \times 30^3}. \tag{123}$$

Fig. 10 shows the performance of GMRES for different maximum Krylov space dimension and the CG method (as benchmark, since it is the ideal method for this SPD system).

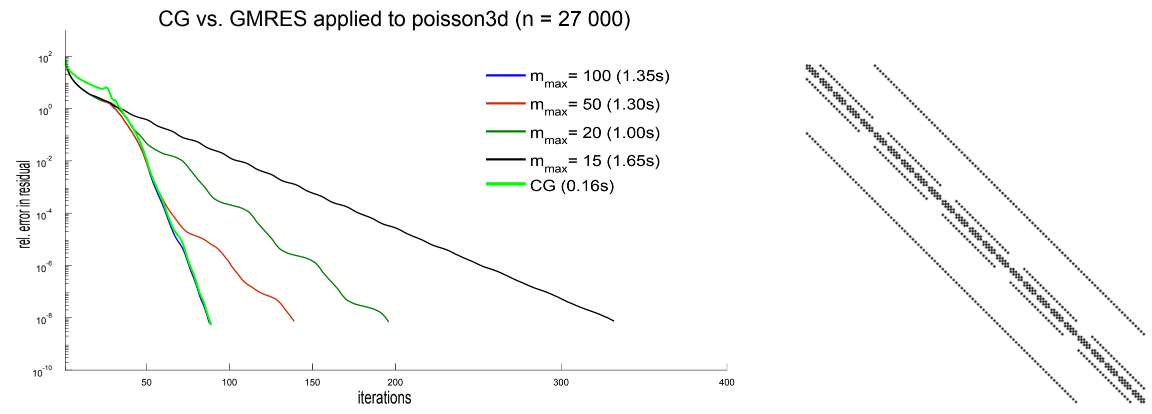


Figure 10: GMRES with different maximum Krylov space dimension and the CG method applied to (122). Right: Sparsity pattern of A.

On convergence of GMRES.

The following convergence result for GMRES is obtained from polynomial representation of the Krylov space $\mathcal{K}_m = \{x_0 + p(A)r_0 : p \in \mathcal{P}_{m-1}\}$ and the rather restrictive assumption of diagonalizable A.

Theorem 1.38 (Proof [3]). *Let A be diagonalizable: $A = X\Lambda X^{-1}$, with $\Lambda = diag(\lambda_1, \ldots, \lambda_n)$ the diagonal matrix of eigenvalues. Let further* [19]

$$\epsilon_m := \min_{q \in \mathcal{P}_m,\, q(0)=1} \max_{i=1,\ldots,n} |q(\lambda_i)|. \tag{124}$$

Then there holds for the 2-norm of the m-th residual

$$\|r_m\|_2 \leq \epsilon_m \, \kappa_2(X) \, \|r_0\|_2. \tag{125}$$

Remark 1.39. Note that the condition number $\kappa_2(X)$ can be large even if that of A is small, e.g. if A is close to non-diagonalizable. Also, X is not known in practice, so this result is of limited practical value.
Preconditioning does not have such a clear positive effect as in the CG method, e.g. can also have adverse effects on $\kappa_2(X)$. However, 'bunching eigenvalues' is still a valid strategy since ϵ_m can be bounded by a complex analogue of the Chebyshev min-max theorem [1].

[19]Note that the error of the m-th iteration has a representation $\varepsilon_m = \varepsilon_0 + p_{m-1}(A)r_0 = (I - p_{m-1}(A)A)\,\varepsilon_0 =: q_m(A)\varepsilon_0$ with $q_m \in \mathcal{P}_m$ and $q_m(0) = 1$.