Zentrum für Technomathematik
Fachbereich 3 – Mathematik und Informatik

A Short Introduction to Iterative Methods for Large Linear Systems

Angelika Bunse-Gerstner

Report 02–11

Berichte aus der Technomathematik

Report 02–11 **November 2002**

A SHORT INTRODUCTION TO ITERATIVE METHODS FOR LARGE LINEAR SYSTEMS

Angelika Bunse-Gerstner
Zentrum für Technomathematik
Fachbereich Mathematik/Informatik
Universität Bremen
Bremen, Germany
`Bunse-Gerstner@math.uni-Bremen.de`

Morelia, Mexico
March 2002

Contents

Chapter 1

Introduction

In scientific computing most computational time is spent with solving systems of linear equations. They occur for example as subproblems in numerical methods for the solution of partial differential equations and integral equations. These linear systems arising in applications are usually of rather large dimension such that solving them with the standard Gaußian elimination method is impossible.

These notes give a short introduction into the basic ideas of a class of iterative methods for solving large linear systems, which are often called Krylov subspace methods. They are most popular and frequently used for practical problems, because they are easy to implement and have low storage and CPU time requirements. This introduction is based merely on well-known and some new linear and numerical linear algebra results and allows an easy understanding of general construction principles and the applicability of the methods.

1.1 Large Linear Systems

Consider the problem of solving

$$Ax = b \tag{1.1}$$

for $A \in \mathbb{R}^{N\times N}$, $b \in \mathbb{R}^N$ with a large dimension N, e.g. $N_1 = 250000$ or $N_2 = 700000$.

For Gaußian elimination, the standard method to compute the solution x of a linear system, the storage requirement is N^2, e.g. $6.25 * 10^{10}$ for N_1 and $4.9 * 10^{11}$ for N_2. The number of floating point operations needed in this computation is essentially $\frac{8}{3}N^3$, e.g. $4.17 * 10^{16}$ for N_1 and $9.15 * 10^{17}$ for N_2.
If we have a computer which can perform 10^8 floating point operations per second, which is a fast computer for today's standards, then for a system of dimension $N_1 = 250000$ our computer would need a bit more than 13 years full time running to compute the solution in this way, and for $N_2 = 700000$ it would take approximately 291 years of computing time. Even if we could overcome the problem with the large storage requirement for our fast computer, we would not live long enough to ever receive the solution in the second case. Thus we obviously need a different approach to compute the solution x of (1.1) for large N.

There are two principal ways of solving (1.1), where in all of the following we will assume that the system matrix A is nonsingular. In a "direct method" A and b are transformed stepwise, i.e. they are in general multiplied by elementary matrices, such that after a finite number of operations the method produces (in exact arithmetic) the exact solution. Gaußian elimination is such a method.

If (1.1) comes from the discretization of a partial differential equation, then the matrix A is typically large and sparse, which means that in addition to the dimension of A being large, also most entries of A are zero. Often the number of nonzero entries in A is then only of order N. The simplest and best known example is the linear system that results from the standard five point discretization of $-\Delta u = f$ on $\Omega =]0,1[\times]0,1[$ with Dirichlet boundary conditions $u|_{\partial\Omega} = 0$ and a regular grid on Ω, which stems from equally spaced knots $0 < \frac{1}{n+1} < 2\frac{1}{n+1} < \ldots < n\frac{1}{n+1} < 1$ on $[0,1]$ in both directions. The unknowns in the discretized problem are the values of u at the n^2 inner grid points and if we arrange them row-wise in an n^2 dimensional vector x of unknowns then the n^2-dimensional matrix A is of the form

$$A = \begin{bmatrix} D & -I_n & 0 & 0 & \cdots & 0 \\ -I_n & D & -I_n & 0 & \cdots & 0 \\ 0 & -I_n & D & -I_n & & \\ 0 & 0 & -I_n & \ddots & \ddots & \vdots \\ \vdots & \vdots & & \ddots & & \\ 0 & 0 & \cdots & & -I_n & D \end{bmatrix}, \tag{1.2}$$

where

$$D = \begin{bmatrix} 4 & -1 & 0 & \cdots & 0 \\ -1 & 4 & -1 & \ddots & \vdots \\ 0 & -1 & 4 & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & -1 \\ 0 & \cdots & 0 & -1 & 4 \end{bmatrix}$$

, and I_n is the n-dimensional identity matrix.

With a reasonably fine discretization, i.e. a big n, we get a large but very sparse matrix A and the complete information about A is very easy to store. We would just store the numbers 4 and -1 together with the information about their positions in the matrix.

The sparsity pattern of a large matrix is often made visible by a "portrait" of the matrix, which is an $N \times N$ dimensional field in which the (j,j) - position is blank if $A(j,j) = 0$ and is dark otherwise. For our matrix A above we get for example for $n = 10$ the left hand side picture in Figure 1.1 below. There are 10 000 entries in the matrix but nz, the number of nonzero elements, is only 460.

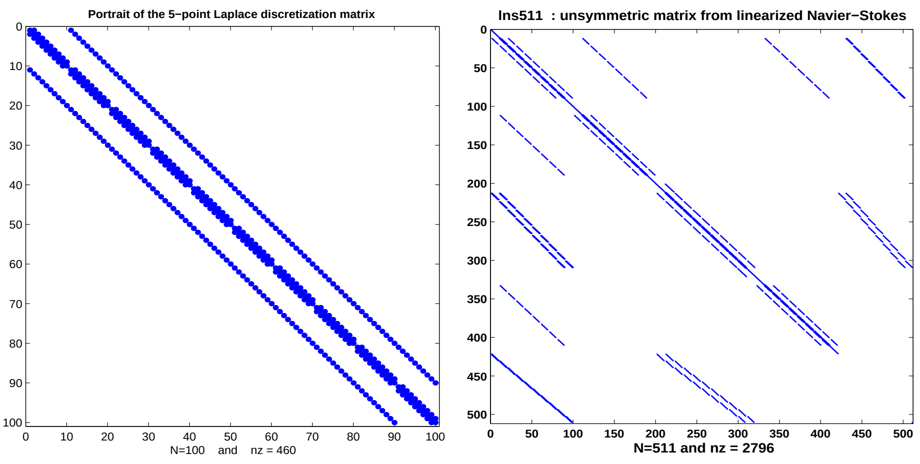


Figure 1.1: Portraits of sparse matrices

A good source for large, sparse matrices for test purposes is the Harwell-Boeing Sparse Matrix Collection, which one can get access to via

http://math.nist.gov/MatrixMarket/collections/hb.html.

The right hand side picture of Figure 1.1 is a portrait of a 511×511 unsymmetric matrix arising in the discretization of a Navier-Stokes problem, called lns511, from the Harwell-Boeing Collection.

If we now transform a large sparse matrix in the Gaußian elimination process, then nonzero elements will be created in places where A had zeros before. These new nonzero entries are called "fill" or "fill-in". In other words, in the LU-factorization $PA = LU$ which is computed in the Gaußian elimination process, the matrices L and U will have many more nonzero entries than the matrix A. For lns511 the following figure shows the portraits of the permuted version of L as it arises in Gaußian elimination with partial pivoting and of U. Note the difference of the number nz of nonzero entries in lns511 and its factors.

One approach to solve large, sparse linear systems consists, roughly speaking, in modifying Gaußian elimination in a sophisticated way, such that the amount of fill-in is small and an elimination process is possible. For these direct methods for large and sparse linear systems see e.g. [2], [7]. If N is very large, however, these methods reach their limits, because even though the fill-in is small compared to N^2, the number of nonzero entries gets too big to be handled.

If the system (1.1) comes from a discretization of an integral equation, e.g. if a partial differential equation is numerically solved by a boundary element method, then A is "dense", i.e. all or almost all entries of A are nonzero. Then a direct method cannot be applied at all. An alternative way to solve (1.1) is an iterative

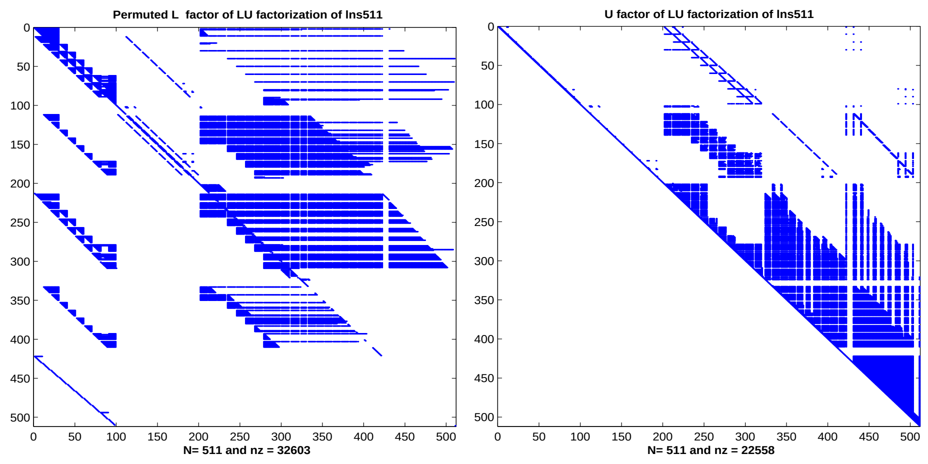


Figure 1.2: L U factors of the LU decomposition of lns511

method, where we start with an initial vector x_0 and compute vectors $\tilde{x}_1, \tilde{x}_2, \tilde{x}_3, \ldots$ such that $x_0+\tilde{x}_1, x_0+\tilde{x}_2, x_0+\tilde{x}_3, \ldots$ are (hopefully) better and better approximations of x. The computation of the correction vectors $\tilde{x}_j$ has to be such that the matrix A is not altered. Only matrix-vector products Aw for given vectors w should be used to determine the $\tilde{x}_j$, because this is in most cases reasonably cheap to compute, see for instance our example (1.2).

It is remarkable that Gauß himself, in 1823, already proposed an iterative method for solving linear systems [1]. At his time, however, $N = 40$ or $N = 50$ was considered to be large, because all computations had to be done by human beings. With the availabiltiy of computers iterative methods became more and more important, because increasing complexity of practical problems gave rise to larger and larger linear systems that had to be solved in the numerical methods for these problems.

Early iterative methods, sometimes called standard iterations, were the Jacobi- and Gauß-Seidel Iteration, Successive-Over-Relaxation (SOR), SSOR and Chebychev Semi-Iterative Method, see e.g. [10], [11] and [5]. The convergence of these methods is often slow and SOR, Chebychev Semi-Iteration and related methods depend on parameters that are sometimes difficult to choose. Other newer iterative methods like the multigrid methods are developed for situations in which knowledge of the underlying analytic problem can be used.

In this note we will confine ourselves to Krylov subspace methods in which no parameter determination is necessary and which work for very general problems. They are very easy to implement and are very frequently used in practice. There are still many open questions in this area and it is still a field of very active research.

This text gives a first introduction to the basic ideas of these methods.

1.2 Basic Ideas

Recall that our problem is to solve (1.1):

$$Ax = b,$$

$A \in \mathbb{R}^{N\times N}, b \in \mathbb{R}^N, N$ large and A nonsingular.

Let x_0 be an initial guess for the solution, which may come from some additional information we may have, or which may be the zero vector or a random vector. With $r_0 = b - Ax_0$, the "initial residual", and the solution $\tilde{x}$ of

$$A\tilde{x} = r_0 \tag{1.3}$$

we can correct x_0 to get the exact solution x:

$$A(x_0 + \tilde{x}) = Ax_0 + A\tilde{x} = b - r_0 + r_0 = b.$$

A first very general idea for approximating the solution of (1.3) is the following:

Choose a sequence of suitable subspaces $\mathcal{K}_l \subseteq \mathbb{R}^N$, for $l = 1, 2, 3, \dots$ with increasing dimension and approximate $\tilde{x}$, the solution of (1.3) by suitable elements $\tilde{x}_l$ of $\mathcal{K}_l$. We could think of choosing $\mathcal{K}_l$ such that

$$\mathcal{K}_1 \subseteq \mathcal{K}_2 \subseteq \mathcal{K}_3 \subseteq \dots. \tag{1.4}$$

In most cases we will study the residual vectors

$$r_l := r_0 - A\tilde{x}_l, \quad l = 1, 2, 3, \dots \tag{1.5}$$

and we want them to be small. Note that this is also the residual for the approximation $x_0 + \tilde{x}_l$ to the solution x of our original system, because

$$b - A(x_0 + \tilde{x}_l) = b - Ax_0 - A\tilde{x}_l = r_0 - A\tilde{x}_l = r_l. \tag{1.6}$$

Let us assume that $\dim(\mathcal{K}_l) = l$.

To compute an approximating vector from the subspace $\mathcal{K}_l$ we need a "good" basis of $\mathcal{K}_l$ which is not too expensive to compute. In view of (1.4) it would be useful if we had one set of basis vectors $q_1, q_2, q_3, \dots$ such that

$$\text{span}\{q_1, \dots, q_l\} = \mathcal{K}_l \text{ for all } l = 1, 2, 3, \dots \tag{1.7}$$

Let $Q_l \in \mathbb{R}^{N\times l}$ be the matrix having the q_j's as column vectors, i.e.

$$Q_l = [q_1, q_2, \ldots, q_l]. \tag{1.8}$$

Then any $w \in \mathcal{K}_l$ can be represented as $w = Q_l z_l$ with a suitable vector $z_l \in \mathbb{R}^l$. To choose a $z_l \in \mathbb{R}^l$ such that $Q_l z_l$ is a good approximation to $\tilde{x}$ four strategies are often considered:

1. **The Ritz–Galerkin approach**:

   Compute $z_l \in \mathbb{R}^l$ such that the residual is orthogonal to the subspace $\mathcal{K}_l$, i.e.

   $$0 = Q_l^T r_l = Q_l^T r_0 - Q_l^T A Q_l z_l.$$

   Thus in this case we have to solve the l–dimensional linear system

   $$Q_l^T A Q_l z_l = Q_l^T r_0 \tag{1.9}$$

   to receive z_l. We can hope that with increasing dimension of $\mathcal{K}_l$ the residuals become smaller and smaller.

2. **The minimum residual approach**:

   Determine $z_l \in \mathbb{R}^l$ such that the Euclidean norm $\| r_l \|_2$ of the residual is minimal over $\mathcal{K}_l$, i.e.

   $$\| r_l \|_2 = \| r_0 - AQ_l z_l \|_2 = \min_{z \in \mathbb{R}^l} \| AQ_l z - r_0 \|_2 . \tag{1.10}$$

   To receive z_l we have to solve the least squares problem for the $N \times l$ system matrix AQ_l and the right hand side vector r_0.

3. **The Petrov–Galerkin approach**:

   Choose $z_l \in \mathbb{R}^l$ such that the residual $r_l = r_0 - AQz_l$ is orthogonal to some other suitable subspace $\tilde{\mathcal{K}}_l$ of dimension l. If the columns of $\tilde{Q}_l$ are a basis for $\tilde{K}_l$ than in analogy to the Ritz–Galerkin approach we have to compute z_l as the solution of the linear system

   $$\tilde{Q}_l^T A Q_l z_l = \tilde{Q}_l^T r_0. \tag{1.11}$$

4. **The minimum error approach**:

   Compute $z_l \in \mathbb{R}^l$ such that the error $\| \tilde{x} - Q_l z_l \|_A$ for a suitable norm $\| \, . \, \|_A$ is minimal.

In the first three cases we aim for a small residual. But note that a very small residual does not necessarily mean that the error $\tilde{x} - \tilde{x}_l$ is small. $\| \tilde{x} - \tilde{x}_l \|_2$ may be big even for small $\| r_l \|_2$ if A's condition number is large. Here we will not consider this problem.

Two questions remain:

1. Which subspaces are suitable in the sense that already for an l which is small compared to N we get a sufficiently small residual r_l?
2. Is there an efficient way to compute the basis vectors $q_1, q_2, q_3, \ldots$? These vectors are in $\mathbb{R}^N$, and if we have to store too many of them and if we need all $q_1, q_2, \ldots, q_l$ to compute the next q_{l+1} and the z_{l+1} then we run into problems with the storage and CPU time requirements again.

To understand the interest in Krylov subspaces for this purpose it is useful to recall that A^{-1} is a polynomial in A. If $m(\lambda) = \sum_{j=0}^{d} \alpha_j \lambda^j$ is the minimal polynomial of A, then we know that $\alpha_0 \neq 0$, because A is nonsingular, and that $0 = m(A) = \sum_{j=1}^{d} \alpha_j A^j + \alpha_0 I_n$. Rearranging the equation and multiplying by A^{-1} we get

$$A^{-1} = \sum_{j=1}^{d} -\frac{\alpha_j}{\alpha_0} A^{j-1} = \sum_{j=0}^{d-1} -\frac{\alpha_{j+1}}{\alpha_0} A^j$$

and for the solution $\tilde{x}$ of $A\tilde{x} = r_0$:

$$\tilde{x} = A^{-1} r_0 = \sum_{j=0}^{d-1} -\frac{\alpha_{j+1}}{\alpha_0} A^j r_0, \qquad (1.12)$$

$$\text{i.e.} \quad \tilde{x} \in \operatorname{span}\{r_0, Ar_0, A^2 r_0, \ldots, A^{d-1} r_0\}$$

Moreover, let k be the smallest integer such that r_0 is contained in a k–dimensional invariant subspace, i.e. there exists a $k-\dim$ subspace $\mathcal{V}$ of $\mathbb{R}^N$, such that $A(\mathcal{V}) \subseteq \mathcal{V}$ and $r_0 \in \mathcal{V}$ and k is the smallest integer with this property. Then it is easy to see that $A^k r_0$ is a linear combination of $r_0, Ar_0, \ldots, A^{k-1} r_0$.

Together with (1.12) we get therefore that the solution $\tilde{x}$ of $A\tilde{x} = r_0$ satisfies

$$\tilde{x} \in \operatorname{span}\{r_0, Ar_0, A^2 r_0, \ldots, A^{\min\{d,k\}-1} r_0\}. \qquad (1.13)$$

Chapter 2

Krylov Subspaces

2.1 Properties of Krylov Subspaces

Definition 1
The **l-th Krylov subspace** for the linear system $Ax = b$ with initial residual $r_0 = b - Ax_0$ is defined as

$$\mathcal{K}_l(A, r_0) = \operatorname{span}\{r_0, Ar_0, A^2 r_0, \dots, A^{l-1} r_0\}. \tag{2.1}$$

Krylov subspaces have remarkable properties which are very convenient for the approximation of the solution x following the general ideas of the previous chapter. Note that

$$\mathcal{K}_1(A, r_0) \subseteq \mathcal{K}_2(A, r_0) \subseteq \mathcal{K}_3(A, r_0) \subseteq \dots \tag{2.2}$$

and as we have just discussed at the end of the last chapter, the maximal reachable dimension is $\min\{d, k\}$, where d is the degree of A's minimal polynomial and k is the smallest integer such that r_0 is contained in a k–dimensional invariant subspace of A. Also we know that the solution $\tilde{x}$ of $A\tilde{x} = r_0$ is contained in $\mathcal{K}_{\min\{d,k\}}(A, r_0)$, such that for the solution x of $Ax = b$ we have

$$x \in x_0 + \mathcal{K}_{\min\{d,k\}}(A, r_0). \tag{2.3}$$

Obviously $\mathcal{K}_l(A, r_0)$ is also characterized by

$$\mathcal{K}_l(A, r_0) = \{p(A)r_0 | p \in \Pi_{l-1}\}, \tag{2.4}$$

where Π_{l-1} is the set of real polynomials of degree at most $l - 1$. Moreover, for any $p(A)r_0 \in \mathcal{K}_l(A, r_0)$ the residual $r_l = r_0 - Ap(A)r_0$ satisfies

$$r_l = (I - Ap(A))r_0 = q(A)r_0, \tag{2.5}$$

where

$$q(\lambda) = 1 - \lambda p(\lambda). \tag{2.6}$$

Note that

$$q \in \Pi_l \quad \text{and} \quad q(0) = 1. \tag{2.7}$$

Define

$$\hat{\Pi}_l := \{q \in \Pi_l | q(0) = 1\}. \tag{2.8}$$

Then it is very easy to see that

$$\hat{\Pi}_l = \{1 - \lambda p(\lambda) | p \in \Pi_{l-1}\}.$$

Thus we can reformulate our minimum residual approach as: Find $q \in \hat{\Pi}_l$ such that

$$\| q(A)r_0 \|_2 = \min_{\tilde{q} \in \hat{\Pi}_l} \| \tilde{q}(A)r_0 \|_2 \, . \tag{2.9}$$

If A is diagonalizable and $A = X^{-1}\Lambda X$, where $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_{\mathrm{n}})$, then for any $\tilde{q} \in \hat{\Pi}_l$

$$\| \tilde{q}(A)r_0 \|_2 \;\leq\; \| X^{-1}\tilde{q}(\Lambda)X \|_2 \;\; \| r_0 \|_2 \;\leq\; \| X^{-1} \|_2 \;\; \| X \|_2 \;\; \| \tilde{q}(\Lambda) \|_2 \;\; \| r_0 \|_2,$$

such that the minimal $q \in \hat{\Pi}_l$ satisfies

$$\| r_l \|_2 \;=\; \| q(A)r_0 \|_2 \;\leq\; \| X \|_2 \;\; \| X^{-1} \|_2 \;\; \| r_0 \|_2 \min_{\tilde{q} \in \hat{\Pi}_l} \; \max_{\lambda \in \{\lambda_1, \ldots, \lambda_n\}} |\tilde{q}(\lambda)|. \tag{2.10}$$

We see that this upper bound for $\| r_l \|_2$ depends crucially on the eigenvalues and eigenvectors of A.

2.2 Arnoldi and Lanczos method

For the computation of an approximation from $\mathcal{K}_l(A, r_0)$ we still need a good basis. The obvious basis $r_0, Ar_0, A^2r_0, \ldots$ is not suitable for computational purposes. These vectors are the sequence of the power method for A with starting vector r_0. If we compute them without normalization, then $A^k r_0$ will quickly get very large or very small, depending on the norm of A being larger or smaller than 1. If we normalize them, then often they quickly tend to the dominant eigenvector of A and therefore the vectors become almost linearly dependent. An orthonormal basis would overcome these problems (in exact arithmetic). We could construct orthonormal vectors $q_1, q_2, q_3, \ldots$ from the vector sequence $r_0, Ar_0, A^2r_0, \ldots$ by the Gram–Schmidt orthonormalization procedure. This would, however, still force us to compute the vectors $A^k r_0$. In the following we will give a characterization of the vectors $q_1, q_2, q_3, \ldots$ from the Gram–Schmidt procedure which allows us to compute them without forming the original sequence $r_0, Ar_0, A^2r_0, \ldots$ explicitly.

Definition 2
$K_l(A, r_0) := [r_0, Ar_0, A^2r_0, \ldots, A^{l-1}r_0] \in \mathbb{R}^{N\times l}$ is the **l-th Krylov matrix** for A with initial vector r_0.

If $Q_l = [q_1, \ldots, q_l]$, where $q_1, \ldots, q_l$ are the results of the Gram–Schmidt procedure for the vector sequence $r_0, Ar_0, A^2r_0, \ldots, A^{l-1}r_0$, then there exists an upper triangular matrix $R_l = \left[\,\diagdown\!\!\rceil\,\right] \in \mathbb{R}^{l\times l}$ such that

$$K_l(A, r_0) = Q_lR_l \tag{2.11}$$

Define $m = \min\{d, k\}$, where as before d is the degree of A's minimal polynomial and k is the maximal dimension such that r_0 is contained in a k–dimensional invariant subspace of A.

Then

$$K_m(A, r_0) = Q_mR_m \tag{2.12}$$

and because m is the maximal reachable dimension for $\mathcal{K}_l(A, r_0)$ for all $l = 1, 2, \ldots$ $K_m(A, r_0)$ is nonsingular and therefore R_m is nonsingular. Because of the maximality of m we have

$$A^mr_0 = \sum_{j=0}^{m-1} c_jA^jr_0$$

for a suitable vector $[c_0, \ldots, c_{m-1}]^T \in \mathbb{R}^m \setminus \{0\}$.

Theorem 3
Let the assumptions and notations above be given.

(i) The following are equivalent:

(a) $K_m(A, r_0) = Q_mR_m$ *is the QR decomposition with* $Q_m \in \mathbb{R}^{N\times m}$, $Q_m^TQ_m = I_m$ *and* $R_m = (r_{ij})_{i,j\in\{1,\ldots,m\}} = \left[\,\diagdown\!\!\rceil\,\right] \in \mathbb{R}^{m\times m}$ *and* $r_{jj} > 0$ *for all* $j \in \{1, \ldots, m\}$,

(b)

$$AQ_m = Q_mH_m, \tag{2.13}$$

where $Q_m^TQ_m = I_m, Q_me_1 = \frac{1}{\|r_0\|_2}r_0$ *and*
$H_m := (h_{ij})_{i,j\in\{1,\ldots,m} = \left[\,\diagdown\!\!\diagdown\!\!\rceil\,\right] = R_mCR_m^{-1}$ *is an upper Hessenberg ma-*

trix with $h_{j+1,j} > 0$ *for all* $j \in \{1, \dots, m-1\}$ *and*

$$C = \begin{bmatrix} 0 & & & & c_0 \\ 1 & 0 & & & c_1 \\ & 1 & \ddots & & \vdots \\ & & \ddots & 0 & \\ & & & 1 & c_{m-2} \\ & & & & c_{m-1} \end{bmatrix}.$$

(ii) *If* A *is symmetric, then* H_m *in (i) is symmetric and tridiagonal, i.e.*

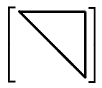
$$AQ_m = Q_m T_m, \text{ where } T_m = \begin{bmatrix} \alpha_1 & \beta_2 & & \\ \beta_2 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \beta_m \\ & & \beta_m & \alpha_m \end{bmatrix}. \tag{2.14}$$

Proof

(i) If $AQ_m = Q_m H_m$ and $Q_m e_1 = \frac{1}{\|r_0\|_2} r_0$, then $A^k Q_m = A^{k-1} Q_m H_m = \dots = Q_m H_m^k$ for all $k \in \mathbb{N}$.

Then

$$\begin{aligned} K_m(A, r_0) &= [r_0, Ar_0, \dots, A^{m-1} r_0] = \| r_0 \|_2 \, [Q_m e_1, AQ_m e_1, \dots, , A^{m-1} Q_m e_1] \\ &= Q_m \underbrace{\| r_0 \|_2 \, [e_1, H_m e_1, \dots, H^{m-1} e_1]}_{=: \; R_m =} \end{aligned}$$



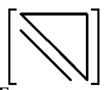
is the QR decomposition, where it is easily checked that $h_{j+1,j} > 0$ for all j implies $r_{jj} > 0$ for all j.

If on the other hand $K_m(A, r_0) = Q_m R_m$ is the QR decomposition with $r_{jj} > 0$ for all j, then

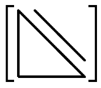
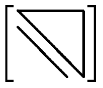
$$AK_m(A, r_0) = [Ar_0, A^2 r_0, \dots, A^m r_0] = K_m(A, r_0) C,$$

which is equivalent to

$$A \; Q_m R_m = Q_m R_m C.$$

Then $Q_m^T A Q_m = R_m C R_m^{-1} =: H_m =$  . It is easily seen that $r_{jj} > 0$ for all j implies $h_{j+1,j} > 0$ for all j. $Q_m^T Q_m = I_m$ by construction and $Q_m e_1 = \frac{1}{\|r_0\|_2} r_0$ is obvious because $K_m(A, r_0) = Q_m R_m$.

(ii) If A is symmetric and $AQ_m = Q_m H_m$, $Q_m^T Q_m = I_m$, then

$$H_m^T =$$

$$= Q_m^T A^T Q_m = Q_m^T A Q_m = H_m =$$

.

Therefore H_m is an upper Hessenberg as well as a lower Hessenberg matrix and thus has to be a tridiagonal matrix, which in addition is symmetric. □

Recall that the QR decomposition $Q_m R_m$ always exists and is uniquely determined because $K_m(A, r_0)$ is of full rank and $r_{jj} > 0$ for all j. This theorem tells us that we can compute the orthonormal Gram-Schmidt basis $\{q_1, \dots, q_l\}$ of $\mathcal{K}_l(A, r_0)$ for all $l \in \{1, \dots, m\}$ by computing the matrix Q_m with $Q_m e_1 = \frac{1}{\|r_0\|} r_0$, which transforms A to upper Hessenberg form $H_m = Q_m^T A Q_m$. The first l columns of Q_m are the desired basis $\{q_1, \dots, q_l\}$ for $\mathcal{K}_l(A, r_0)$.

Denote $H_l = (h_{ij})_{i,j \in \{1,\dots,l\}} \in \mathbb{R}^{l \times l}$ for $l \in \{1, \dots, m\}$ then a closer look at (2.13) yields

$$AQ_l = Q_l H_l + h_{l+1,l} q_{l+1} e_l^T \text{ for all } l \in \{1, \dots, m-1\} \tag{2.15}$$

and

$$Q_l^T A Q_l = H_l \text{ for all } l \in \{1, \dots, m\}. \tag{2.16}$$

Evaluating these two equations stepwise for $l = 1, 2, \dots, m-1$, we derive a procedure to compute $q_1, q_2, \dots, q_m$ and $H_1, H_2, \dots, H_m$ subsequently:
We know $q_1 = \frac{1}{\|r_0\|} r_0$ and from (2.16) we get

$$q_1^T A q_1 = h_{11}.$$

Assume that we have already computed Q_{l-1} and H_{l-1}. Then from (2.15) evaluated in the last column we get

$$Aq_l = Q_l \begin{bmatrix} h_{1l} \\ \vdots \\ h_{ll} \end{bmatrix} + h_{l+1,l} q_{l+1} = \sum_{j=1}^{l} h_{jl} q_j + h_{l+1,l} q_{l+1}.$$

Set $w_{l+1} := h_{l+1,l} q_{l+1}$. We can compute

$$w_{l+1} = Aq_l - \sum_{j=1}^{l} h_{jl} q_j \tag{2.17}$$

because all quantities on the right hand side are known. Moreover, we know that $h_{l+1,l} > 0$ and $\| q_{l+1} \|_2 = 1$. Therefore $h_{l+1,l} = \| w_{l+1} \|_2$ and $q_{l+1} = \frac{1}{h_{l+1,l}} w_{l+1}$.

This procedure to compute $q_1, q_2, \dots$ and $H_1, H_2, \dots$ is called Arnoldi process. We summarize the computation:

```
Arnoldi method:
        $w = r_0$
        $h_{1,0} = \| r_0 \|_2$
        $l = 0$
        while $h_{l+1,l} \neq 0$
            $q_{l+1} = w/h_{l+1,l}$
            $l = l+1$
            $w = Aq_l$
            for $j = 1 : l$
                $h_{j,l} = q_j^T w$
                $w = w - h_{j,l}qj$
            end
                $h_{l+1,l} = \| w \|_2$
        end
```

It is not difficult to check that the computation in the j–loop does indeed compute w_{l+1} as in (2.17).
With our assumptions we know that $h_{l+1,l} > 0$ for all $l \in \{1, \dots, m-1\}$ and $h_{m+1,m}$ from the algorithm above will be zero (all in exact arithmetic).

For symmetric A the computation simplifies, because then

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

Instead of (2.15) and (2.16) we get here

$$AQ_l = Q_l T_l + \beta_{l+1} q_{l+1} e_l^T \text{ for all } l \in \{1, \dots, m-1\} \tag{2.18}$$

and

$$Q_l^T A Q_l = T_l \text{ for all } l \in \{1, \dots, m\} \tag{2.19}$$

and instead of (2.17) we get in this case for $l = 1, \dots, m-1$:

$$w_{l+1} = Aq_l - \alpha_l q_l - \beta_l q_{l-1}, \tag{2.20}$$

where we set $\beta_1 =\| r_0 \|_2$, $q_0 := 0$ and

$$\beta_{l+1} =\| w_{l+1} \|_2, \quad q_{l+1} = \frac{1}{\beta_{l+1}} w_{l+1}.$$

This procedure is called the Lanzcos process and we summarize analogously:

```
Lanczos method:
        $w = r_0$
        $\beta_1 = \| r_0 \|_2$
        $l = 0$
        while $\beta_{l+1} \neq 0$
            $q_{l+1} = w/\beta_{l+1}$
            $l = l + 1$
            $w = Aq_l$
            $\alpha_l = q_l^T w$
            $w = w - \alpha_l q_l - \beta_l q_{l-1}$
            $\beta_{l+1} = \| w \|_2$
        end
```

As before with our assumption we have $\beta_l \neq 0$ for $l \leq m$ and $\beta_{m+1} = 0$ in exact arithmetic. Note that in both cases in each step of the iteration above we need only one matrix–vector product of the form Aq_l, which is very convenient for large sparse matrices.

Remark 4

1. The equations (2.13) and (2.14) imply that the columns of Q_m span an m–dimensional invariant subspace of A. The eigenvalues of H_m or T_m, respectively, are then also eigenvalues of A. If in (2.15) and (2.18) the quantities $h_{l+1,l}$ or β_{l+1}, respectively, are very small, then we can consider the columns of Q_l to span a subspace which is approximately an invariant subspace. The eigenvalues of H_m or T_m can then be considered as approximations of eigenvalues of A. The Arnoldi and Lanczos method are therefore used to compute such approximations for large matrices, see e.g. [3] and references therein.

2. If the vectors $q_1, q_2, \ldots$ are computed with the Arnoldi or Lanczos process in finite precision arithmetic then one observes that orthogonality gets lost with increasing l due to roundoff errors. A monitoring of the orthogonality together with a reorthogonalization of the computed vectors is often necessary. In this introduction of the basic ideas we will not consider these rounding errror effects, but they are important and have to be handled carefully in practice.

Chapter 3

Iterative Methods

3.1 GMRES and MINRES

We have now an efficient way to compute orthonormal basis vectors $q_1, \ldots, q_l$ of $\mathcal{K}_l(A, r_0)$. If we want to follow the minimal residual approach in Chapter 1 we have according to (1.10) to compute z_l such that

$$\| AQ_l z_l - r_0 \|_2 = \min_{z \in \mathbb{R}^l} \| AQ_l z - r_0 \|_2 .$$

From (2.15) we have

$$AQ_l = Q_l H_l + h_{l+1,l} q_{l+1} e_l^T \text{ for } l = 1, \ldots, m-1$$

and $AQ_m = Q_m H_m$.

Note that with

$$H_{l+1,l} := \begin{bmatrix} & H_l & \\ 0 \ \ldots \ 0 & & h_{l+1,l} \end{bmatrix} = \begin{bmatrix} & & \diagdown\!\!\!\urcorner \\ 0 & \ldots \ 0 & h_{l+1,l} \end{bmatrix} \in \mathbb{R}^{l+1,l}$$

we have $Q_l H_l + h_{l+1,l} q_{l+1} e_l^T = Q_{l+1} H_{l+1,l}$. Thus (2.15) can be rewritten as

$$AQ_l = Q_{l+1} H_{l+1,l}. \tag{3.1}$$

Because $r_0 = \| r_0 \|_2 \, Q_{l+1} e_1$ for all l we get

$$AQ_l z - r_0 = Q_{l+1}(H_{l+1,l} z - \| r_0 \|_2 \, e_1) \text{ for } l \in \{1, \ldots, m-1\}.$$

Therefore for any $z \in \mathbb{R}^l$ we have

$$\begin{aligned} \| AQ_l z - r_0 \|_2 &= \| Q_{l+1}(H_{l+1,l} z - \| r_0 \|_2 \, e_1) \|_2 \\ &= \| H_{l+1,l} z - \| r_0 \|_2 \, e_1 \|_2, \end{aligned}$$

where the last equation holds because $Q_{l+1}^T Q_{l+1} = I_{l+1}$.

Thus to compute z_l we have to solve the linear least squares problem

$$\min_{z \in \mathbb{R}^l} \| H_{l+1,l} z - \| r_0 \|_2 \, e_1 \|_2 . \tag{3.2}$$

An efficient way to compute this solution is the following:

Compute the QR–decomposition

$$H_{l+1,l} = V_{l+1} R_l,$$

where $V_{l+1} \in \mathbb{R}^{l+1 \times l+1}$ is orthogonal and

$$R_l = \begin{bmatrix} & \hat{R}_l & \\ 0 & \dots & 0 \end{bmatrix} \in \mathbb{R}^{l+1 \times l} \text{ with } \hat{R}_l = \begin{bmatrix} \diagdown \end{bmatrix} \text{upper triangular.}$$

Note that in our situation $H_{l+1,l}$ has full rank, because $h_{j+1,j} > 0$ for all $j \in \{1, \dots, l\}$ and $l < m$. Therefore $\hat{R}_l$ is invertible and for any $z \in \mathbb{R}^l$ we have

$$\| H_{l+1,l} z - \| r_0 \|_2 \, e_1 \|_2 = \| R_l z - \| r_0 \|_2 \, V_{l+1}^T e_1 \|_2 \, .$$

The solution z_l of (3.2) is then received as

$$z_l = \| r_0 \|_2 \, [\hat{R}_l^{-1}, 0] V_{l+1}^T e_1 \tag{3.3}$$

and the residual r_l is the absolute value of the last entry in the vector $\| r_0 \|_2 \, V_{l+1}^T e_1$, i.e.

$$\| r_l \|_2 = \| r_0 \|_2 \, e_{l+1}^T V_{l+1}^T e_1. \tag{3.4}$$

Fortunately the QR decomposition of Hessenberg matrices is very special and therefore the solution z_l of the least squares problem as well as $\| r_l \|_2$ can be computed as a simple update of z_{l-1}, the solution of the previous step, and $\| r_{l-1} \|_2$, respectively.

This method is known as GMRES (**g**eneralized **m**inimal **res**idual) and was introduced by Saad and Schultz 1986 [9]. As the residuals are minimized over the sequence of increasing subspaces $\mathcal{K}_1(A, r_0) \subseteq \mathcal{K}_2(A, r_0) \subseteq \dots$, the norms of the residuals must decrease monotonically.

In exact arithmetic we will get the exact solution after $m = \min\{d, k\}$ steps with a residual equal to zero. We hope, however, that the $\| r_l \|_2$ becomes small enough to let $x_0 + \tilde{x}_l$ be a sufficiently good approximation to x for much smaller l. This is what we will call "convergence". By "speed of convergence" we mean the speed with which $\| r_l \|_2$ goes to zero with growing l. The GMRES procedure can be sketched as follows:

GMRES (basic form):

```
    $w = r_0$
    $h_{10} = \| r_0 \|_2$
    $l = 0$
    while $h_{l+1,l} > 0$
        $q_{l+1} = w/h_{l+1,l}$
        $l = l+1$
        $w = Aq_l$
        for $j = 1 : l$
            $h_{jl} = q_j^T w$
            $w = w - h_{jl} q_j$
        end
        $h_{l+1,l} = \| w \|_2$
        Compute $z_l$ such that $\| H_{l+1,l}, z_l - h_{10} e_1 \|_2$ minimal
        $x_l = x_0 + Q_l z_l$
        if $\varphi_l = \| b - Ax_l \|_2 < tol$ then STOP
    end
```

tol is here a tolerance value, which we have to specify for our residual norm to be considered small enough. Note that for the computation of z_l and φ_l we will of course make use of the updating methods mentioned above. Also it is not neccesary to compute x_l for each l. Multiplication with Q_l in each step is a bit expensive. It is sufficient if we compute x_l for the step l in which φ_l is small enough.

For the computation in step l we need all $l-1$ previously computed vectors $q_1, \dots, q_{l-1}$ which are in $\mathbb{R}^N$. If N is very large and if we need many steps of the iteration to get a small residual, then storage requirements and computational costs become a problem again. To avoid these difficulties GMRES is usually restarted after a fixed number j of steps with the current approximtion as new initial guess. This method is called restarted GMRES or GMRES (j). The choice of a suitable j is not easy. The speed of convergence depends critically on j and may vary drastically with j.

Figure 3.1 presents the relative residual norms $\| r_l \| / \| r_0 \|_2$ as a function of the iteration step l in a semi logarithmic scale for the matrix lns511 from Chapter 1 for GMRES and GMRES (10), where $x_0 = 0$ and $tol = 0.001$. For GMRES the residuals decrease nicely, but not as fast as we might want. Note that the dimension is here only 511. GRMES (10) does not do well on this example.

If A is symmetric then we already know that we can substitute the Arnoldi method by the Lanzcos method to compute the basis $q_1, q_2, \dots$ for the Krylov subspaces.

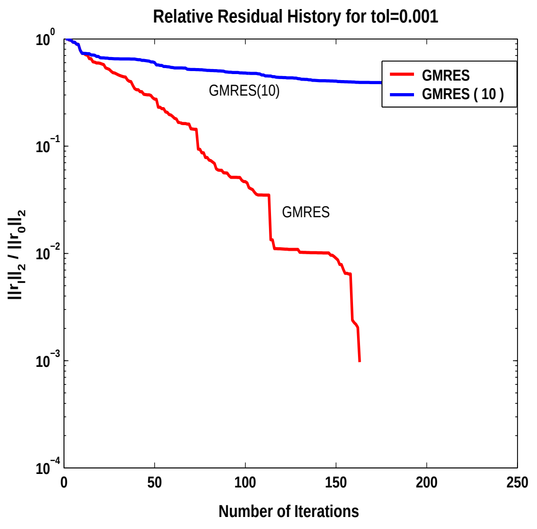


Figure 3.1: GMRES and GMRES(10) for lns511

Here we only need q_{l-2} and q_{l-1} to compute q_l and the entries $\alpha_{l-1}, \beta_{l-1}$ and β_l of T_l. Thus we do not run into the storage and computational costs problems as with GMRES.

In the minimal residual approach we now have to solve the least squares problem

$$\min_{z \in \mathbb{R}^l} \| T_{l+1,l} z - \| r_0 \|_2 \, e_1 \|_2$$

in step l, and in analogy to the Hessenberg case the solution z_l can be received by simple updating from z_{l-1}.

This method is essentially the method MINRES. Storage requirements and computational costs are here much less than for GMRES due to the fact that we have the 3–term recursion for the computation of q_l and only tridiagonal matrices T_l instead of $l \times l$ upper Hessenberg matrices H_l for $l = 1, 2, \ldots$.

Another matrix from the Harwell-Boeing Collection is sherman1, whose portrait is given in Figure 3.2. It is a symmetric 1000×1000 matrix arising from a discretization of a partial differential equation in an oil recovery problem. The first plot in Figure 3.3 displays the relative residual curve for sherman1 if MINRES is applied with $x_0 = 0$ and $tol = 0.001$. One may of course also use GMRES, disregarding the symmetry of A. The right hand side plot shows the relative residual for this case. In exact arithmetic GMRES should produce the same residuals as MINRES in this case, and the GMRES convergence displayed in the figure confirms this. Also GMRES(10) is not too bad in this case.

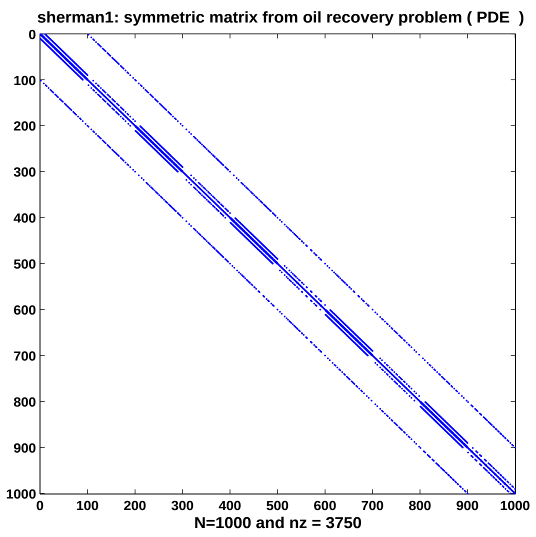


Figure 3.2: Portrait of sherman1

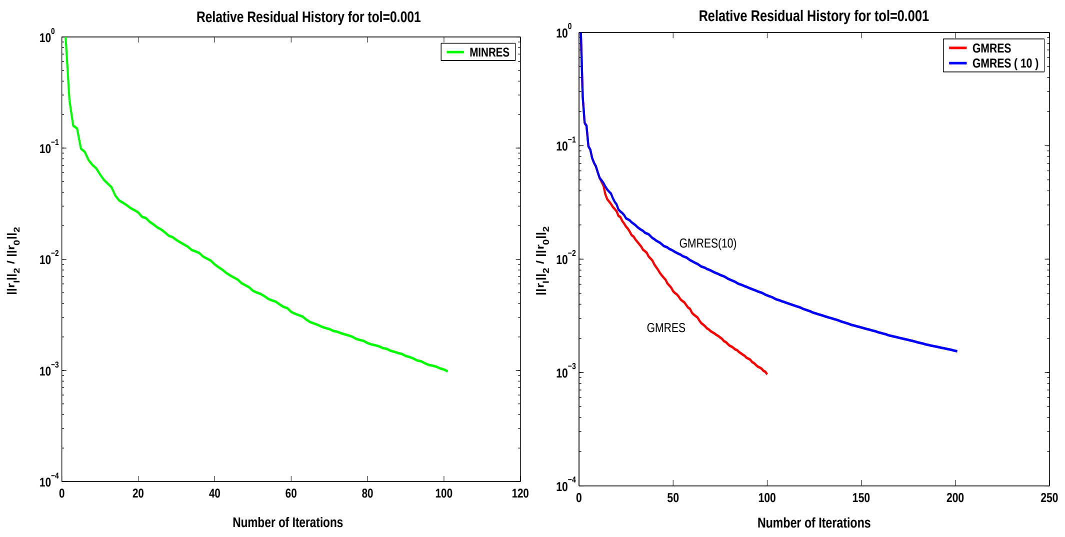


Figure 3.3: MINRES and GMRES for sherman1

3.2 CG Method

Large linear systems coming from the discretization of partial differential equations often have a system matrix A which is not only symmetric but also, like our example (1.2), positive definite.

We can of course apply MINRES to exploit the symmetry of A. But positive definiteness permits the derivation of an even simpler method.

Let us assume in the following that A is symmetric and positive definite (spd). Then it is easy to see that $T_l = Q_l^T A Q_l$ from (2.19) is also spd. Therefore T_l has a Cholesky decomposition $T_l = \tilde{B}_l \tilde{B}_l^T$, where B_l is a nonsingular lower triangular matrix. Because T_l is tridiagonal $\tilde{B}_l$ must even be bidiagonal, i.e. we have

$$T_l = \begin{bmatrix} \alpha_1 & \beta_2 & & \\ \beta_2 & \ddots & \ddots & \\ & \ddots & \ddots & \beta_l \\ & & \beta_l & \alpha_l \end{bmatrix} = \tilde{B}_l \tilde{B}_l^T = \begin{bmatrix} a_1 & & & \\ b_2 & \ddots & & \\ & \ddots & \ddots & \\ & & b_l & a_l \end{bmatrix} \begin{bmatrix} a_1 & b_2 & & \\ & \ddots & \ddots & \\ & & \ddots & b_l \\ & & & a_l \end{bmatrix}.$$

Solving a linear system $w = T_l z$ is therefore extremely simple and has almost negligible computational costs.

In the Ritz–Galerkin approach to choose an approximation of $\tilde{x}$ from $\mathcal{K}_l(A, r_0)$ we compute, according to (1.9), $z_l \in \mathbb{R}^l$ such that

$$Q_l^T A Q_l z_l = Q_l^T r_0 = \| r_0 \|_2 \, e_1.$$

Here $Q_l^T A Q_l = T_l$ and we have to solve

$$T_l z_l = \| r_0 \|_2 \, e_1.$$

We can thus take full advantage of the symplicity with which such systems can be solved.

This is up to a modification in the computations of $x_l = Q_l z_l$ the Conjugate Gradient method (CG method). The modifications make it possible to get not only the norms of the residuals in the computational process but also the residuals themselves. The modification consists in a diagonal transformation of T_l.

Theorem 5

Let the assumptions and notations above hold.

There exists a diagonal matrix $D_m = \mathrm{diag}(\mathrm{d}_1, \ldots, \mathrm{d}_\mathrm{m}) \in \mathbb{R}^\mathrm{m}$ *with positive diagonal entries such that the following holds*

- *For* $W_m := Q_m D_m$ *we have*
$$W_m^T W_m = D_m Q_m^T Q_m D_m = D_m^2 =: \Lambda_l = \operatorname{diag}(\lambda_1, \dots, \lambda_{\mathrm{m}}) \tag{3.5}$$
and $\lambda_1 = \| r_0 \|_2^2$.

- *For* $\tilde{T}_m := D_m T_m D_m = \begin{bmatrix} \ddots & \ddots & \\ \ddots & \ddots & \ddots \\ & \ddots & \ddots \end{bmatrix}$ *we get*
$$\tilde{T}_m = B_m \Omega_m B_m^T$$
with
$$B_m = \begin{bmatrix} 1 & & & & \\ -\frac{\lambda_2}{\lambda_1} & 1 & & & \\ & -\frac{\lambda_3}{\lambda_2} & 1 & & \\ & & \ddots & \ddots & \\ & & & -\frac{\lambda_m}{\lambda_{m-1}} & 1 \end{bmatrix}$$
and
$$\Omega_m = \operatorname{diag}(\omega_1, \dots, \omega_{\mathrm{m}}).$$

Proof
The proof is not too difficult but tedious and therefore omitted. □

Note that the columns of W_m differ from $q_1, \dots, q_m$ only by a factor. Thus for all $l \in \{1, \dots, m\}$ the first l column vectors of W_m are still an orthogonal (but not orthonormal) basis for $\mathcal{K}_l(A, r_0)$. With these new matrices we get from (2.19)
$$AW_m = AQ_m D_m = Q_m T_m D_m = Q_m D_m D_m^{-1} D_m^{-1} \tilde{T}_m = W_m \Lambda_m^{-1} \tilde{T}_m = W_m \Lambda_m^{-1} B_m \Omega_m B_m^T.$$
If we define $Y_m := W_m B_m^{-T}$ then
$$AY_m = W_m \Lambda_m^{-1} B_m \Omega_m \tag{3.6}$$
and
$$Y_m B_m^T = W_m. \tag{3.7}$$
Note that according to (3.6) and the definition of Y_m:
$$Y_m^T A Y_m = Y_m^T W_m \Lambda_m^{-1} B_m \Omega_m = B_m^{-1} \Lambda_m \Lambda_m^{-1} B_m \Omega_m = \Omega_m = \operatorname{diag}(\omega_1, \dots, \omega_{\mathrm{m}}) \tag{3.8}$$
i.e. the columns $y_1, \dots, y_m$ of Y_m are A-orthogonal.
Moreover
$$\Lambda_m^{-1} B_m = \begin{bmatrix} \frac{1}{\lambda_1} & & & & \\ -\frac{1}{\lambda_1} & \frac{1}{\lambda_2} & & & \\ & -\frac{1}{\lambda_2} & \frac{1}{\lambda_3} & & \\ & & \ddots & \ddots & \\ & & & -\frac{1}{\lambda_{m-1}} & \frac{1}{\lambda_m} \end{bmatrix} \tag{3.9}$$

and if $W_l = [w_1, \ldots, w_m]$ then

$$w_1 = q_1 d_1 = \frac{r_0}{\| r_0 \|_2} \sqrt{\lambda_1} = r_0. \tag{3.10}$$

Evaluating (3.7) and (3.6) successively column by column together with (3.5), (3.8) and (3.9) we get in analogy to the derivation of the Arnoldi and Lanzcos method a recursion for $w_1, w_2, \ldots$ and $y_1, y_2, \ldots$ as follows

$$\begin{aligned} w_1 &= y_1 = r_0 \\ \lambda_1 &= \| r_0 \|_2^2 = r_0^T r_0 \text{ and } \omega_1 = r_0^T A r_0 \end{aligned}$$

and for $l \geq 1$:

$$\begin{aligned} Ay_l = \frac{\omega_l}{\lambda_l} w_l - \frac{\omega_l}{\lambda_l} w_{l+1}, \text{ i.e. } w_{l+1} &= w_l - \frac{\lambda_l}{\omega_l} A y_l, \\ \lambda_{l+1} &= w_{l+1}^T w_{l+1} \end{aligned}$$

and

$$\begin{aligned} -\frac{\lambda_{l+1}}{\lambda_l} y_l + y_{l+1} = w_{l+1}, \text{ i.e. } y_{l+1} &= \frac{\lambda_{l+1}}{\lambda_l} y_l + w_{l+1} \\ \omega_{l+1} &= y_{l+1}^T A y_{l+1}. \end{aligned}$$

Note that $\lambda_l = w_l^T w_l \neq 0$ as long as $w_l \neq 0$ and $\omega_l = y_l^T A y_l \neq 0$ as long as $y_l \neq 0$ because A is positive definite. We summarize the computation:

The CG–Lanczos procedure:

```
$y_1 = w_1 = r_0$
$l = 1$
while $w_l \neq 0$
    $\beta_l = \frac{w_l^T w_l}{y_l^T A y_l}$
    $w_{l+1} = w_l - \beta_l A y_l$
    $\gamma_{l+1} = \frac{w_{l+1}^T w_{l+1}}{w_l^T w_l}$
    $y_{l+1} = w_{l+1} + \gamma_{l+1} y_l$
    $l = l + 1$
end
```

Our aim is still to compute the Ritz–Galerkin approximation $\tilde{x}_l$ from the $\mathcal{K}_l(A, r_o)$, i.e.

$$\tilde{x}_l = Q_l z_l, \text{ where } T_l z_l = \| r_0 \|_2 \, e_1 \text{ for } l \in \{1, \ldots, m\}.$$

It remains to show that $\tilde{x}_l$ can be derived from $w_1, w_2, \ldots$ and $y_1, y_2, \ldots$.
To see this we look at

$$\tilde{x}_l = Q_l D_l D_l^{-1} z_l = W_l \tilde{z}_l \text{ with } \tilde{z}_l := D_l^{-1} z_l. \tag{3.11}$$

Then

$$\tilde{T}_l \tilde{z}_l = D_l T_l D_l D_l^{-1} z_l = D_l \| r_0 \|_2 \, e_1 = r_0^T r_0 e_1.$$

Because $\tilde{T}_m = B_m \Omega_l B_m^T$ and B_m is a lower triangular we get here for all $l \leq m$ $\tilde{T}_l = B_l \Omega_l B_l^T$ and thus

$$\tilde{z}_l = r_0^T r_0 B_l^{-T} \Omega_l^{-1} B_l^{-1} e_1.$$

Therefore for all l

$$\tilde{x}_l = r_0^T r_0 W_l B_l^{-T} \Omega_l^{-1} B_l^{-1} e_1 = r_0^T r_0 Y_l \Omega_l^{-1} B_l^{-1} e_1.$$

A simple proof by induction shows that elements of $g_l = [\eta_1, \ldots, \eta_l] := \Omega_l^{-1} B_l^{-1} r_0^T r_0 e_1$ are given by $\eta_j = \frac{\lambda_j}{\omega_j}$ for $j = 1, \ldots, l$ and all l.
Thus

$$\tilde{x}_l = \sum_{j=1}^{l} \frac{\lambda_j}{\omega_j} y_j = \tilde{x}_{l-1} + \frac{\lambda_j}{\omega_l} y_l. \tag{3.12}$$

Inserting this recursion into the CG–Lanczos procedure and recalling that $x_l = x_0 + \tilde{x}_l$ we get

```
CG–method (basic form) :
    Choose an initial guess $x_0$
    Compute $r_0 = b - Ax_0$
    Set $w_1 = y_1 = r_0$
        $l = 1$
      while $w_l \neq 0$
          $\beta_l = \frac{w_l^T w_l}{y_l^T A y_l}$
          $x_l = x_{l-1} + \beta_l y_l$
          $w_{l+1} = w_l - \beta_l A y_l$
          $\gamma_{l+1} = \frac{w_{l+1}^T w_{l+1}}{w_l^T w_l}$
          $y_{l+1} = w_{l+1} + \gamma_{l+1} y_l$
          $l = l + 1$
      end
```

The computational costs per step are obviously very low. As before we need only one matrix vector product Ay_l per iteration step.

Moreover, we can prove that w_l is the $(l-1)$-st residual.

Lemma 6
With the notations above we have

$$r_{l-1} = b - Ax_{l-1} = w_l \textit{ for all } l \in \{1, \ldots, m\}.$$

Proof
$w_1 = r_0 = b - Ax_0$

Assume that the statement holds for an $l \geq 1$. Then

$$\begin{aligned} r_l = b - Ax_l &= b - A(x_{l-1} + \beta_l y_l) = r_{l-1} - A\beta_l y_l \\ &= w_l - A\beta_l y_l = w_{l+1}. \end{aligned} \qquad \square$$

So far we have seen that the CG–method is an extremely efficient and inexpensive way to compute the Ritz–Galerkin approximations from the Krylov subspaces $\mathcal{K}_l(A, r_0)$, where we also get the residual vectors in each step.

Because A is spd

$$\| v \|_A := \sqrt{v^T A v}$$

is a norm on $\mathbb{R}^N$. It can be shown that

$$\| x - (x_0 + Q_l z_l) \|_A = \min_{z \in \mathcal{K}_l(A, r_0)} \| x - (x_0 + Q_l z) \|_A .$$

Thus $x_0 + Q_l z_l$ is the best approximation from $x_0 + \mathcal{K}_l(A, r_0)$ with respect to $\| \cdot \|_A$. Here the Ritz–Galerkin approach coincides with the minimum norm approach with respect to $\| \cdot \|_A$.

Figure 3.4 displays the relative residual curve for the spd matrix (1.2) for n=50, i.e. $N = 2500$. Here *tol* $= 0.0001$ and $x_0 = 0$. CG converges nicely. The realtive residual drops below *tol* within 70 steps of the Iteration.

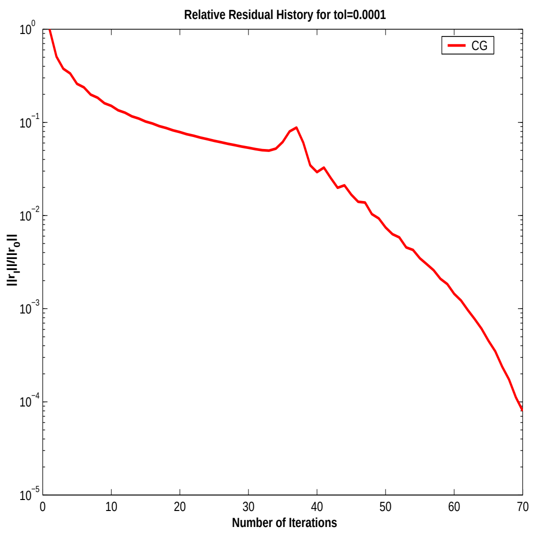


Figure 3.4: CG for Laplace(50)

The CG–method was actually the first Krylov subspace method. It was introduced in 1952 by Hestenes and Stiefel [6] and has had a large influence on the developments of scientific computing, see [4] for historical perspectives.

3.3 BiCG and QMR

For unsymmetric linear systems we can apply GMRES. But as mentioned already the main problem is that the computational costs are growing with the number of iterations. At step l the number of operations is of order lN, because we need all the preceding basis vectors $q_1, \dots, q_l$. We would prefer to have only to deal with two previous basis vectors like in the MINRES or CG method. The short recursion for computing $q_1, \dots, q_l$ was due to the fact that in $AQ_m = Q_m T_m$ the matrix T_m was tridiagonal. If we want such a short recursion also for unsymmetric matrices A, then we have to give up the orthogonality of Q_m. We can try to develop a corresponding recursive computation of basis vectors $v_1, \dots, v_l$ for $\mathcal{K}_l(A, r_0)$ as before from a similarity transformation of A to tridiagonal form like in the Lanczos process, where we only require linear independence of the vectors $v_1, \dots, v_l$. The following theorem is the basis for such a process and corresponds to Theorem 3.

Let $s_0, r_0 \in \mathbb{R}^N \backslash \{0\}$. For ease of notation we will assume that $\mathcal{K}_N(A, r_0)$ and $\mathcal{K}_N(A^T, s_0)$ are nonsingular.

Theorem 7

Let the assumptions and notations above be given.

The following statements are equivalent

(a) *$K_N(A^T, s_0)^T K_N(A, r_0)$ has an LU-decomposition*

$$K_N(A^T, s_0)^T K_N(A, r_0) = LU = \begin{bmatrix} 1 & & \\ \vdots & \ddots & \\ \cdots & \cdots & 1 \end{bmatrix} \begin{bmatrix} \cdots & \cdots & \cdots \\ & \ddots & \vdots \\ & & \cdot \end{bmatrix}.$$

(b) *There exists a nonsingular matrix $V \in \mathbb{R}^{N \times N}$ and $\delta_1, \delta_2 \in \mathbb{R} \backslash \{0\}$ such that*

$$V e_1 = \delta_1 r_0 \text{ and } e_1^T V^{-1} = \delta_2 s_0^T$$

and

$$V^{-1} A V = T = \begin{bmatrix} \ddots & \ddots & \\ \ddots & \ddots & \ddots \\ & \ddots & \ddots \end{bmatrix} = (t_{ij})_{i,j \in \{1,\dots,N\}}$$

is a tridiagonal matrix with non zero subdiagonal elements $t_{j+1,j}$ and $t_{j,j+1}$ for all j.

Proof

$$K_N(A^T, s_0)^T K_N(A, r_0) = (s_0^T A^{i+j-2} r_0)_{i,j \in \{1,\dots,N\}}.$$

This matrix is obviously symmetric and if it has an LU decomposition

$$K_N(A^T, s_0)^T K_N(A, r_0) = LU$$

then with $D = \mathrm{diag}(\mathrm{u}_{11}, \dots, \mathrm{u}_{\mathrm{NN}})$ we have

$$K_N(A^T, s_0) K_N(A, r_0) = LD\tilde{U} \text{ where } \tilde{U} = \begin{bmatrix} 1 & \cdots & \cdots \\ & \ddots & \vdots \\ & & 1 \end{bmatrix}.$$

Because $LD\tilde{U} = \tilde{U}^T DL^T$ it follows that $\tilde{U} = L^T$. Define

$$V := K_N(A, r_0)L^{-T}.$$

Then

$$K_N(A, r_0) = VL^T = V \left[{}^{1}\overline{\diagdown}\!\!\!\mid_{1} \right] \tag{3.13}$$

and

$$\begin{aligned} K_N(A^T, s_0)^T = (LDL^T K_N(A, r_0)^{-1})^T &= K_N(A, r_0)^{-T} LDL^T \\ &= V^{-T} DL^T = V^{-T} \left[\overline{\diagdown}\!\!\!\mid \right]. \end{aligned} \tag{3.14}$$

Because $K_N(A, r_0)$ is nonsingular it follows that $r_0, Ar_0, \ldots, A^{N-1}r_0$ is a basis for $\mathbb{R}^N$. Thus there exist $c_0, \ldots, c_{N-1} \in \mathbb{R}$ such that

$$A^N r_0 = \sum_{j=0}^{N-1} c_j A^j r_0.$$

Like in Theorem 3 it follows from (3.13)

$$AV = AK_N(A, r_0)L^{-T} = K_N(A, r_0)CL^{-T} = VL^T CL^{-T}$$

where $C = \begin{bmatrix} 0 & & & c_0 \\ 1 & & & \vdots \\ & \ddots & & \vdots \\ & & 1 & c_{N-1} \end{bmatrix}$ and therefore $H = L^T CL^{-T} = \left[\overline{\diagdown\!\diagdown}\!\!\!\mid \right]$. Likewise from (3.14) it follows

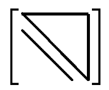
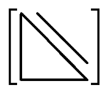
$$\begin{aligned} A^T V^{-T} = A^T K_N(A^T, s_0)^T L^{-T} D^{-1} &= K_N(A^T, s_0)^T \tilde{C} L^{-T} D^{-1} \\ &= V^{-T} DL^T \tilde{C} L^{-T} D^{-1} \end{aligned}$$

where $\tilde{C} = \begin{bmatrix} 0 & & & \tilde{c}_0 \\ 1 & & & \vdots \\ & \ddots & & \vdots \\ & & 1 & \tilde{c}_{N-1} \end{bmatrix}$ and therefore $\tilde{H} = DL^T \tilde{C} L^{-T} D^{-1} = \left[\overline{\diagdown\!\diagdown}\!\!\!\mid \right]$.

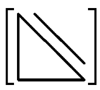
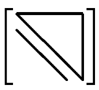
Thus we have on one hand

$$V^{-1}AV = H = \left[\overline{\diagdown\!\diagdown}\!\!\!\mid \right]$$

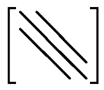
and on the other

$$V^TA^TV^{-T} = \tilde{H} =$$  $$= (V^{-1}AV)^T = H^T =$$  .

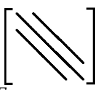
From

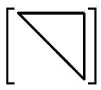
$$H^T =$$  $$= \tilde{H} =$$ 

it follows that

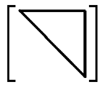
$$H =$$  $$=: T.$$

The subdiagonal entries of $H = L^TCL^{-T}$ are all 1 and of $\tilde{H} = DL^T\tilde{C}L^{-T}D^{-1}$ are all non zero. Therefore T's subdiagonal entries $t_{j+1,j}$ and $t_{j,j+1}$ are all non zero. From (3.13) we see that $Ve_1 = r_0$ and from (3.14) $e_1^TV^{-1} = \delta s_0^T$ for a $\delta \in \mathbb{R}\backslash\{0\}$.

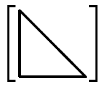
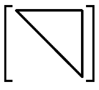
If on the other hand $V^{-1}AV = T =$  with $Ve_1 = \delta_1 r_0$ and $e_1^TV^{-1} = \delta_2 s_0^T$, then $A^l = VT^lV^{-1}$ and $A^{l^T} = V^{-T}T^{l^T}V^T$ for all $l \in \mathbb{N}$.
Then

$$\begin{aligned} K_N(A, r_0) &= [r_0, Ar_0, \ldots, A^{N-1}r_0] \\ &= \underbrace{V\frac{1}{\delta_1}[e_1, Te_1, \ldots, T^{N-1}e_1]}_{=U} = VU = V \end{aligned} \qquad (3.15)$$ 

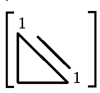
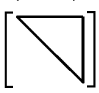
and

$$\begin{aligned} K_N(A^T, s_0) &= [s_0, A^Ts_0, \ldots, A^{N-1^T}s_0] \\ &= \underbrace{V^{-T}\frac{1}{\delta_2}[e_1, T^Te_1, \ldots, T^{N-1^T}e_1]}_{=\tilde{U}} = V^{-T}\tilde{U} = V^{-T} \end{aligned} \qquad (3.16)$$  .

Therefore

$$K_N(A^T, s_0)^TK_N(A, r_0) = \tilde{U}^TV^{-1}VU = \tilde{U}^TU =$$  

and with $\tilde{D} = \mathrm{diag}(\tilde{\mathrm{u}}_{11}, \ldots, \tilde{\mathrm{u}}_{\mathrm{nn}})$ we get

$$\begin{aligned} K_N(A^T, s_0)^TK_N(A, r_0) &= (\tilde{U}^T\tilde{D}^{-1})(\tilde{D}U) \\ &= \end{aligned}$$   $\square$

The theorem tells us that a similarity transformation of A to tridiagonal form is possible if and only if all principal submatrices of the symmetric matrix $K_N(A^T, s_0)^T K_N(A, r_0)$ are non zero. (This is the characterization for the existence of an LU decomposition.) Whether this condition is satisfied or not depends on the choice of s_0 and r_0. If for example we choose $s_0 \perp r_0$ then the $(1,1)$ element in $K_N(A^T, s_0)^T K_N(A, r_0)$, which is the first principal submatrix, is zero.

It can be shown that the set of vector pairs (s_0, r_0) for which $K_N(A^T, s_0) K_N(A, r_0)$ has no LU decomposition is of measure zero. For numerical purposes, however, this insight does not help very much, because in the computations we have also problems if (s_0, r_0) is close to a pair of vectors, for which the product of Krylov matrices does not have an LU decomposition, as we will see later.

There is no method of choosing "good" starting vectors which does not need complete information on A's eigenvalue and eigenvector structure. But eigenvector and eigenvalue computation is a much more difficult problem than solving the linear system $Ax = b$.

For the following development we will choose $r_0 = b - Ax_0$ and $s_0 = r_0$ and assume that $K_N(A^T, r_0)^T K_N(A, r_0)$ has all principal submatrices nonzero. Then from the theorem we know that there exists a nonsingular $V \in \mathbb{R}^{N \times N}$, such that

$$Ve_1 = \delta_1 r_0 \quad \text{and} \quad V^{-T} e_1 = \delta_2 r_0 \quad \text{and} \tag{3.17}$$
$$V^{-1}AV \quad = \quad T, \tag{3.18}$$

Denote $V = [v_1, \dots v_N]$ and $Y := V^{-T} = [y_1, \dots y_N]$

Then from (3.15) and (3.16) we see that

$$v_1, \dots v_l \quad \text{span } \mathcal{K}_l(A, r_0) \qquad \text{for all } l \in \{1, \dots N\} \text{ and} \tag{3.19}$$
$$y_1, \dots, y_l \quad \text{span } \mathcal{K}_l(A^T, r_0) \qquad \text{for all } l \in \{1, \dots N\}. \tag{3.20}$$

Note that (3.17), (3.18),(3.19) and (3.20) still hold, if we replace T by $D^{-1}TD$ and V by VD, where D is a nonsingular diagonal matrix, i.e. there is freedom in the way we scale the off diagonal entries of T by a diagonal similarity transformation. Let us use this freedom here to get $|t_{j,j+1}| = t_{j+1,j}$ for $j \in \{1, \dots, N-1\}$ and $\| v_1 \|_2 = \| y_1 \|_2 = 1$.

Then we can summarize the properties for the scaled T and V and $Y = V^{-T}$ as follows:

$$Ve_1 \ = \ \frac{1}{\| r_0 \|_2} r_0, Ye_1 = \frac{1}{\| r_0 \|_2} r_0 \tag{3.21}$$
$$Y^T V \ = \ I_N \tag{3.22}$$

$$
\begin{aligned}
Y^T A V &= T = \begin{bmatrix} \alpha_1 & \beta_2 & & \\ \gamma_2 & \ddots & \ddots & \\ & \ddots & \ddots & \beta_N \\ & & \gamma_N & \alpha_N \end{bmatrix}, |\beta_j| = \gamma_j && (3.23) \\
AV &= VT && (3.24) \\
A^T Y &= Y T^T && (3.25)
\end{aligned}
$$

$v_1, \dots, v_l$ span $\mathcal{K}_l(A, r_0)$ and $y_1, \dots, y_l$ span $\mathcal{K}_l(A^T, r_0)$.
Evaluating these equations column by column we get

from (3.22): $v_1 = \frac{1}{\| r_0 \|_2} r_0 \quad y_1 = \frac{1}{\| r_0 \|_2} r_0$

from (3.23): $\alpha_1 = y_1^T A v_1$.

Assume that we have already computed $v_1, \dots, v_l, y_1, \dots, y_l$ and $\alpha_1, \dots, \alpha_l, \beta_1, \dots, \beta_l, \gamma_1, \dots, \gamma_l$ for $l \geq 1$, where we set $\beta_1 = \gamma_1 = \| r_0 \|_2$ and $v_0 = y_0 = 0$, then we get

from (3.25): $A v_l = \beta_l v_{l-1} + \alpha_l v_l + \gamma_{l+1} v_{l+1}$

from (3.25): $A^T y_l = \gamma_l y_{l-1} + \alpha_l y_l + \beta_{l+1} y_{l+1}$.

We can compute $u_{l+1} := \gamma_{l+1} v_{l+1} = A v_l - \beta_l v_{l-1} - \alpha_l v_l$
$w_{l+1} := \beta_{l+1} y_{l+1} = A^T y_l - \gamma_l y_{l-1} - \alpha_l y_l$

and from (3.23) : $w_{l+1}^T u_{l+1} = \gamma_{l+1} \beta_{l+1} y_{l+1}^T v_{l+1} = \gamma_{l+1} \beta_{l+1}$.

Because $\gamma_{l+1} > 0$ and $|\beta_{l+1}| = \gamma_{l+1}$ we know that

$$\gamma_{l+1} = \sqrt{|w_{l+1}^T u_{l+1}|} \quad \text{and} \quad \beta_{l+1} = \operatorname{sgn}(w_{l+1}^T u_{l+1}) \gamma_{l+1}$$

and we get

$$v_{l+1} = \frac{u_{l+1}}{\gamma_{l+1}} \text{ and } y_{l+1} = \frac{w_{l+1}}{\beta_{l+1}}.$$

This leads to the unsymmetric Lanczos method.

Unsymmetric Lanczos method:

```
$u = w = r_0$
$\beta_1 = \gamma_1 = \| r_0 \|_2, v_0 = y_0 = 0$
$l = 0$
while $\beta_{l+1}\gamma_{l+1} \neq 0$
    $v_{l+1} = \frac{1}{\gamma_{l+1}} u, y_{l+1} = \frac{1}{\beta_{l+1}} w$
    $l = l+1$
    $u = Av_l \quad w = A^T y_l$
    $\alpha_l = y_l^T u$
    $u = u - \beta_l v_{l-1} - \alpha_l v_l$
    $w = w - \gamma_l y_{l-1} - \alpha_l y_l$
    $\delta = w^T u$
    $\gamma_{l+1} = \sqrt{|\delta|}, \beta_{l+1} = sgn\ (\delta)\gamma_{l+1}$
end
```

Comparing this procedure with the Lanczos method, we see that the computational steps look very similar, but here each iteration requires twice as many computations, in particular we have to compute **2 matrix vector products** Av_l and $A^T y_l$ in each step.

Note that in step l we divide by β_{l+1} and γ_{l+1}. β_{l+1} and γ_{l+1} are zero for $\delta = w^T u = 0$. It is easy to see that if $u = 0$ or $w = 0$ in the step, where β_{l+1} and γ_{l+1} are computed, then $v_1, \ldots, v_l$ or $y_1, \ldots, y_l$ span an invariant subspace of A or A^T, respectively. We will have a breakdown in our computation, because we cannot go on dividing by β_{l+1} and γ_{l+1}, but we have an invariant subspace, which contains the solution of $A\tilde{x} = r_0$. This is called a "lucky breakdown". But it is also possible that $w \neq 0$ and $u \neq 0$ but $\delta = w^T u = 0$. Then we cannot go on with our computation. Here we do not get an invariant subspace of A or A^T and the information we have computed so far does not enable us to compute the solution of $A\tilde{x} = r_0$. This is called a "serious breakdown". A closer look at the proof of Theorem (7) can show that this happens if and only if the $(l+1)$st principal submatrix of $K_N(A^T, r_0)^T K_N(A, r_0)$ is zero, i.e. there is no LU–decomposition.

If $|\beta_{l+1}|$ and γ_{l+1} are very small compared to the norm of the vectors u and v from which they are computed, then the vectors v_{l+1} and y_{l+1} will have very large entries and then the results will be contaminated by round-off errors.

The price that we have to pay for using the basis $v_1, \ldots, v_l$ of $\mathcal{K}_l(A, r_0)$ with the short recursion is this danger of possible serious breakdowns or nearly serious breakdowns. In practice this method works often reasonably well.

So let us assume we get through with this computation. Then we may use a Petrov-Galerkin approach to compute approximations to $A\tilde{x} = r_0$. A suitable second subspace is obviously $\mathcal{K}_l(A^T, r_0)$ because we have $y_1, \ldots, y_l$ as its basis. According to (1.11) we then compute $z_l \in \mathbb{R}^l$ such that with $Y_l = [y_1, \ldots, y_l]$ and $V_l = [v_1, \ldots, v_l]$

$$Y_l^T A V_l z_l = T_l z_l = Y_l^T r_0 = \| r_0 \|_2 \, Y_l^T V_l e_1 = \| r_0 \|_2 \, e_1.$$

Thus we have to solve the $l \times l$ system

$$T_l z_l = \| r_0 \| \, e_1. \tag{3.26}$$

If we try to stay as close as possible to the CG–method, then we would use a Cholesky–type decomposition of the form

$$T_l = B_l \Omega_l B_l^T, \tag{3.27}$$

where B_l is a bidiagonal matrix and Ω_l is a diagonal matrix. Such decompositions exist for all $l \in \{1, \ldots, N\}$ if and only if T_N has an LU–decomposition, i.e. if all leading principal submatrices are nonsingular.

Note that if we follow this way of solving (3.26) we have here an additional source of possible breakdowns.

But if we do so, then we can also use exactly the same modification as in Theorem 5. The derivation is completely analogous. In analogy to the CG Lanczos method we get here a method with four sets of vectors instead of two. This method is called BiCG method.

```
BiCG:
    Chooose an initial guess $x_0$
    Compute $r_0 = b - Ax_0$
    Set $w_1 = y_1 = \tilde{w}_1 = \tilde{y}_1 = r_0$
    $l = 1$
        while $w_l \neq 0$
            $\alpha_l = \frac{\tilde{w}_l^T w_l}{\tilde{y}_l^T A y_l}$
            $x_l = x_{l-1} + \alpha_l y_l$
            $w_{l+1} = w_l - \alpha_l A y_l$
            $\tilde{w}_{l+1} = \tilde{w}_l - \alpha_l A^T y_l \tilde{y}$
            $\beta_l = \frac{\tilde{w}_{l+1}^T w_{l+1}}{\tilde{w}_l^T w_l}$
            $y_{l+1} = w_{l+1} + \beta_l y_l$
            $\tilde{y}_{l+1} = \tilde{w}_{l+1} + \beta_l \tilde{y}_l$
            $l = l + 1$
        end
```

It can easily be shown that

$$w_l = b - Ax_{l-1} = r_{l-1} \quad \text{for} \quad l = 1, 2, \ldots$$

The two types of possible breakdowns show in the following way. If $K_N(A^T, r_0)^T K_N(A, r_0)$ has a singular leading principal submatrix then $\tilde{w}_l^T w_l = 0$ for a corresponding l while $\tilde{w}_l \neq 0$ and $w_l \neq 0$.

If the tridiagonal T_N has a singular leading principal submatrix, then one of the factors $\tilde{y}_l^T A y_l$ will be zero.

Instead of the Petrov–Galerkin approach we could think of a minimum residual approach. In this case we compute that same vectors in $V_l = [v_1, \ldots, v_l]$ and $W_l = [w_1, \ldots, w_l]$ for $l = 1, 2, \ldots$ with the unsymmetric Lanczos method and then have to solve the least squares problem

$$\min_{z \in \mathbb{R}^l} \| AV_l z - r_0 \|_2 .$$

But unfortunately this does not correspond to a least squares problem with parts of the tridiagonal matrix T_N that we have at hand. If we want to use the T_l's , then we could compute z_l as the solution of the least squares problem

$$\min_{z \in \mathbb{R}^l} \| T_{l+1,l} z_l - \| r_0 \|_2 e_1 \|_2,$$

where as before $T_{l+1,l} = \begin{bmatrix} T_l & \\ 0 \ldots 0 & \beta_{l+1} \end{bmatrix}$. This is called a quasi–minimization of the residual and the method is known as QMR.

Figure 3.5 displays the relative residual curve for lns131, a smaller dimensional version of lns511. Here $tol = 0.001$ and $x_0 = 0$. The erratic behavior of the BiCG residuals is typical.

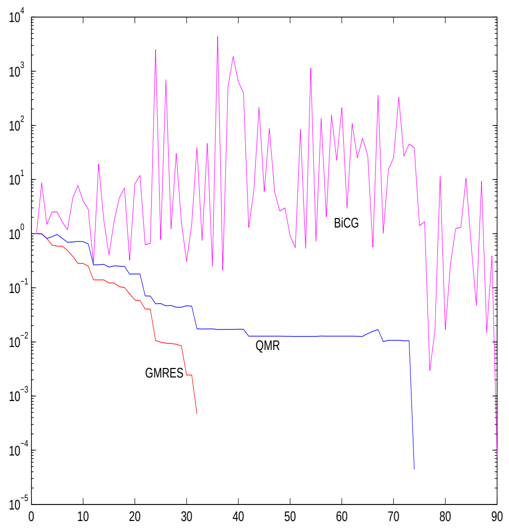


Figure 3.5: BiCG and QMR for lns131

There are many more Krylov subspace methods and the choice of a suitable one is not easy and depends on the class problem that has to be treated. A good assesment of the major algorithms is given in [8].

Chapter 4

Preconditioning

In many cases the basic iterative methods converge too slowly or even fail to converge. Studying the convergence of the methods one finds, roughly speaking, that the convergence is very good for matrices which have eigenvalues clustered in principle around 1. In Chapter 1 we saw that the minimum residual approach allowed an upper bound for the residual (2.10):

$$\| r_l \|_2 \;=\; \| q(A) r_0 \|_2 \;\leq\; \| X \|_2 \;\; \| X^{-1} \|_2 \;\; \| r_0 \|_2 \;\; \min_{\tilde{q} \in \hat{\Pi}_l} \; \max_{\lambda \in \{\lambda_1, \ldots, \lambda_n\}} |\tilde{q}(\lambda)|.$$

This equation is the starting point for a number of results on upper bounds for the residuals. It can, for example, be shown that if we have an unstructured A for which the symmetric matrix $A^T + A$ is positive definite then this inequality can be used to proof that for the residuals in GMRES we have

$$\| r_l \|_2 \;\leq\; \left(1 - \frac{\lambda_{\min} \left(\frac{A^T + A}{2} \right)^2}{\lambda_{\max} \left(A^T A \right)} \right)^{\frac{l}{2}} \| r_0 \|_2,$$

where $\lambda_{\min}(M)$ and $\lambda_{\max}(M)$ denote the minimal and maximal eigenvalue of the symmetric matrix M, respectively. In particular for A spd this bound specializes to

$$\| r_l \|_2 \;\leq\; \left(\frac{\mathrm{cond}_2(A)^2 - 1}{\mathrm{cond}_2(A)^2} \right)^{\frac{l}{2}} \| r_0 \|_2,$$

where $\mathrm{cond}_2(M)$ denotes the condition number $\| M \|_2 \| M^{-1} \|_2$ of M. There are similar bounds for the residuals in other Krylov subspace methods. Such bounds indicate that the residuals shrink faster if the eigenvalues of A are clustered around one point or the condition number is close to one.

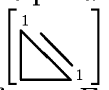
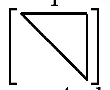
Therefore the iterative methods are combined with preconditioning, i.e. instead of $Ax = b$ one solves $M^{-1}Ax = M^{-1}b$ or $AM^{-1}y = b$, where $M^{-1}A$ or AM^{-1} have better convergence properties for the iterative method. M^{-1} is called a preconditioner of A. We could try to find a suitable matrix M or M^{-1} directly. If we have M, then we would of course not form M^{-1} explicitely. In the matrix vector products we need in the iterative methods, we would compute $z = M^{-1}y$ by solving the linear system $y = Mz$.

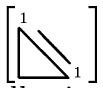
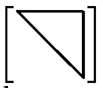
M or M^{-1} should be such that

- M^{-1} is close to A^{-1} in some sense
- computing M or M^{-1} is not too expensive
- the system $My = z$ is much easier to solve than the original system (for the case that we compute M)

Applying a good preconditioner is crucial for the success of an iterative method. Here we will focus on two choices of M which are based only an algebraic techniques and work for general matrices.

4.1 Incomplete LU Preconditioning

The standard Gaußian elimination for A is equivalent to the computation of the LU decomposition $A = LU$ of A, where $L =$  and $U =$ . L and U are easy to invert, because of their triangular form. For large sparse A the computation of the exact L and U is prohibited because of the arising fill-in. The basic idea in the incomplete LU preconditioning (ILU preconditioning) is to use L and U but skip some parts of the computations and the corresponding entries in L and U

If $A = LU$ with $L =$ , $U =$ , the entries of L and U can be computed successively by the following formulas:

$$\begin{aligned} u_{ij} &= a_{ij} - \sum_{k=1}^{i-1} l_{ik} \\ l_{ij} &= \tfrac{1}{u_{jj}} \left(a_{ij} - \sum_{k=1}^{j-1} u_{kj} l_{ik} \right). \end{aligned}$$

The $ILU(0)$ preconditioning computes these quantities with the constraint that entries of L and U whose position corresponds to zero entries in A are ignored (set to zero):

$$a_{ij} = 0 \Rightarrow l_{ij} = u_{ij} = 0.$$

The number 0 in $ILU(0)$ indicates that in this case we allow no fill–in in the approximate LU factors.

A simple implementation can be derived if we use (a copy of) the matrix A and overwrite it stepwise by the entries of L and U.

```
ILU (0)
    FOR $i = 2, \dots, N$
        FOR $k = 1, \dots, i-1$
            IF $a_{ik} \neq 0$
                $a_{ik} := a_{ik}/a_{kk}$
            END
            FOR $j = k+1, \dots, N$
                IF $a_{ij} \neq 0$
                    $a_{ij} = a_{ij} - a_{ik}a_{kj}$
            END
        END
    END
```

The entries of L and U are then in the lower and upper triangular part of (the copy of) A. This preconditioning is easy to implement but often there are stability problems.

We could try to improve the situation by allowing a moderate amount of fill–in. One might for instance allow in addition to the sparsity pattern of A that p co–diagonals on both sides of the main diagonal fill–in. This is often denoted by $ILU(p)$. The computation is still relatively simple, e.g.

```
ILU (1)
    FOR $i = 2, \dots, N$
        FOR $k = 1, \dots, i-1$
            IF $a_{ik} \neq 0$
                $a_{ik} = a_{ik}/a_{kk}$
            END
            FOR $j = k+1, \dots, N$
                IF $a_{ij} \neq 0$ OR $a_{ik}a_{kj} \neq 0$
                    $a_{ij} = a_{ij} - a_{ik}a_{kj}$
                END
            END
        END
    END
```

Another variant is the incomplete LU–factorization with numerical dropping. There we choose a drop tolerance factor $\epsilon \ll 1$ and do not compute entries in L and U whose absolute values decrease more than by this factor. This variant is often denoted by ILUT.

```
ILUT:
    FOR $i = 1, \ldots, N$
        $w = (a_{i1}, \ldots, a_{iN}), \epsilon_i = \epsilon \parallel w \parallel$
        FOR $k = 1, \ldots, i-1$
            IF $w_k \neq 0$
                $w_k = a_{ik}/u_{kk}$
                IF $|w_k| \leq \epsilon_i$
                    $w_k = 0$
                END
            END
            IF $w_k \neq 0$
                FOR $j = k+1, \ldots, N$
                    IF $u_{kj} \neq 0$
                        $w_j = w_j - w_k u_{kj}$
                    END
                END
            END
        FOR $j = 1, \ldots, i-1$
            IF $w_j \neq 0$
                IF $|w_j| > \epsilon_i$
                    $l_{ij} = w_j$
                END
            END
        END
        FOR $j = i, \ldots, N$
            IF $w_j \neq 0$
                IF $|w_j| > \epsilon_i$
                    $u_{ij} = w_j$
                END
            END
        END
        $w = 0$
    END
```

There are many variants of incomplete triangular decompositions and choosing a suitable preconditioner together with a suitable Krylov subspace method is not all easy.

Depending on the problem one might also need column permutations in the ILU preconditioning, which makes the procedure more complicated. The methods still have numerous problems. But their advantage is the easy implementation.

Figure 4.2 shows the residual curves for various Krylov subspace methods with ILU preconditioner. Compare with Figure 3.1 to see the big improvement in the convergence for GMRES(10).

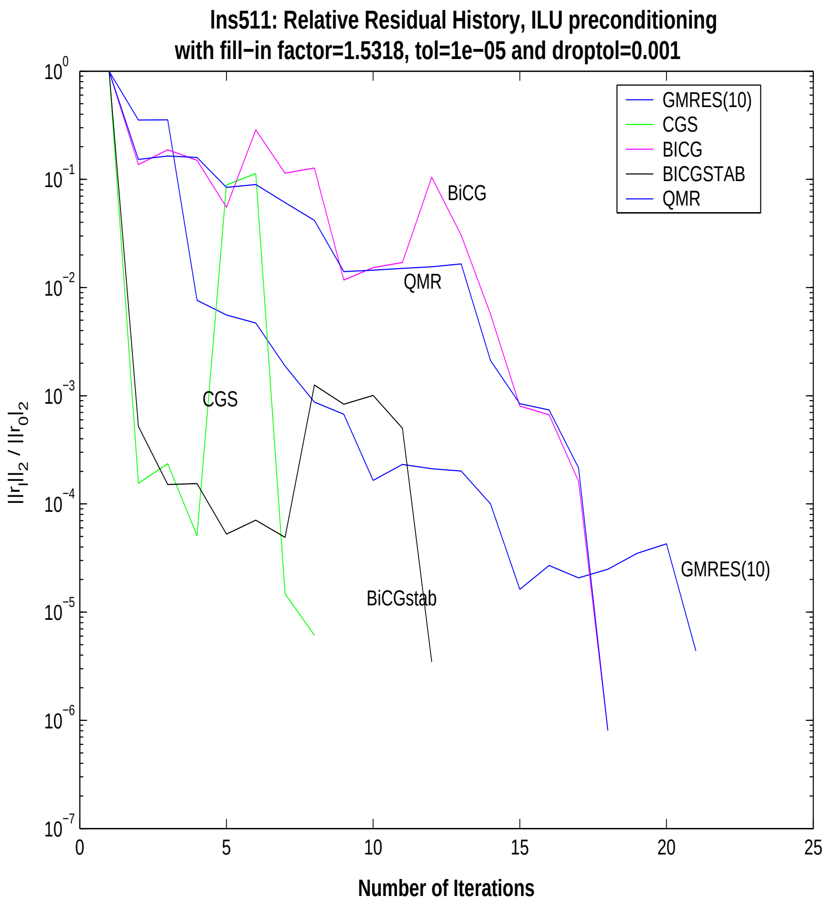


Figure 4.1: lns511 with ILU(0)

4.2 Sparse Approximative Inverse

One of the severe drawbacks of ILU is the fact that the computation of the factors is very sequential. But for large systems one would also like to perform the iterative solution method on parallel computers. There are a number of techniques to get parallelizable preconditioners. We will here only briefly mention the concept of one of them, the SPAI (**sp**arse **a**pproximative **i**nverse). The idea is here to compute a matrix M^{-1} with $M^{-1} \approx A^{-1}$ directly as the solution of the optimization problem

$$\min_{M \in \mathcal{S}} \| AM - I \|_F,$$

where $\| C \|_F = \sqrt{\sum_{i,j=1}^{N} c_{ij}^2}$ is the Frobenius norm of a matrix $C \in \mathbb{R}^{N \times N}$ and $\mathcal{S}$ is the set of all matrices with a prescribed sparsity pattern.

The following observation shows that this optimization performs nicely. Let $A = [a_1, \ldots, a_N]$ and denote $M = (m_{ij}) = [m_1, \ldots, m_N]$. Then

$$\min \| AM - I \|_F^2 = \min \sum_{j=1}^{N} \| Am_j - e_j \|_F^2 = \sum_{j=1}^{N} (\min \| Am_j - e_j \|_F^2),$$

i.e. the minimization can be done for each column independently.

If $\mathcal{I}_j$ is the index set of prescribed non zero entries in the j–th column of M, then $m_j = \sum_{i \in \mathcal{I}_j} m_{ij} e_j$ and therefore

$$\min \| Am_j - e_j \|_F^2 = \min \| \sum_{i \in \mathcal{I}_j} a_i m_{ij} - e_j \|_F^2 .$$

The solution of the problem requires only the columns $a_i, i \in \mathcal{I}_j$, of A to compute m_j. This is highly parallelizable.

If we choose a sparsity pattern for M, e.g. the sparsity pattern of A, A^T or $|A| + |A|^T$ then for all $j \in \{1, \ldots, N\}$ the set $\mathcal{I}_j$ is defined and we have to solve a problem of the form

$$\min_x \| A_j x - e_j \|_2^2$$

where A_j contains the columns a_k of A from $\mathcal{I}_j$. $x \in \mathbb{R}^l$ corresponds to the m_{ij} and therefore l is the number of indices in $\mathcal{I}_j$. This is a simple least squares problem that we can solve using the QR decomposition of A_j or via the normal equations.

Figure 4.2 displays the relative residual curve for lns511 for various Krylov subspace methods using a SPAI preconditioner. Here it is only GMRES that works well with the preconditioner.

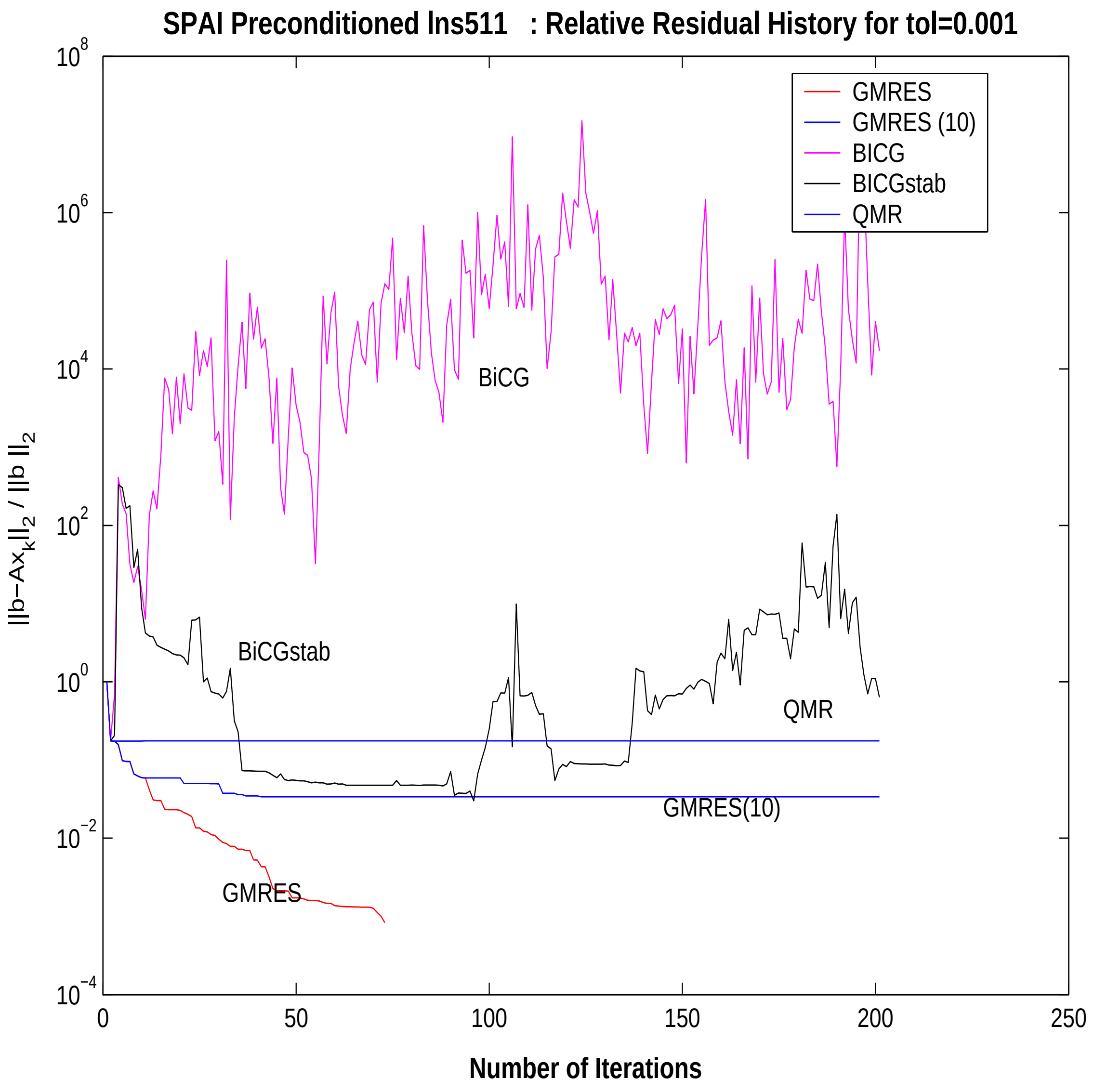


Figure 4.2: lns511 with SPAI

Bibliography

[1] C.F. Gauß. *Werke, Band IX.* Teubner, Leipzig, 1903.

[2] A. George and J.W-H. Liu. *Computer Solution of Large Sparse Positive Definite Systems.* Prentice Hall Inc., Englewood Cliffs, New Jersey, 1981.

[3] G.H. Golub and C.F. Van Loan. *Matrix Computations.* John Hopkins, Baltimore, 1996.

[4] G.H. Golub and D. O'Leary. Some history of the conjugate gradient and lanczos method. *SIAM Revies*, 31:50 – 102, 1989.

[5] L.A. Hageman and D.M. Young. *Applied Iterative Methods.* Academic Press, New York, 1981.

[6] M.R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Res. Nat. Bur. Stand.*, 49:409 – 36, 1952.

[7] A.M. Erisman I.S. Duff and J.K. Reid. *Direct Methods for Sparse Matrices.* Oxford University Press, London, 1986.

[8] T.F. Chan J. Demmel J. Donato J. Dongarra V. Eijkhout R. Pozo C. Ronnie R. Barret, M. Berry and H. van der Vorst. *Templates for the Solution of Linear Systems.* SIAM Publictions, Philadelphia, PA, 1993.

[9] Y. Saad and M.H. Schultz. A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Scientific and Stat.Comp.*, 7:856 – 869, 1986.

[10] R.S. Varga. *Matrix Iterative Analysis.* Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1962.

[11] D.M. Young. *Iterative Solution of Large Linear Systems.* Academic Press, New York, 1971.

Berichte aus der Technomathematik ISSN 1435-7968

http://www.math.uni-bremen.de/zetem/berichte.html

— Vertrieb durch den Autor —

Reports **Stand: 11. Dezember 2002**

98–01. Peter Benner, Heike Faßbender:
An Implicitly Restarted Symplectic Lanczos Method for the Symplectic Eigenvalue Problem, Juli 1998.

98–02. Heike Faßbender:
Sliding Window Schemes for Discrete Least-Squares Approximation by Trigonometric Polynomials, Juli 1998.

98–03. Peter Benner, Maribel Castillo, Enrique S. Quintana-Ortí:
Parallel Partial Stabilizing Algorithms for Large Linear Control Systems, Juli 1998.

98–04. Peter Benner:
Computational Methods for Linear–Quadratic Optimization, August 1998.

98–05. Peter Benner, Ralph Byers, Enrique S. Quintana-Ortí, Gregorio Quintana-Ortí:
Solving Algebraic Riccati Equations on Parallel Computers Using Newton's Method with Exact Line Search, August 1998.

98–06. Lars Grüne, Fabian Wirth:
On the rate of convergence of infinite horizon discounted optimal value functions, November 1998.

98–07. Peter Benner, Volker Mehrmann, Hongguo Xu:
A Note on the Numerical Solution of Complex Hamiltonian and Skew-Hamiltonian Eigenvalue Problems, November 1998.

98–08. Eberhard Bänsch, Burkhard Höhn:
Numerical simulation of a silicon floating zone with a free capillary surface, Dezember 1998.

99–01. Heike Faßbender:
The Parameterized SR Algorithm for Symplectic (Butterfly) Matrices, Februar 1999.

99–02. Heike Faßbender:
Error Analysis of the symplectic Lanczos Method for the symplectic Eigenvalue Problem, März 1999.

99–03. Eberhard Bänsch, Alfred Schmidt:
Simulation of dendritic crystal growth with thermal convection, März 1999.

99–04. Eberhard Bänsch:
Finite element discretization of the Navier-Stokes equations with a free capillary surface, März 1999.

99–05. Peter Benner:
Mathematik in der Berufspraxis, Juli 1999.

99–06. Andrew D.B. Paice, Fabian R. Wirth:
Robustness of nonlinear systems and their domains of attraction, August 1999.

99–07. Peter Benner, Enrique S. Quintana-Ortí, Gregorio Quintana-Ortí:
Balanced Truncation Model Reduction of Large-Scale Dense Systems on Parallel Computers, September 1999.

99–08. Ronald Stöver:
Collocation methods for solving linear differential-algebraic boundary value problems, September 1999.

99–09. Huseyin Akcay:
Modelling with Orthonormal Basis Functions, September 1999.

99–10. Heike Faßbender, D. Steven Mackey, Niloufer Mackey:
Hamilton and Jacobi come full circle: Jacobi algorithms for structured Hamiltonian eigenproblems, Oktober 1999.

99–11. Peter Benner, Vincente Hernández, Antonio Pastor:
On the Kleinman Iteration for Nonstabilizable System, Oktober 1999.

99–12. Peter Benner, Heike Faßbender:
A Hybrid Method for the Numerical Solution of Discrete-Time Algebraic Riccati Equations, November 1999.

99–13. Peter Benner, Enrique S. Quintana-Ortí, Gregorio Quintana-Ortí:
Numerical Solution of Schur Stable Linear Matrix Equations on Multicomputers, November 1999.

99–14. Eberhard Bänsch, Karol Mikula:
Adaptivity in 3D Image Processing, Dezember 1999.

00–01. Peter Benner, Volker Mehrmann, Hongguo Xu:
Perturbation Analysis for the Eigenvalue Problem of a Formal Product of Matrices, Januar 2000.

00–02. Ziping Huang:
Finite Element Method for Mixed Problems with Penalty, Januar 2000.

00–03. Gianfrancesco Martinico:
Recursive mesh refinement in 3D, Februar 2000.

00–04. Eberhard Bänsch, Christoph Egbers, Oliver Meincke, Nicoleta Scurtu:
Taylor-Couette System with Asymmetric Boundary Conditions, Februar 2000.

00–05. Peter Benner:
Symplectic Balancing of Hamiltonian Matrices, Februar 2000.

00–06. Fabio Camilli, Lars Grüne, Fabian Wirth:
A regularization of Zubov's equation for robust domains of attraction, März 2000.

00–07. Michael Wolff, Eberhard Bänsch, Michael Böhm, Dominic Davis:
Modellierung der Abkühlung von Stahlbrammen, März 2000.

00–08. Stephan Dahlke, Peter Maaß, Gerd Teschke:
Interpolating Scaling Functions with Duals, April 2000.

00–09. Jochen Behrens, Fabian Wirth:
A globalization procedure for locally stabilizing controllers, Mai 2000.

00–10. Peter Maaß, Gerd Teschke, Werner Willmann, Günter Wollmann:
Detection and Classification of Material Attributes – A Practical Application of Wavelet Analysis, Mai 2000.

00–11. Stefan Boschert, Alfred Schmidt, Kunibert G. Siebert, Eberhard Bänsch, Klaus-Werner Benz, Gerhard Dziuk, Thomas Kaiser:
Simulation of Industrial Crystal Growth by the Vertical Bridgman Method, Mai 2000.

00–12. Volker Lehmann, Gerd Teschke:
Wavelet Based Methods for Improved Wind Profiler Signal Processing, Mai 2000.

00–13. Stephan Dahlke, Peter Maass:
A Note on Interpolating Scaling Functions, August 2000.

00–14. Ronny Ramlau, Rolf Clackdoyle, Frédéric Noo, Girish Bal:
Accurate Attenuation Correction in SPECT Imaging using Optimization of Bilinear Functions and Assuming an Unknown Spatially-Varying Attenuation Distribution, September 2000.

00–15. Peter Kunkel, Ronald Stöver:
Symmetric collocation methods for linear differential-algebraic boundary value problems, September 2000.

00–16. Fabian Wirth:
The generalized spectral radius and extremal norms, Oktober 2000.

00–17. Frank Stenger, Ahmad Reza Naghsh-Nilchi, Jenny Niebsch, Ronny Ramlau:
A unified approach to the approximate solution of PDE, November 2000.

00–18. Peter Benner, Enrique S. Quintana-Ortí, Gregorio Quintana-Ortí:
Parallel algorithms for model reduction of discrete–time systems, Dezember 2000.

00–19. Ronny Ramlau:
A steepest descent algorithm for the global minimization of Tikhonov–Phillips functional, Dezember 2000.

01–01. Efficient methods in hyperthermia treatment planning:
Torsten Köhler, Peter Maass, Peter Wust, Martin Seebass, Januar 2001.

01–02. Parallel Algorithms for LQ Optimal Control of Discrete-Time Periodic Linear Systems:
Peter Benner, Ralph Byers, Rafael Mayo, Enrique S. Quintana-Ortí, Vicente Hernández, Februar 2001.

01–03. Peter Benner, Enrique S. Quintana-Ortí, Gregorio Quintana-Ortí:
Efficient Numerical Algorithms for Balanced Stochastic Truncation, März 2001.

01–04. Peter Benner, Maribel Castillo, Enrique S. Quintana-Ortí:
Partial Stabilization of Large-Scale Discrete-Time Linear Control Systems, März 2001.

01–05. Stephan Dahlke:
Besov Regularity for Edge Singularities in Polyhedral Domains, Mai 2001.

01–06. Fabian Wirth:
A linearization principle for robustness with respect to time-varying perturbations, Mai 2001.

01–07. Stephan Dahlke, Wolfgang Dahmen, Karsten Urban:
Adaptive Wavelet Methods for Saddle Point Problems - Optimal Convergence Rates, Juli 2001.

01–08. Ronny Ramlau:
Morozov's Discrepancy Principle for Tikhonov regularization of nonlinear operators, Juli 2001.

01–09. Michael Wolff:
Einführung des Drucks für die instationären Stokes–Gleichungen mittels der Methode von Kaplan, Juli 2001.

01–10. Stephan Dahlke, Peter Maaß, Gerd Teschke:
Reconstruction of Reflectivity Desities by Wavelet Transforms, August 2001.

01–11. Stephan Dahlke:
Besov Regularity for the Neumann Problem, August 2001.

01–12. Bernard Haasdonk, Mario Ohlberger, Martin Rumpf, Alfred Schmidt, Kunibert G. Siebert:
h–p–Multiresolution Visualization of Adaptive Finite Element Simulations, Oktober 2001.

01–13. Stephan Dahlke, Gabriele Steidl, Gerd Teschke:
Coorbit Spaces and Banach Frames on Homogeneous Spaces with Applications to Analyzing Functions on Spheres, August 2001.

02–01. Michael Wolff, Michael Böhm:
Zur Modellierung der Thermoelasto-Plastizität mit Phasenumwandlungen bei Stählen sowie der Umwandlungsplastizität, Februar 2002.

02–02. Stephan Dahlke, Peter Maaß:
An Outline of Adaptive Wavelet Galerkin Methods for Tikhonov Regularization of Inverse Parabolic Problems, April 2002.

02–03. Alfred Schmidt:
A Multi-Mesh Finite Element Method for Phase Field Simulations, April 2002.

02–04. Sergey N. Dachkovski, Michael Böhm:
A Note on Finite Thermoplasticity with Phase Changes, Juli 2002.

02–05. Michael Wolff, Michael Böhm:
Phasenumwandlungen und Umwandlungsplastizität bei Stählen im Konzept der Thermoelasto-Plastizität, Juli 2002.

02–06. Gerd Teschke:
Construction of Generalized Uncertainty Principles and Wavelets in Anisotropic Sobolev Spaces, August 2002.

02–07. Ronny Ramlau:
TIGRA – an iterative algorithm for regularizing nonlinear ill–posed problems, August 2002.

02–08. Michael Lukaschewitsch, Peter Maaß, Michael Pidcock:
Tikhonov regularization for Electrical Impedance Tomography on unbounded domains, Oktober 2002.