

Chapter 2

Solution of linear systems of equation

Introduction

This chapter is devoted to the numerical solution of linear problems of the following form:

$$\text{Find } \mathbf{x} \in \mathbf{R}^n \text{ such that } \quad \mathbf{Ax} = \mathbf{b}, \quad \mathbf{A} \in \mathbf{R}^{n \times n}, \quad \mathbf{b} \in \mathbf{R}^n. \quad (2.1)$$

Systems of this type appear in a variety of applications. They naturally arise in the context of linear partial differential equations, which we use as main motivating example. Partial differential equations govern a wide range of physical phenomena including heat propagation, gravity, and electromagnetism, to mention just a few. Linear systems in this context often have a particular structure: the matrix \mathbf{A} is generally very sparse, which means that most of the entries are equal to 0, and it is often symmetric and positive definite, provided that these properties are satisfied by the underlying operator.

There are two main approaches for solving linear systems:

- Direct methods enable to calculate the exact solution to systems of linear equations, up to round-off errors. Although this is an attractive property, direct methods are usually too computationally costly for large systems: The cost of inverting a general $n \times n$ matrix, measured in number of floating operations, scales as n^3 !
- Iterative methods, on the other hand, enable to progressively calculate increasingly accurate approximations of the solution. Iterations may be stopped once the *the residual* is sufficiently small. These methods are often preferable when the dimension n of the linear system is very large.

This chapter is organized as follows.

- In [Section 2.1](#), we introduce the concept of *conditioning*. The condition number of a matrix provides information on the sensitivity of the solution to perturbations of the right-hand side \mathbf{b} or matrix \mathbf{A} . It is useful, for example, in order to determine the potential impact of round-off errors.
- In [Section 2.2](#), we present the direct method for solving systems of linear equations. We

study in particular the LU decomposition for an invertible matrix, as well as its variant for symmetric positive definite matrices, which is called the Cholesky decomposition.

- In [Section 2.3](#), we present iterative methods for solving linear systems. We focus in particular on basic iterative methods based on a splitting, and on the conjugate gradient method.

2.1 Conditioning

The condition number for a given problem measures the sensitivity of the solution to the input data. In order to define this concept precisely, we consider a general problem of the form $F(x, d) = 0$, with unknown x and data d . The linear system (2.1) can be recast in this form, with the input data equal to \mathbf{b} or \mathbf{A} or both. We denote the solution corresponding to perturbed input data $d + \Delta d$ by $x + \Delta x$. The absolute and relative condition numbers are defined as follows.

Definition 2.1 (Condition number for the problem $F(x, d) = 0$). The absolute and relative condition numbers with respect to perturbations of d are defined as

$$K_{\text{abs}}(d) = \lim_{\varepsilon \rightarrow 0} \left(\sup_{\|\Delta d\| \leq \varepsilon} \frac{\|\Delta x\|}{\|\Delta d\|} \right), \quad K(d) = \lim_{\varepsilon \rightarrow 0} \left(\sup_{\|\Delta d\| \leq \varepsilon} \frac{\|\Delta x\|/\|x\|}{\|\Delta d\|/\|d\|} \right).$$

The short notation K is reserved for the relative condition number, which is often more useful in applications.

In the rest of this section, we obtain an upper bound on the relative condition number for the linear system (2.1) with respect to perturbations first of \mathbf{b} , and then of \mathbf{A} . We use the notation $\|\cdot\|$ to denote both a vector norm on \mathbf{R}^n and the induced operator norm on matrices.

Proposition 2.1 (Perturbation of the right-hand side). *Let $\mathbf{x} + \Delta \mathbf{x}$ denote the solution to the perturbed equation $\mathbf{A}(\mathbf{x} + \Delta \mathbf{x}) = \mathbf{b} + \Delta \mathbf{b}$. Then it holds that*

$$\frac{\|\Delta \mathbf{x}\|}{\|\mathbf{x}\|} \leq \|\mathbf{A}\| \|\mathbf{A}^{-1}\| \frac{\|\Delta \mathbf{b}\|}{\|\mathbf{b}\|}, \quad (2.2)$$

Proof. It holds by definition of $\Delta \mathbf{x}$ that $\mathbf{A}\Delta \mathbf{x} = \Delta \mathbf{b}$. Therefore, we have

$$\|\Delta \mathbf{x}\| = \|\mathbf{A}^{-1}\Delta \mathbf{b}\| \leq \|\mathbf{A}^{-1}\| \|\Delta \mathbf{b}\| = \frac{\|\mathbf{A}\mathbf{x}\|}{\|\mathbf{b}\|} \|\mathbf{A}^{-1}\| \|\Delta \mathbf{b}\| \leq \frac{\|\mathbf{A}\| \|\mathbf{x}\|}{\|\mathbf{b}\|} \|\mathbf{A}^{-1}\| \|\Delta \mathbf{b}\|. \quad (2.3)$$

Here we employed (A.7), proved in [Appendix A](#), in the first and last inequalities. Rearranging the inequality (2.3), we obtain (2.2). \square

[Proposition 2.1](#) implies that the relative condition number of (2.1) with respect to perturbations of the right-hand side is bounded from above by $\|\mathbf{A}\| \|\mathbf{A}^{-1}\|$. [Exercise 2.1](#) shows that there are values of \mathbf{x} and $\Delta \mathbf{b}$ for which the inequality (2.2) is sharp.

Studying the impact of perturbations of the matrix \mathbf{A} is slightly more difficult, because this time the variation $\Delta \mathbf{x}$ of the solution does not depend linearly on the perturbation.

Proposition 2.2 (Perturbation of the matrix). *Let $\mathbf{x} + \Delta \mathbf{x}$ denote the solution to the perturbed equation $(\mathbf{A} + \Delta \mathbf{A})(\mathbf{x} + \Delta \mathbf{x}) = \mathbf{b}$. If \mathbf{A} is invertible and $\|\Delta \mathbf{A}\| < \|\mathbf{A}^{-1}\|^{-1}$, then*

$$\frac{\|\Delta \mathbf{x}\|}{\|\mathbf{x}\|} \leq \|\mathbf{A}\| \|\mathbf{A}^{-1}\| \frac{\|\Delta \mathbf{A}\|}{\|\mathbf{A}\|} \left(\frac{1}{1 - \|\mathbf{A}^{-1} \Delta \mathbf{A}\|} \right). \quad (2.4)$$

Before proving this result, we show an ancillary lemma.

Lemma 2.3. *Let $\mathbf{B} \in \mathbf{R}^{n \times n}$ be such that $\|\mathbf{B}\| < 1$. Then $\mathbf{I} - \mathbf{B}$ is invertible and*

$$\|(\mathbf{I} - \mathbf{B})^{-1}\| \leq \frac{1}{1 - \|\mathbf{B}\|}, \quad (2.5)$$

where $\mathbf{I} \in \mathbf{R}^{n \times n}$ is the identity matrix.

Proof. It holds for any matrix $\mathbf{B} \in \mathbf{R}^{n \times n}$ that

$$\mathbf{I} - \mathbf{B}^{n+1} = (\mathbf{I} - \mathbf{B})(\mathbf{I} + \mathbf{B} + \cdots + \mathbf{B}^n).$$

Since $\|\mathbf{B}\| < 1$ in a submultiplicative matrix norm, both sides of the equation are convergent in the limit as $n \rightarrow \infty$, with the left-hand side converging to identity matrix \mathbf{I} . Equating the limits, we obtain

$$\mathbf{I} = (\mathbf{I} - \mathbf{B}) \sum_{i=0}^{\infty} \mathbf{B}^i.$$

This implies that $(\mathbf{I} - \mathbf{B})$ is invertible with inverse given by a so-called *Neumann series*

$$(\mathbf{I} - \mathbf{B})^{-1} = \sum_{i=0}^{\infty} \mathbf{B}^i.$$

Applying the triangle inequality repeatedly, and then using the submultiplicative property of the norm, we obtain

$$\forall n \in \mathbf{N}, \quad \left\| \sum_{i=0}^n \mathbf{B}^i \right\| \leq \sum_{i=0}^n \|\mathbf{B}^i\| \leq \sum_{i=0}^n \|\mathbf{B}\|^i = \frac{1}{1 - \|\mathbf{B}\|}.$$

where we used the summation formula for geometric series in the last equality. Letting $n \rightarrow \infty$ in this equation and using the continuity of the norm enables to conclude the proof. \square

Proof of Proposition 2.2. Left-multiplying both side with \mathbf{A}^{-1} , we obtain

$$(\mathbf{I} + \mathbf{A}^{-1} \Delta \mathbf{A})(\mathbf{x} + \Delta \mathbf{x}) = \mathbf{x} \quad \Leftrightarrow \quad (\mathbf{I} + \mathbf{A}^{-1} \Delta \mathbf{A}) \Delta \mathbf{x} = -\mathbf{A}^{-1} \Delta \mathbf{A} \mathbf{x}. \quad (2.6)$$

Since $\|\mathbf{A}^{-1}\Delta\mathbf{A}\| \leq \|\mathbf{A}^{-1}\|\|\Delta\mathbf{A}\| < 1$ by assumption, we deduce from [Lemma 2.3](#) that the matrix on the left-hand side is invertible with a norm bounded as in [\(2.5\)](#). Consequently, using in addition the assumed submultiplicative property of the norm, we obtain that

$$\|\Delta\mathbf{x}\| = \|(I + \mathbf{A}^{-1}\Delta\mathbf{A})^{-1}\mathbf{A}^{-1}\Delta\mathbf{A}\mathbf{x}\| \leq \frac{\|\mathbf{A}^{-1}\Delta\mathbf{A}\|}{1 - \|\mathbf{A}^{-1}\Delta\mathbf{A}\|}\|\mathbf{x}\|.$$

which enables to conclude the proof. \square

Using [Proposition 2.2](#), we deduce that the relative condition number of [\(2.1\)](#) with respect to perturbations of the matrix \mathbf{A} is also bounded from above by $\|\mathbf{A}\|\|\mathbf{A}^{-1}\|$, because the term between brackets on the right-hand side of [\(2.4\)](#) converges to 1 as $\|\Delta\mathbf{A}\| \rightarrow 0$.

[Propositions 2.1](#) and [2.2](#) show that the condition number, with respect to perturbations of either \mathbf{b} or \mathbf{A} , depends only on \mathbf{A} . This motivates the following definition.

Definition 2.2 (Condition number of a matrix). The condition number of a matrix \mathbf{A} associated to a vector norm $\|\bullet\|$ is defined as

$$\kappa(\mathbf{A}) = \|\mathbf{A}\|\|\mathbf{A}^{-1}\|.$$

The condition number for the p -norm, defined in [Definition A.3](#), is denoted by $\kappa_p(\mathbf{A})$.

Note that the condition number $\kappa(\mathbf{A})$ associated with an induced norm is at least one. Indeed, since the identity matrix has induced norm 1, it holds that

$$1 = \|\mathbf{I}\| = \|\mathbf{A}\mathbf{A}^{-1}\| \leq \|\mathbf{A}\|\|\mathbf{A}^{-1}\|.$$

Since the 2-norm of an invertible matrix $\mathbf{A} \in \mathbf{R}^{n \times n}$ coincides with the spectral radius $\rho(\mathbf{A}^T\mathbf{A})$, the condition number κ_2 corresponding to the 2-norm is equal to

$$\kappa_2(\mathbf{A}) = \sqrt{\frac{\lambda_{\max}(\mathbf{A}^T\mathbf{A})}{\lambda_{\min}(\mathbf{A}^T\mathbf{A})}},$$

where λ_{\max} and λ_{\min} are the maximal and minimal (both real and positive) eigenvalues of \mathbf{A} .

Example 2.1 (Perturbation of the matrix). Consider the following linear system with perturbed matrix

$$(\mathbf{A} + \Delta\mathbf{A}) \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ .01 \end{pmatrix}, \quad \mathbf{A} = \begin{pmatrix} 1 & 0 \\ 0 & .01 \end{pmatrix}, \quad \Delta\mathbf{A} = \begin{pmatrix} 0 & 0 \\ 0 & \varepsilon \end{pmatrix},$$

where $0 < \varepsilon \ll .01$. Here the eigenvalues of \mathbf{A} are given by $\lambda_1 = 1$ and $\lambda_2 = 0.01$. The solution when $\varepsilon = 0$ is given by $(0, 1)^T$, and the solution to the perturbed equation is

$$\begin{pmatrix} x_1 + \Delta x_1 \\ x_2 + \Delta x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ \frac{1}{1+100\varepsilon} \end{pmatrix}.$$

Consequently, we deduce that, in the 2-norm,

$$\frac{\|\Delta \mathbf{x}\|}{\|\mathbf{x}\|} = \left| \frac{100\varepsilon}{1 + 100\varepsilon} \right| \approx 100\varepsilon = 100 \frac{\|\Delta \mathbf{A}\|}{\|\mathbf{A}\|}.$$

In this case, the relative impact of perturbations of the matrix is close to $\kappa_2(\mathbf{A}) = 100$.

❁ **Exercise 2.1.** In the simple case where \mathbf{A} is symmetric, find values of \mathbf{x} , \mathbf{b} and $\Delta \mathbf{b}$ for which the inequality (2.2) is in fact an equality?

2.2 Direct solution method

In this section, we present the *direct method* for solving linear systems of the form (2.1) with a general invertible matrix $\mathbf{A} \in \mathbf{R}^{n \times n}$. The direct method can be decomposed into three steps:

- First calculate the so-called LU decomposition of \mathbf{A} , i.e. find an upper triangular matrix \mathbf{U} and a *unit* lower triangular matrix \mathbf{L} such that $\mathbf{A} = \mathbf{L}\mathbf{U}$. A unit lower triangular matrix is a lower triangular matrix with only ones on the diagonal.
- Then solve $\mathbf{L}\mathbf{y} = \mathbf{b}$ using a method called *forward substitution*.
- Finally, solve $\mathbf{U}\mathbf{x} = \mathbf{y}$ using a method called *backward substitution*.

By construction, the solution \mathbf{x} thus obtained is a solution to (2.1). Indeed, we have that

$$\mathbf{A}\mathbf{x} = \mathbf{L}\mathbf{U}\mathbf{x} = \mathbf{L}\mathbf{y} = \mathbf{b}.$$

2.2.1 LU decomposition

In this section, we first discuss the existence and uniqueness of the LU factorization. We then describe a numerical algorithm for calculating the factors \mathbf{L} and \mathbf{U} , based on *Gaussian elimination*.

Existence and uniqueness of the decomposition

We present a necessary and sufficient condition for the existence of a unique LU decomposition of a matrix. To this end, we define the principal submatrix of order i of a matrix $\mathbf{A} \in \mathbf{R}^{n \times n}$ as the matrix $\mathbf{A}_i = \mathbf{A}[1 : i, 1 : i]$, in Julia notation.

Proposition 2.4. The LU factorization of a matrix $\mathbf{A} \in \mathbf{R}^{n \times n}$ exists and is unique if and only if the principal submatrices of \mathbf{A} of all orders are nonsingular.

Proof. We prove only the “if” direction; see [9, Theorem 3.4] for the “only if” implication.

The statement is clear if $n = 1$. Reasoning by induction, we assume that the result is proved up to $n - 1$. Since the matrix \mathbf{A}_{n-1} and all its principal submatrices are nonsingular by assumption, it holds that $\mathbf{A}_{n-1} = \mathbf{L}_{n-1}\mathbf{U}_{n-1}$ for a unit lower triangular matrix \mathbf{L}_{n-1} and an

upper triangular matrix U_{n-1} . These two matrices are nonsingular, for if either of them were singular then the product $A_{n-1} = L_{n-1}U_{n-1}$ would be singular as well. Let us decompose A as follows:

$$A = \begin{pmatrix} A_{n-1} & \mathbf{c} \\ \mathbf{d}^T & a_{nn} \end{pmatrix}.$$

Let $\boldsymbol{\ell}$ and \mathbf{u} denote the solutions to $L_{n-1}\mathbf{u} = \mathbf{c}$ and $U_{n-1}^T\boldsymbol{\ell} = \mathbf{d}$. These solutions exist and are unique, because the matrices L_{n-1} and U_{n-1} are nonsingular. Letting $u_{nn} = a_{nn} - (\boldsymbol{\ell}^T\mathbf{u})^{-1}$, we check that A factorizes as

$$\begin{pmatrix} A_{n-1} & \mathbf{c} \\ \mathbf{d}^T & a_{nn} \end{pmatrix} = \begin{pmatrix} L_{n-1} & \mathbf{0}_{n-1} \\ \boldsymbol{\ell}^T & 1 \end{pmatrix} \begin{pmatrix} U_{n-1} & \mathbf{u} \\ \mathbf{0}_{n-1}^T & u_{nn} \end{pmatrix}.$$

This completes the proof of the existence of the decomposition. The uniqueness of the factors follows from the uniqueness of $\boldsymbol{\ell}$, \mathbf{u} and u_{nn} . \square

[Proposition 2.4](#) raises the following question: are there classes of matrices whose principal matrices are all nonsingular? The answer is positive, and we mention, as an important example, the class of positive definite matrices. Proving this is the aim of [Exercise 2.4](#).

Gaussian elimination algorithm for computing L and U

So far we have presented a condition under which the LU decomposition of a matrix exists and is unique, but not a practical method for calculating the matrices L and U . We describe in this section an algorithm, known as *Gaussian elimination*, for calculating the LU decomposition of a matrix. We begin by introducing the concept of *Gaussian transformation*.

Definition 2.3. A Gaussian transformation is a matrix of the form $M_k = I - \mathbf{c}^{(k)}\mathbf{e}_k^T$, where \mathbf{e}_k is the column vector with entry at index k equal to 1 and all the other entries equal to zero, and $\mathbf{c}^{(k)}$ is a column vector of the following form:

$$\mathbf{c}^{(k)} = \begin{pmatrix} 0 & 0 & \dots & 0 & c_{k+1}^{(k)} & c_{k+2}^{(k)} & \dots & c_n^{(k)} \end{pmatrix}^T.$$

The action of a Gaussian transformation M_k left-multiplying a matrix $A \in \mathbf{R}^{n \times n}$ is to replace the rows from index $k+1$ to index n by a linear combination involving themselves and the k -th row. To see this, let us denote by $(\mathbf{r}^{(i)})_{1 \leq i \leq n}$ the rows of a matrix $T \in \mathbf{R}^{n \times n}$. Then, we have

$$M_k T = (I - \mathbf{c}^{(k)}\mathbf{e}_k^T) T = \begin{pmatrix} 1 & & & & & & & & \\ & 1 & & & & & & & \\ & & \ddots & & & & & & \\ & & & 1 & & & & & \\ & & & -c_{k+1}^{(k)} & 1 & & & & \\ & & & \vdots & & \ddots & & & \\ & & & -c_n^{(k)} & & & 1 & & \end{pmatrix} \begin{pmatrix} \mathbf{r}^{(1)} \\ \mathbf{r}^{(2)} \\ \vdots \\ \mathbf{r}^{(k)} \\ \mathbf{r}^{(k+1)} \\ \vdots \\ \mathbf{r}^{(n)} \end{pmatrix} = \begin{pmatrix} \mathbf{r}^{(1)} \\ \mathbf{r}^{(2)} \\ \vdots \\ \mathbf{r}^{(k)} \\ \mathbf{r}^{(k+1)} - c_{k+1}^{(k)}\mathbf{r}^{(k)} \\ \vdots \\ \mathbf{r}^{(n)} - c_n^{(k)}\mathbf{r}^{(k)} \end{pmatrix}$$

is upper triangular. Let us use the notation $A^{(k)} = M_k \cdots M_1 A$. Of all the Gaussian transformations, only M_1 acts on the second line of the matrix it multiplies. Therefore, the entry $(2, 1)$ of U coincides with the entry $(2, 1)$ of $A^{(1)}$, which implies that $a_{2,1}^{(1)} = 0$. Then notice that $a_{3,1}^{(k)} = a_{3,1}^{(1)}$ for all $k \geq 1$, because the entry $(3, 1)$ of the matrix $M_2 A^{(1)}$ is given by $a_{3,1}^{(1)} - c_3^{(2)} a_{2,1}^{(1)} = a_{3,1}^{(1)}$, and the other transformation matrices M_3, \dots, M_{n-1} leave the third line invariant. Consequently, it holds that $a_{3,1}^{(1)} = u_{3,1} = 0$. Continuing in this manner, we deduce that $A^{(1)}$ is upper triangular in the first column and that, since A is invertible by assumption, the first pivot a_{11} is nonzero. Since this pivot is nonzero, the matrix M_1 is uniquely defined by [Lemma 2.6](#).

The reasoning can then be repeated with other columns, in order to deduce that $A^{(k)}$ is upper triangular up to column k and that all the pivots $a_{kk}^{(k-1)}$ are nonzero. \square

Computer implementation

The Gaussian elimination procedure is summarized as follows.

```

A(0) ← A, L ← I
for i ∈ {1, ..., n - 1} do
    Construct Mi as in Lemma 2.6.
    A(i) ← MiA(i-1), L ← LMi-1
end for
U ← A(n-1).

```

Of course, in practice it is not necessary to explicitly create the Gaussian transformation matrices, or to perform full matrix multiplications. A more realistic, but still very simplified, version of the algorithm in Julia is given below. The code exploits the relation (2.8) between L and the parameters of the Gaussian transformations.

```

1  # A is an invertible matrix of size n x n
2  L = [i == j ? 1.0 : 0.0 for i in 1:n, j in 1:n]
3  U = copy(A)
4  for i in 1:n-1
5      for r in i+1:n
6          U[i, i] == 0 && error("Pivotal entry is zero!")
7          ratio = U[r, i] / U[i, i]
8          L[r, i] = ratio
9          U[r, i:end] -= U[i, i:end] * ratio
10     end
11 end
12 # L is unit lower triangular and U is upper triangular

```

Computational cost

The computational cost of the algorithm, measured as the number of floating point operations (flops) required, is dominated by the Gaussian transformations, in line 9 in the above code. All the other operations amount to a computational cost scaling as $\mathcal{O}(n^2)$, which is negligible

compared to the cost of the LU factorization when n is large. This factorization requires

$$\underbrace{2 \times}_{\text{- and *}} \sum_{i=1}^{n-1} \underbrace{(n-i)}_{\text{for r in i+1:n}} \underbrace{(n-i+1)}_{\substack{\text{indices [i:end]} \\ \text{indices [i:end]}}} \text{ flops} = \frac{2}{3}n^3 + \mathcal{O}(n^2) \text{ flops.}$$

2.2.2 Backward and forward substitution

Once the LU factorization has been completed, the solution to the linear system can be obtained by first using forward, and then backward substitution, which are just bespoke methods for solving linear systems with lower and upper triangular matrices, respectively. Let us consider the case of a lower triangular system:

$$Ly = b$$

Notice that the unknown y_1 may be obtained from the first line of the system. Then, since y_1 is known, the value of y_2 can be obtained from the second line, etc. A simple implementation of this algorithm is as follows:

```
# L is unit lower triangular
y = copy(b)
for i in 2:n
    for j in 1:i-1
        y[i] -= L[i, j] * y[j]
    end
end
```

2.2.3 Gaussian elimination with pivoting

The Gaussian elimination algorithm that we presented in [Section 2.2.1](#) relies on the existence of an LU factorization. In practice, this assumption may not be satisfied, and in this case a modified algorithm, called Gaussian elimination *with pivoting*, is required.

In fact, pivoting is useful even if the usual LU decomposition of A exists, as it enables to reduce the condition number of the matrices L and U . There are two types of pivoting: partial pivoting, where only the rows are rearranged through a permutation at each iteration, and complete pivoting, where both the rows and the columns are rearranged at each iteration.

Showing rigorously why pivoting is useful requires a detailed analysis and is beyond the scope of this course. In this section, we only present the partial pivoting method. Its influence on the condition number of the factors L and U is studied empirically in [Exercise 2.6](#). It is useful at this point to introduce the concept of a row permutation matrix.

Row permutation matrix

Definition 2.4. Let $\sigma : \{1, \dots, n\} \rightarrow \{1, \dots, n\}$ be a permutation, i.e. a bijection on the

set $\{1, \dots, n\}$. The row permutation matrix associated with σ is the matrix with entries

$$p_{ij} = \begin{cases} 1 & \text{if } i = \sigma(j), \\ 0 & \text{otherwise.} \end{cases}$$

When a row permutation P left-multiplies a matrix $B \in \mathbf{R}^{n \times n}$, row i of matrix B is moved to row index $\sigma(i)$ in the resulting matrix, for all $i \in \{1, \dots, n\}$. A permutation matrix has a single entry equal to 1 per row and per column, and its inverse coincides with its transpose: $P^{-1} = P^T$.

Partial pivoting

Gaussian elimination with partial pivoting applies for any invertible matrix A , and it outputs 3 matrices: a row permutation P , a unit triangular matrix L , and an upper triangular matrix U . These are related by the relation

$$PA = LU.$$

This is sometimes called a PLU decomposition of the matrix A . It is not unique in general but, unlike the usual LU decomposition, it always exists provided that A is invertible. We take this for granted in this course.

The idea of partial pivoting is to rearrange the rows at each iteration of the Gaussian elimination procedure in such a manner that the pivotal entry is as large as possible in absolute value. One step of the procedure reads

$$A^{(k+1)} = M_{k+1}P_{k+1}A^{(k)}. \quad (2.9)$$

Here P_{k+1} is a simple row permutation matrix which, when acting on $A^{(k)}$, interchanges row $k+1$ and row ℓ , for some index $\ell \geq k+1$. The row index ℓ is selected in such a way that the absolute value of the pivotal entry, in position $(k+1, k+1)$ of the product $P_{k+1}A^{(k)}$, is maximum. The matrix M_{k+1} is then the unique Gaussian transformation matrix ensuring that $A^{(k+1)}$ is upper triangular up to column $k+1$, obtained as in [Lemma 2.6](#). The resulting matrix $A^{(n-1)}$ after $n-1$ steps of the form (2.9) is upper triangular and satisfies

$$A^{(n-1)} = M_{n-1}P_{n-1} \cdots M_1P_1A \quad \Leftrightarrow \quad A = (M_{n-1}P_{n-1} \cdots M_1P_1)^{-1}A^{(n-1)}.$$

The first factor in the decomposition of A is not necessarily lower triangular. However, using the notation $M = M_{n-1}P_{n-1} \cdots M_1P_1$ and $P = P_{n-1} \cdots P_1$, we have

$$PA = PM^{-1}U = (PM^{-1})U =: LU. \quad (2.10)$$

[Lemma 2.8](#) below shows that, as the notation L suggests, the matrix $L = (PM^{-1})$ on the right-hand side is indeed lower triangular. Before stating and proving the lemma, we note that P is a row permutation matrix, and so the solution to the linear system $A\mathbf{x} = \mathbf{b}$ can be obtained by solving $LU\mathbf{x} = P^T\mathbf{b}$ by forward and backward substitution. Since P is a very sparse matrix, the right-hand side $P^T\mathbf{b}$ can be calculated very efficiently.

Lemma 2.8. *The matrix $L = PM^{-1}$ is unit lower triangular with all entries bounded in absolute value from above by 1. It admits the expression*

$$L = I + (P_{n-1} \cdots P_2 \mathbf{c}^{(1)}) \mathbf{e}_1^T + (P_{n-1} \cdots P_3 \mathbf{c}^{(2)}) \mathbf{e}_2^T + \cdots + (P_{n-1} \mathbf{c}^{(n-2)}) \mathbf{e}_{n-2}^T + \mathbf{c}^{(n-1)} \mathbf{e}_{n-1}^T.$$

Proof. Let $M^{(k)} = M_k P_k \cdots M_1 P_1$ and $P^{(k)} = P_k \cdots P_1$. It is sufficient to show that

$$P^{(k)} (M^{(k)})^{-1} = I + (P_k \cdots P_2 \mathbf{c}^{(1)}) \mathbf{e}_1^T + (P_k \cdots P_3 \mathbf{c}^{(2)}) \mathbf{e}_2^T + \cdots + (P_k \mathbf{c}^{(k-1)}) \mathbf{e}_{k-1}^T + \mathbf{c}^{(k)} \mathbf{e}_k^T \quad (2.11)$$

for all $k \in \{1, \dots, n-1\}$. The statement is clear for $k = 1$, and we assume by induction that it is true up to $k-1$. Then notice that

$$\begin{aligned} P^{(k)} (M^{(k)})^{-1} &= P_k \left(P^{(k-1)} (M^{(k-1)})^{-1} \right) P_k^{-1} M_k^{-1} \\ &= P_k \left(I + (P_{k-1} \cdots P_2 \mathbf{c}^{(1)}) \mathbf{e}_1^T + \cdots + (P_{k-1} \mathbf{c}^{(k-2)}) \mathbf{e}_{k-2}^T + \mathbf{c}^{(k-1)} \mathbf{e}_{k-1}^T \right) P_k^{-1} M_k^{-1} \\ &= \left(I + (P_k P_{k-1} \cdots P_2 \mathbf{c}^{(1)}) \mathbf{e}_1^T + \cdots + (P_k P_{k-1} \mathbf{c}^{(k-2)}) \mathbf{e}_{k-2}^T + (P_k \mathbf{c}^{(k-1)}) \mathbf{e}_{k-1}^T \right) M_k^{-1}. \end{aligned}$$

In the last equality, we used that $\mathbf{e}_i^T P_k^{-1} = (P_k \mathbf{e}_i)^T = \mathbf{e}_i^T$ for all $i \in \{1, \dots, k-1\}$, because the row permutation P_k does not affect rows 1 to $k-1$. Using the expression $M_k^{-1} = I + \mathbf{c}^{(k)} \mathbf{e}_k^T$, expanding the product and noting that $\mathbf{e}_j^T \mathbf{c}^{(k)} = 0$ if $j \leq k$, we obtain (2.11). The statement that the entries are bounded in absolute value from above by 1 follows from the choice of the pivot at each iteration. \square

The expression of L in Lemma 2.8 suggests the iterative procedure given in Algorithm 2 for performing the LU factorization with partial pivoting. A Julia implementation of this algorithm is presented in Listing 2.1.

Algorithm 2 LU decomposition with partial pivoting

```

Assign  $A^{(0)} \leftarrow A$  and  $P \leftarrow I$ 
for  $i \in \{1, \dots, n-1\}$  do
    Find the row index  $k \geq i$  such that  $A_{k,i}^{(i-1)}$  is maximum in absolute value.
    Interchange the rows  $i$  and  $k$  of matrices  $A^{(i-1)}$  and  $P$ , and of vectors  $\mathbf{c}^{(1)}, \dots, \mathbf{c}^{(i-1)}$ .
    Construct  $M_i$  with corresponding column vector  $\mathbf{c}^{(i)}$  as in Lemma 2.6.
    Assign  $A^{(i)} \leftarrow M_i A^{(i-1)}$ 
end for
Assign  $U \leftarrow A^{(n-1)}$ .
Assign  $L \leftarrow I + \begin{pmatrix} \mathbf{c}^{(1)} & \dots & \mathbf{c}^{(n-1)} & \mathbf{0}_n \end{pmatrix}$ .
    
```

```

# Auxiliary function
function swap_rows!(i, j, matrices...)
    for M in matrices
        M_row_i = M[i, :]
        M[i, :] = M[j, :]
    end
end
    
```

```

        M[j, :] = M_row_i
    end
end

n = size(A)[1]
L, U = zeros(n, 0), copy(A)
P = [i == j ? 1.0 : 0.0 for i in 1:n, j in 1:n]
for i in 1:n-1
    # Pivoting
    index_row_pivot = i - 1 + argmax(abs.(U[i:end, i]))
    swap_rows!(i, index_row_pivot, U, L, P)

    # Usual Gaussian transformation
    c = [zeros(i-1); 1.0; zeros(n-i)]
    for r in i+1:n
        ratio = U[r, i] / U[i, i]
        c[r] = ratio
        U[r, i:end] -= U[i, i:end] * ratio
    end
    L = [L c]
end
L = [L [zeros(n-1); 1.0]]
# It holds that P*A = L*U

```

Listing 2.1: LU factorization with partial pivoting.

Remark 2.1. It is possible to show that, if the matrix A is column diagonally dominant in the sense that

$$\forall j \in \{1, \dots, n\}, \quad |a_{jj}| \geq \sum_{i=1, i \neq j}^n |a_{ij}|,$$

then pivoting does not have an effect: at each iteration, the best pivot is already on the diagonal.

2.2.4 Direct method for symmetric positive definite matrices

The LU factorization with partial pivoting applies to any matrix $A \in \mathbf{R}^{n \times n}$ that is invertible. If A is symmetric positive definite, however, it is possible to compute a factorization into lower and upper triangular matrices at half the computational cost, using the so-called *Cholesky decomposition*.

Lemma 2.9 (Cholesky decomposition). *If A is symmetric positive definite, then there exists a lower-triangular matrix $C \in \mathbf{R}^{n \times n}$ such that*

$$A = CC^T. \tag{2.12}$$

Equation (2.12) is called the Cholesky factorization of A . The matrix C is unique if we require that all its diagonal entries are positive.

Proof. Since A is positive definite, its LU decomposition exists and is unique by [Propositions 2.4](#) and [2.7](#). Let D denote the diagonal matrix with the same diagonal as that of U . Then

$$A = LD(D^{-1}U).$$

Note that the matrix $D^{-1}U$ is unit upper triangular. Since A is symmetric, we have

$$A = A^T = (D^{-1}U)^T(LD)^T.$$

The first and second factors on the right-hand side are respectively unit lower triangular and upper triangular, and so we deduce, by uniqueness of the LU decomposition, that $L = (D^{-1}U)^T$ and $U = (LD)^T$. But then

$$A = LU = LDL^T = (L\sqrt{D})(\sqrt{D}L)^T.$$

Here \sqrt{D} denotes the diagonal matrix whose diagonal entries are obtained by taking the square root of those of D , which are necessarily positive because A is positive definite. This implies the existence of a Cholesky factorization with $C = L\sqrt{D}$. \square

Calculation of the Choleski factor

The matrix C can be calculated from [\(2.12\)](#). For example, developing the matrix product gives that $a_{1,1} = c_{1,1}^2$ and so $c_{1,1} = \sqrt{a_{1,1}}$. It is then possible to calculate $c_{2,1}$ from the equation $a_{2,1} = c_{2,1}c_{1,1}$, and so on. Implementing the Cholesky factorization is the goal of [Exercise 2.7](#).

2.2.5 Direct methods for banded matrices

In applications related to partial differential equations, the matrix $A \in \mathbf{R}^{n \times n}$ very often has a bandwidth which is small in comparison with n .

Definition 2.5. The bandwidth of a matrix $A \in \mathbf{R}^{n \times n}$ is the smallest number $k \in \mathbf{N}$ such that $a_{ij} = 0$ for all $(i, j) \in \{1, \dots, n\}^2$ with $|i - j| > k$.

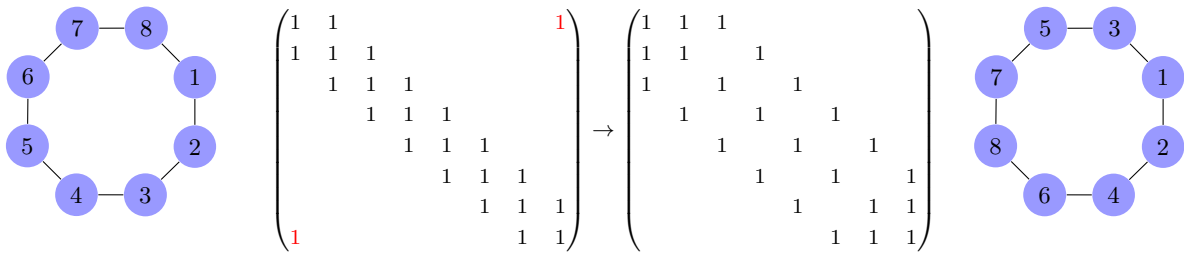
It is not difficult to show that, if A is a matrix with bandwidth k , then so are L and U in the absence of pivoting. This can be proved by equating the entries of the product LU with those of the matrix A . We emphasize, however, that the sparsity structure *within* the band of A may be destroyed in L and U ; this phenomenon is called *fill-in*.

Reducing the bandwidth: the Cuthill–McKee algorithm

The computational cost of calculating the LU or Cholesky decomposition of a matrix with bandwidth k scales as $\mathcal{O}(nk^2)$, which is much better than the general scaling $\mathcal{O}(n^3)$ if $k \ll n$. In applications, the bandwidth k is often related to the matrix size n . For example, if A arises from the discretization of the Laplacian operator, then $k = \mathcal{O}(\sqrt{n})$ provided that a good ordering of the vertices is employed. In this case, the computational cost scales as $\mathcal{O}(n^2)$.

Since a narrow band is associated with a lower computational cost of the LU decomposition, it is natural to wonder whether the bandwidth of a matrix A can be reduced. A possible strategy to this end is to use permutations. More precisely, is it possible to identify a row permutation matrix P such that PAP^T has minimal bandwidth? Given such a matrix, the solution to the linear system (2.1) can be obtained by first solving $(PAP^T)\mathbf{y} = P\mathbf{b}$, and then letting $\mathbf{x} = P^T\mathbf{y}$.

The Cuthill–McKee algorithm is a heuristic method for finding a good, but sometimes not optimal, permutation matrix P in the particular case where A is a *symmetric* matrix. It is based on the fact that, to a symmetric matrix A , we can associate a unique undirected graph whose adjacency matrix A_* has the same sparsity structure as that of A , i.e. zeros in the same places. For any row permutation matrix P_σ with corresponding permutation $\sigma: \{1, \dots, n\} \rightarrow \{1, \dots, n\}$ (see Definition 2.4), the matrices $P_\sigma A P_\sigma^T$ and $P_\sigma A_* P_\sigma^T$ also have the same sparsity structure. Therefore, minimizing the bandwidth of $P_\sigma A P_\sigma^T$ is equivalent to minimizing the bandwidth of $P_\sigma A_* P_\sigma^T$. The key insight for understanding the Cuthill–McKee method is that $P_\sigma A_* P_\sigma^T$ is the adjacency matrix of the graph obtained by renumbering the nodes according to the permutation σ , i.e. by changing the number of the nodes from i to $\sigma(i)$. Consider, for example, the following graph and renumbering:



Here we also wrote the adjacency matrices associated to the graphs. We assume that the nodes are all self-connected, although this is not depicted, and so the diagonal entries of the adjacency matrices are equal to 1. This renumbering corresponds to the permutation

$$\begin{pmatrix} i : & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ \sigma(i) : & 1 & 2 & 4 & 6 & 8 & 7 & 5 & 3 \end{pmatrix},$$

and we may verify that the adjacency matrix of the renumbered graph can be obtained from the associated row permutation matrix:

$$PA_*P^T = \begin{pmatrix} 1 & & & & & & & & \\ & 1 & & & & & & & \\ & & 1 & & & & & & \\ & & & 1 & & & & & \\ & & & & 1 & & & & \\ & & & & & 1 & & & \\ & & & & & & 1 & & \\ & & & & & & & 1 & \\ & & & & & & & & 1 \end{pmatrix} \begin{pmatrix} 1 & & & & & & & & \\ 1 & 1 & & & & & & & \\ & 1 & 1 & & & & & & \\ & & 1 & 1 & & & & & \\ & & & 1 & 1 & & & & \\ & & & & 1 & 1 & & & \\ & & & & & 1 & 1 & & \\ & & & & & & 1 & 1 & \\ & & & & & & & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & & & & & & & & \\ & 1 & & & & & & & \\ & & 1 & & & & & & \\ & & & 1 & & & & & \\ & & & & 1 & & & & \\ & & & & & 1 & & & \\ & & & & & & 1 & & \\ & & & & & & & 1 & \\ & & & & & & & & 1 \end{pmatrix}$$

In this example, renumbering the nodes of the graph enables a significant reduction of the bandwidth, from 7 to 2. The Cuthill–McKee algorithm, which was employed to calculate the

permutation, is an iterative method that produces an ordered n -tuple R containing the nodes in the new order; in other words, it returns $(\sigma^{-1}(1), \dots, \sigma^{-1}(n))$. The first step of the algorithm is to find the node i with the lowest *degree*, i.e. with the smallest number of connections to other nodes, and to initialize $R = (i)$. Then the following steps are repeated until R contains all the nodes of the graph:

- Define A_i as the set containing all the nodes which are adjacent to a node in R but not themselves in R ;
- Sort the nodes in A_i according to the following rules: a node $i \in A_i$ comes before $j \in A_i$ if i is connected to a node in R that comes before all the nodes in R to which j is connected. As a tiebreak, precedence is given to the node with highest degree.
- Append the nodes in A_i to R , in the order determined in the previous item.

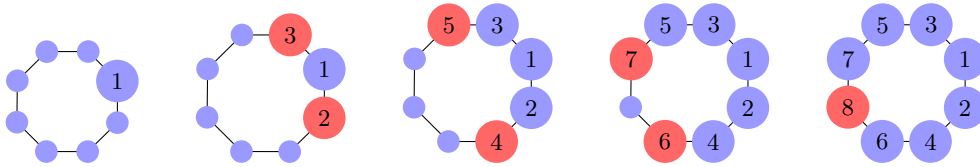


Figure 2.1: Illustration of the Cuthill–McKee algorithm. The new numbering of the nodes is illustrated. The first node was chosen randomly since all the nodes have the same degree. In this example, the ordered tuple R evolves as follows: $(1) \rightarrow (1, 2, 8) \rightarrow (1, 2, 8, 3, 7) \rightarrow (1, 2, 8, 3, 7, 4, 6) \rightarrow (1, 2, 8, 3, 7, 4, 6, 5)$.

The steps of the algorithm for the example above are depicted in Figure 2.1. Another example, taken from the original paper by Cuthill and McKeen [1], is presented in Figure 2.2.

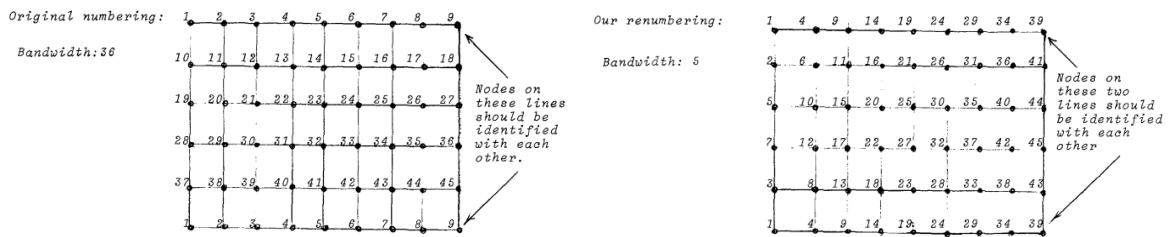


Figure 2.2: Example from the original Cuthill–McKee paper [1].

2.2.6 Exercises

- ⚙ **Exercise 2.2** (Inverse of Gaussian transformation). *Prove the formula (2.7).*
- ⚙ **Exercise 2.3.** *Prove that the product of two lower triangular matrices is lower triangular.*
- ⚙ **Exercise 2.4.** *Assume that $A \in \mathbf{R}^{n \times n}$ is positive definite, i.e. that*

$$\forall \mathbf{x} \in \mathbf{R}^n \setminus \{\mathbf{0}_n\}, \quad \mathbf{x}^T \mathbf{A} \mathbf{x} > 0.$$

Show that all the principal submatrices of A are nonsingular.

□ **Exercise 2.5.** Implement the backward substitution algorithm for solving $Ux = y$. What is the computational cost of the algorithm?

□ **Exercise 2.6.** Compare the condition number of the matrices L and U with and without partial pivoting. For testing, use a matrix with pseudo-random entries generated as follows

```
import Random
# Set the seed so that the code is deterministic
Random.seed!(0)
n = 1000 # You can change this parameter
A = randn(n, n)
```

Solution. See the Jupyter notebook for this chapter.

□ **Exercise 2.7.** Write a code for calculating the Cholesky factorization of a symmetric positive definite matrix A by comparing the entries of the product CC^T with those of the matrix A . What is the associated computational cost, and how does it compare with that of the LU factorization?

Extra credit: ... if your code is able to exploit the potential banded structure of the matrix passed as argument for better efficiency. Specifically, your code will be tested with a matrix is of the type `BandedMatrix` defined in the `BandedMatrices.jl` package, which you will need to install. The following code can be useful for testing purposes.

```
import BandedMatrices
import LinearAlgebra

function cholesky(A)
    m, n = size(A)
    m != n && error("Matrix must be square")
    # Convert to banded matrix
    B = BandedMatrices.BandedMatrix(A)
    B.u != B.l && error("Matrix must be symmetric")
    # --> Your code comes here <--
end

n, u, l = 20000, 2, 2
A = BandedMatrices.brand(n, u, l)
A = A*A'
# so that A is symmetric and positive definite (with probability 1).
C = @time cholesky(A)
LinearAlgebra.norm(C*C' - A, Inf)
```

For information, my code takes about 1 second to run with the parameters given here.

⚙️ **Exercise 2.8** (Matrix square root). Let A be a symmetric positive definite matrix. Show that A has a square root, i.e. that there exists a symmetric matrix B such that $BB = A$.

2.3 Iterative methods for linear systems

Iterative methods enjoy more flexibility than direct methods, because they can be stopped at any point if the residual is deemed sufficiently small. This generally enables to obtain a good solution at a computational cost that is significantly lower than that of direct methods. In this section, we present and study two classes of iterative methods: basic iterative methods based on a splitting of the matrix A , and the so-called Krylov subspace methods.

2.3.1 Basic iterative methods

The basic iterative methods are particular cases of a general *splitting method*. Given a splitting of the matrix of the linear system as $A = M - N$, for a nonsingular matrix $M \in \mathbf{R}^{n \times n}$ and a matrix $N \in \mathbf{R}^{n \times n}$, together with an initial guess $\mathbf{x}^{(0)}$ of the solution, one step of this general method reads

$$M\mathbf{x}^{(k+1)} = N\mathbf{x}^{(k)} + \mathbf{b}. \quad (2.13)$$

For any choice of splitting, the exact solution \mathbf{x}_* to the linear system is a fixed point of this iteration, in the sense that if $\mathbf{x}^{(0)} = \mathbf{x}_*$, then $\mathbf{x}^{(k)} = \mathbf{x}_*$ for all $k \geq 0$. Equation (2.13) is a linear system with matrix M , unknown $\mathbf{x}^{(k+1)}$, and right-hand side $N\mathbf{x}^{(k)} + \mathbf{b}$. There is a compromise between the cost of a single step and the speed of convergence of the method. In the extreme case where $M = A$ and $N = 0$, the method converges to the exact solution in one step, but performing this step amounts to solving the initial problem. In practice, in order for the method to be useful, the linear system (2.13) should be relatively simple to solve. Concretely, this means that the matrix M should be diagonal, triangular, block diagonal, or block triangular. The error $\mathbf{e}^{(k)}$ and residual $\mathbf{r}^{(k)}$ at iteration k are defined as follows:

$$\mathbf{e}^{(k)} = \mathbf{x}^{(k)} - \mathbf{x}_*, \quad \mathbf{r}^{(k)} = A\mathbf{x}^{(k)} - \mathbf{b}.$$

Convergence of the splitting method

Before presenting concrete examples of splitting methods, we obtain a necessary and sufficient condition for the convergence of (2.13) for any initial guess $\mathbf{x}^{(0)}$.

Proposition 2.10 (Convergence). *The splitting method (2.13) converges for any initial guess $\mathbf{x}^{(0)}$ if and only if $\rho(M^{-1}N) < 1$. In addition, for any $\varepsilon > 0$ there exists $K > 0$ such that*

$$\forall k \geq K, \quad \|\mathbf{e}^{(k)}\| \leq (\rho(A) + \varepsilon)^k \|\mathbf{e}^{(0)}\|.$$

Proof. Let \mathbf{x}_* denote the solution to the linear system. Since $M\mathbf{x}_* - N\mathbf{x}_* = \mathbf{b}$, we have

$$M(\mathbf{x}^{(k+1)} - \mathbf{x}_*) = N(\mathbf{x}^{(k)} - \mathbf{x}_*).$$

Using the assumption that M is nonsingular, we obtain that the error satisfies the equation

$$\mathbf{e}^{(k+1)} = (M^{-1}N)\mathbf{e}^{(k)}.$$

Applying this equality repeatedly, we deduce

$$\mathbf{e}^{(k+1)} = (\mathbf{M}^{-1}\mathbf{N})^2\mathbf{e}^{(k-1)} = \dots = (\mathbf{M}^{-1}\mathbf{N})^{k+1}\mathbf{e}^{(0)}.$$

Therefore, it holds that

$$\forall k \geq 0, \quad \|\mathbf{e}^{(k)}\| \leq \|(\mathbf{M}^{-1}\mathbf{N})^k\| \|\mathbf{e}^{(0)}\|.$$

In order to conclude the proof, we will use Gelfand's formula, proved in [Proposition A.10](#) of [Appendix A](#). This states that

$$\lim_{k \rightarrow \infty} \|(\mathbf{M}^{-1}\mathbf{N})^k\|^{\frac{1}{k}} = \rho(\mathbf{M}^{-1}\mathbf{N}).$$

In particular, for all $\varepsilon > 0$ there is $K \in \mathbf{N}$ such that

$$\forall k \geq K, \quad \|(\mathbf{M}^{-1}\mathbf{N})^k\|^{\frac{1}{k}} \leq \rho(\mathbf{M}^{-1}\mathbf{N}) + \varepsilon.$$

Rearranging this inequality gives that $\|(\mathbf{M}^{-1}\mathbf{N})^k\|$ the statement. \square

At this point, it is natural to wonder whether there exist sufficient conditions on the matrix \mathbf{A} such that the inequality $\rho(\mathbf{M}^{-1}\mathbf{N}) < 1$ is satisfied, which is best achieved on a case by case basis. In the next sections, we present four instances of splitting methods. For each of them, we obtain a sufficient condition for convergence. We are particularly interested in the case where the matrix \mathbf{A} is symmetric (or Hermitian) and positive definite, which often arises in applications, and in the case where \mathbf{A} is strictly row or column diagonally dominant. We recall that a matrix \mathbf{A} is said to be row or column diagonally dominant if, respectively,

$$|a_{ii}| \geq \sum_{j \neq i} |a_{ij}| \quad \forall i \quad \text{or} \quad |a_{jj}| \geq \sum_{i \neq j} |a_{ij}| \quad \forall j.$$

Richardson's method

Arguably the simplest splitting of the matrix \mathbf{A} is given by $\mathbf{A} = \frac{1}{\omega}\mathbf{I} - (\frac{1}{\omega}\mathbf{I} - \mathbf{A})$, which leads to *Richardson's method*:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \omega(\mathbf{b} - \mathbf{A}\mathbf{x}^{(k)}). \quad (2.14)$$

In this case the spectral radius which enters in the asymptotic rate of convergence is given by

$$\rho(\mathbf{M}^{-1}\mathbf{N}) = \rho\left(\omega\left(\frac{1}{\omega}\mathbf{I} - \mathbf{A}\right)\right) = \rho(\mathbf{I} - \omega\mathbf{A})$$

The eigenvalues of the matrix $\mathbf{I} - \omega\mathbf{A}$ are given by $1 - \omega\lambda_i$, where $(\lambda_i)_{1 \leq i \leq L}$ are the eigenvalues of \mathbf{A} . Therefore, the spectral radius is given by

$$\rho(\mathbf{M}^{-1}\mathbf{N}) = \max_{1 \leq i \leq L} |1 - \omega\lambda_i|.$$

Case of symmetric positive definite A. If the matrix \mathbf{A} is symmetric and positive definite, it is possible to explicitly calculate the optimal value of ω for convergence. In order to make convergence as fast as possible, we want the spectral radius of $\mathbf{M}^{-1}\mathbf{N}$ to be as small as possible, in view of [Proposition 2.10](#). Denoting by λ_{\min} and λ_{\max} the minimum and maximum eigenvalues of \mathbf{A} , it is not difficult to show that

$$\rho(\mathbf{M}^{-1}\mathbf{N}) = \max_{1 \leq i \leq L} |1 - \omega\lambda_i| = \max\{|1 - \omega\lambda_{\min}|, |1 - \omega\lambda_{\max}|\}.$$

The maximum is minimized when its two arguments are equal, i.e. when $1 - \omega\lambda_{\min} = \omega\lambda_{\max} - 1$. From this we deduce the optimal value of ω and the associated spectral radius:

$$\omega_{\text{opt}} = \frac{2}{\lambda_{\max} + \lambda_{\min}}, \quad \rho_{\text{opt}} = 1 - \frac{2\lambda_{\min}}{\lambda_{\max} + \lambda_{\min}} = \frac{\lambda_{\max} - \lambda_{\min}}{\lambda_{\max} + \lambda_{\min}} = \frac{\kappa_2(\mathbf{A}) - 1}{\kappa_2(\mathbf{A}) + 1}.$$

We observe that the smaller the condition number of the matrix \mathbf{A} , the better the asymptotic rate of convergence.

Remark 2.2 (Link to optimization). In the case where \mathbf{A} is symmetric and positive definite, the Richardson update (2.14) may be viewed as a step of the steepest descent algorithm for the function $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} - \mathbf{b}^T\mathbf{x}$:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \omega\nabla f(\mathbf{x}^{(k)}). \quad (2.15)$$

The gradient of this function is $\nabla f(\mathbf{x}) = \mathbf{A}\mathbf{x} - \mathbf{b}$, and its Hessian matrix is \mathbf{A} . Since the Hessian matrix is positive definite, the function is convex and attains its global minimum when ∇f is zero, i.e. when $\mathbf{A}\mathbf{x} = \mathbf{b}$.

Jacobi's method

In Jacobi's method, the matrix \mathbf{M} in the splitting is the diagonal matrix \mathbf{D} with the same entries as those of \mathbf{A} on the diagonal. We denote by \mathbf{L} and \mathbf{U} the lower and upper triangular parts of \mathbf{A} , without the diagonal. One step of the method reads

$$\mathbf{D}\mathbf{x}^{(k+1)} = (\mathbf{D} - \mathbf{A})\mathbf{x}^{(k)} + \mathbf{b} = -(\mathbf{L} + \mathbf{U})\mathbf{x}^{(k)} + \mathbf{b} \quad (2.16)$$

Since the matrix \mathbf{D} on the left-hand side is diagonal, this linear system with unknown $\mathbf{x}^{(k+1)}$ is very simple to solve. The equation (2.16) can be rewritten equivalently as

$$\begin{cases} a_{11}x_1^{(k+1)} + a_{12}x_2^{(k)} + \cdots + a_{1n}x_n^{(k)} = b_1 \\ a_{21}x_1^{(k)} + a_{22}x_2^{(k+1)} + \cdots + a_{2n}x_n^{(k)} = b_2 \\ \vdots \\ a_{n1}x_1^{(k)} + a_{n2}x_2^{(k)} + \cdots + a_{nn}x_n^{(k+1)} = b_n. \end{cases}$$

The updates for each of the entries of $\mathbf{x}^{(k+1)}$ are independent, and so the Jacobi method lends itself well to parallel implementation. The computational cost of one iteration, measured in

number of floating point operations required, scales as $\mathcal{O}(n^2)$ if A is a full matrix, or $\mathcal{O}(nk)$ if A is a sparse matrix with k nonzero elements per row on average. It is simple to prove the convergence of Jacobi's method is the case where A is diagonally dominant.

Proposition 2.11. *Assume that A is strictly (row or column) diagonally dominant. Then it holds that $\rho(M^{-1}N) < 1$ for the Jacobi splitting.*

Proof. Assume that λ is an eigenvalue of $M^{-1}N$ and \mathbf{v} is the associated unit eigenvector. Then

$$M^{-1}N\mathbf{v} = \lambda\mathbf{v} \quad \Leftrightarrow \quad N\mathbf{v} = \lambda M\mathbf{v} \quad \Leftrightarrow \quad (N - \lambda M)\mathbf{v} = 0.$$

In the case of Jacobi's splitting, this is equivalent to

$$-(L + \lambda D + U)\mathbf{v} = 0.$$

If $|\lambda| > 1$, then the matrix on the left-hand side of this equation is diagonally dominant and thus invertible (see [Exercise 2.9](#)). Therefore $\mathbf{v} = 0$, but this is a contradiction because \mathbf{v} is vector of unit norm. Consequently, all the eigenvalues are bounded from above strictly by 1 in modulus. \square

Gauss–Seidel's method

In Gauss Seidel's method, the matrix M in the splitting is the lower triangular part of A , including the diagonal. One step of the method then reads

$$(L + D)\mathbf{x}^{(k+1)} = -U\mathbf{x}^{(k)} + \mathbf{b} \tag{2.17}$$

The system can be solved by forward substitution. The equation (2.17) can be rewritten equivalently as

$$\begin{cases} a_{11}x_1^{(k+1)} + a_{12}x_2^{(k)} + a_{13}x_3^{(k)} + \cdots + a_{1n}x_n^{(k)} = b_1 \\ a_{21}x_1^{(k+1)} + a_{22}x_2^{(k+1)} + a_{23}x_3^{(k)} + \cdots + a_{2n}x_n^{(k)} = b_2 \\ a_{32}x_1^{(k+1)} + a_{32}x_2^{(k+1)} + a_{33}x_3^{(k+1)} + \cdots + a_{3n}x_n^{(k)} = b_3 \\ \vdots \\ a_{n1}x_1^{(k+1)} + a_{n2}x_2^{(k+1)} + a_{n3}x_3^{(k+1)} + \cdots + a_{nn}x_n^{(k+1)} = b_n. \end{cases}$$

Given $\mathbf{x}^{(k)}$, the first entry of $\mathbf{x}^{(k+1)}$ can be obtained from the first equation. Then the value of the second entry can be obtained from the second equation, etc. Unlike Jacobi's method, the Gauss–Seidel method is sequential and the entries of $\mathbf{x}^{(k+1)}$ cannot be updated in parallel.

It is possible to prove the convergence of the Gauss–Seidel method in particular cases. For example, the method converges if A is strictly diagonally dominant. Proving this, using an approach similar to that in the proof of [Proposition 2.11](#), is the goal of [Exercise 2.19](#). It is also possible to prove convergence when A is Hermitian and positive definite. We show this in the next section for the relaxation method, which generalizes the Gauss–Seidel method.

Relaxation method

The relaxation method generalizes the Gauss–Seidel method. It corresponds to the splitting

$$\mathbf{A} = \left(\frac{\mathbf{D}}{\omega} + \mathbf{L} \right) - \left(\frac{1-\omega}{\omega} \mathbf{D} - \mathbf{U} \right). \quad (2.18)$$

When $\omega = 1$, this is simply the Gauss–Seidel splitting. The idea is that, by letting ω be a parameter that can differ from 1, faster convergence can be achieved. This intuition will be verified later. The equation (2.13) for this splitting can be rewritten equivalently as

$$\begin{cases} a_{11}(x_1^{(k+1)} - x_1^{(k)}) = -\omega (a_{11}x_1^{(k)} + a_{12}x_2^{(k)} + \cdots + a_{1n}x_n^{(k)} - b_1) \\ a_{22}(x_2^{(k+1)} - x_2^{(k)}) = -\omega (a_{21}x_1^{(k+1)} + a_{22}x_2^{(k)} + \cdots + a_{2n}x_n^{(k)} - b_2) \\ \vdots \\ a_{nn}(x_n^{(k+1)} - x_n^{(k)}) = -\omega (a_{n1}x_1^{(k+1)} + a_{n2}x_2^{(k+1)} + \cdots + a_{nn}x_n^{(k)} - b_n). \end{cases}$$

The coefficient on the right-hand side is larger than in the Gauss–Seidel method if $\omega > 1$, and smaller if $\omega < 1$. These regimes are called *over-relaxation* and *under-relaxation*, respectively.

To conclude this section, we establish a sufficient condition for the convergence of the relaxation method, and also of the Gauss–Seidel method as a particular case when $\omega = 1$, when the matrix \mathbf{A} is Hermitian and positive definite. To this end, we begin by showing the following preparatory result, which concerns a general splitting $\mathbf{A} = \mathbf{M} - \mathbf{N}$.

Proposition 2.12. *Let \mathbf{A} be Hermitian and positive definite. If the Hermitian matrix $\mathbf{M}^* + \mathbf{N}$ is positive definite, then $\rho(\mathbf{M}^{-1}\mathbf{N}) < 1$.*

Proof. First, notice that $\mathbf{M} + \mathbf{N}^*$ is indeed Hermitian because

$$(\mathbf{M} + \mathbf{N}^*)^* = \mathbf{M}^* + \mathbf{N} = \mathbf{A}^* + \mathbf{N}^* + \mathbf{N} = \mathbf{A} + \mathbf{N}^* + \mathbf{N}.$$

We will show that $\|\mathbf{M}^{-1}\mathbf{N}\|_{\mathbf{A}} < 1$, where $\|\bullet\|_{\mathbf{A}}$ is the matrix norm induced by the following norm on vectors:

$$\|\mathbf{x}\|_{\mathbf{A}} := \sqrt{\mathbf{x}^* \mathbf{A} \mathbf{x}}.$$

Showing that this indeed defines a vector norm is the goal of [Exercise 2.11](#). Since $\mathbf{N} = \mathbf{M} - \mathbf{A}$, it holds that $\|\mathbf{M}^{-1}\mathbf{N}\|_{\mathbf{A}} = \|\mathbf{I} - \mathbf{M}^{-1}\mathbf{A}\|_{\mathbf{A}}$, and so

$$\|\mathbf{M}^{-1}\mathbf{N}\|_{\mathbf{A}} = \sup\{\|\mathbf{x} - \mathbf{M}^{-1}\mathbf{A}\mathbf{x}\|_{\mathbf{A}} : \|\mathbf{x}\|_{\mathbf{A}} \leq 1\}.$$

Letting $\mathbf{y} = \mathbf{M}^{-1}\mathbf{A}\mathbf{x}$, we calculate

$$\begin{aligned} \forall \mathbf{x} \in \mathbf{R}^n \text{ with } \|\mathbf{x}\|_A \leq 1, \quad \|\mathbf{x} - \mathbf{M}^{-1}\mathbf{A}\mathbf{x}\|_A^2 &= \mathbf{x}^* \mathbf{A} \mathbf{x} - \mathbf{y}^* \mathbf{A} \mathbf{x} - \mathbf{x}^* \mathbf{A} \mathbf{y} + \mathbf{y}^* \mathbf{A} \mathbf{y} \\ &= \mathbf{x}^* \mathbf{A} \mathbf{x} - \mathbf{y}^* \mathbf{M} \mathbf{M}^{-1} \mathbf{A} \mathbf{x} - (\mathbf{M}^{-1} \mathbf{A} \mathbf{x})^* \mathbf{M}^* \mathbf{y} + \mathbf{y}^* \mathbf{A} \mathbf{y} \\ &= \mathbf{x}^* \mathbf{A} \mathbf{x} - \mathbf{y}^* \mathbf{M} \mathbf{y} - \mathbf{y}^* \mathbf{M}^* \mathbf{y} + \mathbf{y}^* (\mathbf{M} - \mathbf{N}) \mathbf{y} \\ &= \mathbf{x}^* \mathbf{A} \mathbf{x} - \mathbf{y}^* (\mathbf{M}^* + \mathbf{N}) \mathbf{y} \leq 1 - \mathbf{y}^* (\mathbf{M}^* + \mathbf{N}) \mathbf{y} < 1, \end{aligned}$$

where we used in the last inequality the assumption that $\mathbf{M}^* + \mathbf{N}$ is positive definite. This shows that $\|\mathbf{M}^{-1}\mathbf{N}\|_A < 1$, and so $\rho(\mathbf{M}^{-1}\mathbf{N}) < 1$. \square

As a corollary, we obtain a sufficient condition for the convergence of the relaxation method.

Corollary 2.13. *Assume that \mathbf{A} is Hermitian and positive definite. Then the relaxation method converges if $\omega \in (0, 2)$.*

Proof. For the relaxation method, we have

$$\mathbf{M} + \mathbf{N}^* = \left(\frac{\mathbf{D}}{\omega} + \mathbf{L} \right) + \left(\frac{1-\omega}{\omega} \mathbf{D} - \mathbf{U} \right)^*.$$

Since \mathbf{A} is Hermitian, it holds that $\mathbf{D}^* = \mathbf{D}$ and $\mathbf{U}^* = \mathbf{L}$. Therefore,

$$\mathbf{M} + \mathbf{N}^* = \frac{2-\omega}{\omega} \mathbf{D}.$$

The diagonal elements of \mathbf{D} are all positive, because \mathbf{A} is positive definite. (Indeed, if there was an index i such that $d_{ii} \leq 0$, then it would hold that $\mathbf{e}_i^T \mathbf{A} \mathbf{e}_i = d_{ii} \leq 0$, contradicting the assumption that \mathbf{A} is positive definite.) We deduce that $\mathbf{M} + \mathbf{N}^*$ is positive definite if and only if $\omega \in (0, 2)$. We can then conclude the proof by using [Proposition 2.12](#). \square

Note that [Corollary 2.13](#) implies as a particular case the convergence of the Gauss–Seidel method when \mathbf{A} is Hermitian and positive definite. The condition $\omega \in (0, 2)$ is fact necessary for the convergence of the relaxation method, not only in the case of a Hermitian positive definite matrix \mathbf{A} but in general.

Proposition 2.14 (Necessary condition for the convergence of the relaxation method). *Let $\mathbf{A} \in \mathbf{C}^{n \times n}$ be an invertible matrix, and let $\mathbf{A} = \mathbf{M}_\omega - \mathbf{N}_\omega$ denote the splitting of the relaxation method with parameter ω . It holds that*

$$\forall \omega \neq 0, \quad \rho(\mathbf{M}_\omega^{-1} \mathbf{N}_\omega) \geq |\omega - 1|.$$

Proof. We recall the following facts:

- the determinant of a product of matrices is equal to the product of the determinants.
- the determinant of a triangular matrix is equal to the product of its diagonal entries;

- the determinant of a matrix is equal to the product of its eigenvalues, to the power of their algebraic multiplicity. This can be shown from the previous two items, by passing to the Jordan normal form.

Therefore, we have that

$$\det(\mathbf{M}_\omega^{-1}\mathbf{N}_\omega) = \det(\mathbf{M}_\omega)^{-1} \det(\mathbf{N}_\omega) = \frac{\det\left(\frac{1-\omega}{\omega}\mathbf{D} - \mathbf{U}\right)}{\det\left(\frac{\mathbf{D}}{\omega} + \mathbf{L}\right)} = (1 - \omega)^n.$$

Since the determinant on the left-hand side is the product of the eigenvalues of $\mathbf{M}_\omega^{-1}\mathbf{N}_\omega$, it is bounded from above in modulus by $\rho(\mathbf{M}_\omega^{-1}\mathbf{N}_\omega)^n$, and so we deduce $\rho(\mathbf{M}_\omega^{-1}\mathbf{N}_\omega)^n \geq |1 - \omega|^n$. The statement then follows by taking the n -th root. \square

Comparison between Jacobi and Gauss–Seidel for tridiagonal matrices

For tridiagonal matrices, the convergence rate of the Jacobi and Gauss–Seidel methods satisfy an explicit relation, which we prove in this section. We denote the Jacobi and Gauss–Seidel splittings by $\mathbf{M}_\mathcal{J} - \mathbf{N}_\mathcal{J}$ and $\mathbf{M}_\mathcal{G} - \mathbf{N}_\mathcal{G}$, respectively, and use the following notation for the entries of the matrix \mathbf{A} :

$$\begin{pmatrix} a_1 & b_1 & & & \\ c_1 & \ddots & \ddots & & \\ & \ddots & \ddots & b_{n-1} & \\ & & c_{n-1} & a_n & \end{pmatrix}.$$

Before presenting and proving the result, notice that for any $\mu \neq 0$ it holds that

$$\begin{pmatrix} \mu & & & & \\ & \mu^2 & & & \\ & & \ddots & & \\ & & & \ddots & \\ & & & & \mu^n \end{pmatrix} \mathbf{A} \begin{pmatrix} \mu^{-1} & & & & \\ & \mu^{-2} & & & \\ & & \ddots & & \\ & & & \ddots & \\ & & & & \mu^{-n} \end{pmatrix} = \begin{pmatrix} a_1 & \mu^{-1}b_1 & & & \\ \mu c_1 & \ddots & \ddots & & \\ & \ddots & \ddots & \mu^{-1}b_{n-1} & \\ & & \mu c_{n-1} & a_n & \end{pmatrix}. \quad (2.19)$$

Proposition 2.15. *Assume that \mathbf{A} is tridiagonal with nonzero diagonal elements, so that both $\mathbf{M}_\mathcal{J} = \mathbf{D}$ and $\mathbf{M}_\mathcal{G} = \mathbf{L} + \mathbf{D}$ are invertible. Then*

$$\rho(\mathbf{M}_\mathcal{G}^{-1}\mathbf{N}_\mathcal{G}) = \rho(\mathbf{M}_\mathcal{J}^{-1}\mathbf{N}_\mathcal{J})^2$$

Proof. If λ is an eigenvalue of $\mathbf{M}_\mathcal{G}^{-1}\mathbf{N}_\mathcal{G}$ with associated unit eigenvector \mathbf{v} , then

$$\mathbf{M}_\mathcal{G}^{-1}\mathbf{N}_\mathcal{G}\mathbf{v} = \lambda\mathbf{v} \quad \Leftrightarrow \quad \mathbf{N}_\mathcal{G}\mathbf{v} = \lambda\mathbf{M}_\mathcal{G}\mathbf{v} \quad \Leftrightarrow \quad (\mathbf{N}_\mathcal{G} - \lambda\mathbf{M}_\mathcal{G})\mathbf{v} = 0.$$

For fixed λ , there exists a nontrivial solution \mathbf{v} to the last equation if and only if

$$p_\mathcal{G}(\lambda) := \det(\mathbf{N}_\mathcal{G} - \lambda\mathbf{M}_\mathcal{G}) = -\det(\lambda\mathbf{L} + \lambda\mathbf{D} + \mathbf{U}) = 0.$$

Likewise, λ is an eigenvalue of $M_{\mathcal{J}}^{-1}N_{\mathcal{J}}$ if and only if

$$p_{\mathcal{J}}(\lambda) := \det(N_{\mathcal{J}} - \lambda M_{\mathcal{J}}) = -\det(L + \lambda D + U) = 0.$$

Now notice that

$$p_{\mathcal{G}}(\lambda^2) = -\det(\lambda^2 L + \lambda^2 D + U) = -\lambda^n \det(\lambda L + (\lambda D) + \lambda^{-1} U).$$

Applying (2.19) with $\mu = \lambda \neq 0$, we deduce

$$p_{\mathcal{G}}(\lambda^2) = -\lambda^n \det(L + \lambda D + U) = \lambda^n p_{\mathcal{J}}(\lambda)$$

It is clear that this relation is true also if $\lambda = 0$. Consequently, it holds that if λ is an eigenvalue of the matrix $M_{\mathcal{J}}^{-1}N_{\mathcal{J}}$ then λ^2 is an eigenvalue of $M_{\mathcal{G}}^{-1}N_{\mathcal{G}}$. Conversely, if λ is an eigenvalue of $M_{\mathcal{G}}^{-1}N_{\mathcal{G}}$, then the two square roots of λ are eigenvalues of $M_{\mathcal{J}}^{-1}N_{\mathcal{J}}$. \square

If a matrix A is tridiagonal and Toeplitz, i.e. if it is of the form

$$\begin{pmatrix} a & b & & & \\ c & \ddots & \ddots & & \\ & \ddots & \ddots & b & \\ & & c & a & \end{pmatrix},$$

then it is possible to prove that the eigenvalues of A are given by

$$\lambda_k = a + 2\sqrt{bc} \cos\left(\frac{k\pi}{n+1}\right), \quad k = 1, \dots, n. \quad (2.20)$$

In this case, the spectral radius of $M_{\mathcal{J}}^{-1}N_{\mathcal{J}}$ can be determined explicitly.

Monitoring the convergence

In practice, we have access to the residual $\mathbf{r}^{(k)} = A\mathbf{x}^{(k)} - \mathbf{b}$ at each iteration, but not to the error $\mathbf{e}^{(k)} = \mathbf{x}^{(k)} - \mathbf{x}_*$, as calculating the latter would require to know the exact solution of the problem. Nevertheless, the two are related by the equation

$$\mathbf{r}^{(k)} = A\mathbf{e}^{(k)} \quad \Leftrightarrow \quad \mathbf{e}^{(k)} = A^{-1}\mathbf{r}^{(k)}.$$

Therefore, it holds that $\|\mathbf{e}^{(k)}\| \leq \|A^{-1}\| \|\mathbf{r}^{(k)}\|$. Likewise, the relative error satisfies

$$\frac{\|\mathbf{e}^{(k)}\|}{\|\mathbf{x}\|} = \frac{\|A^{-1}\mathbf{r}^{(k)}\|}{\|A^{-1}\mathbf{b}\|}$$

and since $\|\mathbf{b}\| = \|AA^{-1}\mathbf{b}\| \leq \|A\| \|A^{-1}\mathbf{b}\|$, we deduce

$$\frac{\|\mathbf{e}^{(k)}\|}{\|\mathbf{x}\|} \leq \kappa(A) \frac{\|\mathbf{r}^{(k)}\|}{\|\mathbf{b}\|}.$$

The fraction on the right-hand side is the *relative residual*. If the system is well conditioned, that is if $\kappa(A)$ is close to one, then controlling the residual enables a good control of the error.

Stopping criterion

In practice, several criteria can be employed in order to decide when to stop iterating. Given a small number ε (unrelated to the machine epsilon in Chapter 1), the following alternatives are available:

- Stop when $\mathbf{r}^{(k)} \leq \varepsilon$. The downside of this approach is that it is not *scaling invariant*: when used for solving the following rescaled system

$$k\mathbf{A}\mathbf{x} = k\mathbf{b}, \quad k \neq 1,$$

a splitting method with rescaled initial guess $k\mathbf{x}^{(0)}$ will require a number of iterations that depends on k : fewer if $k \ll 1$ and more if $k \gg 1$. In practice, controlling the relative residual and the relative error is often preferable.

- Stop when $\|\mathbf{r}^{(k)}\|/\|\mathbf{r}^{(0)}\| \leq \varepsilon$. This criterion is scaling invariant, but the number of iterations is dependent on the quality of the initial guess $\mathbf{x}^{(0)}$.
- Stop when $\|\mathbf{r}^{(k)}\|/\|\mathbf{b}\| \leq \varepsilon$. This criterion is generally the best, because it is both scaling invariant and the quality of the final iterate is independent of that of the initial guess.

2.3.2 The conjugate gradient method

As already mentioned in Remark 2.2, when the matrix $\mathbf{A} \in \mathbf{R}^{n \times n}$ in the linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$ is symmetric and positive definite, the system can be interpreted as a minimization problem for the function

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} - \mathbf{b}^T\mathbf{x}. \quad (2.21)$$

The fact that the exact solution \mathbf{x}_* to the linear system is the unique minimizer of this function appears clearly when rewriting f as follows:

$$f(\mathbf{x}) = \frac{1}{2}(\mathbf{x} - \mathbf{x}_*)^T\mathbf{A}(\mathbf{x} - \mathbf{x}_*) - \frac{1}{2}\mathbf{x}_*^T\mathbf{A}\mathbf{x}_*. \quad (2.22)$$

The second term is constant with \mathbf{x} , and the first term is strictly positive if $\mathbf{x} - \mathbf{x}_* \neq \mathbf{0}$, because \mathbf{A} is positive definite. We saw that Richardson's method can be interpreted as a steepest descent with fixed step size,

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \omega\nabla f(\mathbf{x}^{(k)}).$$

In this section, we will present and study other methods for solving the linear system (2.1) which can be viewed as optimization methods. Since \mathbf{A} is symmetric, it is diagonalizable and

the function f can be rewritten as

$$\begin{aligned} f(\mathbf{x}) &= \frac{1}{2}(\mathbf{x} - \mathbf{x}_*)^T \mathbf{Q} \mathbf{D} \mathbf{Q}^T (\mathbf{x} - \mathbf{x}_*) - \frac{1}{2} \mathbf{x}_*^T \mathbf{A} \mathbf{x}_* \\ &= \frac{1}{2} (\mathbf{Q}^T \mathbf{e})^T \mathbf{D} (\mathbf{Q}^T \mathbf{e}) - \frac{1}{2} \mathbf{x}_*^T \mathbf{A} \mathbf{x}_*, \quad \mathbf{e} = \mathbf{x} - \mathbf{x}_*. \end{aligned}$$

Therefore, we have that

$$f(\mathbf{x}) = \frac{1}{2} \sum_{i=1}^n \lambda_i \eta_i^2 - \frac{1}{2} \mathbf{x}_*^T \mathbf{A} \mathbf{x}_*, \quad \boldsymbol{\eta} = \mathbf{Q}^T (\mathbf{x} - \mathbf{x}_*),$$

where $(\lambda_i)_{1 \leq i \leq n}$ are the diagonal entries of \mathbf{D} . This shows that f is a paraboloid after a change of coordinates.

Steepest descent method

The steepest descent method is more general than Richardson's method in the sense that the step size changes from iteration to iteration and the method is not restricted to quadratic functions of the form (2.21). Each iteration is of the form

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \omega_k \nabla f(\mathbf{x}^{(k)}).$$

It is natural to wonder whether the step size ω_k can be fixed in such a way that $f(\mathbf{x}^{(k+1)})$ is as small as possible. For the case of the quadratic function (2.21), this value of ω_k can be calculated explicitly for a general search direction \mathbf{d} , and in particular also when $\mathbf{d} = \nabla f(\mathbf{x}^{(k)})$. We calculate that

$$\begin{aligned} f(\mathbf{x}^{(k+1)}) &= f(\mathbf{x}^{(k)} - \omega_k \mathbf{d}) = \frac{1}{2} (\mathbf{x}^{(k)} - \omega_k \mathbf{d})^T \mathbf{A} (\mathbf{x}^{(k)} - \omega_k \mathbf{d}) - (\mathbf{x}^{(k)} - \omega_k \mathbf{d})^T \mathbf{b} \\ &= f(\mathbf{x}^{(k)}) + \frac{\omega_k^2}{2} \mathbf{d}^T \mathbf{A} \mathbf{d} - \omega_k \mathbf{d}^T \mathbf{r}^{(k)}. \end{aligned} \quad (2.23)$$

When viewed as a function of the real parameter ω_k , the right-hand side is a convex quadratic function. It is minimized when its derivative is equal to zero, i.e. when

$$\omega_k \mathbf{d}^T \mathbf{A} \mathbf{d} - \mathbf{d}^T (\mathbf{A} \mathbf{x}^{(k)} - \mathbf{b}) = 0 \quad \Rightarrow \quad \omega_k = \frac{\mathbf{d}^T \mathbf{r}^{(k)}}{\mathbf{d}^T \mathbf{A} \mathbf{d}}. \quad (2.24)$$

The steepest descent algorithm with step size obtained from this equation is summarized in [Algorithm 3](#) below. By construction, the function value $f(\mathbf{x}^{(k)})$ is nonincreasing with k , which is equivalent to saying that the error $\mathbf{x} - \mathbf{x}_*$ is nonincreasing in the norm $\mathbf{x} \mapsto \sqrt{\mathbf{x}^T \mathbf{A} \mathbf{x}}$. In order to quantify more precisely the decrease of the error in this norm, we introduce the notation

$$E_k = \|\mathbf{x} - \mathbf{x}_*\|_{\mathbf{A}}^2 := (\mathbf{x}^{(k)} - \mathbf{x}_*)^T \mathbf{A} (\mathbf{x}^{(k)} - \mathbf{x}_*) = (\mathbf{A} \mathbf{x}^{(k)} - \mathbf{b})^T \mathbf{A}^{-1} (\mathbf{A} \mathbf{x}^{(k)} - \mathbf{b}).$$

We begin by showing the following auxiliary lemma.

Lemma 2.16 (Kantorovich inequality). *Let $\mathbf{A} \in \mathbf{R}^{n \times n}$ be a symmetric and positive definite matrix, and let $\lambda_0 \leq \dots \leq \lambda_n$ denote its eigenvalues. Then for all nonzero $\mathbf{z} \in \mathbf{R}^n$ it holds that*

$$\frac{(\mathbf{z}^T \mathbf{z})^2}{(\mathbf{z}^T \mathbf{A} \mathbf{z})(\mathbf{z}^T \mathbf{A}^{-1} \mathbf{z})} \geq \frac{4\lambda_1 \lambda_n}{(\lambda_1 + \lambda_n)^2}.$$

Proof. By the AM-GM (arithmetic mean-geometric mean) inequality, it holds for all $t > 0$ that

$$\begin{aligned} \sqrt{(\mathbf{z}^T \mathbf{A} \mathbf{z})(\mathbf{z}^T \mathbf{A}^{-1} \mathbf{z})} &= \sqrt{(t\mathbf{z}^T \mathbf{A} \mathbf{z})(t^{-1}\mathbf{z}^T \mathbf{A}^{-1} \mathbf{z})} \leq \frac{1}{2} \left(t\mathbf{z}^T \mathbf{A} \mathbf{z} + \frac{1}{t}\mathbf{z}^T \mathbf{A}^{-1} \mathbf{z} \right) \\ &= \frac{1}{2}\mathbf{z}^T \left(t\mathbf{A} + \frac{1}{t}\mathbf{A}^{-1} \right) \mathbf{z}. \end{aligned}$$

The matrix on the right-hand side is also symmetric and positive definite, with eigenvalues equal to $t\lambda_i + (t\lambda_i)^{-1}$. Therefore, we deduce

$$\forall t \geq 0, \quad \sqrt{(\mathbf{z}^T \mathbf{A} \mathbf{z})(\mathbf{z}^T \mathbf{A}^{-1} \mathbf{z})} \leq \frac{1}{2} \left(\max_{i \in \{1, \dots, n\}} t\lambda_i + (t\lambda_i)^{-1} \right) \mathbf{z}^T \mathbf{z}. \quad (2.25)$$

The function $x \mapsto x + x^{-1}$ is convex, and so over any closed interval $[x_{\min}, x_{\max}]$ it attains its maximum either at x_{\min} or at x_{\max} . Consequently, it holds that

$$\left(\max_{i \in \{1, \dots, n\}} t\lambda_i + (t\lambda_i)^{-1} \right) = \max \left\{ t\lambda_1 + \frac{1}{t\lambda_1}, t\lambda_n + \frac{1}{t\lambda_n} \right\}.$$

In order to obtain the best possible bound from (2.25), we should let t be such that the maximum is minimized, which occurs when the two arguments of the maximum are equal:

$$t\lambda_1 + \frac{1}{t\lambda_1} = t\lambda_n + \frac{1}{t\lambda_n} \quad \Rightarrow \quad t = \frac{1}{\sqrt{\lambda_1 \lambda_n}}.$$

For this value of t , the maximum in (2.25) is equal to

$$\sqrt{\frac{\lambda_1}{\lambda_n}} + \sqrt{\frac{\lambda_n}{\lambda_1}}.$$

By substituting this expression in (2.25) and rearranging, we obtain the statement. \square

We are now able to prove the convergence of the steepest descent method.

Theorem 2.17 (Convergence of the steepest descent method). *It holds that*

$$E_{k+1} \leq \left(\frac{\kappa_2(\mathbf{A}) - 1}{\kappa_2(\mathbf{A}) + 1} \right)^2 E_k.$$

Proof. Substituting $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \omega_k \mathbf{d}$ in the expression for E_{k+1} , we obtain

$$\begin{aligned} E_{k+1} &= (\mathbf{x}^{(k)} - \omega_k \mathbf{d} - \mathbf{x}_*)^T \mathbf{A} (\mathbf{x}^{(k)} - \omega_k \mathbf{d} - \mathbf{x}_*) \\ &= E_k - 2\omega_k \mathbf{d}^T \mathbf{A} \mathbf{r}^{(k)} + \omega_k^2 \mathbf{d}^T \mathbf{A} \mathbf{d} \\ &= E_k - \frac{(\mathbf{d}^T \mathbf{d})^2}{\mathbf{d}^T \mathbf{A} \mathbf{d}} = \left(1 - \frac{(\mathbf{d}^T \mathbf{d})^2}{(\mathbf{d}^T \mathbf{A} \mathbf{d})(\mathbf{d}^T \mathbf{A}^{-1} \mathbf{d})} \right) E_k, \end{aligned}$$

Using the Kantorovich inequality, we have

$$E_{k+1} \leq \left(1 - \frac{4\lambda_1 \lambda_n}{(\lambda_1 + \lambda_n)^2} \right) E_k \leq \left(\frac{\lambda_1 - \lambda_n}{\lambda_1 + \lambda_n} \right)^2 E_k = \left(\frac{\kappa_2(\mathbf{A}) - 1}{\kappa_2(\mathbf{A}) + 1} \right)^2 E_k.$$

We immediately deduce the statement from this inequality. \square

Algorithm 3 Steepest descent method

- 1: Pick ε and initial \mathbf{x}
 - 2: $\mathbf{r} \leftarrow \mathbf{A}\mathbf{x} - \mathbf{b}$
 - 3: **while** $\|\mathbf{r}\| \geq \varepsilon\|\mathbf{b}\|$ **do**
 - 4: $\mathbf{d} \leftarrow \mathbf{r}$
 - 5: $\omega \leftarrow \mathbf{d}^T \mathbf{r} / \mathbf{d}^T \mathbf{A} \mathbf{d}$
 - 6: $\mathbf{x} \leftarrow \mathbf{x} - \omega \mathbf{d}$
 - 7: $\mathbf{r} \leftarrow \mathbf{A}\mathbf{x} - \mathbf{b}$
 - 8: **end while**
-

Preconditioned steepest descent

We observe from [Theorem 2.17](#) that the convergence of the steepest descent method is faster when the condition number of the matrix \mathbf{A} is low. This naturally leads to the following question: can we reformulate the minimization of $f(\mathbf{x})$ in (2.21) as another optimization problem which is of the same form but involves a matrix with a lower condition number, thereby providing scope for faster convergence? In order to answer this question, we consider a linear change of coordinates $\mathbf{y} = \mathbf{T}^{-1}\mathbf{x}$, where \mathbf{T} is an invertible matrix, and we define

$$\tilde{f}(\mathbf{y}) = f(\mathbf{T}\mathbf{y}) = \frac{1}{2} \mathbf{y}^T (\mathbf{T}^T \mathbf{A} \mathbf{T}) \mathbf{y} - (\mathbf{T}^T \mathbf{b})^T \mathbf{y}. \quad (2.26)$$

This function is of the same form as f in (2.21), with the matrix $\tilde{\mathbf{A}} := \mathbf{T}^T \mathbf{A} \mathbf{T}$ instead of \mathbf{A} and the vector $\tilde{\mathbf{b}} := \mathbf{T}^T \mathbf{b}$ instead of \mathbf{b} . Its minimizer is $\mathbf{y}_* = \mathbf{T}^{-1} \mathbf{x}_*$. The steepest descent algorithm can be applied to (2.26) and, from an approximation $\mathbf{y}^{(k)}$ of the minimizer \mathbf{y}_* , an approximation $\mathbf{x}^{(k)}$ of \mathbf{x}_* is obtained by the change of variable $\mathbf{x}^{(k)} = \mathbf{T}\mathbf{y}^{(k)}$. This approach is called *preconditioning*. By [Theorem 2.17](#), the steepest descent method satisfies the following error estimate when applied to the function (2.26):

$$\begin{aligned} E_{k+1} &\leq \left(\frac{\kappa_2(\mathbf{T}^T \mathbf{A} \mathbf{T}) - 1}{\kappa_2(\mathbf{T}^T \mathbf{A} \mathbf{T}) + 1} \right)^2 E_k, & E_k &= (\mathbf{y}^{(k)} - \mathbf{y}_*)^T \tilde{\mathbf{A}} (\mathbf{y}^{(k)} - \mathbf{y}_*), \\ & & &= (\mathbf{x}^{(k)} - \mathbf{x}_*)^T \mathbf{A} (\mathbf{x}^{(k)} - \mathbf{x}_*). \end{aligned}$$

Consequently, the convergence is faster than that of the usual steepest descent method if $\kappa_2(\mathsf{T}^T\mathsf{A}\mathsf{T}) < \kappa_2(\mathsf{A})$. The optimal change of coordinates is given by $\mathsf{T} = \mathsf{C}^{-T}$, where C is the factor of the Cholesky factorization of A as $\mathsf{C}\mathsf{C}^T$. Indeed, in this case

$$\mathsf{T}^T\mathsf{A}\mathsf{T} = \mathsf{C}^{-1}\mathsf{C}\mathsf{C}^T\mathsf{C}^{-T} = \mathsf{I} \quad \Rightarrow \quad \kappa_2(\mathsf{T}^T\mathsf{A}\mathsf{T}) = 1,$$

and the method converges in a single iteration! However, this iteration amounts to solving the linear system by direct Cholesky factorization of A . In practice, it is usual to define T from an approximation of the Cholesky factorization, such as the *incomplete Cholesky factorization*.

To conclude this section, we demonstrate that the change of variable from \mathbf{x} to \mathbf{y} need not be performed explicitly in practice. Indeed, one step of the steepest descent algorithm applied to function \tilde{f} reads

$$\mathbf{y}^{(k+1)} = \mathbf{y}^{(k)} - \tilde{\omega}_k(\tilde{\mathsf{A}}\mathbf{y}^{(k)} - \tilde{\mathbf{b}}), \quad \tilde{\omega}_k = \frac{(\tilde{\mathsf{A}}\mathbf{y}^{(k)} - \tilde{\mathbf{b}})^T(\tilde{\mathsf{A}}\mathbf{y}^{(k)} - \tilde{\mathbf{b}})}{(\tilde{\mathsf{A}}\mathbf{y}^{(k)} - \tilde{\mathbf{b}})^T\tilde{\mathsf{A}}(\tilde{\mathsf{A}}\mathbf{y}^{(k)} - \tilde{\mathbf{b}})}.$$

Letting $\mathbf{x}^{(k)} = \mathsf{T}\mathbf{y}^{(k)}$, this equation can be rewritten as the following iteration:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \tilde{\omega}_k\mathbf{d}_k, \quad \tilde{\omega}_k = \frac{\mathbf{d}_k^T\mathbf{r}^{(k)}}{\mathbf{d}_k^T\mathsf{A}\mathbf{d}_k}, \quad \mathbf{d}_k = \mathsf{T}^T\mathsf{T}(\mathsf{A}\mathbf{x}^{(k)} - \mathbf{b}).$$

A comparison with (2.24) shows that the step size $\tilde{\omega}_k$ is such that $f(\mathbf{x}^{(k+1)})$ is minimized. This reasoning shows that the preconditioned conjugate gradient method amounts to choosing the direction $\mathbf{d}^{(k)} = \mathsf{T}\mathsf{T}^T\mathbf{r}^{(k)}$ at each iteration, instead of just $\mathbf{r}^{(k)}$, as is apparent in Algorithm 4. It is simple to check that $-\mathbf{d}^{(k)}$ is a descent direction for f :

$$-\nabla f(\mathbf{x})^T(\mathsf{T}^T\mathsf{T}(\mathsf{A}\mathbf{x} - \mathbf{b})) = -(\mathsf{T}(\mathsf{A}\mathbf{x} - \mathbf{b}))^T(\mathsf{T}(\mathsf{A}\mathbf{x} - \mathbf{b})) \leq 0.$$

Algorithm 4 Preconditioned steepest descent method

- 1: Pick ε , invertible T and initial \mathbf{x}
 - 2: $\mathbf{r} \leftarrow \mathsf{A}\mathbf{x} - \mathbf{b}$
 - 3: **while** $\|\mathbf{r}\| \geq \varepsilon\|\mathbf{b}\|$ **do**
 - 4: $\mathbf{d} \leftarrow \mathsf{T}^T\mathsf{T}\mathbf{r}$
 - 5: $\omega \leftarrow \mathbf{d}^T\mathbf{r}/\mathbf{d}^T\mathsf{A}\mathbf{d}$
 - 6: $\mathbf{x} \leftarrow \mathbf{x} - \omega\mathbf{d}$
 - 7: $\mathbf{r} \leftarrow \mathsf{A}\mathbf{x} - \mathbf{b}$
 - 8: **end while**
-

Conjugate directions method

Definition 2.6 (Conjugate directions). Let A be a symmetric positive definite matrix. Two vectors \mathbf{d}_1 and \mathbf{d}_2 are called A -orthogonal or conjugate with respect to A if $\mathbf{d}_1^T\mathsf{A}\mathbf{d}_2 = 0$, i.e. if they are orthogonal for the inner product $\langle \mathbf{x}, \mathbf{y} \rangle_{\mathsf{A}} = \mathbf{x}^T\mathsf{A}\mathbf{y}$.

Assume that $\mathbf{d}_0, \dots, \mathbf{d}_{n-1}$ are n pairwise A -orthogonal nonzero directions. By Exercise 2.20,

these vectors are linearly independent, and so they form a basis of \mathbf{R}^n . Consequently, for any initial guess $\mathbf{x}^{(0)}$, the vector $\mathbf{x}^{(0)} - \mathbf{x}_*$, where \mathbf{x}_* is the solution to the linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$, can be decomposed as

$$\mathbf{x}^{(0)} - \mathbf{x}_* = \alpha_0 \mathbf{d}_0 + \cdots + \alpha_{n-1} \mathbf{d}_{n-1}.$$

Taking the $\langle \bullet, \bullet \rangle_{\mathbf{A}}$ inner product of both sides with \mathbf{d}_k , with $k \in \{0, \dots, n-1\}$, we obtain an expression for the scalar coefficient α_k

$$\alpha_k = \frac{\mathbf{d}_k^T \mathbf{A}(\mathbf{x}^{(0)} - \mathbf{x}_*)}{\mathbf{d}_k^T \mathbf{A} \mathbf{d}_k} = \frac{\mathbf{d}_k^T (\mathbf{A} \mathbf{x}^{(0)} - \mathbf{b})}{\mathbf{d}_k^T \mathbf{A} \mathbf{d}_k}.$$

Therefore, calculating the expression of the coefficient does not require to know the exact solution \mathbf{x}_* . Given conjugate directions, the exact solution can be obtained as

$$\mathbf{x}_* = \mathbf{x}^{(0)} - \sum_{k=0}^{n-1} \alpha_k \mathbf{d}_k, \quad \alpha_k = \frac{\mathbf{d}_k^T \mathbf{r}^{(0)}}{\mathbf{d}_k^T \mathbf{A} \mathbf{d}_k}. \quad (2.27)$$

From this equation we deduce, denoting by \mathbf{D} the matrix $(\mathbf{d}_0 \ \dots \ \mathbf{d}_{n-1})$, that the inverse of \mathbf{A} can be expressed as

$$\mathbf{A}^{-1} = \mathbf{D}(\mathbf{D}^T \mathbf{A} \mathbf{D})^{-1} \mathbf{D}^T = \sum_{k=0}^{n-1} \frac{\mathbf{d}_k \mathbf{d}_k^T}{\nu_k}, \quad \nu_k = \mathbf{d}_k^T \mathbf{A} \mathbf{d}_k.$$

In this equation, ν_k is the k -th diagonal entry of the matrix $\mathbf{D}^T \mathbf{A} \mathbf{D}$. The conjugate directions method is illustrated in [Algorithm 5](#). Its implementation is very similar to the steepest descent method, the only difference being that the descent direction at iteration k is given by \mathbf{d}_k instead of $\mathbf{r}^{(k)}$. In particular, the step size at each iteration is such that $f(\mathbf{x}^{(k+1)})$ is minimized.

Algorithm 5 Conjugate directions method

- 1: Assuming $\mathbf{d}_0, \dots, \mathbf{d}_{n-1}$ are given.
 - 2: Pick initial $\mathbf{x}^{(0)}$
 - 3: **for** k in $\{0, \dots, n-1\}$ **do**
 - 4: $\mathbf{r}^{(k)} = \mathbf{A}\mathbf{x}^{(k)} - \mathbf{b}$
 - 5: $\omega_k = \mathbf{d}_k^T \mathbf{r}^{(k)} / \mathbf{d}_k^T \mathbf{A} \mathbf{d}_k$
 - 6: $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \omega_k \mathbf{d}_k$
 - 7: **end for**
-

Let us now establish the connection between the [Algorithm 5](#) and (2.27), which is not immediately apparent because (2.27) involves only the initial residual $\mathbf{A}\mathbf{x}^{(0)} - \mathbf{b}$, while the residual at the current iteration $\mathbf{r}^{(k)}$ is used in the algorithm.

Proposition 2.18 (Convergence of the conjugate directions method). *The vector $\mathbf{x}^{(k)}$ obtained after k iterations of the conjugate directions method is given by*

$$\mathbf{x}^{(k)} = \mathbf{x}^{(0)} - \sum_{i=0}^{k-1} \alpha_i \mathbf{d}_i, \quad \alpha_i = \frac{\mathbf{d}_i^T \mathbf{r}^{(0)}}{\mathbf{d}_i^T \mathbf{A} \mathbf{d}_i}. \quad (2.28)$$

In particular, the method converges in at most n iterations.

Proof. Let us denote by $\mathbf{y}^{(k)}$ the solution obtained after k steps of Algorithm 5. Our goal is to show that $\mathbf{y}^{(k)}$ coincides with $\mathbf{x}^{(k)}$ defined in (2.28). The result is trivial for $k = 0$. Reasoning by induction, we assume that it holds true up to k . Then performing step $k + 1$ of the algorithm gives

$$\mathbf{y}^{(k+1)} = \mathbf{y}^{(k)} - \omega_k \mathbf{d}_k, \quad \omega_k = \frac{\mathbf{d}_k^T \mathbf{r}^{(k)}}{\mathbf{d}_k^T \mathbf{A} \mathbf{d}_k}.$$

On the other hand, it holds from (2.28) that

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \alpha_k \mathbf{d}_k, \quad \alpha_k = \frac{\mathbf{d}_k^T \mathbf{r}^{(0)}}{\mathbf{d}_k^T \mathbf{A} \mathbf{d}_k}.$$

By the induction hypothesis it holds that $\mathbf{y}^{(k)} = \mathbf{x}^{(k)}$ and so, in order to prove the equality $\mathbf{y}^{(k+1)} = \mathbf{x}^{(k+1)}$, it is sufficient to show that $\omega_k = \alpha_k$, i.e. that

$$\mathbf{d}_k^T \mathbf{r}^{(k)} = \mathbf{d}_k^T \mathbf{r}^{(0)} \Leftrightarrow \mathbf{d}_k^T (\mathbf{r}^{(k)} - \mathbf{r}^{(0)}) = 0 \Leftrightarrow \mathbf{d}_k^T \mathbf{A} (\mathbf{x}^{(k)} - \mathbf{x}^{(0)}) = 0.$$

The latter equality is obvious from the A-orthonormality of the directions. \square

Since ω_k in Algorithm 5 coincides with the expression in (2.24), the conjugate directions algorithm satisfies the following “local optimization” property: the iterate $\mathbf{x}^{(x+1)}$ minimizes f on the straight line $\omega \mapsto \mathbf{x}^{(k)} - \omega \mathbf{d}_k$. In fact, it also satisfies the following stronger property.

Proposition 2.19 (Optimality of the conjugate directions method). *The iterate $\mathbf{x}^{(k)}$ is the minimizer of f over the set $\mathbf{x}^{(0)} + \mathcal{B}_k$, where $\mathcal{B}_k = \text{span}\{\mathbf{d}_0, \dots, \mathbf{d}_{k-1}\}$.*

Proof. By (2.27), it holds that

$$\mathbf{x}_* = \mathbf{x}^{(0)} - \sum_{i=0}^{n-1} \alpha_i \mathbf{d}_i, \quad \alpha_i = \frac{\mathbf{d}_i^T \mathbf{r}^{(0)}}{\mathbf{d}_i^T \mathbf{A} \mathbf{d}_i}$$

On the other hand, any vector $\mathbf{y} \in \mathbf{x}^{(0)} + \mathcal{B}_k$ can be expanded as

$$\mathbf{y} = \mathbf{x}^{(0)} - \beta_0 \mathbf{d}_0 - \dots - \beta_{k-1} \mathbf{d}_{k-1}.$$

Employing these two expressions, the formula for f in (2.22), and the A-orthogonality of the directions, we obtain

$$\begin{aligned} f(\mathbf{y}) &= \frac{1}{2} (\mathbf{y} - \mathbf{x}_*)^T \mathbf{A} (\mathbf{y} - \mathbf{x}_*) - \frac{1}{2} \mathbf{x}_*^T \mathbf{A} \mathbf{x}_* \\ &= \frac{1}{2} \sum_{i=1}^{k-1} (\beta_i - \alpha_i)^2 \mathbf{d}_i^T \mathbf{A} \mathbf{d}_i + \frac{1}{2} \sum_{i=k}^{n-1} \alpha_i^2 \mathbf{d}_i^T \mathbf{A} \mathbf{d}_i - \frac{1}{2} \mathbf{x}_*^T \mathbf{A} \mathbf{x}_* \end{aligned}$$

This is minimized when $\beta_i = \alpha_i$ for all $i \in \{0, \dots, k-1\}$, in which case \mathbf{y} coincides with the k -th iterate $\mathbf{x}^{(k)}$ of the conjugate directions method in view of Proposition 2.18. \square

Remark 2.3. Let $\|\cdot\|_A$ denote the norm induced by the inner product $(\mathbf{x}, \mathbf{y}) \mapsto \mathbf{x}^T \mathbf{A} \mathbf{y}$. Since

$$\|\mathbf{x}^{(k)} - \mathbf{x}_*\|_A = \sqrt{2f(\mathbf{x}^{(k)}) + \mathbf{x}_*^T \mathbf{A} \mathbf{x}_*},$$

[Proposition 2.19](#) shows that $\mathbf{x}^{(k)}$ minimizes the norm $\|\mathbf{x}^{(k)} - \mathbf{x}_*\|_A$ over $\mathbf{x}^{(0)} + \mathcal{B}_k$.

A corollary of (2.19) is that the gradient of f at $\mathbf{x}^{(k)}$, i.e. the residual $\mathbf{r}^{(k)} = \mathbf{A}\mathbf{x}^{(k)} - \mathbf{b}$, is orthogonal to any vector in $\{\mathbf{d}_0, \dots, \mathbf{d}_{k-1}\}$ for the usual Euclidean inner product. This can also be checked directly from the formula

$$\mathbf{x}^{(k)} - \mathbf{x}_* = \sum_{i=k}^{n-1} \alpha_i \mathbf{d}_i, \quad \alpha_i = \frac{\mathbf{d}_i^T \mathbf{r}^{(0)}}{\mathbf{d}_i^T \mathbf{A} \mathbf{d}_i},$$

which follows directly from [Proposition 2.18](#). Indeed, it holds that

$$\forall j \in \{0, \dots, k-1\}, \quad \mathbf{d}_j^T \mathbf{r}^{(k)} = \mathbf{A}(\mathbf{x}^{(k)} - \mathbf{x}_*) = \sum_{i=k}^{n-1} \alpha_i \mathbf{d}_j^T \mathbf{d}_i = 0. \quad (2.29)$$

The conjugate gradient method

In the previous section, we showed that, given n conjugate directions, the solution to the linear system $\mathbf{A}\mathbf{x} = \mathbf{b}$ can be obtained in a finite number of iterations using [Algorithm 5](#). The conjugate gradient method can be viewed as a particular case of the conjugate directions method. Instead of assuming that the conjugate directions are given, they are constructed iteratively as part of the algorithm. Given an initial guess $\mathbf{x}^{(0)}$, the first direction is the residual $\mathbf{r}^{(0)}$, which coincides with the gradient of f at $\mathbf{x}^{(0)}$. The directions employed for the next iterations are obtained by applying the Gram-Schmidt process to the residuals. More precisely, given conjugate directions $\mathbf{d}_1, \dots, \mathbf{d}_{k-1}$, and letting $\mathbf{x}^{(k)}$ denote the k -th iterate of the conjugate directions method, the direction \mathbf{d}_k is obtained by

$$\mathbf{d}_k = \mathbf{r}^{(k)} - \sum_{i=0}^{k-1} \frac{\mathbf{d}_i^T \mathbf{A} \mathbf{r}^{(k)}}{\mathbf{d}_i^T \mathbf{A} \mathbf{d}_i} \mathbf{d}_i, \quad \mathbf{r}^{(k)} = \mathbf{A}\mathbf{x}^{(k)} - \mathbf{b}. \quad (2.30)$$

It is simple to check that \mathbf{d}_k is indeed \mathbf{A} -orthogonal to \mathbf{d}_i for $i \in \{0, \dots, k-1\}$, and that \mathbf{d}_k is nonzero if $\mathbf{r}^{(k)}$ is nonzero. To prove the latter claim, we can take the Euclidean inner product of both sides with $\mathbf{r}^{(k)}$ and use [Proposition 2.19](#) to deduce that

$$\mathbf{d}_k^T \mathbf{r}^{(k)} = (\mathbf{r}^{(k)})^T \mathbf{r}^{(k)} > 0. \quad (2.31)$$

It appears from (2.30) that the cost of calculating a new direction grows linearly with the iteration index. In fact, it turns out that only the last term in the sum is nonzero, and so the cost of computing a new direction does not grow with the iteration index k . In order to explain why only the last term of the sum in (2.30) is nonzero, we begin by making a couple of observations:

- Since the directions are obtained from the residuals, it holds that

$$\forall k \in \{0, \dots, n-1\}, \quad \text{span}\{\mathbf{d}_0, \dots, \mathbf{d}_k\} = \text{span}\{\mathbf{r}^{(0)}, \dots, \mathbf{r}^{(k)}\}. \quad (2.32)$$

- A calculation gives that

$$\mathbf{r}^{(i+1)} = \mathbf{A}\mathbf{x}^{(i+1)} - \mathbf{b} = \mathbf{A}(\mathbf{x}^{(i)} - \omega_i \mathbf{d}_i) - \mathbf{b} = \mathbf{r}^{(i)} - \omega_i \mathbf{A}\mathbf{d}_i. \quad (2.33)$$

Note that $\omega_i > 0$ if $\mathbf{r}^{(i)} \neq \mathbf{0}$ by (2.31). Rearranging this equation and noting that the difference $\mathbf{r}^{(i+1)} - \mathbf{r}^{(i)}$ is a linear combination of the first $i+1$ conjugate directions by (2.32), we deduce that there exist scalar coefficients $(\alpha_{i,j})_{0 \leq j \leq i+1}$ such that

$$\mathbf{A}\mathbf{d}_i = \frac{1}{\omega_i} \left(\mathbf{r}^{(i+1)} - \mathbf{r}^{(i)} \right) = \sum_{j=0}^{i+1} \alpha_{i,j} \mathbf{d}_j.$$

These two observations imply that

$$\mathbf{d}_i^T \mathbf{A}\mathbf{r}^{(k)} = (\mathbf{A}\mathbf{d}_i)^T \mathbf{r}^{(k)} = \left(\sum_{j=0}^{i+1} \alpha_{i,j} \mathbf{d}_j \right)^T \mathbf{r}^{(k)}.$$

Using the orthonormality property (2.29) for the residual of the conjugate directions method, we deduce that the right-hand side of this equation is zero for all $i \in \{0, \dots, k-2\}$, implying that only the last term in the sum of (2.30) is nonzero. This leads to Algorithm 6.

Algorithm 6 Conjugate gradient method

- 1: Pick initial $\mathbf{x}^{(0)}$
 - 2: $\mathbf{d}_0 = \mathbf{r}^{(0)} = \mathbf{A}\mathbf{x}^{(0)} - \mathbf{b}$
 - 3: **for** k in $\{0, \dots, n-1\}$ **do**
 - 4: **if** $\|\mathbf{r}^{(k)}\| = 0$ **then**
 - 5: Stop
 - 6: **end if**
 - 7: $\omega_k = \mathbf{d}_k^T \mathbf{r}^{(k)} / \mathbf{d}_k^T \mathbf{A}\mathbf{d}_k$
 - 8: $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \omega_k \mathbf{d}_k$
 - 9: $\mathbf{r}^{(k+1)} = \mathbf{A}\mathbf{x}^{(k+1)} - \mathbf{b}$
 - 10: $\beta_k = \mathbf{d}_k^T \mathbf{A}\mathbf{r}^{(k+1)} / \mathbf{d}_k^T \mathbf{A}\mathbf{d}_k$.
 - 11: $\mathbf{d}_{k+1} = \mathbf{r}^{(k+1)} - \beta_k \mathbf{d}_k$.
 - 12: **end for**
-

The subspace spanned by the descent directions of the conjugate gradient method can be characterized precisely as follows.

Proposition 2.20. *Assume that $\|\mathbf{r}^{(k)}\| \neq 0$ for $k < m \leq n$, Then it holds that*

$$\forall k \in \{0, \dots, m\}, \quad \text{span}\{\mathbf{r}^{(0)}, \mathbf{r}^{(1)}, \dots, \mathbf{r}^{(k)}\} = \text{span}\{\mathbf{r}^{(0)}, \mathbf{A}\mathbf{r}^{(0)}, \dots, \mathbf{A}^k \mathbf{r}^{(0)}\}$$

The subspace on the right-hand side is called a Krylov subspace and denoted \mathcal{B}_{k+1} .

Proof. The result is clear for $k = 0$. Reasoning by induction, we prove that if the result is true up to $k < m$, then it is also true for $k + 1$. From (2.33) we have

$$\mathbf{r}^{(k+1)} = \mathbf{r}^{(k)} - \omega_k \mathbf{A} \mathbf{d}_k. \quad (2.34)$$

By the induction hypothesis and (2.32), there exist scalar coefficients $(\gamma_i)_{0 \leq i \leq k}$ such that

$$\mathbf{d}_k = \gamma_0 \mathbf{r}^{(0)} + \gamma_1 \mathbf{A} \mathbf{r}^{(0)} + \cdots + \gamma_k \mathbf{A}^k \mathbf{r}^{(0)}$$

The coefficient γ_k is necessarily nonzero, otherwise \mathbf{d}_k would be a linear combination of the previous conjugate directions. The residual $\mathbf{r}^{(k)}$ admits a decomposition of the same form:

$$\mathbf{r}^{(k)} = \beta_0 \mathbf{r}^{(0)} + \beta_1 \mathbf{A} \mathbf{r}^{(0)} + \cdots + \beta_k \mathbf{A}^k \mathbf{r}^{(0)}$$

Substituting these expressions in (2.34), we conclude that

$$\mathbf{r}^{(k+1)} = \beta_0 \mathbf{r}^{(0)} + (\beta_1 - \omega_0 \gamma_0) \mathbf{A} \mathbf{r}^{(0)} + \cdots + (\beta_k - \omega_{k-1} \gamma_{k-1}) \mathbf{A}^k \mathbf{r}^{(0)} - \omega_k \gamma_k \mathbf{A}^{(k+1)} \mathbf{r}^{(0)}.$$

Since both ω_k and γ_k are nonzero, the proof is complete. \square

Although the conjugate gradient method converges in a finite number of iterations, performing n iterations for very large systems would require an excessive computational cost, and so it is sometimes desirable to stop iterating when the residual is sufficiently small. To conclude this section, we study the convergence of the method.

Theorem 2.21 (Convergence of the conjugate gradient method). *The error for the conjugate gradient method, measured as $E_{k+1} := (\mathbf{x}^{(k+1)} - \mathbf{x}_*)^T \mathbf{A} (\mathbf{x}^{(k+1)} - \mathbf{x}_*)$, satisfies the following inequality:*

$$\forall q_k \in \mathbf{P}(k), \quad E_{k+1} \leq \max_{1 \leq i \leq n} (1 + \lambda_i q_k(\lambda_i))^2 E_0. \quad (2.35)$$

Here $\mathbf{P}(k)$ is the vector space of polynomials of degree less than or equal to k .

Proof. In view of Proposition 2.20, the iterate $\mathbf{x}^{(k+1)}$ can be written as

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(0)} + \sum_{i=0}^k \alpha_i \mathbf{A}^i \mathbf{r}^{(0)} = \mathbf{x}^{(0)} + p_k(\mathbf{A}) \mathbf{r}^{(0)},$$

where p_k is a polynomial of degree k . By Proposition 2.19, p_k is in fact the polynomial of degree k such that $f(\mathbf{x}^{(k+1)})$, and thus also E_{k+1} by (2.22), is minimized. Noting that

$$\begin{aligned} \mathbf{x}^{(k+1)} - \mathbf{x}_* &= \mathbf{x}^{(0)} - \mathbf{x}_* + p_k(\mathbf{A}) \mathbf{r}^{(0)} = \mathbf{x}^{(0)} - \mathbf{x}_* + p_k(\mathbf{A}) \mathbf{A} (\mathbf{x}^{(0)} - \mathbf{x}_*) \\ &= (\mathbf{I} + \mathbf{A} p_k(\mathbf{A})) (\mathbf{x}^{(0)} - \mathbf{x}_*), \end{aligned}$$

we deduce that

$$\forall q_k \in \mathbf{P}(k), \quad E_{k+1} \leq (\mathbf{x}^{(0)} - \mathbf{x}_*)^T \mathbf{A} (1 + \mathbf{A}q_k(\mathbf{A}))^2 (\mathbf{x}^{(0)} - \mathbf{x}_*).$$

In order to exploit this inequality, it is useful to diagonalize \mathbf{A} as $\mathbf{A} = \mathbf{Q}\mathbf{D}\mathbf{Q}^T$, for an orthogonal matrix \mathbf{Q} and a diagonal matrix \mathbf{D} . Since $q_k(\mathbf{A}) = \mathbf{Q}q_k(\mathbf{D})\mathbf{Q}^T$ for all $q_k \in \mathbf{P}(k)$, it holds that

$$\begin{aligned} \forall q_k \in \mathbf{P}(k), \quad E_{k+1} &= (\mathbf{Q}^T(\mathbf{x}^{(0)} - \mathbf{x}_*))^T \mathbf{D} (1 + \mathbf{D}q_k(\mathbf{D}))^2 (\mathbf{Q}^T(\mathbf{x}^{(0)} - \mathbf{x}_*)) \\ &\leq \left(\max_{1 \leq i \leq n} (1 + \lambda_i q_k(\lambda_i))^2 \right) \underbrace{(\mathbf{Q}^T(\mathbf{x}^{(0)} - \mathbf{x}_*))^T \mathbf{D} (\mathbf{Q}^T(\mathbf{x}^{(0)} - \mathbf{x}_*))}_{E_0}, \end{aligned}$$

which completes the proof. \square

An corollary of [Theorem 2.21](#) is that, if \mathbf{A} has $m \leq n$ distinct eigenvalues, then the conjugate gradient method converges in at most m iterations. Indeed, in this case we can take

$$q_k(\lambda) = \frac{1}{\lambda} \left(\frac{(\lambda_1 - \lambda) \dots (\lambda_m - \lambda)}{\lambda_1 \dots \lambda_m} - 1 \right).$$

It is simple to check that the right-hand side is indeed a polynomial, and that $q_k(\lambda_i) = 0$ for all eigenvalues of \mathbf{A} .

In general, finding the polynomial that minimizes the right-hand side of (2.35) is not possible, because the eigenvalues of the matrix \mathbf{A} are unknown. However, it is possible to derive from this equation an error estimate with a dependence on the condition number of $\kappa_2(\mathbf{A})$.

Theorem 2.22. *It holds that*

$$\forall k \geq 0, \quad E_k \leq 4 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^{2(k+1)} E_0,$$

Sketch of the proof. [Theorem 2.21](#) implies that

$$\forall q_k \in \mathbf{P}(k), \quad E_{k+1} \leq \max_{\lambda \in [\lambda_1, \lambda_n]} (1 + \lambda q_k(\lambda))^2 E_0,$$

where λ_1 and λ_n are the minimum and maximum eigenvalues of \mathbf{A} . We show at the end of the proof that the right-hand side is minimized when

$$1 + \lambda q_k(\lambda) = \frac{T_{k+1} \left(\frac{\lambda_n + \lambda_1 - 2\lambda}{\lambda_n - \lambda_1} \right)}{T_{k+1} \left(\frac{\lambda_n + \lambda_1}{\lambda_n - \lambda_1} \right)}, \quad (2.36)$$

where T_{k+1} is the *Chebyshev* polynomial of degree $k + 1$. These polynomials may be defined

defined from the formula

$$T_i(x) = \begin{cases} \cos(i \arccos x) & \text{for } |x| \leq 1 \\ \frac{1}{2}(x - \sqrt{x^2 - 1})^i + \frac{1}{2}(x + \sqrt{x^2 - 1})^i & \text{for } |x| \geq 1. \end{cases} \quad (2.37)$$

It is clear from this definition that $|T_i(x)| \leq 1$ for all $x \in [-1, 1]$. Consequently, the following inequality holds true for all $\lambda \in [\lambda_1, \lambda_n]$:

$$\begin{aligned} |1 + \lambda q_k(\lambda)| &\leq \frac{1}{T_{k+1}\left(\frac{\lambda_n + \lambda_1}{\lambda_n - \lambda_1}\right)} = 2 \left((r - \sqrt{r^2 - 1})^{k+1} + (r + \sqrt{r^2 - 1})^{k+1} \right)^{-1}, \\ &= 2 \left(\left(\frac{\sqrt{\kappa} + 1}{\sqrt{\kappa} - 1} \right)^{k+1} + \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^{k+1} \right)^{-1}. \end{aligned}$$

where $r = \frac{\lambda_n + \lambda_1}{\lambda_n - \lambda_1}$. Since the second term in the bracket converges to zero as $k \rightarrow \infty$, it is natural to bound this expression by keeping only the first term, which after simple algebraic manipulations leads to

$$\forall \lambda \in [\lambda_1, \lambda_n], \quad |1 + \lambda q_k(\lambda)| \leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^{k+1}.$$

From this inequality, the statement of the theorem follows immediately.

Let us now prove the optimality of (2.36). To this end, we first observe that since

$$\forall j \in \{0, \dots, k+1\}, \quad T_{k+1}\left(\cos\left(\frac{j\pi}{k+1}\right)\right) = \cos(j\pi) = (-1)^j,$$

it holds for all $\forall j \in \{0, \dots, k+1\}$ that

$$1 + \mu_j q_k(\mu_j) = T_{k+1}\left(\frac{\lambda_n + \lambda_1}{\lambda_n - \lambda_1}\right)^{-1} (-1)^j, \quad \mu_j = \frac{\lambda_n + \lambda_1 - (\lambda_n - \lambda_1) \cos(j\pi)}{2}. \quad (2.38)$$

Note that the points $(\mu_j)_{0 \leq j \leq k+1}$ are all in the interval $[\lambda_1, \lambda_n]$. Reasoning by contradiction, we assume that there is another polynomial $\tilde{q}_k \in \mathbf{P}(k)$ such that

$$\max_{\lambda \in [\lambda_1, \lambda_n]} |1 + \lambda \tilde{q}_k(\lambda)| < T_{k+1}\left(\frac{\lambda_n + \lambda_1}{\lambda_n - \lambda_1}\right)^{-1}. \quad (2.39)$$

Then $t_{k+1}(\lambda) = (1 + \lambda q_k(\lambda)) - (1 + \lambda \tilde{q}_k(\lambda))$ is a polynomial of degree $k+1$ with a zero at $\lambda = 0$. In addition, by (2.38) and (2.39) it holds that

$$\forall j \in \{0, \dots, k+1\}, \quad t_{k+1}(\mu_j) \text{ is } \begin{cases} \text{strictly positive if } j \text{ is even,} \\ \text{strictly negative if } j \text{ is odd.} \end{cases}$$

This implies, by the intermediate value theorem, that t_{k+1} has $k+1$ roots in the interval $[\lambda_1, \lambda_n]$, so $k+2$ roots in total, but this is impossible for a nonzero polynomial of degree $k+1$. \square

2.3.3 Exercises

⚙️ **Exercise 2.9.** Show that if A is row or column diagonally dominant, then A is invertible.

⚙️ **Exercise 2.10.** Let T be a nonsingular matrix. Show that

$$\|A\|_T := \|T^{-1}AT\|_2$$

defines a matrix norm induced by a vector norm.

⚙️ **Exercise 2.11.** Let $A \in \mathbf{R}^{n \times n}$ be a symmetric positive definite matrix. Show that the functional

$$\|\bullet\|_A : \mathbf{x} \mapsto \sqrt{\mathbf{x}^T A \mathbf{x}}$$

defines a norm on \mathbf{R}^n .

⚙️ **Exercise 2.12.** Show that the residual satisfies the equation

$$\mathbf{r}^{(k+1)} = N M^{-1} \mathbf{r}^{(k)} = (I - A M^{-1}) \mathbf{r}^{(k)}.$$

⚙️ **Exercise 2.13.** Show that, if A and B are two square matrices, then $\rho(AB) = \rho(BA)$.

⚙️ **Exercise 2.14.** Is $\rho(\bullet)$ a norm? Prove or disprove.

⚙️ **Exercise 2.15.** Prove that, if A is a diagonal matrix, then

$$\|A\|_1 = \|A\|_2 = \|A\|_\infty = \rho(A).$$

⚙️ **Exercise 2.16.** Show that, for any matrix norm $\|\bullet\|$ induced by a vector norm,

$$\rho(A) \leq \|A\|.$$

⚙️ **Exercise 2.17.** Let $\|\bullet\|$ denote the Euclidean vector norm on \mathbf{R}^n . We define in [Appendix A](#) the induced matrix norm as

$$\|A\| = \sup\{\|A\mathbf{x}\| : \|\mathbf{x}\| \leq 1\}.$$

Show from this definition that, if A is symmetric and positive definite, then

$$\|A\| = \|A\|_* := \sup\{|\mathbf{x}^T A \mathbf{x}| : \|\mathbf{x}\| \leq 1\}.$$

Solution. By the Cauchy–Schwarz inequality and the definition of $\|A\|$, it holds that

$$\forall \mathbf{x} \in \mathbf{R}^n \text{ with } \|\mathbf{x}\| \leq 1, \quad |\mathbf{x}^T A \mathbf{x}| \leq \|\mathbf{x}\| \|A \mathbf{x}\| \leq \|\mathbf{x}\| \|A\| \|\mathbf{x}\| \leq \|A\|.$$

This shows that $\|A\|_* \leq \|A\|$. Conversely, letting B denote a matrix square root of A (see [Exercise 2.8](#)), we have

$$\begin{aligned} \forall \mathbf{x} \in \mathbf{R}^n \text{ with } \|\mathbf{x}\| \leq 1, \quad \|A \mathbf{x}\| &= \sqrt{\mathbf{x}^T A^T A \mathbf{x}} = \sqrt{(\mathbf{B} \mathbf{x})^T \mathbf{B} \mathbf{B} (\mathbf{B} \mathbf{x})} = \sqrt{(\mathbf{B} \mathbf{x})^T A (\mathbf{B} \mathbf{x})} \\ &= \|\mathbf{B} \mathbf{x}\| \sqrt{\mathbf{y}^T A \mathbf{y}}, \quad \mathbf{y} = \frac{\mathbf{B} \mathbf{x}}{\|\mathbf{B} \mathbf{x}\|}. \end{aligned}$$

It holds that $\|\mathbf{B}\mathbf{x}\| = \sqrt{\mathbf{x}^T \mathbf{A} \mathbf{x}} \leq \sqrt{\|\mathbf{A}\|_*}$. In addition $\|\mathbf{y}\| = 1$, so the expression inside the square root is bounded from above by $\|\mathbf{A}\|_*$, which enables to conclude the proof.

□ **Exercise 2.18.** Implement an iterative method based on a splitting for finding a solution to the following linear system on \mathbf{R}^n .

$$\frac{1}{h^2} \begin{pmatrix} 2 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & -1 & 2 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 2 & -1 \\ & & & & -1 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-1} \\ x_n \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \\ 1 \end{pmatrix}, \quad h = \frac{1}{n}.$$

Plot the norm of the residual as a function of the iteration index. Use as stopping criterion the condition

$$\|\mathbf{r}^{(k)}\| \leq \varepsilon \|\mathbf{b}\|, \quad \varepsilon = 10^{-8}.$$

As initial guess, use a vector of zeros. The code will be tested with $n = 5000$.

Extra credit: Find a formula for the optimal value of ω in the relaxation method given n . The proof of Proposition 2.15, as well as the formula (2.20) for the eigenvalues of a tridiagonal matrix, are useful to this end.

Solution. Corollary 2.13 and Proposition 2.14 imply that a sufficient and necessary condition for convergence, when \mathbf{A} is Hermitian and positive definite, is that $\omega \in (0, 2)$. Let $M_\omega = \frac{1}{\omega} \mathbf{D} + \mathbf{L}$ and $N_\omega = \frac{1-\omega}{\omega} \mathbf{D} - \mathbf{U}$. A nonzero scalar $\lambda \in \mathbf{C}$ is an eigenvalue of $M_\omega^{-1} N_\omega$ if and only if

$$\det(M_\omega^{-1} N_\omega - \lambda \mathbf{I}) = 0 \quad \Leftrightarrow \quad \det(M_\omega^{-1}) \det(N_\omega - \lambda M_\omega) = 0 \quad \Leftrightarrow \quad \det(\lambda M_\omega - N_\omega) = 0.$$

Substituting the expressions of M_ω and N_ω , we obtain that this condition can be equivalently rewritten as

$$\det \left(\lambda \mathbf{L} + \left(\frac{\lambda + \omega - 1}{\omega} \right) \mathbf{D} + \mathbf{U} \right) = 0 \quad \Leftrightarrow \quad \det \left(\sqrt{\lambda} \mathbf{L} + \left(\frac{\lambda + \omega - 1}{\omega} \right) \mathbf{D} + \sqrt{\lambda} \mathbf{U} \right) = 0$$

where we used (2.19) for the last equivalence. The equality of the determinants in these two equations is valid for $\sqrt{\lambda}$ denoting either of the two complex square roots of λ . This condition is equivalent to

$$\det \left(\mathbf{L} + \left(\frac{\lambda + \omega - 1}{\sqrt{\lambda} \omega} \right) \mathbf{D} + \mathbf{U} \right) = 0.$$

We recognize from the proof of Proposition 2.15 that this condition is equivalent to

$$\frac{\lambda + \omega - 1}{\sqrt{\lambda} \omega} \in \text{spectrum}(M_{\mathcal{J}}^{-1} N_{\mathcal{J}}).$$

In other words, for any $(\lambda, \mu) \in \mathbf{C}^2$ such that

$$\frac{(\lambda + \omega - 1)^2}{\lambda \omega^2} = \mu^2, \tag{2.40}$$

it holds that $\mu \in \text{spectrum}(\mathbf{M}_{\mathcal{J}}^{-1}\mathbf{N}_{\mathcal{J}})$ if and only if $\lambda \in \text{spectrum}(\mathbf{M}_{\omega}^{-1}\mathbf{N}_{\omega})$. By (2.20), the eigenvalues of $\mathbf{M}_{\mathcal{J}}^{-1}\mathbf{N}_{\mathcal{J}}$ are real and given by

$$\mu_j = \cos\left(\frac{j\pi}{n+1}\right), \quad 1 \leq j \leq n. \quad (2.41)$$

Rearranging (2.40), we find

$$\lambda^2 + \lambda(2(\omega - 1) - \omega^2\mu^2) + (\omega - 1)^2 = 0.$$

For given $\omega \in (0, 2)$ and $\mu \in \mathbf{R}$, this is a quadratic equation for λ with solutions

$$\lambda_{\pm} = \left(\frac{\omega^2\mu^2}{2} + 1 - \omega\right) \pm \omega\mu\sqrt{\frac{\omega^2\mu^2}{4} + 1 - \omega},$$

Since the first bracket is positive when the argument of the square root is positive, it is clear that

$$\max\{|\lambda_-|, |\lambda_+|\} = \left|\frac{\omega^2\mu^2}{2} + 1 - \omega + \omega|\mu|\sqrt{\frac{\omega^2\mu^2}{4} + 1 - \omega}\right|.$$

Combining this with (2.41), we deduce that the spectral radius of $\mathbf{M}_{\omega}^{-1}\mathbf{N}_{\omega}$ is given by

$$\rho(\mathbf{M}_{\omega}^{-1}\mathbf{N}_{\omega}) = \max_{j \in \{1, \dots, n\}} \left|\frac{\omega^2\mu_j^2}{2} + 1 - \omega + \omega|\mu_j|\sqrt{\frac{\omega^2\mu_j^2}{4} + 1 - \omega}\right|. \quad (2.42)$$

We wish to minimize this expression over the interval $\omega \in (0, 2)$. While this can be achieved by algebraic manipulations, we content ourselves here with graphical exploration. Figure 2.3 depicts the amplitude of the modulus in (2.42) for different values of μ . It is apparent that, for given ω , the modulus increases as μ increases, which suggests that

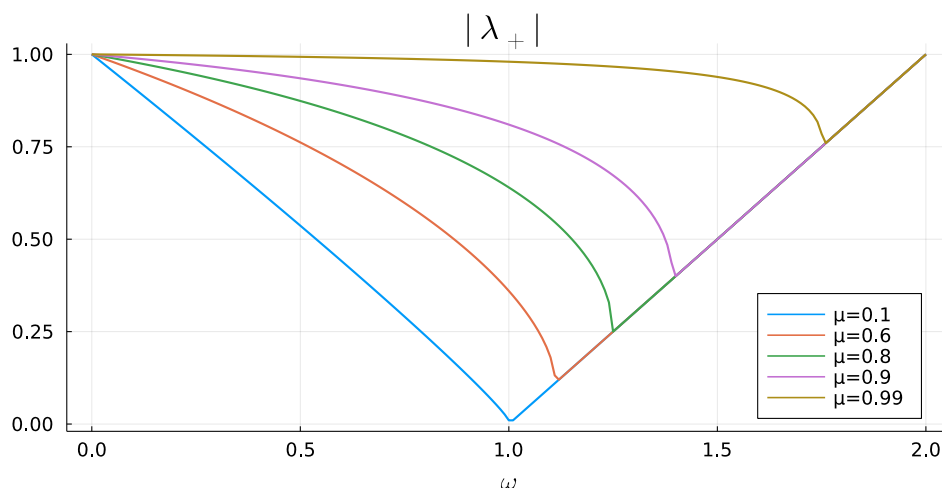
$$\rho(\mathbf{M}_{\omega}^{-1}\mathbf{N}_{\omega}) = \left|\frac{\omega^2\mu_*^2}{2} + 1 - \omega + \omega|\mu_*|\sqrt{\frac{\omega^2\mu_*^2}{4} + 1 - \omega}\right|, \quad \mu_* = \rho(\mathbf{M}_{\mathcal{J}}^{-1}\mathbf{N}_{\mathcal{J}}). \quad (2.43)$$

The figure also suggests that for a given value of μ , the modulus is minimized at the discontinuity of the first derivative, which occurs when the argument of the square root is zero. We conclude that the optimal ω satisfies

$$\frac{\omega_{\text{opt}}^2\mu_*^2}{4} + 1 - \omega_{\text{opt}} = 0 \quad \xrightarrow{\omega < 2} \quad \omega_{\text{opt}} = 2\frac{1 - \sqrt{1 - \mu_*^2}}{\mu_*^2} = \frac{2}{1 + \sqrt{1 - \mu_*^2}} = \frac{2}{1 + \sin\left(\frac{\pi}{n+1}\right)}.$$

⚙️ **Exercise 2.19.** Prove that, if the matrix \mathbf{A} is strictly diagonally dominant (by rows or columns), then the Gauss–Seidel method converges, i.e. $\rho(\mathbf{M}^{-1}\mathbf{N}) < 1$. You can use the same approach as in the proof of Proposition 2.11.

⚙️ **Exercise 2.20.** Let $\mathbf{A} \in \mathbf{R}^{n \times n}$ denote a symmetric positive definite matrix, and assume that the vectors $\mathbf{d}_1, \dots, \mathbf{d}_n$ are pairwise \mathbf{A} -orthogonal directions. Show that $\mathbf{d}_1, \dots, \mathbf{d}_n$ are linearly independent.

Figure 2.3: Modulus of $|\lambda_+|$ as a function of ω , for different eigenvalues of μ .

□ **Exercise 2.21** (Steepest descent algorithm). Consider the linear system

$$\mathbf{A}\mathbf{x} := \begin{pmatrix} 3 & 1 \\ 1 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix} =: \mathbf{b}. \quad (2.44)$$

- Show that \mathbf{A} is positive definite.
- Draw the contour lines of the function

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{A}\mathbf{x} - \mathbf{b}^T\mathbf{x}.$$

- Plot the contour lines of f in Julia using the function `contourf` from the package `Plots`.
- Using [Theorem 2.17](#), estimate the number K of iterations of the steepest descent algorithm required in order to guarantee that $E_K \leq 10^{-8}$, when starting from the vector $\mathbf{x}^{(0)} = (2 \ 3)^T$.
- Implement the steepest descent method for finding the solution to (2.44), and plot the iterates as linked dots over the filled contour of f .
- Plot the error E_k as a function of the iteration index, using a linear scale for the x axis and a logarithmic scale for the y axis.

⚙️ **Exercise 2.22.** Compute the number of floating point operations required for performing one iteration of the conjugate gradient method, assuming that the matrix \mathbf{A} contains $\alpha \ll n$ nonzero elements per row.

□ **Exercise 2.23** (Solving the Poisson equation over a rectangle). We consider in this exercise Poisson's equation in the domain $\Omega = (0, 2) \times (0, 1)$, equipped with homogeneous Dirichlet boundary conditions:

$$\begin{aligned} -\Delta f(x, y) &= b(x, y), & x \in \Omega, \\ f(x) &= 0, & x \in \partial\Omega. \end{aligned}$$

The right-hand side is

$$b(x, y) = \sin(4\pi x) + \sin(2\pi y).$$

A number of methods can be employed in order to discretize this partial differential equation. After discretization, a finite-dimensional linear system of the form $\mathbf{Ax} = \mathbf{b}$ is obtained. A Julia function for calculating the matrix \mathbf{A} and the vector \mathbf{b} using the finite difference method is given to you on the course website, as well as a function to plot the solution. The goal of this exercise is to solve the linear system using the conjugate gradient method. Use the same stopping criterion as in [Exercise 2.18](#).

⚙️ **Exercise 2.24.** Show that (2.37) indeed defines a polynomial, and find its expression in the usual polynomial notation.

⚙️ **Exercise 2.25.** Show that if $\mathbf{A} \in \mathbf{R}^{n \times n}$ is nonsingular, then the solution to the equation $\mathbf{Ax} = \mathbf{b}$ belongs to the Krylov subspace

$$\mathcal{K}_n(\mathbf{A}, \mathbf{b}) = \text{span}\{\mathbf{b}, \mathbf{Ab}, \mathbf{A}^2\mathbf{b}, \dots, \mathbf{A}^{n-1}\mathbf{b}\}.$$

2.4 Discussion and bibliography

In this chapter, we presented direct methods and some of the standard iterative methods for solving linear systems. We focused particularly on linear systems with a symmetric positive definite matrix. [Section 2.2](#) is based on [9, 15] and [Section 2.3](#) roughly follows [13, Chapter 2]. The book [10] is a very detailed reference on iterative methods for solving sparse linear systems. The reference [12] is an excellent introduction to the conjugate gradient method.