Chapter 2

Direct linear solvers

2.1 Introduction

A linear system is a collection of m linear equations involving n unknowns, with $n, m \in \mathbb{N}$, taking the form

$$\sum_{k=1}^{n} a_{j,k}\, x_k = b_j \quad \forall j = 1 \dots m. \tag{2.1}$$

In these equations $\boldsymbol{x} = (x_k)_{k=1}^n$ is the unknown vector, the $a_{j,k}$ are the entries (or coefficients) of the matrix of the linear system, and $\boldsymbol{b} = (b_j)_{j=1}^m$ is the right hand side. In what follows the matrix of the linear system will be denoted $\mathrm{A} = (a_{j,k})$, so that this linear system rewrites as

$$\mathrm{A} \cdot \boldsymbol{x} = \boldsymbol{b}. \tag{2.2}$$

In this course we shall only consider invertible linear systems, which corresponds to the situations where the matrix A admits an inverse denoted A^{-1}. According to the rank-nullity theorem, a consequence of this assumption is that linear systems are square $m = n$ so that the matrix A shall admit as many rows as columns.

Effective solution methods for problems of the form (2.1) are called linear solvers. There exists essentially two families of linear solvers

- **Direct solvers** where the solution $\boldsymbol{x}$ is computed in a finite (but potentially large) number of operations.
- **Iterative solvers** where the linear system is reformulated as a fixed point problem $\boldsymbol{x} = \Phi(\boldsymbol{x})$. The solution $\boldsymbol{x}$ is then approximated by means of an algorithm of the form $\boldsymbol{x}^{(p+1)} = \Phi(\boldsymbol{x}^{(p)})$.

In this chapter we will focus on direct solvers. Iterative solvers are the subject of subsequent chapters of this course.

2.2 Triangular systems

Firstly let us consider the case of a lower triangular linear system, that is the case where $a_{j,k} = 0$ for $j < k$. In this case the matrix A admits the form

$$\mathrm{A} = \begin{bmatrix} a_{1,1} & 0 & \cdots & 0 \\ a_{2,1} & a_{2,2} & \ddots & \vdots \\ \vdots & * & \ddots & 0 \\ a_{n,1} & a_{n,2} & \cdots & a_{n,n} \end{bmatrix}$$

The algorithm to solve the problem $\mathrm{A}\boldsymbol{x} = \boldsymbol{b}$, forward substitution method, is quite natural. One first observes that the j-th row (j-th equation) can be re-written

$$x_j = \frac{1}{a_{j,j}} \big(b_j - \sum_{k=1}^{j-1} a_{j,k} x_k \big) \tag{2.3}$$

From this identity, the value of x_j is deduced from the values of $x_{j-1}, x_{j-2}, \dots x_1$. One thus starts wit the first row that writes $x_1 = b_1/a_{1,1}$, then $x_2, x_3, \dots x_n$ are successively obtained by means of the reccurence relations (2.3).

Case of an upper triangular system In the case of an upper triangular system, that is when $a_{j,k} = 0$ for $j > k$, the solution principle is the same. This time the method is called backward substitution method. The solution procedure starts from the last row, then the following reccursion formula is applied

$$x_j = \frac{1}{a_{j,j}} \big(b_j - \sum_{k=j+1}^{n} a_{j,k} x_k \big) \tag{2.4}$$

Algorithmic complexity Let us examine the number of operations required for the forward substitution method with respect to n, when solving a lower triangular system. If we do not make any particular assumption on the matrix A, the solution procedure that we have just discussed requires

- n divisions,
- $n - 1$ subtrations,
- $\sum_{i=2}^{n} \sum_{j=1}^{i-1} 1 = \sum_{i=2}^{n} (i-1) = n(n-1)/2$ multiplications,
- $\sum_{i=3}^{n} \sum_{j=1}^{i-2} 1 = \sum_{i=2}^{n} (i-2) = (n-1)(n-2)/2$ additions.

This leads to a total cost of $n + (n-1) + n(n-1)/2 + (n-1)(n-2)/2 = n^2$ elementary operations ("floating point operation" = flop). The algorithm that we have just described is thus said to admit an algorithmic complexity of $\mathcal{O}(n^2)$.

We we assume that the matrix A admits only $\mathcal{O}(1)$ nonzero term in each row (for example it admits band structure with a band of fixed width), then the solution to this triangular system only costs $\mathcal{O}(n)$, which is the cost of a matrix-vector product.

2.3 Gaussian elimination

We come back to the case of an arbitrary linear system. The gaussian elimination method consists in reducing the initial linear system to an equivalent upper triangular system (obviously, with right hand side modified accordingly). In the sequel $\boldsymbol{e}_j, j = 1 \dots n$ shall refer to the canonical basis of $\mathbb{C}^{n\times n}$.

We know that $\mathrm{A}\boldsymbol{x} = \boldsymbol{b} \iff \mathcal{L}^{(1)}\mathrm{A}\boldsymbol{x} = \mathcal{L}^{(1)}\boldsymbol{b}$ if the matrix $\mathcal{L}^{(1)} \in \mathbb{C}^{n\times n}$ is invertible. The matrix of the transformed linear system takes the form $\mathrm{A}^{(1)} := \mathcal{L}^{(1)}\mathrm{A} = (a^{(1)}_{j,k})_{j,k=1\dots n} \in \mathbb{C}^{n\times n}$, and $\boldsymbol{b}^{(1)} := \mathcal{L}^{(1)}\boldsymbol{b}$. Let us choose $\mathcal{L}^{(1)}$ so as to make sure that $a^{(1)}_{j,1} = 0$ for $j = 2, \dots n$ i.e. so that $\mathrm{A}^{(1)}$ takes the (block lower triangular) form

$$\mathrm{A}^{(1)} = \begin{bmatrix} a^{(1)}_{1,1} & * & \cdots & * \\ 0 & \vdots & & \vdots \\ \vdots & \vdots & & \vdots \\ 0 & * & \cdots & * \end{bmatrix}$$

It suffices to choose

$$\mathcal{L}^{(1)} := \mathrm{Id} - \boldsymbol{\ell}_1 \cdot \boldsymbol{e}_1^\top$$
$$\text{where} \quad \boldsymbol{\ell}_1^\top := (0, a_{2,1}/a_{1,1}, \dots, a_{n,1}/a_{1,1}).$$

Note that $\boldsymbol{\ell}_1 \cdot \boldsymbol{e}_1^\top$ is indeed a rank 1 matrix of size $n \times n$ not to be confused with $\boldsymbol{e}_1^\top \cdot \boldsymbol{\ell}_1$ (which is a simple scalar value). Besides we have $\boldsymbol{e}_1^\top \cdot \boldsymbol{\ell}_1 = 0$. Let us verify that $\mathcal{L}^{(1)}$ is invertible. We have $(\mathrm{Id} - \boldsymbol{\ell}_1 \cdot \boldsymbol{e}_1^\top) \cdot (\mathrm{Id} + \boldsymbol{\ell}_1 \cdot \boldsymbol{e}_1^\top) = \mathrm{Id} - (\boldsymbol{\ell}_1 \cdot \boldsymbol{e}_1^\top)^2 = \mathrm{Id} - \boldsymbol{\ell}_1 \cdot (\boldsymbol{e}_1^\top \cdot \boldsymbol{\ell}_1) \cdot \boldsymbol{e}_1^\top = \mathrm{Id}$. We have just proved that

$$(\mathcal{L}^{(1)})^{-1} = \mathrm{Id} + \boldsymbol{\ell}_1 \cdot \boldsymbol{e}_1^\top.$$

Note that $\boldsymbol{e}_1^\top \cdot \mathrm{A}^{(1)} = \boldsymbol{e}_1^\top \cdot \mathrm{A}$. To conclude, note the particular role plaied by the coefficient $a_{1,1}$ in this procedure. This coefficient is called first *pivot*. For the matrix $\mathcal{L}^{(1)}$ to be properly defined, this pivot needs to be non-zero.

Second iteration We can reiterate the procedure that we have just described. Indeed we have $\mathrm{A}^{(1)}\boldsymbol{x} = \boldsymbol{b}^{(1)} \iff \mathcal{L}^{(2)}\mathrm{A}^{(1)}\boldsymbol{x} = \mathcal{L}^{(2)}\boldsymbol{b}^{(1)}$ provided that $\mathcal{L}^{(2)}$ is invertible. Set $\mathrm{A}^{(2)} = \mathcal{L}^{(2)}\mathrm{A}^{(1)} = (\mathcal{L}^{(2)} \cdot \mathcal{L}^{(1)}) \cdot \mathrm{A} = (a^{(2)}_{j,k})_{j,k=1\dots n}$ and $\boldsymbol{b}^{(2)} = \mathcal{L}^{(2)}\boldsymbol{b}^{(1)}$. We will choose $\mathcal{L}^{(2)}$ so as to make sure that $\mathrm{A}^{(2)}$ takes the form

$$\mathrm{A}^{(2)} = \begin{bmatrix} \mathrm{U}_2 & * \\ 0 & * \end{bmatrix} \quad \text{où} \quad \mathrm{U}_2 = \begin{bmatrix} a^{(2)}_{1,1} & * \\ 0 & a^{(2)}_{2,2} \end{bmatrix}$$

Here the matrix U_2 is of size 2×2. To obtain the form above, it suffices then to define $\mathcal{L}^{(2)}$ as a lower triangular matrix given by

$$\mathcal{L}^{(2)} := \mathrm{Id} - \boldsymbol{\ell}_2 \cdot \boldsymbol{e}_2^\top$$
$$\text{avec} \quad \boldsymbol{\ell}_2^\top = (0, 0, a^{(1)}_{3,2}/a^{(1)}_{2,2}, \dots, a^{(1)}_{n,2}/a^{(1)}_{2,2}).$$

Again the matrix $\mathcal{L}^{(2)}$ is properly defined provided that the second pivot verifies $a^{(1)}_{2,2} \neq 0$ and, in this case, $\mathcal{L}^{(2)}$ is invertible with inverse given by $(\mathcal{L}^{(2)})^{-1} = \mathrm{Id} + \boldsymbol{\ell}_2 \cdot \boldsymbol{e}_2^\top$. Then we have $\boldsymbol{e}_1^\top \mathrm{A}^{(2)} = \boldsymbol{e}_1^\top \mathrm{A}^{(1)} = \boldsymbol{e}_1^\top \mathrm{A}$ (in particular $a^{(2)}_{1,1} = a^{(1)}_{1,1} = a_{1,1}$), and $\boldsymbol{e}_2^\top \mathrm{A}^{(2)} = \boldsymbol{e}_2^\top \mathrm{A}^{(1)}$.

General iteration We can generalize the procedure above to up any order. Assume that $\mathrm{A}^{(p-1)}$ has been defined for $p = 2 \dots n$, and that it is upper triangular on its $p-1$ first columns i.e. it admits the following form

$$\mathrm{A}^{(p-1)} = \begin{bmatrix} \mathrm{U}_{p-1} & * \\ 0 & * \end{bmatrix}$$

where U_{p-1} is an upper triangular matrix of size $(p-1) \times (p-1)$. We then define $\mathrm{A}^{(p)} = \mathcal{L}^{(p)} \cdot \mathrm{A}^{(p-1)} = (a^{(p)}_{j,k})_{j,k=1\dots n}$ and $\boldsymbol{b}^{(p)} = \mathcal{L}^{(p)} \boldsymbol{b}^{(p-1)}$, where $\mathcal{L}^{(p)} \in \mathbb{C}^{n \times n}$ is a lower triangular matrix given by the formula

$$\begin{aligned} &\mathcal{L}^{(p)} := \mathrm{Id} - \boldsymbol{\ell}_p \cdot \boldsymbol{e}_p^\top \\ &\text{avec} \quad \boldsymbol{\ell}_p^\top = (0, \dots, 0, a^{(p-1)}_{p+1,p}/a^{(p-1)}_{p,p}, \dots, a^{(p-1)}_{n,p}/a^{(p-1)}_{p,p}). \end{aligned} \tag{2.5}$$

A matrix of the form (2.5) is called a Gauss transformation. In the definition of $\boldsymbol{\ell}_p$, the first p coefficients are zero, which writes $\boldsymbol{e}_j^\top \cdot \ell_p = 0 \forall j = 1 \dots p$. This implies $\boldsymbol{e}_j^\top \mathcal{L}^{(p)} = \boldsymbol{e}_j^\top, \forall j = 1 \dots p$. This construction of $\mathrm{A}^{(p)}$ hence guarantees that $\boldsymbol{e}_j^\top \mathrm{A}^{(p)} = \boldsymbol{e}_j^\top \mathcal{L}^{(p)} \mathrm{A}^{(p-1)} = \boldsymbol{e}_j^\top \mathrm{A}^{(p-1)}$ for all $j = 1 \dots p$ and, reccursively, we deduce that

$$\boldsymbol{e}_j^\top \mathrm{A}^{(p)} = \boldsymbol{e}_j^\top \mathrm{A}^{(j-1)} \quad \forall j = 1 \dots p$$

setting $\mathrm{A}^{(0)} := \mathrm{A}$. Here again, the matrix $\mathcal{L}^{(p)}$ is properly defined only if the p-th pivot satisfies $a^{(p-1)}_{p,p} \neq 0$, and its inverse is given by $(\mathcal{L}^{(p)})^{-1} = \mathrm{Id} + \boldsymbol{\ell}_p \cdot \boldsymbol{e}_p^\top$. To conclude, the construction above garantees that $\mathrm{A}^{(p)}$ admits the form

$$\mathrm{A}^{(p)} = \begin{bmatrix} \mathrm{U}_p & \star \\ 0 & \star \end{bmatrix}$$

where U_p is an upper triangular matrix of size $p \times p$. From what precedes, we see that the first p rows and columns remain unchanged when transforming the system from $\mathrm{A}^{(p-1)}$ to $\mathrm{A}^{(p)}$. We obtain the following formulas expressing $a^{(p)}_{j,k}$ with respect to $a^{(p-1)}_{j,k}$,

$$\begin{cases} a^{(p)}_{j,k} = a^{(p-1)}_{j,k} & \text{for } 1 \leq j \leq p \text{ or } 1 \leq k < p \\ a^{(p)}_{j,k} = 0 & \text{if } p < j \leq n \text{ and } k = p \\ a^{(p)}_{j,k} = a^{(p-1)}_{j,k} - a^{(p-1)}_{j,p} a^{(p-1)}_{p,k} / a^{(p-1)}_{p,p} & \text{if } p < j \leq n \text{ and } p < k \leq n \end{cases} \tag{2.6}$$

Conclusion of the algorithm The gaussian elimination algorithm terminates whenever $p = n - 1$, since the matrix $\mathrm{A}^{(n-1)}$ is then completely upper triangular. We have obtained that $\mathrm{A}\boldsymbol{x} = \boldsymbol{b} \iff \mathrm{A}^{(n-1)}\boldsymbol{x} = \boldsymbol{b}^{(n-1)}$. The later system being upper triangular (by construction...), it can be solved by a backward substitution algorithm as explained in the previous section 2.2.

Algorithmic complexity ONe can show that the gaussian elimination algorithm requires $2n^3/3 + n^2/3 - n$ operations (this is left as an exercise).

Remarks on the pivots The gaussian elimination algorithm is well defined provided that $a_{p,p}^{(p-1)} \neq 0$ for all p i.e. no pivot equals zero. There are classes of matrices for which this condition is systematicaly fulfilled. Here are three examples:

- symetric positive definite matrices,
- row-wise diagonally dominant matrices: $|a_{j,j}| > \sum_{k\neq j} |a_{j,k}| \; \forall j = 1 \dots n$,
- column-wise diagonally dominant matrices: $|a_{j,j}| > \sum_{k\neq j} |a_{k,j}| \; \forall j = 1 \dots n$.

2.4 LU decomposition

The gaussian elimination method leads to a factorization of the matrix under the form $\mathrm{A} = \mathrm{L}\cdot\mathrm{U}$ where L is lower triangular with $\mathrm{L}_{j,j} = 1, j = 1 \dots n$ and U is upper triangular. Indeed, let us start by setting $\mathrm{U} = \mathrm{A}^{(n-1)}$ which is the matrix obtained after the $n-1$-th step of the Gauss method. We thus have $\mathcal{L}^{(n-1)}\mathcal{L}^{(n-2)} \cdots \mathcal{L}^{(1)}\mathrm{A} = \mathrm{U} \iff \mathrm{A} = \mathrm{L}\cdot\mathrm{U}$ with

$$\mathrm{L} := (\mathcal{L}^{(1)})^{-1}(\mathcal{L}^{(2)})^{-1} \cdots (\mathcal{L}^{(n-1)})^{-1}$$

Let us study in more details the matrix L. It appears in factorized form. We will develop this product, taking account of the elementary property $\boldsymbol{e}_j^\top \boldsymbol{\ell}_p = 0$ for $j = 1 \dots p$. As a particular case of this property, we see that $\boldsymbol{e}_1^\top \boldsymbol{\ell}_2 = 0$, which leads to the conclusion that

$$\begin{aligned}(\mathcal{L}^{(1)})^{-1}(\mathcal{L}^{(2)})^{-1} &= (\mathrm{Id} + \boldsymbol{\ell}_1\boldsymbol{e}_1^\top)(\mathrm{Id} + \boldsymbol{\ell}_2\boldsymbol{e}_2^\top) \\ &= \mathrm{Id} + \boldsymbol{\ell}_1\boldsymbol{e}_1^\top + \boldsymbol{\ell}_2\boldsymbol{e}_2^\top + \boldsymbol{\ell}_1(\boldsymbol{e}_1^\top\boldsymbol{\ell}_2)\boldsymbol{e}_2^\top \\ &= \mathrm{Id} + \boldsymbol{\ell}_1\boldsymbol{e}_1^\top + \boldsymbol{\ell}_2\boldsymbol{e}_2^\top\end{aligned}$$

We then proceed by recurrence on p to show that $(\mathcal{L}^{(1)})^{-1}(\mathcal{L}^{(2)})^{-1} \cdots (\mathcal{L}^{p})^{-1} = \mathrm{Id} + \sum_{j=1}^{p} \boldsymbol{\ell}_j\boldsymbol{e}_j^\top$. We have just proved that this property holds for $p = 1$. Assume that it holds for p, and let us prove that it holds for $p+1$ as well. We have

$$\begin{aligned}(\mathcal{L}^{(1)})^{-1}(\mathcal{L}^{(2)})^{-1} \cdots (\mathcal{L}^{(p)})^{-1}(\mathcal{L}^{(p+1)})^{-1} &= (\mathrm{Id} + \sum_{j=1}^{p} \boldsymbol{\ell}_j\boldsymbol{e}_j^\top)(\mathrm{Id} + \boldsymbol{\ell}_{p+1}\boldsymbol{e}_{p+1}^\top) \\ &= \mathrm{Id} + (\sum_{j=1}^{p} \boldsymbol{\ell}_j\boldsymbol{e}_j^\top) + \boldsymbol{\ell}_{p+1}\boldsymbol{e}_{p+1}^\top + (\sum_{j=1}^{p} \boldsymbol{\ell}_j\boldsymbol{e}_j^\top)\boldsymbol{\ell}_{p+1}\boldsymbol{e}_{p+1}^\top \\ &= \mathrm{Id} + \sum_{j=1}^{p+1} \boldsymbol{\ell}_j\boldsymbol{e}_j^\top + (\sum_{j=1}^{p} \boldsymbol{\ell}_j\boldsymbol{e}_j^\top)\boldsymbol{\ell}_{p+1}\boldsymbol{e}_{p+1}^\top \\ &= \mathrm{Id} + \sum_{j=1}^{p+1} \boldsymbol{\ell}_j\boldsymbol{e}_j^\top + \sum_{j=1}^{p} \boldsymbol{\ell}_j(\boldsymbol{e}_j^\top\boldsymbol{\ell}_{p+1})\boldsymbol{e}_{p+1}^\top\end{aligned}$$

Since $\boldsymbol{e}_j^\top \boldsymbol{\ell}_{p+1} = 0$ for $j = 1 \dots p$, the last term in the right hand side above is zero, so that the property that we want to prove holds for $p+1$. We finally obtain, for $p = n-1$,

$$\mathrm{L} = \mathrm{Id} + \sum_{j=1}^{n-1} \boldsymbol{\ell}_j\boldsymbol{e}_j^\top$$

Let us examine the j-th column of this matrix. We have $\mathrm{L} \cdot \boldsymbol{e}_j = \boldsymbol{e}_j + \boldsymbol{\ell}_j$, with $(\boldsymbol{e}_j + \boldsymbol{\ell}_j)^\top = (0, \dots, 0, 1, a_{j+1,j}^{(j-1)}/a_{j,j}^{(j-1)}, \dots, a_{n,j}^{(j-1)}/a_{j,j}^{(j-1)})$. We see that L is indeed upper triangular, with 1's on the dioagonal, and the expression that we have just obtained for it scolumns suggests that this matrix should be assembled on-the-fly during the gaussian elimination algorithm.

2.5 Factorization algorithm

Let us now examine the algorithmic details of an LU decomposition. Before going into the description of the algorithm itself, we introduce a few simple notations.If J and K are two subsets of $[\![1, n]\!] := \{1, \dots n\}$, we will denote $\mathrm{A}_{\mathrm{J,K}} \in \mathbb{C}^{|\mathrm{J}| \times |\mathrm{K}|}$ the submatrix obtained out of A by extracting the columns $k \in \mathrm{K}$ and the rows $j \in \mathrm{J}$ (here we denote $|\mathrm{J}|$ the cardinal of J, and accordingly for $|\mathrm{K}|$). For a $j \in [\![1, n]\!]$ and a subset $\mathrm{K} \subset [\![1, n]\!]$, we denote $\mathrm{A}_{j,\mathrm{K}}$ instead of $\mathrm{A}_{\{j\},\mathrm{K}}$ and, similarly, for $k \in [\![1, n]\!]$ and $\mathrm{J} \subset [\![1, n]\!]$ we will denote $\mathrm{A}_{\mathrm{J},k}$ instead of $\mathrm{A}_{\mathrm{J},\{k\}}$.

As suggested by the algorithm of the gaussian elimination described above, an LU decomposition method involves $n - 1$ for a square matrix $\mathrm{A} \in \mathbb{C}^{n \times n}$. Iterationb p consists in an update of the matrix $\mathrm{A}^{(p-1)}$ to obtain the matrix $\mathrm{A}^{(p)}$. On the theoretical side, this update takes the form of a left-multiplication by the matrix $\mathcal{L}^{(p)} := \mathrm{Id} - \ell_p \cdot \boldsymbol{e}_p^\top$, but this is not the actual way this update takes place in practice: this would be unecessarily costly. Here are a few simple observations concerning this update:

- it does not modify the rows $1, \dots, p$
- it does not modify the columns $1, \dots, p-1$
- in column p, it cancels out the elements $a_{j,p}^{(p-1)}$ for $j = p+1, \dots n$

As a consequence, to change $\mathrm{A}^{(p-1)}$ into $\mathrm{A}^{(p)}$, it suffices to modify the block $\mathrm{A}_{\mathrm{J,J}}^{(p-1)}$ for $\mathrm{J} = [\![p+1, n]\!]$. Besides, the matrix $\mathcal{L}^{(p)}$ is entirely determined by ℓ_p. These elementary remarks together with §2.3, lead to Algorithm 1 that follows

Algorithm 1

function LU_NAIF(A)
  L = Id
  $\mathrm{A}^{(0)} = \mathrm{A}$
  for $p = 1 \dots n-1$ **do**
    $\mathrm{J} = [\![p+1, n]\!]$
    $\mathrm{L}_{\mathrm{J},p} = \mathrm{A}_{\mathrm{J},p}^{(p-1)} / \mathrm{A}_{p,p}^{(p-1)}$
    $\mathrm{A}_{\mathrm{J,J}}^{(p)} = \mathrm{A}_{\mathrm{J,J}}^{(p-1)} - \mathrm{L}_{\mathrm{J},p} \mathrm{A}_{p,\mathrm{J}}^{(p-1)}$
  end for
  $\mathrm{U} = \mathrm{A}^{(n-1)}$
  return (L, U)
end function

Algorithm 2

function LU_DECOMPOSE(A)
  T = A
  for $p = 1 \dots n-1$ **do**
    $\mathrm{J} = [\![p+1, n]\!]$
    $\mathrm{T}_{\mathrm{J},p} = \mathrm{T}_{\mathrm{J},p} / \mathrm{T}_{p,p}$
    $\mathrm{T}_{\mathrm{J,J}} = \mathrm{T}_{\mathrm{J,J}} - \mathrm{T}_{\mathrm{J},p} \mathrm{T}_{p,\mathrm{J}}$
  end for
  return (T)
end function

In fact Algorithm 1 makes use of many unnecessary intermediate variables. One may store L (resp. U) in the lower (resp. upper) triangular part of a single matrix $\mathrm{T} \in \mathbb{C}^{n\times n}$. During this construction, it is also possible to store the matrices $\mathrm{A}^{(p)}$ inside T. This leads to Algorithm 2 that only involves a single additional matrix.

Once the LU decomposition of the matrix A is available et stored in the matrix T, we can solve the linear system $\mathrm{A}\boldsymbol{x} = \boldsymbol{b}$ by means of the successive application of a backward and a forward substitution method based on the upper and lower triangular parts of T (the diagonal coefficients of the lower triangular system must equal 1 hence do not need to be stored). The effective solution of a linear system making use of the a priori knowledge of the LU decomposition then takes the form of Algorithm 3 below.

Algorithm 3

function LU_SOLVE(A,$\boldsymbol{b}$)
  T = LU_DECOMPOSE(A)
  //descente
  for $j = 1 \dots n$ **do**
    $\boldsymbol{v}_j = \boldsymbol{b}_j$
    for $k = 1 \dots j-1$ **do**
      $\boldsymbol{v}_j = \boldsymbol{v}_j - \mathrm{T}_{j,k}\boldsymbol{v}_k$
    end for
  end for
  //remontee
  for $p = 1 \dots n$ **do**
    $j = n - p + 1$
    $\boldsymbol{u}_j = \boldsymbol{v}_j$
    for $k = j+1 \dots n$ **do**
      $\boldsymbol{u}_j = \boldsymbol{u}_j - \mathrm{T}_{j,k}\boldsymbol{u}_k$
    end for
    $\boldsymbol{u}_j = \boldsymbol{u}_j / \mathrm{T}_{j,j}$
  end for
  return $(\boldsymbol{u})$
end function

Algorithm 4

function LU_PIVOT(A)
  T = A, P = Id
  for $q = 1 \dots n-1$ **do**
    //pivoting
    $r = \mathrm{argmax}_{j=q,\dots n} |\mathrm{T}_{j,q}|$
    $\mathrm{P} = \tau(q,r) \cdot \mathrm{P}$, $\mathrm{T} = \tau(q,r) \cdot \mathrm{T}$
    //gaussian elimination
    $\mathrm{J} = [\![p+1, n]\!]$
    $\mathrm{T}_{\mathrm{J},p} = \mathrm{T}_{\mathrm{J},p} / \mathrm{T}_{p,p}$
    $\mathrm{T}_{\mathrm{J},\mathrm{J}} = \mathrm{T}_{\mathrm{J},\mathrm{J}} - \mathrm{T}_{\mathrm{J},p}\mathrm{T}_{p,\mathrm{J}}$
  end for
  return (T, P)
end function

2.6 Partial pivoting

As we saw, the gaussian elimination algorithm stops if one of the pivots is zero i.e. $a_{q,q}^{(q-1)} = 0$. One way to circumvent this issue relies on a so-called partial pivoting strategy. It consists in a preliminary step taking place at each iteration q, where the rows q and r are swapped with r chosen so that

$$|a_{r,q}^{(q-1)}| = \max_{j=q\dots n} |a_{j,q}^{(q-1)}| \tag{2.7}$$

This is equivalent to a left multiplication of the matrix A^{q-1} by a permutation matrix P_q defined by: $\mathrm{P}_q \cdot \boldsymbol{e}_q = \boldsymbol{e}_r$, $\mathrm{P}_q \cdot \boldsymbol{e}_r = \boldsymbol{e}_q$ and $\mathrm{P}_q \cdot \boldsymbol{e}_j = \boldsymbol{e}_j$ if $j \neq q, r$. It is important to note

that $(\mathrm{P}_q)^2 = \mathrm{Id}$ and that $\mathrm{P}_q^\top = \mathrm{P}_q$. In addition observe that the coefficient $a_{r,q}^{(q-1)}$ in (2.7) necessarily satisfies $a_{r,q}^{(q-1)} \neq 0$ otherwise the matrix A would not be invertible.

With the pivoting strategy described above, the result of $n-1$ iterations of the gaussian elimination algorithm writes $\mathcal{L}^{(n-1)}\mathrm{P}_{n-1}\cdots\mathrm{P}_2\mathcal{L}^{(1)}\mathrm{P}_1\cdot\mathrm{A} = \mathrm{U}$. This can be rewritten equivalently as follows:

$$\begin{aligned}
& \mathrm{A} = \mathrm{P}_1(\mathcal{L}^{(1)})^{-1}\mathrm{P}_2\cdots\mathrm{P}_{n-1}(\mathcal{L}^{(n-1)})^{-1}\cdot\mathrm{U} \\
\iff\ & \mathrm{PA} = \mathrm{P}\cdot\mathrm{P}_1(\mathcal{L}^{(1)})^{-1}\mathrm{P}_2\cdots\mathrm{P}_{n-1}(\mathcal{L}^{(n-1)})^{-1}\cdot\mathrm{U} \\
& \quad \text{with} \quad \mathrm{P} = \mathrm{P}_{n-1}\mathrm{P}_{n-2}\cdots\mathrm{P}_1 \\
\iff\ & \mathrm{PA} = \mathcal{T}_1\cdot\mathcal{T}_2\cdots\mathcal{T}_{n-1}\cdot\mathrm{U} \\
& \quad \text{where} \quad \mathcal{T}_q = \mathrm{P}_{n-1}\cdots\mathrm{P}_{q+1}\cdot(\mathcal{L}^{(q)})^{-1}\cdot\mathrm{P}_{q+1}\cdots\mathrm{P}_{n-1}
\end{aligned}$$

In the calculation above, we have considered that $\mathcal{T}_{n-1} = (\mathcal{L}^{(n-1)})^{-1}$. Let us check that the matrices $\mathcal{T}_q$ are lower triangular. Recall that $(\mathcal{L}^{(q)})^{-1} = \mathrm{Id} + \boldsymbol{\ell}_q\cdot\boldsymbol{e}_q^\top$, where $\boldsymbol{\ell}_q \in \mathbb{C}^{n\times n}$, verifies $\boldsymbol{e}_j^\top\cdot\boldsymbol{\ell}_q = 0$ for $j = 1\ldots q$. As a consequence, we have

$$\begin{aligned}
\mathcal{T}_q &= \mathrm{P}_{n-1}\cdots\mathrm{P}_{q+1}\cdot(\mathrm{Id} + \boldsymbol{\ell}_q\cdot\boldsymbol{e}_q^\top)\cdot\mathrm{P}_{q+1}\cdots\mathrm{P}_{n-1} \\
&= \mathrm{Id} + (\mathrm{P}_{n-1}\cdots\mathrm{P}_{q+1}\cdot\boldsymbol{\ell}_q)\cdot(\boldsymbol{e}_q^\top\cdot\mathrm{P}_{q+1}\cdots\mathrm{P}_{n-1}) \\
&= \mathrm{Id} + (\mathrm{P}_{n-1}\cdots\mathrm{P}_{q+1}\cdot\boldsymbol{\ell}_q)\cdot(\mathrm{P}_{n-1}\cdots\mathrm{P}_{q+1}\cdot\boldsymbol{e}_q)^\top \\
&= \mathrm{Id} + \boldsymbol{\ell}'_q\cdot\boldsymbol{e}_q^\top \qquad \text{with} \quad \boldsymbol{\ell}'_q := \mathrm{P}_{n-1}\cdots\mathrm{P}_{q+1}\cdot\boldsymbol{\ell}_q
\end{aligned}$$

In the last step of the calculation above, we have used the fact that $\mathrm{P}_q\boldsymbol{e}_j = \boldsymbol{e}_j$ whenever $q > j$. Using this same property, we also see that, for all $j = 1\ldots q$, we have $\boldsymbol{e}_j^\top\cdot\boldsymbol{\ell}'_q = \boldsymbol{e}_j^\top\cdot\mathrm{P}_{n-1}\cdots\mathrm{P}_{q+1}\cdot\boldsymbol{\ell}_q = (\mathrm{P}_{q+1}\cdots\mathrm{P}_{n-1}\cdot\boldsymbol{e}_j)^\top\cdot\boldsymbol{\ell}_q = \boldsymbol{e}_j^\top\cdot\boldsymbol{\ell}_q = 0$. This proves that $\mathcal{T}_q$ is lower triangular. Setting this time $\mathrm{L} = \mathcal{T}_1\cdot\mathcal{T}_2\cdots\mathcal{T}_{n-1}$, we have obtained

$$\mathrm{P}\cdot\mathrm{A} = \mathrm{L}\cdot\mathrm{U}$$

where L is lower triangular, and U is upper triangular. Such a decomposition holds as soon as A is invertible.

A modified version of Algorithm 2 taking account of the row-wise partial pivoting is given in Algorithm 4 above. In this algorithm $\tau(q,r) \in \mathbb{C}^{n\times n}$ is the transposition matrix such that $\tau(q,r)\boldsymbol{e}_k = \boldsymbol{e}_k$ if $k \neq q, r$, $\tau(q,r)\boldsymbol{e}_q = \boldsymbol{e}_r$ and $\tau(q,r)\boldsymbol{e}_r = \boldsymbol{e}_q$. Obviously, to compute $\tau(q,r)\cdot\mathrm{P}$ and $\tau(q,r)\cdot\mathrm{T}$, no need to perform a full matrix-matrix product (a costly operation with $\mathcal{O}(n^3)$ algorithmic complexity a priori); it suffices to swap rows q and r of matrices P and T (a fast operation with $\mathcal{O}(n)$ algorithmic complexity). At the end of the day, the solution algorithm proceeds as in Algorithm 3, applying the permutation P on the right hand side as a preliminary step.

2.7 Theoretical results regarding LU

Apart from the construction and practical details of the LU decomposition that we have presented, theoretical questions naturally arise concerning this factorization.

Théorème 2.1.
Given an invertible matrix $\mathrm{A} \in \mathbb{C}^{n\times n}$*, we have* $\det(\mathrm{A}_{\mathrm{J,J}}) \neq 0$ *for* $\mathrm{J} = [\![1, j]\!]$ *and for all* $j = 1 \dots n-1$ *if and only if there exists a unique pair* $\mathrm{L}, \mathrm{U} \in \mathbb{C}^{n\times n}$ *satisfying* $\mathrm{A} = \mathrm{LU}$ *with* U *upper triangular and* L *lower triangular with* $\mathrm{L}_{j,j} = 1 \, \forall j = 1 \dots n$*.*

Proof:

We proceed by reccurence on the dimension n. The result for $n = 1$ is obvious. Assume that he result holds for $\mathrm{A} \in \mathbb{C}^{m\times m}$ invertible with $m = 1 \dots n$, and let us prove that this result still holds $\mathrm{A} \in \mathbb{C}^{(n+1)\times(n+1)}$ invertible.

Now assume for a moment the existence/uniqueness of the decomposition $\mathrm{A} = \mathrm{LU}$. Then we necessarily have $\mathrm{A}_{\mathrm{J,J}} = \mathrm{L}_{\mathrm{J,J}}\mathrm{U}_{\mathrm{J,J}}$ for all $\mathrm{J} = [\![1, k]\!]$ and all $k = 1 \dots n$. On the other hand, $0 \neq \det(\mathrm{A}) = \det(\mathrm{U}) = \Pi_{j=1}^{n+1}\mathrm{U}_{j,j} \Rightarrow 0 \neq \Pi_{j=1}^{k}\mathrm{U}_{j,j} = \det(\mathrm{A}_{\mathrm{J,J}})$. Now let us set $\mathrm{B} := \mathrm{A}_{\mathrm{J,J}}$ for $\mathrm{J} = [\![1, n]\!]$, so that

$$\mathrm{A} = \begin{bmatrix} \mathrm{B} & \boldsymbol{a} \\ \boldsymbol{b}^* & \alpha \end{bmatrix} \tag{2.8}$$

where $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{C}^n$ and $\alpha \in \mathbb{C}$. Now we assume that $\det(\mathrm{A}_{\mathrm{J,J}}) \neq 0$ for $\mathrm{J} = [\![1, j]\!]$ and for all $j = 1, \dots, n$, as well as $\det(\mathrm{A}) \neq 0$. According to the recurrence hypothesis, there is existence/uniqueness of the factorization $\mathrm{B} = \tilde{\mathrm{L}}\tilde{\mathrm{U}}$. As a consequence we have the decomposition $\mathrm{A} = \mathrm{LU}$ if and only if there exist $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{C}^n$ and $\beta \in \mathbb{C} \setminus \{0\}$ such that

$$\mathrm{L} = \begin{bmatrix} \tilde{\mathrm{L}} & 0 \\ \boldsymbol{v}^* & 1 \end{bmatrix} \quad \mathrm{U} = \begin{bmatrix} \tilde{\mathrm{U}} & \boldsymbol{u} \\ 0 & \beta \end{bmatrix} \quad \text{et} \quad \begin{array}{l} \tilde{\mathrm{L}}\boldsymbol{u} = \boldsymbol{a} \\ \tilde{\mathrm{U}}^*\boldsymbol{v} = \boldsymbol{b} \\ \beta + \boldsymbol{v}^*\boldsymbol{u} = \alpha \end{array} \tag{2.9}$$

Besides $\det(\tilde{\mathrm{L}}) = 1$ and $\det(\tilde{\mathrm{U}}) = \det(\tilde{\mathrm{B}}) \neq 0$. Hence this leads to existence and uniqueness of $\boldsymbol{u}, \boldsymbol{v}, \beta$ solution to (2.9) and, from the existence/uniqueness of $\tilde{\mathrm{L}}, \tilde{\mathrm{U}}$ we deduce existence/uniqueness of L, U. □

The criterium on the $\det(\mathrm{A}_{\mathrm{J,J}})$'s mentionned above is not very easy to verify in practice. However there exist certain classes of matrices for which this criterium is systematically satisfied. This is the case of diagonally dominant matrices. Recall that a matrix $\mathrm{A} = (a_{j,k}) \in \mathbb{C}^{n\times n}$ is said row-wise diagonally dominant if $|a_{j,j}| > \sum_{k\neq j} |a_{j,k}| \ \forall j = 1 \dots n$. Similalry it is said column-wise diagonally dominant when $|a_{j,j}| > \sum_{k\neq j} |a_{k,j}| \ \forall j = 1 \dots n$.

Proposition 2.2.
If the matrix $\mathrm{A} \in \mathbb{C}^{n\times n}$ *is either row-wise or column-wise diagonally dominant, then it admits a unique LU-decomposition.*

Démo:

Let us assume row-wise diagonal dominance, since the case of a column-wise diagonal dominance proceed very similarly. According to Theorem 2.1, it suffices to show that $\det(\mathrm{A}_{\mathrm{J,J}}) \neq 0$ for $\mathrm{J} = [\![1, k]\!]$ for all $k = 1 \dots n$. It appears clear that if A is row-wise diagonally dominant, this is also the case for $\mathrm{A}_{\mathrm{J,J}}$. Hence there remains to verify that a rwo-wise diagonally dominant matrix is necessarily invertible.

Let us proceed by contradiction and consider a row-wise diagonally dominant matrix $\mathrm{B} = (b_{j,k}) \in \mathbb{C}^{n\times n}$ such that $\mathrm{B}\boldsymbol{u} = 0$ for a certain $\boldsymbol{u} = (u_j) \in \mathbb{C}^n \setminus \{0\}$. Let $p \in \{1 \dots n\}$ such that $|u_p| = \max_{j=1\dots n} |u_j| = |\boldsymbol{u}|_\infty$. Considering $\boldsymbol{u}/u_p$ instead of $\boldsymbol{u}$ if necessary, we may assume

that $u_p = 1$ and thus $|u_j| \leq 1$ for all $j = 1 \dots n$. Then we have $0 = b_{p,p} + \sum_{j \neq p} b_{p,j} u_j$ and thus $|b_{p,p}| = |\sum_{j \neq p} b_{p,j} u_j| \leq \sum_{j \neq p} |b_{p,j}|$ which contradicts the diagonal dominance. □

The next result shows that if the matrix A a band structure with a certain width, then each of the factors of the LU decomposition admits the same band structure.

Proposition 2.3.
Let $\mathrm{A} \in \mathbb{C}^{n \times n}$ *a matrix admitting a unique LU factorization. Let us assume in addition that there exists* $p \geq 0$ *such that* $\mathrm{A}_{j,k} = 0$ *if* $|j - k| > p$. *Then we also have* $L_{j,k} = U_{j,k} = 0$ *for* $|j - k| > p$.

Démo:

Again we proceed by recurrence on the band width, and assume that the result holds for all matrices uniquely LU-factorizable with a band width of m with $1 \leq m \leq n$. We pick a matrix $\mathrm{A} \in \mathbb{C}^{(n+1)\times(n+1)}$ satsifying the assumptions of the proposition we are seeking to prove. Coming back to the notations of the proof of Theorem 2.1, we must in particular have (2.9) with $\tilde{\mathrm{L}}_{j,k} = \tilde{\mathrm{U}}_{j,k} = 0$ whenever $|j - k| > p$ according to the recurrence hypothesis.

Let us also take the notation $\boldsymbol{u} = (u_j), \boldsymbol{v} = (v_j), \boldsymbol{a} = (a_j), \boldsymbol{b} = (b_j)$ for the vectors coming into play in (2.9) and (2.8). To prove that L and U admit the same band structure, that is $\mathrm{L}_{j,k} = \mathrm{U}_{j,k} = 0$ for $|j - k| > p$, there remains to verify that $u_j = v_j = 0$ for $|j - n| > p \iff j < n - p$. We already know that $a_j = b_j = 0$ for $j < n - p$ according to the band structure satisfied by A. On the other hand $\tilde{\mathrm{L}}$ and $\tilde{\mathrm{U}}^*$ are lower triangular, hence $(\tilde{\mathrm{L}})^{-1}$ and $(\tilde{\mathrm{U}}^*)^{-1}$ also, and we have $\boldsymbol{u} = (\tilde{\mathrm{L}})^{-1}\boldsymbol{a}$ and $\boldsymbol{v} = (\tilde{\mathrm{U}}^*)^{-1}\boldsymbol{b}$, which indeed implies $u_j = v_j = 0$ for $j < n - p$. □

2.8 Cholesky factorization

Recall that a matrix $\mathrm{A} \in \mathbb{C}^{n \times n}$ is is hermitian whenever $\mathrm{A} = \mathrm{A}^*$, and it is called positive when $\boldsymbol{x}^* \mathrm{A} \boldsymbol{x} \in (0, +\infty)$ for all $\boldsymbol{x} \in \mathbb{C}^n$. Finally, when it is positive, the matrix A is called definite if $\boldsymbol{x}^* \mathrm{A} \boldsymbol{x} = 0 \Rightarrow \boldsymbol{x} = 0$. In the case where the matrix $\mathrm{A} \in \mathbb{C}^{n \times n}$ is hermitian positive definite (HPD), its LU-factorization simplifies, and we can obtain more explicit formulas.

Théorème 2.4.
Let $\mathrm{A} = (a_{j,k})_{j,k=1\dots n} \in \mathbb{C}^{n \times n}$ *be hermitian positive definite. Then there exists a lower triangular matrix* $\mathrm{H} \in \mathbb{C}^{n \times n}$ *whose diagonal terms are strictly real positive and such that the following so-called Cholesky factorization holds*

$$\mathrm{A} = \mathrm{H}\mathrm{H}^*.$$

Démo:

Again we proceed by recurrence on the size n. In the case $n = 1$, we have $\mathrm{A} = (a_{1,1})$. Since A is hermitian positive definite, we have $a_{1,1} > 0$. We can thus define $\mathrm{H} = (h_{1,1})$ where $h_{1,1} = \sqrt{a_{1,1}}$. Let su assume that the Cholesky factorization uniquely exists for any $\mathrm{A} \in \mathbb{C}^{p \times p}$ HPD, for $1 \leq p \leq n$, and let us prove that this property still holds $\mathrm{A} \in \mathbb{C}^{(n+1)\times(n+1)}$ HPD. Pick an arbitrary HPD matrix $\mathrm{A} \in \mathbb{C}^{(n+1)\times(n+1)}$. We can write it in the following form

$$\mathrm{A} = \begin{bmatrix} \mathrm{B} & \boldsymbol{a} \\ \boldsymbol{a}^* & \alpha \end{bmatrix} \tag{2.10}$$

where $\mathrm{B} \in \mathbb{C}^{n\times n}$ is hermitian, $\boldsymbol{a} \in \mathbb{C}^n$ and $\alpha \in \mathbb{R}_+^*$. Let us prove that B is positive definite, that is $\boldsymbol{y}^*\mathrm{B}\boldsymbol{y} > 0$, for all $\boldsymbol{y} \in \mathbb{C}^n \setminus \{0\}$. Pick an arbitrary $\boldsymbol{y} \in \mathbb{C}^n \setminus \{0\}$, and define $\boldsymbol{x} \in \mathbb{C}^{n+1}$ by $\boldsymbol{x}^\top = (\boldsymbol{y}^\top, 0)$. Since A is HPD, we have $0 < \boldsymbol{x}^*\mathrm{A}\boldsymbol{x} = \boldsymbol{y}^*\mathrm{B}\boldsymbol{y}$ hence B is HPD. According to the recurrence hypothesis, there exists a matrix $\mathrm{M} \in \mathbb{C}^{n\times n}$, $\mathrm{M} = (m_{i,j})_{i,j=1}^n$, such that $m_{i,j} = 0$ if $j > i$, $m_{i,i} > 0$ and $\mathrm{B} = \mathrm{MM}^*$. We look for H under the form:

$$\mathrm{H} = \begin{bmatrix} \mathrm{M} & 0 \\ \boldsymbol{b}^* & \beta \end{bmatrix} \tag{2.11}$$

with $\boldsymbol{b} \in \mathbb{C}^n$, $\beta \in \mathbb{R}_+^*$ and such that $\mathrm{HH}^* = \mathrm{A}$. To determine $\boldsymbol{b}$ and β, let us compute HH^* from (2.11) and let us identify with A :

$$\mathrm{HH}^* = \begin{bmatrix} \mathrm{MM}^* & \mathrm{M}\boldsymbol{b} \\ (\mathrm{M}\boldsymbol{b})^* & |\boldsymbol{b}|^2 + \beta^2 \end{bmatrix} \tag{2.12}$$

Comparing the above identity with (2.10), we deduce that the following equations must be satisfied

$$\mathrm{M}\boldsymbol{b} = \boldsymbol{a} \quad \text{et} \quad |\boldsymbol{b}|^2 + \beta^2 = \alpha.$$

Let us note that M is invertible since $\det(\mathrm{M}) = m_{1,1} \cdot m_{2,2} \cdots m_{n,n} > 0$. We can thus set $\boldsymbol{b} := \mathrm{M}^{-1}\boldsymbol{a}$ as a definition. In practice we can determine $\boldsymbol{b}$ by a simple forward substitution method since M is lower triangular. Plugging $\mathrm{B} = \mathrm{MM}^*$ into the second equality yields $\boldsymbol{a}^*\mathrm{B}^{-1}\boldsymbol{a} + \beta^2 = \alpha$. It thus suffices to set $\beta := (\alpha - \boldsymbol{a}^*\mathrm{B}^{-1}\boldsymbol{a})^{1/2}$. However such a definition is valid only if

$$\alpha - \boldsymbol{a}^*\mathrm{B}^{-1}\boldsymbol{a} > 0 \tag{2.13}$$

Let us prove that this condition is indeed satisfied by (2.10). Let us consider the vector $\boldsymbol{z} \in \mathbb{C}^{n+1} \setminus \{0\}$ defined by $\boldsymbol{z}^\top = ((\mathrm{B}^{-1}\boldsymbol{a})^\top, -1)$. Then we have $0 < \boldsymbol{z}^*\mathrm{A}\boldsymbol{z} = \alpha - \boldsymbol{a}^*\mathrm{B}^{-1}\boldsymbol{a}$. This leads to the conclusion. □

A very explicit algorithm can be proposied for Cholesky factorization. By directly expressing the matrix product $\mathrm{A} = \mathrm{HH}^*$, coefficient by coefficient, we obtain $a_{j,j} = \sum_{k=1}^j |h_{j,k}|^2$ and $a_{j,k} = \sum_{p=1}^k h_{j,p}\overline{h}_{k,p}$ for $k < j$. From there we deduce the following formulas

$$\begin{aligned} h_{j,j} &= (a_{j,j} - \textstyle\sum_{k=1}^{j-1} |h_{j,k}|^2)^{1/2}, \\ h_{j,k} &= (a_{j,k} - \textstyle\sum_{p=1}^{k-1} h_{j,p}\overline{h}_{k,p})/\overline{h}_{k,k} \quad \text{pour } k < j. \end{aligned}$$

The construction of the matrix H then proceeds for j growing from 1 to n and for k growing from 1 to j. These formulas yield Algorithm 5 below.

Algorithm 5

```
function CHOLESKY(A)
    H = 0
    for j = 1...n do
        for k = 1...j do
            H_{j,k} = A_{j,k}
            for p = 1...k-1 do
                H_{j,k} = H_{j,k} - H_{j,p} conj(H_{k,p})
            end for
            if k < j then
                H_{j,k} = H_{j,k} / H_{k,k}
            else
                H_{j,j} = sqrt(H_{j,j})
            end if
        end for
    end for
    return (H)
end function
```