7 Gaussian Elimination and LU Factorization

In this final section on matrix factorization methods for solving $A\boldsymbol{x} = \boldsymbol{b}$ we want to take a closer look at Gaussian elimination (probably the best known method for solving systems of linear equations).

The basic idea is to use left-multiplication of $A \in \mathbb{C}^{m\times m}$ by (elementary) lower triangular matrices, $L_1, L_2, \ldots, L_{m-1}$ to convert A to upper triangular form, i.e.,

$$\underbrace{L_{m-1}L_{m-2}\ldots L_2L_1}_{=\widetilde{L}}A = U.$$

Note that the product of lower triangular matrices is a lower triangular matrix, and the inverse of a lower triangular matrix is also lower triangular. Therefore,

$$\widetilde{L}A = U \quad \Longleftrightarrow \quad A = LU,$$

where $L = \widetilde{L}^{-1}$. This approach can be viewed as *triangular triangularization*.

7.1 Why Would We Want to Do This?

Consider the system $A\boldsymbol{x} = \boldsymbol{b}$ with LU factorization $A = LU$. Then we have

$$L\underbrace{U\boldsymbol{x}}_{=\boldsymbol{y}} = \boldsymbol{b}.$$

Therefore we can perform (a now familiar) 2-step solution procedure:

1. Solve the lower triangular system $L\boldsymbol{y} = \boldsymbol{b}$ for $\boldsymbol{y}$ by forward substitution.
2. Solve the upper triangular system $U\boldsymbol{x} = \boldsymbol{y}$ for $\boldsymbol{x}$ by back substitution.

Moreover, consider the problem $AX = B$ (i.e., many different right-hand sides that are associated with the same system matrix). In this case we need to compute the factorization $A = LU$ only once, and then

$$AX = B \quad \Longleftrightarrow \quad LUX = B,$$

and we proceed as before:

1. Solve $LY = B$ by many forward substitutions (in parallel).
2. Solve $UX = Y$ by many back substitutions (in parallel).

In order to appreciate the usefulness of this approach note that the operations count for the matrix factorization is $\mathcal{O}(\frac{2}{3}m^3)$, while that for forward and back substitution is $O(m^2)$.

Example Take the matrix

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 3 & 5 \\ 4 & 6 & 8 \end{bmatrix}$$

and compute its LU factorization by applying elementary lower triangular transformation matrices.

We choose L_1 such that left-multiplication corresponds to subtracting multiples of row 1 from the rows below such that the entries in the first column of A are zeroed out (cf. the first homework assignment). Thus

$$L_1 A = \begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -4 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 2 & 3 & 5 \\ 4 & 6 & 8 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 3 \\ 0 & 2 & 4 \end{bmatrix}.$$

Next, we repeat this operation analogously for L_2 (in order to zero what is left in column 2 of the matrix on the right-hand side above):

$$L_2(L_1 A) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -2 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 3 \\ 0 & 2 & 4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 3 \\ 0 & 0 & -2 \end{bmatrix} = U.$$

Now $L = (L_2 L_1)^{-1} = L_1^{-1} L_2^{-1}$ with

$$L_1^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 4 & 0 & 1 \end{bmatrix} \qquad \text{and } L_2^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 2 & 1 \end{bmatrix},$$

so that

$$L = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 4 & 2 & 1 \end{bmatrix}.$$

Remark Note that L always is a *unit lower triangular* matrix, i.e., it has ones on the diagonal. Moreover, L is always obtained as above, i.e., the multipliers are accumulated into the lower triangular part with a change of sign.

The claims made above can be verified as follows. First, we note that the multipliers in L_k are of the form

$$\ell_{jk} = \frac{a_{jk}}{a_{kk}}, \qquad j = k+1, \ldots, m,$$

so that

$$L_k = \begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & -\ell_{k+1,k} & 1 & & \\ & & \vdots & & \ddots & \\ & & -\ell_{m,k} & & & 1 \end{bmatrix}.$$

Now, let

$$\boldsymbol{\ell}_k = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ \ell_{k+1,k} \\ \vdots \\ \ell_{m,k} \end{bmatrix}.$$

Then $L_k = I - \boldsymbol{\ell}_k \boldsymbol{e}_k^*$, and therefore

$$\underbrace{(I - \boldsymbol{\ell}_k \boldsymbol{e}_k^*)}_{=L_k} \underbrace{(I + \boldsymbol{\ell}_k \boldsymbol{e}_k^*)}_{L_k^{-1}} = I - \boldsymbol{\ell}_k \boldsymbol{e}_k^* \boldsymbol{\ell}_k \boldsymbol{e}_k^* = I,$$

since the inner product $\boldsymbol{e}_k^* \boldsymbol{\ell}_k = 0$ because the only nonzero entry in $\boldsymbol{e}_k$ (the 1 in the k-th position) does not "hit" any nonzero entries in $\boldsymbol{\ell}_k$ which start in the $k+1$-st position.

So, for any k we have

$$L_k^{-1} = \begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & \ell_{k+1,k} & 1 & & \\ & & \vdots & & \ddots & \\ & & \ell_{m,k} & & & 1 \end{bmatrix}$$

as claimed.

In addition,

$$\begin{aligned} L_k^{-1} L_{k+1}^{-1} &= (I + \boldsymbol{\ell}_k \boldsymbol{e}_k^*)\left(I + \boldsymbol{\ell}_{k+1} \boldsymbol{e}_{k+1}^*\right) \\ &= I + \boldsymbol{\ell}_k \boldsymbol{e}_k^* + \boldsymbol{\ell}_{k+1} \boldsymbol{e}_{k+1}^* + \boldsymbol{\ell}_k \underbrace{\boldsymbol{e}_k^* \boldsymbol{\ell}_{k+1}}_{=0} \boldsymbol{e}_{k+1} \\ &= \begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & \ell_{k+1,k} & 1 & & \\ & & \vdots & \ell_{k+2,k+1} & & \\ & & & \vdots & \ddots & \\ & & \ell_{m,k} & \ell_{m,k+1} & & 1 \end{bmatrix}, \end{aligned}$$

and in general we have

$$L = L_1^{-1} \ldots L_{m-1}^{-1} = \begin{bmatrix} 1 & & & & \\ \ell_{2,1} & 1 & & & \\ \vdots & \ell_{3,2} & \ddots & & \\ & \vdots & & 1 & \\ \ell_{m,1} & \ell_{m,2} & \ldots & \ell_{m,m-1} & 1 \end{bmatrix}.$$

We can summarize the factorization in

Algorithm (LU Factorization)

Initialize $U = A$, $L = I$

for $k = 1 : m-1$

for $j = k+1 : m$

$L(j,k) = U(j,k)/U(k,k)$
$U(j,k:m) = U(j,k:m) - L(j,k)U(k,k:m)$

end

end

Remark 1. In practice one can actually store both L and U in the original matrix A since it is known that the diagonal of L consists of all ones.

2. The LU factorization is the cheapest factorization algorithm. Its operations count can be verified to be $\mathcal{O}(\frac{2}{3}m^3)$.

However, *LU factorization cannot be guaranteed to be stable*. The following examples illustrate this fact.

Example A fundamental problem is given if we encounter a *zero pivot* as in

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 2 & 5 \\ 4 & 6 & 8 \end{bmatrix} \qquad \Longrightarrow \qquad L_1 A = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 3 \\ 0 & 2 & 4 \end{bmatrix}.$$

Now the (2,2) position contains a zero and the algorithm will break down since it will attempt to divide by zero.

Example A more subtle example is the following *backward instability*. Take

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 2+\varepsilon & 5 \\ 4 & 6 & 8 \end{bmatrix}$$

with small ε. If $\varepsilon = 1$ then we have the initial example in this chapter, and for $\varepsilon = 0$ we get the previous example.

LU factorization will result in

$$L_1 A = \begin{bmatrix} 1 & 1 & 1 \\ 0 & \varepsilon & 3 \\ 0 & 2 & 4 \end{bmatrix}$$

and

$$L_2 L_1 A = \begin{bmatrix} 1 & 1 & 1 \\ 0 & \varepsilon & 3 \\ 0 & 0 & 4 - \frac{6}{\varepsilon} \end{bmatrix} = U.$$

The multipliers were

$$L = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 4 & \frac{2}{\varepsilon} & 1 \end{bmatrix}.$$

Now we assume that a right-hand side $\boldsymbol{b}$ is given as

$$\boldsymbol{b} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

and we attempt to solve $A\boldsymbol{x} = \boldsymbol{b}$ via

1. Solve $L\boldsymbol{y} = \boldsymbol{b}$.

2. Solve $U\boldsymbol{x} = \boldsymbol{y}$.

If ε is on the order of machine accuracy, then the 4 in the entry $4 - \frac{6}{\varepsilon}$ in U is insignificant. Therefore, we have

$$\tilde{U} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & \varepsilon & 3 \\ 0 & 0 & -\frac{6}{\varepsilon} \end{bmatrix} \qquad \text{and } \tilde{L} = L,$$

which leads to

$$\tilde{L}\tilde{U} = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 2+\varepsilon & 5 \\ 4 & 6 & 4 \end{bmatrix} \neq A.$$

In fact, the product is significantly different from A. Thus, using $\tilde{L}$ and $\tilde{U}$ we are not able to solve a "nearby problem", and thus the LU factorization method is *not backward stable*.

If we use the factorization based on $\tilde{L}$ and $\tilde{U}$ with the above right-hand side $\boldsymbol{b}$, then we obtain

$$\tilde{\boldsymbol{x}} = \begin{bmatrix} \frac{11}{2} - \frac{2}{3}\varepsilon \\ -2 \\ \frac{2}{3}\varepsilon - \frac{2}{3} \end{bmatrix} \approx \begin{bmatrix} \frac{11}{2} \\ -2 \\ -\frac{2}{3} \end{bmatrix}.$$

Whereas if we were to use the exact factorization $A = LU$, then we get the exact answer

$$\boldsymbol{x} = \begin{bmatrix} \frac{4\varepsilon-7}{2\varepsilon-3} \\ \frac{2}{2\varepsilon-3} \\ -2\frac{\varepsilon-1}{2\varepsilon-3} \end{bmatrix} \approx \begin{bmatrix} \frac{7}{3} \\ -\frac{2}{3} \\ -\frac{2}{3} \end{bmatrix}.$$

Remark Even though $\tilde{L}$ and $\tilde{U}$ are close to L and U, the product $\tilde{L}\tilde{U}$ is not close to $LU = A$ and the computed solution $\tilde{\boldsymbol{x}}$ is worthless.

7.2 Pivoting

Example The breakdown of the algorithm in our earlier example with

$$L_1 A = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 3 \\ 0 & 2 & 3 \end{bmatrix}$$

can be prevented by simply *swapping rows*, i.e., instead of trying to apply L_2 to $L_1 A$ we first create

$$PL_1 A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} L_1 A = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 2 & 3 \\ 0 & 0 & 3 \end{bmatrix}$$

— and are done.

More generally, stability problems can be avoided by swapping rows before applying L_k, i.e., we perform

$$L_{m-1}P_{m-1}\dots L_2P_2L_1P_1A = U.$$

The strategy we use for swapping rows in step k is to find the largest element in column k below (and including) the diagonal — the so-called *pivot element* — and swap its row with row k. This process is referred to as *partial* (row) *pivoting*. Partial column pivoting and complete (row and column) pivoting are also possible, but not very popular.

Example Consider again the matrix

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 2+\varepsilon & 5 \\ 4 & 6 & 8 \end{bmatrix}$$

The largest element in the first column is the 4 in the $(3,1)$ position. This is our first pivot, and we swap rows 1 and 3. Therefore

$$P_1A = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}\begin{bmatrix} 1 & 1 & 1 \\ 2 & 2+\varepsilon & 5 \\ 4 & 6 & 8 \end{bmatrix} = \begin{bmatrix} 4 & 6 & 8 \\ 2 & 2+\varepsilon & 5 \\ 1 & 1 & 1 \end{bmatrix},$$

and then

$$L_1P_1A = \begin{bmatrix} 1 & 0 & 0 \\ -\frac{1}{2} & 1 & 0 \\ -\frac{1}{4} & 0 & 1 \end{bmatrix}\begin{bmatrix} 4 & 6 & 8 \\ 2 & 2+\varepsilon & 5 \\ 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 4 & 6 & 8 \\ 0 & \varepsilon-1 & 1 \\ 0 & -\frac{1}{2} & -1 \end{bmatrix}.$$

Now we need to pick the second pivot element. For sufficiently small ε (in fact, unless $\frac{1}{2} < \varepsilon < \frac{3}{2}$), we pick $\varepsilon - 1$ as the largest element in the second column below the first row. Therefore, the second permutation matrix is just the identity, and we have

$$P_2L_1P_1A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 4 & 6 & 8 \\ 0 & \varepsilon-1 & 1 \\ 0 & -\frac{1}{2} & -1 \end{bmatrix} = \begin{bmatrix} 4 & 6 & 8 \\ 0 & \varepsilon-1 & 1 \\ 0 & -\frac{1}{2} & -1 \end{bmatrix}.$$

To complete the elimination phase, we need to perform the elimination in the second column:

$$L_2P_2L_1P_1A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & \frac{1}{2(\varepsilon-1)} & 1 \end{bmatrix}\begin{bmatrix} 4 & 6 & 8 \\ 0 & \varepsilon-1 & 1 \\ 0 & -\frac{1}{2} & -1 \end{bmatrix} = \begin{bmatrix} 4 & 6 & 8 \\ 0 & \varepsilon-1 & 1 \\ 0 & 0 & \frac{3-2\varepsilon}{2(\varepsilon-1)} \end{bmatrix} = U.$$

The lower triangular matrix L is given by

$$L = L_1^{-1}L_2^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{2} & 1 & 0 \\ \frac{1}{4} & -\frac{1}{2(\varepsilon-1)} & 1 \end{bmatrix},$$

and assuming that $\varepsilon - 1 \approx -1$ we get

$$\tilde{L} = \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{2} & 1 & 0 \\ \frac{1}{4} & \frac{1}{2} & 1 \end{bmatrix} \quad \text{and} \quad \tilde{U} = \begin{bmatrix} 4 & 6 & 8 \\ 0 & -1 & 1 \\ 0 & 0 & -\frac{3}{2} \end{bmatrix}.$$

If we now check the computed factorization $\tilde{L}\tilde{U}$, then we see

$$\tilde{L}\tilde{U} = \begin{bmatrix} 4 & 6 & 8 \\ 2 & 2 & 5 \\ 1 & 1 & 1 \end{bmatrix} = P\tilde{A},$$

which is just a permuted version of the original matrix A with permutation matrix

$$P = P_2 P_1 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}.$$

Thus, this approach was backward stable.

Finally, since we have the factorization $PA = LU$, we can solve the linear system $A\boldsymbol{x} = \boldsymbol{b}$ as

$$PA\boldsymbol{x} = P\boldsymbol{b} \quad \Longleftrightarrow \quad LU\boldsymbol{x} = P\boldsymbol{b},$$

and apply the usual two-step procedure

1. Solve the lower triangular system $L\boldsymbol{y} = P\boldsymbol{b}$ for $\boldsymbol{y}$.
2. Solve the upper triangular system $U\boldsymbol{x} = \boldsymbol{y}$ for $\boldsymbol{x}$.

This yields

$$\boldsymbol{x} = \begin{bmatrix} \frac{-7+4\varepsilon}{-3+2\varepsilon} \\ \frac{2}{2\varepsilon-3} \\ -2\frac{\varepsilon-1}{2\varepsilon-3} \end{bmatrix} \approx \begin{bmatrix} \frac{7}{3} \\ -\frac{2}{3} \\ -\frac{2}{3} \end{bmatrix}.$$

If we use the rounded factors $\tilde{L}$ and $\tilde{U}$ instead, then the computed solution is

$$\tilde{\boldsymbol{x}} = \begin{bmatrix} \frac{7}{3} \\ -\frac{2}{3} \\ -\frac{2}{3} \end{bmatrix},$$

which is the exact answer to the problem (see also the Maple worksheet `473_LU.mws`).

In general, LU factorization with pivoting results in

$$PA = LU,$$

where $P = P_{m-1}P_{m-2}\dots P_2P_1$, and $L = (L'_{m-1}L'_{m-2}\dots L'_2L'_1)^{-1}$ with

$$L'_k = P_{m-1}\dots P_{k+1}L_kP_{k+1}^{-1}\dots P_{m-1}^{-1},$$

i.e., L'_k is the same as L_k except that the entries below the diagonal are appropriately permuted. In particular, L'_k is still lower triangular.

Remark Since the permutation matrices used here involve only a single row swap each we have $P_k^{-1} = P_k$ (while in general, of course, $P^{-1} = P^T$).

In the example above $L'_2 = L_2$, and $L'_1 = P_2L_1P_2^{-1} = L_1$ since $P_2 = I$.

Remark Due to the pivoting strategy the multipliers will always satisfy $|\ell_{ij}| \leq 1$.

A possible interpretation of the pivoting strategy is that the matrix P is determined so that it would yield a permuted matrix A whose standard LU factorization is backward stable. Of course, we do not know how to do this in advance, and so the P is determined as the algorithm progresses.

An algorithm for the factorization of an $m \times m$ matrix A is given by

Algorithm (LU Factorization with Partial Pivoting)

Initialize $U = A$, $L = I$, $P = I$

for $k = 1 : m - 1$

find $i \geq k$ to maximize $|U(i, k)|$
$U(k, k : m) \longleftrightarrow U(i, k : m)$
$L(k, 1 : k - 1) \longleftrightarrow L(i, 1 : k - 1)$
$P(k, :) \longleftrightarrow P(i, :)$
for $j = k + 1 : m$
$L(j, k) = U(j, k)/U(k, k)$
$U(j, k : m) = U(j, k : m) - L(j, k)U(k, k : m)$
end

end

The operations count for this algorithm is also $\mathcal{O}(\frac{2}{3}m^2)$. However, while the swaps for partial pivoting require $\mathcal{O}(m^2)$ operations, they would require $\mathcal{O}(m^3)$ operations in the case of complete pivoting.

Remark The algorithm above is not really practical since one would usually not physically swap rows. Instead one would use pointers to the swapped rows and store the permutation operations instead.

7.3 Stability

We saw earlier that Gaussian elimination without pivoting is can be unstable. According to our previous example the algorithm with pivoting seems to be stable. What can be proven theoretically?

Since the entries of L are at most 1 in absolute value, the LU factorization becomes unstable if the entries of U are unbounded relative to those of A (we need $\|L\|\|U\| = \mathcal{O}(\|PA\|)$. Therefore we define a *growth factor*

$$\rho = \frac{\max_{i,j} |U(i, j)|}{\max_{i,j} |A(i, j)|}.$$

One can show

Theorem 7.1 *Let $A \in \mathbb{C}^{m\times m}$. Then LU factorization with partial pivoting guarantees that $\rho \leq 2^{m-1}$.*

This bound is unacceptably high, and indicates that the algorithm (with pivoting) can be unstable. The following (contrived) example illustrates this.

Example Take

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ -1 & 1 & 0 & 0 & 1 \\ -1 & -1 & 1 & 0 & 1 \\ -1 & -1 & -1 & 1 & 1 \\ -1 & -1 & -1 & -1 & 1 \end{bmatrix}.$$

Then LU factorization produces the following sequence of matrices:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 2 \\ 0 & -1 & 1 & 0 & 2 \\ 0 & -1 & -1 & 1 & 2 \\ 0 & -1 & -1 & -1 & 2 \end{bmatrix} \longrightarrow \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 2 \\ 0 & 0 & 1 & 0 & 4 \\ 0 & 0 & -1 & 1 & 4 \\ 0 & 0 & -1 & -1 & 4 \end{bmatrix}$$

$$\longrightarrow \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 2 \\ 0 & 0 & 1 & 0 & 4 \\ 0 & 0 & 0 & 1 & 8 \\ 0 & 0 & 0 & -1 & 8 \end{bmatrix} \longrightarrow \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 2 \\ 0 & 0 & 1 & 0 & 4 \\ 0 & 0 & 0 & 1 & 8 \\ 0 & 0 & 0 & 0 & 16 \end{bmatrix} = U,$$

and we see that the largest element in U is $16 = 2^{m-1}$.

Remark BUT in practice it turns out that matrices which such large growth factors *almost never* arise. For most practical cases a realistic bound on ρ seems to be $\sqrt{m}$. This is still an active area of research and some more details can be found in the [Trefethen/Bau] book.

In summary we can say that for most practical purposes LU factorization with partial pivoting is considered to be a *stable* algorithm.

7.4 Cholesky Factorization

In the case when the matrix A is Hermitian (or symmetric) positive definite we can devise a faster (and more stable) algorithm.

Recall that $A \in \mathbb{C}^{m\times m}$ is Hermitian if $A^* = A$. This implies $\boldsymbol{x}^* A\boldsymbol{y} = \overline{\boldsymbol{y}^* A\boldsymbol{x}}$ for any $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{C}^m$ (see homework for details). If we let $\boldsymbol{y} = \boldsymbol{x}$ then we see that $\boldsymbol{x}^* A\boldsymbol{x}$ is real.

Now, if $\boldsymbol{x}^* A\boldsymbol{x} > 0$ for any $\boldsymbol{x} \neq \boldsymbol{0}$ then A is called Hermitian *positive definite*.

Remark Symmetric positive definite matrices are defined analogously with * replaced by ^{T}.

7.4.1 Some Properties of Positive Definite Matrices

Assume $A \in \mathbb{C}^{m\times m}$ is positive definite.

1. If $X \in \mathbb{C}^{m\times n}$ has full rank, then $X^* AX \in \mathbb{C}^{n\times n}$ is positive definite.

2. Any principal submatrix (i.e., the intersection of a set of rows and the corresponding columns) of A is positive definite.

3. All diagonal entries of A are positive and real.

4. The maximum element of A is on the diagonal.

5. All eigenvalues of A are positive.

7.4.2 How Does Cholesky Factorization Work?

Consider the Hermitian positive definite $A \in \mathbb{C}^{m\times m}$ in block form

$$A = \begin{bmatrix} 1 & \boldsymbol{w}^* \\ \boldsymbol{w} & K \end{bmatrix}$$

and apply the first step of the LU factorization algorithm. Then

$$A = \begin{bmatrix} 1 & \mathbf{0}^T \\ \boldsymbol{w} & I \end{bmatrix} \begin{bmatrix} 1 & \boldsymbol{w}^* \\ \mathbf{0} & K - \boldsymbol{w}\boldsymbol{w}^* \end{bmatrix}.$$

The main idea of the Cholesky factorization algorithms is to take advantage of the fact that A is Hermitian and *perform operations symmetrically*, i.e., also zero the first row. Thus

$$A = \begin{bmatrix} 1 & \mathbf{0}^T \\ \boldsymbol{w} & I \end{bmatrix} \begin{bmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & K - \boldsymbol{w}\boldsymbol{w}^* \end{bmatrix} \begin{bmatrix} 1 & \boldsymbol{w}^* \\ \mathbf{0} & I \end{bmatrix}.$$

Note that the three matrices are lower triangular, block-diagonal, and upper triangular (in fact the adjoint of the lower triangular factor).

Now we continue iteratively. However, in general A will not have a 1 in the upper left-hand corner. We will have to be able to deal with an arbitrary (albeit positive) entry in the (1,1) position. Therefore we reconsider the first step with slightly more general notation, i.e.,

$$A = \begin{bmatrix} a_{11} & \boldsymbol{w}^* \\ \boldsymbol{w} & K \end{bmatrix}$$

so that

$$A = \begin{bmatrix} \sqrt{a_{11}} & \mathbf{0}^T \\ \frac{1}{\sqrt{a_{11}}}\boldsymbol{w} & I \end{bmatrix} \begin{bmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & K - \frac{1}{a_{11}}\boldsymbol{w}\boldsymbol{w}^* \end{bmatrix} \begin{bmatrix} \sqrt{a_{11}} & \frac{1}{\sqrt{a_{11}}}\boldsymbol{w}^* \\ \mathbf{0} & I \end{bmatrix} = R_1^* A_1 R_1.$$

Now we can iterate on the inner matrix A_1. Note that the (1,1) entry of $K - \boldsymbol{w}\boldsymbol{w}^*/a_{11}$ has to be positive since A was positive definite and R_1 is nonsingular. This guarantees that $A_1 = R_1^{-*} A R_1^{-1}$ is positive definite and therefore has positive diagonal entries.

The iteration yields

$$A_1 = R_2^* A_2 R_2$$

so that

$$A = R_1^* R_2^* A_2 R_2 R_1,$$

and eventually

$$A = \underbrace{R_1^* R_2^* \dots R_m^*}_{=R^*} \underbrace{R_m \dots R_2 R_1}_{=R},$$

the *Cholesky factorization* of A. Note that R is upper triangular and its diagonal is positive (because of the square roots). Thus we have proved

Theorem 7.2 *Every Hermitian positive definite matrix A has a unique Cholesky factorization $A = R^*R$ with R an upper triangular matrix with positive diagonal entries.*

Example Consider

$$A = \begin{bmatrix} 4 & 2 & 4 \\ 2 & 5 & 6 \\ 4 & 6 & 9 \end{bmatrix}.$$

Cholesky factorization as explained above yields the sequence

$$\begin{aligned} A &= \begin{bmatrix} 2 & 0 & 0 \\ 1 & 1 & 0 \\ 2 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 4 & 4 \\ 0 & 4 & 5 \end{bmatrix} \begin{bmatrix} 2 & 1 & 2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 2 & 0 & 0 \\ 1 & 1 & 0 \\ 2 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 2 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & 2 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \end{aligned}$$

Distributing 2 copies of the identity matrix in the middle we arrive at

$$R^* = \begin{bmatrix} 2 & 0 & 0 \\ 1 & 2 & 0 \\ 2 & 2 & 1 \end{bmatrix}.$$

Algorithm (Cholesky Factorization)

initialize $R =$ upper triangular part of A (including the diagonal)

for $k = 1 : m$

for $j = k + 1 : m$

$R(j, j : m) = R(j, j : m) - R(k, j : m)\frac{R(k,j)}{R(k,k)}$

end

$R(k, k : m) = R(k, k : m)/\sqrt{R(k, k)}$

end

The operations count for the Cholesky algorithm can be computed as

$$\sum_{k=1}^{m} \left[(m - k + 1) + \sum_{j=k+1}^{m} (2(m - j + 1) + 1) \right] = \frac{1}{3}m^3 + m^2 - \frac{1}{m}.$$

Thus the count is $\mathcal{O}(\frac{1}{3}m^3)$, which is half the amount needed for the LU factorization.

Another nice feature of Cholesky factorization is that it is *always stable* — even without pivoting.

Remark The simplest (and cheapest) way to test whether a matrix is positive definite is to run Cholesky factorization it. If the algorithm works it is, if not, it isn't.

The solution of $A\boldsymbol{x} = \boldsymbol{b}$ with Hermitian positive definite A is given by:

1. Compute Cholesky factorization $A = R^* R$.
2. Solve the lower triangular system $R^* \boldsymbol{y} = \boldsymbol{b}$ for $\boldsymbol{y}$.
3. Solve the upper triangular system $R\boldsymbol{x} = \boldsymbol{y}$ for $\boldsymbol{x}$.