7 Gaussian Elimination and LU Factorization

In this final section on matrix factorization methods for solving $Ax = b$ 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.,

$$L_{m-1}L_{m-2} \ldots L_2 L_1 A = U,$$

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,

$$\tilde{L}A = U \iff A = LU,$$

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

7.1 Why Would We Want to Do This?

Consider the system $Ax = b$ with LU factorization $A = LU$. Then we have

$$LUx = b.$$ 

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

1. Solve the lower triangular system $Ly = b$ for $y$ by forward substitution.
2. Solve the upper triangular system $Ux = y$ for $x$ 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

$$AX = B \iff LUX = B,$$

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 $O\left(\frac{2}{3}m^3\right)$, 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 $L_2$ 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

$$L_1A = \begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -4 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 2 & 3 & 5 \\ 4 & 6 & 8 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 3 \\ 0 & 2 & 4 \end{bmatrix}.$$

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):

$$L_2(L_1A) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -2 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 3 \\ 0 & 2 & 4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 3 \\ 0 & 0 & -2 \end{bmatrix} = U.$$

Now $L = (L_2L_1)^{-1} = L_1^{-1}L_2^{-1}$ with

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

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

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

so that

$$L_k = \begin{bmatrix} 1 \\ \vdots \\ \ell_k \\ \vdots \\ \ell_{mk} \end{bmatrix}.$$

Now, let

$$\ell_k = \begin{bmatrix} 0 \\ \vdots \\ \ell_{k+1,k} \\ \vdots \\ \ell_{mk} \end{bmatrix}.$$
Then $L_k = I - \ell_k e_k^*$, and therefore

$$
\begin{aligned}
(I - \ell_k e_k^*) (I + \ell_k e_k^*) &= I - \ell_k e_k^* \ell_k e_k^* = I, \\
\end{aligned}
$$

since the inner product $e_k^* \ell_k = 0$ because the only nonzero entry in $e_k$ (the 1 in the $k$-th position) does not “hit” any nonzero entries in $\ell_k$ which start in the $k + 1$-st position.

So, for any $k$ we have

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

as claimed.

In addition,

$$
L_k^{-1} L_{k+1}^{-1} = (I + \ell_k e_k^*) (I + \ell_{k+1} e_{k+1}^*) = I + \ell_k e_k^* + \ell_{k+1} e_{k+1}^* + \ell_k e_k^* \ell_{k+1} e_{k+1}^* = 0
$$

and in general we have

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

We can summarize the factorization in

**Algorithm** (LU Factorization)

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

for $k = 1 : m - 1$

for $j = k + 1 : m$

61
\[ L(j, k) = U(j, k) / U(k, k) \]
\[ U(j, k : m) = U(j, k : m) - L(j, k) U(k, k : m) \]

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 \( O(\frac{2}{3}m^3) \).

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

\[
A = \begin{bmatrix}
1 & 1 & 1 \\
2 & 2 & 5 \\
4 & 6 & 8
\end{bmatrix} \quad \implies \quad L_1 A = \begin{bmatrix}
1 & 1 & 1 \\
0 & 0 & 3 \\
0 & 2 & 4
\end{bmatrix}.
\]

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 \( \varepsilon \). 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 \( b \) is given as

\[
b = \begin{bmatrix}
1 \\
0 \\
0
\end{bmatrix}
\]

and we attempt to solve \( Ax = b \) via

62
1. Solve $Ly = b$.

2. Solve $Ux = y$.

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

$$
\hat{U} = \begin{bmatrix}
1 & 1 & 1 \\
0 & \varepsilon & 3 \\
0 & 0 & -\frac{\varepsilon}{\varepsilon}
\end{bmatrix}
$$

and $\hat{L} = L$,

which leads to

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

In fact, the product is significantly different from $A$. Thus, using $\hat{L}$ and $\hat{U}$ 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 $\hat{L}$ and $\hat{U}$ with the above right-hand side $b$, then we obtain

$$
\hat{x} = \begin{bmatrix}
\frac{11}{2} - \frac{2}{3}\varepsilon \\
-2 \\
\frac{2}{3} - \frac{2}{3}\varepsilon
\end{bmatrix} \approx \begin{bmatrix}
\frac{11}{2} \\
-2 \\
\frac{7}{3}
\end{bmatrix}.
$$

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

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

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

### 7.2 Pivoting

**Example** The breakdown of the algorithm in our earlier example with

$$
L_1A = \begin{bmatrix}
1 & 1 & 1 \\
0 & 0 & 3 \\
0 & 2 & 3
\end{bmatrix}
$$

can be prevented by simply *swapping rows*, i.e., instead of trying to apply $L_2$ to $L_1A$ we first create

$$
P L_1A = \begin{bmatrix}
1 & 0 & 0 \\
0 & 0 & 1 \\
0 & 1 & 0
\end{bmatrix} L_1A = \begin{bmatrix}
1 & 1 & 1 \\
0 & 2 & 3 \\
0 & 0 & 3
\end{bmatrix}
$$

— and are done.
More generally, stability problems can be avoided by swapping rows before applying $L_k$, i.e., we perform

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

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

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

Now we need to pick the second pivot element. For sufficiently small $\varepsilon$ (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

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

$$P = P_2P_1 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}.$$

Thus, this approach was backward stable.

Finally, since we have the factorization $PA = LU$, we can solve the linear system $Ax = b$ as

$$PAx = Pb \iff LUx = Pb,$$

and apply the usual two-step procedure

1. Solve the lower triangular system $Ly = Pb$ for $y$.
2. Solve the upper triangular system $Ux = y$ for $x$.

This yields

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

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

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

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

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

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 $P_2 = I$.  

65
**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 $|U(i, k)|$
  - $U(k, k : m) \leftarrow U(i, k : m)$
  - $L(k, 1 : k - 1) \leftarrow L(i, 1 : k - 1)$
  - $P(k,:) \leftarrow P(i,:)$
  - 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 $\rho \leq 2^{m-1}$. 

66