4 LU-factorization with pivoting

Consider the solution of the linear system of equations

$$A\mathbf{x} = \mathbf{b}, \tag{1}$$

where

$$A = \begin{bmatrix} 0 & 1 \\ -1 & 1 \end{bmatrix}, \qquad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \qquad \mathbf{b} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}. \tag{2}$$

The matrix has determinant 1. It therefore is nonsingular and the linear system of equations (1) has a unique solution. In fact, it is easy to verify that the solution is $\mathbf{x} = [2, 3]^T$.

Now apply Algorithm 3.1 of Chapter 3 to determine the LU-factorization of the matrix A. You will see that the algorithm breaks down already for $k = 1$, because $u_{1,1} = 0$ causes division by zero. We conclude that LU-factorization as described by Algorithm 3.1 cannot be applied to the solution of all linear systems of equations with a nonsingular matrix.

As you already may have noticed, the system of equations (1) can be solved quite easily without LU-factorization by applying backsubstitution in an appropriate order. We first determine the solution component x_2 from the first equation of (1), $1 \cdot x_2 = 2$, and compute x_1 from the second equation, $-1 \cdot x_1 + 1 \cdot x_2 = 1$. Note that your MATLAB/Octave function *backsubst* from Exercise 3.5 of Chapter 3 cannot be applied to carry out this variant of backsubstitution.

There is a simple remedy that allows the application of your MATLAB function *backsubst* to the solution of (1), namely to interchange the rows of the matrix and right-hand side before solution. This gives the matrix and right-hand side vector

$$\tilde{A} = \begin{bmatrix} -1 & 1 \\ 0 & 1 \end{bmatrix}, \qquad \tilde{\mathbf{b}} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}.$$

The matrix $\tilde{A}$ is upper triangular and therefore the linear system of equations

$$\tilde{A}\mathbf{x} = \tilde{\mathbf{b}},$$

can be solved by standard backsubstitution. Note that interchanging rows of a linear system of equations does not change the solution. You have used this fact when you have solved linear systems of equations by Gaussian elimination with an ad-hoc elimination order.

The interchange of rows is commonly referred to as *pivoting* and the divisors $u_{k,k}$ in Algorithm 3.1 as *pivot elements* or simply *pivots*. Pivoting has to be employed whenever a pivot $u_{k,k}$ in Algorithm 3.1 vanishes. One can show that all linear systems of equations with a square nonsingular matrix can be solved by Gaussian elimination with pivoting, or equivalently, by LU-factorization with pivoting. The factor L is not lower triangular when pivoting is employed.

Example 1. The function *lu* in MATLAB and Octave determines the LU-factorization of a matrix A with pivoting. When applied to the matrix (2), it produces

$$L = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \qquad U = \begin{bmatrix} -1 & 1 \\ 0 & 1 \end{bmatrix}.$$

Thus, L is not lower triangular. The matrix L can be thought of as a lower triangular matrix with the rows interchanged. More details on the function *lu* are provided in Exercise 4.1. □

In exact arithmetic, we can compute the LU-factorization of any nonsingular 2×2 matrix with a nonvanishing $(1, 1)$ element. However, in floating point arithmetic, the computed factors might not be accurate.

Example 2. Apply a MATLAB or Octave implementation of Algorithm 3.1 on a standard PC to the matrix

$$A = \begin{bmatrix} 10^{-20} & 1 \\ 1 & 1 \end{bmatrix}.$$

We obtain the factors

$$L = \begin{bmatrix} 1 & 0 \\ 10^{20} & 1 \end{bmatrix}, \qquad U = \begin{bmatrix} 10^{-20} & 1 \\ 0 & -10^{20} \end{bmatrix},$$

where the entry $u_{2,2}$ of U is computed as $1 - 10^{20} \cdot 1$, which is stored as -10^{20}.

The relative error in $u_{2,2}$ is

$$\frac{-10^{20} - (1 - 10^{20})}{1 - 10^{20}} \approx 10^{-20},$$

which is tiny; however, the absolute error in $u_{2,2}$, given by $-10^{20} - (1 - 10^{20}) = -1$, is not. Multiplying L and U yields

$$LU = \begin{bmatrix} 10^{-20} & 1 \\ 1 & 0 \end{bmatrix},$$

which is not close to the matrix A. The error in the matrix LU is caused by round-off errors during the computations of $u_{2,2}$ and by the fact that there are intermediate quantities of very large magnitude formed during the computations. The difficulties are caused by the large factor -10^{20}. □

Example 3. A row interchange in the matrix of the above example remedies the accuracy problems encountered. Let

$$\tilde{A} = \begin{bmatrix} 1 & 1 \\ 10^{-20} & 1 \end{bmatrix}.$$

Then a MATLAB or Octave implementation of Algorithm 3.1 determines the LU-factorization

$$\tilde{L} = \begin{bmatrix} 1 & 0 \\ 10^{-20} & 1 \end{bmatrix}, \qquad \tilde{U} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}.$$

Multiplication of $\tilde{L}$ and $\tilde{U}$ gives the matrix $\tilde{A}$ in floating point arithmetic. The high accuracy depends on that the matrices $\tilde{L}$ and $\tilde{U}$ do not contain large entries. □

The above example illustrates that row interchange can improve the accuracy significantly in the computed LU-factors, also when in exact arithmetic no row interchanges are required. The row interchange gave a lower triangular factor $\tilde{L}$ with entries of significantly smaller magnitude than the magnitude of the entries of the lower triangular matrix L of Example 2. Looking at Algorithm 3.1, we see that the magnitude of all subdiagonal entries $\ell_{j,k}$ of L is bounded by one, if $|u_{k,k}|$ is larger than or equal to

$$\max\{|u_{k+1,k}|, |u_{k+2,k}|, \ldots, |u_{n,k}|\}$$

for $k = 1, 2, \ldots, n-1$. This suggests that we interchange the rows of the U-matrix during the factorization process to achieve that

$$\frac{|u_{j,k}|}{|u_{k,k}|} \leq 1, \qquad j = k+1, k+2, \ldots, n. \tag{3}$$

LU-factorization with the rows (re)ordered so that (3) holds is commonly referred as *LU-factorization with partial pivoting*. There are also other pivoting strategies. We will comment on some of them in Exercises 4.5 and 4.6.

When we solve a linear system of equations and interchange the rows of the matrix, we also need to interchange the corresponding rows of the right-hand side in order to obtain the correct solution. LU-factorization with partial pivoting may be carried out without access to the right-hand side. We have to keep track of the row interchanges carried out during the factorization, so that we can apply them to the right-hand side when the latter is available. We do this by applying all the row interchanges carried out during the LU-factorization to the identity matrix. This gives a so-called *permutation matrix* P. Permutation matrices are matrices obtained be interchanging rows or columns in the identity matrix. They have precisely one nonvanishing entry (one) in each row and column. A permutation matrix is an example of a matrix, whose transpose is its inverse.

Example 4. Let

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

Then its transpose is given by

$$P^T = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

and it is easy to verify that $P^TP = PP^T = I$. □

In order to illustrate LU-factorization with partial pivoting, we apply the method to the matrix

$$A = \begin{bmatrix} 2 & 1 & 1 & 0 \\ 4 & 3 & 3 & 1 \\ 8 & 7 & 9 & 5 \\ 6 & 7 & 9 & 8 \end{bmatrix},$$

which we factored in Chapter 3 without partial pivoting pivoting. We denote the 4×4 permutation matrix, which keeps track of the row interchanges by P; it is initialized as the identity matrix and so is the lower triangular matrix L in the factorization. We set $U = A$. The first step of factorization process is to determine the entry of largest magnitude in column 1. This is the entry 8 in row 3. We therefore swap rows 1 and 3 of the matrices U and P to obtain,

$$U = \begin{bmatrix} 8 & 7 & 9 & 5 \\ 4 & 3 & 3 & 1 \\ 2 & 1 & 1 & 0 \\ 6 & 7 & 9 & 8 \end{bmatrix}, \qquad P = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

We then subtract suitable multiples of row 1 of U from rows 2 through 4, to create zeros in the first column of U below the diagonal. The multiples are stored in the subdiagonal entries of the first column of the matrix L. This gives the matrices

$$U = \begin{bmatrix} 8 & 7 & 9 & 5 \\ 0 & -1/2 & -3/2 & -3/2 \\ 0 & -3/4 & -5/4 & -5/4 \\ 0 & 7/4 & 9/4 & 1/4 \end{bmatrix}, \qquad L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1/2 & 1 & 0 & 0 \\ 1/4 & 0 & 1 & 0 \\ 3/4 & 0 & 0 & 1 \end{bmatrix}, \qquad P = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

We now proceed to the entries 2 through 4 of column 2 of U, and note that the last entry is of largest magnitude. Hence, we interchange rows 2 and 4 of the matrices U and P. We also need to swap the subdiagonal entries of rows 2 and 4 of L. This gives

$$U = \begin{bmatrix} 8 & 7 & 9 & 5 \\ 0 & 7/4 & 9/4 & 1/4 \\ 0 & -3/4 & -5/4 & -5/4 \\ 0 & -1/2 & -3/2 & -3/2 \end{bmatrix}, \qquad L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 3/4 & 1 & 0 & 0 \\ 1/4 & 0 & 1 & 0 \\ 1/2 & 0 & 0 & 1 \end{bmatrix}, \qquad P = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}.$$

We subtract suitable multiples of row 2 from rows 3 and 4 to obtain zeros in the second column of U below the diagonal. The multiples are stored in the subdiagonal entries of the second column of the matrix L. This yields the matrices

$$U = \begin{bmatrix} 8 & 7 & 9 & 5 \\ 0 & 7/4 & 9/4 & 1/4 \\ 0 & 0 & -2/7 & 4/7 \\ 0 & 0 & -6/7 & -2/7 \end{bmatrix}, \qquad L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 3/4 & 1 & 0 & 0 \\ 1/4 & -3/7 & 1 & 0 \\ 1/2 & -2/7 & 0 & 1 \end{bmatrix}, \qquad P = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}.$$

We turn to the last 2 entries of column 3 of U, and notice that the last entry has the largest magnitude. Hence, we swap rows 3 and 4 of the matrices U and P, and we interchange the subdiagonal entries in rows 3 and 4 of L to obtain

$$U = \begin{bmatrix} 8 & 7 & 9 & 5 \\ 0 & 7/4 & 9/4 & 1/4 \\ 0 & 0 & -6/7 & -2/7 \\ 0 & 0 & -2/7 & 4/7 \end{bmatrix}, \qquad L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 3/4 & 1 & 0 & 0 \\ 1/2 & -2/7 & 1 & 0 \\ 1/4 & -3/7 & 0 & 1 \end{bmatrix}, \qquad P = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}.$$

The final step of the factorization is to subtract 1/3 times row 3 of U from row 4 and store 1/3 in the subdiagonal entry of column 3 of L. This yields

$$U = \begin{bmatrix} 8 & 7 & 9 & 5 \\ 0 & 7/4 & 9/4 & 1/4 \\ 0 & 0 & -6/7 & -2/7 \\ 0 & 0 & 0 & 2/3 \end{bmatrix}, \qquad L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 3/4 & 1 & 0 & 0 \\ 1/2 & -2/7 & 1 & 0 \\ 1/4 & -3/7 & 1/3 & 1 \end{bmatrix}, \qquad P = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

and we can verify that

$$PA = LU. \tag{4}$$

When applying the factorization (4) to the solution of a linear system of equations

$$A\mathbf{x} = \mathbf{b}, \tag{5}$$

we first multiply the right-hand side $\mathbf{b}$ by the permutation matrix P to obtain

$$LU\mathbf{x} = PA\mathbf{x} = P\mathbf{b}.$$

We then solve $L\mathbf{y} = P\mathbf{b}$ by forward substitution and $U\mathbf{x} = \mathbf{y}$ by back substitution.

Assume that we apply LU-factorization with partial pivoting to an $n \times n$ matrix A and find that for some k all entries $u_{j,k}$, $j = k, k+1, \ldots, n$ of U vanish. Then pivoting does not help us to proceed and LU-factorization with partial pivoting breaks down. One can show that this situation only can occur when A is singular. Thus, LU-factorization with partial pivoting can be applied to solve all linear systems of equations with a nonsingular matrix.

Exercises

Exercise 4.1: (a) Apply the MATLAB/Octave function *lu* to the matrix (2) by using the call $[L, U] = lu(A)$. What is L? Read the help file for a description of L. (b) Use instead the function call $[L, U, P] = lu(A)$. What are L, U, and P?

Exercise 4.2: Determine the LU-factorization with partial pivoting of the matrix

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

by hand computations. □

Exercise 4.3: Solve $A\mathbf{x} = \mathbf{b}$, where A is the matrix in Exercise 4.2 and $\mathbf{b} = [3, 5]^T$, by using the LU-factorization from Exercise 4.2. □

Exercise 4.4: The storage of the permutation matrix P is a a bit clumsy. After all, we only need to keep track of the row interchanges we have carried out and this does not require storage of an $n \times n$ matrix with most entries zero. Describe a more compact way to represent the information. □

Exercise 4.5: If an entry $u_{k,k}$ in Algorithm 3.1 vanishes, then we could interchange columns instead of rows to obtain a nonvanishing replacement for $u_{k,k}$. Mention a difficulty with this approach that is not present when interchanging rows. Hint: How does the solution change when we interchange rows and columns? □

Exercise 4.6: In LU-factorization with complete pivoting rows and columns are interchanged so as to maximize the denominator $u_{k,k}$ in each step. Discuss possible advantages and disadvantages of complete pivoting?