Chapter 4. Numerical Solution of Systems of Linear Algebraic Equations

Systems of simultaneous linear equations occur in solving problems in a wide variety of disciplines, including Mathematics, Statistics, physical, biological and social sciences, engineering, business and many more. They arise directly in solving real-world problems, and they also occur as part of the solution process for other problems. Numerical solutions of boundary value problems and initial boundary value problems for differential equations are a rich source of linear systems, especially large-size ones.

In this chapter, we will examine the following problem: given a matrix $A \in \mathbb{R}^{n\times n}$ and a vector $b \in \mathbb{R}^n$, find $x \in \mathbb{R}^n$ such that

$$Ax \ = \ b.$$

There are two types of methods for the solution of algebraic linear systems:

- *direct (exact)* methods, that provide a solution in a finite number of steps (e.g., Cramer, Gaussian elimination, factorizations);
- *iterative* methods, which approximate the solution by a sequence converging to it (e.g., Jacobi, Gauss-Seidel, SOR).

1 Direct Methods

1.1 Gaussian Elimination

A linear system is easy to solve when the matrix of the system is *triangular*:

Definition 1.1. *A matrix $A = [a_{ij}]_{i,j=\overline{1,n}}$ is called*

- ***upper triangular**, if $a_{ij} \ = \ 0, \forall i \ > \ j$,*

$$A \ = \ \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ & a_{22} & \dots & a_{2n} \\ & & \ddots & \vdots \\ 0 & & & a_{nn} \end{bmatrix}, \tag{1.1}$$

- ***lower triangular***, *if* $a_{ij} = 0, \forall i < j$,

$$A = \begin{bmatrix} a_{11} & & & 0 \\ a_{21} & a_{22} & & \\ \vdots & & \ddots & \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix}, \tag{1.2}$$

- ***diagonal***, *if it is both upper and lower triangular,* $a_{ij} = 0, \forall i \neq j$,

$$A = \begin{bmatrix} a_{11} & & & 0 \\ & a_{22} & & \\ & & \ddots & \\ 0 & & & a_{nn} \end{bmatrix}. \tag{1.3}$$

Remark 1.2. The determinant of an upper or lower triangular matrix is equal to the product of its diagonal elements

$$\det(A) = a_{11}a_{22}\dots a_{nn}.$$

So, an upper or lower triangular matrix is nonsingular if and only if all of its diagonal entries are nonzero.

Example 1.3. Solve the triangular systems

a)

$$\begin{bmatrix} 2 & 4 & 2 \\ 0 & -1 & 1 \\ 0 & 0 & -1 \end{bmatrix} x = \begin{bmatrix} 8 \\ 0 \\ -1 \end{bmatrix}, \tag{1.4}$$

b)

$$\begin{bmatrix} 1 & 0 & 0 \\ 1/2 & 1 & 0 \\ 1/2 & 1 & 1 \end{bmatrix} x = \begin{bmatrix} 8 \\ 4 \\ 3 \end{bmatrix}. \tag{1.5}$$

Solution.

a) The upper triangular system is

$$\begin{cases} 2x_1 + 4x_2 + 2x_3 = 8 \\ \quad -x_2 + x_3 = 0 \\ \quad\quad -x_3 = -1 \end{cases}$$

We start from the bottom (the last equation) and solve recursively for each unknown:

$$\begin{aligned} x_3 &= \frac{-1}{-1} = 1, \\ x_2 &= \frac{1}{-1}\big(0 - x_3\big) = 1, \\ x_1 &= \frac{1}{2}\big(8 - 4x_2 - 2x_3\big) = 1. \end{aligned}$$

We found the solution

$$x = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = [1\ 1\ 1]^T.$$

b) For the lower triangular system:

$$\begin{cases} x_1 = 8 \\ 1/2x_1 + x_2 = 4 \\ 1/2x_1 + x_2 + x_3 = 3 \end{cases}$$

we start from the top and solve each equation going down:

$$\begin{aligned} x_1 &= \frac{8}{1} = 8, \\ x_2 &= \frac{1}{1}\Big(4 - \frac{1}{2}x_1\Big) = 0, \\ x_3 &= \frac{1}{1}\Big(3 - \frac{1}{2}x_1 - x_2\Big) = -1. \end{aligned}$$

So the solution is

$$x \;=\; \begin{bmatrix} 8 \\ 0 \\ -1 \end{bmatrix} \;=\; [8 \quad 0 \quad -1]^T.$$

■

So, in general, for a nonsingular upper triangular matrix U, the system $Ux = b$ is easily solved by **backward substitution**:

$$\begin{aligned} x_n &= \frac{b_n}{u_{nn}}, \\ x_i &= \frac{1}{u_{ii}}\Big(b_i - \sum_{j=i+1}^{n} u_{ij}x_j\Big),\ i = \overline{n-1, 1} \end{aligned} \tag{1.6}$$

and if the nonsingular matrix L is lower triangular, then the system $Lx = b$ is solved by **forward substitution**:

$$\begin{aligned} x_1 &= \frac{b_1}{l_{11}}, \\ x_i &= \frac{1}{l_{ii}}\Big(b_i - \sum_{j=1}^{i-1} l_{ij}x_j\Big),\ i = \overline{2, n}. \end{aligned} \tag{1.7}$$

Gaussian elimination is a procedure for transforming a system into an equivalent (upper) triangular one, by doing the following elementary row operations:

- multiplying a row (equation) by a constant $\lambda \neq 0$,

$$(\lambda R_i) \;\to\; (R_i),$$

- multiplying a row by a constant $\lambda \neq 0$ and adding it to another row,

$$(R_i + \lambda R_j) \;\to\; (R_i),$$

- interchanging (permuting) two rows,

$$(R_i) \;\longleftrightarrow\; (R_j).$$

All these elementary operations are performed on the *augmented (extended)* matrix of the system

$$\widetilde{A} \;=\; [A \mid b] \;=\; \left[\begin{array}{cccc|c} a_{11} & a_{12} & \dots & a_{1n} & a_{1,n+1} \\ a_{21} & a_{22} & \dots & a_{2n} & a_{2,n+1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} & a_{n,n+1} \end{array}\right], \tag{1.8}$$

where $a_{i,n+1} = b_i,\ i = \overline{1,n}$.
Gaussian elimination goes as follows:
Assuming $a_{11} \neq 0$, at the first step, we eliminate (make 0) the coefficients of x_1 from every row below, i.e., every $R_j, j = \overline{2,n}$, using a_{11}, i.e. by

$$\Big(R_j - \frac{a_{j1}}{a_{11}} R_1\Big) \;\to\; (R_j).$$

Then we proceed the same for the coefficients of each $x_i, i = \overline{2,n-1}, j = \overline{i+1,n}$. This way we obtain a finite sequence of augmented matrices

$$\widetilde{A}^{(1)},\ \widetilde{A}^{(2)},\ \dots,\ \widetilde{A}^{(n)},$$

where $\widetilde{A}^{(1)} \;=\; \widetilde{A}$ and (at step k) $\widetilde{A}^{(k)} \;=\; \left[a_{ij}^{(k)}\right]$ obtained by

$$\Big(R_i - \frac{a_{i,k-1}^{(k-1)}}{a_{k-1,k-1}^{(k-1)}} R_{k-1}\Big) \;\to\; (R_i).$$

We denoted by $a_{ij}^{(l)}$ the (i,j) entry at step l. The system corresponding to the augmented matrix $\widetilde{A}^{(k)}$ is equivalent to the original linear system and in it, the variable x_{k-1} was eliminated from the equations $E_k, E_{k+1}, \dots, E_n$. Then the system corresponding to $\widetilde{A}^{(n)}$ is an equivalent upper triangular one:

$$\left\{\begin{array}{ccccccccc} a_{11}^{(1)}x_1 & + & a_{12}^{(1)}x_2 & + & \dots & + & a_{1n}^{(1)}x_n & = & a_{1,n+1}^{(1)} \\ & + & a_{22}^{(2)}x_2 & + & \dots & + & a_{2n}^{(2)}x_n & = & a_{2,n+1}^{(2)} \\ & & & \ddots & & & \vdots & & \\ & & & & & & a_{nn}^{(n)}x_n & = & a_{n,n+1}^{(n)} \end{array}\right., \tag{1.9}$$

which is solved by backward substitution (1.6).

Of course, in all this we need $a_{ii}^{(i)} \;\neq\; 0$.The element $a_{ii}^{(i)}$ is called **pivot**. If at any time during the elimination process, we find $a_{kk}^{(k)} \;=\; 0$, then we look further down in that column for a pivot,

i.e., we interchange rows

$$(R_k) \longleftrightarrow (R_p),$$

where p is the smallest integer $k+1 \leq p \leq n$ with $a_{pk}^{(k)} \neq 0$.
In fact, in practice, when implementing Gaussian elimination, pivoting is necessary even if the pivot is *not* zero, but small, compared to the rest of the elements in that column. That is because such a pivot can produce substantial rounding errors and even cancellations. This can be fixed by doing several types of *pivoting*.

- We can choose the pivot to be the largest element (in absolute value) in that column, below the main diagonal, i.e.

$$a_{pk}^{(k)} = \max_{k \leq l \leq n} \left| a_{lk}^{(k)} \right|. \tag{1.10}$$

  This is called **partial pivoting (maximal pivoting on columns)**.

- We can do **scaled pivoting on columns**: First, we define a scaling factor for each row

$$s_i = \max_{j=\overline{1,n}} |a_{ij}| \text{ or } s_i = \sum_{j=1}^{n} |a_{ij}|, \ i = \overline{1,n}.$$

  If there exists an i such that $s_i = 0$, then the matrix is singular. For a nonsingular matrix, we use the scaling factor to choose the pivot. At each step i, we find the smallest p, $i \leq p \leq n$ such that

$$\frac{|a_{pi}|}{s_i} = \max_{1 \leq j \leq n} \frac{|a_{ji}|}{s_j} \Big| \tag{1.11}$$

  and then interchange rows $(R_i) \longleftrightarrow (R_p)$ so the pivot is a_{pi}. This ensures the fact that the maximal element in each column has the relative size 1, before we compare and interchange rows. Also, dividing by the scaling factor does not produce any extra rounding errors.

- The third method is **total (maximal) pivoting**. At each step k, we find

$$|a_{pq}| = \max\{|a_{ij}|, i, j = \overline{k,n}\} \tag{1.12}$$

  and interchange both the rows and the columns,

$$(R_k) \longleftrightarrow (R_p), \ (C_k) \longleftrightarrow (C_q).$$

But then we have to keep track of the columns (unknowns) interchanges.

Remark 1.4. If A is singular of rank $p-1$, then at step p we get

$$\widetilde{A}^{(p)} = \left[\begin{array}{ccccccc|c} a_{11}^{(1)} & a_{12}^{(1)} & \dots & a_{1,p-1}^{(1)} & a_{1p}^{(1)} & \dots & a_{1n}^{(1)} & a_{1,n+1}^{(1)} \\ 0 & a_{22}^{(2)} & \dots & a_{2,p-1}^{(2)} & a_{2p}^{(2)} & \dots & a_{2n}^{(2)} & a_{2,n+1}^{(2)} \\ \vdots & & \ddots & \vdots & \vdots & & \vdots & \vdots \\ \vdots & & & a_{p-1,p-1}^{(p-1)} & a_{p-1,p}^{(p-1)} & & a_{p-1,n}^{(p-1)} & a_{p-1,n+1}^{(p-1)} \\ \vdots & & & & 0 & \dots & 0 & a_{p,n+1}^{(p)} \\ \vdots & & & & & \ddots & \vdots & \vdots \\ 0 & \dots & \dots & \dots & \dots & \dots & 0 & a_{n,n+1}^{(n)} \end{array}\right].$$

So, if $a_{i,n+1}^{(i)} = b_i^{(i)} = 0$, for *all* $i = p, p+1, \dots, n$, then the system is compatible, but undetermined (i.e., it has an infinite number of solutions), otherwise, the system is incompatible (no solution). Thus, Gaussian elimination can also be used to discuss the solvability of the linear system.

Example 1.5. Solve the system

$$\left\{\begin{array}{rcrcrcr} x_1 & - & x_2 & + & x_3 & = & -1 \\ -2x_1 & + & 2x_2 & + & x_3 & = & 2 \\ -3x_1 & - & x_2 & + & 5x_3 & = & -5 \end{array}\right.,$$

by Gaussian elimination with different types of pivoting.

Solution. The augmented matrix of the system is

$$\widetilde{A} = \left[\begin{array}{ccc|c} 1 & -1 & 1 & -1 \\ -2 & 2 & 1 & 2 \\ -3 & -1 & 5 & -5 \end{array}\right],$$

Partial pivoting

On the first column, the largest element in absolute value is -3, so that will be the pivot. Thus, first we interchange $(R_1) \longleftrightarrow (R_3)$. We get

$$\widetilde{A} \sim \left[\begin{array}{ccc|c} \boxed{-3} & -1 & 5 & -5 \\ -2 & 2 & 1 & 2 \\ 1 & -1 & 1 & -1 \end{array}\right] \sim \left[\begin{array}{ccc|c} -3 & -1 & 5 & -5 \\ 0 & 8/3 & -7/3 & 16/3 \\ 0 & -4/3 & 8/3 & -8/3 \end{array}\right] \begin{array}{r} (-2/3\ R_1 + R_2) \ \to \ (R_2) \\ (1/3\ R_1 + R_3) \ \to \ (R_3) \end{array}$$

Further, we have

$$\widetilde{A} \sim \left[\begin{array}{ccc|c} -3 & -1 & 5 & -5 \\ 0 & \boxed{8/3} & -7/3 & 16/3 \\ 0 & -4/3 & 8/3 & -8/3 \end{array}\right] \sim \left[\begin{array}{ccc|c} -3 & -1 & 5 & -5 \\ 0 & 8/3 & -7/3 & 16/3 \\ 0 & 0 & 3/2 & 0 \end{array}\right] \quad (\tfrac{1}{2}\, R_2 + R_3) \;\to\; (R_3)$$

Now we solve by back substitution (1.6), to get

$$x \;=\; [1 \;\; 2 \;\; 0]^T.$$

Scaled partial pivoting

We compute the scaling factors using sums on each row. At the first step $k = 1$, we get

$$\begin{aligned} s &= [3,\ 5,\ 9] \\ \left[\frac{|a_{j,1}|}{s_j}\right] &= [1/3,\ 2/5,\ 3/9] \;=\; [5/15,\ 6/15,\ 5/15],\ j = 1, 2, 3. \end{aligned}$$

The maximum of the three fractions is the second, so $p = 2$. We interchange $(R_1) \longleftrightarrow (R_2)$ and make zeros below it.

$$\widetilde{A} \sim \left[\begin{array}{ccc|c} \boxed{-2} & 2 & 1 & 2 \\ 1 & -1 & 1 & -1 \\ -3 & -1 & 5 & -5 \end{array}\right] \sim \left[\begin{array}{ccc|c} -2 & 2 & 1 & 2 \\ 0 & 0 & 3/2 & 0 \\ 0 & -4 & 7/2 & -8 \end{array}\right] \begin{array}{r} (1/2\, R_1 + R_2) \;\to\; (R_2) \\ (-3/2\, R_1 + R_3) \;\to\; (R_3) \end{array}$$

At step $k = 2$, obviously, $p = 3$, so we interchange $(R_2) \longleftrightarrow (R_3)$,

$$\widetilde{A} \sim \left[\begin{array}{ccc|c} -2 & 2 & 1 & 2 \\ 0 & -4 & 7/2 & -8 \\ 0 & 0 & 3/2 & 0 \end{array}\right]$$

and we are done. By back substitution we get the (obviously, same) solution

$$x \;=\; [1 \;\; 2 \;\; 0]^T.$$

Total pivoting

At step $k = 1$, since

$$\max_{i,j=\overline{1,3}} |a_{ij}| \;=\; 5 \;=\; |a_{33}|,$$

we interchange both rows and columns, $(R_1) \longleftrightarrow (R_3),\ (C_1) \longleftrightarrow (C_3)$, to get

$$\widetilde{A} \ \sim \ \left[\begin{array}{ccc|c} -3 & -1 & 5 & -5 \\ -2 & 2 & 1 & 2 \\ 1 & -1 & 1 & -1 \end{array}\right] \ \sim \ \left[\begin{array}{ccc|c} 5 & -1 & -3 & -5 \\ 1 & 2 & -2 & 2 \\ 1 & -1 & 1 & -1 \end{array}\right],$$

which is now a system for the *new unknown* $x' \ = \ [x_3 \ x_2 \ x_1]^T$. We proceed to make zeros on the first column below the diagonal.

$$\widetilde{A} \ \sim \ \left[\begin{array}{ccc|c} \boxed{5} & -1 & -3 & -5 \\ 1 & 2 & -2 & 2 \\ 1 & -1 & 1 & -1 \end{array}\right] \ \sim \ \left[\begin{array}{ccc|c} 5 & -1 & -3 & -5 \\ 0 & 11/5 & -7/5 & 3 \\ 0 & -4/5 & 8/5 & 0 \end{array}\right] \quad \begin{array}{l} (-1/5\ R_1 + R_2) \ \to \ (R_2) \\ (-1/5\ R_1 + R_3) \ \to \ (R_3) \end{array}$$

At step $k = 2$,

$$\max_{i,j=2,3} |a_{ij}| \ = \ \frac{11}{5} \ = \ |a_{22}|,$$

so no (row or column) interchanges are necessary. We have

$$\widetilde{A} \ \sim \ \left[\begin{array}{ccc|c} 5 & -1 & -3 & -5 \\ 0 & \boxed{11/5} & -7/5 & 3 \\ 0 & -4/5 & 8/5 & 0 \end{array}\right] \ \sim \ \left[\begin{array}{ccc|c} 5 & -1 & -3 & -5 \\ 0 & 11/5 & -7/5 & 3 \\ 0 & 0 & 12/11 & 12/11 \end{array}\right] \quad (\tfrac{4}{11}\ R_2 + R_3) \ \to \ (R_3)$$

By back substitution, we get

$$x' \ = \ [0 \ \ 2 \ \ 1]^T \text{ and } \ x \ = \ [1 \ \ 2 \ \ 0]^T.$$

■

Remark 1.6.

1. The elements under the main diagonal (which become 0) need not be computed.

2. When pivoting, we do not need to *physically* interchange rows or columns. Just keep one (or two) permutation vector(s) p (q) with $p[i]$ $(q[j])$ meaning that the row (column) p (q) has been interchanged with row (column) i (j). This is especially a good solution if matrices are stored row by row or column by column.

3. Gaussian elimination can be used to find the inverse A^{-1} of a nonsingular matrix. For each $k = \overline{1,n}$, column k of A^{-1} can be found by solving the system $Ax \ = \ e_k$, where $\{e_k\}$ is the canonical basis of $\mathbb{R}^n$, $e_k \ = \ [0 \ 0 \ \ldots \ 0 \ 1 \ 0 \ \ldots \ 0]^T$, with 1 on the kth slot. Alternatively, the

inverse of A can be found by Gaussian elimination on the matrix

$$[A \mid I] \;\sim\; \ldots \sim\; [I \mid A^{-1}].$$

4. Let us assess the computational cost of Gaussian elimination. At step $k = 1$, we perform $n - 1$ divisions, $(n-1)n$ multiplications and $(n-1)n$ additions, so a total of $2n(n-1) + (n-1)$ flops. At step $k = 2$, there are $2(n-1)(n-2) + (n-2)$ flops and so on until step $k = n - 1$. So, the actual elimination process requires

$$\sum_{k=1}^{n-1} \Big[2(n-k)(n-k+1) + (n-k)\Big] \;=\; \sum_{i=1}^{n-1} \Big[2i(i+1) + i\Big] \;=\; \frac{n(n-1)(4n+7)}{6}$$

flops. Back substitution adds another

$$1 + 3 + \cdots + 2n - 1 \;=\; \sum_{i=1}^{2n-1} i - 2\sum_{i=1}^{n-1} i \;=\; n^2$$

flops, for a total of

$$\frac{n(4n^2 + 9n - 7)}{6} \;=\; O\Big(\frac{2}{3}n^3\Big)$$

flops. For comparison, if the determinants in Cramer's rule are computed using expansion by minors, then the operation count is $(n+1)!$. For $n = 10$, Gaussian elimination uses about 805 operations, while Cramer's rule uses around $3,628,800$ operations. This should emphasize the point that Cramer's rule is not a practical computational tool, and that it should be considered as just a theoretical mathematics tool.

1.2 Factorization Based Methods

These are methods using the fact that the matrix of coefficients of a linear system being solved can be *factored (decomposed)* into the product of two triangular matrices.

1.2.1 LU Factorization

Theorem 1.7. *If no row interchanges are necessary in the Gaussian elimination process for solving the system $Ax = b$, then A can be factored as*

$$A \;=\; LU, \tag{1.13}$$

where L and U are lower and upper triangular matrices, respectively. The pair (L, U) is called an LU ***factorization (decomposition)*** *of the matrix A.*

Sketch of Proof. The first step is to partition A as

$$A = \left[\begin{array}{c|ccc} a_{11} & a_{12} & \cdots & a_{1n} \\ \hline a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{array}\right] = \begin{bmatrix} a_{11} & w^* \\ v & A' \end{bmatrix},$$

where v is a column vector of length $n-1$, w^* is a row vector of length $n-1$ and A' is an $(n-1) \times (n-1)$ matrix. Then we can factor A as

$$A = \begin{bmatrix} a_{11} & w^* \\ v & A' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ v/a_{11} & I_{n-1} \end{bmatrix} \begin{bmatrix} a_{11} & w^* \\ 0 & A' - vw^*/a_{11} \end{bmatrix}.$$

The matrix $A' - vw^*/a_{11}$ is called the **Schur complement** of A with respect to a_{11}.
Then, we proceed recursively:

$$A' - vw^*/a_{11} = L'U'.$$

So

$$\begin{aligned} A &= \begin{bmatrix} 1 & 0 \\ v/a_{11} & I_{n-1} \end{bmatrix} \begin{bmatrix} a_{11} & w^* \\ 0 & A' - vw^*/a_{11} \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 \\ v/a_{11} & I_{n-1} \end{bmatrix} \begin{bmatrix} a_{11} & w^* \\ 0 & L'U' \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 \\ v/a_{11} & L' \end{bmatrix} \begin{bmatrix} a_{11} & w^* \\ 0 & U' \end{bmatrix} \end{aligned}$$

until we get a scalar (a 1×1 matrix) that can no longer be partitioned.

□

Remark 1.8.
1. If $A = LU$, then solving the system $Ax = b$ is reduced to solving two triangular systems

$$\begin{aligned} Ly &= b \text{ and} \\ Ux &= y. \end{aligned} \tag{1.14}$$

2. If all that is required is that L be lower and U be upper triangular, then the LU decomposition is *not* unique. We can make it unique by imposing more conditions. For instance, if we require $l_{ii} = 1, i = \overline{1,n}$, we have *Doolittle* factorization and if we impose $u_{ii} = 1, i = \overline{1,n}$, we get the *Crout* factorization. The procedure described in the proof of Theorem 1.7 leads to Doolittle factorization.

3. The matrix $U = [u_{ij}]$ in the Doolittle factorization is the upper triangular matrix obtained by Gaussian elimination,

$$u_{ij} = a_{i,j}^{(i)}, \ i \leq j, \tag{1.15}$$

while $L = [l_{ij}]$ is the matrix of the *multipliers*

$$l_{ij} = m_{ij} = \frac{a_{i,j}^{(j)}}{a_{j,j}^{(j)}}, \ i \geq j. \tag{1.16}$$

4. Examples of cases when no row interchanges are necessary:

- A is **diagonally dominant on rows**,

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|, \ i = \overline{1,n}. \tag{1.17}$$

- A is **positive definite**,

$$x^T A x > 0, \ \forall x \neq 0. \tag{1.18}$$

Example 1.9. Use LU decomposition to solve the system

$$\begin{cases} 2x_1 + 4x_2 + 2x_3 = 8 \\ x_1 + x_2 + 2x_3 = 4 \\ x_1 + x_2 + x_3 = 3 \end{cases}$$

Solution. We have

$$A = \begin{bmatrix} 2 & 4 & 2 \\ 1 & 1 & 2 \\ 1 & 1 & 1 \end{bmatrix} = \left[\begin{array}{c|cc} 2 & 4 & 2 \\ \hline 1 & 1 & 2 \\ 1 & 1 & 1 \end{array}\right],$$

so, at the first step,

$$a_{11} = 2,\ v = \begin{bmatrix} 1 \\ 1 \end{bmatrix},\ w^* = [4\ 2],\ A' = \begin{bmatrix} 1 & 2 \\ 1 & 1 \end{bmatrix},\ \left[\begin{array}{c|cc} 2 & 4 & 2 \\ \hline 1/2 & & \\ 1/2 & & \end{array}\right].$$

The first Schur complement is

$$A' - vw^*/a_{11} = \begin{bmatrix} 1 & 2 \\ 1 & 1 \end{bmatrix} - \frac{1}{2}\begin{bmatrix} 1 \\ 1 \end{bmatrix}[4\ 2] = \begin{bmatrix} 1 & 2 \\ 1 & 1 \end{bmatrix} - \begin{bmatrix} 2 & 1 \\ 2 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 1 \\ -1 & 0 \end{bmatrix}$$

and, for now, we have

$$\left[\begin{array}{c|cc} 2 & 4 & 2 \\ \hline 1/2 & -1 & 1 \\ 1/2 & -1 & 0 \end{array}\right] = \left[\begin{array}{c|cc} 2 & 4 & 2 \\ \hline 1/2 & -1 & 1 \\ \hline 1/2 & -1 & 0 \end{array}\right] = \left[\begin{array}{c|cc} 2 & 4 & 2 \\ \hline 1/2 & -1 & 1 \\ \hline 1/2 & 1 & \end{array}\right].$$

The last Schur complement is

$$0 - (-1)/(-1) \cdot 1 = -1$$

and the final decomposition is

$$\left[\begin{array}{c|cc} 2 & 4 & 2 \\ \hline 1/2 & -1 & 1 \\ \hline 1/2 & 1 & -1 \end{array}\right].$$

We take the upper triangular part (*including* the main diagonal) for U and the lower triangular part (*without* the main diagonal) for L (which will have all 1's on the main diagonal), to get

$$L = \begin{bmatrix} 1 & 0 & 0 \\ 1/2 & 1 & 0 \\ 1/2 & 1 & 1 \end{bmatrix},\ U = \begin{bmatrix} 2 & 4 & 2 \\ 0 & -1 & 1 \\ 0 & 0 & -1 \end{bmatrix}$$

and check that indeed $A = LU$.

Now, solve $Ly = b = [8\ 4\ 3]^T$, which, from Example 1.3b) has solution $y = [8\ 0\ -1]^T$ and then $Ux = [8\ 0\ -1]^T$, which, from Example 1.3a), gives the solution

$$x = [1\ 1\ 1]^T.$$

Check that $Ax = b$. ■

1.2.2 LUP Factorization

What if row interchanges (pivoting) *are* necessary? A row interchange is a permutation of two rows. We keep track of those in a *permutation* matrix, which is simply a matrix obtained from the corresponding identity matrix I by permuting rows. So, for a matrix A we find its LUP **factorization (decomposition)**, i.e., a triplet (L, U, P), with L a lower triangular, U an upper triangular and P a permutation matrix, such that

$$PA = LU. \tag{1.19}$$

Remark 1.10.
1. Solving the system $Ax = b$ is now equivalent to solving two triangular systems

$$\begin{aligned} Ly &= Pb \text{ and} \\ Ux &= y. \end{aligned} \tag{1.20}$$

2. Multiplication of a matrix A to the *left* by a permutation matrix P will yield the same *row* interchanges on the matrix A as in P, while multiplication on the *right* will result in the same *column* interchanges in A as in P.
3. The procedure for obtaining an LUP factorization is similar to the previous one, while keeping track of the row interchanges in a permutation matrix P.

Example 1.11. Find an LUP factorization for the matrix

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

Solution. At the first step, we do partial pivoting and interchange $(R_1) \longleftrightarrow (R_3)$.
At each row interchange, instead of writing the entire matrix P, we only emphasize which rows are permuted. Other than that, we proceed as before. We have

$$A \sim \begin{bmatrix} 3 & -1 & 1 \\ 1 & 1 & -1 \\ 2 & 1 & -2 \end{bmatrix} = \left[\begin{array}{c|cc} 3 & -1 & 1 \\ \hline 1 & 1 & -1 \\ 2 & 1 & -2 \end{array}\right] \sim \left[\begin{array}{c|cc} 3 & -1 & 1 \\ \hline 1/3 & & \\ 2/3 & & \end{array}\right], \quad \begin{bmatrix} 3 \\ 2 \\ 1 \end{bmatrix}.$$

Schur complement

$$\begin{bmatrix} 1 & -1 \\ 1 & -2 \end{bmatrix} - \frac{1}{3}\begin{bmatrix} 1 \\ 2 \end{bmatrix}[-1\ \ 1] = \begin{bmatrix} 1 & -1 \\ 1 & -2 \end{bmatrix} - \begin{bmatrix} -1/3 & 1/3 \\ -2/3 & 2/3 \end{bmatrix} = \begin{bmatrix} 4/3 & -4/3 \\ 5/3 & -8/3 \end{bmatrix},$$

so,

$$A \sim \left[\begin{array}{c|cc} 3 & -1 & 1 \\ \hline 1/3 & 4/3 & -4/3 \\ 2/3 & 5/3 & -8/3 \end{array}\right] \sim \left[\begin{array}{c|cc} 3 & -1 & 1 \\ \hline 2/3 & 5/3 & -8/3 \\ 1/3 & 4/3 & -4/3 \end{array}\right], \quad \begin{bmatrix} 3 \\ 1 \\ 2 \end{bmatrix},$$

because we interchanged $(R_2) \longleftrightarrow (R_3)$. Further, we have

$$A \sim \left[\begin{array}{c|c|c} 3 & -1 & 1 \\ \hline 2/3 & 5/3 & -8/3 \\ \hline 1/3 & 4/5 & \end{array}\right] \sim \left[\begin{array}{c|c|c} 3 & -1 & 1 \\ \hline 2/3 & 5/3 & -8/3 \\ \hline 1/3 & 4/5 & 4/5 \end{array}\right],$$

the last Schur complement being

$$-\frac{4}{3} - \frac{3}{5}\cdot\frac{4}{3}\Big(-\frac{8}{3}\Big) = \frac{4}{5}.$$

So, we obtained

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

Check that $PA = LU$.

■

Remark 1.12.

1. The computational cost for LU (and LUP) factorization is about the same as for Gaussian elimination, $O(n^3)$ flops. However, for *tridiagonal* matrices, that cost drops to $O(n)$ operations. The *Thomas algorithm*, based on LUP decomposition is an efficient way of solving tridiagonal matrix systems. In addition, only three one-dimensional arrays for the three diagonals are needed to store the matrix. This means that very large systems can be solved rapidly and efficiently, and systems of order over $n = 10,000$ are not unusual in some applications, for example, in solving boundary value problems for differential equations.

2. More generally, a *band* or *banded* matrix is a sparse matrix whose non-zero entries are confined to a diagonal band, comprising of the main diagonal and zero or more diagonals on either side. If all matrix elements are zero outside a diagonally bordered band whose range is determined by constants $k_1, k_2 \geq 0$,

$$a_{ij} \quad = \quad 0, \text{ if } \; j < i - k_1 \; \text{ or } \; j > i + k_2$$

then the quantities k_1 and k_2 are called the *lower bandwidth* and *upper bandwidth*, respectively. The *bandwidth* of the matrix is then defined as

$$w \quad = \quad \max\{k_1, k_2\},$$

i.e., it is the number w such that

$$a_{ij} \quad = \quad 0, \text{ if } \; |i - j| > w.$$

It can be shown that LU factorization with partial pivoting for $n \times n$ banded matrices with bandwidth w requires $O(w^2 n)$ flops, while triangular solvers require $O(wn)$ flops.