5.4.1 Standard Gaussian Elimination

From the first equation of system (5.44), express x_1 through the other variables:

$$x_1 = a'_{12}x_2 + \ldots + a'_{1n}x_n + f'_1. \tag{5.45}$$

Substitute expression (5.45) for x_1 into the remaining $n-1$ equations of system (5.44) and obtain a new system of $n-1$ linear algebraic equations with respect to $n-1$ unknowns: $x_2, x_3, \ldots, x_n$. From the first equation of this updated system express x_2 as a function of all other variables:

$$x_2 = a'_{23}x_3 + \ldots + a'_{2n}x_n + f'_2, \tag{5.46}$$

substitute into the remaining $n-2$ equations and obtain yet another system of further reduced dimension: $n-2$ equations with $n-2$ unknowns. Repeat this procedure, called elimination, for $k = 3, 4, \ldots, n-1$:

$$x_k = a'_{k,k+1}x_{k+1} + \ldots + a'_{kn}x_n + f'_k, \tag{5.47}$$

and every time obtain a new smaller linear system of dimension $(n-k)\times(n-k)$. At the last stage $k = n$, no substitution is needed, and we simply solve the 1×1 system, i.e., a single scalar linear equation from the previous stage $k = n-1$, which yields:

$$x_n = f'_n. \tag{5.48}$$

Having obtained the value of x_n by formula (5.48), we can find the values of the unknowns $x_{n-1}, x_{n-2}, \ldots, x_1$ one after another by employing formulae (5.47) in the ascending order: $k = n-1, n-2, \ldots, 1$. This procedure, however, is not fail-proof. It may break down because of a division by zero. Even if no division by zero occurs, the algorithm may still end up generating a large error in the solution due to an amplification of the small round-off errors. Let us explain these phenomena.

If the coefficient a_{11} in front of the unknown x_1 in the first equation of system (5.44) is equal to zero, $a_{11} = 0$, then already the first step of the elimination algorithm, i.e., expression (5.45), becomes invalid, because $a'_{12} = -a_{12}/a_{11}$. A division by zero can obviously be encountered at any stage of the algorithm. For example, having the coefficient in front of x_2 equal to zero in the first equation of the $(n-1)\times(n-1)$ system invalidates expression (5.46). But even if no division by zero is ever encountered, and formulae (5.47) are obtained for all $k = 1, 2, \ldots, n$, then the method can still develop a computational instability when solving for $x_n, x_{n-1}, \ldots, x_1$. For example, if it happens that $a'_{k,k+1} = 2$ for $k = n-1, n-2, \ldots, 1$, while all other a'_{ij} are equal to zero, then the small round-off error committed when evaluating x_n by formula (5.48) increases by a factor of two when computing x_{n-1}, then by another factor of two when computing x_{n-2}, and eventually by a factor of 2^{n-1} when computing x_n. Already for $n = 11$ the error grows more than a thousand times.

Another potential danger that one needs to keep in mind is that the quantities f'_k can rapidly increase when k increases. If this happens, then even small relative errors committed when computing f'_k in expression (5.47) can lead to a large absolute error in the value of x_k.

Let us now formulate a sufficient condition that would guarantee the computational stability of the Gaussian elimination algorithm.

THEOREM 5.6

Let the matrix A of system (5.44) be a matrix with diagonal dominance of magnitude $\delta > 0$, see formula (5.40). Then, no division by zero will be encountered in the standard Gaussian elimination algorithm. Moreover, the following inequalities will hold:

$$\sum_{j=1}^{n-k} |a'_{k,k+j}| < 1, \quad k = 1, 2, \ldots, n-1, \tag{5.49}$$

$$|f'_k| \leq \frac{2}{\delta} \max_j |f_j|, \quad k = 1, 2, \ldots, n. \tag{5.50}$$

To prove Theorem 5.6, we will first need the following Lemma.

LEMMA 5.1

Let

$$\begin{aligned} b_{11}y_1 + b_{12}y_2 + \ldots + b_{1m}y_m &= g_1, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots& \\ b_{m1}y_1 + b_{m2}y_2 + \ldots + b_{mm}y_m &= g_m, \end{aligned} \tag{5.51}$$

be a system of linear algebraic equations with diagonal dominance of magnitude $\delta > 0$:

$$|b_{ll}| \geq \sum_{j \neq l} |b_{lj}| + \delta, \quad l = 1, 2, \ldots, m. \tag{5.52}$$

Then, when reducing the first equation of system (5.51) to the form

$$y_1 = b'_{12}y_2 + \ldots + b'_{1m}y_m + g'_1 \tag{5.53}$$

there will be no division by zero. Besides, the inequality

$$\sum_{j=2}^{m} |b'_{1j}| < 1 \tag{5.54}$$

will hold. Finally, if $m > 1$, then the variable y_1 can be eliminated from system (5.51) with the help of expression (5.53). In doing so, the resulting $(m-1) \times (m-1)$ system with respect to the unknowns $y_2, y_3, \ldots, y_m$ will also be a system with diagonal dominance of the same magnitude δ as in formula (5.52).

PROOF According to formula (5.52), for $l = 1$ we have:

$$|b_{11}| \geq |b_{12}| + |b_{13}| + \ldots + |b_{1m}| + \delta.$$

Consequently, $b_{11} \neq 0$, and expression (5.53) makes sense, where

$$b'_{1j} = -\frac{b_{1j}}{b_{11}} \text{ for } j = 2,3,\dots,m, \text{ and } g'_1 = \frac{g_1}{b_{11}}.$$

Moreover,

$$\sum_{j=2}^{m} |b'_{1j}| = \frac{|b_{12}| + |b_{13}| + \dots + |b_{1m}|}{|b_{11}|},$$

hence inequality (5.54) is satisfied.

It remains to prove the last assertion of the Lemma for $m > 1$. Substituting expression (5.53) into the equation number j $(j > 1)$ of system (5.51), we obtain:

$$(b_{j2} + b_{j1}b'_{12})y_2 + (b_{j3} + b_{j1}b'_{13})y_3 + \dots (b_{jm} + b_{j1}b'_{1m})y_m = g'_j,$$
$$j = 2,3,\dots,m.$$

In this system of $m-1$ equations, equation number l $(l = 1,2,\dots,m-1)$ is the equation:

$$(b_{l+1,2} + b_{l+1,1}b'_{12})y_2 + (b_{l+1,3} + b_{l+1,1}b'_{13})y_3 + \dots (b_{l+1,m} + b_{l+1,1}b'_{1m})y_m = g'_{l+1},$$
$$l = 1,2,\dots,m-1. \tag{5.55}$$

Consequently, the entries in row number l of the matrix of system (5.55) are:

$$(b_{l+1,2} + b_{l+1,1}b'_{12}),\ (b_{l+1,3} + b_{l+1,1}b'_{13}),\ \dots,\ (b_{l+1,m} + b_{l+1,1}b'_{1m}),$$

and the corresponding diagonal entry is: $(b_{l+1,l+1} + b_{l+1,1}b'_{1,l+1})$.

Let us show that there is a diagonal dominance of magnitude δ, i.e., that the following estimate holds:

$$|b_{l+1,l+1} + b_{l+1,1}b'_{1,l+1}| \geq \sum_{\substack{j=2,\\ j\neq l+1}}^{m} |b_{l+1,j} + b_{l+1,1}b'_{1j}| + \delta. \tag{5.56}$$

We will prove an even stronger inequality:

$$|b_{l+1,l+1}| - |b_{l+1,1}b'_{1,l+1}| \geq \sum_{\substack{j=2,\\ j\neq l+1}}^{m} \left[|b_{l+1,j}| + |b_{l+1,1}b'_{1j}|\right] + \delta,$$

which, in turn, is equivalent to the inequality:

$$|b_{l+1,l+1}| \geq \sum_{\substack{j=2,\\ j\neq l+1}}^{m} |b_{l+1,j}| + |b_{l+1,1}| \sum_{j=2}^{m} |b'_{1j}| + \delta. \tag{5.57}$$

Let us replace the quantity $\sum\limits_{j=2}^{m} |b'_{1j}|$ in formula (5.57) by the number 1:

$$|b_{l+1,l+1}| \geq \sum_{\substack{j=2,\\ j\neq l+1}}^{m} |b_{l+1,j}| + |b_{l+1,1}| + \delta = \sum_{\substack{j=1,\\ j\neq l+1}}^{m} |b_{l+1,j}| + \delta. \tag{5.58}$$

According to estimate (5.54), if inequality (5.58) holds, then inequality (5.57) will automatically hold. However, inequality (5.58) is true because of estimate (5.52). Thus, we have proven inequality (5.56). □

PROOF OF THEOREM 5.6 We will first establish the validity of formula (5.47) and prove inequality (5.49). To do so, we will use induction with respect to k. If $k = 1$ and $n > 1$, formula (5.47) and inequality (5.49) are equivalent to formula (5.53) and inequality (5.54), respectively, proven in Lemma 5.1. In addition, Lemma 5.1 implies that the $(n-1)\times(n-1)$ system with respect to $x_2, x_3, \dots, x_n$ obtained from (5.44) by eliminating the variable x_1 using (5.45) will also be a system with diagonal dominance of magnitude δ.

Assume now that formula (5.47) and inequality (5.49) have already been proven for all $k = 1, 2, \dots, l$, where $l < n$. Also assume that the $(n-l)\times(n-l)$ system with respect to $x_{l+1}, x_{l+2}, \dots, x_n$ obtained from (5.44) by consecutively eliminating the variables $x_1, x_2, \dots, x_l$ using (5.47) is a system with diagonal dominance of magnitude δ. Then this system can be considered in the capacity of system (5.51), in which case Lemma 5.1 immediately implies that the assumption of the induction is true for $k = l+1$ as well. This completes the proof by induction.

Next, let us justify inequality (5.50). According to Theorem 5.5, the following estimate holds for the solution of system (5.44):

$$\max_j |x_j| \leq \frac{1}{\delta} \max_i |f_i|.$$

Employing this inequality along with formula (5.47) and inequality (5.49) that we have just proven, we obtain for any $k = 1, 2, \dots, n$:

$$\begin{aligned} |f'_k| &= \left| x_k - \sum_{j=k+1}^{n} a'_{kj} x_j \right| \leq |x_k| + \sum_{j=k+1}^{n} |a'_{kj}||x_j| \\ &\leq \max_{1\leq j\leq n} |x_j| \left(1 + \sum_{j=k+1}^{n} |a'_{kj}| \right) \leq 2 \max_{1\leq j\leq n} |x_j| \leq \frac{2}{\delta} \max_i |f_i|. \end{aligned}$$

This estimate obviously coincides with (5.50). □

Let us emphasize that the hypothesis of Theorem 5.6 (diagonal dominance) provides a sufficient but not a necessary condition for the applicability of the standard Gaussian elimination procedure. There are other linear systems (5.44) that lend themselves to the solution by this method. If, for a given system (5.44), the Gaussian elimination is successfully implemented on a computer (no divisions by zero and no instabilities), then the accuracy of the resulting exact solution will only be limited by the machine precision, i.e., by the round-off errors.

Having computed the approximate solution of system (5.44), one can substitute it into the left-hand side and thus obtain the residual Δf of the right-hand side. Then, estimate

$$\frac{\|\Delta x\|}{\|x\|} \leq \mu(A)\frac{\|\Delta f\|}{\|f\|},$$

can be used to judge the error of the solution, provided that the condition number $\mu(A)$ or its upper bound is known. This is one of the ideas behind the so-called a posteriori analysis.

Computational complexity of the standard Gaussian elimination algorithm is cubic with respect to the dimension n of the system. More precisely, it requires roughly $\frac{2}{3}n^3+2n^2=\mathcal{O}(n^3)$ arithmetic operations to obtain the solution. In doing so, the cubic component of the cost comes from the elimination per se, whereas the quadratic component is the cost of solving back for $x_n, x_{n-1}, \ldots, x_1$.

5.4.2 Tri-Diagonal Elimination

The Gaussian elimination algorithm of Section 5.4.1 is particularly efficient for a system (5.44) of the following special kind:

$$\begin{array}{llllll}
b_1x_1 + c_1x_2 & & & & = & f_1, \\
a_2x_1 + b_2x_2 + & c_2x_3 & & & = & f_2, \\
\quad a_3x_2 + & b_3x_3 & + & c_3x_4 & = & f_3, \\
\multicolumn{6}{c}{\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots} \\
& & & a_{n-1}x_{n-2} + b_{n-1}x_{n-1} + c_{n-1}x_n & = & f_{n-1}, \\
& & & a_nx_{n-1} + b_nx_n & = & f_n.
\end{array} \tag{5.59}$$

The matrix of system (5.59) is tri-diagonal, i.e., all its entries are equal to zero except those on the main diagonal, on the super-diagonal, and on the sub-diagonal. In other words, if $A=\{\mathfrak{a}_{ij}\}$, then $\mathfrak{a}_{ij}=0$ for $j>i+1$ and $j<i-1$. Adopting the notation of formula (5.59), we say that $\mathfrak{a}_{ii}=b_i$, $i=1,2,\ldots,n$; $\mathfrak{a}_{i,i-1}=a_i$, $i=2,3,\ldots,n$; and $\mathfrak{a}_{i,i+1}=c_i$, $i=1,2,\ldots,n-1$.

The conditions of diagonal dominance (5.40) for system (5.59) read:

$$\begin{aligned}
&|b_1| \geq |c_1| + \delta, \\
&|b_k| \geq |a_k| + |c_k| + \delta, \quad k=2,3,\ldots,n-1, \\
&|b_n| \geq |a_n| + \delta.
\end{aligned} \tag{5.60}$$

Equalities (5.45)–(5.48) transform into:

$$\begin{aligned}
&x_k = A_kx_{k+1} + F_k, \quad k=1,2,\ldots,n-1, \\
&x_n = F_n,
\end{aligned} \tag{5.61}$$

where A_k and F_k are some coefficients. Define $A_n=0$ and rewrite formulae (5.61) as a unified expression:

$$x_k = A_kx_{k+1} + F_k, \quad k=1,2,\ldots,n. \tag{5.62}$$

From the first equation of system (5.59) it is clear that for $k = 1$ the coefficients in formula (5.62) are:

$$A_1 = -\frac{c_1}{b_1}, \quad F_1 = \frac{f_1}{b_1}. \tag{5.63}$$

Suppose that all the coefficients A_k and F_k have already been computed up to some fixed k, $1 \le k \le n-1$. Substituting the expression $x_k = A_k x_{k+1} + F_k$ into the equation number $k+1$ of system (5.59) we obtain:

$$x_{k+1} = -\frac{c_{k+1}}{b_{k+1} + a_{k+1}A_k} x_{k+2} + \frac{f_{k+1} - a_{k+1}F_k}{b_{k+1} + a_{k+1}A_k}.$$

Therefore, the coefficients A_k and F_k satisfy the following recurrence relations:

$$A_{k+1} = -\frac{c_{k+1}}{b_{k+1} + a_{k+1}A_k}, \quad F_{k+1} = \frac{f_{k+1} - a_{k+1}F_k}{b_{k+1} + a_{k+1}A_k}, \tag{5.64}$$
$$k = 1, 2, \dots, n-1.$$

As such, the algorithm of solving system (5.59) gets split into two stages. At the first stage, we evaluate the coefficients A_k and F_k for $k = 1, 2, \dots, n$ using formulae (5.63) and (5.64). At the second stage, we solve back for the actual unknowns $x_n, x_{n-1}, \dots, x_1$ using formulae (5.62) for $k = n, n-1, \dots, 1$.

In the literature, one can find several alternative names for the tri-diagonal Gaussian elimination procedure that we have described. Sometimes, the term *marching* is used. The first stage of the algorithm is also referred to as the forward stage or forward marching, when the marching coefficients A_k and B_k are computed. Accordingly, the second stage of the algorithm, when relations (5.62) are applied consecutively in the reverse order is called backward marching.

We will now estimate the computational complexity of the tri-diagonal elimination. At the forward stage, the elimination according to formulae (5.63) and (5.64) requires $\mathcal{O}(n)$ arithmetic operations. At the backward stage, formula (5.62) is applied n times, which also requires $\mathcal{O}(n)$ operations. Altogether, the complexity of the tri-diagonal elimination is $\mathcal{O}(n)$ arithmetic operations. It is clear that no algorithm can be built that would be asymptotically cheaper than $\mathcal{O}(n)$, because the number of unknowns in the system is also $\mathcal{O}(n)$.

Let us additionally note that the tri-diagonal elimination is apparently the only example available in the literature of a direct method with linear complexity, i.e., of a method that produces the exact solution of a linear system at a cost of $\mathcal{O}(n)$ operations. In other words, the computational cost is directly proportional to the dimension of the system. We will later see examples of direct methods that produce the exact solution at a cost of $\mathcal{O}(n \ln n)$ operations, and examples of iterative methods that cost $\mathcal{O}(n)$ operations but only produce an approximate solution. However, no other method of computing the exact solution with a genuinely linear complexity is known.

The algorithm can also be generalized to the case of the banded matrices. Matrices of this type may contain non-zero entries on several neighboring diagonals, including the main diagonal. Normally we would assume that the number m of the non-zero diagonals, i.e., the bandwidth, satisfies $3 \leq m \ll n$. The complexity of the Gaussian elimination algorithm when applied to a banded system (5.44) is $\mathcal{O}(m^2 n)$ operations. If m is fixed and n is arbitrary, then the complexity, again, scales as $\mathcal{O}(n)$.

High-order systems of type (5.59) drew the attention of researchers in the fifties. They appeared when solving the heat equation numerically with the help of the so-called implicit finite-difference schemes. These schemes, their construction and their importance, will be discussed later in Part III of the book, see, in particular, Section 10.6.

The foregoing tri-diagonal marching algorithm was apparently introduced for the first time by I. M. Gelfand and O. V. Lokutsievskii around the late forties or early fifties. They conducted a complete analysis of the algorithm, showed that it was computationally stable, and also built its continuous "closure," see Appendix II to the book [GR64] written by Gelfand and Lokutsievskii.

Alternatively, the tri-diagonal elimination algorithm is attributed to L. H. Thomas [Tho49], and is referred to as the Thomas algorithm.

The aforementioned work by Gelfand and Lokutsievskii was one of the first papers in the literature where the question of stability of a computational algorithm was accurately formulated and solved for a particular class of problems. This question has since become one of the key issues for the entire large field of knowledge called scientific computing. Having stability is crucial, as otherwise computer codes that implement the algorithms will not execute properly. In Part III of the book, we study computational stability for the finite-difference schemes.

Theorem 5.6, which provides sufficient conditions for applicability of the Gaussian elimination, is a generalization to the case of full matrices of the result by Gelfand and Lokutsievskii on stability of the tri-diagonal marching.

Note also that the conditions of strict diagonal dominance (5.60) that are sufficient for stability of the tri-diagonal elimination can actually be relaxed. In fact, one can only require that the coefficients of system (5.59) satisfy the inequalities:

$$\begin{aligned} &|b_1| \geq |c_1|, \\ &|b_k| \geq |a_k| + |c_k|, \quad k = 2,3,\dots,n-1, \\ &|b_n| \geq |a_n|, \end{aligned} \tag{5.65}$$

so that at least one out of the total of n inequalities (5.65) actually be strict, i.e., "$>$" rather than "$\geq$." We refer the reader to [SN89a, Chapter II] for detail.

Overall, the idea of transporting, or marching, the condition $b_1 x_1 + c_1 x_2 = f_1$ specified by the first equation of the tri-diagonal system (5.59) is quite general. In the previous algorithm, this idea is put to use when obtaining the marching coefficients (5.63), (5.64) and relations (5.62). It is also exploited in many other elimination algorithms, see [SN89a, Chapter II]. We briefly describe one of those algorithms, known as the cyclic tri-diagonal elimination, in Section 5.4.3.

5.4.3 Cyclic Tri-Diagonal Elimination

In many applications, one needs to solve a system which is "almost" tri-diagonal, but is not quite equivalent to system (5.59):

$$\begin{array}{lllll}
b_1x_1 + c_1x_2 & & & + \; a_1x_n & = f_1, \\
a_2x_1 + b_2x_2 + & c_2x_3 & & & = f_2, \\
\quad a_3x_2 + & b_3x_3 & + \quad c_3x_4 & & = f_3, \\
\multicolumn{5}{c}{\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots} \\
& & a_{n-1}x_{n-2} + b_{n-1}x_{n-1} & + \; c_{n-1}x_n & = f_{n-1}, \\
c_nx_1 & & + \quad a_nx_{n-1} & + \; b_nx_n & = f_n.
\end{array} \tag{5.66}$$

In Section 2.3.2, we have analyzed one particular example of this type that arises when constructing the nonlocal Schoenberg splines; see the matrix A given by formula (2.66) on page 52. Other typical examples include the so-called central difference schemes (see Section 9.2.1) for the solution of second order ordinary differential equations with periodic boundary conditions, as well as many schemes built for solving partial differential equations in the cylindrical or spherical coordinates. Periodicity of the boundary conditions gave rise to the name *cyclic* attached to the version of the tri-diagonal elimination that we are about to describe.

The coefficients a_1 and c_n in the first and last equations of system (5.66), respectively, are, generally speaking, non-zero. Their presence does not allow one to apply the tri-diagonal elimination algorithm of Section 5.4.2 to system (5.66) directly. Let us therefore consider two auxiliary linear systems of dimension $(n-1)\times(n-1)$:

$$\begin{array}{llll}
b_2u_2 + & c_2u_3 & & = f_2, \\
a_3u_2 + & b_3u_3 & + \quad c_3u_4 & = f_3, \\
\multicolumn{4}{c}{\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots} \\
& & a_{n-1}u_{n-2} + b_{n-1}u_{n-1} + c_{n-1}u_n & = f_{n-1}, \\
& & \qquad a_nu_{n-1} \; + \; b_nu_n & = f_n.
\end{array} \tag{5.67}$$

and

$$\begin{array}{llll}
b_2v_2 + & c_2v_3 & & = -a_2, \\
a_3v_2 + & b_3v_3 & + \quad c_3v_4 & = \; 0, \\
\multicolumn{4}{c}{\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots} \\
& & a_{n-1}v_{n-2} + b_{n-1}v_{n-1} + c_{n-1}v_n & = \; 0, \\
& & \qquad a_nv_{n-1} \; + \; b_nv_n & = -c_1.
\end{array} \tag{5.68}$$

Having obtained the solutions $\{u_2,u_3,\dots,u_n\}$ and $\{v_2,v_3,\dots,v_n\}$ to systems (5.67) and (5.68), respectively, we can represent the solution $\{x_1,x_2,\dots,x_n\}$ to system (5.66) in the form:

$$x_i = u_i + x_1v_i, \quad i = 1,2,\dots,n, \tag{5.69}$$

where for convenience we additionally define $u_1 = 0$ and $v_1 = 1$. Indeed, multiplying each equation of system (5.68) by x_1, adding with the corresponding equation of system (5.67), and using representation (5.69), we immediately see that the equations number 2 through n of system (5.66) are satisfied. It only remains to satisfy equation number 1 of system (5.66). To do so, we use formula (5.69) for $i=2$ and $i=n$, and substitute $x_2 = u_2 + x_1 v_2$ and $x_n = u_n + x_1 v_n$ into the first equation of system (5.66), which yields one scalar equation for x_1:

$$b_1 x_1 + c_1(u_2 + x_1 v_2) + a_1(u_n + x_1 v_n) = f_1.$$

As such, we find:

$$x_1 = \frac{f_1 - a_1 u_n - c_1 u_2}{b_1 + a_1 v_n + c_1 v_2}. \tag{5.70}$$

Altogether, the solution algorithm for system (5.66) reduces to first solving the two auxiliary systems (5.67) and (5.68), then finding x_1 with the help of formula (5.70), and finally obtaining $x_2, x_3, \dots, x_n$ according to formula (5.69). As both systems (5.67) and (5.68) are genuinely tri-diagonal, they can be solved by the original tri-diagonal elimination described in Section 5.4.2. In doing so, the overall computational complexity of solving system (5.66) obviously remains linear with respect to the dimension of the system n, i.e., $\mathcal{O}(n)$ arithmetic operations.

5.4.4 Matrix Interpretation of the Gaussian Elimination. *LU* Factorization

Consider system (5.44) written as $\boldsymbol{Ax} = \boldsymbol{f}$ or alternatively, $\boldsymbol{A}^{(0)}\boldsymbol{x} = \boldsymbol{f}^{(0)}$, where

$$\boldsymbol{A} \equiv \boldsymbol{A}^{(0)} = \begin{bmatrix} a_{11}^{(0)} & \dots & a_{1n}^{(0)} \\ \vdots & \ddots & \vdots \\ a_{n1}^{(0)} & \dots & a_{nn}^{(0)} \end{bmatrix} \quad \text{and} \quad \boldsymbol{f} \equiv \boldsymbol{f}^{(0)} = \begin{bmatrix} f_1^{(0)} \\ \vdots \\ f_n^{(0)} \end{bmatrix},$$

and the superscript "(0)" emphasizes that this is the beginning, i.e., the zeroth stage of the Gaussian elimination procedure. The notations in this section are somewhat different from those of Section 5.4.1, but we will later show the correspondence.

At the first stage of the algorithm we assume that $a_{11}^{(0)} \neq 0$ and introduce the transformation matrix:

$$\boldsymbol{T}_1 = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ -t_{21} & 1 & 0 & \dots & 0 \\ -t_{31} & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -t_{n1} & 0 & 0 & \dots & 1 \end{bmatrix}, \quad \text{where} \quad t_{i1} = \frac{a_{i1}^{(0)}}{a_{11}^{(0)}}, \quad i = 2,3,\dots,n.$$

Applying this matrix is equivalent to eliminating the variable x_1 from equations number 2 through n of system (5.44):

$$\boldsymbol{T}_1\boldsymbol{A}^{(0)}\boldsymbol{x} \equiv \boldsymbol{A}^{(1)}\boldsymbol{x} = \begin{bmatrix} a_{11}^{(0)} & a_{12}^{(0)} & \dots & a_{1n}^{(0)} \\ 0 & a_{22}^{(1)} & \dots & a_{2n}^{(1)} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & a_{n2}^{(1)} & \dots & a_{nn}^{(1)} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} f_1^{(0)} \\ f_2^{(1)} \\ \vdots \\ f_n^{(1)} \end{bmatrix} = \boldsymbol{f}^{(1)} \equiv \boldsymbol{T}_1\boldsymbol{f}^{(0)}.$$