2.10. FIXED-POINT ITERATION METHOD (LINEAR SYSTEMS)

The fixed-point iteration method belongs to iterative methods for solving systems of linear algebraic equations. Methods of successive approximations are called iterative methods, in which the previous, already known approximate solutions are used to calculate the next approximation of the solution. The iterative methods do not give a fundamentally exact solution x^*, since it is attained as the limit of the sequence of approximations $x^{(k)}, k = 0,1,2,...$ at $k \to \infty$. However, iterative methods have a property that allows to obtain a solution with any predetermined accuracy if the convergence of the method is proved. Therefore, when solving any problem by the iterative method, there inevitably appears the need to set the required level of accuracy for the answer, i.e. to set the permissible error of the solution ε.

It should be noted that, in case of a large number of equations, direct methods for solving SLAEs (excluding the sweep method) are losing efficiency when implemented on a computer primarily due to the complexity of large-dimensional matrices storing and processing. At the same time, the characteristic feature of a number of SLAEs that are often encountered in applied problems is the sparsity of their matrices. The number of nonzero elements of such matrices is relatively small as compared to their dimension. Therefore, to solve SLAEs with sparse matrices, it is preferable to use iterative methods.

Suppose we are given a SLAE with a nondegenerate matrix ($\det A \neq 0$):

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n = b_n \end{cases}.$$

Let's reduce this SLAE to the following equivalent form:

$$\begin{cases} x_1 = \beta_1 + \alpha_{11}x_1 + \alpha_{12}x_2 + \ldots + \alpha_{1n}x_n \\ x_2 = \beta_2 + \alpha_{21}x_1 + \alpha_{22}x_2 + \ldots + \alpha_{2n}x_n \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_n = \beta_n + \alpha_{n1}x_1 + \alpha_{n2}x_2 + \ldots + \alpha_{nn}x_n \end{cases},$$

or in the vector-matrix form:

$x = \beta + \alpha x$, where $x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$, $\beta = \begin{pmatrix} \beta_1 \\ \vdots \\ \beta_n \end{pmatrix}$, $\alpha = \begin{pmatrix} \alpha_{11} & \cdots & \alpha_{1n} \\ \vdots & \cdots & \vdots \\ \alpha_{n1} & \ldots & \alpha_{nn} \end{pmatrix}$.

This reduction can be performed in various ways, including through the identical equation $x = x + C(Ax - b)$, where C is the number chosen in accordance with possible additional requirements or, in the most general case, a matrix.

However, one of the most common ways is the detachment of the diagonal elements. Let's solve the SLAE with respect to the unknowns with the nonzero diagonal elements $a_{ii} \neq 0, i = 1,\ldots,n$ (if any coefficient on the main diagonal is equal to zero, it is sufficient to swap the corresponding equation with any other equation). We shall obtain the following expressions for the components of the vector β and the matrix α of the equivalent system:

$$\beta_i = \frac{b_i}{a_{ii}}; \alpha_{ij} = -\frac{a_{ij}}{a_{ii}}, i \neq j; \alpha_{ij} = 0, i = j; i, j = 1,\ldots,n,$$

and the SLAE in the equivalent form will be written as

$$x_i = \left[\frac{b_i}{a_{ii}} - \sum_{j=1, j\neq i}^{n} \frac{a_{ij}}{a_{ii}} x_j \right], \; i = 1,...,n.$$

With this way of reducing the original SLAE to an equivalent form, the fixed-point iteration method is called the Jacobi method.

Regardless of the way the SLAE is reduced to an equivalent form, the fixed-point iteration method has the following form:

$$\begin{cases} x^{(0)} = \beta \\ x^{(1)} = \beta + \alpha x^{(0)} \\ x^{(2)} = \beta + \alpha x^{(1)} \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ x^{(k)} = \beta + \alpha x^{(k-1)} \end{cases}$$, where k is the number of iterations performed.

Now, the advantage of iterative methods is obvious in comparison with, for example, the Gaussian method. Only matrix-vector products are involved in the computational process, which allows working only with non-zero matrix elements, significantly optimizing the process of storing and processing matrices using a computer.

When constructing any iterative process to find the exact solution x^* to a certain problem, it is necessary to answer three questions: where does it begin, i.e. what is the zero approximation value $x^{(0)}$; under what conditions does it converge to a solution, i.e. $\lim\limits_{k\to\infty}\left\|x^{(k)}-x^*\right\|=0$; when to stop it, i.e. when the required accuracy ε is achieved $\left\|x^{(k)}-x^*\right\|\le\varepsilon$.

The vector $x^{(0)}=\beta$, $(x_1^{(0)}\quad x_2^{(0)}\quad ...\quad x_n^{(0)})^T=(\beta_1\quad \beta_2\quad ...\quad \beta_n)^T$ can be taken as a zeroth order approximation $x^{(0)}$ of the vector of unknowns; however, if the iterative process converges, it will converge for any other initial vector.

There takes place the following sufficient condition for the convergence of the fixed-point iteration method. The fixed-point iteration method converges to the unique solution of the SLAE at any initial approximation $x^{(0)}$ if any norm of the matrix α of the equivalent system is less than one: $\|\alpha\|<1$.

Indeed, according to the chosen iterative method, $x^{(k)}=\beta+\alpha x^{(k-1)}$ is true, and according to the original equation, $x^*=\beta+\alpha x^*$ is the identity if x^* is the desired solution, as $x=\beta+\alpha x$ is equivalent to $Ax=b$.

Then, after subtraction,

$x^{(k)}-x^*=\alpha(x^{(k-1)}-x^*)$, and so, also true is $\left\|x^{(k)}-x^*\right\|=\left\|\alpha(x^{(k-1)}-x^*)\right\|$.

According to the fourth property of the norms for matrices,

$\left\|x^{(k)}-x^*\right\|=\left\|\alpha(x^{(k-1)}-x^*)\right\|\le\|\alpha\|\cdot\left\|x^{(k-1)}-x^*\right\|$.

Applying this ratio to itself *k* times, we can get:

$\left\|x^{(k)}-x^*\right\|\le\|\alpha\|\cdot\left\|x^{(k-1)}-x^*\right\|\le\|\alpha\|^2\cdot\left\|x^{(k-2)}-x^*\right\|\le\ldots\le\|\alpha\|^k\cdot\left\|x^{(0)}-x^*\right\|$.

From here, it is obvious that when $k\to\infty$, $\left\|x^{(k)}-x^*\right\|\to 0$ is true if

$0\le\left\|x^{(k)}-x^*\right\|\le\|\alpha\|^k\cdot\left\|x^{(0)}-x^*\right\|\to 0$, but $\|\alpha\|^k\to 0$ when $k\to\infty$ only if $\|\alpha\|<1$.

So, it is proved that $\|\alpha\|<1\Rightarrow\lim\limits_{k\to\infty}\left\|x^{(k)}-x^*\right\|=0$.

Moreover, the convergence rate is not less than the rate of geometric progression with the denominator $q\le\|\alpha\|$.

If the Jacobi method is used, then the diagonal dominance of the matrix *A* is a sufficient condition for the convergence, i.e. $|a_{ii}| > \sum_{j=1, i\neq j}^{n} |a_{ij}|, i = 1,\ldots,n$ (for each row of the matrix *A*, the modules of the elements on the main diagonal are larger than the sum of the modules of the non-diagonal elements). Obviously, in this case, $\|\alpha\|_c$ is less than one and, therefore, the iterative process converges.

Let's note once again that the considered forms of the convergence condition for the fixed-point iteration method are only sufficient. Their fulfilment guarantees method convergence, but their nonfulfillment, in the general case, does not mean that the fixed-point iteration method diverges.

Let's also present a necessary and sufficient condition for the convergence of the fixed-point iteration method (it is rarely used in computational practice). For the convergence of the iterative process, it is necessary and sufficient that the spectrum of the matrix α of the equivalent system is inside a circle with the radius equal to one, i.e. that the maximal in modulus eigenvalue of the matrix (spectral radius of the matrix α) $\rho(\alpha) = \max_i |\lambda_i| < 1$.

If the sufficient condition for the convergence is satisfied, the estimate of the error of the solution at the *k*-th iteration $\|x^{(k)} - x^*\|$ is given by the expression

$\|x^{(k)} - x^*\| \le \varepsilon^{(k)} = \frac{\|\alpha\|}{1-\|\alpha\|}\|x^{(k)} - x^{(k-1)}\|$, where x^* is the exact solution to the SLAE.

Indeed, according to the third property of the norms for vectors, it is true that

$$\|x^{(k)} - x^*\| = \|\alpha(x^{(k-1)} - x^*)\| \le \|\alpha\| \cdot \|x^{(k-1)} - x^*\| = \|\alpha\| \cdot \|x^{(k-1)} - x^{(k)} + x^{(k)} - x^*\| \le$$

$$\le \|\alpha\| \cdot \left(\|x^{(k-1)} - x^{(k)}\| + \|x^{(k)} - x^*\|\right) = \|\alpha\| \cdot \|x^{(k)} - x^{(k-1)}\| + \|\alpha\| \cdot \|x^{(k)} - x^*\|.$$

Expressing $\|x^{(k)} - x^*\|$, we get $\|x^{(k)} - x^*\| \le \frac{\|\alpha\|}{1-\|\alpha\|} \cdot \|x^{(k)} - x^{(k-1)}\| = \varepsilon^{(k)}$.

So, the iteration process stops when the condition $\varepsilon^{(k)} \le \varepsilon$ is fulfilled, where ε is the accuracy specified by the calculator.

Taking into account that $\left\|x^{(k)}-x^{*}\right\| \le \frac{\|\alpha\|^{k}}{1-\|\alpha\|}\left\|x^{(1)}-x^{(0)}\right\|$, we can get an a-priori estimate of the number of iterations necessary to achieve the given accuracy. When the vector β is used as an initial approximation, such an estimate is determined by the inequality $\frac{\|\alpha\|^{k+1}}{1-\|\alpha\|}\|\beta\| \le \varepsilon$, wherefrom we obtain an a-priori estimate of the number of iterations *k* when $\|\alpha\|<1$: $k+1 \ge \frac{\lg\varepsilon-\lg\|\beta\|+\lg\left(1-\|\alpha\|\right)}{\lg\|\alpha\|}$. It should be emphasized that this inequality gives an overestimated number of iterations *k*, therefore, it is rarely used in practice.

Since $\|\alpha\|<1$ is only a sufficient (not necessary) condition for the convergence of the fixed-point iteration method, the iterative process can converge as well if it is not fulfilled. Then the inequality $\left\|x^{(k)}-x^{(k-1)}\right\| \le \varepsilon$ can serve as a criterion for the completion of iterations.

2.11. SEIDEL METHOD

The fixed-point iteration method converges rather slowly. The Seidel iteration method is intended to accelerate it, as it uses the obvious possibility of improving the convergence of the iterative process – the immediate introduction of newly calculated components of the vector $x^{(k)}$ into the calculation. The Seidel method implies the following: when calculating the component $x_i^{(k)}$ of the vector of unknowns at the *k*-th iteration, the already calculated values $x_1^{(k)}, x_2^{(k)}, \ldots, x_{i-1}^{(k)}$ are used at the same *k*-th iteration. The values for the remaining components $x_i^{(k-1)}, x_{i+1}^{(k-1)}, \ldots, x_n^{(k-1)}$ are taken from the previous iteration. Similar to the fixed-point iteration method, an equivalent SLAE is constructed, the vector $x^{(0)}=\beta$ is taken as the initial approximation, and the iterative process $x^{(k)}=\beta+\alpha x^{(k-1)}$ is constructed. The difference is that the Seidel method for the vector $x^{(k)}$ at the *k*-th iteration looks as follows:

$$\begin{cases} x_1^{(k)} = \beta_1 + \alpha_{11}x_1^{(k-1)} + \alpha_{12}x_2^{(k-1)} + \ldots + \alpha_{1n}x_n^{(k-1)} \\ x_2^{(k)} = \beta_2 + \alpha_{21}x_1^{(k)} + \alpha_{22}x_2^{(k-1)} + \ldots + \alpha_{2n}x_n^{(k-1)} \\ x_3^{(k)} = \beta_3 + \alpha_{31}x_1^{(k)} + \alpha_{32}x_2^{(k)} + \alpha_{33}x_3^{(k-1)} + \ldots + \alpha_{13}x_n^{(k-1)} \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_n^{(k)} = \beta_n + \alpha_{n1}x_1^{(k)} + \alpha_{n2}x_2^{(k)} + \ldots + \alpha_{nn-1}x_{n-1}^{(k)} + \alpha_{nn}x_n^{(k-1)} \end{cases},$$

or, in short:

$$x_i^{(k)} = \beta_i + \sum_{j=1}^{i-1} \alpha_{ij}x_j^{(k)} + \sum_{j=i+1}^{n} \alpha_{ij}x_j^{(k-1)}, \quad i = 1,\ldots,n$$

$$\left(x_i^{(k)} = \left[\frac{b_i}{a_{ii}} - \sum_{j=1}^{i-1} \frac{a_{ij}}{a_{ii}} x_j^{(k)} - \sum_{j=i+1}^{n} \frac{a_{ij}}{a_{ii}} x_j^{(k-1)} \right], \; i = 1,\ldots,n \text{ for the Jacobi method} \right).$$

It can be seen from here that $x^{(k)} = \beta + Bx^{(k)} + Cx^{(k-1)}$, where B is the lower triangular matrix with diagonal elements equal to zero, and C is the upper triangular matrix with diagonal elements other than zero, and $\alpha = B + C$ is true. Therefore,

$$(E - B)x^{(k)} = \beta + Cx^{(k-1)},$$

wherefrom, since the matrix $(E - B)$ is always nondegenerate,

$$x^{(k)} = (E - B)^{-1}\beta + (E - B)^{-1}Cx^{(k-1)}.$$

So, the Seidel method is equivalent to the fixed-point iteration method with the matrix of the right-hand sides $\tilde{\alpha} = (E - B)^{-1}C$ and the vector of the right-hand sides $\tilde{\beta} = (E - B)^{-1}\beta$, and, therefore, the convergence conditions and error formulas of the Seidel method can be studied using the formulas derived for the fixed-point iteration method, in which the matrix $\tilde{\alpha} = (E - B)^{-1}C$ is applied instead of the matrix α and the vector $\tilde{\beta} = (E - B)^{-1}\beta$ is applied instead of the vector β.

For practical calculations, it is important that the conditions given above for the fixed-point iteration method ($\|\alpha\| < 1$ or, if an equivalent SLAE in the Jacobi form is used, the diagonal dominance of the matrix A) can be used as sufficient conditions for the convergence of the Seidel method. If these conditions are met, the following expression can be used to estimate the error at the k-th iteration:

$$\left\| x^{(k)} - x^* \right\| \le \frac{\|C\|}{1 - \|\alpha\|} \cdot \left\| x^{(k)} - x^{(k-1)} \right\| \le \frac{\|\alpha\|}{1 - \|\alpha\|} \cdot \left\| x^{(k)} - x^{(k-1)} \right\| = \varepsilon^{(k)}.$$

Let's note that, like the fixed-point iteration method, the Seidel method can converge even if the condition $\|\alpha\| < 1$ is violated. In this case, $\varepsilon^{(k)} = \left\| x^{(k)} - x^{(k-1)} \right\|$.

Moreover, when using the Seidel method, a problem of finding the optimal sequence of refinement of the components of the vector of unknowns $x^{(k)}$ at each iteration can be stated. No satisfactory methods for constructing such a sequence are known. In practice, the ordering of the components in decreasing order of the difference of their values at two successive iterations is sometimes used.

The iterative process of the Seidel method can be generalized and written in the following form:

$$x_i^{(k)} = \omega\left(\beta_i + \sum_{j=1}^{i-1} \alpha_{ij} x_j^{(k)} + \sum_{j=i+1}^{n} \alpha_{ij} x_j^{(k-1)} \right) + (1-\omega) x_i^{(k-1)},\ i = 1,\ldots,n,$$

with the introduction of the value $x_i^{(k-1)}$ into the iterative process at the *k*-th step of calculating $x_i^{(k)}$, which is used neither in the fixed-point iteration method (with transformation by the Jacobi method) nor in the Seidel method.

In this case, when $\omega > 1$, the iterative process is called the over-relaxation method, when $\omega = 1$ (Seidel method) – the complete relaxation method, and when $\omega < 1$ – the under-relaxation method.

Example. Using the fixed-point iteration method and the Seidel method, solve the SLAE $Ax = b$ with an accuracy of $\varepsilon = 0.2$:

$$\begin{cases} 5x_1 - x_2 + 2x_3 + x_4 = 7 \\ 4x_1 + 11x_2 - 2x_3 - 3x_4 = 10 \\ 6x_1 - 3x_2 + 16x_3 - 4x_4 = 15 \\ 7x_1 + 2x_2 + 4x_3 - 14x_4 = -1 \end{cases};\ A = \begin{pmatrix} 5 & -1 & 2 & 1 \\ 4 & 11 & -2 & -3 \\ 6 & -3 & 16 & -4 \\ 7 & 2 & 4 & -14 \end{pmatrix},\ b = \begin{pmatrix} 7 \\ 10 \\ 15 \\ -1 \end{pmatrix}.$$

Let's reduce the SLAE to the equivalent form $x = \beta + \alpha x$ using the Jacobi method:

$$\begin{cases} x_1 = \frac{7}{5} + \frac{1}{5}x_2 - \frac{2}{5}x_3 - \frac{1}{5}x_4 \\ x_2 = \frac{10}{11} - \frac{4}{11}x_1 + \frac{2}{11}x_3 + \frac{3}{11}x_4 \\ x_3 = \frac{15}{16} - \frac{6}{16}x_1 + \frac{3}{16}x_2 + \frac{4}{16}x_4 \\ x_4 = \frac{1}{14} + \frac{7}{14}x_1 + \frac{2}{14}x_2 + \frac{4}{14}x_3 \end{cases}; \ \alpha = \begin{pmatrix} 0 & 1/5 & -2/5 & -1/5 \\ -4/11 & 0 & 2/11 & 3/11 \\ -3/8 & 3/16 & 0 & 1/4 \\ 1/2 & 1/7 & 2/7 & 0 \end{pmatrix}, \ \beta = \begin{pmatrix} 7/5 \\ 10/11 \\ 15/16 \\ 1/14 \end{pmatrix}.$$

The norm of the matrix α is $q = \|\alpha\|_c = \frac{13}{14} = 0.929 < 1$, therefore, the sufficient condition for the convergence is fulfilled. Initial approximation:

$$\alpha = \begin{pmatrix} 0 & 0.200 & -0.400 & -0.200 \\ -0.364 & 0 & 0.182 & 0.273 \\ -0.375 & 0.188 & 0 & 0.250 \\ 0.500 & 0.143 & 0.286 & 0 \end{pmatrix}, \ x^{(0)} = \beta = \begin{pmatrix} 1.400 \\ 0.909 \\ 0.938 \\ 0.071 \end{pmatrix},$$

$$\varepsilon^{(0)} = \frac{q}{1-q}\left\|x^{(0)}\right\| = \frac{0.929}{0.071} \cdot 1.4 = 18.200 > \varepsilon .$$

Then, the iterative process of the fixed-point iteration method goes as follows:

$$x^{(1)} = \beta + \alpha x^{(0)}, \ \begin{cases} x_1^{(1)} = 1.400 + 0.200 \cdot 0.909 - 0.400 \cdot 0.938 - 0.200 \cdot 0.071 = 1.193 \\ x_2^{(1)} = 0.909 - 0.364 \cdot 1.400 + 0.182 \cdot 0.938 + 0.273 \cdot 0.071 = 0.590 \\ x_3^{(1)} = 0.938 - 0.375 \cdot 1.400 + 0.188 \cdot 0.909 + 0.250 \cdot 0.071 = 0.601 \\ x_4^{(1)} = 0.071 + 0.500 \cdot 1.400 + 0.143 \cdot 0.909 + 0.286 \cdot 0.938 = 1.169 \end{cases},$$

$\varepsilon^{(1)} = \frac{q}{1-q}\left\|x^{(1)} - x^{(0)}\right\| = 14.270 > \varepsilon$; further in the same way:

$$x^{(2)} = \beta + \alpha x^{(1)} = \begin{pmatrix} 1.044 \\ 0.904 \\ 0.893 \\ 0.924 \end{pmatrix}, \ \varepsilon^{(2)} = 4.077 > \varepsilon; \ x^{(3)} = \beta + \alpha x^{(2)} = \begin{pmatrix} 1.039 \\ 0.944 \\ 0.946 \\ 0.978 \end{pmatrix}, \ \varepsilon^{(3)} = 0.702 > \varepsilon;$$

$$x^{(4)} = \beta + \alpha x^{(3)} = \begin{pmatrix} 1.015 \\ 0.970 \\ 0.969 \\ 0.996 \end{pmatrix}, \ \varepsilon^{(4)} = 0.341 > \varepsilon; \ x^{(5)} = \beta + \alpha x^{(4)} = \begin{pmatrix} 1.007 \\ 0.988 \\ 0.988 \\ 0.994 \end{pmatrix}, \ \varepsilon^{(5)} = 0.241 > \varepsilon;$$

$$x^{(6)} = \beta + \alpha x^{(5)} = \begin{pmatrix} 1.004 \\ 0.994 \\ 0.994 \\ 0.998 \end{pmatrix},\ \varepsilon^{(6)} = 0.075 < \varepsilon .$$

So, the computational process is completed in 6 iterations. Solution to the SLAE: $x^* \approx x^{(6)} = \begin{pmatrix} 1.004 & 0.994 & 0.994 & 0.998 \end{pmatrix}^T$.

Let's note that the exact solution to the original SLAE in this case is known: $x^* = \begin{pmatrix} 1 & 1 & 1 & 1 \end{pmatrix}^T$. This implies that the solution obtained at the fifth iteration already satisfies the given accuracy $\varepsilon = 0.2$: $\left\| x^{(5)} - x^* \right\| < \varepsilon$. But by virtue of using the evaluating expression for error calculation, the process stops only at the sixth iteration (as $\varepsilon^{(5)} > \varepsilon$).

Let's also note that the a-priori estimate of the required number of iterations in this problem gives $k \geq \dfrac{\lg 0.2 - \lg 1.4 + \lg 0.071}{\lg 0.929} - 1 = 60.869$, which is an inadequately overestimated value.

The iterative process of the Seidel method goes as follows.

The convergence condition $q = \|\alpha\|_c = 0.929 < 1$ and the initial approximation are the same.

$$x^{(0)} = \beta = \begin{pmatrix} 1.400 \\ 0.909 \\ 0.938 \\ 0.071 \end{pmatrix},\ \begin{cases} x_1^{(1)} = 1.400 + 0.200 \cdot 0.909 - 0.400 \cdot 0.938 - 0.200 \cdot 0.071 = 1.193 \\ x_2^{(1)} = 0.909 - 0.364 \cdot 1.193 + 0.182 \cdot 0.938 + 0.273 \cdot 0.071 = 0.665 \\ x_3^{(1)} = 0.938 - 0.375 \cdot 1.193 + 0.188 \cdot 0.665 + 0.250 \cdot 0.071 = 0.633 \\ x_4^{(1)} = 0.071 + 0.500 \cdot 1.193 + 0.143 \cdot 0.665 + 0.286 \cdot 0.633 = 0.944 \end{cases},$$

$\varepsilon^{(1)} = \dfrac{q}{1-q} \left\| x^{(1)} - x^{(0)} \right\| = 11.338 > \varepsilon$; further in the same way:

$$x^{(2)} = \begin{pmatrix} 1.091 \\ 0.885 \\ 0.930 \\ 1.009 \end{pmatrix},\ \varepsilon^{(2)} = 3.863 > \varepsilon;\ x^{(3)} = \begin{pmatrix} 1.003 \\ 0.989 \\ 0.999 \\ 1.000 \end{pmatrix},\ \varepsilon^{(3)} = 1.351 > \varepsilon;$$

$$x^{(4)} = \begin{pmatrix} 0.998 \\ 1.000 \\ 1.001 \\ 0.999 \end{pmatrix},\ \varepsilon^{(4)} = 0.152 < \varepsilon .$$

So, the computational process is completed in 4 iterations. Solution to the SLAE: $x^* \approx x^{(4)} = \begin{pmatrix} 0.998 & 1.000 & 1.001 & 0.999 \end{pmatrix}^T$.

It is easy to see that the Seidel method in this case converges faster than the fixed-point iteration method.

2.12. PROGRAM #03

Below is a proposed variant of the program algorithm for solving SLAE using the fixed-point iteration method or the Seidel method.

```
ALGORITHM “Fixed-point iteration method or Seidel method”
INPUT     n, c[n][n], d[n], e
OUTPUT    i, j, f, g, h, a[n][n], b[n], y[n], x[n], k
BEGIN
    #Reduction to an equivalent form using the Jacobi method
    CYCLE “Rows” FOR i FROM 1 TO n BY 1
        IF (c[i][i]=0) END
        CYCLE “Columns” FOR j FROM 1 TO n BY 1
            IF (i≠j) a[i][j]:=-c[i][j]/c[i][i] ELSE a[i][j]:=0
        b[i]:=d[i]/c[i][i]
    #Calculation of the norm of the matrix
    g:=0
    CYCLE “Rows” FOR i FROM 1 TO n BY 1
        h:=0
        CYCLE “Columns” FOR j FROM 1 TO n BY 1
            IF (a[i][j]>0) h:=h+a[i][j] ELSE h:=h-a[i][j]
        IF (h>g) g:=h
    IF (g>=1)
        PRINT “The convergence condition is not fulfilled!”
        END
    f:=0
    CYCLE “Rows” FOR i FROM 1 TO n BY 1
```

```
            x[i]:=b[i]
            IF (x[i]>f) f:=x[i]
            IF (-x[i]>f) f:=-x[i]
        f:=f*g/(1-g)
        k:=0
        CYCLE “Iteration” WHILE (f>e)
            CYCLE “Rows” FOR i FROM 1 TO n BY 1
                y[i]:=x[i]
            CYCLE “Rows” FOR i FROM 1 TO n BY 1
                x[i]:=b[i]
                #For the fixed-point iteration method
                CYCLE “Element” FOR j FROM 1 TO n BY 1
                    x[i]:=x[i]+a[i][j]*y[j]
                #Alternatively for the Seidel method
                CYCLE “Element” FOR j FROM 1 TO i-1 BY 1
                    x[i]:=x[i]+a[i][j]*x[j]
                CYCLE “Element” FOR j FROM i TO n BY 1
                    x[i]:=x[i]+a[i][j]*y[j]
            f:=0
            CYCLE “Rows” FOR i FROM 1 TO n BY 1
                IF (x[i]-y[i]>f) f:=x[i]-y[i]
                IF (y[i]-x[i]>f) f:=y[i]-x[i]
            f:=f*g/(1-g)
            k:=k+1
        PRINT x, k
END
```