Comparison of Two Stationary Iterative Methods

Oleksandra Osadcha
Faculty of Applied Mathematics
Silesian University of Technology
Gliwice, Poland
Email:olekosa338@student.polsl.pl

***Abstract*—This paper illustrates comparison of two stationary iteration methods for solving systems of linear equations. The aim of the research is to analyze, which method is faster solve this equations and how many iteration required each method for solving. This paper present some ways to deal with this problem.**

***Index Terms*—comparison, analisys, stationary methods, Jacobi method, Gauss-Seidel method**

I. Introduction

The term "iterative method" refers to a wide range of techniques that use successive approximations to obtain more accurate solutions to a linear system at each step. In this publication we will cover two types of iterative methods. Stationary methods are older, simpler to understand and implement, but usually not as effective. Nonstationary methods are a relatively recent development; their analysis is usually harder to understand, but they can be highly effective. The nonstationary methods we present are based on the idea of sequences of orthogonal vectors. (An exception is the Chebyshev iteration method, which is based on orthogonal polynomials.)

The rate at which an iterative method converges depends greatly on the spectrum of the coeffcient matrix. Hence, iterative methods usually involve a second matrix that transforms the coeffcient matrix into one with a more favorable spectrum. The transformation matrix is called a preconditioner. A good preconditioner improves the convergence of the iterative method, suffciently to overcome the extra cost of constructing and applying the preconditioner. Indeed, without a preconditioner the iterative method may even fail to converge [1].

There are a lot of publications with stationary iterative methods. As is well known, a real symmetric matrix can be transformed iteratively into diagonal form through a sequence of appropriately chosen elementary orthogonal transformations (in the following called Jacobi rotations): $A_k \rightarrow A_{k+1} = U_k^T A_k U_k \quad (A_0 = given\, matrix)$, this transformations presented in [2]. The special cyclic Jacobi method for computing the eigenvalues and eigenvectors of a real symmetric matrix annihilates the off-diagonal elements of the matrix successively, in the natural order, by rows or columns. Cyclic Jacobi method presented in [4].

Copyright held by the author(s).

Some convergence result for the block Gauss-Seidel method for problems where the feasible set is the Cartesian product of m closed convex sets, under the assumption that the sequence generated by the method has limit points presented in.[5]. In [9] presented improving the modiefied Gauss-Seidel method for Z-matrices. In [11] presented the convergence Gauss-Seidel iterative method, because matrix of linear system of equations is positive definite, but it does not afford a conclusion on the convergence of the Jacobi method. However, it also proved that Jacobi method also converges.
Also these methods were present in [12], where described each method.

This publication present comparison of Jacobi and Gauss-Seidel methods. These methods are used for solving systems of linear equations. We analyze, how many iterations required each method and which method is faster.

II. Jacobi method

We consider the system of linear equations

$$Ax = B$$

where $A \in M_{n \times n}$ i $detA \neq 0$.
We write the matrix in form of a sum of three matricies:

$$A = L + D + U$$

where:

- L - the strictly lower-triangular matrix
- D - the diagonal matrix
- U - the strictly upper-triangular matrix

If the matrix has next form:

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix}$$

then

$$L = \begin{bmatrix} 0 & 0 & \dots & 0 \\ a_{21} & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & 0 \end{bmatrix}$$

$$D = \begin{bmatrix} a_{11} & 0 & \dots & 0 \\ 0 & a_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & a_{nn} \end{bmatrix}$$

$$U = \begin{bmatrix} 0 & a_{12} & \dots & a_{1n} \\ 0 & 0 & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \end{bmatrix}$$

If the matrix A is nonsingular ($detA \neq 0$), then we can rearrange its rows, so that on the diagonal there are no zeros, so $a_{ii} \neq 0$, $\forall_{i \in (1,2,\dots,n)}$

If the diagonal matrix D is nonsingular matrix (so $a_{ii} \neq 0$, $i \in 1, \dots, n$), then inverse matrix has following form:

$$D^{-1} = \begin{bmatrix} \frac{1}{a_{11}} & 0 & \dots & 0 \\ 0 & \frac{1}{a_{22}} & \dots & \vdots \\ \vdots & \vdots & \ddots & 0 \\ 0 & \dots & 0 & \frac{1}{a_{nn}} \end{bmatrix}$$

By substituting the above distribution for the system of equations, we obtain:

$$(L + D + U)x = b$$

After next convertions, we have:

$$Dx = -(L + U)x + b$$

$$x = -D^{-1}(L + U)x + D^{-1}b$$

Using this, we get the iterative scheme of the Jacobi method:

$$\begin{cases} x^{i+1} = -D^{-1}(L+U)x^i + D^{-1}b \\ x^0 - \text{initial approximation} \end{cases} \qquad i = 0, 1, 2 \dots$$

Matrix $-D^{-1}(L+U)$ and vector $D^{-1}b$ do not change during calculations.

Algorithm of Jacobi method:

1) Data :
   - Matrix A
   - vector b
   - vector x^0
   - approximation ε ; $(\varepsilon > 0)$
2) We set matrices $L + U$ i D^{-1}
3) We count matrix $M = -D^{-1}(L + U)$ and vector $w = D^{-1}b$
4) We count $x^{i+1} = Mx^i + w$ so long until $\left|\left|Ax^{i+1} - b\right|\right| \geqslant \varepsilon$
5) The result is the last counted x^{i+1}

Example. *Use Jacobi's method to solve the system of equations. The zero vector is assumed as the initial approximation.*

$$\begin{cases} 4x - y = 2 \\ -x + 4y - z = 6 \\ -y - 4z = 2 \end{cases} \qquad v^0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

We have:

$$A = \begin{bmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{bmatrix} \qquad b = \begin{bmatrix} 2 \\ 6 \\ 2 \end{bmatrix}$$

$$L + U = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 0 & -1 \\ 0 & -1 & 0 \end{bmatrix} \qquad D^{-1} \begin{bmatrix} \frac{1}{4} & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ 0 & 0 & -\frac{1}{4} \end{bmatrix}$$

We count

$$M = -D^{-1} \cdot (L + U) = \begin{bmatrix} -\frac{1}{4} & 0 & 0 \\ 0 & -\frac{1}{4} & 0 \\ 0 & 0 & \frac{1}{4} \end{bmatrix} \cdot$$

$$\cdot \begin{bmatrix} 0 & -1 & 0 \\ -1 & 0 & -1 \\ 0 & -1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{4} & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & -\frac{1}{4} & 0 \end{bmatrix}$$

$$W = D^{-1} \cdot b = \begin{bmatrix} \frac{1}{4} & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ 0 & 0 & -\frac{1}{4} \end{bmatrix} \cdot \begin{bmatrix} 2 \\ 6 \\ 2 \end{bmatrix} = \begin{bmatrix} \frac{1}{2} \\ \frac{3}{2} \\ -\frac{1}{2} \end{bmatrix}$$

The iterative scheme has the next form:

$$v^{i+1} = Mv^i + w = \begin{bmatrix} 0 & \frac{1}{4} & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & -\frac{1}{4} & 0 \end{bmatrix} \cdot \begin{bmatrix} x^i \\ y^i \\ z^i \end{bmatrix} + \begin{bmatrix} \frac{1}{2} \\ \frac{3}{2} \\ -\frac{1}{2} \end{bmatrix}$$

i=0

$$v^1 = \begin{bmatrix} 0 & \frac{1}{4} & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & -\frac{1}{4} & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} \frac{1}{2} \\ \frac{3}{2} \\ -\frac{1}{2} \end{bmatrix} = \begin{bmatrix} \frac{1}{2} \\ \frac{3}{2} \\ -\frac{1}{2} \end{bmatrix}$$

$$||Av^i - b|| = \left|\left| \begin{bmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{2} \\ \frac{3}{2} \\ -\frac{1}{2} \end{bmatrix} - \begin{bmatrix} 2 \\ 6 \\ 2 \end{bmatrix} \right|\right| =$$

$$= \left|\left| \begin{bmatrix} \frac{1}{2} \\ 6 \\ \frac{1}{2} \end{bmatrix} - \begin{bmatrix} 2 \\ 6 \\ 2 \end{bmatrix} \right|\right| = \left|\left| \begin{bmatrix} -\frac{3}{2} \\ 0 \\ -\frac{3}{2} \end{bmatrix} \right|\right| =$$

$$= \sqrt{\left(-\frac{3}{2}\right)^2 + 0^2 + \left(-\frac{3}{2}\right)^2} = \frac{3}{2}\sqrt{2}$$

i=1

$$v^1 = \begin{bmatrix} 0 & \frac{1}{4} & 0 \\ \frac{1}{4} & 0 & \frac{1}{4} \\ 0 & -\frac{1}{4} & 0 \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{2} \\ \frac{3}{2} \\ -\frac{1}{2} \end{bmatrix} + \begin{bmatrix} \frac{1}{2} \\ \frac{3}{2} \\ -\frac{1}{2} \end{bmatrix} = \begin{bmatrix} \frac{3}{8} \\ 0 \\ -\frac{3}{8} \end{bmatrix} + \begin{bmatrix} \frac{1}{2} \\ \frac{3}{2} \\ -\frac{1}{2} \end{bmatrix} =$$

$$= \begin{bmatrix} \frac{7}{8} \\ \frac{3}{2} \\ -\frac{7}{8} \end{bmatrix}$$

$$||Av^i - b|| = \left|\left| \begin{bmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{bmatrix} \cdot \begin{bmatrix} \frac{7}{8} \\ \frac{3}{2} \\ -\frac{7}{8} \end{bmatrix} - \begin{bmatrix} 2 \\ 6 \\ 2 \end{bmatrix} \right|\right| =$$

$$= \left|\left| \begin{bmatrix} 2 \\ 6 \\ 2 \end{bmatrix} - \begin{bmatrix} 2 \\ 6 \\ 2 \end{bmatrix} \right|\right| = \left|\left| \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \right|\right| = 0$$

III. Gauss-Seidel method

Analogously to the Jacobi method, in the Gauss-Seidel method, we write the matrix A in the form of a sum:

$$A = L + D + U$$

where matrices L,D and U they are defined in the same way as in the Jacobi method. Next we get:

$$Dx = -Lx - Ux + b$$

where

$$x = -D^{-1}Lx - D^{-1}Ux + D^{-1}b$$

Based on the above formula, we can write an iterative scheme of the Gauss-Seidel method:

$$\begin{cases} x^{i+1} = -D^{1}Lx^{i+1} - D^{-1}Ux + D^{-1}b \\ x^0 - \ initial\ approximation \end{cases} \qquad i = 0, 1, 2 \ldots$$

In the Jacobi method, we use the corrected values of subsequent coordinates of the solution only in the next step of the iteration. In the Gauss - Siedel method, these corrected values are used immediately when we count the next coordinates. The formula of Gauss-Seidel method for coordinates has next form

$$x_k^{i+1} = \frac{1}{a_{kk}} \sum_{j=1}^{k-1} a_{kj} x_j^{i+1} -$$

$$-\frac{1}{a_{kk}} \sum_{j=k=1}^{n} a_{kj} x_j^i + \frac{1}{a_{kk}} b_k \qquad k = 1, 2, \ldots, n$$

Algorithm of Gauss-Seidel method:

1) Data :
   - Matrix A
   - vector b
   - vector x^0
   - approximation ε ; $(\varepsilon > 0)$
2) We count as long as $||Ax^{i+1} - b|| \geqslant \varepsilon$:
   For $k = 1, 2, \ldots, n$
   $$x_k^{i+1} = \frac{1}{a_{kk}} \sum_{j=1}^{k-1} a_{kj} x_j^{i+1} - \frac{1}{a_{kk}} \sum_{j=k=1}^{n} a_{kj} x_j^i + \frac{1}{a_{kk}} b_k$$
3) Result: last count x^{i+1}

Example. Calculate the third approximation of $Ax = b$

$$A = \begin{bmatrix} 3 & -1 & 1 \\ 1 & 4 & 1 \\ 0 & -1 & -2 \end{bmatrix} \qquad b = \begin{bmatrix} 3 \\ -2 \\ -3 \end{bmatrix} \qquad x^0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$L = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix} \qquad D^{-1} = \begin{bmatrix} \frac{1}{3} & 0 & 0 \\ 0 & -\frac{1}{4} & 0 \\ 0 & 0 & -\frac{1}{2} \end{bmatrix}$$

$$U = \begin{bmatrix} 0 & -1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

$$M_1 = -D^{-1} \cdot L = \begin{bmatrix} -\frac{1}{3} & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ 0 & 0 & \frac{1}{2} \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix} =$$

$$= \begin{bmatrix} 0 & 0 & 0 \\ \frac{1}{4} & 0 & 0 \\ 0 & -\frac{1}{2} & 0 \end{bmatrix}$$

$$M_2 = -D^{-1} \cdot U = \begin{bmatrix} -\frac{1}{3} & 0 & 0 \\ 0 & \frac{1}{4} & 0 \\ 0 & 0 & \frac{1}{2} \end{bmatrix} \cdot \begin{bmatrix} 0 & -1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} =$$

$$= \begin{bmatrix} 0 & \frac{1}{3} & -\frac{1}{3} \\ 0 & 0 & \frac{1}{4} \\ 0 & 0 & 0 \end{bmatrix}$$

$$w = D^{-1} \cdot b = \begin{bmatrix} \frac{1}{3} & 0 & 0 \\ 0 & -\frac{1}{4} & 0 \\ 0 & 0 & -\frac{1}{2} \end{bmatrix} \cdot \begin{bmatrix} 3 \\ -2 \\ -3 \end{bmatrix} = \begin{bmatrix} 1 \\ \frac{1}{2} \\ \frac{3}{2} \end{bmatrix}$$

The iterative scheme has following form:

$$\begin{bmatrix} x_1^{i+1} \\ x_2^{i+1} \\ x_3^{i+1} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ \frac{1}{4} & 0 & 0 \\ 0 & -\frac{1}{2} & 0 \end{bmatrix} \cdot \begin{bmatrix} x_1^{i+1} \\ x_2^{i+1} \\ x_3^{i+1} \end{bmatrix} +$$

$$+ \begin{bmatrix} 0 & \frac{1}{3} & -\frac{1}{3} \\ 0 & 0 & \frac{1}{4} \\ 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} x_1^{i} \\ x_2^{i} \\ x_3^{i} \end{bmatrix} + \begin{bmatrix} 1 \\ \frac{1}{2} \\ \frac{3}{2} \end{bmatrix}$$

first step (i=0)

$$x_1^1 = \frac{1}{3}x_2^0 - \frac{1}{3}x_3^0 + 1 = 1$$

$$x_2^1 = \frac{1}{4}x_1^1 + \frac{1}{4}x_3^0 + \frac{1}{2} = \frac{1}{4} + 0 + \frac{1}{2} = \frac{3}{4}$$

$$x_3^1 = -\frac{1}{2}x_2^1 + \frac{3}{2} = \frac{9}{8}$$

$$x^1 = \left(1, \frac{3}{4}, \frac{9}{8}\right)^T$$

$$||Ax^1 - b|| = \frac{3}{4}\sqrt{\frac{5}{2}}$$

second step (i=1)

$$x_1^2 = \frac{1}{3}x_2^1 - \frac{1}{3}x_3^1 + 1 = \frac{1}{3} \cdot \frac{3}{4} - \frac{1}{3} \cdot \frac{9}{8} + 1 = \frac{7}{8}$$

$$x_2^2 = \frac{1}{4}x_1^2 + \frac{1}{4}x_3^1 + \frac{1}{2} = \frac{1}{4} \cdot \frac{7}{8} + \frac{1}{4} \cdot \frac{9}{8} + \frac{1}{2} = 1$$

$$x_3^2 = -\frac{1}{2}x_2^2 + \frac{3}{2} = -\frac{1}{2} \cdot 1 + \frac{3}{2} = 1$$

$$x^2 = \left(\frac{7}{8}, 1, 1\right)^T$$

$$||Ax^2 - b|| = \frac{1}{4}\sqrt{\frac{5}{2}}$$

third step (i=2)

$$x_1^3 = \frac{1}{3}x_2^2 - \frac{1}{3}x_3^2 + 1 = 1$$

Tab. I
COMPARISON OF METHODS

		Gauss-Seidel method			Jacobi method		
N	n	Iteration	Time in ms	Time in ti	Iteration	Time in ms	Time in ti
5	25	28	5	10639	54	12	22360
10	100	103	53	98264	205	65	121525
15	225	229	504	933331	451	528	978357
20	400	405	2370	4388630	808	3769	6977776
25	625	636	9147	16933626	1266	9506	17597628
30	900	918	27755	51379812	1839	30047	55623008

$$x_2^3 = \frac{1}{4}x_1^3 + \frac{1}{4}x_3^2 + \frac{1}{2} = \frac{1}{4} + \frac{1}{4} + \frac{1}{2} = 1$$

$$x_3^3 = -\frac{1}{2}x_2^3 + \frac{3}{2} = 1$$

$$x^3 = \Big(1, 1, 1\Big)^T$$

$$\Big|\Big|Ax^3 - b\Big|\Big| = 0$$

IV. COMPARISON OF METHODS

Here we will present differences between these tests. We carry out the series of research, where we analyze how much time require each test and also we compare number of iteration of each test.

A. Analysis of time

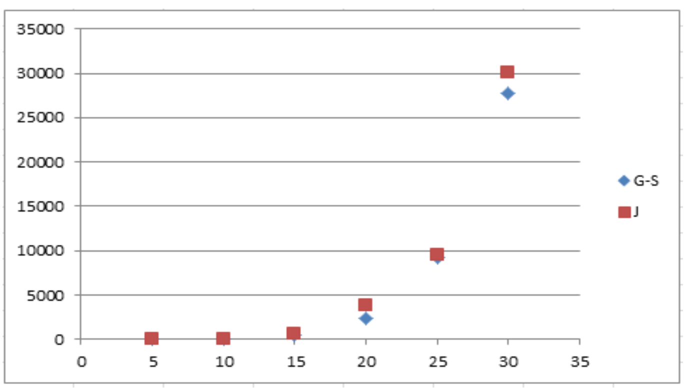

Fig. 1. Graph of time comparison

First, we analize speed of each test. After research, we get data with time in ms and CPU ticks, which presented in Table.I. Also, we prepared graph of time comparison in ms. So, we can conclude that Gauss-Seidel method is faster than Jacobi method.

B. Analisys of efficiency

Next, we analize the efficiency of these tests. We do research for different number of n, where n is size of matrix A.
When we compare the numbers of each tests, we can see that Gauss-Seidel method required less number of iterations than Jacobi method. So, we can conclude, that Gauss-Seidel method has better efficiency.

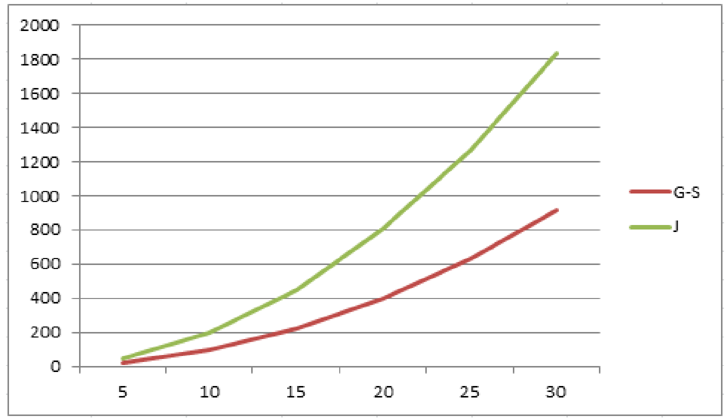

Fig. 2. Graph of iterations number

V. CONCLUSIONS

We have compared two method implemented to solve systems of linear equations. This paper can helps us to decide, which method is better. After all research, we can conclude that Gauss-Seidel method is better method than Jacobi method, because it faster and required less number of iterations. So, if we want to get better results and do it faster, we should use Gauss-Seidel method. Advantages of our proposition is ease and clarity of description of each method with shown algorithms. Also in our paper presented examples of each method, which show how these methods could be used.

REFERENCES

[1] Richard Barrett, Michael Berry, Tony F. Chan, James Demmel, June M. Donato, Jack Dongarra, Victor Eijkhout, Roldan Pozo, Charles Romine and Henk Van der Vorst *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods.* 1994, ISBN: 978-0-89871-328-2

[2] H.Rutishauser *The Jacobi Method for Real Symmetric Matrices* Numerische Mathematik 9, 1-10 (1966)

[3] Marszalek, Z., Marcin, W., Grzegorz, B., Raniyah, W., Napoli, C., Pappalardo, G., and Tramontana, E. *Benchmark tests on improved merge for big data processing.* In Asia-Pacific Conference on Computer Aided System Engineering (APCASE), pp. 96-101. 2015.

[4] H. P. M. Van Kempen *On the Quadratic Convergence of the Special Cyclic Jacobi Method* Numerische Mathematik 9, 19-22 (1966)

[5] L. Grippo, M. Sciandrone *On the convergence of the block nonlinear GaussSeidel method under convex constraints*, Operations Research Letters 26 (2000) 127136

[6] Woniak M, Poap D, Napoli C, Tramontana E. *Graphic object feature extraction system based on cuckoo search algorithm.* Expert Systems with Applications. Volume 66, pp. 20-31, 2016.

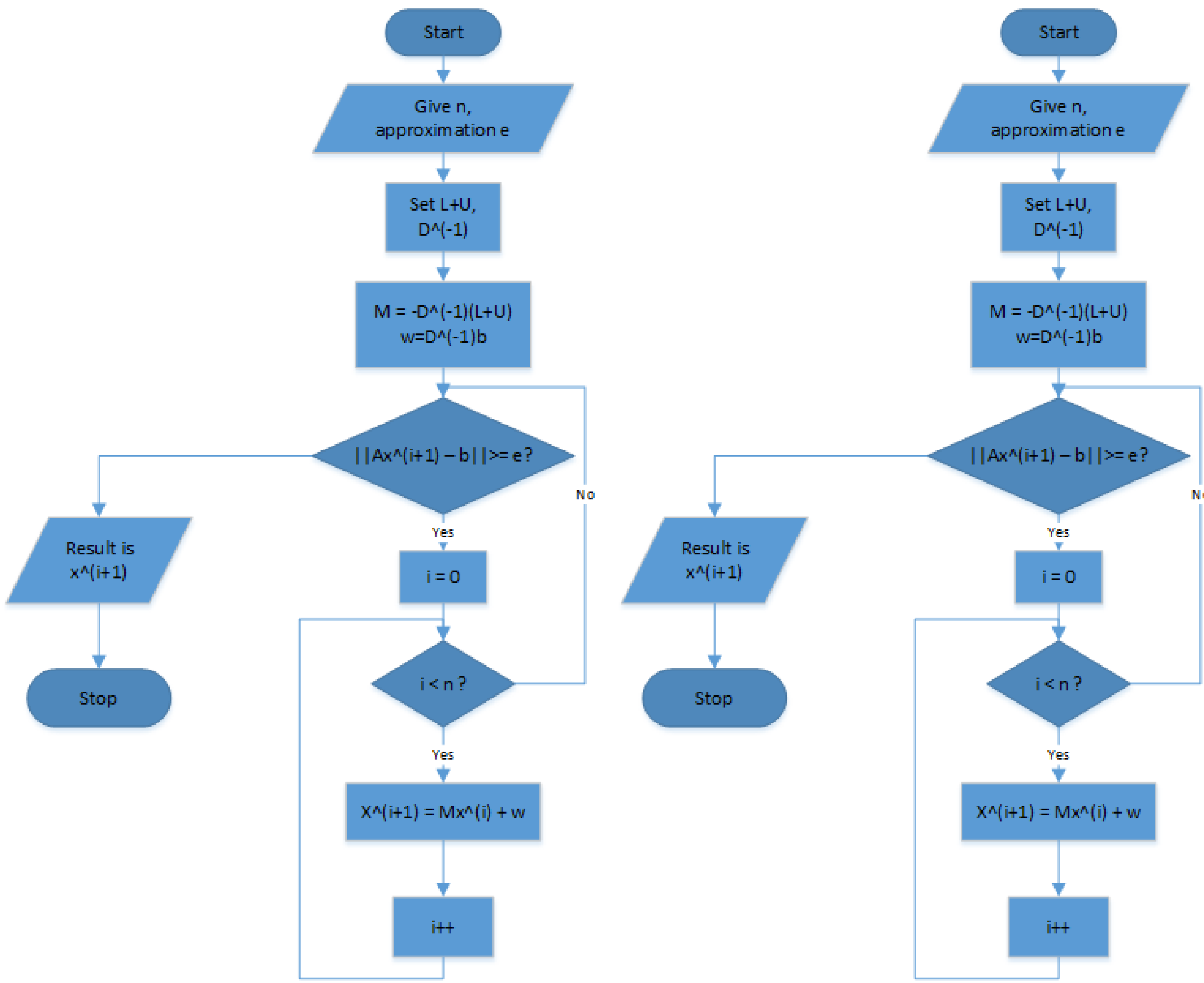

Fig. 3. Jacobi algorithm (left) and Gauss-Seidel algorithm (right).

[7] Woniak M, Poap D, Napoli C, Tramontana E. *Graphic object feature extraction system based on cuckoo search algorithm.* Expert Systems with Applications. Volume 66, pp. 20-31, 2016.

[8] Capizzi, G., Sciuto, G.L., Napoli, C., Tramontana, E. and Woniak, M. *Automatic classification of fruit defects based on co-occurrence matrix and neural networks.* In Federated Conference on Computer Science and Information Systems (FedCSIS), pp. 861-867, 2015.

[9] Toshiyuki Kohno, Hisashi Kotakemori, Hiroshi Niki and Masataka Usui *Improving the Modified Gauss-Seidel Method for Z-Matrices*, 3 January 1997

[10] Wozniak, M., Polap, D., Borowik, G., and Napoli, C. *A first attempt to cloud-based user verification in distributed system.* In Asia-Pacific Conference on Computer Aided System Engineering (APCASE), pp. 226-231, 2015.

[11] A. Mitiche, A.-R. Mansouri *On convergence of the Horn and Schunck optical-flow estimation method*, DOI: 10.1109/TIP.2004.827235, 18 May 2004.

[12] Feng Ding, Tongweng Chen *Iterative least-squares solutions of coupled Sylvester matrix equations*, Systems & Control Letters Volume 54, Issue 2, February 2005, Pages 95-107