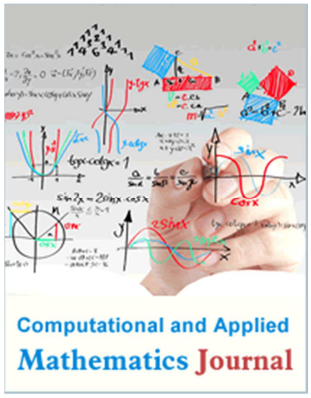
Computational and Applied Mathematics Journal
2015; 1(2): 21-28
Published online February 20, 2015 (http://www.aascit.org/journal/camj)



Keywords
Iterative Method,
Jacobi Method,
Modified Jacobi Method

Received: January 17, 2015
Revised: January 27, 2015
Accepted: January 28, 2015

On Some Iterative Methods for Solving Systems of Linear Equations

Fadugba Sunday Emmanuel

Department of Mathematical Sciences, Ekiti State University, Ado Ekiti, Nigeria

Email address

emmasfad2006@yahoo.com, sunday.fadugba@eksu.edu.ng

Citation

Fadugba Sunday Emmanuel. On Some Iterative Methods for Solving Systems of Linear Equations. *Computational and Applied Mathematics Journal*. Vol. 1, No. 2, 2015, pp. 21-28.

Abstract

This paper presents some iterative methods for solving system of linear equations namely the Jacobi method and the modified Jacobi method. The Jacobi method is an algorithm for solving system of linear equations with largest absolute values in each row and column dominated by the diagonal elements. The modified Jacobi method also known as the Gauss Seidel method or the method of successive displacement is useful for the solution of system of linear equations. The comparative results analysis of the two methods was considered. We also discussed the rate of convergence of the Jacobi method and the modified Jacobi method. Finally, the results showed that the modified Jacobi method is more efficient, accurate and converges faster than its counterpart "the Jacobi Method".

1. Introduction

In computational mathematics, an iterative method attempts to solve a system of linear equations by finding successive approximations to the solution starting from an initial guess. This approach is in contrast to direct methods, which attempt to solve the problem by a finite sequence of some operations and in the absence of rounding errors. These direct methods lead to exact solution. Iterative methods are usually the only choice for nonlinear problems involving a large number of variables where there is no closed form solution or where the problems cannot be solved analytically. Probably the first iterative method for solving a linear system appeared in a letter of Gauss to his student. He proposed a method of solving a 4x4 system of equations. The theory of stationary iterative methods was established with the work of D.M. Young starting in the 1950s. The most well-known and accepted method for solving a system of linear equations is the so called "The Cramer's Rule". However, there are other direct methods for solving systems of linear equations such as the Gauss Elimination, the Gauss Jordan method, LU Decomposition and Tridiagonal matrix algorithm.

But, when solving a system of linear equations with more than four unknown variables (that is, when dealing with a higher order matrix) all these methods become very lengthy and tedious to solve by hand. Therefore there is need for the use of a dedicated coding language like MATLAB.

The paper is structured as follows: Section 2 gives a brief overview of iterative methods for solving systems of linear equations. Section 3 presents some numerical experiments, discussion of results and conclusion.

2. Iterative Methods for Solving Systems of Linear Equations

An iterative method is a mathematical procedure that generates a sequence of improving approximate solutions for a class of problems. A specific implementation of an iterative method, including the termination criteria, is an algorithm of the iterative method.

An iterative method is said to be convergent if the corresponding sequence converges for some given initial approximations. A rigorous convergence analysis of an iterative method is usually performed; however, heuristic based iterative methods are also common.

In this paper we shall consider the system of linear equations of the form

$$Ax = b \tag{1.1}$$

Equation (1.1) can also be written as

$$\left.\begin{aligned} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n &= b_2 \\ \vdots \quad\quad \vdots \quad\quad & \vdots \quad \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n &= b_n \end{aligned}\right\} \tag{1.2}$$

where $a_{11}, a_{12,}\ldots, a_{nn}$ are constant coefficients and $b_1, b_2, b_3, \ldots, b_n$ are given real constants in the system of n-linear algebraic equations in n-unknown variables $x_1, x_2, x_3, \ldots, x_n$. The solution to (1.1) can be classified in the following ways:

- If a system of linear equations is solved and unique values $x_1, x_2, x_3, \ldots, x_n$ are obtained, then the system is said to be consistent and independent.
- If on the other hand when the system has no definite solution, it is said to be inconsistent.
- If the system has infinitely many solutions, then it is said to be consistent and dependent.

In the sequel we shall consider two iterative methods for solving system of linear equations namely; the Jacobi method and the modified Jacob method.

2.1. The Jacobi Iterative Method

In numerical linear algebra, the Jacobi method is an algorithm for determining the solutions of system of linear equations with largest absolute values in each row and column dominated by the diagonal element. Each diagonal element is solved for, and approximate values are substituted, the process is then iterated until it converges. This algorithm is a stripped-down version of the Jacobi transformation method of matrix diagonalization. This method is named after Carl Gustav Jakob Jacobi (1804-1851). Let us consider the general system of linear equations given by (1.1) which can be written in the form:

$$\sum_{j=1}^{n} a_{ij}x_j = b_i,\ i = 1, 2\cdots, n \tag{2.1}$$

The following are the assumptions of the Jacobi method:

- The system given by (2.1) has a unique solution (consistent and independent)
- The coefficient matrix A has no zeros on its main diagonal.

If any of the diagonal entries $a_{11}, a_{22}, a_{33}, \ldots, a_{nn}$ is zero, then rows or columns must be interchanged to obtain a coefficient matrix that has non zero entries on the main diagonal. In Jacobi method, each equation of the system is solved for the component of the solution vector associated with the diagonal element, that is, x_i as follows:

$$\left.\begin{aligned} x_1 &= \frac{1}{a_{11}}\left(b_1 - a_{12}x_2 - \ldots - a_{1n}x_n\right) \\ x_2 &= \frac{1}{a_{22}}\left(b_2 - a_{21}x_1 - \ldots - a_{2n}x_n\right) \\ &\vdots \\ x_n &= \frac{1}{a_{nn}}\left(b_n - a_{n1}x_1 - \ldots - a_{nn-1}x_{n-1}\right) \end{aligned}\right\} \tag{2.2}$$

Equation (2.2) can also be written as

$$x_i = \frac{1}{a_{ii}}\left\{b_i - \sum_{j=1}^{n-1} a_{ij}x_j\right\},\ i = 1, 2\cdots n$$

Setting the initial estimates $x_1 = x_1^0, x_2 = x_2^0, \ldots, x_{n-1} = x_{n-1}^0$. Then (2.2) becomes

$$\left.\begin{aligned} x_1^1 &= \frac{1}{a_{11}}\left(b_1 - a_{12}x_2^0 - \ldots - a_{1n}x_n^0\right) \\ x_2^1 &= \frac{1}{a_{22}}\left(b_2 - a_{21}x_1^0 - \ldots - a_{2n}x_n^0\right) \\ &\vdots \\ x_n^1 &= \frac{1}{a_{nn}}\left(b_n - a_{n1}x_1^0 - \ldots - a_{nn-1}x_{n-1}^0\right) \end{aligned}\right\} \tag{2.3}$$

Equation (2.3) is called the first approximation. In the same way, the second approximation is obtained by substituting the first approximation (x-values) into the right hand side of (2.3), and then we have

$$\left.\begin{aligned} x_1^2 &= \frac{1}{a_{11}}\left(b_1 - a_{12}x_2^1 - \ldots - a_{1n}x_n^1\right) \\ x_2^2 &= \frac{1}{a_{22}}\left(b_2 - a_{21}x_1^1 - \ldots - a_{2n}x_n^1\right) \\ &\vdots \\ x_n^2 &= \frac{1}{a_{nn}}\left(b_n - a_{n1}x_1^1 - \ldots - a_{nn-1}x_{n-1}^1\right) \end{aligned}\right\} \tag{2.4}$$

By repeated iterations, a sequence of approximations that often converges to the actual solution is formed.

In general, the Jacobi iterative method is given by

$$\left.\begin{aligned} x_i^k = \frac{1}{a_{ii}}\left\{b_i - \sum_{j=1}^{n-1} a_{ij} x_j^{k-1}\right\}, i = 1,2,\cdots n \\ k = 1,2,3,... \end{aligned}\right\} \quad (2.5)$$

The summation is taken over all j's except $j=i$. The initial estimate or vector $x^{(0)}$ can be chosen to be zero unless we have a better a-priori estimate for the true solution. This procedure is repeated until some convergence criterion is satisfied. The Jacobi method is sometimes called the method of simultaneous iteration because all the values of x_i are iterated simultaneously. That is, all values of $x_i^{(k)}$ depend only on the values of $x_i^{(k-1)}$.

Remark 1: The Jacobi iterative method in (2.6) can also be written as

$$\left.\begin{aligned} x_i^{k+1} = \frac{1}{a_{ii}}\left\{b_i - \sum_{j=1}^{n-1} a_{ij} x_j^{k}\right\}, i = 1,2,\cdots, n \\ k = 0,1,2,\cdots \end{aligned}\right\} \quad (2.6)$$

The following result summarizes the convergence property of the Jacobi method.

Theorem 1

Suppose that A is diagonal dominant, i.e.

$$|a_{ii}| > \sum_{j:j\neq i} |a_{ij}|, \quad \forall\, i \quad (2.7)$$

then the Jacobi method converges.

Remark 2: We do not give the rigorous proof of the above Theorem 1. We just comment that the condition (2.7) says that in every row the diagonal entry is bigger (in absolute value) than the sum of all the other entries (in absolute value) in the row.

Next, we consider the modified Jacobi method for the solution of system of linear equations.

2.2. The Modified Jacobi Method

In numerical linear algebra, the modified Jacobi method, also known as the Gauss Seidel method, is an iterative method used for solving a system of linear equations. It was named after the German mathematicians Carl Friedrich Gauss (1777-1855) and Philip Ludwig Von Seidel (1821-1896)**.** This method is similar to the Jacobi method. Though it can be applied to any matrix with non-zero elements on the diagonals, convergence is only guaranteed if the matrix is either diagonally dominant, or symmetric and positive definite.

The modified Jacobi method for solving a set of linear equations can be thought of as just an extension of the Jacobi method. Start out using an initial value of zero for each of the parameters. Then, solve for $x_1^{(1)}$ as in the Jacobi method. When solving for $x_2^{(1)}$, insert the just computed value for $x_1^{(1)}$. In other words, for each calculation, the most current estimate of the parameter value is used. The Gauss-Seidel method (Modified Jacobi Method) converges about twice as fast as Jacobi, but may still be very slow.

In the Jacobi method, all values of $x^{(k)}$ are based on $x^{(k-1)}$. The modified Jacobi method is similar to the Jacobi method, except that the most recently computed values of all x_1 are used in all computations. Like the Jacobi method, the modified Jacobi method requires diagonal dominance to ensure convergence. Thus, in the modified Jacobi method, each equation of the system is solved for the component of the solution vector associated with the diagonal element, that is, x_i as follows:

$$\left.\begin{aligned} x_1 &= \frac{1}{a_{11}}(b_1 - a_{12}x_2 - ... - a_{1n}x_n) \\ x_2 &= \frac{1}{a_{22}}(b_2 - a_{21}x_1 - ... - a_{2n}x_n) \\ &\vdots \\ x_n &= \frac{1}{a_{nn}}(b_n - a_{n1}x_1 - ... - a_{nn-1}x_{n-1}) \end{aligned}\right\} \quad (2.8)$$

Equation (2.2) can also be written as

$$x_i = \frac{1}{a_{ii}}\left\{b_i - \sum_{j=1}^{n-1} a_{ij} x_j\right\}, i = 1,2\cdots n$$

Setting the initial estimates $x_1 = x_1^0, x_2 = x_2^0, ..., x_{n-1} = x_{n-1}^0$ be the initial values. Then (2.8) becomes

$$\left.\begin{aligned} x_1^1 &= \frac{1}{a_{11}}\left(b_1 - a_{12}x_2^0 - a_{13}x_3^0 ... - a_{1n}x_n^0\right) \\ x_2^1 &= \frac{1}{a_{22}}\left(b_2 - a_{21}x_1^1 - a_{23}x_3^0 ... - a_{2n}x_n^0\right) \\ &\vdots \\ x_n^1 &= \frac{1}{a_{nn}}\left(b_n - a_{n1}x_1^1 - a_{n2}x_2^1 ... - a_{nn-1}x_{n-1}^1\right) \end{aligned}\right\} \quad (2.9)$$

Equation (2.9) is called the first approximation. In the same way, the second approximation is formed by substituting the first approximation (x-values) into the right hand side of (2.9), and then we have

$$\left.\begin{aligned} x_1^2 &= \frac{1}{a_{11}}\left(b_1 - a_{12}x_2^1 - a_{13}x_3^1 ... - a_{1n}x_n^1\right) \\ x_2^2 &= \frac{1}{a_{22}}\left(b_2 - a_{21}x_1^2 - a_{23}x_3^1 ... - a_{2n}x_n^1\right) \\ &\vdots \\ x_n^2 &= \frac{1}{a_{nn}}\left(b_n - a_{n1}x_1^2 - a_{n2}x_2^2 ... - a_{nn-1}x_{n-1}^2\right) \end{aligned}\right\} \quad (2.10)$$

By repeated iterations, a sequence of approximations that often converges to the actual solution is formed. In general,

the modified Jacobi algorithm is obtained from the Jacobi algorithm, (2.5), by using $x_j^{(k)}$ values in the summation from (*j = 1 to i – 1*), (assuming the sweeps through the equations proceed from *i = 1 to n*) is given by

$$x_i^k = \frac{1}{a_{ii}}\left\{ b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{k-1} - \sum_{j=i+1}^{n} a_{ij} x_j^{k-1} \right\}, \begin{matrix} i = 1,2,\cdots n, \\ k = 1,2,3,... \end{matrix} \quad (2.11)$$

The modified Jacobi method is sometimes called the method of successive iteration because the most recent values of all x_i are used in all the calculations. The modified Jacobi method which is also called the Gauss-Seidel method is generally converges faster than the Jacobi method.

2.2.1. Theorem 2

The modified Jacobi method is said to be convergent if the matrix *A* is strictly diagonally dominant i.e.

$$|a_{ii}| > \sum_{j:j\neq i} |a_{ij}|, \quad \forall\, i \quad (2.12)$$

The speed of convergence of the modified Jacobi iteration is given by the largest eigenvalue of $(L+D)^{-1}U$ where *L* is called the Lower triangular matrix containing all element of *A* strictly below the diagonal, *U* is called the Upper triangular matrix containing all element of *A* strictly above the diagonal and *D* is called Diagonal matrix containing the diagonal elements of *A* respectively.

2.2.2. Comments

- An $n \times n$ matrix *A* is said to be *strictly diagonally dominant* if the absolute value of each entry on the main diagonal is greater than the sum of the absolute values of each of the other entries in the same row. i.e.

$$\left.\begin{array}{l} |a_{11}| > |a_{12}| + |a_{13}| + ... + |a_{1n}| \\ |a_{22}| > |a_{21}| + |a_{23}| + ... + |a_{2n}| \\ |a_{33}| > |a_{31}| + |a_{32}| + ... + |a_{3n}| \\ \vdots \quad\quad \vdots \quad\quad \vdots \quad\quad \vdots \\ |a_{nn}| > |a_{n1}| + |a_{n2}| + ... + |a_{nn-1}| \end{array}\right\} \quad (2.13)$$

- If *A* is strictly diagonally dominant, then the system of linear equations in (1.1) has a unique solution in which the Jacobi method and the modified Jacobi method will converge for any initial approximation.
- Most of these iterative techniques entail a process that converts the system in (1.1) into an equivalent system of the form $x = Tx + c$ for some $n \times n$ matrix *T* and vector *c*. In general, we can write the iteration method for solving a system of linear equations in the form

$$x^k = Tx^{k-1} + c, \quad k = 0,1,2,... \quad (2.14)$$

where *T* is called the iteration matrix and depends on *A*, *c* is a column vector which depends on *A* and *b*.

- Iterative methods are generally used when the system is large (*n>50*) and the matrix is sparse *(A matrix with very few non-zero entries)*.

2.3. Accuracy and Convergence of Iterative Methods

All non-singular systems of linear algebraic equations have an exact solution. In principle, when solved by direct methods, the exact solution can be obtained. However, all real calculations are performed with finite precession numbers, so round-off errors pollute the solution. Iterative methods are less susceptible to round-off errors than direct elimination methods for three reasons:

- The system of linear equations is strictly diagonally dominant
- The System of linear equations is typically sparse
- Each iteration through the system of linear equations is independent of the round-off errors of the previous iteration.

When solved by iterative methods, the exact solution of a system of linear algebraic equations is approached asymptotically as the number of iterations increases. When the number of iterations increases without bound, the numerical solution yields the exact solution within the round-off limit of the computing device. Such solutions are said to be corrected to machine accuracy. In most practical solutions, machine accuracy is not required. Thus, the iterative process should be terminated when some type of accuracy criterion (or criteria) has been satisfied in iterative methods, the term accuracy refers to the number of significant figures obtained in the calculations, and the term convergence refers to the point in the iterative process when the desired accuracy is obtained.

3. Numerical Experiments

This section presents some numerical experiments and discussion of results.

3.1. Numerical Experiments

3.1.1. Problem 1

Let us consider the systems of linear equations of the form:
We consider the system of linear equations of the form

$$\left.\begin{array}{l} 5x_1 - 2x_2 + 3x_3 = -1 \\ -3x_1 + 9x_2 + x_3 = 2 \\ 2x_1 - x_2 - 7x_3 = 3 \end{array}\right\} \quad (2.15)$$

3.1.2. Problem 2

$$\left.\begin{array}{l} -2x_1 + x_2 = 5 \\ x_1 - 2x_2 + x_3 = 1 \\ x_2 - 2x_3 + x_4 = 0 \\ x_3 - 2x_4 = 8 \end{array}\right\} \quad (2.16)$$

Solving (2.15) and (2.16) using the two methods under consideration, we have the following results as shown in the Tables below.

3.2. Table of Results

Table 1. *The Jacobi Method for Problem 1*

n	0	1	2	3	4	5	6	7
x_1	0.000	-0.200	0.146	0.192	0.181	0.185	0.186	0.186
x_2	0.000	0.222	0.203	0.328	0.328	0.329	0.331	0.331
x_3	0.000	-0.429	-0.517	-0.416	-0.421	-0.424	-0.423	-0.423

Table 2. *The Modified Jacobi Method for Problem 1*

n	0	1	2	3	4	5
x_1	0.000	-0.200	0.167	0.191	0.186	0.186
x_2	0.000	0.156	0.334	0.323	0.331	0.331
x_3	0.000	-0.508	-0.429	-0.422	-0.423	-0.423

Table 3. *The Jacobi Method for Problem 2*

n	x_1	x_2	x_3	x_4
[0]	[0.0000]	[0.0000]	[0.0000]	[0.0000]
[1]	[-2.5000]	[-0.5000]	[0.0000]	[-4.0000]
[2]	[-2.7500]	[-1.7500]	[-2.2500]	[-4.0000]
[3]	[-3.3750]	[-3.0000]	[-2.8750]	[-5.1250]
[4]	[-4.0000]	[-3.6250]	[-4.0625]	[-5.4375]
[5]	[-4.3125]	[-4.5313]	[-4.5313]	[-6.0313]
[6]	[-4.7656]	[-4.9219]	[-5.2813]	[-6.2656]
[7]	[-4.9609]	[-5.5234]	[-5.5938]	[-6.6406]
[8]	[-5.2617]	[-5.7773]	[-6.0820]	[-6.7969]
[9]	[-5.3887]	[-6.1719]	[-6.2871]	[-7.0410]
[10]	[-5.5859]	[-6.3379]	[-6.6064]	[-7.1436]
[11]	[-5.6689]	[-6.5962]	[-6.7407]	[-7.3032]
[12]	[-5.7981]	[-6.7048]	[-6.9497]	[-7.3704]
[13]	[-5.8524]	[-6.8739]	[-7.0376]	[-7.4749]
[14]	[-5.9370]	[-6.9450]	[-7.1744]	[-7.5188]
[15]	[-5.9725]	[-7.0557]	[-7.2319]	[-7.5872]
[16]	[-6.0278]	[-7.1022]	[-7.3214]	[-7.6160]
[17]	[-6.0511]	[-7.1746]	[-7.3591]	[-7.6607]
[18]	[-6.0873]	[-7.2051]	[-7.4177]	[-7.6795]
[19]	[-6.1025]	[-7.2525]	[-7.4423]	[-7.7088]
[20]	[-6.1262]	[-7.2724]	[-7.4807]	[-7.7212]
[21]	[-6.1362]	[-7.3035]	[-7.4968]	[-7.7403]
[22]	[-6.1517]	[-7.3165]	[-7.5219]	[-7.7484]
[23]	[-6.1583]	[-7.3368]	[-7.5325]	[-7.7609]
[24]	[-6.1684]	[-7.3454]	[-7.5489]	[-7.7662]
[25]	[-6.1727]	[-7.3586]	[-7.5558]	[-7.7744]
[26]	[-6.1793]	[-7.3642]	[-7.5665]	[-7.7779]
[27]	[-6.1821]	[-7.3729]	[-7.5711]	[-7.7833]
[28]	[-6.1865]	[-7.3766]	[-7.5781]	[-7.7855]
[29]	[-6.1883]	[-7.3823]	[-7.5811]	[-7.7891]
[30]	[-6.1911]	[-7.3847]	[-7.5857]	[-7.7905]
[31]	[-6.1923]	[-7.3884]	[-7.5876]	[-7.7928]
[32]	[-6.1942]	[-7.3900]	[-7.5906]	[-7.7938]
[33]	[-6.1950]	[-7.3924]	[-7.5919]	[-7.7953]
[34]	[-6.1962]	[-7.3934]	[-7.5939]	[-7.7959]
[35]	[-6.1967]	[-7.3950]	[-7.5947]	[-7.7969]
[36]	[-6.1975]	[-7.3957]	[-7.5960]	[-7.7973]
[37]	[-6.1979]	[-7.3967]	[-7.5965]	[-7.7980]
[38]	[-6.1984]	[-7.3972]	[-7.5974]	[-7.7983]
[39]	[-6.1986]	[-7.3979]	[-7.5977]	[-7.7987]
[40]	[-6.1989]	[-7.3982]	[-7.5983]	[-7.7989]
[41]	[-6.1991]	[-7.3986]	[-7.5985]	[-7.7991]
[42]	[-6.1993]	[-7.3988]	[-7.5989]	[-7.7993]
[43]	[-6.1994]	[-7.3991]	[-7.5990]	[-7.7994]
[44]	[-6.1995]	[-7.3992]	[-7.5993]	[-7.7995]
[45]	[-6.1996]	[-7.3994]	[-7.5994]	[-7.7996]
[46]	[-6.1997]	[-7.3995]	[-7.5995]	[-7.7997]
[47]	[-6.1997]	[-7.3996]	[-7.5996]	[-7.7998]
[48]	[-6.1998]	[-7.3997]	[-7.5997]	[-7.7998]
[49]	[-6.1998]	[-7.3997]	[-7.5997]	[-7.7998]
[50]	[-6.1999]	[-7.3998]	[-7.5998]	[-7.7999]
[51]	[-6.1999]	[-7.3998]	[-7.5998]	[-7.7999]
[52]	[-6.1999]	[-7.3999]	[-7.5999]	[-7.7999]
[53]	[-6.1999]	[-7.3999]	[-7.5999]	[-7.7999]
[54]	[-6.1999]	[-7.3999]	[-7.5999]	[-7.7999]
[55]	[-6.2000]	[-7.3999]	[-7.5999]	[-7.8000]
[56]	[-6.2000]	[-7.3999]	[-7.5999]	[-7.8000]
[57]	[-6.2000]	[-7.4000]	[-7.5999]	[-7.8000]
[58]	[-6.2000]	[-7.4000]	[-7.6000]	[-7.8000]
[59]	[-6.2000]	[-7.4000]	[-7.6000]	[-7.8000]

Table 4. *The Modified Jacobi Method for Problem 2*

n	x_1	x_2	x_3	x_4
[0]	[0.0000]	[0.0000]	[0.0000]	[0.0000]
[1]	[-2.5000]	[-1.7500]	[-0.8750]	[-4.4375]
[2]	[-3.3750]	[-2.6250]	[-3.5313]	[-5.7656]
[3]	[-3.8125]	[-4.1719]	[-4.9688]	[-6.4844]
[4]	[-4.5859]	[-5.2773]	[-5.8809]	[-6.9404]
[5]	[-5.1387]	[-6.0098]	[-6.4751]	[-7.2375]
[6]	[-5.5049]	[-6.4900]	[-6.8638]	[-7.4319]
[7]	[-5.7450]	[-6.8044]	[-7.1181]	[-7.5591]
[8]	[-5.9022]	[-7.0102]	[-7.2846]	[-7.6423]
[9]	[-6.0051]	[-7.1448]	[-7.3936]	[-7.6968]
[10]	[-6.0724]	[-7.2330]	[-7.4649]	[-7.7324]
[11]	[-6.1165]	[-7.2907]	[-7.5116]	[-7.7558]
[12]	[-6.1453]	[-7.3285]	[-7.5421]	[-7.7711]
[13]	[-6.1642]	[-7.3532]	[-7.5621]	[-7.7811]
[14]	[-6.1766]	[-7.3694]	[-7.5752]	[-7.7876]
[15]	[-6.1847]	[-7.3799]	[-7.5838]	[-7.7919]
[16]	[-6.1900]	[-7.3869]	[-7.5894]	[-7.7947]
[17]	[-6.1934]	[-7.3914]	[-7.5930]	[-7.7965]
[18]	[-6.1957]	[-7.3944]	[-7.5955]	[-7.7977]
[19]	[-6.1972]	[-7.3963]	[-7.5970]	[-7.7985]
[20]	[-6.1982]	[-7.3976]	[-7.5981]	[-7.7990]
[21]	[-6.1988]	[-7.3984]	[-7.5987]	[-7.7994]
[22]	[-6.1992]	[-7.3990]	[-7.5992]	[-7.7996]
[23]	[-6.1995]	[-7.3993]	[-7.5995]	[-7.7997]
[24]	[-6.1997]	[-7.3996]	[-7.5996]	[-7.7998]
[25]	[-6.1998]	[-7.3997]	[-7.5998]	[-7.7999]
[26]	[-6.1999]	[-7.3998]	[-7.5998]	[-7.7999]
[27]	[-6.1999]	[-7.3999]	[-7.5999]	[-7.7999]
[28]	[-6.1999]	[-7.3999]	[-7.5999]	[-7.8000]
[29]	[-6.2000]	[-7.3999]	[-7.6000]	[-7.8000]
[30]	[-6.2000]	[-7.4000]	[-7.6000]	[-7.8000]
[31]	[-6.2000]	[-7.4000]	[-7.6000]	[-7.8000]

3.3. Discussion of Results and Conclusion

In this paper two main iterative methods for solving a system of linear equations have been presented namely the Jacobi method and the modified Jacobi method. Two practical examples were considered, a 3x3 and 4x4 system of linear equations, even though the software can accommodate up to 10x10 systems of linear equations. The analysis of results in Tables 1, 2, 3 and 4 shows that Jacobi method converges at the 7th iteration for 3x3 and 59 iterations to converge for 4x4 matrix. Also the modified Jacobi method takes 5 iterations to converge for 3x3 and 31 iterations to converge for 4x4 matrix. This shows that the modified Jacobi

method requires less computer storage than the Jacobi method. Thus, the modified Jacobi method is more accurate; the numbers of iterations are minima and the rate of its convergence is faster than its counterpart "the Jacobi method". Finally, we conclude the paper by stating the advantages and disadvantages of the Jacobi method and the modified Jacobi method.

3.3.1. Advantages of the Jacobi Method

- Jacobi method is the simplest method for solving a system of linear equations
- Jacobi method requires non-zero diagonal entries.
- Jacobi method is known as the method of simultaneous displacement and it is very easy to implement.

3.3.2. Disadvantages of the Jacobi Method

- It requires more time to converge.
- It diverges when the matrix is not strictly diagonally dominant.

3.3.3. Advantages of the Modified Jacobi Method

- It can be applied to any matrix with non-zero elements on the diagonals.
- Convergence is only guaranteed if the matrix is either diagonally dominant, or symmetric and positive definite.
- The modified Jacobi method converges faster than the Jacobi method.

3.3.4. Disadvantages of the Modified Jacobi Method

- Error is ineffable
- For tridiagonal matrices, the modified Jacobi method converges or diverges simultaneously.
- Diagonal dominance is a restrictive sufficient condition for the convergence of the modified Jacobi method.

Appendix

The Jacobi Method for 4x4 Matrix Using Matlab

```
clc;
close all;
N0=1000;
count=1;
% JACOBI ITERATIVE METHOD FOR 3 BY 3 MATRICES
X1{1} = input('ENTER THE INITIAL VALUES OF X1 = ');
X2{1} = input('ENTER THE INITIAL VALUES OF X2 = ');
X3{1}= input('ENTER THE INITIAL VALUES OF X3 = ');
X4{1}=input('ENTER THE INITIAL VALUES OF X4 = ');
% ENTER VALUES FOR THE FIRST EQUATION
% X11+X12+X13+X14=A1
disp('Enter in this form a11X1+a12X2+a13X3=A1');
a11=input('Enter a11=');
a12=input('enter a12=');
a13=input('enter a13=');
a14=input('enter a14=');
A1=input('enter A1=');
% ENTER VALUES FOR THE SECOND EQUATION
% X21+X22+X23+X24=A2
disp('Enter in this form a21X1+a22X2+a23X3=A2');
a21=input('Enter a21=');
a22=input('enter a22=');
a23=input('enter a23=');
a24=input('enter a24=');
A2=input('enter A2=');
% ENTER VALUES FOR THE THIRD EQUATION
% X31+X32+X33+X34=A3
disp('Enter in this form a31X1+a32X2+a33X3=A1');
a31=input('enter a31=');
a32=input('enter a32=');
a33=input('enter a33=');
a34=input('enter a34=');
A3=input('enter A3=');
% ENTER VALUES FOR THE FOURTH EQUATION
% X31+X32+X33+X34=A3
```

```
36- %disp('Enter in this form a31X1+a32X2+a33X3=A1');
37- a41=input('enter a41=');
38- a42=input('enter a42=');
39- a43=input('enter a43=');
40- a44=input('enter a44=');
41- A4=input('enter A4=');
42- %X1{0}=X1;
43- %X2{0}=X2;
44- %X3{0}=X3;
45- %X4{0}=X4;
46- for  N= 1:N0
47- % R CALCULATION
48- R1{N}=A1+(a11*X1{N})+(a12*X2{N})+(a13*X3{N});%+(a14*X4{N});
49- R2{N}=A2+(a21*X1{N})+(a22*X2{N})+(a23*X3{N});%+(a24*X4{N});
50- R3{N}=A3+(a31*X1{N})+(a32*X2{N})+(a33*X3{N});%+(a34*X4{N});
51- R4{N}=A4+(a41*X1{N})+(a42*X2{N})+(a43*X3{N});%+(a44*X4{N});
52- % calculate values for x
53- X1{N+1}=X1{N}+(R1{N}/abs(a11));
54- X2{N+1}=X2{N}+(R2{N}/abs(a11));
55- X3{N+1}=X3{N}+(R3{N}/abs(a11));
56- X4{N+1}=X4{N}+(R4{N}/abs(a11));
57- RESULT={N, X1{N+1}, X2{N+1}, X3{N+1}}, X4{N+1}};
58- disp(RESULT);
59- if((X1{N}==X1{N+1}) && (X2{N}==X2{N+1}) && (X3{N}==X3{N+1})&&(X4{N}==X4{N+1}));
60- goto 74;
61- else count=count+1;
62- end
63- disp(count);
64- %for N=1:count
65- % RESULT={N, X1{N+1}, X2{N+1}, X3{N+1}, X4{N+1}};
66- %  disp(RESULT);
67- %end
```

The Modified Jacobi Method for 4x4 Matrix Using Matlab

```
1.  clc;
2.  close all;
3.  N0=1000;
4.  count=1;
5.  % enter initial values of X
6.  X1{1} = input('ENTER THE INITIAL VALUES OF X1 = ');
7.  X2{1} = input('ENTER THE INITIAL VALUES OF X2 = ');
8.  X3{1}= input('ENTER THE INITIAL VALUES OF X3 = ');
9.  X4{1}=input('ENTER THE INITIAL VALUES OF X4 = ');
10. % ENTER VALUES FOR THE FIRST EQUATION
11. % X11+X12+X13+X14=A1
12. disp('Enter in this form a11X1+a12X2+a13X3+a14x4=A1');
13. a11=input('Enter a11=');
14. a12=input('enter a12=');
15. a13=input('enter a13=');
16. a14=input('enter a14=');
17. A1=input('enter A1=');
18. % ENTER VALUES FOR THE SECOND EQUATION
19. % X21+X22+X23+X24=A2
20. disp('Enter in this form a21X1+a22X2+a23X3+a24x4=A2');
21. a21=input('Enter a21=');
22. a22=input('enter a22=');
23. a23=input('enter a23=');
```

```
a24=input('enter a24=');
A2=input('enter A2=');
% ENTER VALUES FOR THE THIRD EQUATION
% X31+X32+X33+X34=A3
disp('Enter in this form a31X1+a32X2+a33X3+a34X4=A3');
a31=input('enter a31=');
a32=input('enter a32=');
a33=input('enter a33=');
a34=input('enter a34=');
A3=input('enter A3=');
% ENTER VALUES FOR THE FOURTH EQUATION
% X31+X32+X33+X34=A4
disp('Enter in this form a41X1+a42X2+a43X3+a44X4=A4');
a41=input('enter a41=');
a42=input('enter a42=');
a43=input('enter a43=');
a44=input('enter a44=');
A4=input('enter A4=');
for  N= 1:N0
% R1 CALCULATION
R1{N}=A1+(a11*X1{N})+(a12*X2{N})+(a13*X3{N})+(a14*X4{N});
% calculate values for x1
X1{N+1}=X1{N}+(R1{N}/-a11);
% R2 CALCULATION
R2{N}=A2+(a21*X1{N+1})+(a22*X2{N})+(a23*X3{N})+(a24*X4{N});
% calculate values for x2
X2{N+1}=X2{N}+(R2{N}/-a11);
% R3 CALCULATION
R3{N}=A3+(a31*X1{N+1})+(a32*X2{N+1})+(a33*X3{N})+(a34*X4{N});
% calculate values for x3
X3{N+1}=X3{N}+(R3{N}/-a11);
% R4 CALCULATION
R4{N}=A4+(a41*X1{N+1})+(a42*X2{N+1})+(a43*X3{N+1})+(a44*X4{N});
% calculate values for x4
X4{N+1}=X4{N}+(R4{N}/-a11);
RESULT={{N, X1{N+1}, X2{N+1}, X3{N+1}}, X4{N+1};
disp(RESULT);
% if((X1{N}==X1{N+1}) && (X2{N}==X2{N+1}) && (X3{N}==X3{N+1}) && (X4{N}==X4{N+1});
% goto 85;
% else  count=count+1;
% end
end
% disp(count);
```

References

[1] Atkinson, Kendall A. (1989), An Introduction to numerical analysis (2nd ed.), New York: John Wiley & Sons..

[2] Bolch, G. et al (2006), Queuing networks and markov chains: Modeling and performance evaluation with computer science applications (2nd ed.), Wiley-Interscience.

[3] Farebrother, R.W. (1988), Linear least squares computations, statistics: Textbooks and monographs, Marcel Dekker.

[4] Grcar, J. F. (2011b),"Mathematicians of Gaussian elimination", Notices of the American Mathematical Society.

[5] Ibrahim B.K. (2010), A Survey of Three iterative methods for the solution of linear equations, IJNM. Vol. 5 Number 1, page 153-162

[6] Lipson, M. and Lipschutz, S. (2001), Schaum's outline of theory and problems of linear algebra, Schaum's outline series, Fourth edition.