COMPUTATIONAL METHOD FOR SOLVING A SYSTEM OF LINEAR ALGEBRAIC EQUATIONS

BY

ADAGHE OSAZUWA JOSEPH

(PGD / MCS / 203 / 96)

A PROJECT SUBMITTED TO THE DEPARTMENT OF MATHS / COMPUTER SCIENCE, FEDERAL UNIVER SITY OF TECHNOLOGY MINNA, IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE AWARD OF POST-GRADUATE DIPLOMA IN COMPUTER SCIENCE.

MARCH 1998

CERTIFICATION

We hereby certify that I have supervised, read and approved this project which I found in scope and quality for the partial fulfilment of the requirement for the award of Post-graduate Diploma in computer science of the Federal University of Technology, Minna, Niger state.

………………………………… ……… ……………………
PRINCE .R. BADMOS DATE
(Project supervisor)

………………………………………………… ……………………
DR K.R ADEBOYE DATE
(H.O.D Maths/Computer science)

………………………………… ……………………
EXTERNAL EXAMINNER DATE

DEDICATION

I shall forever be grateful to almighty God and my loving parent late Mr John Idahosa Adaghe and Theresa Adaghe also to my kind and lovng brother, Mr Lucky Adaghe for their sincere parental care, moral, financial supports and great advice throughout the course. To them all, I dedicate this work.

ACKNOWLEDGEMENT

My profound gratitude goes to almighty God for providing me the wisdom that guided me through this Project work and for sparing my life to complete yet another stage in my academic pursuit.

I owe a great debt of gratitude to my project supervisor, Prince R. Badmos for his invaluable guidance, comment and suggestions which enable me accomplished my desire goals. Also my appreciation goes to my able lecturers like Dr K.R. Adeboye (HOD Math/Computer), Dr S.A Reju, Dr Y.M Ayesimi, Mr Kola R, Mr L.N Azeako, Mr I K Adewale and Mr Dogara all of the Maths/Computer department.

I acknowledge and appreciate the efforts of Mrs Theresa Adaghe, Mr Lucky Adaghe and my guidance-Major Joe Komolafe for their endless support towards the successful completion of this program. I am also indebted to express special gratitude to my colleagues in person of Mr Sani S.I Atsu and my good friend Turayo Falade and Adegbola Steve for their advice and suggestions during the completion of this project.

ABSTRACT

Algebraic equation is an equation in which factors on both sides of an equality sign(=) are the same , but if the highest power of the variable that occurs in the equation is one (1), that equation is regarded as a system of linear algebraic equation and if otherwise, it is non-linear equation.

This project focused on the computational method for solving a system of linear algebraic equation by the use of computer application, due to complexity of the topic itself and the repetitive nature involve in the solving of linear algebraic equation using iterative method (i.e Gauss-Seidel and Jacobs methods), the adoption of the computer application inton the computation of linear algebraic eliminate the complexities involved in the computation of linear algebraic equation manually.

Besides the Direct methods and Indirect methods under which the Gauss and Gauss-Jordan elimination also Jacobs and Gauss -Seidel iterative methods considered some system of linear algebraic equation with the use of computer application written in dbase Language on the different system discussed with the output attached. In addition, the project looked also into linear algebraic equation with matrices and the various types of matrix and their meaning with examples.

In conclusion, the use of computer application in computations of linear algebraic equation fasten the process in solving such equation and getting accurate result in shortest possible time

ABSTRACT

Algebraic equation is an equation in which factors on both sides of an equality sign(=) are the same , but if the highest power of the variable that occurs in the equation is one (1), that equation is regarded as a system of linear algebraic equation and if otherwise, it is non-linear equation.

This project focused on the computational method for solving a system of linear algebraic equation by the use of computer application, due to complexity of the topic itself and the repetitive nature involve in the solving of linear algebraic equation using iterative method (i.e Gauss-Seidel and Jacobs methods), the adoption of the computer application inton the computation of linear algebraic eliminate the complexities involved in the computation of linear algebraic equation manually.

Besides the Direct methods and Indirect methods under which the Gauss and Gauss-Jordan elimination also Jacobs and Gauss -Seidel iterative methods considered some system of linear algebraic equation with the use of computer application written in dbase Language on the different system discussed with the output attached. In addition, the project looked also into linear algebraic equation with matrices and the various types of matrix and their meaning with examples.

In conclusion, the use of computer application in computations of linear algebraic equation fasten the process in solving such equation and getting accurate result in shortest possible time

TABLE OF CONTENT

APPENDICES

(i) Programs and Documentation

DIRECT METHOD

(ii) Gauss elimination method

(iii) Gauss - Jordan elimination method

INDIRECT METHOD

(iv) Gauss seidel method

(v) Jacobi iterative process.

REFERENCES

CHAPTER ONE

1.0 INTRODUCTION

In the simplest term an algebraic equation is a statement stating that whatever is to the left of the equality symbol (=) names the same thing as whatever is to the right of the symbol. There is nothing in this statement that requires it to be true. A mathematical equation may be always true, always false, or true sometimes and false sometimes. Any equation contains at least one variables.

To solve an equation implies to find it's solution set, i.e the set of all valves of the variables (s) employed for which the equation is a true statement. The elements in the solution set are called the Roots of the equation and these are said to satisfy the equation.

1.1 SYSTEM OF LINEAR ALGEBRAIC EQUATIONS.

An algebraic equation is linear if the highest power of the variables(s) that occur is one otherwise it is non-linear. A system of n linear equation is unknown has the general form

$$
\begin{array}{l}
a_{11}X_1 + a_{12}X_2 + a_{13}X_3 + \text{---------------} + a_{1n}x_n = C_1 \\
a_{21}X_1 + a_{22}X_2 + a_{23}X_3 + \text{-----------------} + a_{2n}X_n = C_2 \\
a_{31}X_1 + a_{32}X_2 + a_{33}X_3 + \ldots\ldots\ldots\ldots + a_{3n}X_n = C_3 \\
\bullet \quad \bullet \quad \bullet \qquad \bullet \\
\bullet \quad \bullet \quad \bullet \qquad \bullet \\
\bullet \quad \bullet \quad \bullet \qquad \bullet \\
a_{n1}X_1 + a_nX_2 + a_n3X_3 + \ldots\ldots\ldots\ldots\ldots + a_{3n}X_n = C_n
\end{array}
$$

Where the a's i.e

$$
\left.\begin{array}{l}
a_{11}, a_{12}, a_{13}, \ldots\ldots\ldots\ldots a_{1n} \\
a_{21}, a_{22}, a_{23}, \ldots\ldots\ldots\ldots a_{2n} \\
a_{31}, a_{32}, a_{33}, \ldots\ldots\ldots\ldots a_{3n} \\
a_{n1}, a_{n2}, a_{n3} \ldots\ldots\ldots\ldots a_{nn}
\end{array}\right] \quad a_{ij} \quad \begin{array}{l} i = 1(1)\,n \\ j = 1(1)\,n \end{array}
$$

are constant coefficients and the C's i.e C_1, C_2, C_3,.....C_n are given real constant in a systems of n linear algebraic equation in n unknowns.

In finding the solution of a system of Linear algebraic equation, one need to write out the full equation at each step taken or to carry the variables $X_1, X_2, X_3 \ldots\ldots\ldots\ldots\ldots X_{n-1}$ and X_n through calculations since they always remain in the same column. The only variation from system to system occurs in the coefficients of the unknowns and in the values on the right side of the equations. Due to this, a linear system is often replaced by a matrix which contains all the information about the system that is necessary to determine its solution sett. But in a computer form one can represent the above system of equations by

$AX = C$

where A is called the coefficient MATRIX

$$A = \begin{bmatrix} a_{11}\ a_{12}\ a_{13} \ldots\ldots\ldots\ldots\ldots\ldots a_{1n} \\ a_{21}\ a_{22}\ a_{23} \ldots\ldots\ldots\ldots\ldots\ldots a_{2n} \\ a_{31}\ a_{32}\ a_{33} \ldots\ldots\ldots\ldots\ldots\ldots a_{3n} \\ \cdot \qquad\qquad\qquad\qquad\qquad \cdot \\ \cdot \qquad\qquad\qquad\qquad\qquad \cdot \\ a_{n1}\ a_{n2}\ a_{n3} \ldots\ldots\ldots\ldots\ldots\ldots a_{nn} \end{bmatrix}$$

X is the vector or matrix of unknown variables.

$$X = \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ \\ \cdot \\ X_n \end{bmatrix}$$

and C is the vector of constants

$$C = \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ \cdot \\ \cdot \\ C_n \end{bmatrix}$$

When the vector C is the zero vector, the set of equation is called homogeneous otherwise it is non-homogeneous. Greatest emphasis will be placed on finding numerical solution of sets of a simultaneous linear equations with n unknowns, this is the general

form of a linear system. As a rule, the if $m > n$, the equations cannot be satisfied. If $m < n$, the system usually has an . Infinite number of solutions. For $m = n$
and the systems usually has well defined solution set. Further, as much as possible we shall give considerations to simple linear system of n equations with n unknowns if only for illustration and clarity purposes. Where n is $3 <= n < 5$

1.2 SOLUTIONS FOR SYSTEM LINEAR ALGEBRAIC EQUATION

Suppose matrix A is non-singular then n^{-1} exists and we can multiply both sides of linear equations (1.2) by A^{-1} so that $A^{-1}\ AX = Ix = X$
and so $X = A^{-1}C$
which gives formally the solution of the equations. However, obtaining A^{-1} manually gives much trouble in terms of the significant and often unnecessary computation involved. Finding A^{-1} on a computer is rarely attempted because it is not only a space consuming process but also a time , hence money consuming process. Due to these reasons numerical approaches are adopted for finding the solution of singular equations. Before proceeding to give the analysis of the various numerical method to be considered in this project it is essential for us to stress the need for employment of computer. Manually, the solutions of a given linear system can be obtained by using any of the existing methods for solving linear systems. For simple linear system (e.g three-equations in three unknowns) obtaining solution manually does not give much trouble. However, the solution of a linear system (equation) of quite order. (e.g fifty linear equations in fifty unknowns) is tedious unless arithmetic are mistakes are no occurring often since a considerable amount of arithmetic is involved.

On the alternative, a digital computer may be relied upon to solve a very large system of equation without making any mistakes. The flexibility, precise details of arithmetical facilities as well as fixed point operation are particularly advantageous and that is why attention is focused on computer.

1.3 OBJECTIVE OF THE STUDY

The main purpose of this project is to consider various computational method for solving a system of linear algebraic equations with the use od computer which give fast and more accurate result. Therefore , eliminating the complementing (i.e repetitive nature) involved and solving linear algebraic equations manually.

Also, more details meaning with example about most linear algebraic equation with matrices and types of matrix are emphasis in this project by the use of Computer application written in Qbasic which now pointed out the advantages and disadvantages of the Computer and manually method of computational method for solving a system of linear algebraic equation.

CHAPTER TWO

2.0 LITERATURE REVIEW

The references that have most influenced the presentation of Gaussian elimination and other topics in this project are the texts of forsythe and Moler (1967), Golub and Van Loan (1983) , Isaacson and Keller (1966), Wilkmson (1963), (1965), along with the paper of Kahan (1966). Other very good Methods are given in Conte and Deboor (1980), Noble (1969) and Sewart(1973), more elementary introductions are given in Anton (1984) and Strang (1980).

The best codes for the direct solution of both general and special forms of linear systems of small to moderate size, are based on those given in the package LINPACK, described in Dongaract (1979). These are completely portable programs, and they are available in single and double precision, in both real and complex arithmetic. Along with the solution of the systems, they also can estimate the condition number of the matrix under consideration. The linear equation programs in IMSL and NAG are variants and improvements the programs in LINPACK.

There is a very large literature on solving the linear equation arising from the numerical solutions of partial differential equations (PDES). For some general texts on the numerical solutions of PDES see Birthoff and Lynch (1984), Forsyth and Wasow (1960), Lapidus and Rinder (1982), for texts denoted to classical iterative method for solving the linear equation arising from the numerical solutions of PDES, see Hageman and Young (1981) and Varga (1962).

Integral equation head to dense linear system (equation) and other types of iterative methods have been used for their solutions for some finite successful methods.

One of the most important forces that will be determining the direction of future research in numerical linear algebra is the growing use of Vector and parallel processor computers. The vector machines such as the CRAY-2, work best when doing basic operations on Vector quantities, such as those specified in the BLAS used in LINPACK.

2.1 MATRICES AND LINEAR ALGEBRAIC EQUATIONS

Many of the problems of numerical analysis can be reduced to the problems of solving linear equations. Among the problems which can be so treated are the solution of ordinary or partial differential equation by finite difference methods, the solution of linear algebraic equations, the eigenvalues problems of mathematical physics, ploymominal approximation.

The use of matrix notation is not only convenient but extremely powerful, in bringing out fundamental relationships, the abstract mapping transformations and function between vectors. Matrix notation and algebra are useful because they provide a concuse way to represent and manipulate linear algebraic equations.

2.2 MATRIX

A matrix is a rectangular array of numbers in which not only the number is important but also its position in the array. The size of the matrix is described by the number of its rows and columns. Capital letters are used to refer to matrices e.g (2.1). As doputed in (2.1) [A] is the shorthand notation for the matrix and a_{ij} designates an individual element of the matrix

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ . & . & & . \\ . & . & & . \\ a_{m1} & a_{m2} & & a_{mn} \end{bmatrix}$$

A horizontal set of element is called a row and a vertical set is called a column. The first subscript I is always designates the number of the row in which the element lies. The second subscript j designates the column. For example, in 2.1 has m row and n column and is said to have a dimension of m by n (or mXn) it is referred to as an m-n matrix.

2.3 ROW VECTOR

Matrices with row dimension m = 1, such as $[B] = [\ b_1,\ b_2,\ b_3, \ldots\ldots\ldots b_n\ ]$ are called row vectors.

Note:

That for simplicity the first subscript of each element is dropped. Also, it should be mentioned that there are times when it is desirable to employ a special shorthand notation

to distinguish a row matrix from other types of matrices. One way to accomplish this is to employ special open-topped bracket as in [B]

2.4 COLUMN VECTOR

Matrices with column dimension n = 1, such as

$$A = \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ . \\ . \\ . \\ C_n \end{bmatrix}$$

are referred to as column vectors. For simplicity, the second subscript is dropped. As with the row vector, there are special shorthand notation to distinguish a column matrix from other types of matrices. One way to accomplish this is to employ special brackets as in [B], where this special brackets are called curly brackets. We have the left curly brackets (c) and the right curly bracket ( )).

2.5 SQUARE MATRICES

Matrices where m = n are called square matrices e.g 4 - by - 4 matrix is

$$[A] = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}$$

Note

Square matrices are particularly important when solving sets of simultaneous linear algebraic equation for such systems, the number of equations (corresponding to rows) must be equal in order for unique solution to be possible.

2.6 SPECIAL TYPES OF SQUARE MATRICES

SYMMETRY MATRIX

A square matrix is said to be symmetric if it is symmetric about the leading diagonal, i.e $a_{ij} = a_{ij}$ for all values of i and j. It implies that the i^{th} row, j^{th} column = j^{th} row, i^{th} column in a symmetric matrix the diagonal will be like a mirror. A symmetric matrix must be equal to its own transpose, i.e $A = A^T$, symmetric matrices frequently a rise in the analysis of conservative systems and least squares minimisation and the symmetric property can normally be utilised in numerical operations.

$$[A] \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & -1 \\ 3 & -1 & 5 \end{bmatrix}$$

2.7 SKEW SYMMETRY MATRIX

A skew symmetric matrix is such that $a_{ij} = a_{ji}$, hence $A^T = -A$ and the leading diagonal element aji must be zero.

Any square matrix may be split into the sum of symmetry and askew symmetric matrix thus

$$A = ½(A+A^T) + ½(A - A^T)$$

where $½ (A + A^T)$ is symmetric and $½(A - A^T)$ is skew symmetric.

2.8 RECTANGULAR MATRICES

Otherwise i.e m<>n are called rectangular matrices e.g a 2-by-4 matrix is

$$[B] = \begin{matrix} a_{11} & a_{12} & a_{31} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \end{matrix}$$

It is a 2-by-4 matrix, where m = 2 = number of rows and n =4 number of column.

2.9 THE PRICIPAL OR MAIN DIAGONAL OF THE MATRIX

The diagonal consisting of the elements a_{11}, a_{22}, a_{33} & a_{44} in (2.4) is termed the principal or main diagonal of the matrix

$$\begin{matrix} a_{11} & a_{12} & a_{13} & a_{1n} \\ a_{21} & a_{22} & a_{23} & a_{2n} \\ a_{31} & a_{32} & a_{33} & a_{3n} \\ a_{41} & a_{42} & a_{43} & a_{4n} \end{matrix}$$

2.10 HERMITIAN MATRIX

A square matrix having A = A is called a Hermitan matrix and if it is written as A = C + i D and must be symmetric and D skew symmetric.

2.11 HERMITIAN TRANSPOSE

This is the same as the normal transpose except that the complex conjugate of each element is used. Thus if

$$A = \begin{bmatrix} 5+I & 2-I & 1 \\ 6i & 4 & 9-i \end{bmatrix}$$

$$A^H \begin{bmatrix} 5-i & -6i \\ 2+I & 4 \\ 1 & 9+i \end{bmatrix}$$

2.12 DIAGONAL MATRIX

A square matrix where all the element of the main diagonal are equal to ZERO is called a diagonal matrix, i.e $a_{ij} = 0$ for i =

$$[A] \begin{bmatrix} a_{11} & & & \\ & a_{22} & & \\ & & a_{33} & \\ & & & a_{44} \end{bmatrix}$$

NOTE: That where large blocks of elements are Zero, they are left blanks, The importance of the diagonal matrix is that it can be used for row and Column sealing.

2.13 AN IDENTITY MATRIX

An identity matrix is a diagonal matrix where all the elements on the main diagonal are equal to 1 as in

$$[1] = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{bmatrix}$$

The symbol [1] is to denote the identity matrix the identity matrix has the properties similar to unity

2.14 TRIANGULAR MATRICES

2.14.1 UPPER TRIANGULAR MATRIX

An upper triangular matrix is one where all the element below the main diagonal are ZERO as in

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & a_{22} & a_{23} & a_{24} \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & 0 & a_{44} \end{bmatrix}$$

2.14.11 LOWER TRIANGULAR MATRIX

A lower triangular Matrix is one where all elements above the main diagonal are ZERO, as in

$$\begin{bmatrix} a_{11} & 0 & 0 & 0 \\ a_{21} & a_{22} & 0 & 0 \\ a_{31} & a_{32} & a_{33} & 0 \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}$$

2.15 BANDED MATRIX

A banded matrix has all elements equal to ZERO, with the exception of a band centred on the main diagonal. This matrix has a band width of three and is given a special name- the tri-diagonal matrix. An example below of a tri-diagonal 4 by 4 matrix is shown below.

$$\begin{bmatrix} a_{11} & a_{12} & 0 & 0 \\ a_{21} & a_{22} & a_{23} & 0 \\ 0 & a_{32} & a_{33} & a_{34} \\ 0 & 0 & a_{43} & a_{44} \end{bmatrix}$$

2.16 TRANSPOSE OF A MATRIX

The transpose of a matrix involves transforming its row into columns and its Columns into rows e.g

$$A=(a_{ij})$$
$$A^T = (b_{ij}) \text{ where } b_{ij} = aji$$

A is a symmetric matrix if $A = A^T$

$$[A] = \begin{bmatrix} a_{11} & a_{12} \cdots\cdots\cdots\cdots a_{1n} \\ a_{21} & a_{22} \cdots\cdots\cdots\cdots a_{2n} \\ \cdot & \cdot \\ & \cdot \quad\quad \cdot \\ \cdot & \cdot \\ a_{m1} & a_{m2} \cdots\cdots\cdots\cdots a_{mn} \end{bmatrix}$$

The transpose, designated $[A]^T$ is defined as

$$[A^T] = \begin{bmatrix} a_{11} & a_{21} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots & a_{m1} \\ a_{12} & a_{13} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots & a_{m2} \\ . & . & . \\ & . & . \\ . & . & . \\ a_{1n} & a_{2n} & a_{mn} \end{bmatrix} .$$

In other words, the element aij of the transpose is equal to the aij element of the original matrix. The transpose has a variety of functions in matrix algebra. One simple advantage is that it allows a column vector to be written as a row vector e.g if.

$$[C] = \begin{bmatrix} C1 \\ C2 \\ C3 \\ C4 \end{bmatrix}$$

Then $[C]^T$ = [C1, C2, C3, C4] Where the superscripts T designates the transpose. For example, this can save space when writing a column vector in a manuscript. In addition, the transpose has numerous applications.

2.17 THE TRACE OF A MATRIX

When a matrix is squared, a quality called its trace is defined. The trace of a square matrix is the sum of the elements on its main diagonal it is designated as tr [A] and is computed as

$$\text{tr}\,[A] = \sum_{i=1}^{n} a_{11} + a_{22} + \ldots\ldots\ldots a_{nn}$$

where n = number of rows or columns, since it is a square matrix where number of rows equals number of columns. It should be obvious that the trace remain the same if a square matrix is transposed for

example

$$[A] = \begin{bmatrix} 3 & -1 & 4 \\ 0 & 2 & -3 \\ 1 & 1 & 2 \end{bmatrix}$$

$$\text{tr}[A] = 3+2+2 = 7$$

$$[A]^T = \begin{bmatrix} 3 & 0 & 1 \\ -1 & 2 & 1 \\ 4 & -3 & 2 \end{bmatrix}$$

$$tr[A]^T = 3+2+2 = 7$$

The trace will figure prominently is eigen values problems.

2.18 NULL OR ZERO MATRIX

A null or zero matrix is any matrix with all its elements zero matrix of order 2-by-2

$$[0] = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

FULLY POPULATED AND SPARSE MATRICES

A matrix is fully populated if all of its elements are non-zero and is aparse if only a small proportion of its element are non-zero.

2.20 AUGMENTED MATRIX

A matrix is augmented by the addition of a column (or columns) to the original matrix e.g suppose when a matrix of coefficients.

$$[A] = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

We might wish to augment this matrix [A] with an identify matrix to yield a 3-by-6 dimensional matrix.

$$\begin{bmatrix} a_{11} & a_{12} & a_{13}\vdots & 1 & 0 & 0 \\ a_{21} & a_{22} & a_{23}\vdots & 0 & 1 & 0 \\ a_{31} & a_{32} & a_{33}\vdots & 0 & 0 & 1 \end{bmatrix}$$

such an expression has utility where we must perform a set of identical operations on two matrices. Thus we can perform the operations on the single augmented matrix rather than on two individual matrices.

CHAPTER THREE

DIRECT METHOD FOR SOLVING A SYSTEM OF LINEAR ALGEBRAICEQUATIONS

Method for solving a system linear algebraic equation can be broadly classified into two. These are direct and indirect method. We also have method that can be classified as simi-direct or semi-indirect but we limit ourselves to only direct and indirect methods.

3.1 DIRECT METHODS

Direct methods are methods that give solution to a system linear algebraic equations in a fixed number of steps. Subject only to rounding errors, that is in the absence of round-off errors, these methods will yield exact solution of linear equations after performing a finite number of operations on the system.

Given a system of linear equations

$$R1: a_{11}X_1 + a_{12}X_2 + \dots\dots + a_{1n}X_n = C_1$$

$$R2: a_{21}X_1 + a_{22}X_2 + \dots\dots + a_{2n}X_n = C_2 \qquad (3.1)$$

$$Rn: a_{n1}X_n + a_{n2}X_2 + \dots\dots + a_{nn}X_n = C_n$$

$$AX = C \dots\dots\dots (3.2)$$

as explained in chapter one. We can also represent the system by the corresponding augmented matrix A formed by the coefficients of unknowns and constants where

$$\begin{bmatrix} a_{11} & a_{12}\dots\dots a_{1n} : C_1 \\ a_{21} & a_{22}\dots\dots a_{2n} : C_2 \\ a_{n1} & a_{n2}\dots\dots a_{nn} : C_n \end{bmatrix}$$

$$\begin{bmatrix} a_{11} & a_{22}\dots\dots a_{1n} : a_1, n+1 \\ a_{21} & a_{22}\dots\dots a_{2n} : a_2, n+1 \\ a_{n1} & a_{nn}\dots\dots a_{nn} : an, n+1 \end{bmatrix}$$

This augmented matrix is such that row 1 or R_1, represents the first equation of the system, Row 2, or R_2, the second, and so on, in column1 are the coefficient of X_1 and finally in the last column is the constant term in each equation. This shows that the matrix is an n by (n+1) matrix.

To solve the above system(equation) using direct method, some or all of the following elementary operations can be performed on the equations.

(i) Row 1, R_1(or equation R_1) can be multiplied by a non- zero constant k and the resulting row now used in place of R_1 i.e $KR_1 \longrightarrow R_1$

(ii) R_i can be multiplied by a non-zero constant K, added to row j, Rj, and resulting row used in place of Rj i.e ($R_j + KR_j$) $\longrightarrow$ R_j

(iii) R_i and R_j can be interchanged i.e $R_i \longrightarrow R_j$ by performing a finite number of these elementary operations a linear equation (system) can be transformed into a more easily solved equation with the same set of solution.

This is the principle on which direct methods are based. Some of the known direct methods that will be considered in this project include:

(i) Gauss elimination

(ii) Gauss-Jordan elimination.

3.2 GAUSS ELIMINATION

Gauss elimination method may be regard as a systematic treatment of the basic elimination method in elementary algebra. The main objective is to transform a given system of equation represented by (3.2) into

$$UX = C$$

where UX is an upper triangular matrix and C is a column vector and finally the solution set X are obtained by back substitution.

A systematic method for accomplishing this required transformation is briefly discussed below. Provided a_{11} not equal the operations corresponding to (R_j - ($a_{ji}/a_{ii})R_j$) $\longrightarrow$ R_j, where (a_{ji}/a_{ij}) is called a multiplier, are performed for each j = 2, 3, …………, n to eliminate the coefficient of X_1 in each of these rows. Following a sequential procedure for I = 2, 3,…………, n - 1 and performing the operation (R_j- (a_{ji}/a_{ii}) R_j) R_j for each j = i + 1, i + 2,…………n provided a11 is not equal to all the coefficients of X_1 and will be changed to zero.

The resulting matrix will hence have the form

$$\begin{bmatrix} a_{11} & a_{12}\ldots\ldots\ldots a_{1n}: & a_1, n+1 \\ 0 & a_{22}\ldots\ldots\ldots a_{2n}: & a_2, n+1 \\ 0 & 0\ldots\ldots\ldots a_{nn}: & a_{n1}, n+1 \end{bmatrix} \qquad (3.5)$$

We need to take care here, in each operation some of the elements of the original augmented matrix will be changed for illustration purposes, these new elements or

resulting elements supposed to be differentiated by superscripts which will tell the number of times the elements are modified but for neatness and case of notation we leave the element as they are above.

The new matrix given by (3.5) represents a linear equation (system) with the same solution set as that of equation represented by (3.4). Since the new equivalent linear equation is triangular we can write

$$a_{11}X_1 + a_{12}X_2 + \ldots\ldots + a_{1n}X_n = a_1, n+1$$
$$a_{22}X_2 + \ldots\ldots + a_{2n}X_n = a_2, n+1$$
$$a_{nn}X_n =$$

and back substitution can be performed. By this, the n_{th} equation can be solved for X_n to give

$$X_n = \frac{a_{n1} \ n+1}{a_{nn}}$$

solving the (n - 1) at equation for X_m, and using X_n gives $X_{n-1} = \dfrac{a_{n-1}, n+1 - a_{n-1}, nXn}{a_{n-1}, n-1}$

by successive substitution of known values of X all the unknowns can be found, using the i^{th} row and j^{th} unknown is given by

$$X_i = \left[\frac{a_i, n+1 - a_i, n\,X\,n - a_i, n-1\,Xn-1 \ldots\ldots a_i, i+1\,X\,i+1}{a_{ii}} \right]$$

$$= \left(\left(a_i, n+1 - \sum_{j=i+1}^{n} a_{ij} X_j \right) \right) / a_{ii}$$

for each i = n - 1, n -2..................2, 1

from the foregoing discussion we realise how a given equation may be transformed into an upper triangular matrix and how the complete solution of the equation is obtainable using back substitution.

In the i^{th} divided operation it is always assumed that a_{11} where i = 1, 2,...........n is non-zero. Actually the elements aii are called pivot elements and in our elimination process, to proceed from one stage to another, it is necessary for the pivot elements to be non-zero as they are used as divisor. Modification is necessary at any stage a pivot

elements vanishes. This modification may be in the form of row interchange in order to have non-zero pivot.

Further if a pivot element is small compared with the elements in its column which have to be eliminated a multiplier used at that stage will be greater than one. The use of large multiplier undoutedly, leads to a magnification of round-off error. To avoid this we also need some modification. All the necessary modification analysed above accounted for the two classes of this method.

These are looked at shortly.

3.2.1 GAUSS ELIMINATION WITHOUT PIVOTING

This may be regarded as ordinary Gauss-elimination and all the things said in section 3.2 hold for Gauss elimination without pivoting. The only necessary and sufficient condition is to ensure that none of the pivot element vanishes.

We can have a look at a systematic Gauss elimination without pivoting in the following example.

Example 3.1

Use Gauss elimination without pivoting to solve the following systems (equation).

R_1: $X_1 + X_2 + X_3 = 3$

R_2: $X_1 - X_2 + 2X_3 = 1$

R_3: $-X_1 + X_2 + X_3 = -1$.

The augmented matrix is

$$\begin{bmatrix} 1 & 1 & 1: & 3 \\ 1 & -1 & 2: & 1 \\ -1 & 1 & 1: & -1 \end{bmatrix}$$

basic operations to be operate is given by

$(R_j - (a_{ji}/a_{ii}) R_i) \longrightarrow R_j$

where i = 1, 2, ...n - 1: j =i +1, i + 1,..............n: but n = 3

where I = 1 we perform $(Rj - (a_{ji}/a_{ii}) R_j) \longrightarrow R_j$

j = 2, 3

for j = 2: (a_{21}/a_{ii}) = i/i =1

R2 = 1 - 1 2 1

$(a_{21}/a_{11})R_1 = 1\ \ 1\ \ 1\ \ 3$

$R_2 - (a_{21}/a_{11})R_1 = 0\ \ -2\ \ 1\ \ -2 \longrightarrow R_2$

for $J = 3$, $(a_{31}/a_{11}) = -1/1 = 1$

$R_3 = -1\ \ 1\ \ 1\ \ -1$

$(a_{31}/a_{11})R_1 = -1\ \ -1\ \ -1\ \ -3$

$R_3 - (a_{31}/a_{11})R_1 = 0\ \ 2\ \ 2\ \ 2 \longrightarrow R_3$

These operations reduce the system to

$$\begin{bmatrix} 1 & 1 & 1: & 3 \\ 0 & -2 & 1: & -2 \\ 0 & 2 & 2: & 2 \end{bmatrix}$$

where $i = 2$ we perform $(R_j - a_{j2}/a_{22})R_2 \longrightarrow R_j$

$j = 3$

for $j = 3$, $(a_{32}/a_{22}) = -2/2 = -1$

$R_3 = 0\ \ 2\ \ 2\ \ 2$

$(a_{32}/a_{22})R_2 = 0\ \ 2\ \ -1\ \ 2$

$R_3 - (a_{32}/a_{22})R_2 = 0\ \ 0\ \ 3\ \ 0 \longrightarrow R_3$

thus the new equivalent linear equation (system) is given by

$$\begin{matrix} 1 & 1 & 1: & 3 \\ 0 & -2 & 1: & -2 \\ 0 & 0 & 3: & 0 \end{matrix}$$

i.e

$$X_1 + X_2 + X_3 = 3$$
$$-2X_2 + X_3 = -2$$
$$3X_3 = 0$$

finally, with backward substitution we obtain $X_3 = 0/3 = 0$

$X_2 = \dfrac{-2 - X_3}{-2} = 1$ and $X_1 = \dfrac{3 - X_2 - X_3}{1} = 2$

by direct substitution with the left hand side, LHS of the given equation we obtain

R_1: $X_1 + X_2 + X_3 = 2 + 1 + = 3$

R_2: $X_1 X_2 + 2X_3 = 2 - 1 + 0 = 1$

R_3: $-X_1 + X_2 X_3 = -2 + 1 + 0 = 1$

Compared with the values on the right hand side RHS of the equation we can say that the equation obtained is the exact solution set.

3.2.2 GAUSS ELIMINATION WITH PARTIAL PIVOTING

Gauss elimination with partial pivoting is a modification of Gauss elimination without pivoting. During the derivation of ordinary Gauss elimination, it was found that obtaining a zero for a pivot element necessitated a row interchange. Attention was

drawn to the fact that when large multipliers (a rising as a result of small pivot elements) are employed they could lead to substantial round-off errors. Row interchanges is often desirable too and this is achieved by a process referred to as pivotal condensation or Gauss elimination with partial pivoting.

The rule is quite simple. Before Gauss elimination processes the rows of the augmented matrix are interchanged such that every pivot element is larger in absolute value than (or equation to) any element beneath it in its column. Consequently, the multipliers used at each stage is less then (or equal to one in magnitude).

We summarised the procedure in example 3.2

solve the linear equation

$$R_1: \; 2X_1 + 4X_2 - X_3 = -5$$

$$R_2: \; X_1 + X_2 - 3X_3 = -9$$

$$R_3: \; 4X_1 + X_2 + 2X_3 = 9$$

by Gauss elimination with partial pivoting . The above linear equation can be represented by the matrix

$$\begin{bmatrix} 2 & 4 & -1 & : & -5 \\ 1 & 1 & -3 & : & -9 \\ 4 & 1 & 2 & : & 9 \end{bmatrix}$$

since the pivot elements are not the largest element in their respective column, we need to interchange rows. So the final rearranged augmented matrix assumes form below:

$$\begin{bmatrix} 4 & 1 & 2 & : & 9 \\ 2 & 4 & -1 & : & -5 \\ 1 & 1 & -3 & : & -9 \end{bmatrix}$$

We can now eliminate X_1 from R_2 and R_3 when $i = 1$, perform (R_j - (a_{ji}/a_{ii}) $R_i \longrightarrow R_j$

$j = 2, 3$

for $j = 2$, (a_{21}/a_{11}) = 2/4 =1/2

R_2 = 2 4 4 -1 -5

$(a_{21}/a_{11})\, R_1 = 2$ ½ 1 9/2

$R_2 - (a_{21}/a_{11})R_1 = 0$ 7/2 -2 -19/2 $\longrightarrow R_2$

for $j = 3$, (a_{31}/a_{11}) = ¼

R_3 = 1 1 -3 -9

$(a_{31}/a_{11}) R_1 = 1 \quad ¼ \quad ½ \quad 9/4$

$R_3 - (a_{31}/a_1 = 0 \quad ¾ \quad -3/4 \quad -7/2 \quad -45/4 \longrightarrow R_3$

The equation now takes the following form.

$$\begin{matrix} 4 & 1 & 2 & : & 9 \\ 0 & 7/2 & -2 & : & -19/2 \\ 0 & ¾ & -7/2 & : & -45/4 \end{matrix}$$

To eliminate X_2 from R_3 we perform

$R_j - (a_{j2}/a_{22}) R_2 \longrightarrow R_j$, **where** $j = 3$

$(a_{32}/a_{22}) R_2) = ¾ * 2/7 = 3/14$

$R_3 = 0 \quad ¾ \quad -7/2 \quad -45/4$

$(a_{32}/a_{22}) R_2 = 0 \quad ¾ \quad -3/7 \quad -57/28$

$R_3 - (a_{32}/a_{22}) R_2 = 0 \quad 0 \quad -43/14 \quad -129/14 \longrightarrow R_3$

we now have:-

$$\begin{bmatrix} 4 & 1 & 2 & :9 \\ 0 & 7/2 & -2 & :-19/2 \\ 0 & 0 & -43/14 & :-129/14 \end{bmatrix}$$

and on applying back substitution we have

$$X_1 + X_2 + 2X_3 = 9$$

$$7/2X_2 - 2X_3 = -19/2$$

$$-43/14X_3 = -129/14$$

$$X_1 = (9 - X_2 - 2X_3) * ¼ = 1$$

$$X_2 = (-19/2 + 2X_3) * 2/7 = -1$$

$$X_3 = -129/14 * -14/43 = 3$$

To check our solution set we now substitute for X_1, X_2, X_3 in the original linear equation.

$R_1 :\ 2X_1 + 4X_2 - X_3 = 2(1) + 4(-1) -3 = -5$

$R_2 :\ X_1 + X_2 - 3X_3 = 1 + 1(-3) (3)) = -9$

$R_3 :\ 4X_1 + X_2 + 2X_3 = 4(1) + (-1) + 2(3) = 9$

Since substitution of the solution set into LHS of the equation gives same result as in RHS, we may say that the solution set is exact for the equation.

3.3 GAUSS-JORDAN ELIMINATION

The Gauss-Jordan elimination method is a modification of the Gauss elimination method for solving linear algebraic equation. The purpose of the modification is to eliminate the need for applying back substitution in the gauss-elimination by reducing a linear equation to an equivalent linear equation with zero off diagonal elements. This method can be described as follows.

For row R_i and R_j of linear equation (2.1) perform the operation (R_j - (a_{ji}/a_{ii}) R_i) $\longrightarrow$ R_j

where i, j = 1,2...........n : i not equal j.

In essence Gauss-Jordan elimination uses the i^{th} equation to eliminate not only X_i from the equation R_{i+1}, R_{i+2}......................R_n of a linear equation as was done in the Gauss elimination method, but also form equation R_1, R_2 ...., R_{i-1}.

If we now consider (2.4) which is the matrix form of the equation of n linear algebraic equations in (2.1) where the constants C_i have been denoted by $a_{i,}$ n +1 after the computation routine of Gauss-Jordan elimination method. The final form for the matrix will be

$$\begin{bmatrix} a_{11} & 0 & 0 & a_1, n+1 \\ 0 & a_{22} & 0 & a_2, n+1 \\ 0 & 0 & a_{nn} & a_n, n+1 \end{bmatrix}$$

It must be noted that the entry in each row, say row 1, is expected to change the original value in the augmented matrix (2.4). We retain the entry a11 in the form above just for ease of notation and neatness. Clearly each equation represented by matrix (2.6) takes a reduced form

$$a_{ii} X_i = a_{i,} n + 1 \qquad i = 1, 2, 3 \ldots\ldots\ldots n$$

with solution

$$X_i = \frac{a_i, n+1}{a_{ii}} \qquad i = 1, 2, \ldots\ldots\ldots n$$

we apply this method to solve the linear equation given below

example 2.3

using Gauss-Jordan method, solve the equation.

$$X_1 + 2X_2 + 5X_3 = 20$$

$$2X_1 + X_2 + X_3 = 7$$

$$5X_1 - 3X_2 + 2X_3 = 5$$

The augmented matrix of the above equation is given by

$$\begin{bmatrix} 1 & 2 & 5 & : 20 \\ 2 & 1 & 1 & : 7 \\ 5 & -3 & 2 & : 5 \end{bmatrix}$$

Gauss-Jordan entats the performance of the operation

$(R_j - (a_{ji}/a_{ii}) R_i) \longrightarrow R_j$

$j = 2, 3$

for $j = 2$

$(a_{21}/a_{11}) = 2/1 = 1$

$(a_{21}/a_{11}) R_1 = 2 \quad 4 \quad 10 \quad 40$

$R_2 = 2 \quad 1 \quad 1 \quad 7$

$R_2 - (a_{21}/a_{11}) R_1 = 0 \quad -3 \quad -9 \quad -33 \longrightarrow R_2$

for $j = 3$

$(a_{31}/a_{11}) = 5/1 = 5$

$(a_{31}/a_{11}) R_1 = 5 \quad 10 \quad 25 \quad 100$

$R_3 = 5 \quad -3 \quad 0.2 \quad 0.5$

$R_3 - (a_{31}/a_{11}) R_1 = 0 \quad -13 \quad -23 \quad -95 \longrightarrow R_3$

Thus the equation is first reduced to

$$\begin{bmatrix} 1 & 2 & 5 & 20 \\ 0 & -3 & -9 & -33 \\ 0 & -13 & -23 & -95 \end{bmatrix}$$

when $i = 2$, perform $(R_j - (a_{j2}/a_{22}) R_2 \longrightarrow R_j$

$j = 1, 3$

for $j = 1$

$(a_{12}/a_{22}) = -2/3$

$(a_{12}/a_{22}) R_2 = 0 \quad 2 \quad 6 \quad 22$

$R_1 = 1 \quad 2 \quad 5 \quad 20$

$R_1 - (a_{12}/a_{22}) R_2 = 1 \quad 0 \quad -1 \quad -2 \longrightarrow R_1$

for $j = 3$

$(a_{13}/a_{33}) = -1/16$

$(a_{13}/a_{33}) R_3 = 0 \quad 0 \quad -1 \quad -3$

$R_1 = 1 \quad 0 \quad -1 \quad -2$

$R_i - (a_{13}/a_{33}) R_3 = 1 \quad 0 \quad 0 \quad 1 \longrightarrow R_i$

for $j = 2$

$(a_{23}/a_{33}) = -9/16$

$(a_{23}/a_{33}) R_3 = 0 \quad 0 \quad -9 \quad -27$

$R_2 - (a_{23}/a_{33}) R_3 \; 0 \quad -3 \quad 0 \quad -6 \longrightarrow R_2$

The final reduced equation is given by

$$\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & -3 & 0 & -6 \\ 0 & 0 & 16 & 48 \end{bmatrix}$$

i.e $\quad X_i = 1$

$-3X_2 = -6 \longrightarrow X_2 = 2$

$16X_3 = 48 \longrightarrow X_3 = 3$

We put the solution set $X_1 = 1$, $X_2 = 2$, and $X_3 = 3$ into the given equation to prove the validity of Gauss-Jordan elimination

LHS	**RHS**
$X_1 + 2X_2 + 5X_3 = 1 + 2(2) + 5(3) = 20$	20
$2X_1 + X_2 + X_3 = 2(1) + 2 + 3 = 7$	7
$5X_1 - 3X_2 + 2X_3 = 5(1) + 3(2) + 2(3) = 5$	5

LHS = RHS by direct substitution of the solution set into the equation, Thus Jordan elimination remains valid.

CHAPTER FOUR

4.0 INDIRECT METHOD FOR SOLVING A SYSTEM OF LINEAR ALGEBRAIC EQUATION

Various elimination and factorisation methods for solving linear equation would be discussed . These methods belong to a class of method called Direct methods. The common characteristics is the exact result they give after a finite number of computations and of course, in the absence of round-off errors.

4.1 INDIRECT METHODS.

Indirect methods or iterative methods for solving equation give exact solution to the equation in an infinite number of operations.

This statement reveals the fact that indirect methods do not always give exact solution since we cannot perform an infinite number of operation but get closer and closer to solutions as number of operation increases, provided the methods converge to solutions

Broadly speaking , an indirect method to solve the equation $AX = C$ starts with an initial approximation $X^{(0)}$ to the solution X_1 and generates a sequence of vectors $X^{(k)}$ $k = 0, 1, \ldots\ldots\ldots\ldots$ That converges to X.

Most of the indirect methods involves a process that converts the equation $Ax = C$ into an equivalent equation of the form $X = C + TX$, where C is a vector and T a matrix.

After selecting the initial vectors $X^{(0)}$, the sequence of approximated solution vector is generated by computing

$X^{(k+1)} = C + TX^{(k)}$ $K = 0, 1, 2\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$

This computation can not be carried out indefinitely so we need to apply a suitable termination exterior. Most commonly use stopping criteria include.

$$1. \left| X^{(k+1)} - X^{(k)} \right| \langle \quad \varepsilon$$

$$2. \left| \frac{X^{(k+1)} - X^{(k+1)}}{X^{(k+1)}} \right| \langle \quad \varepsilon$$

where ε is a prescribed tolerance i.e an acceptable error exterior. By formulating the general iterative methods for approximating the solution of linear equation AX = C. The linear system (equation) to be consider is that of (1.1) and would replace this in the form (4.1) below.

$$X_1 = (C_1 - a_{12}X_2 - a_{13}X_3 \ldots\ldots\ldots\ldots\ldots\ldots a_{1n}X_n) / a_{11}$$
$$X_2 = (C_2 - a_{21}X_1 - a_{23}X_3 \ldots\ldots\ldots\ldots\ldots\ldots a_{2n}X_n) / a_{22}$$
$$X_n = (C_2 - a_{21}X_1 - a_{23}X_3 \ldots\ldots\ldots\ldots\ldots\ldots a_{nn}, n - 1X_{n-1}) / a_{nn}$$

equation (3.1) can be written more concisely as

$$X_i = (C_i - \Sigma\, a_{ij}\, X_j\,)_{j=i}\, n/a_{ii} \qquad i = 1, 2, \ldots\ldots\ldots n. \qquad (3.2)$$

which is in the $j \neq i$ form. $X = C + \tau x$

From the above rearrangement is predicted on aii not equal to 0. Usually, rearrange the equations and the unknown so that diagonal dominance is obtained. Then making initial quesses for the X_i and insert these values into the right hand side of (3.1) and generate new and better approximations by successively repeating the process. The following iterative methods will be considered in this section.

(i) Jacobian's iterative methods

(ii) Gauss-seidel iterative method.

4.2 JACOBI ITERATIVE METHOD

Suppose substituting the initial quesses into (3.2) to generate the new approximations for successive approximation then after the (k + 1)st iteration we will have

$X_i^{(k+1)} = (C_i - \Sigma\, a_i X_j^{(k)} / a_{ii} \quad i = 1, 2 \ldots\ldots\ldots\ldots\ldots\ldots\ldots, n$ (3.3)

The above method is the Jacobi iterative method. Let us see how it works

Example (3.1)

Solve to an accuracy of four places of decimal

$$4X_1 + X_2 + 2X_3 = 4$$

$$3X_1 + 8X_2 - X_3 = 20$$

$$2X_1 - X_2 - 4X_3 = 4$$

Using Jacobi method.

NOTE:- The exact solution set is (1, 2, -1) we rewrite the equations as

$$X_1 = (4 - X_2 - 2X_3) /4$$

$$X_2 = (20 - 3X_1 + X_3) /8$$

$$X_3 = (4 - 2X_1 + X_2) /4$$

for an initial approximation let $X_1^{(0)} = (0, 0, 0)$. We generate $X_1^{(1)}$ by:

$X_1^{(1)} = (4 - X_2^{(0)} - 2X_3^{(0)} /4 = (4 - 0 - 0) /4 = 1.0000$

$X_2^{(1)} = (20 - 3X_1^{(0)} + X_3^{(0)} /8 = (20 - 0 + 0) /8 = 2.5000$

$X_3^{(1)} = (-4 + 2X_1^{(0)} - X_2^{(0)} /4 = (-4 + 0 - 0) /4 = 1.0000$

Additional iterative $X_i^{(k)}$, i = 1, 2, 3 are generate in a similar manner and presented in table 1.

TABLE 4.1

K	$X_1^{(k)}$	$X_2^{(k)}$	$X_3^{(k)}$
0	0.0000	0.0000	0.0000
1	1.0000	2.5000	-1.0000
2	0.8000	2.0000	-1.1250
3	1.0625	2.0313	-10625
4	1.0234	1.9687	-0.9766
5	0.9961	1.9941	-0.9805
6	0.9917	2.0039	-1.0005
7	0.9993	2.0030	-1.0051
8	0.0018	1.9996	-0.0110
9	1.0006	1.9992	-1.9999
10	0.9997	1.9998	-1.9995
11	0.9998	2.0002	-1.0001
12	1.0000	2.0000	-1.0002
13	1.0000	1.9990	-1.0000
14	1.0000	1.9999	-0.9999
15	0.9999	2.0000	-0.9999
16	1.0000	2.0000	-1.0000

Hence to 4D the solutions are $X_1 = 1.0000$, $X_2 = 2.0000$, $X_3 = -1.0000$.

It is iterative that the approximations computed at the fifth iteration are roughly within 0.4%, 0.3%, 2.0% i.e the approximations are on the average within 0.3% of the exact solution. The accuracy was improved by performing more iterations. For example at the tenth iteration the approximations are roughly within 0.03% of the exact solution set. Finally, at the fifteenth iterations the approximations are within 0.0 % of the exact solution it is also observed that a whole new solution set is computed before it is used in the next iteration.

4.3 GAUSS-SEIDEL ITERATIVE METHODS

Just as Gauss elimination is the most heavily used method, of the direct methods, Gauss seidel method is the most heavily used, of the iterative method. The major difference between Jacobi and Gauss seidel iterations the newly generated components of the solution set are always used as soon as they are available, whereas in Jacobi iterations the new components are not used until all component of the solution set have been found. Considering equation (1.1) again, the application of Gauss-seidel method starting with an initial quess for the unknowns equation (3.2) which has been proved to be a rephased form of equation (1.1) will take the form

$X_i^{(k+1)} = (C_i - \Sigma\, a_{ij} X_{j\; j=1}\; i - 1 - \Sigma\, a_{ij}X_j^{(k)}) / a_{ii}\; j = i + n^{n}$

I = 1, 2,.........................n (3.4)

after $(k + 1)^{st}$ iteration let us see an application of this method.

Example 3.2

$$4X_1 + 3X_2 = 24$$
$$3X_1 + 4X_2 - X_3 = 30$$
$$X_2 + 4X_3 = -24$$

Which has the solution (3, 4, -5) for an accuracy of four decimal places using. Gauss-seidel method on rewriting the above equations we have for Gauss-seidel method.

$$X_1^{(k+1)} = (24 - 3X_2^{(k)}) /4$$
$$X_2^{(k+1)} = (30 - 3X_2^{(k+1)} + X_3^{(k)} /4$$
$$X_3^{(k+1)} = (-24 - X_2^{(k+1)}) /4$$

we choose $X_i^{(0)} = (0, 0, 0,)$ i = 1, 2, 3. The first iteration gives

$X_1^{(1)} = (24 - 3X_2^{(0)} /4 = (24 - 0) /4 = 6.0000$

$X_2^{(1)} = (30 - 3X_1^{(1)} + X_3^{(0)})/4 = (30 -3(6.0000) + 0) /4 = 3.0000$

$X_3^{(1)} = (-24 + X_2^{(1)}) /4 = (-24 + 3) /4 = -5.2500$

The results of first and other iterative generated in the above manner as tabulated below.

TABLE 4.2

K	$X_1^{(k)}$	$X_2^{(k)}$	$X_3^{(k)}$
0	0. 0000	0. 0000	0. 0000
1	6. 0000	3. 0000	-5. 2500
2	3.7500	3. 7500	-5. 1563
3	3. 4688	3. 6094	-5. 0977
4	3. 2930	3. 7559	-5. 0610
5	3. 1831	3. 8474	-5. 0382
6	3. 1144	3. 9046	-5. 0238
7	3. 0715	3. 9404	-5. 0149
8	3. 0447	3. 9627	-5. 0093
9	3. 0279	3. 9767	-5. 0058
10	3. 0175	3. 9854	-5. 0036
11	3.0109	3. 9909	-5. 0023
12	3. 0068	3. 9943	-5. 0014
13	3. 0042	3. 9964	-5. 0009
14	3. 0027	3. 9977	-5. 0006
15	3. 0016	3. 9986	-5. 0003
16	3. 0010	3. 9991	-5. 0002
17	3. 0006	3. 9995	-5. 0001
18	3. 0004	3. 9996	-5. 0000
19	3. 0002	3. 9997	-5. 0000
20	3. 0001	3. 9998	-5. 0000
21	3. 0001	3. 9999	-5. 0000
22	3. 0000	3. 9999	-5. 0000

To 4D therefore the required solutions are $X_1 = 3.\ 0000$, $X_2 = 4.\ 0000$, $X_3 = -5.\ 0000$. It is necessary to make some remarks about Jacobi and Gauss-seidel methods. Example 3.1 requires 16 iterations suppose we use Gauss-seidel, we require just 8 iterations.

This gives the feeling that the Gauss-seidel method is superior to the Jacobi method. Well, thi is generally the case but is not always true. There are systems of linear equations for which the Jacobi method converges and the Gauss-seidel method does not and vice-visa.

CHAPTER FIVE

5.0 SUMMARY, CONCLUSION AND RECOMMENDATION

Various methods discussed for solving a system of linear equations have been considered in this project. This points to the fact that no single method is best in all situation. Computational time and accuracy of solutions are measure of efficiency and sufficiency of the methods. Time is of importance in solving large system of linear algebraic equation because of large volume of computation involved. Furthermore, because of the round off error involved in performing large volume of computations, accuracy is of concern. This lead to the development of computer programs for computation of such large and small system of linear algebraic equation.

5.1 CONCLUSION

This project have successfully looked into the various types of matrix and express their meaning with examples, also in this project the various method for solving a system of linear algebraic have been applied on some linear algebraic equation and it revealed that, depending on the nature of the system of linear algebraic equation that would determined whether a direct or indirect or an iterative technique (method) is to be apply to give exact solution to the system linear algebraic equation.

5.2 RECOMMENDATIONS.

As it has been explained, the use of the various known methods for solving system of linear algebraic equations is based on the computational the kind of system of linear algebraic equations one intend to solve.

Gauss-Jordan elimination method which is the variant of Gauss elimination is relatively less efficient computation wise. When the system of linear algebraic equation have identical coefficient matrices but different vector constants(as in repeated measurements on a single sample) direct method are generally most efficient since one does not need to solve complete problem for each new vector. Generally, direct methods are used for solving a system of linear algebraic equation of small dimension.

Indirect method are considered for solving a system of linear algebraic equation because the round off errors produced is comparative less seen. There are extremely efficient for solving system of linear algebraic equation with large and random sparse matrices equations of this type arise naturally, For instance, in the numerical solution of partial differential equations efficiency of both direct and indirect techniques can be improved if the coefficient matrices of the system linear algebraic equation exploitable structure, when coefficient matrix is strictly diagonally dominant Gauss-Seidel is most efficient.

•

However based on various examples computation is recommended that :

a. **Direct method is of great benefit in solving a system of linear algebraic equations due to less computation and time involve.**
b. **Direct method is preferable when a system of linear algebraic equation have similar coefficient matrices but different vector constants**
c. **The same method is recommended for a system of linear algebraic equation with little dimension.**

APPENDICES

1. PROGRAMS AND DOCUMENTATION

This appendix contains the steps in each of the methods so far considered. Efforts are made to combine the descriptions of similar methods in order to avoid unnecessary repetitions.

Flowcharts describing the operation and the order of performance of the steps in machine computation as well as the corresponding programs are also included. Sample inputs to the programs are the various example used for illustration in chapter three and four. Of course, the sample outputs from the programs on comparison with results obtained manually confirm the efficiency or effectiveness of the programs.

REFERENCES

1. Alan Jennings(1980): Matrix Computer for Engineers and Scientist
   Mir Publisher. Moscow

2. Alkis Constanides (1988): Applied Numerical method With Personal Computers
   Academic Press Ltd.
   London

3. Cuirtis Gerald (1988): Applied Numerical Analysis
   Ellis Horwood Ltd(Publisher)
   England.

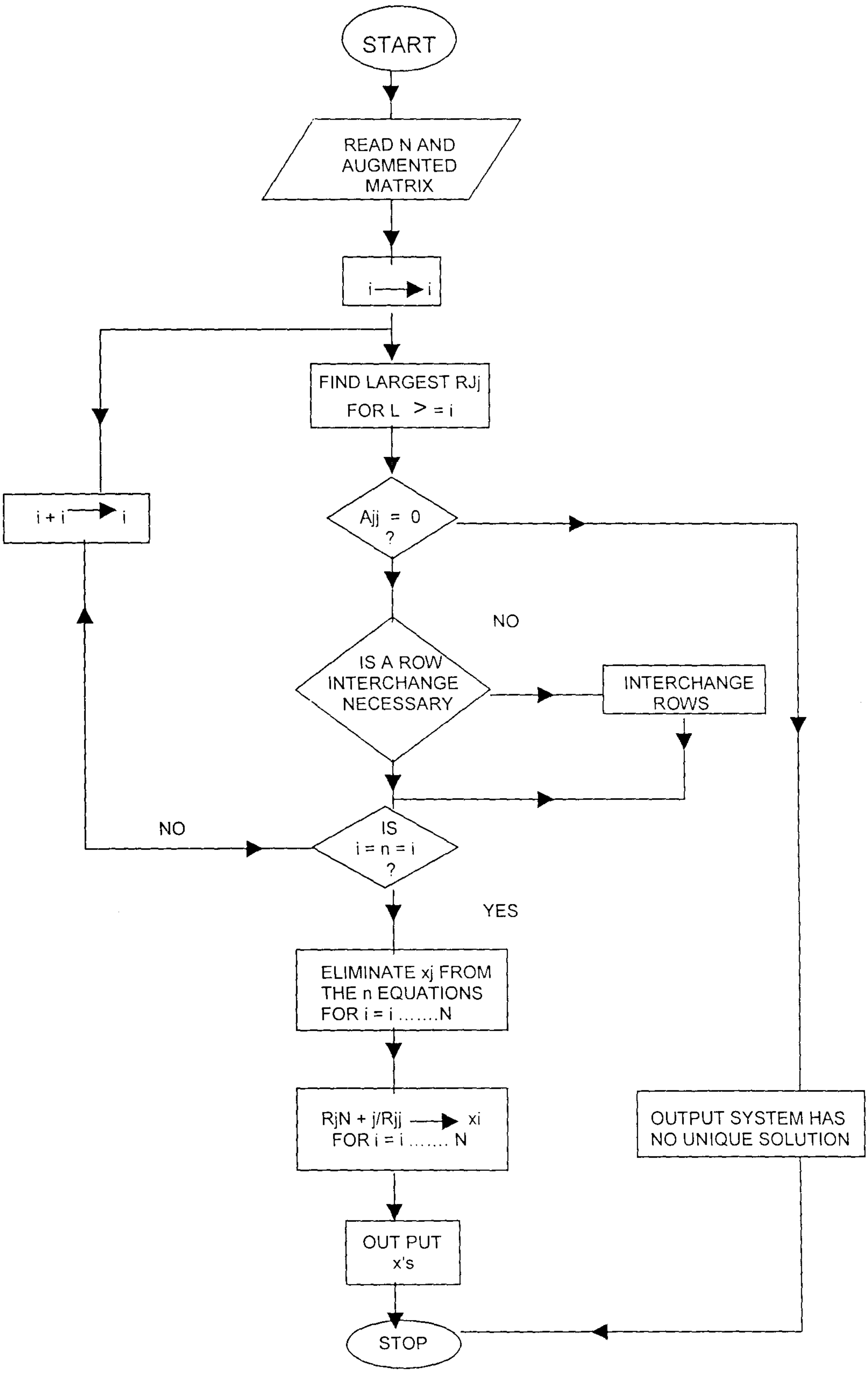
4. Kendalc Atkinson (1976): An Introduction to Numerical Analysis
   Ellis Horwood Ltd.
   England.

5. Peter Turner (1963): Guide to Numerical Analysis
   Macmillan Mathematical Guides
   Publisher-D.Van Nostand Co.
   Canada.

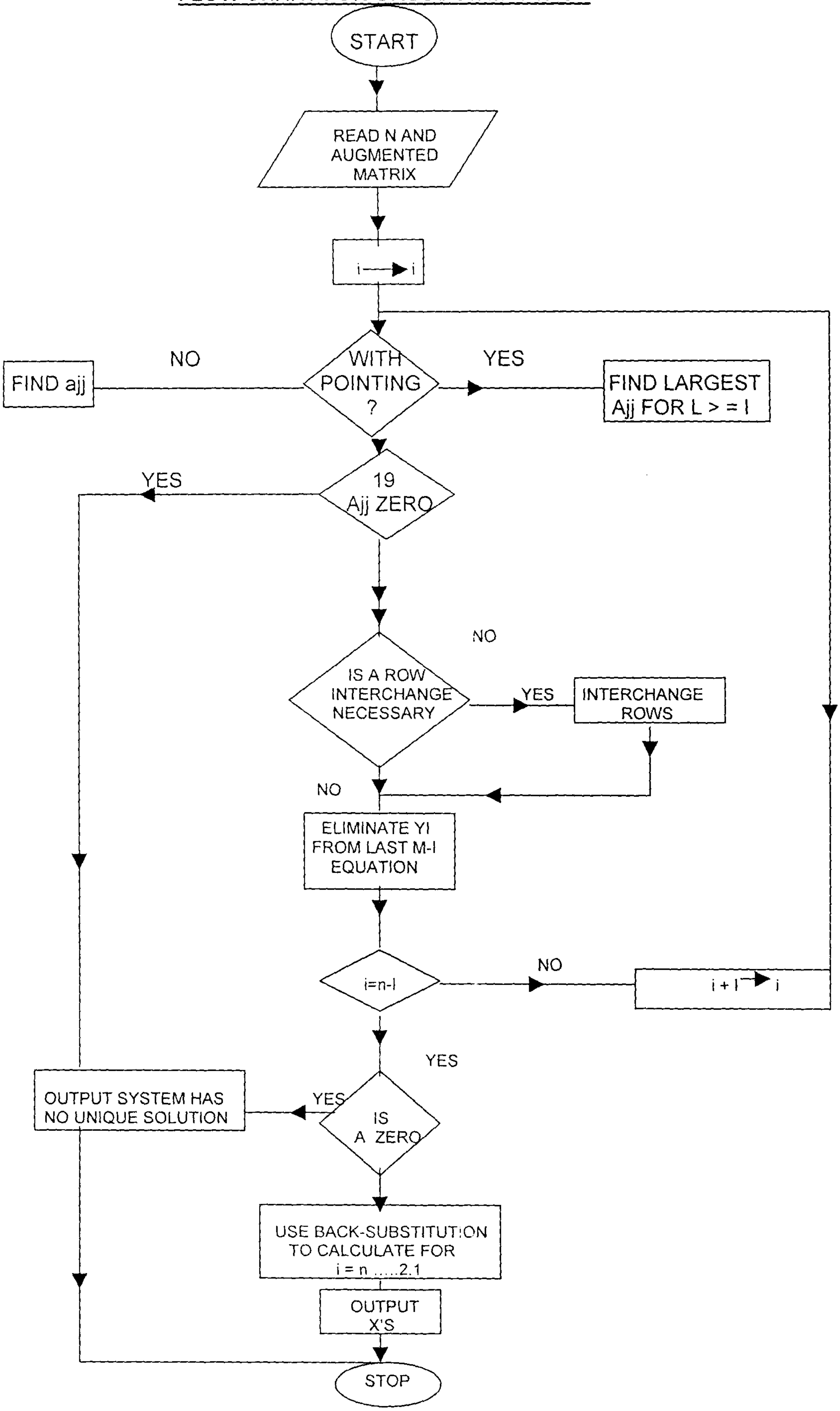
6. Steve Chapra (1997): Numerical Methods for Engineers
   John Wiley & Son,
   United States of America.

APPENDIX A

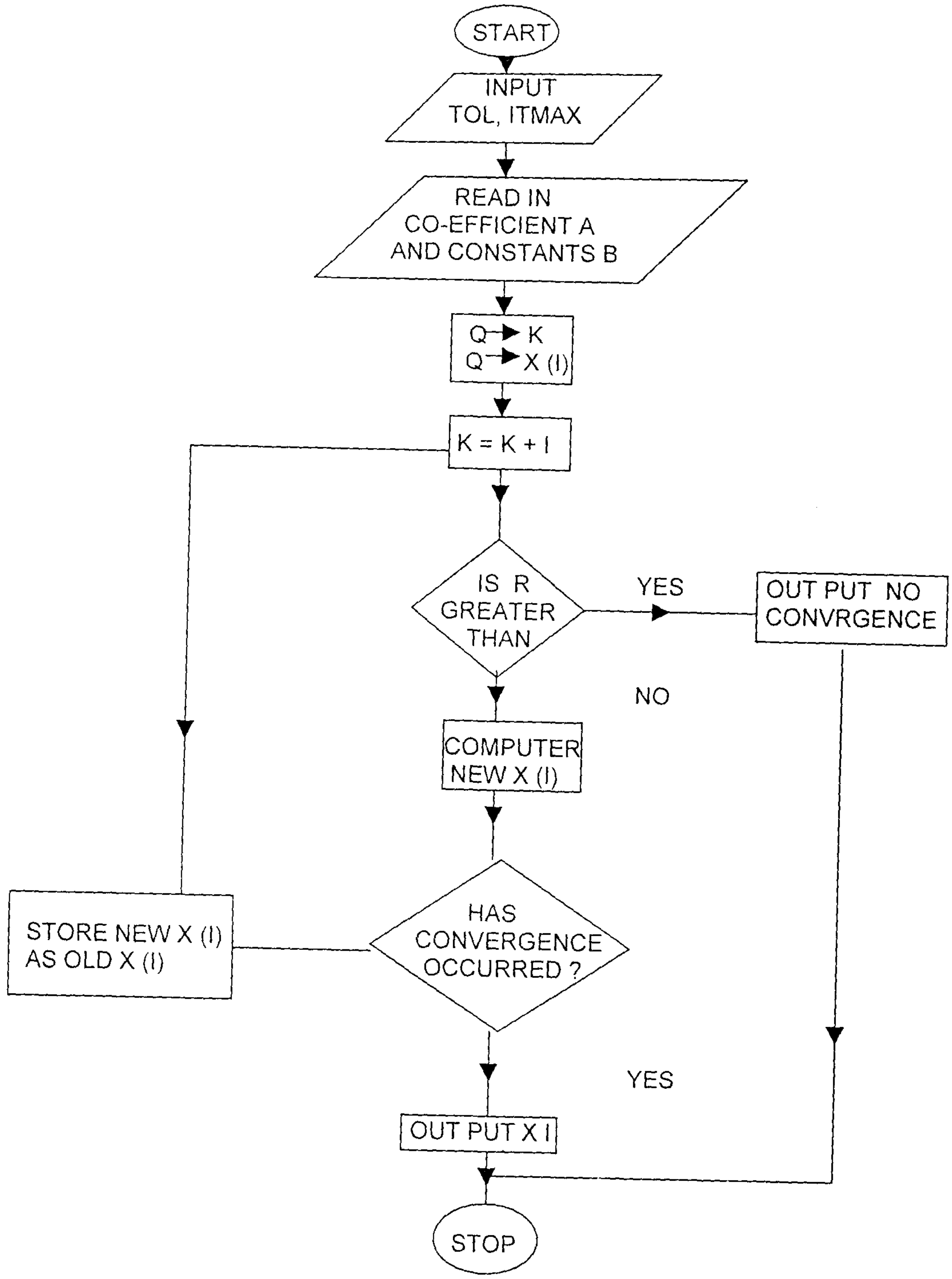
FLOW CHART FOR GAUSS JORDAN ELIMINATION PROCESS



FLOW CHART FOR GROSS ELIMINATION



FLOW CHART ITERATIVE TECHNIQUES, CHOICE OF METHOD DEPENDS ON THE STATEMENT USED IN COMPUTING NEW X (I)



APPENDIX B

```
10  SCREEN 0: WIDTH 80: CLS : KEY OFF
20 PRINT " ****************************"
30 PRINT " * THE GAUSS ELIMINATION       *"
.0 PRINT " * METHOD FOR SIMULTANEOUS     *"
50 PRINT " * LINEAR ALGEBRAIC EQUATION *"
60 PRINT " *                            *"
70 PRINT " *  (GAUSS BAS.)              *"
80 PRINT " *                            *"
90 PRINT " ****************************"
100 PRINT " ******MAIN PROGRAM**********"
110 PRINT "ENTER THE NUMBER OF EQUATIONS, THE COEFFICIENT AND CONSTANT"
120 PRINT : PRINT " NUMBER OF EQUATIONS", : INPUT N
130 DIM A(N, N + 1), B(N, N + 1), X(N), NP1VROW(N.2), NP1VCOL(N.2)
140 PRINT : PRINT "ENTER COEFFICIENTS AND CONSTANT FOR EACH EQUATIONS"
150 FOR K = 1 TO N
160 PRINT : PRINT " EQUATIONS"; K;
170 FOR J = 1 TO N
180 PRINT "COEFFIENT ("; K, " "; J; ") = ,", B(K.J)
190 NEXT J
200 PRINT : PRINT "CONSTANT", K: : : INPUT B(K, N + 1)
210 NEXT K
220 NC = N + 1
230 PRINT
240 PRINT , " GIVE THE MINIMUM ALLDWABLE VALUE OF THE PIVOT ELEMENT": INPUT EI
250 PRINT CHR$(12)
260 DET = 1
270 FOR K = 1 TO N
 80 FOR J = 1 TO NC
 .0 A(K.J) = B(K.J)
  ) NEXT J.K
    PRINT : PRINT
    PRINT "******************"
    PRINT "AUGMENTED MATRIX"
    GOSUB 130
    PRINT , "IS THE AUGMENTED MATRIX CORRECT (Y/N)"; 0: PRINT
    IF O$ = "Y" OR O$ = "Y" THEN 430
     RINT "GIVE THE POSITION OF THE ELEMENT TO BE CORRECTED"; : PRINT
     NPUT "ROW NUMBER"; NROW: INPUT "COLUMN NUMBER,"; NCOL
      INT : INPUT "CORRECT VALUE OF THE ELEMENT", B(NROW, NCOL)
      TO 250

       ning ofthe Gauss elimination procedure.
       T " DO YOU WANT TO SEE STEP-BY-STEP RESULT Y/N "! Q2$ PRINT.
         "********************"
          = 1 TO N
          COMPLETE PIVOTING STRATEGY
          T = ABS(A(K.K))
           (K, 1) = K: NP1 VROW(K.2) = K
            (K, 1) = K: NP1 VCOL(K.2) = K
             TO N
             TO N
             OT >= ABS(A(I,J)) GO TO 560
              ABS(A.(I, J))
               ) = K: NP1 VROW(K.2) = I
               ) = K: NP1 VCOL(K.2) = J

                ERS GOTO 590
                LEMENT SMALLER THAN : EPS: MATRIX MAY BE SINGULAR, GOTO"
                 .2) = K GOTO 660
                  Q2$ = "Y" THEN PRINT INTERCHANGE; ROWS, NP1; VROW(K.2); ",

                 ( (K.2).J),A(K,J)

                 .)
```

```
650 IF Q2$ = "y" OR Q2$ = "Y" THEN GOSUB 1300
660 IF NP1 VCOL(K.2) = K GOTO 740
670 Q2$ = "Y" OR Q2$ = "Y" THEN PRINT "PIVOTINGE COLUMNs"
680 IF Q2$ = "Y" OR Q2$ = "Y" THEN PRINT "INTERCHANGE COLUMNS"; NP1VCOL(K.2);
690 FOR I = 1 TO N
700 SWAP A(I,NP1VCOL(K.2), A(I,K)
710 NEXT I
720 DET = DET * (-1)
730 IF Q2$ = "Y" OR Q2$ = "Y" THEN GOSUB 1300
740 IF K = N THEN GOTO 850
750 IF Q2$ = "Y" OR Q2$ ="Y" THE PRINT "PERFORM ELIMINATION"
760 FOR I = K + 1 TO N
770 IF Q2$ = "Y" OR Q2$ = "Y" THEN PRINT "DIVIDE ROW": K: BY ":A(K.K)"
780 IF Q2$ = "Y" OR Q2$ = "Y" THEN PRINT "MULTIPLY ROW":K:BY A(I.K):"AND SUBTF
790 MULT = -A(J.K) / A(K.K)
800 FOR J = NC TO K STEP -1
810 A(I.J) = A(I.J) + MULT * A(K.J)
820 NEXT I
850 NEXT K
860
870 APPLY THE BACK-SUBSTITUTION FORMULAS
880 RANK = K - 1: PRINT "RANK: NMR = N-RANK"
890 IF RANK = N THEN X(N) = A(N.N + 1) / A(N.N): NCOUT = N - 1: GOTO 940
900 PRINT "THE PROGRAM SETS "; NMR; "UNKNOWN(S) TO UNITY"
910 PRINT "AND REDUCES THE PROBLEM TO FINDING OTHER":RANK:"UNKNOWNS
920 FOR JJ = 1 TO NMR: X(N + 1 - JJ) = 1: NEXT JJ
930 NCOUNT = RANK
 40 FOR I = NCOUNT TO 1 STEP -1
 50 SUM = 0
  0 FOR J = 1 TO N
    SUM = SUM + A(I.J) * X(J)
    NEXT J
    X(I) = (A(I, N + 1) - SUM) / A(I.I)
     NEXT I

     INTERCHANGE THE ORDER OF THE UNKNOWNS TO CORRECT FOR THE COLUMN PIVOTING
     FOR K = N TO 1 STEP -1
     SWAP X(NP1 VCOL(K.2), X(NP1 VCOL(K.1)
     EXT K

       ALUATE THE DETERMINANT OF THE MATRIX
        I = 1 TO N
        = DET * A(I.I)
        J

        : PRINT "RESULTS BY BACK SUBSTITUTION:"PRINT
         = 1 TO N
         X(",J:") = ";X(J)

          PRINT "VALUE OF DETERMINANT = ": DET: PRINT

           RINT "DO YOU WANT TO REPEAT THE CALCULATIONS": PRINT "WITH MINOR
            FFICIENTS (Y/N)":: INPUT V$
             OR V$ = "Y" THEN 1200 ELSE 1210
             50
             "DO YOU WANT TO RESET ALL THE COEFFIENTS (Y/N)": W$
              OR W$ "Y" THE NEW SET OF THE SAME ORDER AS THE PREVIOUS SET",
               OR INW$ = "n" THEN CHR$(12): RUN 100
```

```
1290 SUBROUTINE 1: PRINT the; MATRIX
1300 FOR KA = 1 TO N
1310 PRINT I TO K
1320 FOR J = 1 TO NC
1330 A(KA, 7)
1340 NEXT J: PRINT : NEXT KA: PRINT
1350 FOR DELAY = 1 TO 270.1, NEXT
1360 RETURN
```

```
10 CLS : KEY OFF
20 PRINT "****************************"
30 PRINT "*  GAUSS-SEIDEL ITERATIVE    *"
40 PRINT "*         METHOD             *"
50 PRINT "*                            *"
60 PRINT "*                            *"
70 PRINT "*    SEIDEL.BAS              *"
80 PRINT "*                            *"
90 PRINT "****************************"
100 FOR DEL = 1 TO 5000: NEXT DEL: CLS
110 INPUT " ITERATION NUMBER"; IN
120 R = 0: X = 0: Y = 0: Z = 0
130 FOR ITER = 1 TO N
140 X = (10 - Y - Z) / 5
150 Y = (7 - X + 2 * Z) / 6
160 Z = (16 - X + 3 * Y) / 7
170 R = R + 1
180 PRINT "X (":R:") =": X
190 PRINT "Y(:R:) = ": Y
200 PRINT "Z(:R:) = ": Z
210 PRINT
220 NEXT ITER
230 PRINT
240 LOA "A: MA.BAS": R
250 END
```

```
10 SCREEN 0 WIDTH 80 CLS KEY OFF
20 PRINT ***********************************
30 PRINT * THE GAUSS-JORDAND                 *
 ) PRINT * REDUCTION METHOD FOR              *
 ' PRINT * SIMULTANEOUS LINAER ALGEBRAIC     *
   PRINT * EQUATIONS AND MATRIX INVERSION    *
   PRINT *         (JORDAN.BAS)              *
   PRINT ***********************************
   PRINT **********MAIN PROGRAM***********

    PRINT : PRINT "YOU MAY USE THE THIS PROGRAM TO :"
    RINT : PRINT 1; " SOLV LINEAR ALGEBRAIC EQUATIONS"
     INT : PRINT 2; FIND; THE; INVERSE; OF; A; MATRIX; ""
     INT : PRINT 3  DO BOTH OF THE ABOVE
      NT : INPUT " THE NUMBER OF YOUR SELECTION", SEL
      ER THE NUMBER OF EQUATIONS THE COEFFICIENT AND CONSTATNTS.

       ' IF SEL: = 2 THEN INPUT "NUMBER OF ROWS OF THE MATRIX"; N
        <> 2 THE INPUT "NUMBER OF EQUATIONS: N
        I, 2 * N + 1), B(N, N + 1), C(N, N), XC(N)
          IF SEL = 2! THEN PRINT "ENTER ELEMENTS OF MATRIX " ELSE
          ENTER COEFFICIENTS AND CONSTANT FOR EACH EQUATIONS:
            TO N
            F SEL = 2 THEN PRINT "ROW"; K ELSE PRINT " EQUATIONS": K

             'O N
              PRINT "ELEMENT (";K";"J,")=":: INPUT B(K,J)
              N PRINT "COEFFICIENT (";K;"J";): INPUT B(K,J)

               HEN PRINT "CONSTANT"; K, " ="; : : INPUT B(K, N + 1)

                E MINIMUM ALLDWABLE VALUE OF THE PIVOT ELEMENT"; : INPUT E

                 + 1
```

```
370 A(K, J) = B(K, J)
380 NEXT J
390 FOR J = N + 2 TO 2 * N + 1
400 A(K, J) = 0
410 NEXT J
420 A(K, K - 1 + N + 2) = 1
430 NEXT K
440 PRINT : PRINT : PRINT
450 PRINT
460 PRINT : AUGMENTED MATRIX
470 GOSUB 560
480 PRINT: INPUT "IS THE AUGMENTED MATRIX CORRECT (Y/N)"; Q$PRINT
490 IF Q$ = "Y" OR Q$ = "Y" THEN
500 PRINT "GIVE THE POSITION OF THE ELEMENT TO BE CORRECTED: PRINT "
510 INPUT "ROW NUMBER"; NROW: INPUT "COLUMN NUMBER"; NCOL: B
520 PRINT: INPUT "CORRECT VALUE OF THE ELEMENT: B(NROW,NCOL): PRINT
530 GOTO 350
540
550 "Begining of the Gauss-Jordan reduction procedure.
560 INPUT "DO YOU WANT TO SEE SEPS-BY-SEPS RESULTS(Y/N)".Q2$:INPUT
570 PRINT
580 FOR K = 1 TO N
590 "APPLY PARTIAL, PIVOTING STRATEGY
600 MAX PIVOT = ABS (A(K,K): NPIVOT = K
610 FOR I = K TO N
620 IF MAXPIVOT >= ABS(A(I,K) GOTO 640
630 MAXOIVOT = ABS(A(I,K): NPIVOT =I
640 NEXT I
650 IF MAXPIVOT >= EPS GOTO 670
560 PRINT " PIVOT ELEMENTS SMALLER THAN, "EPS: MATRIX MAY BE SINGULAR
70 RANK = ;K-1: GOTO 1100
30 IF NPIVOT = K GOTO 740
0 IF Q2$ = "Y" OR Q2$ = "Y" THEN PRINT "PARTIAL PIVOTING"
0 IF Q2$ = "Y" OR Q2$ = "Y" THEN PRINT "INTERCHANGE ROW"; NPIVOT; "AND:K"
) FOR J = K TO 2 * N + 1
  SWAP A(NPIVOT, J), A(K, J)
  NEXT J
  IF Q2$ = "Y" OR Q2$ = "Y" THEN GOSUB 1150
  IF Q2$ = "Y" OR Q2$ = "Y" THEN PRINT "PERFORM NORMALIZATION"
  IF Q2$ = "Y" OR Q2$ = "Y" THEN PRINT "DIVIDE ROW" :K; "B"; A(K,K)
    = A(K, K)
   R J = 2 * N + 1 TO K STEP -1
   K, J) = A(K, J) / D
    T J
    Q2$ = "Y" OR Q2$ = "Y" THEN GOSUB 1150
     2$ = "Y" OR Q2$ = "Y" THEN PRINT PERFORM; REDUCTION; ""
       = 1 TO N
       K GOTO 900
       A(J, K)
       ="Y" OR Q2$ ="Y" THEN PRINT "MULTIPLY ROW":K:"BY : A(I,K):AND SUB
        : I
        2 * N + 1 TO K STEP -1
         A(I, J) - MULT * A(K, I)


           THEN GOTO 1100


            "RESULTS", PRINT
            N
             + 1)
            :") = ":X(J)
```

```
1010 PRINT
1020 IF SEL > 1 THEN GOSUB 1250: GOSUB 1340
1030 PRINT
1040 PRINT: PRINT "DO YOU WANT TO REPEAT THE CALCULATIONS": PRINT WITH MINOR
1050 IF V$ = "Y" OR V$ "Y" THEN 1150 ELSE 1100
1060 CLS : GOTO 340
1070 PRINT : INPUT "DO YOU WANT TO RESET ALL THE COEFFICIENTS (Y/N)": W$
1080 IF W$ = "Y" OR W$ = "Y" THEN 990 ELSE 1100
1090 PRINT: INPUT "IS THE NEW SET OF THE SAME ORDER AS THE PREVOUS SET" WW$
1100 IF WW$ = "N" OR WW$ ="n" THEN PRINT CHR$(12) RUN 100
1110 CLS GOTO 220
1120 PRINT : PRINT
1130 PRINT ** END OF PROGRAM***
1140 LOAD "MAT, BAS", R

1150 END

1160 SUBROUTINE 1: PRINT THE; MATRIX
1170
1180 FOR KA = 1 TO N
1190 PRINT
1200 FOR J = 1 TO N + 1
1210 PRINT A(KA, J)
1220 NEXTJ: PRINT : NEXT KA: PRINT
1230 PRINT
1240 FOR DELAY = 1 TO 3000: NEXT
1250 RETURN

 260 SUBROUTINE 2: PRINT THE; INVERSE; OF; THE; MATRIX
 ?70 PRINT INVERSE; OF; MATRIX
  80 FOR KA = 1 TO N
  )0 PRINT
   0 FOR J = N + 2 TO 2 * N + 1
   ) PRINT A(KA, J)
     NEXT J: PRINT : NEXT KA: PRINT
     PRINT
     RETURN

    SUBROUTINE 3: CHECK THE PRODUCT OF THE MATRIX AND INVERSE
     RINT " PRODUCT OF THE MATRIX AND INVERSE SHOULD BE THE IDNTITY MATRIX"
     ?INT
      R I = 1 TO N
        J = 1 TO N
        J) = 0
       K = 1 TO N
        J) = C(I, J) + B(I, K) * A(K, J + N + 1)

         JSING "               :C(J.J)
          J AND ABS(C(I, J) - 1) < EPS THEN GOTO 1490
           J AND ABS(C(J, J)) < EPS THEN GOTO 1490
           AUTION: INVERSE MAY NOT BE CORRECT"
            RINT
            INT
```

```
   30 CLS KEY OFF
   *************************
   3S-JORDAND              *
   ON METHOD FOR           *
```

```
10 CLS KEY OFF
20 PRINT "***************************"
30 PRINT *                          *
40 PRINT *  JACOBI'S ITERATIVE      *
50 PRINT *        METHOD            *
60 PRINT *                          *
70 PRINT *       (JACOBIS.BAS)      *
80 PRINT *                          *
90 PRINT ***************************
100 FOR D = 1 TO 5000: NENT D: CLS
110 INPUT " ITERATION NUMBER"; N
120 R = 0: X = 0: Y = 0: Z = 0
130 FOR ITER = 1 TO N
140 X1 = (10 - Y - Z) / 5
150 Y1 = (7 - X + 2 * Z) / 6
160 Z1 = (16 - X + 3 * Y) / 7
170 R = R + i
180 X = Xi: Y = Yi: Z = Zi
190 PRINT "X"; (":R:") = ":X"
200 PRINT "Y"; (":R:") = ":Y"
210 PRINT "Z"; (":R:") = ":Z"
220 PRINT
230 NEXT ITER
240 IF X = Xi AND Y = Yi AND Z = Zi GOTO 270
250 PRINT : GOTO 150
260 LOAD "A: MAT.BAS", R
270 END
```


```
LS KEY OFF
?INT "***************************"
 INT *                          *
 INT *  JACOBI'S ITERATIVE      *
  NT *        METHOD            *
  NT *                          *
   T *       (JACOBIS.BAS)      *
   T *                          *
     ***************************
    = 1 TO 5000: NENT D: CLS
     " ITERATION NUMBER"; N
      X = 0: Y = 0: Z = 0
      R = 1 TO N
       - Y - Z) / 5
        - X + 2 * Z) / 6
         - X + 3 * Y) / 7

           Yi: Z = Zi
           (":R:") = ":X"
            ":R:") = ":Y"
            ":R:") = ":Z"


              Y = Yi AND Z = Zi GOTO 270
              50
              S", R
```

PROGRAM OUTPUT- TEST DATA

GAUSS-SIEDEL ITERATIVE METHOD

K	$X_1^{(k)}$	$X_2^{(k)}$	$X_3^{(k)}$
0	0. 0000	0. 0000	0. 0000
1	6. 0000	3. 0000	-5. 2500
2	3.7500	3. 7500	-5. 1563
3	3. 4688	3. 6094	-5. 0977
4	3. 2930	3. 7559	-5. 0610
5	3. 1831	3. 8474	-5. 0382
6	3. 1144	3. 9046	-5. 0238
7	3. 0715	3. 9404	-5. 0149
8	3. 0447	3. 9627	-5. 0093
9	3. 0279	3. 9767	-5. 0058
10	3. 0175	3. 9854	-5. 0036
11	3.0109	3. 9909	-5. 0023
12	3. 0068	3. 9943	-5. 0014
13	3. 0042	3. 9964	-5. 0009
14	3. 0027	3. 9977	-5. 0006
15	3. 0016	3. 9986	-5. 0003
16	3. 0010	3. 9991	-5. 0002
17	3. 0006	3. 9995	-5. 0001
18	3. 0004	3. 9996	-5. 0000
19	3. 0002	3. 9997	-5. 0000
20	3. 0001	3. 9998	-5. 0000
21	3. 0001	3. 9999	-5. 0000
22	3. 0000	3. 9999	-5. 0000

PROGRAM OUTPUT - TEST DATA

JACOBI ITERATIVE METHOD

K	$X_1^{(k)}$	$X_2^{(k)}$	$X_3^{(k)}$
0	0.0000	0.0000	0.0000
1	1.0000	2.5000	-1.0000
2	0.8000	2.0000	-1.1250
3	1.0625	2.0313	-10625
4	1.0234	1.9687	-0.9766
5	0.9961	1.9941	-0.9805
6	0.9917	2.0039	-1.0005
7	0.9993	2.0030	-1.0051
8	0.0018	1.9996	-0.0110
9	1.0006	1.9992	-1.9999
10	0.9997	1.9998	-1.9995
11	0.9998	2.0002	-1.0001
12	1.0000	2.0000	-1.0002
13	1.0000	1.9990	-1.0000
14	1.0000	1.9999	-0.9999
15	0.9999	2.0000	-0.9999
16	1.0000	2.0000	-1.0000