CONTINGENCY ANALYSIS OF POWER SYSTEM NETWORKS

By

MOHD HAFIZ BIN ABDUL RAUF

FINAL PROJECT REPORT

Submitted to the Electrical & Electronics Engineering Programme
in Partial Fulfillment of the Requirements
for the Degree
Bachelor of Engineering (Hons)
(Electrical & Electronics Engineering)

Universiti Teknologi Petronas
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

CONTINGENCY ANALYSIS OF POWER SYSTEM NETWORKS

by

Mohd Hafiz bin Abdul Rauf

A project dissertation submitted to the
Electrical & Electronics Engineering Programme
Universiti Teknologi PETRONAS
in partial fulfilment of the requirement for the
Bachelor of Engineering (Hons)
(Electrical & Electronics Engineering)

Approved:

[illegible] 14.12.2006

Professor Dr Ramiah Jegatheesan
Project Supervisor

UNIVERSITI TEKNOLOGI PETRONAS
TRONOH, PERAK

December 2006

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

[illegible]

Mohd Hafiz bin Abdul Rauf

ABSTRACT

In this modern power system, planning and construction of the system with outage minimization as a major design criterion is becoming more important. However, even with the best possible construction and operating practices, outages still can occur. Several techniques have been developed to analyze the status of the system after the real outage occurs. This project presents an alternative way of conducting power flow analysis in determining the new bus voltages and line currents during outage occurrence. The algorithms have been accommodated in one complete program making it an effective tool for contingency analysis. The program has been tested on an IEEE bus test case and the remarkable results obtained in terms of its reduced computation time and accuracy showed that this technique could be implemented, particularly when involving practical work with a lot of calculations.

ACKNOWLEDGEMENTS

This dissertation could not have been written without Dr. Ramiah Jegatheesan who not only served as my supervisor but also encouraged and challenged me throughout my academic program as well as this project. He patiently guided me through the dissertation process, never accepting less than my best efforts.

Finally, I would like to thank my mother and father, who gave me the opportunity and the spirit to educate myself. Mom, Dad, you are the best. To all my sisters and little brother, I hope I can serve as inspiration to all of you. If I can do it, so can you.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

List of Figures

List of Tables

LIST OF ABBREVIATIONS

IEEE	Institute of Electrical and Electronics Enginners
MATLAB	Matrix Laboratory (Mathworks, Inc.)
GUI	Graphical User Interface
AC	Alternating Current
DC	Direct Current
AEP	American Electric Power System
$\mathbf{Z_{bus}}$	Bus Impedance Matrix
$\mathbf{Y_{bus}}$	Bus Admittance Matrix
HLCA	Half Line Charging Admittance
RAM	Random Access Memory

CHAPTER 1
INTRODUCTION

Line outages, whether it is planned or unplanned are the common issue in power system networks. That is why planning and construction of the system with outage minimization as a major design criterion is becoming more important and more common. Of course, the primary requirements of a reliable power system are the proper line construction and maintenance and proper over-current device placement and coordination [1]. Nevertheless, even with the best possible construction and operating practices, outages still can occur.

Whenever an outage occurs, an effective and economical way to improve the reliability of the power system is by restoring the power temporarily using alternative feeds [1]. However, what it more important is to determine the status of the system when any possible line outages occur. This is the point where many techniques of power system analysis have been introduced and developed, such as power flow analysis, contingency analysis, system restoration, load shedding, etc [2].

A system contingency can be defined as a disturbance that can occur in the network and can result in possible loss of parts of the network like buses, lines, transformers or power unit in any of the network areas. Contingency analysis is an evaluation of an existing or proposed electrical distribution system to determine the capability of the system to restore power using alternative feeds [1]. It is performed to assess the status of the power system network for any possible line outage before it is actually occurs.

When a line is switched onto or off the system through the action of circuit breakers, line currents are redistributed throughout the network and bus voltages change. These new steady steady-state bus voltages and line currents can be **predicted quickly** by what is called the *contingency analysis* program [6]. The large-scale network models used for this type of evaluation, like those used for fault calculations, do not have the be exact because the system planners and operators, who must undertake hundreds of studies in a short time period, are more concerned with knowing if *overload levels* of current and *out-of-limit* voltages and power flow exist than with finding the exact values of those quantities [6].

This project aimed at developing the **Contingency Analysis program,** to calculate new steady state bus voltages and line currents during the line outage and other contingencies. The program uses the factored matrices of Bus Admittance Matrix algorithm, which is the main advantage of this program. It is written and demonstrated in **MATLAB**. In realizing the effectiveness of the developed program, validation was performed on an **IEEE** Bus System and the results were compared to the results using the traditional power flow analysis and it indicates a remarkable advantage particularly in its computation time and the accuracy. Results obtained from the studies indicated the proposed technique could be implemented, particularly when involving practical work with a lot of calculations.

CHAPTER 2
LITERATURE REVIEW

Power system network can be characterized by two methods namely, Bus Impedance Matrix, $\mathbf{Z_{bus}}$ and Bus Admittance Matrix, $\mathbf{Y_{bus}}$, which are inverse of each other. Practically, it is much easier to construct the bus admittance matrix, rather than bus impedance matrix.

2.1 Bus Impedance Matrix $\mathbf{Z_{bus}}$ [4]

Bus Impedance Matrix, $\mathbf{Z_{bus}}$ of a power network can be obtained by inverting the bus admittance matrix, $\mathbf{Y_{bus}}$, which is easy to construct. However, when the order of matrix is large, direct inversion requires more core storage and enormous computer computation time. Therefore, inversion of $\mathbf{Y_{bus}}$ is prohibited for large size network.

$\mathbf{Z_{bus}}$ can be constructed by adding the network elements one after another. Using impedance parameters, performance equations in bus frame of reference can be written as:

$$\boldsymbol{E_{bus}} = \boldsymbol{Z_{bus}}\, \boldsymbol{I_{bus}} \tag{1}$$

In expanded form, the above becomes

$$\begin{bmatrix} E_1 \\ E_2 \\ \vdots \\ E_N \end{bmatrix} = \begin{bmatrix} Z_{11} & \cdots & \cdots & Z_{1N} \\ Z_{21} & Z_{22} & \cdots & Z_{2N} \\ \vdots & \vdots & \vdots & \vdots \\ Z_{N1} & Z_{N2} & \cdots & Z_{NN} \end{bmatrix} \begin{bmatrix} I_1 \\ I_2 \\ \vdots \\ I_4 \end{bmatrix} \tag{2}$$

From this we can write

$$E_p = Z_{p1} I_1 + Z_{p2} I_2 + \ldots.. + Z_{pq} I_q + \ldots... + Z_{pN} I_N \tag{3}$$

By injecting 1 p.u. current at bus q and other bus currents are set to zero, we can obtained the Z_{pq} by measuring E_p. The indices p and q can be varied from 1 to N.

When constructing $\mathbf{Z_{bus}}$ building algorithm, elements are added one by one. At any stage, the added element may be a branch or a link. A branch is the added element which will create a new bus in the network. On the other hand, by adding link, no new bus will be created.

As given in [4], the summary of the Bus Impedance Matrix algorithm formulas are as Table 1 below:

	p is not reference bus	**p is the reference bus**
Added element p-q is a branch	$Z_{qi} = Z_{pi}$ $i = 1, 2, \ldots, m$ $i \neq q$ $Z_{qq} = Z_{pq} + Z_a$	$Z_{qi} = 0$ $i = 1, 2, \ldots, m$ $i \neq q$ $Z_{qq} = Z_a$
Added element p-q is a link	$Z_{li} = Z_{pi} - Z_{qi}$ $i = 1, 2, \ldots, m$ $i \neq l$ $Z_{ll} = Z_{pl} - Z_{ql} + z_a$	$Z_{li} = - Z_{qi}$ $i = 1, 2, \ldots, m$ $i \neq l$ $Z_{ll} = - Z_{ql} + z_a$
	$Z_{ij(modified)} = Z_{ij(before\ modification)} - \dfrac{Z_{il} Z_{lj}}{Z_{ll}}$ $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, m$	

Table 1: Summary of the Bus Impedance Matrix algorithm formulas

2.2 Power Flow Analysis

Power flow analysis, also known as load flow, is an important part of power system analysis. It serves the purpose of planning, control of existing system as well as planning its future expansion [5]. Unlike traditional circuit analysis, a power flow study usually uses simplified notation such as a one-line diagram and per-unit system, and focuses on various forms of AC power (i.e. reactive, real, and apparent) rather than voltage and current.

The great importance of power flow or load-flow studies is in the planning the future expansion of power systems as well as in determining the best operation of existing systems. The principal information obtained from the power flow study is the magnitude and phase angle of the voltage at each bus and the real and reactive power flowing in each line.

When conducting a power flow analysis, the system is assumed to be operating under balanced conditions and a single phase model is used. Four quantities are associated with each bus. These are voltage magnitude $|V|$, phase angle δ, real power P, and reactive power Q. Generally, the system buses can be classified into 3 types: [5]

1) **Slack bus**

   It also known as slack or swing bus, which will be the reference bus. In slack bus, voltage magnitude and phase angle are specified.

2) **Load buses (P - Q buses)**

   In these buses the active and reactive power are specified.

3) **Regulated buses (P - V buses)**

   These buses are the generator buses, where the real power and voltage magnitude are specified.

There are several solutions, or methods introduced to solve for the power flow analysis problem, namely:

1) **Gauss-Seidel Method**
2) **Newton-Raphson Method**
3) **Fast Decoupled Method**

2.3 Graphical User Interface (GUI)

A graphical user interface (GUI) is a graphical display that contains devices, or components, that enable a user to perform interactive tasks. To perform these tasks, the user of the GUI does not have to create a script or type commands at the command line. Often, the user does not have to know the details of the task at hand.

The GUI components can be menus, toolbars, push buttons, radio buttons, list boxes, and sliders. In MATLAB, a GUI can also display data in tabular form or as plots, and can group related components. The following figure illustrates a simple GUI.

This project will utilize this MATLAB's feature to make the program more user-friendly and thus make it easier for conducting the analysis.

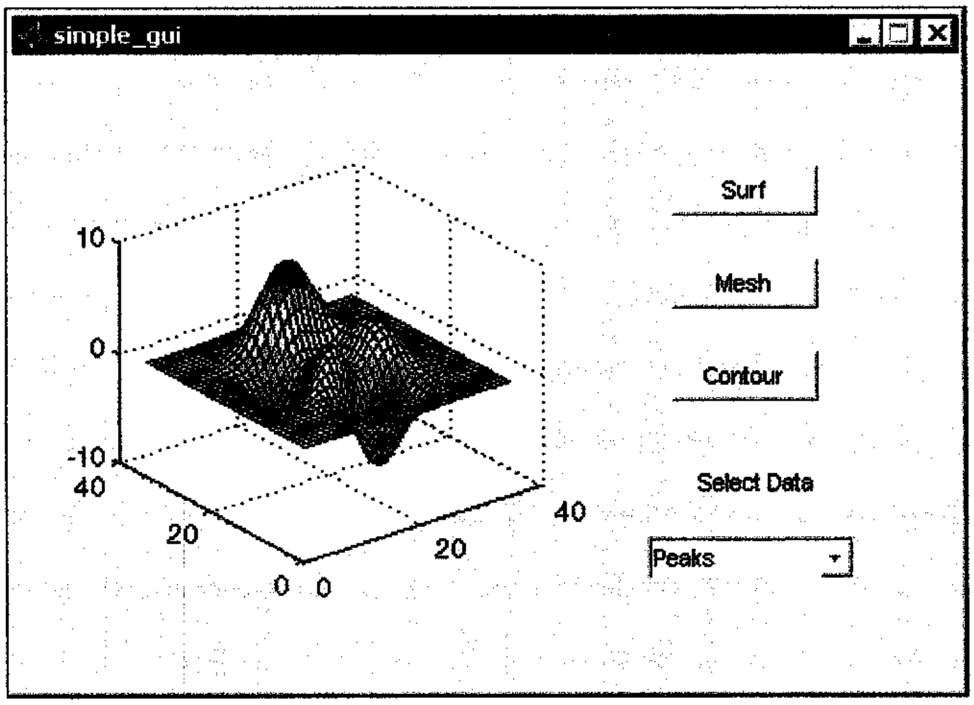


Figure 1 : A simple GUI using MATLAB

CHAPTER 3
METHODOLOGY

In contingency analysis, the network models used do not have to be exact because the power engineers and system operators are more interested in knowing whether or not an insecure or vulnerable condition exists in the steady state following any of the outages.

Therefore, several approximations are made. Often, the resistance is considered negligible and the network model becomes purely reactive. Line charging and off-nominal tap charging of transformers are also omitted [6]. Accordingly, to test for the effects of line and transformer outages on the bus voltages and line flows in the network, approximate ac power-flow techniques are generally employed since they can provide a **fast solution** of the many test cases which need to be run.

In many cases, linear models are considered satisfactory and the principle of superposition is then employed. Methods of contingency analysis which used the factored Bus Admittance Matrix ($\mathbf{Y_{bus}}$) becomes attractive from computational viewpoint, especially if the loads can be treated as constant current injections into the various buses of the system [6]. Hence, the concept of compensating currents is developed, to allow the use of existing system $\mathbf{Z_{bus}}$ without having to modify it whenever a contingency occurs.

3.1 Bus Admittance Matrix

While doing symmetrical short circuit analysis, the bus impedance matrix $\mathbf{Z_{bus}}$ is employed. However, when the system is large, construction of $\mathbf{Z_{bus}}$ matrix itself is tedious. Furthermore, since $\mathbf{Z_{bus}}$ matrix is always full, more computer storage is required to store the matrix. Moreover, modification of the system such as removal or addition of one link calls for recalculation of all the elements in $\mathbf{Z_{bus}}$ matrix.

On the other hand, the bus admittance matrix $\mathbf{Y_{bus}}$ can be easily obtained even for large power system network. The main advantage of this matrix is that, it is always sparse (having many elements of zero) and thus requires less storage capacity. For instance, whenever an element $a - b$ is added, only four elements Y_{aa}, Y_{bb}, Y_{ab} and Y_{ba} need to be modified and the other elements in the $\mathbf{Y_{bus}}$ matrix remain unaltered.

3.1.1 Bus Admittance Matrix building algorithm [6]

This matrix can be build by adding one element at a time. To start with, all the elements of bus admittance matrix are assumed to be **zero**. Then, when a network element of admittance y_{ii} is added between buses p and q, **ONLY 4 elements** of bus admittance matrix are to be modified as below.

$$Y_{pp\ new} = Y_{pp\ old} + y_{ii}$$

$$Y_{qq\ new} = Y_{qq\ old} + y_{ii}$$

$$Y_{pq\ new} = Y_{pq\ old} + y_{ii} \qquad (4)$$

$$Y_{qp\ new} = Y_{qp\ old} + y_{ii}$$

Even with the orientation is from bus q to p, the result will be the same. However, if bus q happens to be reference bus, then Y_{pp} alone gets modified as:

$$Y_{pp\ new} = Y_{pp\ old} + y_{ii} \qquad (5)$$

i.e. if y_{p0} is connected between bus p and reference bus, then

$$Y_{pp\ new} = Y_{pp\ old} + y_{p0} \qquad (6)$$

This method of constructing the bus admittance matrix by adding the elements one by one and modifying the bus admittance matrix each time, is very well suited for large network. Each time, one network element data is read and the bus admittance matrix is updated. When all the network elements are added, the final bus admittance matrix will be obtained.

When any fault occurs at p^{th} bus, only the p^{th} column of the $\mathbf{Z_{bus}}$ matrix and not the full $\mathbf{Z_{bus}}$ matrix is needed. However, it is also possible to obtain the p^{th} column of $\mathbf{Z_{bus}}$ matrix, $Z_{bus}^{(p)}$ from the $\mathbf{Y_{bus}}$ matrix. In fault study, the bus at which the fault occurs (value of p) may keep changing. Therefore, if the $\mathbf{Y_{bus}}$ is factored and the factored matrices are available, any required column of $\mathbf{Z_{bus}}$ matrix can be obtained easily.

The p^{th} column of the $\mathbf{Z_{bus}}$ matrix, $Z_{bus}^{(p)}$ can be obtained from the factor matrices of $\mathbf{Y_{bus}}$ matrix. Previously, we know already that

$$Z_{bus} I_{bus} = V_{bus} \tag{7}$$

Thus,

$$\begin{bmatrix} Z_{11} & Z_{12} & \cdots & Z_{1p} & \cdots & Z_{1N} \\ Z_{21} & Z_{22} & \cdots & Z_{2p} & \cdots & Z_{2N} \\ \vdots & \vdots & & \vdots & & \vdots \\ Z_{p1} & Z_{12} & \cdots & Z_{pp} & \cdots & Z_{pN} \\ \vdots & \vdots & & \vdots & & \vdots \\ Z_{N1} & Z_{N2} & \cdots & Z_{Np} & \cdots & Z_{NN} \end{bmatrix} \begin{bmatrix} I_1 \\ I_2 \\ \vdots \\ I_p \\ \vdots \\ I_N \end{bmatrix} = \begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ V_p \\ \vdots \\ V_N \end{bmatrix} \tag{8}$$

Consider the bus current vector:

$$I_{bus} = \begin{bmatrix} I_1 \\ I_2 \\ \vdots \\ I_p \\ \vdots \\ I_N \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} \quad \text{and denote this as } \mathbf{1}_p \tag{9}$$

Substituting $I_{bus} = 1_p$ in the equation (8), we will get

$$\begin{bmatrix} Z_{1p} \\ Z_{2p} \\ \vdots \\ Z_{pp} \\ \vdots \\ Z_{Np} \end{bmatrix} = \begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ V_P \\ \vdots \\ V_N \end{bmatrix} \text{ i.e. } Z_{bus}^{(p)} = \begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ V_P \\ \vdots \\ V_N \end{bmatrix} \tag{10}$$

Thus, by injecting 1 p.u current at bus p and set the other currents to be zero (I_{bus}=1_p) $Z_{bus}^{(p)}$ is *numerically equal* to V_{bus} obtained from (7). The inverse form of (7) is $Y_{bus} V_{bus} = I_{bus}$.

In the subsequent chapters, we will see how the $\mathbf{Y_{bus}}$ is factorized and how the factored matrices are used to obtained the p^{th} column of the $\mathbf{Z_{bus}}$.

3.1.2 Factored Matrices of $\mathbf{Y_{bus}}$ Matrix [6]

This is the important part of the program. Once the factored matrices of $\mathbf{Y_{bus}}$ matrix are obtained, any required column of the $\mathbf{Z_{bus}}$ can be extracted easily.

The **symmetrical** matrix $\mathbf{Y_{bus}}$ can be factored as:

$$Y_{bus} = L\,D\,L^t \tag{11}$$

where L is a lower triangular matrix

D is a diagonal matrix

L^t is the transpose of L matrix

Because of $Y_{bus} = LDL^t$, we can state that:

$Z_{bus}^{(p)}$ is *numerically equal* to V_{bus} obtained from $LDL^tV_{bus} = I_{bus}$, for $I_{bus} = 1_p$.

If the fault occurs at some other bus, say k, then $Z_{bus}^{(k)}$ can be obtained by solving $LDL^tV_{bus} = I_{bus}$ for V_{bus} taking $I_{bus} = 1_k$. This calculation requires only back substitution as the factor matrices are already available. Thus, only a repeat solution is carried out with different values for right hand side vector.

Let us consider a 4 bus system described by:

$$\begin{bmatrix} Y_{11} & Y_{21} & Y_{31} & Y_{41} \\ Y_{12} & Y_{22} & Y_{32} & Y_{42} \\ Y_{13} & Y_{23} & Y_{33} & Y_{43} \\ Y_{14} & Y_{24} & Y_{34} & Y_{44} \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \end{bmatrix} = \begin{bmatrix} I_1 \\ I_2 \\ I_3 \\ I_4 \end{bmatrix} \tag{12}$$

The coefficient matrix $\mathbf{Y_{bus}}$ can be written as the product of three matrices L, D and L^t. Therefore, $Y_{bus} = LDL^t$. (13)

Matrix L is a lower triangular matrix of the form

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \ell_{21} & 1 & 0 & 0 \\ \ell_{31} & \ell_{32} & 1 & 0 \\ \ell_{41} & \ell_{42} & \ell_{43} & 1 \end{bmatrix} \tag{14}$$

and matrix D is a diagonal matrix of the form

$$D = \begin{bmatrix} d_{11} & 0 & 0 & 0 \\ 0 & d_{22} & 0 & 0 \\ 0 & 0 & d_{33} & 0 \\ 0 & 0 & 0 & d_{44} \end{bmatrix} \tag{15}$$

The matrix L^t is only the transpose of matrix L.

$$L^t = \begin{bmatrix} 1 & \ell_{21} & \ell_{31} & \ell_{41} \\ 0 & 1 & \ell_{32} & \ell_{42} \\ 0 & 0 & 1 & \ell_{43} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{16}$$

Thus, using equation (14) and (15) in equation (13), $\mathbf{Y_{bus}}$ can also be written as:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ \ell_{21} & 1 & 0 & 0 \\ \ell_{31} & \ell_{32} & 1 & 0 \\ \ell_{41} & \ell_{42} & \ell_{43} & 1 \end{bmatrix} \begin{bmatrix} d_{11} & 0 & 0 & 0 \\ 0 & d_{22} & 0 & 0 \\ 0 & 0 & d_{33} & 0 \\ 0 & 0 & 0 & d_{44} \end{bmatrix} \begin{bmatrix} 1 & \ell_{21} & \ell_{31} & \ell_{41} \\ 0 & 1 & \ell_{32} & \ell_{42} \\ 0 & 0 & 1 & \ell_{43} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} Y_{11} & Y_{21} & Y_{31} & Y_{41} \\ Y_{12} & Y_{22} & Y_{32} & Y_{42} \\ Y_{13} & Y_{23} & Y_{33} & Y_{43} \\ Y_{14} & Y_{24} & Y_{34} & Y_{44} \end{bmatrix}$$

After multiplying first 2 matrices, the result as follows:

$$\begin{bmatrix} d_{11} & 0 & 0 & 0 \\ \ell_{21}d_{11} & d_{22} & 0 & 0 \\ \ell_{31}d_{11} & \ell_{32}d_{22} & d_{33} & 0 \\ \ell_{41}d_{11} & \ell_{42}d_{22} & \ell_{43}d_{33} & d_{44} \end{bmatrix} \begin{bmatrix} 1 & \ell_{21} & \ell_{31} & \ell_{41} \\ 0 & 1 & \ell_{32} & \ell_{42} \\ 0 & 0 & 1 & \ell_{43} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} Y_{11} & Y_{21} & Y_{31} & Y_{41} \\ Y_{12} & Y_{22} & Y_{32} & Y_{42} \\ Y_{13} & Y_{23} & Y_{33} & Y_{43} \\ Y_{14} & Y_{24} & Y_{34} & Y_{44} \end{bmatrix} \tag{17}$$

By looking to the above equations, we can generalize the equations to obtain the elements of the matrices. The equations are as follows:

$$d_{ii} = Y_{ii} - \sum_{k=1}^{i-1} \ell_{ik}^{2} d_{kk} \tag{18}$$

$$\ell_{ij} = (Y_{ij} - \sum_{k=1}^{j-1} \ell_{ik} d_{kk} \ell_{jk}) / d_{jj} \tag{19}$$

The elements of the factor matrices *L* and *D* can be obtained using equations (18) and (19). For matrix of order *N*, the elements of factor matrices *L* and *D* are calculated in the order:

$$d_{ii}, \ell_{i+1}, \ell_{i+2i}, \cdots\cdots, \ell_{Ni} \text{ for i} = 1, 2, 3, \ldots\ldots\ldots, N \tag{20}$$

The *L* and *D* matrices are combined into one matrix only, to reduce the computer storage.

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

3.1.3 Solving $Y_{bus}V_{bus} = I_{bus}$ for V_{bus} using the factor matrices [6]

Now the problem is to solve for vector V_{bus} for a given value of vector I_{bus} using

$$LDL^{t}V_{bus} = I_{bus}\,. \tag{22}$$

Let us define two intermediate dummy vectors *V'* and *V"* as

$$L^{t} = V' \tag{23}$$

$$DV' = V'' \tag{24}$$

$$\text{Then } LV'' = I_{bus} \tag{25}$$

In order to solve for the unknown voltage vector V_{bus}, the three equations above must be solved in the order equation (25), (24) and (23).

For a 4 bus system, equation (25) is:

$$\begin{bmatrix} d_{11} & 0 & 0 & 0 \\ \ell_{21} & d_{22} & 0 & 0 \\ \ell_{31} & \ell_{32} & d_{33} & 0 \\ \ell_{41} & \ell_{42} & \ell_{43} & d_{44} \end{bmatrix} \begin{bmatrix} V_1'' \\ V_2'' \\ V_3'' \\ V_4'' \end{bmatrix} = \begin{bmatrix} I_1 \\ I_2 \\ I_3 \\ I_4 \end{bmatrix} \tag{26}$$

The elements of *V"* vector are calculated using the equation below:

$$V_i'' = I_i - \sum_{k=1}^{i-1} \ell_{ik} V_k \qquad i = 1, 2, \ldots\ldots\ldots, N \tag{27}$$

Now equation (24) can be written as:

$$\begin{bmatrix} d_{11} & 0 & 0 & 0 \\ 0 & d_{22} & 0 & 0 \\ 0 & 0 & d_{33} & 0 \\ 0 & 0 & 0 & d_{44} \end{bmatrix} \begin{bmatrix} V_1' \\ V_2' \\ V_3' \\ V_4' \end{bmatrix} = \begin{bmatrix} V_1'' \\ V_2'' \\ V_3'' \\ V_4'' \end{bmatrix} \tag{28}$$

From this above equation, it is clear that the elements of *V'* vector are calculated using the equation below:

$$V_i' = V_i'' / d_{ii} \qquad \mathrm{i} = 1, 2, \ldots\ldots, N \tag{29}$$

Finally, equation (23) can be written in the form of

$$\begin{bmatrix} 1 & \ell_{21} & \ell_{31} & \ell_{41} \\ 0 & 1 & \ell_{32} & \ell_{42} \\ 0 & 0 & 1 & \ell_{43} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \end{bmatrix} = \begin{bmatrix} V_1' \\ V_2' \\ V_3' \\ V_4' \end{bmatrix} \tag{30}$$

From this equation, we can generalize that

$$V_i = V_i' - \sum_{k=i+1}^{N1} \ell_{ki} V_k \qquad \mathrm{i} = N, N\text{-}1, \ldots\ldots\ldots\ldots, 1 \tag{31}$$

It is to be noticed that, while calculating elements of vector V_{bus}, from the above equation, V_i s are computed in the order of i = *N*, *N*-1, ……., 1

3.1.4 Summary of formulas

Elements of factor matrices are calculated from

$$d_{ii} = Y_{ii} - \sum_{k=1}^{i-1} \ell_{ik}{}^2 d_{kk} \quad \text{and} \quad \ell_{ij} = (Y_{ij} - \sum_{k=1}^{j-1} \ell_{ik} d_{kk} \ell_{jk}) / d_{jj}$$

following the order $d_{ii}, \ell_{i+1}, \ell_{i+2i}, \cdots\cdots, \ell_{Ni}$ for i = 1, 2, 3, ..., N

V_{bus} solution is obtained from these equations:

$$V_i'' = I_i - \sum_{k=1}^{i-1} \ell_{ik} V_k \qquad \text{i} = 1, 2, \ldots\ldots\ldots, N \tag{32}$$

$$V_i' = V_i'' / d_{ii} \qquad \text{i} = 1, 2, \ldots\ldots, N \tag{33}$$

$$V_i = V_i' - \sum_{k=i+1}^{N1} \ell_{ki} V_k \qquad \text{i} = \text{N}, \text{N-1}, \ldots\ldots\ldots\ldots, 1 \tag{34}$$

3.1.5 Calculation of the vector $Z_{bus}^{(m-n)}$ from factors of Y_{bus} matrix [6]

In contingency analysis, if a contingency occur between line m and n, we need to calculate the difference of the two vectors ($Z_{bus}^{(m)}$ – $Z_{bus}^{(n)}$) where $Z_{bus}^{(m)}$ and $Z_{bus}^{(n)}$ are the m^{th} and n^{th} column of the bus impedance matrix $\mathbf{Z_{bus}}$, respectively.

Let $Z_{bus}^{(m-n)} = Z_{bus}^{(m)} - Z_{bus}^{(n)}$ (35)

Again let us define vector $1_{(m-n)}$ as

$$1_{(m-n)} = \begin{array}{c} 1 \\ \\ \\ m \\ \\ \\ \\ n \\ \\ \\ N \end{array} \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \\ -1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \tag{36}$$

In $V_{bus} = Z_{bus}I_{bus}$, if $I_{bus} = 1_{(m-n)}$ (1 p.u current is injected at bus *m* and -1 p.u current is injected at bus *n*), then the resultant vector will be the difference of the m^{th} and n^{th} column of the **Zbus**, respectively.

Thus $Z_{bus}^{(m-n)}$ is *numerically equals* to V_{bus}, obtained from $V_{bus} = Z_{bus}I_{bus}$ for $I_{bus} = 1_{(m-n)}$. Equivalently, $Z_{bus}^{(m-n)}$ can be obtained solving $Y_{bus}V_{bus} = I_{bus}$ for V_{bus} corresponding to $I_{bus} = 1_{(m-n)}$.

Knowing that **Ybus** can be factored as LDL^t, we can now state that $Z_{bus}^{(m-n)}$ can be obtained by solving $LDL^tV_{bus} = I_{bus}$ for V_{bus} corresponding to $I_{bus} = 1_{(m-n)}$.

3.2 Addition of one line [6]

When considering line addition to or removal from an existing system, it is not always necessary to build a new **Zbus** or to calculate new triangular factors of **Ybus**, especially if the only interest is to establish the impact of the changes on bus voltages. This is where the *compensating currents* concept is developed as an alternative procedure. The compensating currents are injected into the existing system to account for the effect of the line changes.

Consider the original system with voltages $V_1, V_2, \ldots, V_N$ due to current injection $I_1, I_2, \ldots, I_N$ as illustrated in figure 2 below:

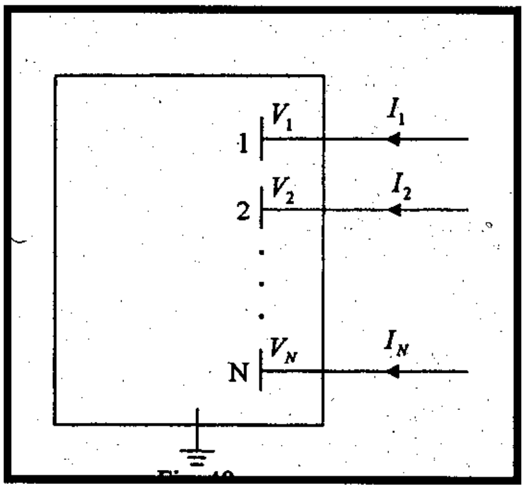


Figure 2 : Original System

We assume that the bus voltages $V_1, V_2, \ldots, V_N$ produced in the original system by the current injections $I_1, I_2, \ldots, I_N$ are known. Suppose that a line of impedance z_a is added between buses m and n. The current injections are fixed in value and therefore are unaffected by the addition of line of impedance z_a. Let the changed bus voltages be $V_1', V_2', \ldots, V_m', \ldots, V_n', \ldots, V_N'$ a shown in Figure 4 on the next page.

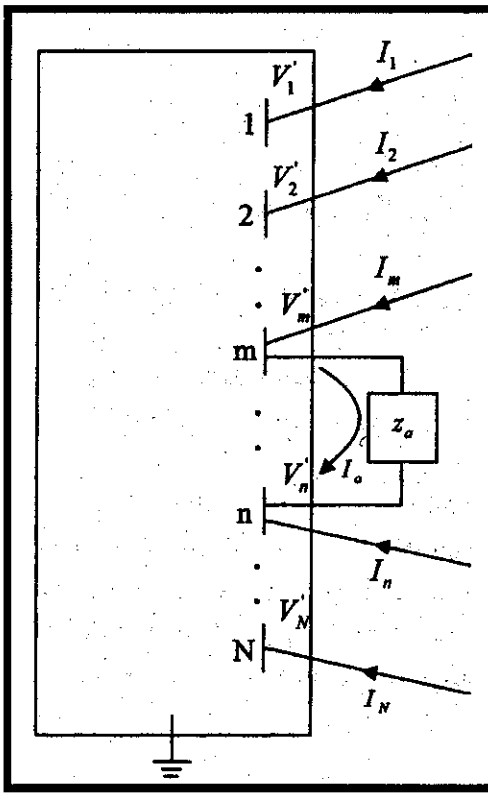


Figure 3

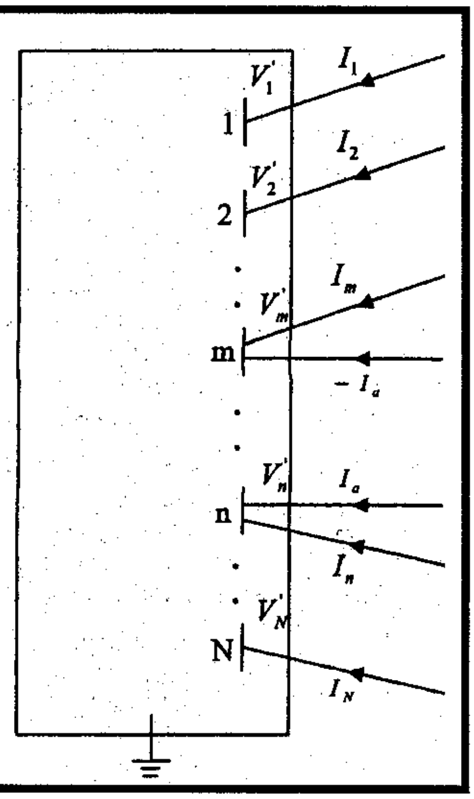


Figure 4

Addition of line causes a current flow of I_a as shown in Figure 3. We have

$$z_a I_a = V_{bus}' - V_n'$$

$$= \begin{array}{c} \begin{array}{ccccccc} & & m & & n & & \\ [\,0 \ldots 0 & & 1 & 0 \ldots 0 & -1 & & 0 \ldots 0\,] \end{array} \end{array} \begin{bmatrix} V_1' \\ V_2' \\ \vdots \\ V_m' \\ \vdots \\ V_n' \\ \vdots \\ V_N' \end{bmatrix} \tag{37}$$

Defining

$$A_c = \begin{array}{ccccccc} & & m & & n & & \\ [0 \ldots 0 & & 1 & 0 \ldots 0 & -1 & & 0 \ldots 0] \end{array} \qquad (38)$$

we get

$$z_a I_a = A_c V'_{bus} \qquad (39)$$

The effect of current I_a shown in Figure 3 can be replaced by current injection of - I_a at bus m and I_a at bus n as shown in Figure 4. By examining Figure 4, it reveals that the bus voltages $V_1', V_2', \ldots, V_N'$ are due to:

1) Current injections $I_1, I_2, \ldots, I_N$ and
2) Current injections - I_a at bus m and I_a at bus n

Hence, to determine the bus voltages $V_1', V_2', \ldots, V_N'$ we shall now apply the **Superposition Theorem**. When current injections $I_1, I_2, \ldots, I_N$ **alone** are there, bus voltages will be the same as the bus voltages in the original system (refer Figure 2) denoted by V_{bus}. When current injections $-I_a$ at bus m and I_a at bus n are **alone** there, bus voltages will be equal to $Z_{bus} I_{comp}$ where

$$I_{comp} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ -I_a \\ 0 \\ \vdots \\ 0 \\ I_a \\ 0 \\ \vdots \\ 0 \end{bmatrix} = -\begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \\ -1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} I_a = -A_c^t I_a \qquad (40)$$

Therefore;

$$V'_{bus'} = V_{bus} + Z_{bus} I_{comp} = V_{bus} - Z_{bus} A_c^t I_a \qquad (41)$$

V'_{bus} can be obtained combining equation (39) and equation (41).

Substituting equation (41) into (39),

$$z_a I_a = A_c V_{bus} - A_c Z_{bus} A_c^t I_a \tag{42}$$

Defining $Z = z_a + A_c Z_{bus} A_c^t$ (43)

we get $ZI_a = V_m - V_n$, thus we get

$$I_a = (V_m - V_n)/Z \tag{44}$$

Now equation (41) can be written as

$$V_{bus'}^{'} = V_{bus} + \Delta V \tag{45}$$

Where $\Delta V = -Z_{bus} A_c^t I_a$ (46)

Thus, new bus voltages $V_{bus}^{'}$ can be written as follows:

1) Calculate Z from equation (43)
2) Calculate I_a from equation (44)
3) Calculation ΔV from equation (46)
4) Calculation $V_{bus}^{'}$ from equation (45)

Note that equation (42) and (45) contain $\mathbf{Z_{bus}}$, the bus impedance matrix of the system, which is not available. Let us now see how this could be overcome.

Calculation of Z

Let us note that $A_c = [\,0 \ldots 0 \;\; \underset{m}{1} \;\; 0 \ldots 0 \;\; \underset{n}{-1} \;\; 0 \ldots 0]$

$$Z = z_a + A_c Z_{bus} A_c^t$$

$$= z_a + A_c \, Z_{bus}^{(m-n)} \tag{47}$$

Denoting $m\,[Z_{bus}^{(m-n)}]$ is the m^th element of $Z_{bus}^{(m-n)}$ and $n\,[Z_{bus}^{(m-n)}]$ is the n^th element of $Z_{bus}^{(m-n)}$ the above equation becomes

$$Z = z_a + m\,[Z_{bus}^{(m-n)}] - n\,[Z_{bus}^{(m-n)}] \tag{48}$$

Calculation of I_a

$$I_a = (V_m - V_n) / Z \qquad (49)$$

Calculation of ΔV

$$\Delta V = - Z_{bus}\ A_c^t I_a \qquad (50)$$

$$= - Z_{bus}^{(m-n)} I_a$$

Calculation of new bus voltage $V_{bus}^{'}$

$$V_{bus'}^{'} = V_{bus}^{'} + \Delta V \qquad (51)$$

It is to be noted that for the above calculations we need only the vector $Z_{bus}^{(m-n)}$ and not the bus impedance matrix $\mathbf{Z_{bus}}$. We should use the above four equations (48 – 51) to calculate Z, I_a, ΔV *and* $V_{bus}^{'}$.

3.3 Removal of one line

In most cases, line outage is more common due to fault or over loading. Hence it is more pertinent to study removal of line rather that addition of line. Accommodating removal of line is not at all difficult. Removal of a line $m - n$ with impedance z_a is equivalent to addition of line $m - n$ with impedance $-z_a$.

3.4 Addition / Removal of two lines

The method discussed can be extended to addition or removal of two lines. Let us say we have a power system for which the bus voltages are V_1', V_2', ..., V_N'. Two lines of impedance z_a and z_b are added between buses $m - n$ and $p - q$ respectively. It is required to find the new bus voltages V_1', V_2', ..., V_N'. The currents in the added lines will be I_a and I_b. Then

$$z_a I_a = V_m^{'} - V_n^{'} \ and \ z_b I_b = V_p^{'} - V_q^{'}$$

i.e.

$$\begin{bmatrix} z_a & 0 \\ 0 & z_b \end{bmatrix}\begin{bmatrix} I_a \\ I_b \end{bmatrix} = \begin{array}{c} \\ \left[\begin{array}{cccccccccccc} & & & m & & & & n & & & & \\ 0 & \cdots & 0 & 1 & 0 & \cdots & 0 & -1 & 0 & \cdots & \cdots & 0 \\ 0 & \cdots & \cdots & 0 & 1 & 0 & \cdots & 0 & -1 & 0 & \cdots & 0 \\ & & & & p & & & & q & & & \end{array}\right] \end{array} \begin{bmatrix} V_1' \\ \vdots \\ V_m' \\ \vdots \\ V_p' \\ \vdots \\ V_n' \\ \vdots \\ V_q' \\ \vdots \\ V_N' \end{bmatrix}$$

$$= A_c V_{bus}' \tag{52}$$

For this case the compensating current is

$$I_{comp} = \begin{array}{c|c|} & 0 \\ & \vdots \\ m & -I_a \\ & \vdots \\ p & -I_b \\ & 0 \\ n & I_a \\ & 0 \\ & \vdots \\ q & I_b \\ & 0 \end{array} = -\begin{bmatrix} 0 & 0 \\ \vdots & \vdots \\ -1 & 0 \\ \vdots & \vdots \\ 0 & -1 \\ & \\ 1 & 0 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 1 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} I_a \\ I_b \end{bmatrix} = -A_c^t \begin{bmatrix} I_a \\ I_b \end{bmatrix} \tag{53}$$

Using Superposition Theorem, new bus voltages can be written as

$$V_{bus}' = V_{bus} + Z_{bus} I_{comp} = V_{bus} - Z_{bus} A_c^t \begin{bmatrix} I_a \\ I_b \end{bmatrix} \tag{54}$$

Using the above equation in equation (51), we get

$$\begin{bmatrix} z_a & 0 \\ 0 & z_b \end{bmatrix}\begin{bmatrix} I_a \\ I_b \end{bmatrix} = A_c V_{bus} - A_c Z_{bus} A_c^t \begin{bmatrix} I_a \\ I_b \end{bmatrix} \tag{55}$$

i.e.

$$\left\{\begin{bmatrix} z_a & 0 \\ 0 & z_b \end{bmatrix} + A_c Z_{bus} A_c^t\right\}\begin{bmatrix} I_a \\ I_b \end{bmatrix} = A_c V_{bus} = \begin{bmatrix} V_m - V_n \\ V_p - V_q \end{bmatrix} \tag{56}$$

Defining $Z = \begin{bmatrix} z_a & 0 \\ 0 & z_b \end{bmatrix} + A_c Z_{bus} A_c^t$ (57)

The above equation becomes

$$Z\begin{bmatrix} I_a \\ I_b \end{bmatrix} = \begin{bmatrix} V_m - V_n \\ V_p - V_q \end{bmatrix} \quad \text{i.e.}$$

$$\begin{bmatrix} I_a \\ I_b \end{bmatrix} = Z^{-1}\begin{bmatrix} V_m - V_n \\ V_p - V_q \end{bmatrix} \tag{58}$$

Then, the new bus voltages can be written as

$$V'_{bus'} = V_{bus} + \Delta V \tag{59}$$

Where $\Delta V = -Z_{bus} A_c^t \begin{bmatrix} I_a \\ I_b \end{bmatrix}$ (60)

Thus, new bus voltages V'_{bus} can be determined as follows:

1) Calculate Z from equation (57)
2) Calculate $\begin{bmatrix} I_a \\ I_b \end{bmatrix}$ from equation (58)
3) Calculation ΔV from equation (60)
4) Calculation V'_{bus} from equation (59)

It is noted that equation (57) and (60) contain **Z_{bus}**, the bus impedance matrix of the system, which is not available. Let us now show that how this problem can be overcome.

Calculation of Z

$$Z = \begin{bmatrix} z_a & 0 \\ 0 & z_b \end{bmatrix} + \begin{bmatrix} (m-n)[Z_{bus}^{(m-n)}] & (m-n)[Z_{bus}^{(p-q)}] \\ (p-q)[Z_{bus}^{(m-n)}] & (p-q)[Z_{bus}^{(p-q)}] \end{bmatrix} \tag{61}$$

Calculation of $\begin{bmatrix} I_a \\ I_b \end{bmatrix}$

$$\begin{bmatrix} I_a \\ I_b \end{bmatrix} = Z^{-1}\begin{bmatrix} V_m - V_n \\ V_p - V_q \end{bmatrix} \tag{62}$$

Calculation of ΔV

$$\Delta V = -Z_{bus}A_c'\begin{bmatrix} I_a \\ I_b \end{bmatrix}$$

$$= -\begin{bmatrix} Z_{bus}^{(m-n)} & Z_{bus}^{(p-q)} \end{bmatrix}\begin{bmatrix} I_a \\ I_b \end{bmatrix} \tag{63}$$

Calculation of a new bus voltage V_{bus}'

$$V_{bus}' = V_{bus} + \Delta V \tag{64}$$

It is to be noted that for the above calculations, we need the vectors $Z_{bus}^{(m-n)}$ and $Z_{bus}^{(p-q)}$ only and not the bus impedance matrix **Z**$\mathbf{Z_{bus}}$. We should use the above four equations (61-64) to calculate Z, $\begin{bmatrix} I_a \\ I_b \end{bmatrix}$, ΔV and V_{bus}'.

3.5 Half Line Charging Admittance [5]

Consider the diagram below:

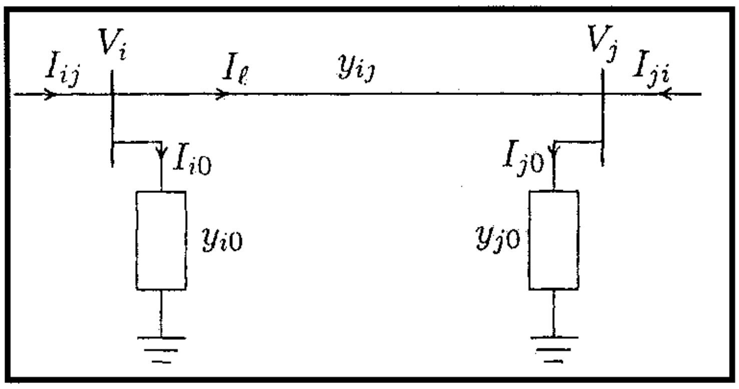


Figure 5

Line current from $i \rightarrow j$ is given by

$$I_{ij} = I_l + I_{i0} = y_{ij}(V_i - V_j) + y_{i0}V_i \tag{65}$$

Similarly, the line current I_{ij} measured from bus j to bus i and defined positive in the direction $j \rightarrow i$ is given by

$$I_{ji} = -I_l + I_{j0} = y_{ij}(V_j - V_i) + y_{j0}V_j \tag{66}$$

The complex powers $\boldsymbol{S_{ij}}$ from bus i to j and $\boldsymbol{S_{ji}}$ from bus j to i are

$$S_{ij} = V_i I_{ij}^* \tag{67}$$

$$S_{ji} = V_j I_{ji}^* \tag{68}$$

3.6 Tap Setting for Transformers [5]

Tap changing transformers as well as regulating transformers can be use to control the real and reactive powers. In a tap changing transformer, when the ratio is at the nominal value, the transformer is represented by a series admittance y_t in per unit. With off-nominal ratio, the per unit admittance is different from both sides of the transformer, and the admittance must be modified to include the effect of the off-nominal ratio.

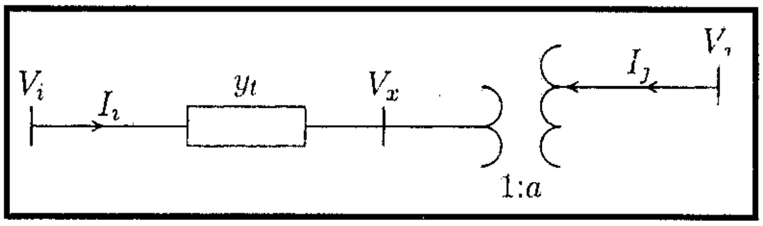


Figure 6

$$\begin{bmatrix} I_i \\ I_j \end{bmatrix} = \begin{bmatrix} y_t & -\dfrac{y_t}{a} \\ -\dfrac{y_t}{a^*} & \dfrac{y_t}{|a|^2} \end{bmatrix} \begin{bmatrix} V_i \\ V_j \end{bmatrix} \tag{69}$$

For the case where a is real, the π model as shown in the figure below represents the admittance matrix above. In the π model, the left side corresponds to the non-tap side and right side corresponds to the tap side of the transformers.

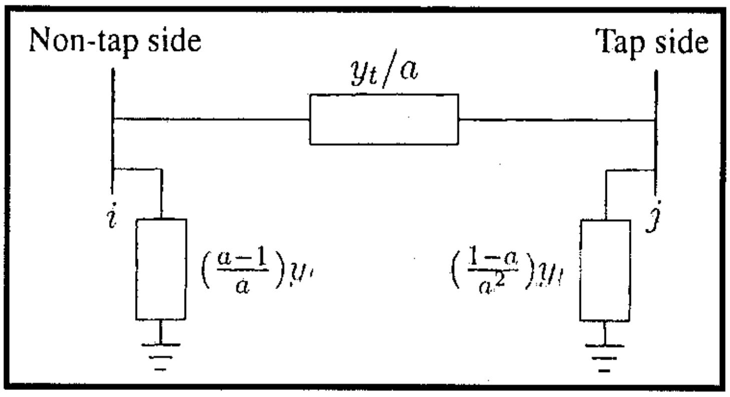

Non-tap side
Tap side
y_t/a
i
j
$\left(\frac{a-1}{a}\right)y_t$
$\left(\frac{1-a}{a^2}\right)y_t$

Figure 7

3.7 MATLAB Programming

A program is written, based on the factored matrices of bus admittance matrix building algorithm as explained before. The programming is written using MATLAB. The flows of the program are illustrated in the flow chart below:

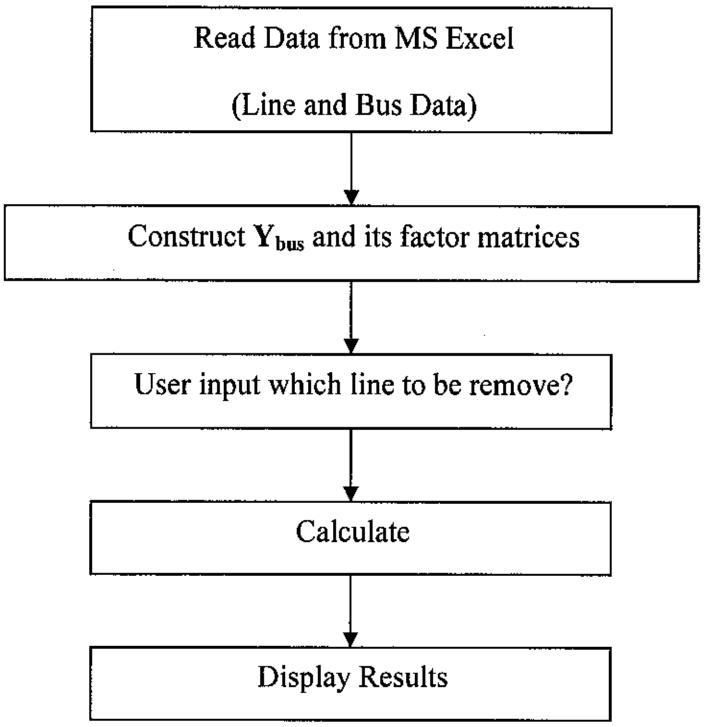

Read Data from MS Excel
(Line and Bus Data)
Construct $\mathbf{Y}_{\text{bus}}$ and its factor matrices
User input which line to be remove?
Calculate
Display Results

Figure 8: The flow chart of the MATLAB program

The line data and initial bus voltages are entered in Microsoft Excel spreadsheet format. Then, the data are imported into MATLAB workspace for further computing.

3.7.1 Program Modules

This program is comprises of several modules, which means it has several functions that working together to achieve the final results. The advantages of writing the program in modular structure are:

1. Each module of function can be run separately, which make it reusable and allow the programmer to work on different modules simultaneously
2. It promotes modularity which can make complex program easier to understand
3. The easiness of doing testing and troubleshooting, when any problem arises

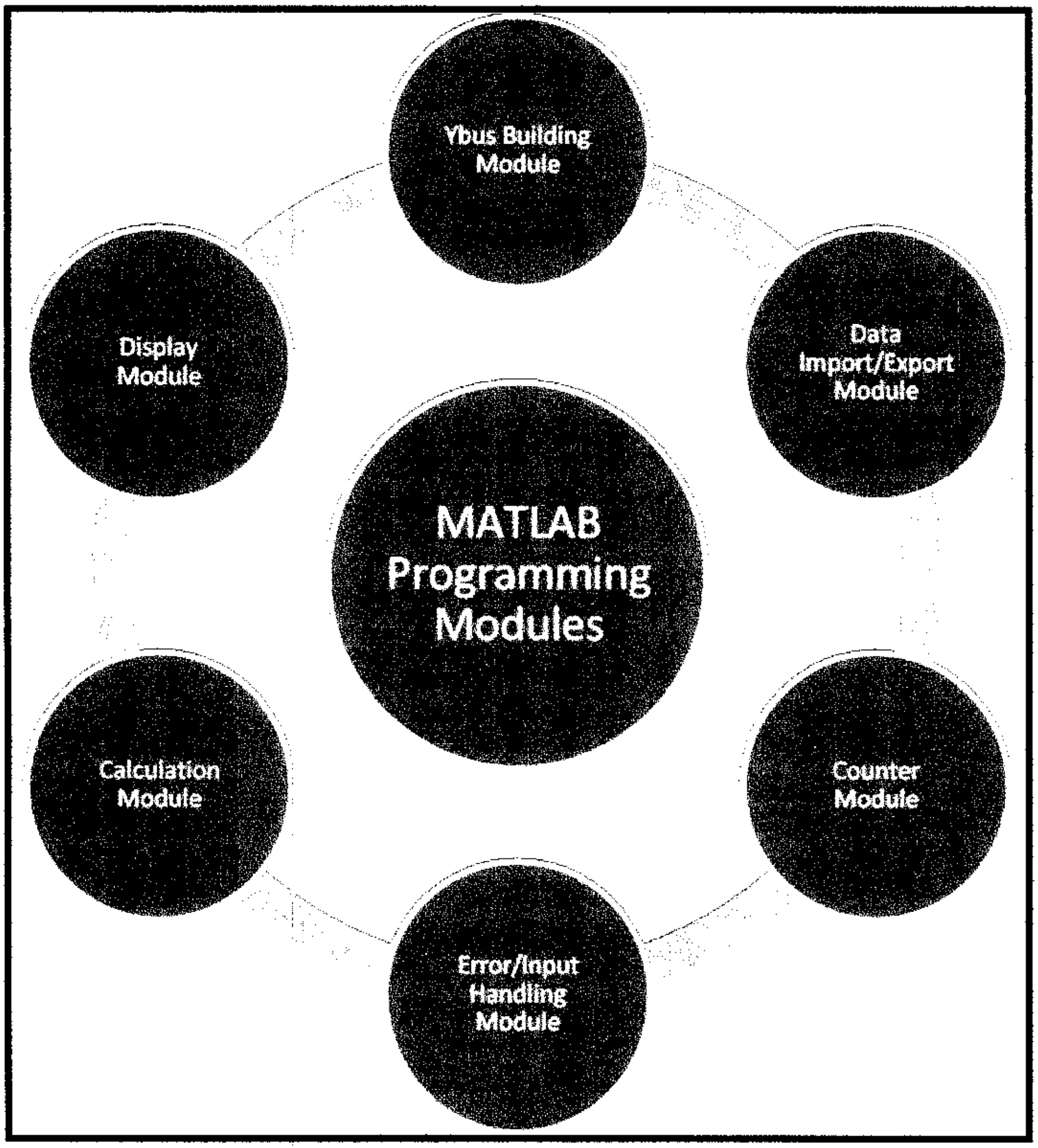
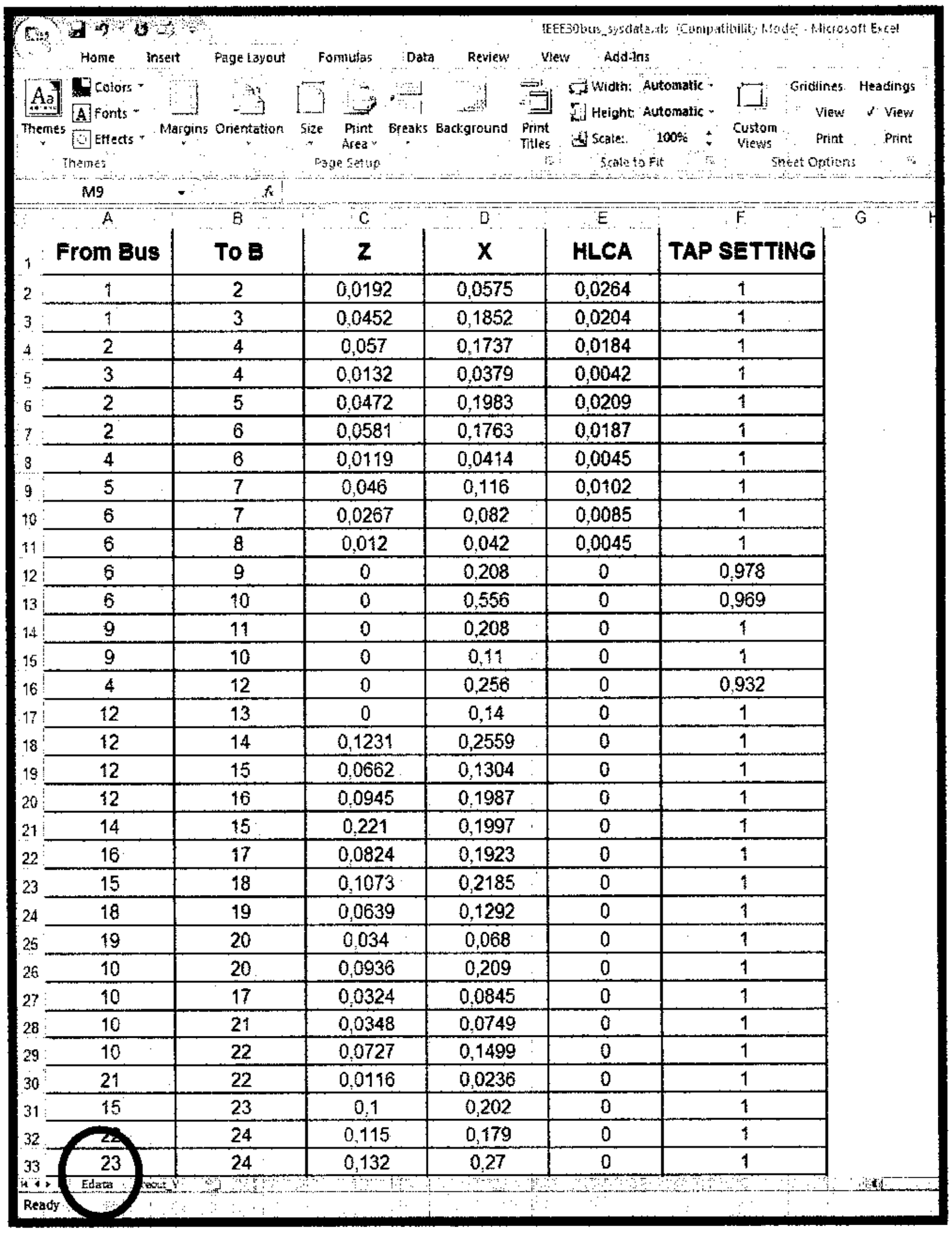

Ybus Building Module
Data Import/Export Module
Counter Module
Error/Input Handling Module
Calculation Module
Display Module
MATLAB Programming Modules

Figure 9: The Program Modules

3.7.2 Creating MS Excel Data File

Before starting the analysis, the line data as well as the pre-outage voltage are to be stored inside the MS Excel. The main purpose for this is, the user will find it much easier to manipulate the data when they are in Excel form, since they can be changed without having to open the MATLAB.

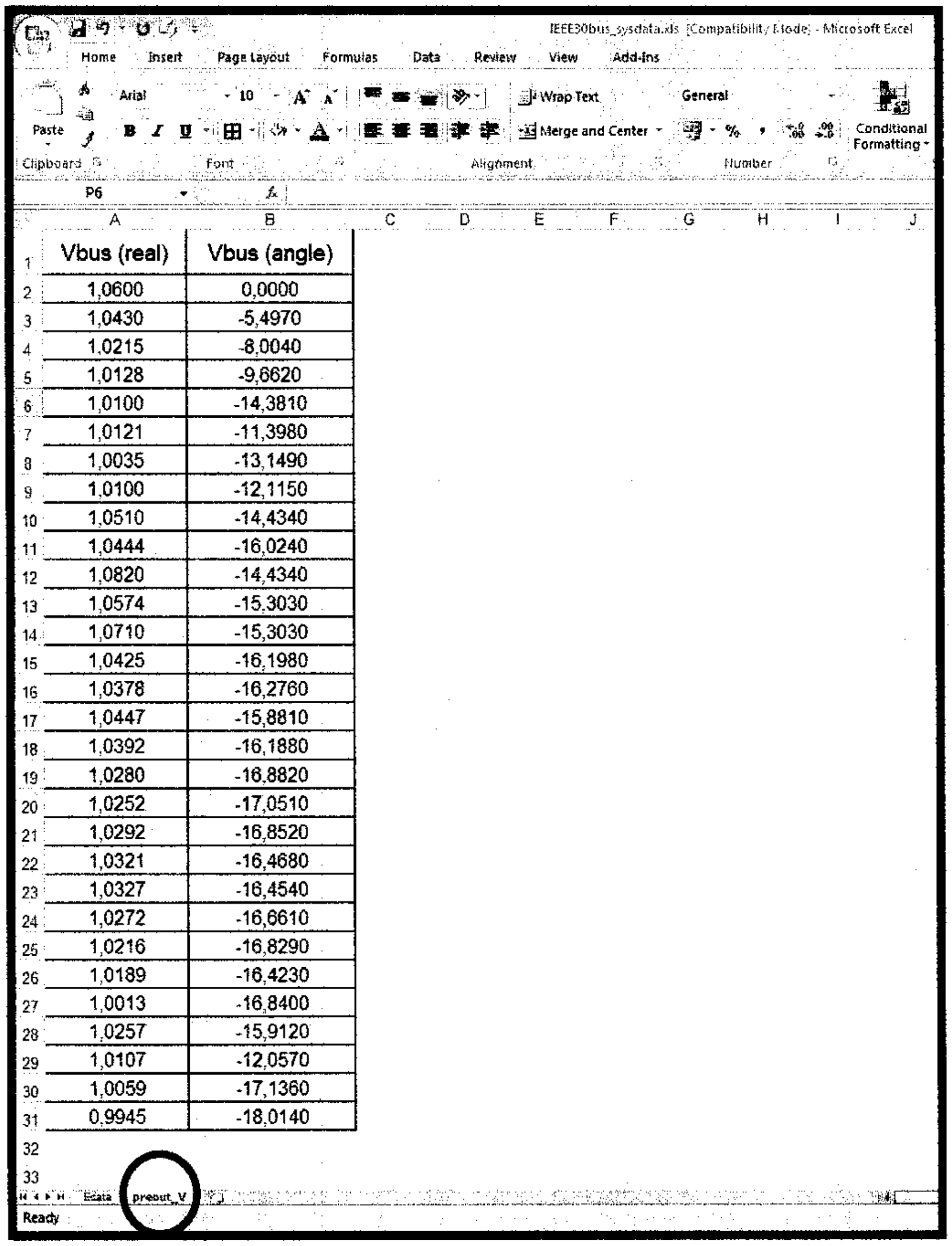


Width: Automatic
Height: Automatic
Scale: 100%

From Bus	To B	Z	X	HLCA	TAP SETTING
1	2	0,0192	0,0575	0,0264	1
1	3	0,0452	0,1852	0,0204	1
2	4	0,057	0,1737	0,0184	1
3	4	0,0132	0,0379	0,0042	1
2	5	0,0472	0,1983	0,0209	1
2	6	0,0581	0,1763	0,0187	1
4	6	0,0119	0,0414	0,0045	1
5	7	0,046	0,116	0,0102	1
6	7	0,0267	0,082	0,0085	1
6	8	0,012	0,042	0,0045	1
6	9	0	0,208	0	0,978
6	10	0	0,556	0	0,969
9	11	0	0,208	0	1
9	10	0	0,11	0	1
4	12	0	0,256	0	0,932
12	13	0	0,14	0	1
12	14	0,1231	0,2559	0	1
12	15	0,0662	0,1304	0	1
12	16	0,0945	0,1987	0	1
14	15	0,221	0,1997	0	1
16	17	0,0824	0,1923	0	1
15	18	0,1073	0,2185	0	1
18	19	0,0639	0,1292	0	1
19	20	0,034	0,068	0	1
10	20	0,0936	0,209	0	1
10	17	0,0324	0,0845	0	1
10	21	0,0348	0,0749	0	1
10	22	0,0727	0,1499	0	1
21	22	0,0116	0,0236	0	1
15	23	0,1	0,202	0	1
22	24	0,115	0,179	0	1
23	24	0,132	0,27	0	1

Figure 10: The IEEE 30 bus line data in MS Excel under "Edata" sheet

Figure 10 illustrate how the line data is written in MS Excel. The pre-outage voltages are to be written, also under the same file name, but different sheet. Please refer to Figure 11 below.



Vbus (real)	Vbus (angle)
1,0600	0,0000
1,0430	-5,4970
1,0215	-8,0040
1,0128	-9,6620
1,0100	-14,3810
1,0121	-11,3980
1,0035	-13,1490
1,0100	-12,1150
1,0510	-14,4340
1,0444	-16,0240
1,0820	-14,4340
1,0574	-15,3030
1,0710	-15,3030
1,0425	-16,1980
1,0378	-16,2760
1,0447	-15,8810
1,0392	-16,1880
1,0280	-16,8820
1,0252	-17,0510
1,0292	-16,8520
1,0321	-16,4680
1,0327	-16,4540
1,0272	-16,6610
1,0216	-16,8290
1,0189	-16,4230
1,0013	-16,8400
1,0257	-15,9120
1,0107	-12,0570
1,0059	-17,1360
0,9945	-18,0140

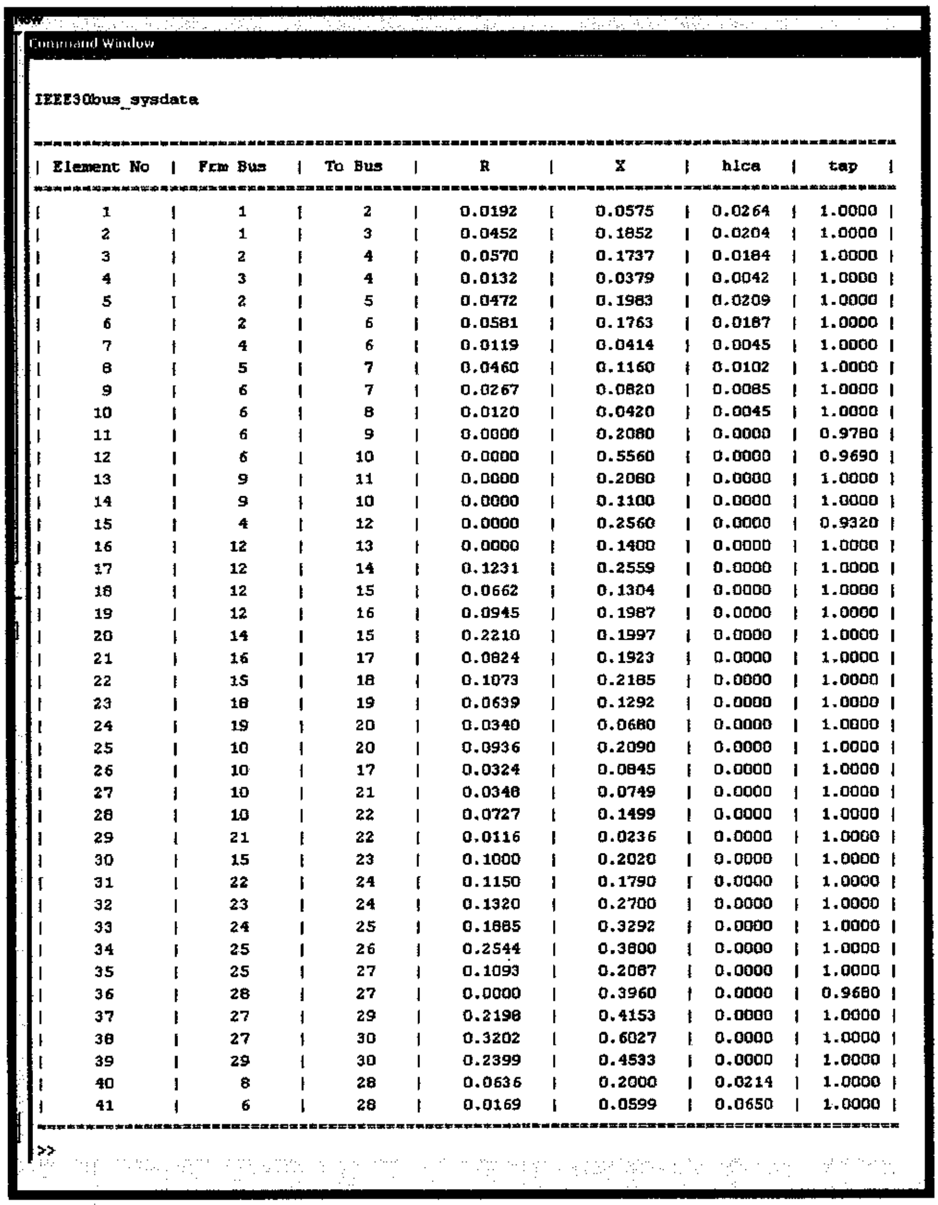
IEEE30bus_sysdata.xls [Compatibility Mode] - Microsoft Excel
Home Insert Page Layout Formulas Data Review View Add-Ins
Paste
Clipboard
Font
Alignment
Number
Wrap Text
Merge and Center
General
Conditional Formatting
P6
Ldata
preout_V
Ready

Figure 11: The IEEE 30 bus pre-outage bus voltages under "preout_V" sheet

3.7.3 Exporting Excel Data into MATLAB

The line data and the bus voltages can be exported into MATLAB by using a single command. In order to make it more flexible, the user can choose the desired file to conduct the analysis. This is achieved by using the command "*uigetfile*" in the program. By using this function, it enables the user to select the appropriate data file. The syntax for this function is appended in the appendix.

Then, by using another function "*xlsread*", MATLAB will read the data in MS Excel and stored them in the workspace in a matrix. Before the matrix can be used for the analysis, it has to be modified slightly, as illustrated in the Figure below.



```
IEEE30bus_sysdata

======================================================================================
| Element No  |  Frm Bus  |  To Bus  |     R     |     X     |   hlca   |   tap   |
======================================================================================
|      1      |     1     |     2    |   0.0192  |   0.0575  |  0.0264  |  1.0000 |
|      2      |     1     |     3    |   0.0452  |   0.1852  |  0.0204  |  1.0000 |
|      3      |     2     |     4    |   0.0570  |   0.1737  |  0.0184  |  1.0000 |
|      4      |     3     |     4    |   0.0132  |   0.0379  |  0.0042  |  1.0000 |
|      5      |     2     |     5    |   0.0472  |   0.1983  |  0.0209  |  1.0000 |
|      6      |     2     |     6    |   0.0581  |   0.1763  |  0.0187  |  1.0000 |
|      7      |     4     |     6    |   0.0119  |   0.0414  |  0.0045  |  1.0000 |
|      8      |     5     |     7    |   0.0460  |   0.1160  |  0.0102  |  1.0000 |
|      9      |     6     |     7    |   0.0267  |   0.0820  |  0.0085  |  1.0000 |
|     10      |     6     |     8    |   0.0120  |   0.0420  |  0.0045  |  1.0000 |
|     11      |     6     |     9    |   0.0000  |   0.2080  |  0.0000  |  0.9780 |
|     12      |     6     |    10    |   0.0000  |   0.5560  |  0.0000  |  0.9690 |
|     13      |     9     |    11    |   0.0000  |   0.2080  |  0.0000  |  1.0000 |
|     14      |     9     |    10    |   0.0000  |   0.1100  |  0.0000  |  1.0000 |
|     15      |     4     |    12    |   0.0000  |   0.2560  |  0.0000  |  0.9320 |
|     16      |    12     |    13    |   0.0000  |   0.1400  |  0.0000  |  1.0000 |
|     17      |    12     |    14    |   0.1231  |   0.2559  |  0.0000  |  1.0000 |
|     18      |    12     |    15    |   0.0662  |   0.1304  |  0.0000  |  1.0000 |
|     19      |    12     |    16    |   0.0945  |   0.1987  |  0.0000  |  1.0000 |
|     20      |    14     |    15    |   0.2210  |   0.1997  |  0.0000  |  1.0000 |
|     21      |    16     |    17    |   0.0824  |   0.1923  |  0.0000  |  1.0000 |
|     22      |    15     |    18    |   0.1073  |   0.2185  |  0.0000  |  1.0000 |
|     23      |    18     |    19    |   0.0639  |   0.1292  |  0.0000  |  1.0000 |
|     24      |    19     |    20    |   0.0340  |   0.0680  |  0.0000  |  1.0000 |
|     25      |    10     |    20    |   0.0936  |   0.2090  |  0.0000  |  1.0000 |
|     26      |    10     |    17    |   0.0324  |   0.0845  |  0.0000  |  1.0000 |
|     27      |    10     |    21    |   0.0348  |   0.0749  |  0.0000  |  1.0000 |
|     28      |    10     |    22    |   0.0727  |   0.1499  |  0.0000  |  1.0000 |
|     29      |    21     |    22    |   0.0116  |   0.0236  |  0.0000  |  1.0000 |
|     30      |    15     |    23    |   0.1000  |   0.2020  |  0.0000  |  1.0000 |
|     31      |    22     |    24    |   0.1150  |   0.1790  |  0.0000  |  1.0000 |
|     32      |    23     |    24    |   0.1320  |   0.2700  |  0.0000  |  1.0000 |
|     33      |    24     |    25    |   0.1885  |   0.3292  |  0.0000  |  1.0000 |
|     34      |    25     |    26    |   0.2544  |   0.3800  |  0.0000  |  1.0000 |
|     35      |    25     |    27    |   0.1093  |   0.2087  |  0.0000  |  1.0000 |
|     36      |    28     |    27    |   0.0000  |   0.3960  |  0.0000  |  0.9680 |
|     37      |    27     |    29    |   0.2198  |   0.4153  |  0.0000  |  1.0000 |
|     38      |    27     |    30    |   0.3202  |   0.6027  |  0.0000  |  1.0000 |
|     39      |    29     |    30    |   0.2399  |   0.4533  |  0.0000  |  1.0000 |
|     40      |     8     |    28    |   0.0636  |   0.2000  |  0.0214  |  1.0000 |
|     41      |     6     |    28    |   0.0169  |   0.0599  |  0.0650  |  1.0000 |
======================================================================================
>>
```

Figure 12: The matrix

By using the $\mathbf{Y_{bus}}$ building algorithm as discussed previously, the $\mathbf{Y_{bus}}$ matrix and its factor matrices are constructed. The construction of the matrix also takes the half line charging admittance (HLCA) and tap setting for transformer into consideration. Next, the user will be prompt to input which line to be removed. After the program has obtained all the necessary data required, it will start calculating for the new bus voltages, line currents as well as power flow in each line in the system. The final results are then displayed for clearer view.

3.7.4 Error Handling

In many cases, it is desirable to take specific actions when different kinds of errors occur. For example, the program may want to prompt the user for more input, display extended error or warning information, or repeat a calculation using default values. The error handling capabilities in MATLAB let the program check for particular error conditions and execute appropriate code depending on the situation.

One of the methods used in the program is warning dialog, or *warndlg* in MATLAB. The example of the warning dialog box is illustrated below.

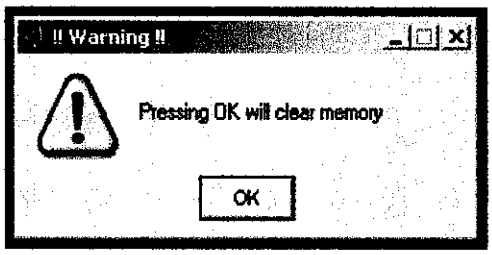


Figure 13: The warning dialog

"*warndlg*" displays a dialog box named 'Warning Dialog' containing the string 'This **is the default warning string.**' The warning dialog box will disappears after you press the "OK" button.

3.7.5 Counter (Stopwatch Timer)

One of the main aspects upon completing this program is to determine what is the program computation time and compare it to the existing approaches (Gauss Seidel, Newton Raphson and Fast Decoupled). Hence, a stopwatch timer has been implemented in this program to calculate how long it will take to complete the calculation.

The counting is started by using command "*tic*" and ended by command "*toc*". The program is appended in between of the two, so that the total time elapsed for the program to complete its computation and come to the final results is obtained. The time will be displayed in the Command Window at the end of the calculation. The syntax for these two functions is appended in the Appendix D.

The speed of the computer computation is depends on the speed of the CPU itself. If a fast computer is used, then computation time should be less. On the other hand, if a slower computer is used, the increase in computation time is expected. However, when we compares the computation time between these four methods either by using a faster or slower computer, we should expect the ratio will be the same for both computers.

3.8 Tools

These tools that are used throughout this project are listed below. They are consist if hardware and software.

1) A PC equipped with at least Pentium III or equivalent processor, 128 MB RAM, Microsoft 2000/ME/XP operating system
2) MATLAB software
3) MS Excel (Microsoft Office)

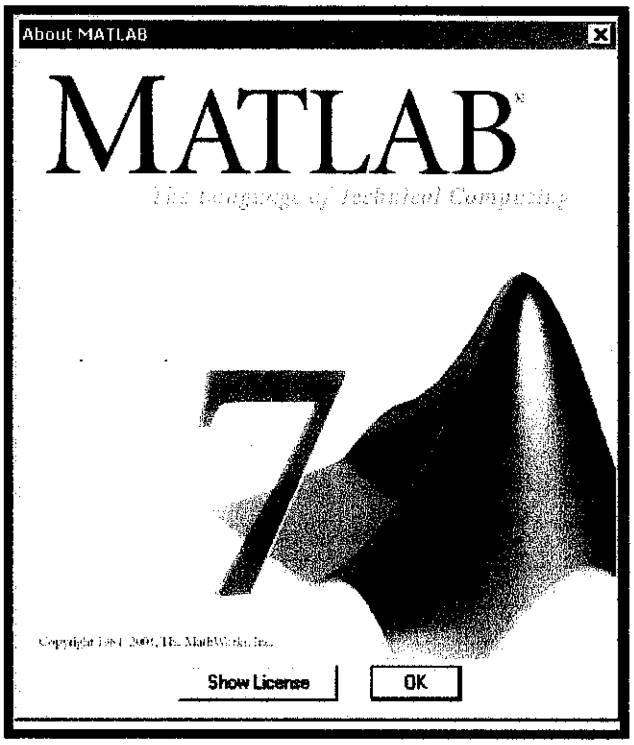


Figure 14: The MATLAB software

CHAPTER 4
RESULTS AND DISCUSSION

The contingency analysis program has been tested on one test system, namely **IEEE 30-bus Reliability Test Systems** to test its practicality and reliability on real system implementation. The IEEE 30-bus test case represents a portion of American Electric Power System (AEP) in the Midwestern US as of December 1961. It is now being made available to the electric utility industry as a standard case for evaluating various analytical methods and computer programs for the solution of power system problems [8]. The one line diagram of the test case, its bus data and line data is illustrated and appended in the Appendix B.

Before the program is run, the power flow analysis is conducted to get the initial value of the bus voltages on each of the 30 buses. These voltages will be stored in the MS Excel data file, and become the pre-outage bus voltages. Then, one element of the line data is removed or in other words, it is open-circuited. Now, instead of conducting power flow analysis again, the new bus voltages are **predicted** using the contingency analysis program.

Then, the computation time for the program is determined and then compared with the existing method available. In order to verify the results of the contingency program, its output is compared with the output of the power flow analysis.

4.1 Results and Analysis

The program has been through series of testing of troubleshooting, to ensure the reliability and the accuracy of the results. During one of the testing of the program using IEEE 30 bus system, it is observed that the results are not so accurate, compared to the power flow analysis program. At that time, the line data that are being used are only inclusive impedance and reactance for each line in the power system. However, it is noticed that they are not sufficient, and therefore the half line charging admittance (HLCA) and the tap setting of the transformers are included in the line data. The effects of these two added elements are discussed earlier in the methodology section.

The Contingency Program will read the line data from MS Excel to construct the bus admittance matrix for the networks. Once the matrix is constructed, it will wait for further instruction. The new line data and the pre-outage bus voltages are appended in the Table 2 and Table 3 below:

Element No	From Bus No	To Bus No	Impedance	Reactance	HLCA	Tap Setting
1	1	2	0.0192	0.0575	0.0264	1
2	1	3	0.0452	0.1852	0.0204	1
3	2	4	0.0570	0.1737	0.0184	1
4	3	4	0.0132	0.0379	0.0042	1
5	2	5	0.0472	0.1983	0.0209	1
6	2	6	0.0581	0.1763	0.0187	1
7	4	6	0.0119	0.0414	0.0045	1
8	5	7	0.0460	0.1160	0.0102	1
9	6	7	0.0267	0.0820	0.0085	1
10	6	8	0.0120	0.0420	0.0045	1
11	6	9	0.0000	0.2080	0.0000	0.978
12	6	10	0.0000	0.5560	0.0000	0.969
13	9	11	0.0000	0.2080	0.0000	1
14	9	10	0.0000	0.1100	0.0000	1

15	4	12	0.0000	0.2560	0.0000	0.932
16	12	13	0.0000	0.1400	0.0000	1
17	12	14	0.1231	0.2559	0.0000	1
18	12	15	0.0662	0.1304	0.0000	1
19	12	16	0.0945	0.1987	0.0000	1
20	14	15	0.2210	0.1997	0.0000	1
21	16	17	0.0824	0.1923	0.0000	1
22	15	18	0.1073	0.2185	0.0000	1
23	18	18	0.0639	0.1292	0.0000	1
24	19	20	0.0340	0.0680	0.0000	1
25	10	20	0.0936	0.2090	0.0000	1
26	10	17	0.0324	0.0845	0.0000	1
27	10	21	0.0348	0.0749	0.0000	1
28	10	22	0.0727	0.1499	0.0000	1
29	21	22	0.0116	0.0236	0.0000	1
30	15	23	0.1000	0.2020	0.0000	1
31	22	24	0.1150	0.1790	0.0000	1
32	23	24	0.01320	0.2700	0.0000	1
33	24	25	0.1885	0.3292	0.0000	1
34	25	26	0.2544	0.3800	0.0000	1
35	25	27	0.1093	0.2087	0.0000	1
36	28	27	0.0000	0.3960	0.0000	0.968

37	27	28	0.2198	0.4153	0.0000	1
38	27	30	0.03202	0.6027	0.0000	1
39	29	30	0.2399	0.4533	0.0000	1
40	8	28	0.0636	0.2000	0.0214	1
41	6	28	0.0169	0.0599	0.0650	1

Table 2: The Line Data for IEEE 30 Bus Systems

Bus No	**Voltage magnitude**	**Angle (°)**
1	1.060	0.00
2	1.043	-5.497
3	1.022	-8.004
4	1.013	-9.662
5	1.010	-14.381
6	1.012	-11.398
7	1.003	-13.149
8	1.010	-12.115
9	1.051	-14.434
10	1.044	-16.024
11	1.082	-14.434
12	1.057	-15.303
13	1.071	-15.303
14	1.042	-16.198
15	1.038	-16.276

16	1.045	-15.881
17	1.039	-16.188
18	1.028	-16.882
19	1.025	-17.051
20	1.029	-16.852
21	1.032	-16.468
22	1.033	-16.454
23	1.027	-16.661
24	1.022	-16.829
25	1.019	-16.423
26	1.001	-16.840
27	1.026	-15.912
28	1.011	-12.057
29	1.006	-17.136
30	0.995	-18.014

Table 3: The initial voltages at each bus after conducting the power flow analysis

4.1.1 Removal of 33rd element (Bus 24 – Bus 25)

When the 33rd element is removed, the new bus voltages are calculated using the Contingency Program and tabulated in the Table 5 below. After the bus voltages are known, the power flows in each line are also calculated and the results can be observed from Table 6.

Then, the results are compared with the results from Power Flow Analysis. As illustrated below, both tables show the comparison between the Contingency Analysis and the Power Flow Analysis results.

At the same time, the total time elapsed for the program to complete its computation is recorded. The program is run for three times and the average time is taken. Similarly, for the same contingency, three other programs for different methods of power flow are run and the average computation time is taken. The results are illustrated both in Table 4 and Figure 15.

Method	Gauss Seidel	Newton Raphson	Fast Decoupled	Contingency Analysis
1	0,266	0,172	0,312	0,047
2	0,25	0,172	0,312	0,062
3	0,25	0,219	0,234	0,047
Average	0,255333333	0,187666667	0,286	0,052

Table 4: Comparison of the computation time (in seconds) between four (4) different methods of power flow, including Contingency Analysis Program in table form

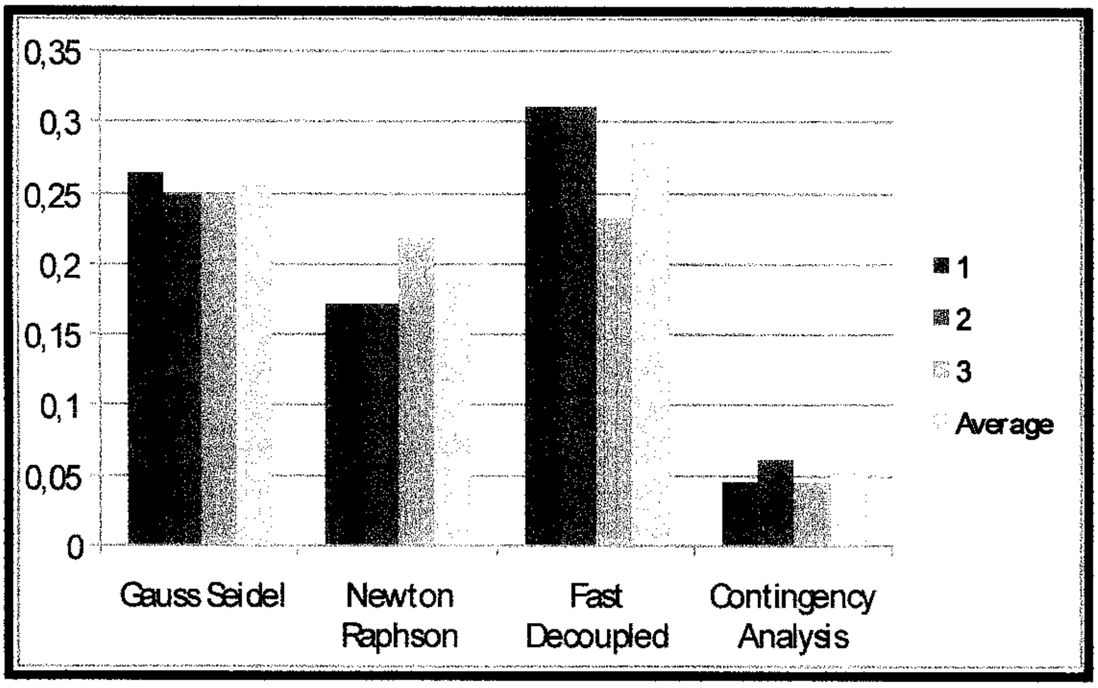


Figure 15: Comparison of the computation time (in seconds) between four (4) different methods of power flow, including Contingency Analysis Program whenever line 24 – 25 was removed. Contingency Analysis program indicates remarkable results in term of its computation time reduction.

From the table and graph above, it is clearly seen that Contingency Analysis gives remarkable reduced computation time compared to the existing methods for Power Flow Analysis. This is really useful and beneficial, when dealing with a much larger system, since it will takes less time to complete its computation.

Table 5: The new voltages at each bus when a line 33 is removed

Bus No	Power Flow Analysis	Contingency Analysis
1	1.0600	1.0596 + 0.0017i
2	1.0333 - 0.0982i	1.0378 - 0.0983i
3	1.0103 - 0.1424i	1.0111 - 0.1406i
4	0.9969 - 0.1701i	0.9980 - 0.1683i
5	0.9781 - 0.2517i	0.9779 - 0.2492i
6	0.9909 - 0.2002i	0.9917 - 0.1984i
7	0.9764 - 0.2287i	0.9768 - 0.2266i
8	0.9874 - 0.2126i	0.9871 - 0.2104i
9	1.0174 - 0.2635i	1.0173 - 0.2601i
10	1.0037 - 0.2905i	1.0033 - 0.2862i
11	1.0474 - 0.2713i	1.0473 - 0.2678i
12	1.0194 - 0.2806i	1.0194 - 0.2771i
13	1.0326 - 0.2842i	1.0325 - 0.2806i

14	1.0007 - 0.2926i	1.0006 - 0.2887i
15	0.9958 - 0.2932i	0.9957 - 0.2887i
16	1.0045 - 0.2877i	1.0043 - 0.2838i
17	0.9977 - 0.2918i	0.9974 - 0.2877i
18	0.9833 - 0.3007i	0.9832 - 0.2964i
19	0.9799 - 0.3028i	0.9796 - 0.2985i
20	0.9848 - 0.3006i	0.9845 - 0.2963i
21	0.9897 - 0.2954i	0.9892 - 0.2904i
22	0.9904 - 0.2955i	0.9899 - 0.2903i
23	0.9839 - 0.2983i	0.9836 - 0.2922i
24	0.9778 - 0.3016i	0.9774 - 0.2931i
25	0.9711 - 0.2769i	0.9773 - 0.2887i
26	0.9520 - 0.2791i	0.9583 - 0.2907i
27	0.9808 - 0.2747i	0.9864 - 0.2812i
28	0.9872 - 0.2105i	0.9880 - 0.2097i
29	0.9556 - 0.2900i	0.9613 - 0.2968i
30	0.9400 - 0.3013i	0.9457 - 0.3073i

Table 6: Comparison between Power Flow Program and Contingency Program during the evaluation of line flows on each line whenever line 24 – 25 was removed

		POWER FLOW ANALYSIS	CONTIGENCY ANALYSIS	Deviation in (%)	
		Using Gauss Seidel (P.U)	in P.U. System	**Real power**	**Reactive power**
1	2	1.7760 – 0.1314i	1.7775 - 0.2186i	0.15	-8.72
1	3	0.8345 + 0.0580i	0.8317 + 0.0526i	-0.28	-0.54
2	4	0.4569 + 0.0062i	0.4569 + 0.0281i	0	2.19
3	4	0.7822 – 0.0257i	0.7803 - 0.0292i	-0.19	-0.35
2	5	0.8306 – 0.0085i	0.8294 + 0.0183i	-0.12	2.68
2	6	0.6177 – 0.0319i	0.6189 - 0.0089i	0.12	2.3
4	6	0.6982 – 0.1843i	0.7011 - 0.1768i	0.29	0.75
5	7	-0.1419 + 0.1107i	-0.1422 + 0.1043i	-0.03	-0.64
6	7	0.3749 – 0.0243i	0.3748 - 0.0183i	-0.01	0.6
6	8	0.2929 – 0.0632i	0.2959 - 0.0364i	0.03	2.66
6	9	0.2822 – 0.0781i	0.2756 - 0.0718i	-0.66	0.63
6	10	0.1613 + 0.0037i	0.1574 + 0.0071i	-0.39	0.34
9	11	0.0000 – 0.1569i	0.0003 - 0.1565i	0.03	0.04
9	10	0.2823 + 0.0622i	0.2753 + 0.0680i	-0.7	0.58
4	12	0.4458 + 0.1401i	0.4398 + 0.1472i	-0.6	0.71
12	13	0.0000 – 0.1032i	0.0002 - 0.1026i	0.02	0.06
12	14	0.0792 + 0.0232i	0.0786 + 0.0244i	-0.06	0.12
12	15	0.1821 + 0.0653i	0.1773 + 0.0703i	-0.48	0.5

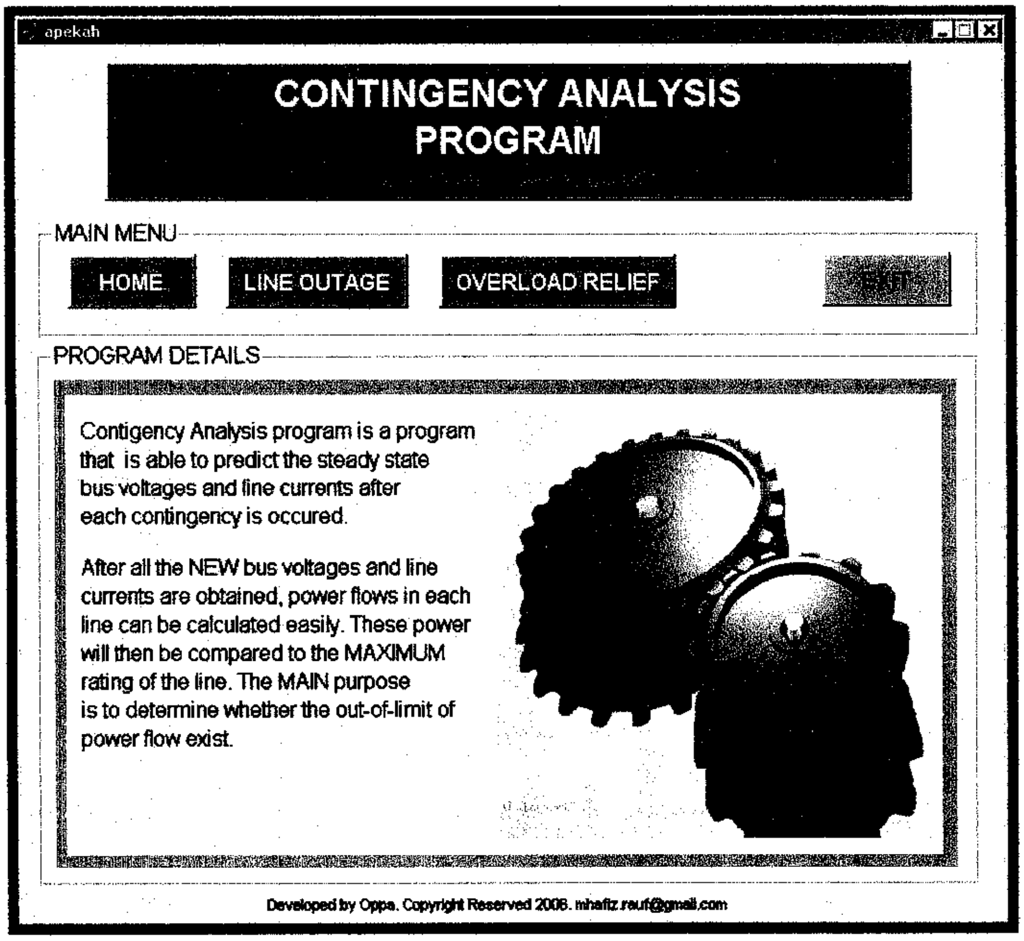
12	16	0.0726 + 0.0323i	0.0719 + 0.0338i	-0.07	0.15
14	15	0.0165 + 0.0056i	0.0154 + 0.0075i	-0.11	0.19
16	17	0.0371 + 0.0131i	0.0363 + 0.0145i	-0.08	0.14
15	18	0.0597 + 0.0173i	0.0601 + 0.0174i	0.04	0.01
18	19	0.0274 + 0.0075i	0.0282 + 0.0084i	0.08	0.09
19	20	-0.0676 – 0.0266i	-0.0670 - 0.0267i	0.06	-0.01
10	20	0.0907 + 0.0358i	0.0902 + 0.0361i	-0.05	0.03
10	17	0.0533 + 0.0455i	0.0535 + 0.0438i	0.02	-0.17
10	21	0.1627 + 0.0922i	0.1560 + 0.0997i	-0.67	0.75
10	22	0.0794 + 0.0407i	0.0750 + 0.0455i	-0.44	0.48
21	22	-0.0131 – 0.0224i	-0.0203 - 0.0160i	-0.72	0.64
15	23	0.0546 + 0.0242i	0.0492 + 0.0303i	-0.54	0.61
22	24	0.0654 + 0.0174i	0.0546 + 0.0292i	-1.08	1.18
23	24	0.0223 + 0.0075i	0.0167 + 0.0133i	-0.56	0.58
24	25	0	0	0	0
25	26	0.0353 + 0.0237i	0.0354 + 0.0236i	0.01	-0.01
25	27	-0.0359 – 0.0234i	-0.0512 - 0.0057i	-1.53	1.77
28	27	0.1686 + 0.0691i	0.1850 + 0.0529i	1.64	-1.62
27	29	0.0618 + 0.0167i	0.0627 + 0.0160i	0.09	-0.07
27	30	0.0708 + 0.0167i	0.0708 + 0.0170i	0	0.03
29	30	0.0369 + 0.0062i	0.0362 + 0.0071i	-0.07	0.09
8	28	-0.0082 – 0.0162i	-0.0051 - 0.0239i	0.31	-0.77
6	28	0.1770 – 0.0891i	0.1910 - 0.0960i	1.4	-0.69

The percentage of deviation from the Power Flow is calculated by finding the different between both methods and multiplies it by 100%. This can be done simply because all the units used have been converted to per unit system (P.U.) and 100MVA is used as the system base.

By looking to the deviation of the results of Contingency Program with respect to the Power Flow Program, it can be seen that it does not exceed 10%, which is a good sign. The maximum deviation is 1.38% for real power and -8.72% for reactive power. On the hand, the program can achieve results as accurate as 0.01% for real power and 0% for reactive power. Hence, at this stage it can be concluded that this program can give the closest results to the Power Flow Program.

4.2 Graphical User Interface

Programming for integrating Graphical User Interface with the MATLAB program is complete but it does not stop there. It will be keep improving from time to time, as the technology changes.



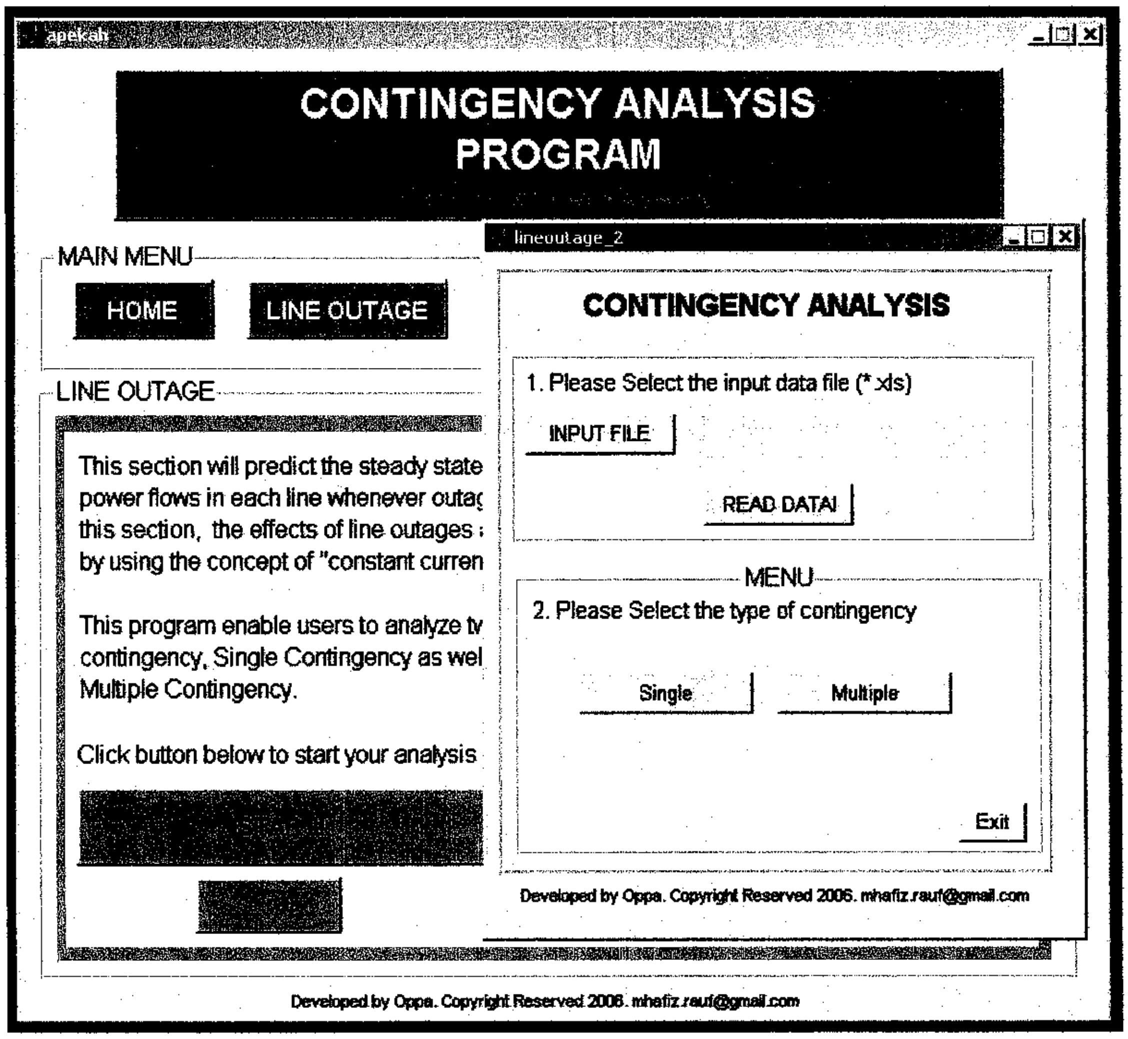
Contigency Analysis program is a program that is able to predict the steady state bus voltages and line currents after each contingency is occured.

After all the NEW bus voltages and line currents are obtained, power flows in each line can be calculated easily. These power will then be compared to the MAXIMUM rating of the line. The MAIN purpose is to determine whether the out-of-limit of power flow exist.

Developed by Oppa. Copyright Reserved 2006. mhafiz.rauf@gmail.com

Figure 16: The main page of the program

The main page of the program consists of three sections, namely **HOME, LINE OUTAGE** and **OVERLOAD RELIEF**. Each has its own functions. To start with the analysis, click the **LINE OUTAGE** button. Then, the **"CONTINGENCY ANALYSIS"** section will be displayed as illustrated in the Figure 17 below.



This section will predict the steady state power flows in each line whenever outag this section, the effects of line outages a by using the concept of "constant curren

This program enable users to analyze tv contingency, Single Contingency as wel Multiple Contingency.

Click button below to start your analysis

1. Please Select the input data file (*.xls)

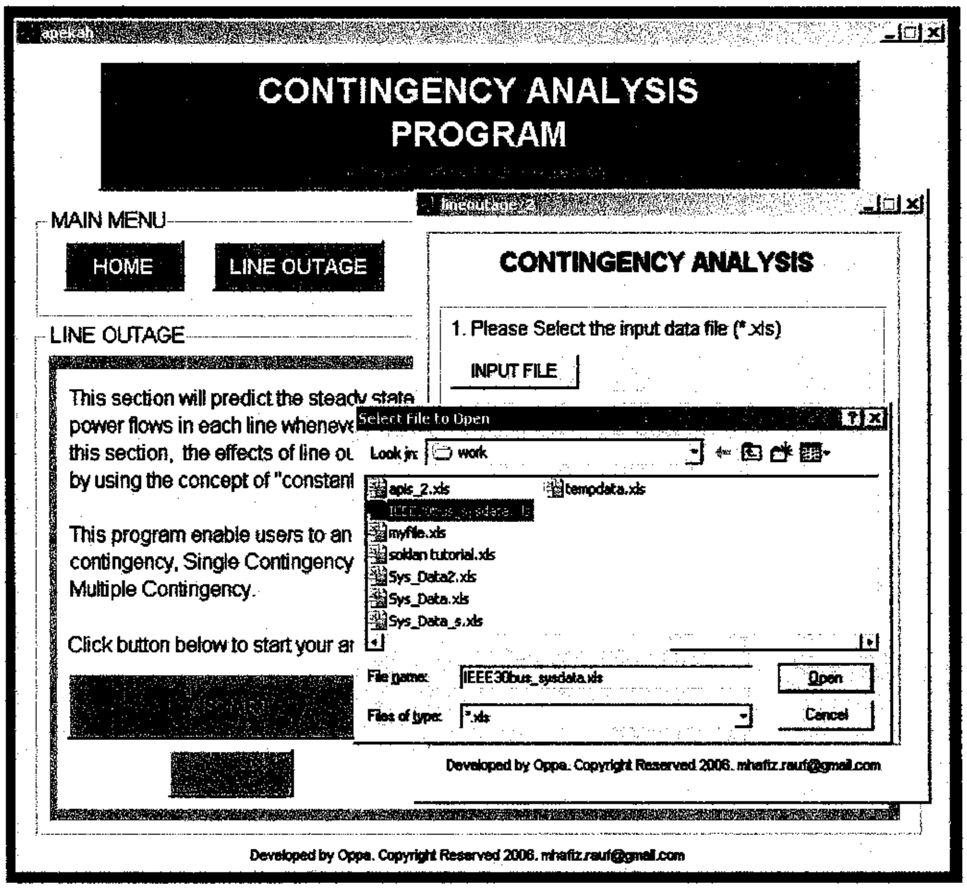
2. Please Select the type of contingency

Developed by Oppa. Copyright Reserved 2006. mhafiz.rauf@gmail.com

Developed by Oppa. Copyright Reserved 2006. mhafiz.rauf@gmail.com

Figure 17: The Single Contingency Section

Next, the data file is selected properly so that the program will read them. The element number to be removed is entered in the box by clicking LOCATION button. After clicking OK and CALCULATE button and the program will start calculating the power flow in each line. Please refer Figure 18 and Figure 19.


apekah

CONTINGENCY ANALYSIS PROGRAM

MAIN MENU

HOME LINE OUTAGE

LINE OUTAGE

This section will predict the steady state power flows in each line whenev this section, the effects of line ou by using the concept of "constant

This program enable users to an contingency, Single Contingency Multiple Contingency.

Click button below to start your ar

lineoutage_2

CONTINGENCY ANALYSIS

1. Please Select the input data file (*.xls)

INPUT FILE

Select File to Open

Look in: work

apis_2.xls
IEEE30bus_sysdata.xls
myfile.xls
soklan tutorial.xls
Sys_Data2.xls
Sys_Data.xls
Sys_Data_s.xls
tempdata.xls

File name: IEEE30bus_sysdata.xls

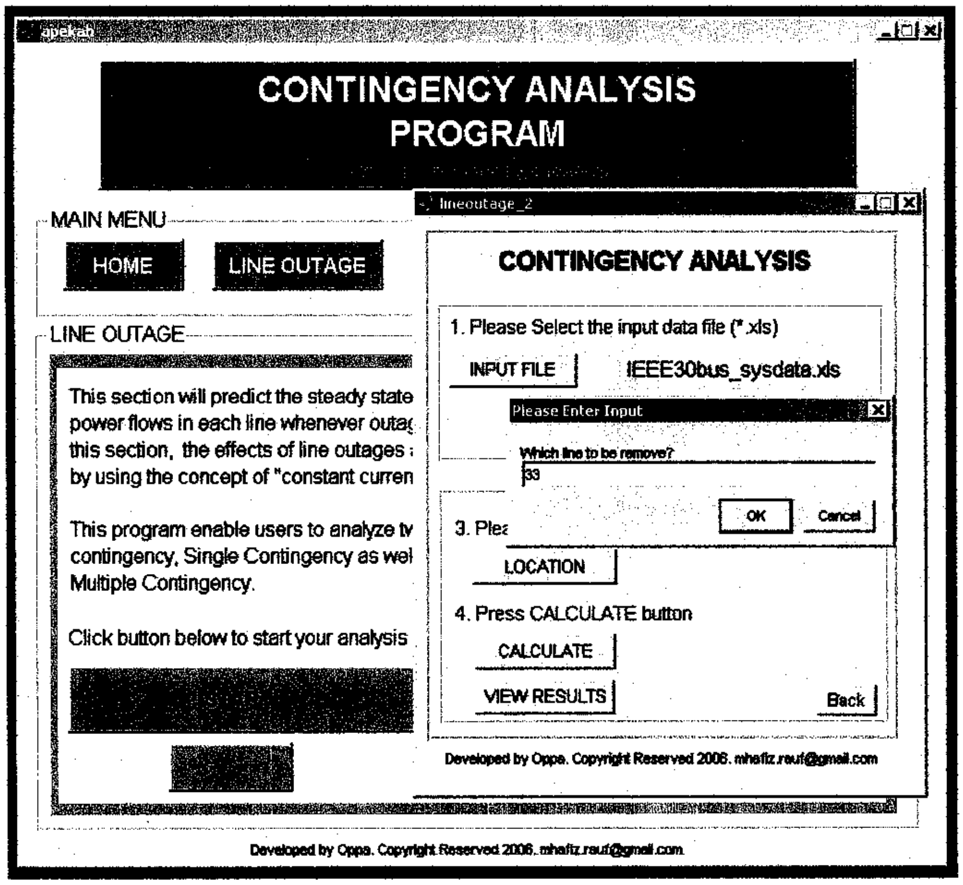
Files of type: *.xls

Open Cancel

Developed by Oppa. Copyright Reserved 2006. mhafiz.rauf@gmail.com

Developed by Oppa. Copyright Reserved 2006. mhafiz.rauf@gmail.com

Figure 18: Select the correct data


apekah

CONTINGENCY ANALYSIS PROGRAM

MAIN MENU

HOME LINE OUTAGE

LINE OUTAGE

This section will predict the steady state power flows in each line whenever outag this section, the effects of line outages by using the concept of "constant curren

This program enable users to analyze tv contingency, Single Contingency as wel Multiple Contingency.

Click button below to start your analysis

lineoutage_2

CONTINGENCY ANALYSIS

1. Please Select the input data file (*.xls)

INPUT FILE IEEE30bus_sysdata.xls

Please Enter Input

Which line to be remove?

33

OK Cancel

3. Plea

LOCATION

4. Press CALCULATE button

CALCULATE

VIEW RESULTS Back

Developed by Oppa. Copyright Reserved 2006. mhafiz.rauf@gmail.com

Developed by Oppa. Copyright Reserved 2006. mhafiz.rauf@gmail.com

Figure 19: Enter the input and click CALCULATE!

In order to see the results, click on the **View Results!** button, and another window will appear. This is actually the one – line diagram of the IEEE 30 bus system. Consequently, by clicking **Update!** button within the window, the power flow on each line in the networks can be seen clearly. The displayed results can be removed just by clicking the **Clear!** button.

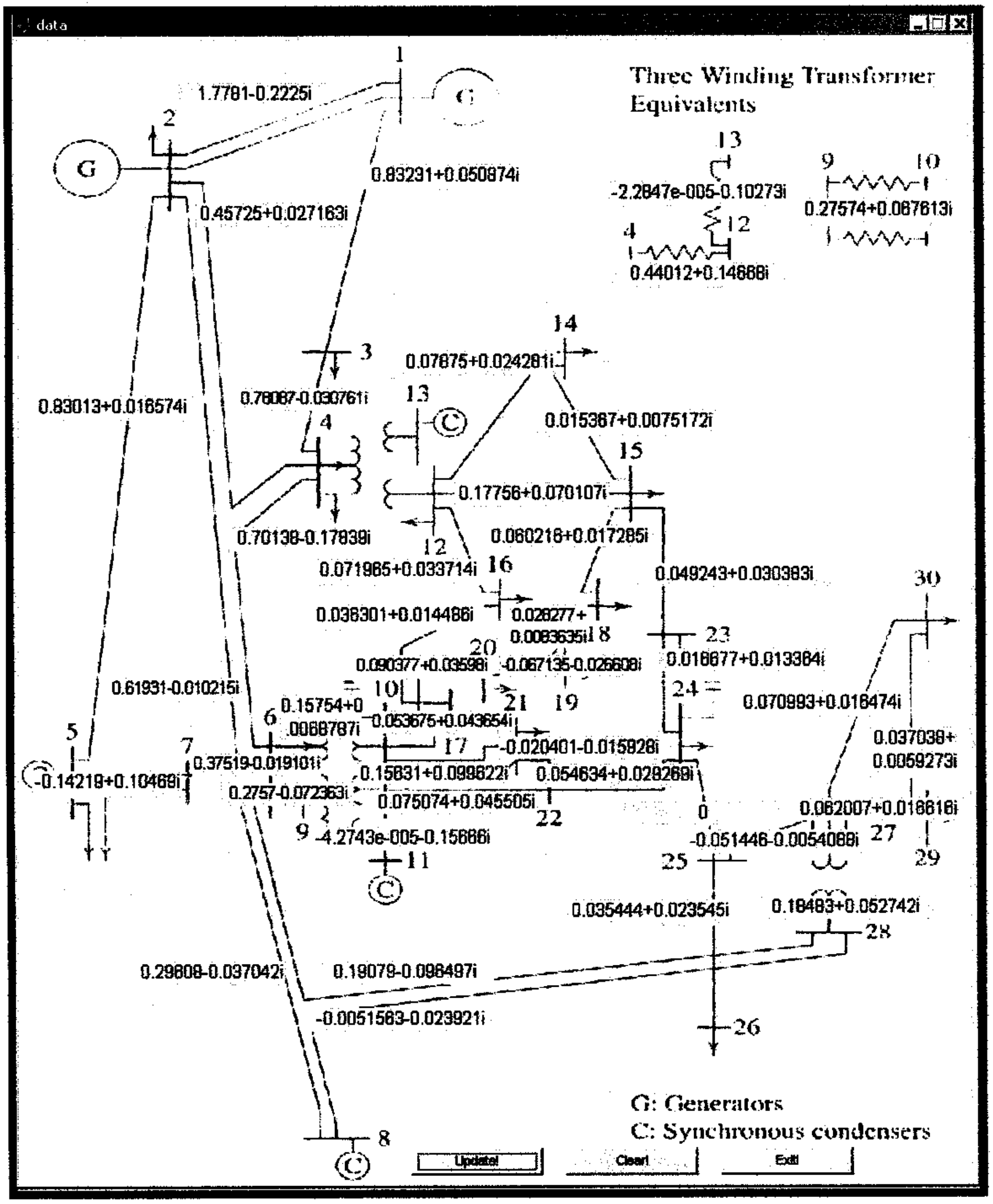


Figure 20: The displayed results

CHAPTER 5
CONCLUSION AND RECOMMENDATION

An alternative way of doing power flow analysis when possible line outage occurs has been presented in this project. Methods of contingency analysis which used the factored Bus Admittance Matrix $\mathbf{Y}_{\mathbf{bus}}$ becomes attractive from computational viewpoint, especially if the loads can be treated as constant current injections into the various buses of the system. Even though the program does not give results as accurate as power flow, it is not so much important because the system planners and operators, who must undertake hundreds of studies in a short time period, are more concerned with knowing if *overload levels* of current and *out-of-limit* voltages and power flow exist than with finding the exact values of those quantities.

This Contingency Analysis program has been tested on an IEEE 30 buses Test System to test its effectiveness, practicality and reliability. From the results and findings, it is can be concluded that this program exhibits remarkable advantage in term of its computation time reduction, as well as its accuracy to the exact solution.

Undoubtedly, it is a big advantage for conducting this analysis on a large system. The computation time will be reduced significantly and therefore this will benefits the electrical utility company itself.

It is therefore important to suggest some improvements and recommendations for the benefit of the project. Since this area is growing, there are opportunities for this approach to be widely applied in the power system engineering. The existing system can be integrated with another system that control the generation of power, in which the effects of both line outage and the power generation control, can be exhibited by the program. If that can be done, this program will be a great success.

REFERENCES

1. **Computer aided contingency analysis [of power systems]**
   Carr, W.;
   Rural Electric Power Conference, 1999
   2-4 May 1999 Page(s):C1/1 - C1/4

2. **Fast automatic contingency analysis and ranking technique for power system security assessment**
   Musirin, I.; Abdul Rahman, T.K.;
   Research and Development, 2003. SCORED 2003. Proceedings. Student Conference on 25-26 Aug. 2003

3. **Comprehensive approach of power system contingency analysis**
   J. Deuse; K. Karoui; A. Bihain; J. Dubois
   IEEE Bologna PowerTech Conference 2003
   June 23-26, Bologna, Italy

4. **Computer Methods in Power System Analysis**, by Stagg, G.W., and EL-Ahmad, A.H, McGraw-Hill Book Company, New York, 1968

5. **Power System Analysis** by Haadi Saadat, McGraw-Hill, Inc. Second Edition 2004.

6. **Power System Analysis** by J.J. Grainger and W.D. Stevenson, McGraw-Hill, Inc. International Edition 1994

7. **IEEE Explore**
   http://ieeexplore.ieee.org/Xplore/guesthome.jsp

8. **IEEE Power System Test Case Archive**
   http://www.ee.washington.edu/research/pstca/pf30/pg_tca30bus.htm

APPENDICES

LIST OF APPENDICES

APPENDIX A – The MATLAB Code

The MAIN program

```
function varargout = Contgency_Analysis(varargin)

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Contgency_Analysis_OpeningFcn, ...
                   'gui_OutputFcn',  @Contgency_Analysis_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before Contgency_Analysis is made visible.

function Contgency_Analysis_OpeningFcn(hObject, eventdata, handles,
varargin)

% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Contgency_Analysis (see VARARGIN)

% Choose default command line output for Contgency_Analysis
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Contgency_Analysis wait for user response (see UIRESUME)
% uiwait(handles.figure1);

%Plot pictures on startup

scout = imread('gears640.jpg');
axes(handles.axes1);
image(scout);
axis off;

yaw_pedals = imread('q0.jpg');
axes(handles.axes3);
image(yaw_pedals);
axis off;

collective_lever = imread('q0.jpg');
axes(handles.axes4);
image(collective_lever);
axis off;

cyclic_stick = imread('q0.jpg');
axes(handles.axes5);
image(cyclic_stick);
axis off;
```

```
aa = imread('q0.jpg');
axes(handles.axes10);
image(aa);
axis off;

utp_logo = imread('q1.jpg');
axes(handles.axes7);
image(utp_logo);
axis off;

% --- Outputs from this function are returned to the command line.
function varargout = Contgency_Analysis_OutputFcn(hObject, eventdata,
handles)

% Get default command line output from handles structure
varargout{1} = handles.output;

%HOME Details
function tab1_ResizeFcn(hObject, eventdata, handles)
% hObject    handle to uipanel1 (see GCBO)

%LINE OUTAGE Details
function tab2_ResizeFcn(hObject, eventdata, handles)
% hObject    handle to uipanel2 (see GCBO)

%OVERLOAD RELIEF Details
function tab3_ResizeFcn(hObject, eventdata, handles)
% hObject    handle to uipanel3 (see GCBO)

% --- Executes on button press in exit.
function exit_Callback(hObject, eventdata, handles)

close Contgency_Analysis;

% --- Executes on button press in HOME.
function scoutinfo_Callback(hObject, eventdata, handles)

handles = guidata(Contgency_Analysis);

set(handles.tab1,'Visible','on');
set(handles.tab2,'Visible','off');
set(handles.tab3,'Visible','off');

% --- Executes on button press in LINE OUTAGE.
function controlinfo_Callback(hObject, eventdata, handles)

handles = guidata(Contgency_Analysis);

set(handles.tab1,'Visible','off');
set(handles.tab2,'Visible','on');
set(handles.tab3,'Visible','off');

% --- Executes on button press OVERLOAD RELIEF.
function program_Callback(hObject, eventdata, handles)

handles = guidata(Contgency_Analysis);

set(handles.tab1,'Visible','off');
set(handles.tab2,'Visible','off');
set(handles.tab3,'Visible','on');
% --- Executes on button press in SINGLE CONTINGENCY(under LINE OUTAGE).
function single_Callback(hObject, eventdata, handles)

lineoutage_2; %call top open a new window to analyze single contingency
```

```
% --- Executes on button press in VIEW DATA.
function data_Callback(hObject, eventdata, handles)

data; % call to open a new window with the one line diagram

% --- Executes on button press in ovrld_outage.
function ovrld_outage_Callback(hObject, eventdata, handles)

over_relief_lineoutage_2;

% --- Executes on button press in data2.
function data2_Callback(hObject, eventdata, handles)

data;
```

The Contingency Analysis program

```
function varargout = lineoutage_2(varargin)

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @lineoutage_2_OpeningFcn, ...
                   'gui_OutputFcn',  @lineoutage_2_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before lineoutage_2 is made visible.
function lineoutage_2_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to lineoutage_2 (see VARARGIN)

% Choose default command line output for lineoutage_2
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes lineoutage_2 wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = lineoutage_2_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes on button press in input_1.
function input_1_Callback(hObject, eventdata, handles)
% hObject    handle to input_1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

filename = uigetfile('*.xls');              %prompt to select the file name
if ~filename
    msgbox('No file is selected. Please select the appropriate file','File
Not Found!','warn');
    set(handles.filename,'String','No file selected')
    return
else
    [tmp,tmp,tmpl] = fileparts(filename);
    if ~strcmp(lower(tmpl),'.xls')
        msgbox('Inappropriate file selected. Please select ONLY THE EXCEL
file','Inappropriate File!','warn');
        set(handles.filename,'String','Wrong file type selected')
        return
    end
end

set(handles.filename,'String',filename);   %display the file name

function filename_Callback(hObject, eventdata, handles)
% hObject    handle to filename (see GCBO)

% --- Executes during object creation, after setting all properties.
function filename_CreateFcn(hObject, eventdata, handles)
% hObject    handle to filename (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

if ispc
    set(hObject,'BackgroundColor','white');
else
    set(hObject,'BackgroundColor',get(0,'defaultUicontrolBackgroundColor'));
end

% --- Executes on button press in read_data.
function read_data_Callback(hObject, eventdata, handles)
% hObject    handle to read_data (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

filename = getappdata(0,'file')
if filename == 1
    msgbox('No file is selected. Please select the appropriate file','File
Not Found!','warn');
    return
end

Edata = xlsread(filename, 'Edata');            %read the line data from the
file
preout_V = xlsread(filename, 'preout_V');   %read the preoutage voltage from
the file

%the wait bar (to make it interesting)
h = waitbar(0,'Please wait...');
for i=1:1000, % computation here %
waitbar(i/100)
end
close(h)
```

```
%get the Ybus matrix,Line Data, number of element, number of bus
[Ybus,Y,Edata_new,Nele] = get_admittance_param(Edata);

%convert the voltage into real and imaginary
preout_V = read_preout_V_edited(preout_V);

fprintf('================================================================
======================\n')
fprintf(['| Element No  |' '  Frm Bus   |' '  To Bus   |' '      R      |' '
X      |' '   hlca    |' '   tap    |\n'])
fprintf('================================================================
=====================')
for k = 1:Nele
    fprintf('    \n'), fprintf('|      %2g', k), fprintf('        |      %2g',
Edata_new(k,1))
    fprintf('      |       %2g', Edata_new(k,2)), fprintf('     |     %1.4f',
real(Edata_new(k,3)))
    fprintf('    |     %1.4f', imag(Edata_new(k,3))), fprintf('    |  %1.4f',
imag(Edata_new(k,4)))
    fprintf('  |  %1.4f', Edata_new(k,5)), fprintf(' |')
end
fprintf('\n================================================================
========================\n')

Edata_x = Edata_new;

%convert all variables to be used to handles
handles.Ybus = Ybus;
handles.Y = Y;
handles.Edata_new = Edata_x;
handles.preout_V = preout_V;
handles.filename = filename;
handles.Nele = Nele;
guidata(hObject,handles)

% --- Executes on button press in sing_cont.
function sing_cont_Callback(hObject, eventdata, handles)

set(handles.menu,'Visible','off');
set(handles.single,'Visible','on');
set(handles.double,'Visible','off');
set(handles.sing_cont,'Visible','off');
set(handles.multi_cont,'Visible','off');
cleardata;

% --- Executes on button press in multi_cont.
function multi_cont_Callback(hObject, eventdata, handles)

set(handles.menu,'Visible','off');
set(handles.single,'Visible','off');
set(handles.double,'Visible','on');
set(handles.sing_cont,'Visible','off');
set(handles.multi_cont,'Visible','off');
cleardata;

% --- Executes on button press in back1.
function back1_Callback(hObject, eventdata, handles)

set(handles.menu,'Visible','on');
set(handles.single,'Visible','off');
set(handles.double,'Visible','off');
set(handles.sing_cont,'Visible','on');
set(handles.multi_cont,'Visible','on');
cleardata;
```

```
% --- Executes on button press in back2.
function back2_Callback(hObject, eventdata, handles)

set(handles.menu,'Visible','on');
set(handles.single,'Visible','off');
set(handles.double,'Visible','off');
set(handles.sing_cont,'Visible','on');
set(handles.multi_cont,'Visible','on');
cleardata;

% --- Executes on button press in location_1.
function location_1_Callback(hObject, eventdata, handles)

Nele = handles.Nele;

out = inputdlg('Which line to be remove?','Please Enter Input');
out = str2num(out{1,1});
setappdata(0,'outs',out);

out = getappdata(0,'outs');

if ((out == 0) | (out > Nele))
    msgbox('The    element    is    out    of    range.    Please    select    the
appropriately','Element Not Found!','warn');
    return
end

% --- Executes on button press in calcul_1.
function calcul_1_Callback(hObject, eventdata, handles)

Nele = handles.Nele;
Ybus = handles.Ybus;
Y = handles.Y;
Edata_x = handles.Edata_new;
preout_V = handles.preout_V;

out = getappdata(0,'outs');

%construction of the Ybus factored matrices
[L,U] = lu(Ybus);
j = diag(U);
L;
U = diag(j);

%reading the line which need to be removed (outage line)
outage = Edata_x(out,:);        %read the entire line
m = outage(1);                  %pick up the 1st bus
n = outage(2);                  %pick up the 2nd bus
Za = outage(3);                 %pick up the impedance of the line

V = find_voltage(m,n,Y,L,U);    %find the (m-n)th column of the Zbus matrix

Z = Za+V(m)-V(n);               %find the total impedance

I = [preout_V(m)-preout_V(n)]/Z;    %calculates the compensating current
deltaV = -V*I;                      %calculates the changes in voltages
Vbus = preout_V + deltaV          %calculates new bus voltages

%converting the voltages to magnitudes and angles
for i = 1:30
    x(i,:) = abs(Vbus(i));          %get the magnitude
    y(i,:) = angle(Vbus(i))*180/pi; %get the angle
    ind (i,:) = i;                  %generate the index (1,2,3,...)
end
Vangle = [ind x y]                 %display results
```

```
%calculate the power flow in each line
nline = length(Edata_x(:,1));         %determine the length
for k = 1:nline
    if k == out                       %power = 0, for the outage line
        S(k) = 0;
        J(k) = 0;
    else
        d = Edata_x(k,1);             %pickup the 1st bus
        f = Edata_x(k,2);             %pickup the 2nd bus
        yt = 1/Edata_x(k,3);          %the admittance of the line
        hlca  =  Edata_x(k,4);               %pickup  the  half  line  charging
admittance value
        tap = Edata_x(k,5);           %pickup the tap setting (a)
        %calculating current from 1st bus tp 2nd bus, including HLCA and TAP
        Idf = [(Vbus(d)*yt/tap^2)-(Vbus(f)*yt/tap)]+[Vbus(d)*hlca];
        S(k,:) = [Vbus(d)*conj(Idf)];   %calculate power flow
        %J(k,:) = [abs(S(k))];          %complex power (MVA)
    end
end
pow = S

%store the answer in each one of the following
setappdata(0,'a1',pow(1));
setappdata(0,'a2',pow(2));
setappdata(0,'a3',pow(3));
setappdata(0,'a4',pow(4));
setappdata(0,'a5',pow(5));
setappdata(0,'a6',pow(6));
setappdata(0,'a7',pow(7));
setappdata(0,'a8',pow(8));
setappdata(0,'a9',pow(9));
setappdata(0,'a10',pow(10));
setappdata(0,'a11',pow(11));
setappdata(0,'a12',pow(12));
setappdata(0,'a13',pow(13));
setappdata(0,'a14',pow(14));
setappdata(0,'a15',pow(15));
setappdata(0,'a16',pow(16));
setappdata(0,'a17',pow(17));
setappdata(0,'a18',pow(18));
setappdata(0,'a19',pow(19));
setappdata(0,'a20',pow(20));
setappdata(0,'a21',pow(21));
setappdata(0,'a22',pow(22));
setappdata(0,'a23',pow(23));
setappdata(0,'a24',pow(24));
setappdata(0,'a25',pow(25));
setappdata(0,'a26',pow(26));
setappdata(0,'a27',pow(27));
setappdata(0,'a28',pow(28));
setappdata(0,'a29',pow(29));
setappdata(0,'a30',pow(30));
setappdata(0,'a31',pow(31));
setappdata(0,'a32',pow(32));
setappdata(0,'a33',pow(33));
setappdata(0,'a34',pow(34));
setappdata(0,'a35',pow(35));
setappdata(0,'a36',pow(36));
setappdata(0,'a37',pow(37));
setappdata(0,'a38',pow(38));
setappdata(0,'a39',pow(39));
setappdata(0,'a40',pow(40));
setappdata(0,'a41',pow(41));
```

```
% --- Executes on button press in viewdata_1.
function viewdata_1_Callback(hObject, eventdata, handles)

data;

% --- Executes on button press in location_2.
function location_2_Callback(hObject, eventdata, handles)

Nele = handles.Nele;

out1 = inputdlg('Which line to be remove?','Please Enter Input');
out1 = str2num(out1{1,1});
out2 = inputdlg('Which line to be remove?','Please Enter Input');
out2 = str2num(out2{1,1});
setappdata(0,'out1',out1);
setappdata(0,'out2',out2);

out1 = getappdata(0,'out1');
out2 = getappdata(0,'out2');

if ((out1 == 0) | (out1 > Nele))|((out2 == 0) | (out2 > Nele))
    msgbox('The   element   is   out   of   range.   Please   select   the
appropriately','Element Not Found!','warn');
    return
end

% --- Executes on button press in calcul_2.
function calcul_2_Callback(hObject, eventdata, handles)

Nele = handles.Nele;
Ybus = handles.Ybus;
Y = handles.Y;
Edata_x = handles.Edata_new;
preout_V = handles.preout_V;

out1 = getappdata(0,'out1');
out2 = getappdata(0,'out2');

[L,U] = lu(Ybus);
j = diag(U);
L;
U = diag(j);

outage1 = Edata_x(out1,:);
m = outage1(1);
n = outage1(2);
z2 = outage1(3);

outage2 = Edata_x(out2,:);
p = outage2(1);
q = outage2(2);
z3 = outage2(3);

V = find_voltage(m,n,Y,L,U);
V1 = find_voltage(m,n,Y,L,U);

Z = [z2 0; 0 z3];
Zadd = [V(m)-V(n) V1(m)-V1(n); V(p)-V(q) V1(p)-V1(q)];;
Z = Z+Zadd;

Vold = [preout_V(m)-preout_V(n); preout_V(p)-preout_V(q)];
I = inv(Z) * Vold;  %% Compensating currents, Ia and Ib
deltaV = -[V V1]*I
Vbusnew = preout_V + deltaV;
X = Vbusnew;
```

```
for i = 1:30;
    x(i,:)=abs(Vbusnew(i));
    y(i,:)=angle(Vbusnew(i))*180/pi;
    ind (i,:) = i;
end
oopa = [ind x y]

%Edata_x(out,:) = [];
nline = length(Edata_x(:,1));
for k = 1:nline
    if (k == out1)|(k == out2)
        S(k) = 0;
        J(k) = 0;
    else
        d = Edata_x(k,1);
        f = Edata_x(k,2);
        yt = 1/Edata_x(k,3);
        hlca = Edata_x(k,4);
        tap = Edata_x(k,5);
        Idf  =  [(X(d)*yt/tap^2)-(X(f)*yt/tap)]+[X(d)*hlca];  %including  HLCA
and TAP setting
        S(k,:) = [X(d)*conj(Idf)];
        J(k,:) = [abs(S(k))];
        number(k,1) = k;
    end
end
power = [number S J];
pow = power(:,2);

setappdata(0,'a1',pow(1));
setappdata(0,'a2',pow(2));
setappdata(0,'a3',pow(3));
setappdata(0,'a4',pow(4));
setappdata(0,'a5',pow(5));
setappdata(0,'a6',pow(6));
setappdata(0,'a7',pow(7));
setappdata(0,'a8',pow(8));
setappdata(0,'a9',pow(9));
setappdata(0,'a10',pow(10));
setappdata(0,'a11',pow(11));
setappdata(0,'a12',pow(12));
setappdata(0,'a13',pow(13));
setappdata(0,'a14',pow(14));
setappdata(0,'a15',pow(15));
setappdata(0,'a16',pow(16));
setappdata(0,'a17',pow(17));
setappdata(0,'a18',pow(18));
setappdata(0,'a19',pow(19));
setappdata(0,'a20',pow(20));
setappdata(0,'a21',pow(21));
setappdata(0,'a22',pow(22));
setappdata(0,'a23',pow(23));
setappdata(0,'a24',pow(24));
setappdata(0,'a25',pow(25));
setappdata(0,'a26',pow(26));
setappdata(0,'a27',pow(27));
setappdata(0,'a28',pow(28));
setappdata(0,'a29',pow(29));
setappdata(0,'a30',pow(30));
setappdata(0,'a31',pow(31));
setappdata(0,'a32',pow(32));
setappdata(0,'a33',pow(33));
setappdata(0,'a34',pow(34));
setappdata(0,'a35',pow(35));
setappdata(0,'a36',pow(36));
setappdata(0,'a37',pow(37));
setappdata(0,'a38',pow(38));
setappdata(0,'a39',pow(39));
setappdata(0,'a40',pow(40));
setappdata(0,'a41',pow(41));

% --- Executes on button press in viewdata_2.
function viewdata_2_Callback(hObject, eventdata, handles)

data;

% --- Executes on button press in exit.
function exit_Callback(hObject, eventdata, handles)
```

Other functions

Calculate the difference of two columns

```
function V = diff_2column(p,q,Y_2,L,U)

for d=1:Y_2
    Ibus(d) = 0;
end
Ibus(p) = 1;
Ibus(q) = -1;
Ibus = transpose(Ibus);

for s=1:Y_2
sum = 0;
    for t = 1:s-1
        sum = L(s,t) * V(t) + sum;
    end
    V(s) = Ibus(s) - sum;
end

for s = 1:Y_2
    V(s) = V(s)/U(s,s);
end

for s = Y_2:-1:1
    d = s+1;
    summ = 0;
    if d > Y_2
        summ = 0;
    else
        for t = d:Y_2
            summ = L(t,s) * V(t) + summ;
        end
    end
    V(s) = V(s) - summ;
end
V = transpose(V);
```

APPENDIX B – IEEE 30 Bus One Line Diagram

6.9. DATA PREPARATION 225

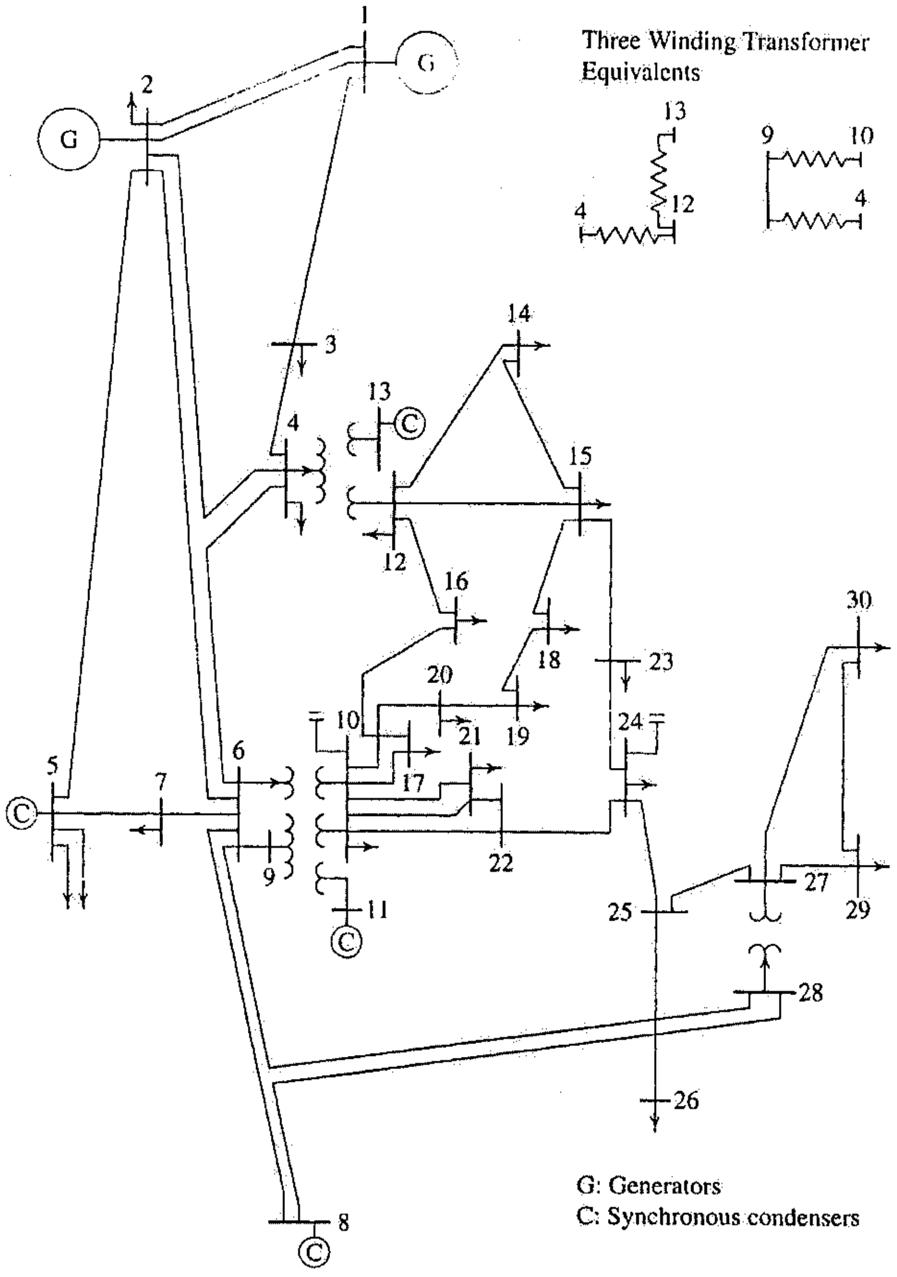
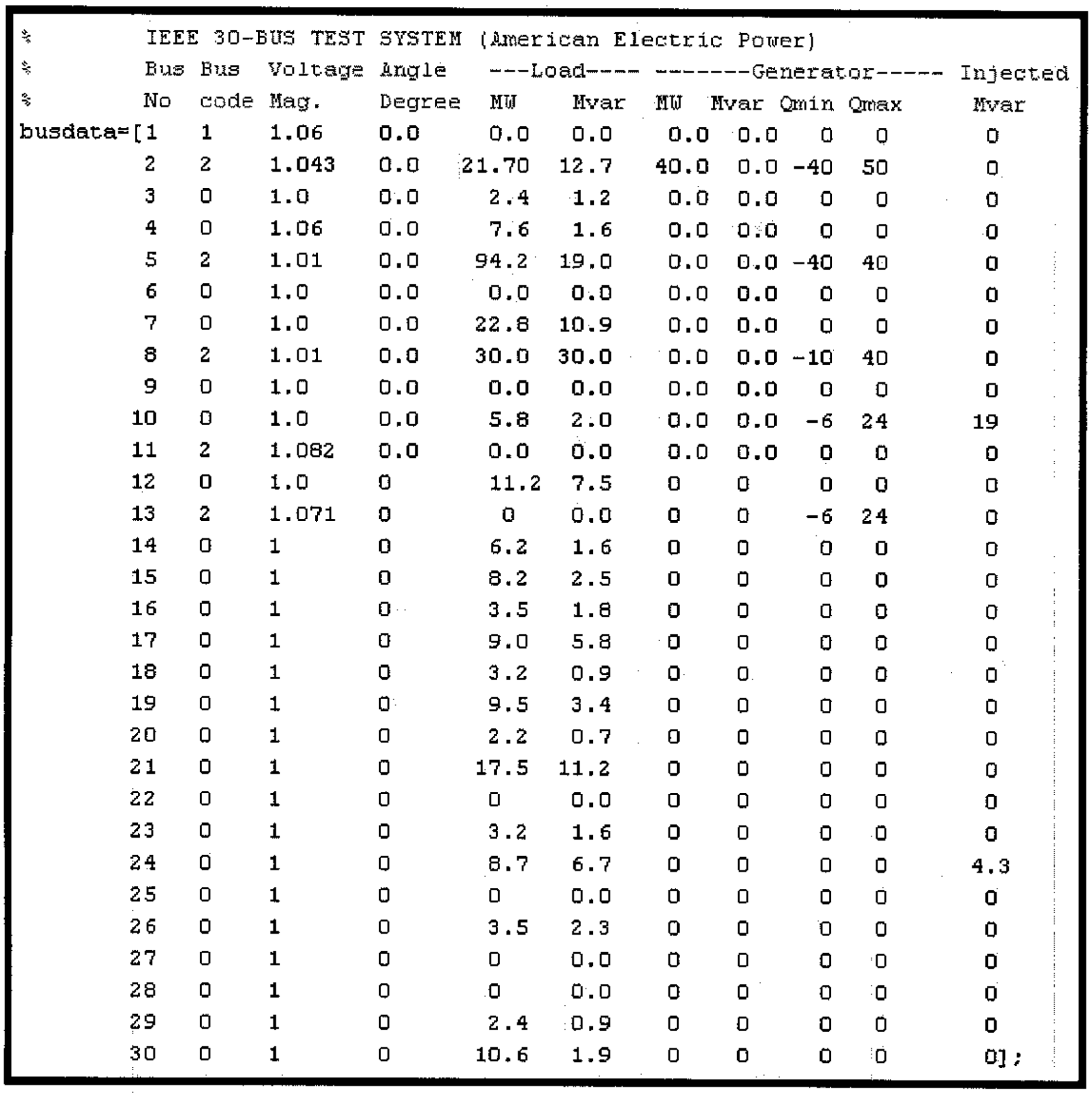


FIGURE 6.16
30-bus IEEE sample system.

APPENDIX C – IEEE 30 Bus Data



```
%        IEEE 30-BUS TEST SYSTEM (American Electric Power)
%        Bus Bus  Voltage Angle    ---Load---- -------Generator----- Injected
%        No  code Mag.    Degree  MW      Mvar  MW   Mvar Qmin Qmax    Mvar
busdata=[1   1    1.06    0.0     0.0     0.0   0.0  0.0  0    0       0
         2   2    1.043   0.0    21.70    12.7  40.0 0.0 -40   50      0
         3   0    1.0     0.0     2.4     1.2   0.0  0.0  0    0       0
         4   0    1.06    0.0     7.6     1.6   0.0  0.0  0    0       0
         5   2    1.01    0.0    94.2     19.0  0.0  0.0 -40   40      0
         6   0    1.0     0.0     0.0     0.0   0.0  0.0  0    0       0
         7   0    1.0     0.0    22.8     10.9  0.0  0.0  0    0       0
         8   2    1.01    0.0    30.0     30.0  0.0  0.0 -10   40      0
         9   0    1.0     0.0     0.0     0.0   0.0  0.0  0    0       0
        10   0    1.0     0.0     5.8     2.0   0.0  0.0 -6    24      19
        11   2    1.082   0.0     0.0     0.0   0.0  0.0  0    0       0
        12   0    1.0     0       11.2    7.5   0    0    0    0       0
        13   2    1.071   0       0       0.0   0    0   -6    24      0
        14   0    1       0       6.2     1.6   0    0    0    0       0
        15   0    1       0       8.2     2.5   0    0    0    0       0
        16   0    1       0       3.5     1.8   0    0    0    0       0
        17   0    1       0       9.0     5.8   0    0    0    0       0
        18   0    1       0       3.2     0.9   0    0    0    0       0
        19   0    1       0       9.5     3.4   0    0    0    0       0
        20   0    1       0       2.2     0.7   0    0    0    0       0
        21   0    1       0      17.5     11.2  0    0    0    0       0
        22   0    1       0      0        0.0   0    0    0    0       0
        23   0    1       0       3.2     1.6   0    0    0    0       0
        24   0    1       0       8.7     6.7   0    0    0    0       4.3
        25   0    1       0      0        0.0   0    0    0    0       0
        26   0    1       0       3.5     2.3   0    0    0    0       0
        27   0    1       0      0        0.0   0    0    0    0       0
        28   0    1       0      0        0.0   0    0    0    0       0
        29   0    1       0       2.4     0.9   0    0    0    0       0
        30   0    1       0      10.6     1.9   0    0    0    0       0];
```

APPENDIX D – MATLAB Functions syntax

uigetfile

Open standard dialog box for retrieving files

Syntax

```
uigetfile
uigetfile('FilterSpec')
uigetfile('FilterSpec','DialogTitle')
uigetfile('FilterSpec','DialogTitle','DefaultName')
uigetfile(...,'Location',[x y])
uigetfile(...,'MultiSelect',selectmode)
[FileName,PathName] = uigetfile(...)
[FileName,PathName,FilterIndex] = uigetfile(...)
```

xlsread

Read Microsoft Excel spreadsheet file (`.xls`)

Syntax

```
num = xlsread(filename)
num = xlsread(filename, -1)
num = xlsread(filename, sheet)
num = xlsread(filename, 'range')
num = xlsread(filename, sheet, 'range')
num = xlsread(filename, sheet, 'range', 'basic')
num = xlsread(filename, ..., functionhandle)
[num, txt]= xlsread(filename, ...)
[num, txt, raw] = xlsread(filename, ...)
[num, txt, raw, X] = xlsread(filename, ..., functionhandle)
xlsread filename sheet range basic
```

warndlg

Open warning dialog box

Syntax

```
warndlg
warndlg('warningstring')
warndlg('warningstring','dlgname')
h = warndlg('warnstring','dlgname',createmode)
h = warndlg(...)
```

tic, toc

Measure performance using stopwatch timer

Synopsis

```
tic
   any statements
toc
t = toc
```