2004

Computer simulation of transient stability analysis of power systems.

N. S. K. Jayasekara Menike
University of Windsor

COMPUTER SIMULATION OF TRANSIENT STABILITY ANALYSIS OF POWER SYSTEMS

by
N. S. K. Jayasekara Menike

A Thesis
submitted to the Faculty of Graduate Studies and Research
through the Department of Electrical and Computer Engineering
in Partial Fulfillment of the Requirements for
The Degree of Master of Applied Science
at
The University of Windsor

Windsor, Ontario, Canada
2004

Your file Votre référence
ISBN: 0-612-92474-2
Our file Notre référence
ISBN: 0-612-92474-2

ABSTRACT

Determination of stability characteristics of a power system is considered as a substantial issue with the increasing complexity of power systems. Power system stability is defined as the behavior of the electrical power system under sudden or sequence of disturbances when it is operating in steady state. It could be a problem of voltage stability, frequency stability or rotor angle stability depending of the type of disturbance. Transient stability which is a sub section of rotor angle stability, is concerned with the condition in which the synchronous machines in the system remain in synchronism or 'in-step' with each other when the system is subjected to severe disturbances. In multi-machine power system, transient stability analysis is an indispensable tool in the areas of planning, design, operation and research.

The nonlinear nature under disturbance is not linearised for the purpose of analyzing transient stability. Therefore, the theories and methodologies for transient stability analysis based on approximations and assumptions. This study is to prepare an educational software package to understand the transient stability behavior while understanding the theories and methodologies behind it.

The software package is developed in MATLAB environment using its facility to develop user friendly graphical user interfaces and programming to handle graphics structure. Several GUIs are developed for entering and saving system data needed for the analysis and to execute the pre-transient stability analysis and transient stability analysis. The software is developed to analyze the steady state behavior of the system using Newton-Raphson power flow method and transient stability simulation using numerical integration. Theories used to develop the mathematical model of the system and the methodologies used for the analysis are included in the package in .pdf format for reference.

In preparing the package, the classical representation of synchronous machine is used while keeping the provision to develop the program to use the dynamic representation of

machine with its controls. In power flow analysis, loads are considered as constant power loads and in transient analysis, they are considered as constant impedances. In the analysis, three phase symmetrical fault is considered as the type of disturbance. Provisions are made in the transient stability analysis GUI to change the position of disturbance, machine damping, machine inertia, fault clearing time to analyze their effects on transient stability response of the system. The critical clearing time for each case can be analyzed.

The software developed is used to analyze the transient behavior of the 9-bus system and 15-bus system while analyzing the effects of fault position, machine damping, machine inertia, clearing time, transient response of the system when complexity of the system increases.

ACKNOWLEDGEMENT

I would like to extend my sincere gratitude to my supervisor Dr. G. R. G. Raju, who has directed my studies and research work and given me an opportunity to select the thesis project in power system area. His fruitful advices helped me to prepare a successful software package. I would also like to thank Dr. X. Chen and Dr. M. K. S. Madugula for their constructive comments and Dr. M. A. Sid-Ahmed for chairing my thesis defense. Further, I would like to thank Dr. N. Kar who helped me to understand the subject matter beyond my knowledge through his course "Advanced Power Systems".

I would also like to thank Mr. Frank Cicchello, Mr. Don Tersigni, Ms. Shelby Merchand and Ms. Dorothy Gabriel for their administrative help.

Finally, I would like to thank my husband and my daughter for their support and understanding.

TABLE OF CONTENTS

LIST OF TABLES

Table

LIST OF FIGURES

Figure

CHAPTER 1
INTRODUCTION

1.1 Power System Stability

Present day power systems are vast and heavily interconnected with hundreds of machines that can interact through the medium of high voltage transmission network [1]. Such large systems are subjected to a wide range of operating conditions depending on load level and equipment availability and may undergo a limitless variety of disturbances. The inherent nonlinear nature of power systems limits the ability to extrapolate from one situation to another with confidence. Thus, the determination of stability characteristics in any large power system is a substantial issue indeed [2].

Power system stability is concerned with the sudden or sequence of changes in the system when it is operating in steady state operating condition. The system is in steady state operating condition if the changes are within narrow bounds and physical quantities which describe the operating condition can be considered as constants for the purpose of analysis [1]. The definition of power system stability according to IEEE task force is

> "Power system stability is the ability of an electric power system, for a given initial operating condition, to regain a state of operating equilibrium after being subjected to a physical disturbance, with system variables bounded so that practically the entire system remains intact [3] [4]."

To better understand the phenomenon of power system stability a further classification [3][4] is according to the

1. physical nature of the problem, as rotor angle stability, frequency stability and voltage stability
2. size of the disturbance, small or large disturbance
3. time span of the instability, short term, mid term or long term

The rotor angle stability is concerned with electromechanical dynamic behavior of synchronous machines of the system [1]. It is the ability to maintain equilibrium between electromagnetic torque and mechanical torque of each machine of the system. For the purpose of analysis, angular stability is sub-grouped as steady state stability and transient stability according to the nature of the disturbance [4].

1.2 Transient Stability

Transient stability is concerned with the ability of maintaining synchronism when the system is subjected to severe disturbances where the equations that describe the machine dynamics cannot be linearised for the purpose of analysis. Sudden change of system configuration - loss of generation, loss of transmission facilities, loss of load, loss of excitation etc. and system faults are the causes of transient instability [1][4]. In power system simulation, it is believed that analysis of transient stability is an indispensable tool in following areas [5].

1. In power system planning - to enable decisions to be made on future transmission and generation requirements
2. In power system operation – to determine operating limits of various system contingencies
3. In research – to simulate past disturbance that occurred in the system for prevention

Transient stability analysis is highly nonlinear and several methods are normally used for the analysis. They can be categorized into three bases [6].

1. Digital simulation (numerical integration method, direct or Lyapunov method, probabilistic method)
2. Heuristic (expert system; artificial intelligence approach) methods
3. Training (pattern recognition and artificial neural network)

Numerical integration method is the traditional method of transient stability simulation and still it is widely used.

1.3 Model Study of Transient Stability

In 1998, IEEE Power Engineering Education Committee sponsored panel reviewed Electric Power Engineering Education Worldwide. According to them, large number of North American universities eliminated power engineering from their curricula due to budget restrictions and the reduction of research support and lack of student's interest. The panel emphasized the use of different educational strategy to change the image of power engineering studies. It was identified that development of new multi media supported computer based courses can increase student's interest and knowledge. This idea tends to develop software tools for educational purposes [7].

The study here is focused on the preparation of educational simulation package to analyze the transient stability of multi machine power system. The theories and methodologies in transient stability analysis are based on approximations and assumptions and rather difficult to understand. This study is an effort to prepare a software tool to understand the transient stability behavior of power systems while understanding the theory and analyzing tools behind it. The study is restricted for the systems with 15 buses or less.

The software tool is solely developed in MATLAB environment. MATALB is a self-contained software environment used for general-purpose numeric computation. Its use in matrix manipulation is rather easy. Further, its Graphical User Interface construction tools provide the facility to develop user friendly method of analyzing the subject matter [8][9].

The presentation of this thesis is developed according to the flow of simulation package. Chapter 2 describes the theories and methodologies which are frequently used in stability studies. Chapter 3 is for Graphical User Interfaces used to link the text files about utilizing the program and theory, and other GUI s which are used for entering data needed for transient stability analysis and to get the results of the analysis. Program flow is written in Chapter 4, in pseudo format. The corresponding MATLAB programs are given in appendix. Chapter 5 gives a simulation demonstration of 9-bus system and 15-bus system and conclusions are finally drawn in Chapter 6.

CHAPTER 2

THEORIES & METHODOLOGIES

The purpose of this chapter is to give an overview of the theories and methodologies required to analyze the transient stability of multi-machine power systems. Theory behind representation of power system modeling, network representation of multi machine power system, steady state power flow and methodologies for transient stability evaluation are described.

2.1 Representation of Power System in Common Reference

In power system analysis, the common practice is to use normalized system variables represented in common frame of reference. Here, all the variables are represented in per unit instead of using them in actual units and measurements. This offers computational simplicity in power system analysis and facilitate understanding of system characteristics as the system is in common base. The per unit value of any quantity is defined as [1][10] [11]

Quantity in per unit = Actual quantity/Base value of quantity

Base (reference) values for 3-phase power system is given by

Apparent Power (3 phase)	S_{Base}	
Voltage (line to line)	V_{Base}	
Current (line)	$I_{Base} = \frac{S_{Base}}{\sqrt{3} \cdot V_{Base}}$	(2.1)
Impedance	$Z_{Base} = \frac{V_{Base}}{\sqrt{3} \cdot I_{Base}} = \frac{V^2{}_{Base}}{S_{Base}}$	(2.2)

According to above base, Per unit Quantities are defined as

$$S(p.u) = \frac{S_{Actual}}{S_{Base}} \tag{2.3}$$

$$V(p.u) = \frac{V_{Actual}}{V_{Base}} \tag{2.4}$$

$$I(p.u) = \frac{I_{Actual}}{I_{Base}} \tag{2.5}$$

$$Z(p.u) = \frac{Z_{Actual}}{Z_{Base}} \tag{2.6}$$

If the per unit impedance of a component of a system is expressed in different base other than the selected base of the system in which the component is located, the per unit value according to new base of reference

$$Z_{New}(p.u) = Z_{Given}(p.u).\left[\frac{V_{Base}(Given)}{V_{Base}(New)}\right]^2.\left[\frac{S_{Base}(New)}{S_{Base}(Given)}\right] \tag{2.7}$$

2.2 Modeling of Power System Components

The first step in power system analysis is to develop an appropriate mathematical model for the power system network considering the terminal behavior of power system components[12]. Here, the network representation of different components are discussed.

2.2.1 Synchronous Generator

The simplest model of a synchronous generator for the purpose of power system analysis is represented by constant voltage source with its synchronous reactance without considering the saliency of the generator and the changes of flux linkage [1][12].

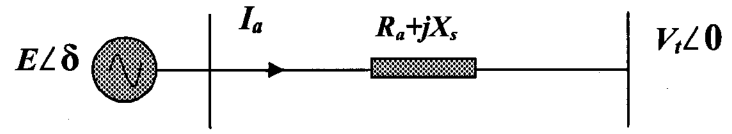


Figure 2.1 : The equivalent diagram for the synchronous machine

Generated voltage at the air gap in this case

$$E\angle\delta = V_t + I_a R_a + jI_a X_s \tag{2.8}$$

where

$E\angle\delta$	-	Generated voltage
$V_t\angle\theta$	-	Terminal voltage of the generator
I_a	-	Armature current
R_a	-	Armature resistance
X_S	-	Synchronous reactance

and the phasor diagram

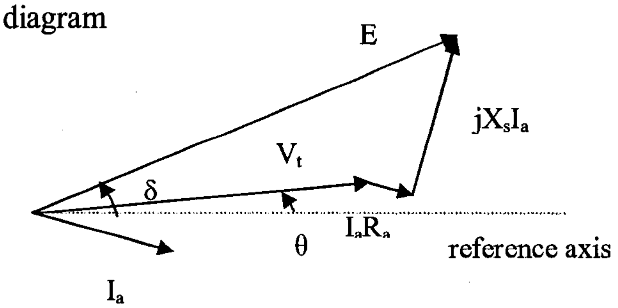


Figure 2.2: Phasor diagram for the simplified model

When the saliency of the generator is considered, the air gap voltage is given by

$$E\angle\delta = V_t + I_a R_a + jI_d X_d + jI_q X_q \tag{2.9}$$

In the steady state, where the flux linkage is considered constant for the purpose of analysis, the phasor diagram for the above representation is given by

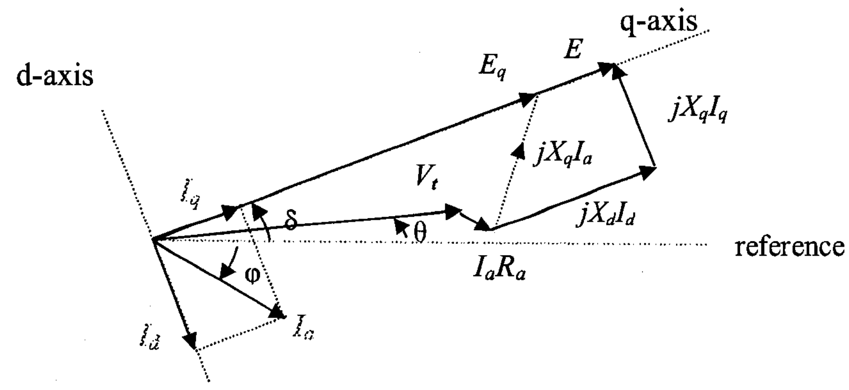


Figure 2.3 : Phasor diagram for overexcited generator with lagging current

E	- Internal voltage of the generator
δ	- Load angle
V_t	- Terminal voltage of generator
θ	- Phase angle of terminal voltage with reference
I_a	- Armature current
φ	- Phase of angle armature current with reference
I_d	- Direct axis component of armature current
I_q	- Quadrature axis component of armature current
X_d	- Direct axis component of synchronous reactance
X_q	- Quadrature axis component of synchronous reactance

Although the above representations are adequate for the purpose of power flow analysis, a detailed analysis of synchronous generator is needed for the stability studies of electrical power systems.

2.2.2 Transformer

The fundamental equivalent diagram of the transformer is represented by ideal transformer with primary and secondary winding impedances (Z_1 and Z_2) in series with the windings and shunt resistance (R_m) and magnetizing reactance (X_m) to denote the eddy current and magnetizing losses of the iron core. **n** denotes the complex turns ratio [1][11].

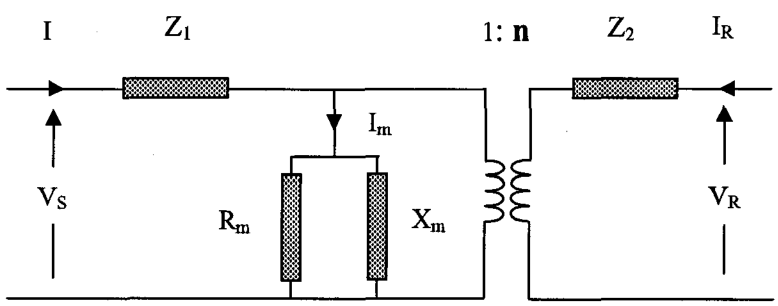


Figure 2.4 : Equivalent diagram of the transformer

$Z_1 = R_1 + jX_1$ and $Z_2 = R_2 + jX_2$

where

R_1, R_2 – winding resistance of primary and secondary windings respectively

X_1, X_2 - winding leakage reactance of primary and secondary windings respectively

The standard equivalent diagram is modeled by eliminating magnetizing branch by considering R_m and X_m are very high and $I_m << I_s$, and by representing Z_1 and Z_2 by their equivalent impedances referred to either primary or secondary side for the purpose of analysis[1][11]. The per unit representation of it is widely used in power flow and stability analysis.

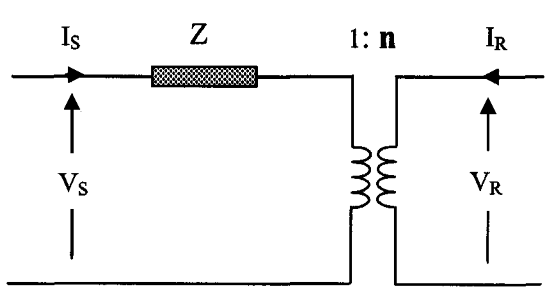


Figure 2.5: Standard equivalent diagram of the transformer
(impedance referred to primary side)

if the impedance is referred to primary $Z = Z_1 + Z_2/|n|^2$

if the impedance is referred to secondary $Z = Z_2 + |n|^2.Z$

When the equivalent impedance is referred to primary and denoted by its corresponding admittance,

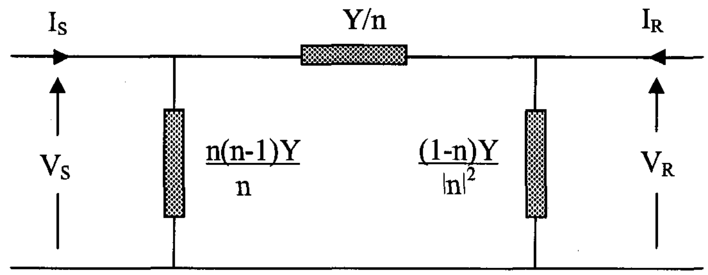


Figure 2.6 : Equivalent Π representation of transformer (impedance referred to primary side)

the admittance form of equation is given by

$$\begin{bmatrix} Y & -\dfrac{Y}{n} \\ -\dfrac{Y}{n^*} & \dfrac{Y}{|n|^2} \end{bmatrix} \begin{bmatrix} V_S \\ V_R \end{bmatrix} = \begin{bmatrix} I_S \\ I_R \end{bmatrix} \qquad (2.10)$$

When the equivalent impedance is referred to secondary and denoted by its corresponding admittance,

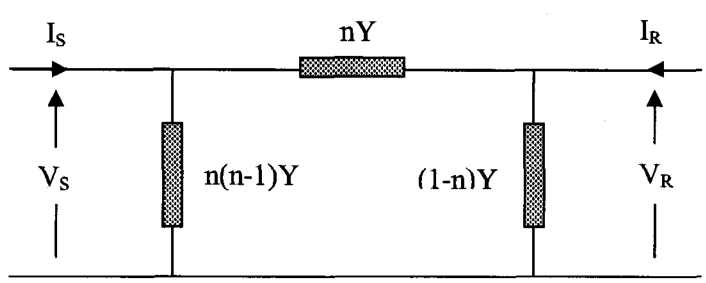


Figure 2.7 : Equivalent Π representation of transformer (impedance referred to secondary side)

the admittance form of equation is given by

$$\begin{bmatrix} |n|^2 Y & -n^* Y \\ -nY & Y \end{bmatrix} \begin{bmatrix} V_S \\ V_R \end{bmatrix} = \begin{bmatrix} I_S \\ I_R \end{bmatrix} \tag{2.11}$$

2.2.3 Transmission Line

In network representation, a transmission line is characterized by four parameters, series resistance (R) due to the conductor resistivity, series inductance (L) due to magnetic field surrounding the conductors, shunt conductance (G) due to leakage currents between the phases and the ground and shunt capacitance (C) due to electric field between conductors [1][11].

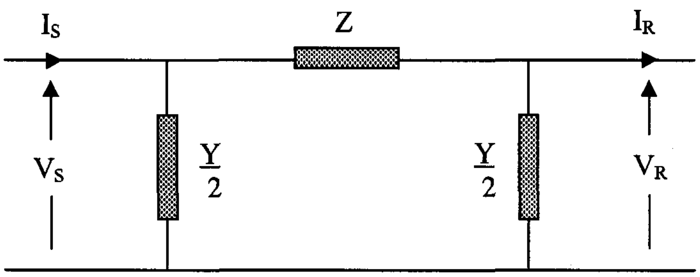


Figure 2.8 : Equivalent Π circuit of transmission line

Series impedance $Z = R + j\omega L$

Shunt admittance $Y = G + j\omega C$

Sending end voltage and current is given by

$$\begin{bmatrix} V_S \\ I_S \end{bmatrix} = \begin{bmatrix} \left(\frac{ZY}{2}+1\right) & Z \\ Y\left(1+\frac{ZY}{4}\right) & \left(1+\frac{ZY}{2}\right) \end{bmatrix} \begin{bmatrix} V_R \\ I_R \end{bmatrix} \tag{2.12}$$

In modeling, transmission lines are classified considering the length of line to ease the evaluation of parameters.

a.) short lines : lines shorter than 80 km (50 miles)

The lines are represented by their series impedance (Z)

b.) Medium-length lines : lines in the range 80 – 240 km (50 – 150 miles)

The lines are represented by nominal Π circuit

c.) Long lines : lines longer than 240 m (150 miles)

The distributed effects of the parameters are significant. If

$z = r + j\omega l$ impedance per unit length

$y = g + j\omega c$ admittance per unit length

$$Z = Z_C \sinh \gamma l \tag{2.14}$$

and

$$\frac{Y}{2} = \frac{1}{Z_C} \tanh \frac{\gamma l}{2} \tag{2.15}$$

where

$\gamma = \sqrt{zy}$ propagation constant

$Z_C = \sqrt{z/y}$ characteristic impedance

In general, series impedance and shunt susceptance per unit length are specified for the transmission lines.

2.2.4 Load

Load modeling has significant effect on system performance. Basically loads are represented as constant impedance, constant current, constant power or combination of all three types. In power flow studies, more basic approach of load modeling is constant MVA and in transient stability, loads are considered as constant impedance loads [13][14]. More specifically loads which are usually considered as the loads seen from bulk supply delivery points represent the aggregation of load components and power system components. Therefore, loads need more sophisticated representation in power flow and dynamic performance analysis of power systems. More precise load modelings are described in references [15][16][17]. In this project loads are considered as constant MVA in power flow study and constant impedance in transient stability analysis.

2.3 Pre-disturbance Analysis

In transient stability analysis, the system conditions prior to the disturbance and network configuration prior and after its occurrence must be known. The pre-transient operating configurations are often considered as steady state, i.e., the physical quantities which describe the operating conditions of the system can be considered constant for the purpose of analysis [1][18].

Thus, the two preliminary steps to be carried out in multi machine stability studies are [1][18]:

1. The pre-disturbance representation of the transmission network which is the basis of the disturbance and post disturbance network representation
2. The power flow analysis to determine the steady state operating conditions of the system

2.3.1 Pre-disturbance Network Representation

The matrix form of network equations formed according to the Kirchchoff's laws describes the behavior of power system network. The equations can be formed in bus frame of reference and branch frame reference. In power flow and power system stability studies admittance form of network representation in bus frame of reference is widely used [1]. Admittance form of network representation is given by

$$[I_{Bus}] = [Y_{Bus}][V_{Bus}] \tag{2.16}$$

where

I_{Bus} Vector of injected bus currents

V_{Bus} Vector of Bus voltages

Y_{Bus} Bus admittance matrix of power system network

Z_{Bus} Bus impedance matrix of power system network

The nodel admittance matrix $[Y_{bus}]$ in a typical power system is a sparse matrix and requires less computer storage. To enhance computational efficiency, sparsity techniques can be utilized. Therefore, in power system analysis, the bus admittance (Y) matrix is mostly preferable [1].

General Rules of Formulation of Y matrix

Network representation of n bus system [1][11]

$$\begin{bmatrix} I_1 \\ I_2 \\ \cdot \\ \cdot \\ I_i \\ \cdot \\ I_n \end{bmatrix}_{n\times 1} = \begin{bmatrix} Y_{11} & Y_{12} & \cdot & \cdot & Y_{1i} & \cdot & Y_{1n} \\ Y_{21} & Y_{22} & \cdot & \cdot & Y_{2i} & \cdot & Y_{2n} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ Y_{i1} & Y_{i2} & \cdot & \cdot & Y_{ii} & \cdot & Y_{in} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ Y_{n1} & Y_{n2} & \cdot & \cdot & Y_{ni} & \cdot & Y_{nn} \end{bmatrix}_{n\times n} \begin{bmatrix} V_1 \\ V_2 \\ \cdot \\ \cdot \\ V_i \\ \cdot \\ V_n \end{bmatrix}_{n\times 1} \tag{2.17}$$

a.) Diagonal elements of the Y matrix

Considering i^{th} bus

$$Y_{ii} = \frac{I_i}{V_i} \tag{2.18}$$

where ($V_j = 0, j = 1,2,\ldots..,n$) and ($i \neq j$)

Y_{ii} is the parallel admittance of all branches directly connected to bus *i*. Therefore

$$Y_{ii} = \sum_{j=1}^{n} y_{ij} \tag{2.19}$$

where ($i \neq j$)

y_{ij} - primitive admittance of branch *i* and *j*

b.) Off-diagonal elements of the Y matrix

$$Y_{ji} = Y_{ij} = \frac{I_j}{V_i} = -y_{ij} \tag{2.20}$$

where ($V_j = 0, j = 1,2,\ldots..,n$) and ($i \neq j$)

c.) Addition of Transformer Branch to Y matrix

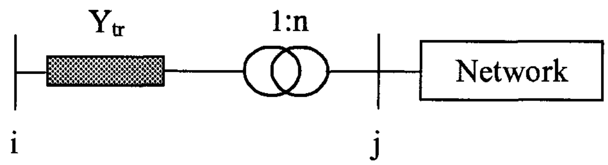


Figure 2.9 : Representation of transformer branch

If the network has n number of buses including i and j, the new bus admittance matrix

$$\begin{bmatrix} Y_{11} & Y_{12} & . & 0 & Y_{j1} & . & Y_{n1} \\ Y_{21} & Y_{22} & . & 0 & Y_{j2} & . & Y_{n2} \\ . & . & . & . & . & . & . \\ 0 & 0 & . & |n|^2 Y_{tr} & -n^* Y_{tr} & . & 0 \\ Y_{j1} & Y_{j2} & . & -nY_{tr} & Y_{tr}+Y_{jj} & . & Y_{nj} \\ . & . & . & . & . & . & . \\ Y_{n1} & Y_{n2} & . & 0 & Y_{jn} & . & Y_{nn} \end{bmatrix}_{n\times n} \tag{2.21}$$

d.) Modification of Y-matrix

Modification of Y-matrix according to the network changes, can be achieved through building block approach. The addition of the branch admittance y_a between buses m and n of an existing system, can be reflected by adding y_a to the elements Y_{mm} and Y_{nn} of system Y_{bus} and subtracting y_a from the symmetrical elements Y_{mn} and Y_{nm}. In other words

$$Y_{Bus(New)} = Y_{Bus(old)} + \Delta Y_{Bus} \tag{2.22}$$

where

$$\Delta Y_{Bus} = \begin{matrix} & m & n \\ m & y_a & -y_a \\ n & -y_a & y_a \end{matrix} \tag{2.23}$$

Removal of a branch can be reflected in a similar manner by subtracting the ΔY_{bus}.

Node Elimination (Kron Reduction)

In case of analyzing the operating parameters of some of the specified buses in a large power system, it is possible to use equivalent reduced order network representation of the power system. The reduced order representation is the Y_{bus} representation for an equivalent network containing only the buses to be analyzed. Gaussian elimination is the method commonly used in network reducing [1].

In case of net current injection of a bus is zero, the need for calculating operating parameters of the particular bus could be eliminated from the Y_{bus} matrix without harming voltage and current of the other buses. The process is called Kron elimination [1].

Considering four bus matrix with net current of bus 1 is zero

$$\begin{bmatrix} 0 \\ I_2 \\ I_3 \\ I_4 \end{bmatrix}_{4\times1} = \begin{bmatrix} Y_{11} & Y_{12} & Y_{13} & Y_{14} \\ Y_{21} & Y_{22} & Y_{23} & Y_{24} \\ Y_{31} & Y_{32} & Y_{33} & Y_{34} \\ Y_{41} & Y_{42} & Y_{43} & Y_{44} \end{bmatrix}_{4\times4} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \end{bmatrix}_{4\times1} \tag{2.24}$$

The matrix can be reduced to

$$\begin{bmatrix} I_2 \\ I_3 \\ I_4 \end{bmatrix}_{3\times1} = \begin{bmatrix} Y_{22(new)} & Y_{23(new)} & Y_{24(new)} \\ Y_{32(new)} & Y_{33(new)} & Y_{34(new)} \\ Y_{42(new)} & Y_{43(new)} & Y_{44(new)} \end{bmatrix}_{3\times3} \begin{bmatrix} V_2 \\ V_3 \\ V_4 \end{bmatrix}_{3\times1} \tag{2.25}$$

where

$$Y_{jk(new)} = Y_{jk} - \frac{Y_{jp} Y_{pk}}{Y_{pp}} \tag{2.26}$$

Here $j,k = 2,3,4$ and $p = 1$

Y_{pp} is the pivot.

2.3.2 Power Flow Analysis

The primary objective of the power flow analysis is to determine the steady state operating conditions of the power system network. The solutions of power flow study are of great importance in determining best operating conditions of the existing systems as well as in planning and designing the future expansion of power systems. The objective of the power flow analysis in stability studies, is to determine the initial operating conditions just before the disturbance affects the system. The primary information of power flow studies are magnitude and phase of voltage of each bus and active and reactive power injected to the system at each bus. The basis of the analysis are power flow equations of the system [1].

For n-bus system, active and reactive power entering to the system at i^{th} bus [1][11][13]

$$P_i = \sum_{j=1}^{n} \left| V_i V_j Y_{ij} \right| \cos(\delta_i - \delta_j - \theta_{ij}) \tag{2.27}$$

$$Q_i = \sum_{j=1}^{n} \left| V_i V_j Y_{ij} \right| \sin(\delta_i - \delta_j - \theta_{ij}) \tag{2.28}$$

For a n-bus system

Number of equations available - $2n$

Number of variables - $4n$

Therefore, in power flow analysis, two variables have to be specified at each bus. The type of bus is specified according to the known variables.

Table 2.1 : Load Flow- Bus Types

Bus type	**Specified variables**
PQ (load)	active and reactive power (P &Q)
PV (voltage controlled)	active power and bus voltage magnitude (P & V)
Swing (reference)	bus voltage magnitude and phase angle of voltage (V & δ)

Due to the non linearity of power flow equations, the calculations usually employ iterative techniques to determine the best solution depending on the specified tolerances of the mismatches of the known parameters of the bus. The two basic iterative techniques in power flow analysis, are Gauss-Seidel technique and Newton-Raphson technique. The former uses the mismatches of bus voltage magnitude and the latter uses the mismatches of active and reactive power. As the number of iterations involve in Gauss-Seidel method depends on the number of buses in the system, Newton-Raphson method is more preferable in power flow study. In case of large scale power systems, the improved versions of Newton-Raphson method – Decoupled power flow and Fast Decoupled power flow, are used to improve the computational efficiency and computer storage requirements. As the number of buses in this study is limited to 15, Newton-Raphson method is used for the analysis. Power flow analysis is described in most of the power system analysis text books including the references [1][12][13][19].

Newton Raphson Power Flow

Newton-Raphson power flow uses the Taylor's series expansion as the basis for power flow solution. For a system with m number of PQ buses and n number of PV buses, the power mismatches are as follows. Here, the bus number 1 is considered as swing bus [1][11].

$$
\begin{bmatrix} \Delta P_2 \\ \vdots \\ \Delta P_{m+n} \\ \hdashline \Delta Q_2 \\ \vdots \\ \Delta Q_n \end{bmatrix}_{(m+2n)\times 1}
=
\underbrace{\left[\begin{array}{ccc:ccc}
\frac{\partial P_2}{\partial \delta_2} & \cdots & \frac{\partial P_2}{\partial \delta_{m+n}} & |V_2|\frac{\partial P_2}{\partial |V_2|} & \cdots & |V_n|\frac{\partial P_2}{\partial |V_n|} \\
\vdots & & \vdots & \vdots & & \vdots \\
\frac{\partial P_{m+n}}{\partial \delta_2} & \cdots & \frac{\partial P_{m+n}}{\partial \delta_{m+n}} & |V_2|\frac{\partial P_{m+n}}{\partial |V_2|} & \cdots & |V_n|\frac{\partial P_{m+n}}{\partial |V_n|} \\
\hdashline
\frac{\partial Q_2}{\partial \delta_2} & \cdots & \frac{\partial Q_2}{\partial \delta_n} & |V_2|\frac{\partial Q_2}{\partial |V_2|} & \cdots & |V_n|\frac{\partial Q_2}{\partial |V_n|} \\
\vdots & & \vdots & \vdots & & \vdots \\
\frac{\partial Q_n}{\partial \delta_2} & \cdots & \frac{\partial Q_n}{\partial \delta_n} & |V_2|\frac{\partial Q_n}{\partial |V_2|} & \cdots & |V_n|\frac{\partial Q_n}{\partial |V_n|}
\end{array}\right]_{(m+2n)\times(m+2n)}}_{\text{Jacobian matrix}}
\begin{bmatrix} \Delta \delta_2 \\ \vdots \\ \Delta \delta_{m+n} \\ \hdashline \frac{\Delta V_2}{|V_2|} \\ \vdots \\ \frac{\Delta V_n}{|V_n|} \end{bmatrix}_{(m+2n)\times 1}
$$

a.) **Off-diagonal elements of Jacobian matrix**

$$\frac{\partial P_i}{\partial \delta_j} = \left|V_i V_j Y_{ij}\right| \sin(\delta_i - \delta_j - \theta_{ij}) \quad (2.31)$$

$$\frac{\partial Q_i}{\partial \delta_j} = -\left|V_i V_j Y_{ij}\right| \cos(\delta_i - \delta_j - \theta_{ij}) \quad (2.32)$$

$$\left|V_j\right| \frac{\partial P_i}{\partial \left|V_j\right|} = \left|V_i V_j Y_{ij}\right| \cos(\delta_i - \delta_j - \theta_{ij}) = -\frac{\partial Q_i}{\partial \delta_j} \quad (2.33)$$

$$\left|V_j\right| \frac{\partial Q_i}{\partial \left|V_j\right|} = \left|V_i V_j Y_{ij}\right| \sin(\delta_i - \delta_j - \theta_{ij}) = \frac{\partial P_i}{\partial \delta_j} \quad (2.34)$$

b.) **Diagonal elements of Jacobain matrix**

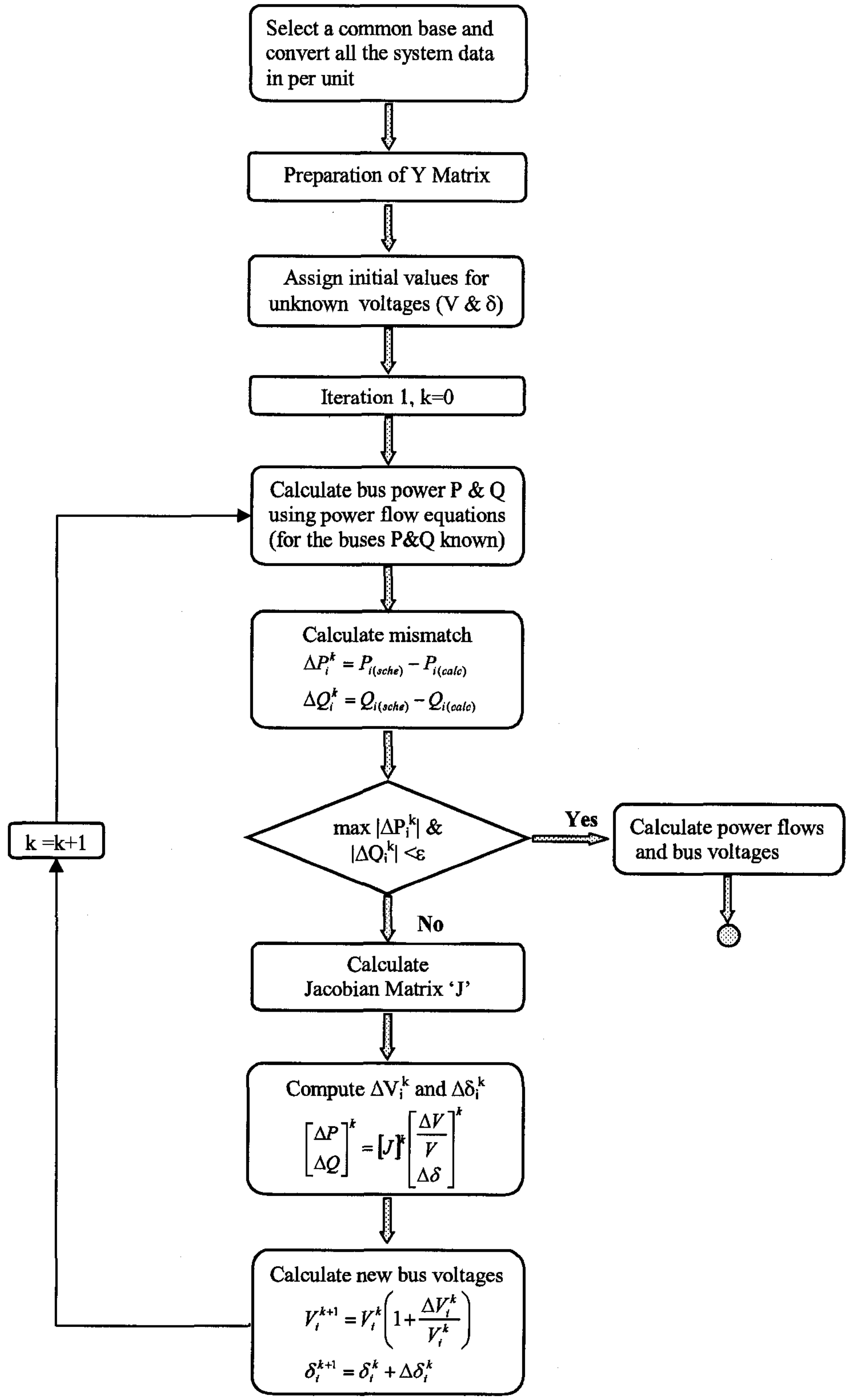
$$\frac{\partial P_i}{\partial \delta_i} = -\sum_{\substack{j=1 \\ j \neq i}}^{N} \left|V_i V_j Y_{ij}\right| \sin(\delta_i - \delta_j - \theta_{ij}) = -\sum_{\substack{j=1 \\ j \neq i}}^{N} \frac{\partial P_i}{\partial \delta_j} \quad (2.35)$$

$$\frac{\partial Q_i}{\partial \delta_i} = \sum_{\substack{j=1 \\ j \neq i}}^{N} \left|V_i V_j Y_{ij}\right| \cos(\delta_i - \delta_j - \theta_{ij}) = -\sum_{\substack{j=1 \\ j \neq i}}^{N} \frac{\partial Q_i}{\partial \delta_j} \quad (2.36)$$

$$\left|V_i\right| \frac{\partial P_i}{\partial \left|V_i\right|} = \sum_{\substack{j=1 \\ j \neq i}}^{N} \left|V_i V_j Y_{ij}\right| \cos(\delta_i - \delta_j - \theta_{ij}) + 2\left|Y_{ii} V_i^2\right| \cos\theta_{ii} \quad (2.37)$$

$$\left|V_i\right| \frac{\partial Q_i}{\partial \left|V_i\right|} = \sum_{\substack{j=1 \\ j \neq i}}^{N} \left|V_i V_j Y_{ij}\right| \sin(\delta_i - \delta_j - \theta_{ij}) - 2\left|Y_{ii} V_i^2\right| \sin\theta_{ii} \quad (2.38)$$

c.) Iterative Algorithm of Newton-Raphson Power Flow



2.4 Modeling of Synchronous Generator for Stability Studies

In multi-machine power system, the stability is concerned with maintaining synchronism in between synchronous machine and another[2]. Therefore, accurate modeling of their dynamic performance and understanding of their characteristics are of fundamental importance to the study of power system stability. Here, the detailed modeling of synchronous machine is presented.

2.4.1 Equation of Motion

Swing equation is the fundamental equation which governs the rotational motion of the synchronous machine and it describes the effects of unbalance between the mechanical torque and electromagnetic torque of individual machines [1].

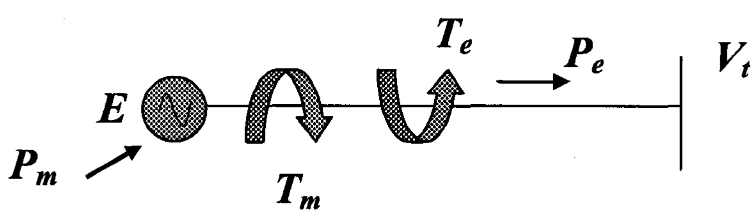


Figure 2.10 : Representation of machine dynamics

The swing equation [16]

$$\frac{2H}{\omega_s}\frac{d\omega}{dt} + K_D\frac{d\delta}{dt} = T_m - T_e \text{ in p.u} \tag{2.39}$$

where

$$\frac{d\delta}{dt} = \omega \tag{2.40}$$

In the above equation, the variation of speed ω reference to synchronous speed ω_S is in radians per second and δ the angular position of the rotor is in radians while electromagnetic torque T_m and T_e are in per unit. H is the inertia constant of the machine in MJ/MVA. Machine damping is denoted by damping constant K_D. In stability analysis, normalized forms of swing equation could be used. In normalized form, all parameters are considered in per unit [18].

2.4.2 Synchronous Generator Modeling for Dynamic Analysis

The basic approach in modeling of synchronous generator is to consider an arrangement of three stator windings 120 electrical degrees apart, and a rotating structure with an excitation or field winding and one or more equivalent rotor body windings. The magnetic axis of the field winding is defined as the direct axis and the axis 90° electrical degrees away from it is defined as the quadrature axis. The equivalent rotor windings reflect induced current paths in round rotor body or in damper bars in round-rotor turbo generators or salient pole hydro generators. One group of equivalent circuits is aligned in the direct axis and the other group is aligned at quadrature axis[5]. The arrangement is shown in figure 2.11.

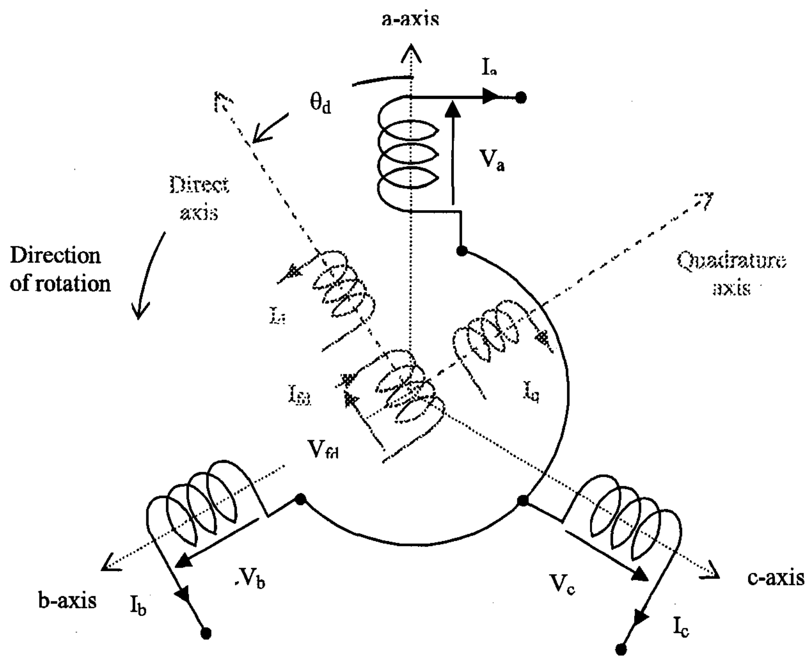


Figure 2.11 : The arrangement of the generator for modeling

In generator modeling, the usual practice is to transform the three phase stator quantities into corresponding two axis quantities. The effect of these transformations is to move time varying three phase stator voltage, current and flux linkages into time invariant direct and quadrature axis quantities under steady state conditions. In modeling, this transformations are visualized by d-axis and q-axis equivalent circuits of the generator. Depending on the number of equivalent circuits, the order of it ranges from first order to third order. Here, we consider the standard second order model with one damper circuit in each axis [5].

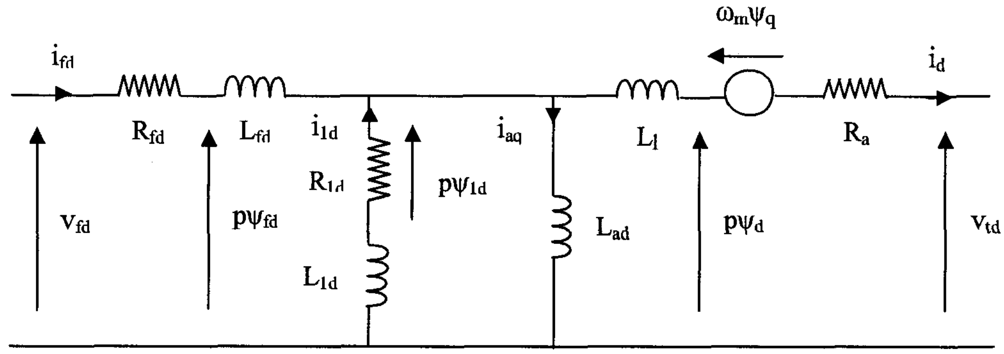


Figure 2.12 : D-axis equivalent circuit of synchronous machine

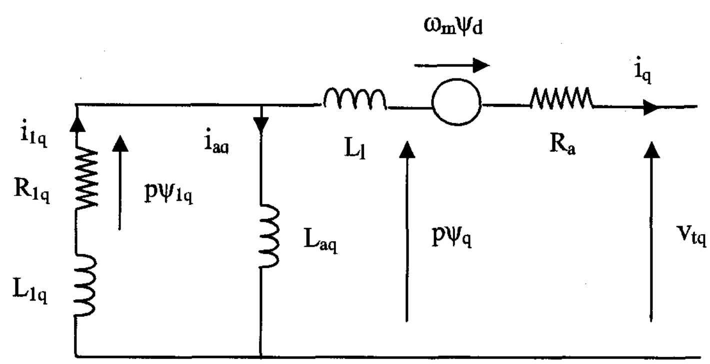


Figure 2.13 : Q-axis equivalent circuit of synchronous machine

Complete set of electrical equations in per unit is given by [16].

Stator voltage equations

$$V_{td} = -R_a i_d - \omega_m \psi_q + p\psi_d \tag{2.41}$$

$$V_{tq} = -R_a I_q + \omega_m \psi_d + p\psi_q \tag{2.42}$$

Rotor Voltage Equations

$$v_{fd} = -R_{fd} i_{fd} + p\psi_{fd} \tag{2.43}$$

$$0 = -R_{1d} i_{1d} + p\psi_{1d} \tag{2.44}$$

$$0 = -R_{1q} i_{1q} + p\psi_{1q} \tag{2.45}$$

Stator flux linkage equations

$$\psi_d = -(L_{ad} + L_l)i_d + L_{ad}(i_{fd} + i_{1d}) \tag{2.46}$$

$$\psi_q = -(L_{aq} + L_l)i_q + L_{aq}i_{1q} \tag{2.47}$$

Rotor flux linkage equations

$$\psi_{fd} = (L_{fd} + L_{ad})i_{fd} + L_{ad}(i_{1d} - i_d) \tag{2.48}$$

$$\psi_{1d} = (L_{1d} + L_{ad})i_{1d} + L_{ad}(i_{fd} - i_d) \tag{2.49}$$

$$\psi_{1q} = (L_{1q} + L_{aq})i_{1q} - L_{aq}i_q \tag{2.50}$$

Air gap torque

$$T_e = \psi_d i_q - \psi_q i_d \tag{2.51}$$

In addition to above equations, in generator modeling the other equations describing generator controls must be included in the mathematical model. Thus the complete mathematical model of a large power system is exceedingly complex, and simplifications are often used in modeling the system. The simplifications depend upon the location of the machine with respect to the disturbance [18]. In this project, the simplified model of the machine, the classical representation is considered. In classical representation, two axis model of the generator is not considered.

2.5 Transient Stability Analysis

2.5.1 Classical Stability Model

In classical stability model, the classical representation of synchronous machine is used to study the system dynamics of the power system. It is the simplest model in transient stability analysis and derived under several simplified assumptions [1][18][20].

1. Mechanical power input (P_m) remains constant during the period of transient.
2. Damping is negligible.
3. The synchronous machine can be represented by a constant voltage source behind transient reactance.
4. The rotor angle of the machine coincides with the phase angle of the voltage behind transient reactance.
5. Loads are represented by passive impedances.

The use of classical stability model is limited to study the transients for the 'first swing' or for the periods in the order of one second. In this period dynamic response is largely dependent on the stored kinetic energy in the rotating masses. This model could be used as preliminary studies to identify problem areas that require further study with more detailed modeling.

The two equations which are used in classical stability analysis (for i^{th} machine) are [20]

$$\frac{d\omega_i}{dt} = \frac{\omega_S}{2H_i}\left(P_{mi} - |E_i|^2 G_{ii} - |E_i| \sum_{\substack{j=1 \\ j \neq i}}^{n} |E_j||Y_{ij}| \cos(\delta_i - \delta_j - \theta_{ij}) \right) \tag{2.52}$$

$$\frac{d\delta_i}{dt} = \omega_i - \omega_S \tag{2.53}$$

The first equation can be derived by substituting electrical power generated by the machine to the swing equation.

Procedure of transient stability study using classical model [12][18][20]

Step 1: Perform load flow analysis (pre-fault analysis) to obtain the values of active power, reactive power, voltage and its angle at each generator terminal and load bus.

Step 2: For each generator in the system, calculate the internal voltage (E) and initial rotor angle (δ_0)

$$I^*_{gen} = \frac{(P_{gen} + Q_{gen})}{V_t \angle \theta} \tag{2.54}$$

$$E \angle \delta_0 = V_t \angle \theta + jX'_d I_{gen} \tag{2.55}$$

Step 3: Convert each load in the system into a constant shunt admittance to ground at its bus

$$Y_L = G_L + jB_L = \frac{P_L - jQ_L}{|V_L|^2} \tag{2.56}$$

Step 4: Augment bus admittance matrix (Y_{bus} used for power flow analysis) to include transient reactance of each generator and shunt admittance of each load. Augmentation method is described in [1].

Step 5: Modify the bus admittance matrix to represent the faulted and post-fault conditions. To represent the fault, the position of the fault should be short circuited to the reference. In post-fault, faulty portion should be eliminated from the network and Y_{bus} should be modified accordingly. The resulting admittance matrix is further reduced by Kron elimination to represent the admittance matrix corresponding to number of generators.

Step 6: Analyze the two equations to determine the variation of active power generated, speed and rotor angle with time by using numerical integration technique (e.g.: Euler, Runge Kutta, trapezoidal etc.) by specifying suitable sampling period Δt.

CHAPTER 3

IMPLEMENTATION OF GRAPHICAL USER INTERFACE

The graphical user interfaces of this program is developed using GUI tools and handle structure environment provided in MATLAB version 6.1. The main purpose of GUI implementation is to provide the facility to enter and store the data required for the pre-transient and transient stability analysis. Apart, some GUI's are developed to make links between them [21].

3.1 'MainGUI' : GUI for Entering into the System

The purpose of developing 'MainGUI' is to provide the access for entering into the system. It links the other files required to understand and carry out transient stability analysis.

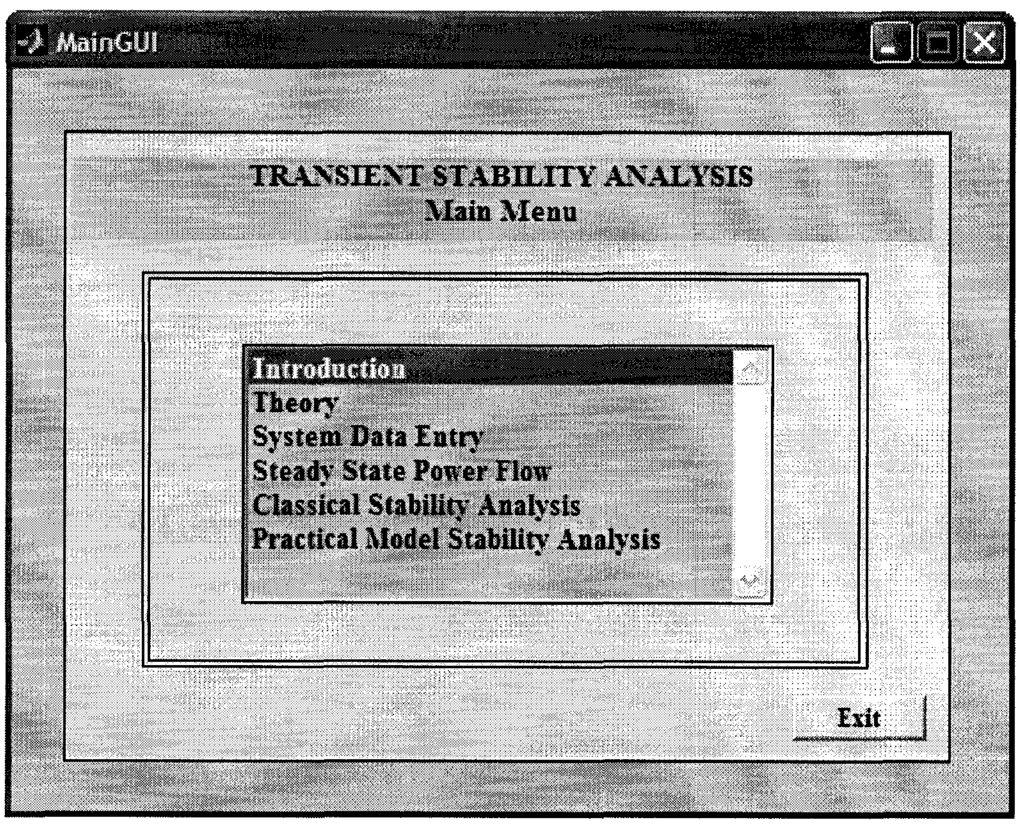


Figure 3.1 – Main GUI

For entering to the 'MainGUI'

Open the MATLAB window and select the current directory in which the program is installed and give the command 'maingui' in command window and 'Enter'.

e.g.: the directory C:\MATLAB6p1\work\GUI

The commands of GUI provides the following facilities.

Introduction - links the .pdf file which describes the utilization of program

Theory - links the .pdf file which describes the theory needed to understand the transient stability analysis.

System Data Entry - links the GUI for system data entry

Steady State Power Flow, Transient Stability Analysis

- links the GUI for transient stability analysis.

3.2 'SystemData' : GUI for Entering to the System Data Entry

The GUI 'SystemData' links the GUIs which are used for system data entry of individual components.

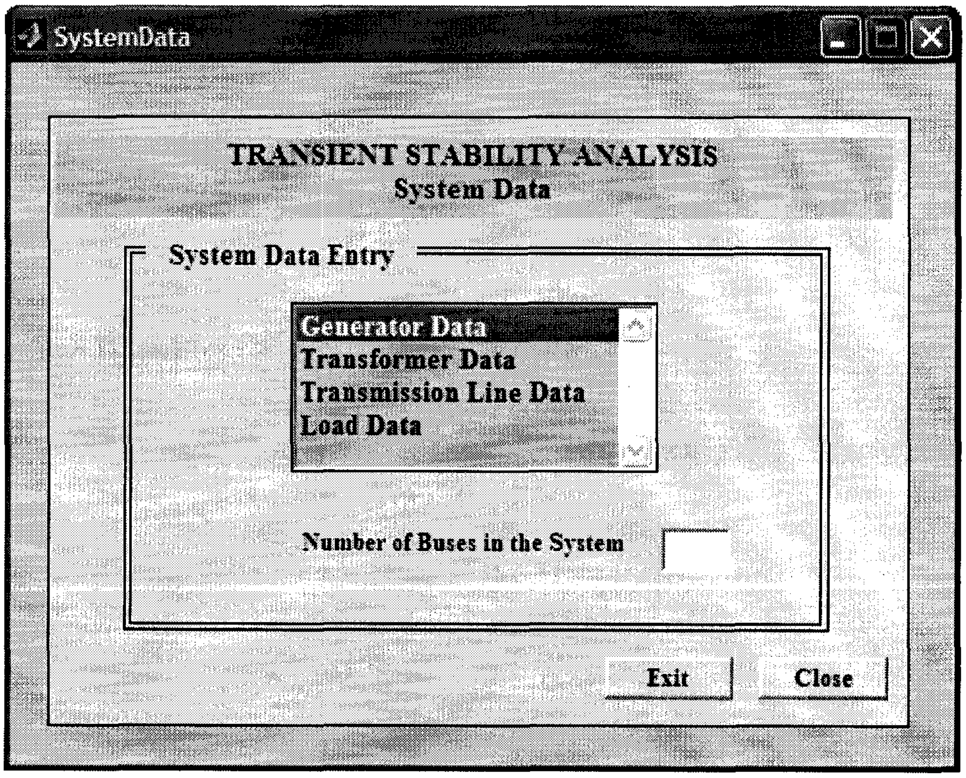


Figure 3.2 – GUI "System Data"

The GUI itself provides the facility to enter the number of buses in the system and get it stored with the 'Exit' command. 'Exit' links the 'MainGUI' back and 'Close' will close the figure.

3.3 GUI's for Entering System Data

Four separate GUIs are developed for data entering of individual components- generators, transformers, transmission lines and loads. Here, the loads are considered as constant impedance loads considering the programming simplicity in reduced Y-bus formation after the disturbance. The program is developed considering the data is in per unit with respect to the common reference except the reactance data of transformers. Transformer reactance data has to be entered in local reference as the global reference is provided in the formation of Y-matrix. Here, it is considered that the global reference is 100MVA.

3.3.1 'System Data_Generator' - Entering Generator Data

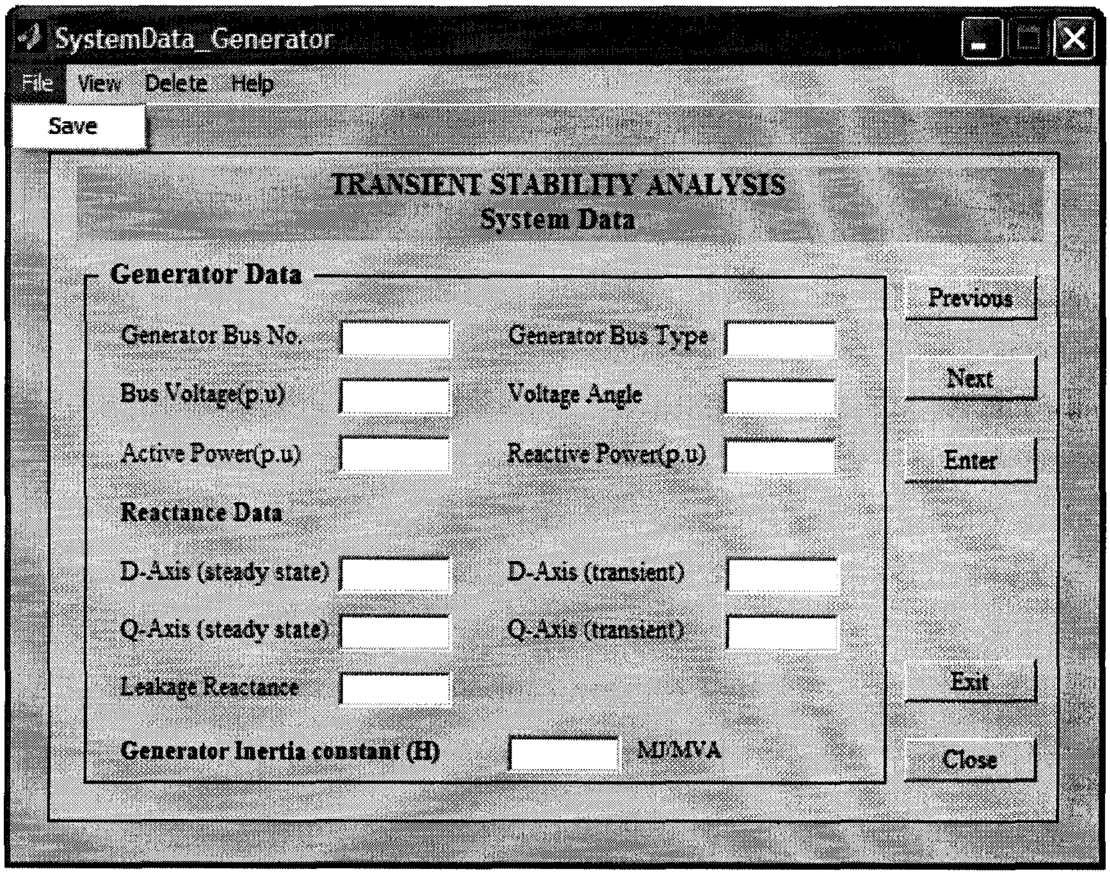


Figure 3.3 – GUI for generator data

The description of the Data Fields:

Generator Bus No.	- user specified bus number	
Generator Bus Type	- type of bus in numeric	1 - swing bus
		2 - PV bus
		3 - PQ bus
Bus Voltage (p.u)	- enter if known, else 1 p.u.	
Voltage Angle	- in radians if known, else 0	
Power in per unit	- active and reactive power separately	
Reactance data	- in per unit w.r.t. global reference	
Inertia constant	- in MJ/MVA	

The description of the Push Buttons:

Previous	- to go to previous set of data
Next	- to go to next set of data if 'Previous' button is used
Enter	- to enter the set of data after entering all the data
Exit	- to go back to 'System Data' GUI
Close	- to close the figure

The description of the Menu Editor:

File-Save	- to save the entered data in 'gendata.mat' file
View-Generator Data	- to view the 'gendata.mat' file
Delete	- delete set of data Editing of 'gendata.mat' file provides the facility to replace the specified data. The file has to be saved as a workspace after changing.
Help	- provides the details of utilization of current GUI

The utilization of 'menu editor' and 'push buttons' in other GUI used for data entry is similar to the utilization them in entering generator data.

3.3.2 'System Data_Transformer' - Entering Transformer Data

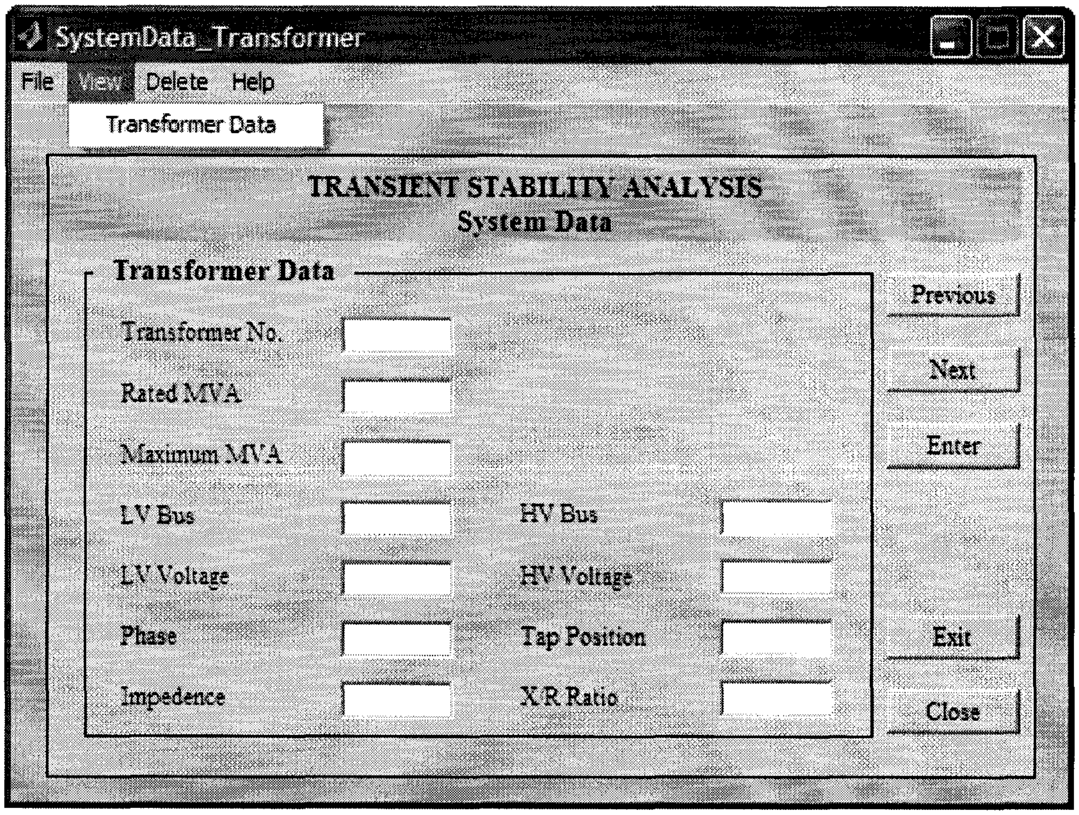


Figure 3.4 – GUI for transformer data

The description of the Data Fields:

Transformer Bus No.	- user specified bus number
Rated MVA	- rated apparent power of the transformer
Maximum MVA	- maximum power the transformer could deliever (not considered in the study)
LV Bus	- the bus which transformer LV side connected
LV Voltage	- rated voltage of low voltage side
HV Bus	- the bus which transformer HV side connected
HV Voltage	- rated voltage of high voltage side
Phase	- phase difference from primary to secondary
Tap Position	- voltage boost in secondary in per unit (eg: 1.04)
Impedance	- transformer impedance referred to secondary
X/R ratio	- impedance to resistance ratio

The transformer data is stored in 'transfdata.mat' file.

3.3.3 'System Data_Transmission' - Entering Transmission Line Data

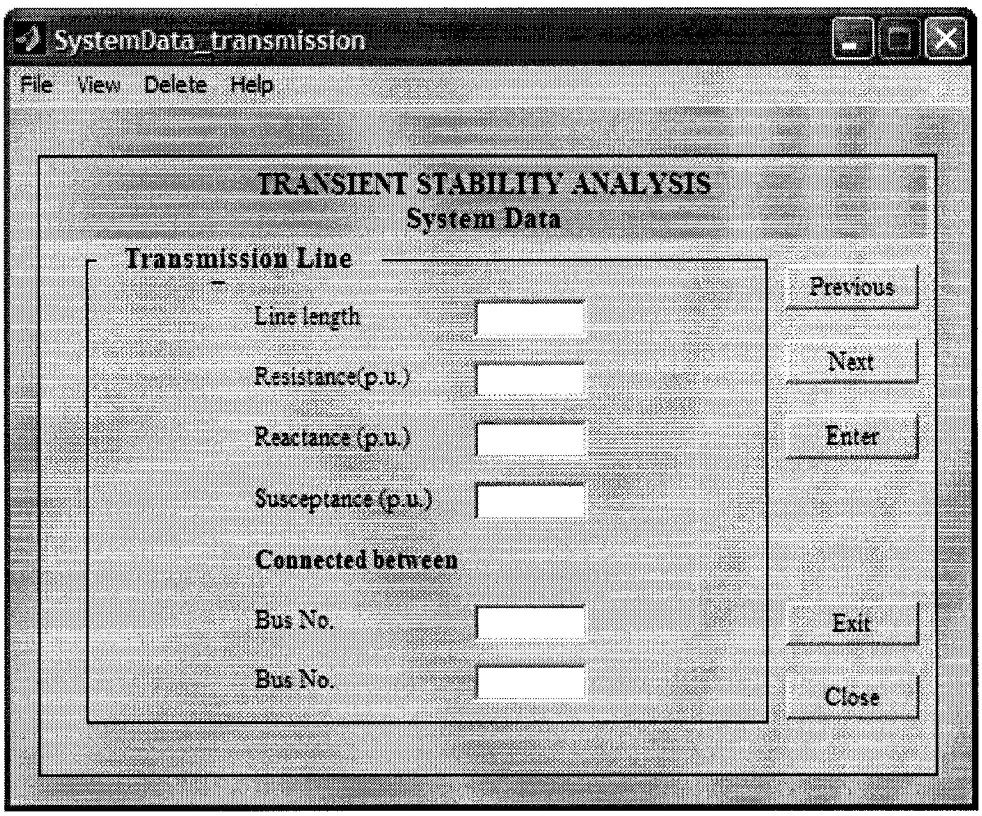


Figure 3.5 – GUI for transmission line data

The description of the Data Fields:

Line length	- length of the line
Resistance	- total series resistance (R) in per unit w.r.t global reference
Reactance	- total series rectance (X) in per unit w.r.t global reference
Susceptance	- shunt susceptance (B/2) in per unit w.r.t global reference
Bus No.	- two buses which the transmission line is connected

The transmission line data is stored in 'transldata.mat' file.

3.3.4 'System Data_Load' - Entering Load Data

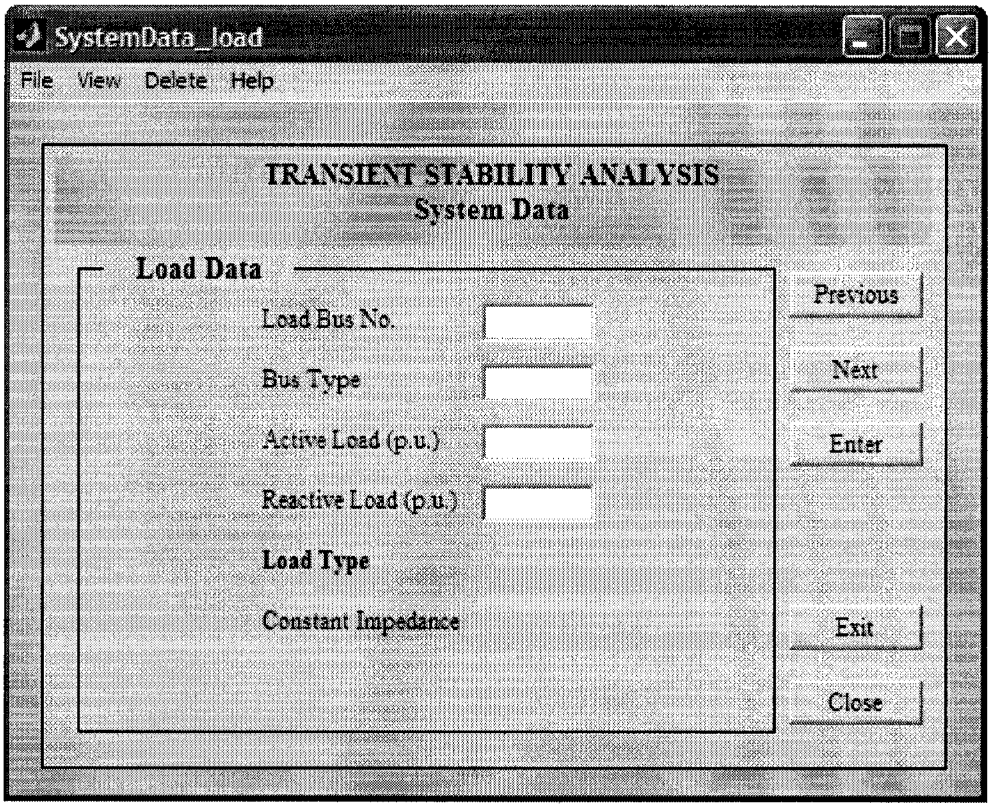


Figure 3.6 – GUI for System Load data

The description of the Data Fields:

Load Bus No.	- user specified bus number
Bus Type	- Load bus (PQ) ,type 3
Active Load	- active power in per unit w.r.t global reference
Reactive Load	- reactive power in per unit w.r.t global reference

The load data is stored in 'loadata.mat' file.

3.4 'Transient Analysis' : GUI for Transient Stability Analysis

The GUI is developed to analyze the power flow and transient stability of the system. The GUI loads the system data to the program to execute the system Y-matrix, power flow results and transient stability results in graphical form. GUI provides the facility to store the results of Y-matrix formation and power flow in Y-matrix.mat and powerflow.mat for use in transient stability analysis. The data required to carry out the numerical integration should be entered before executing the transient stability analysis.

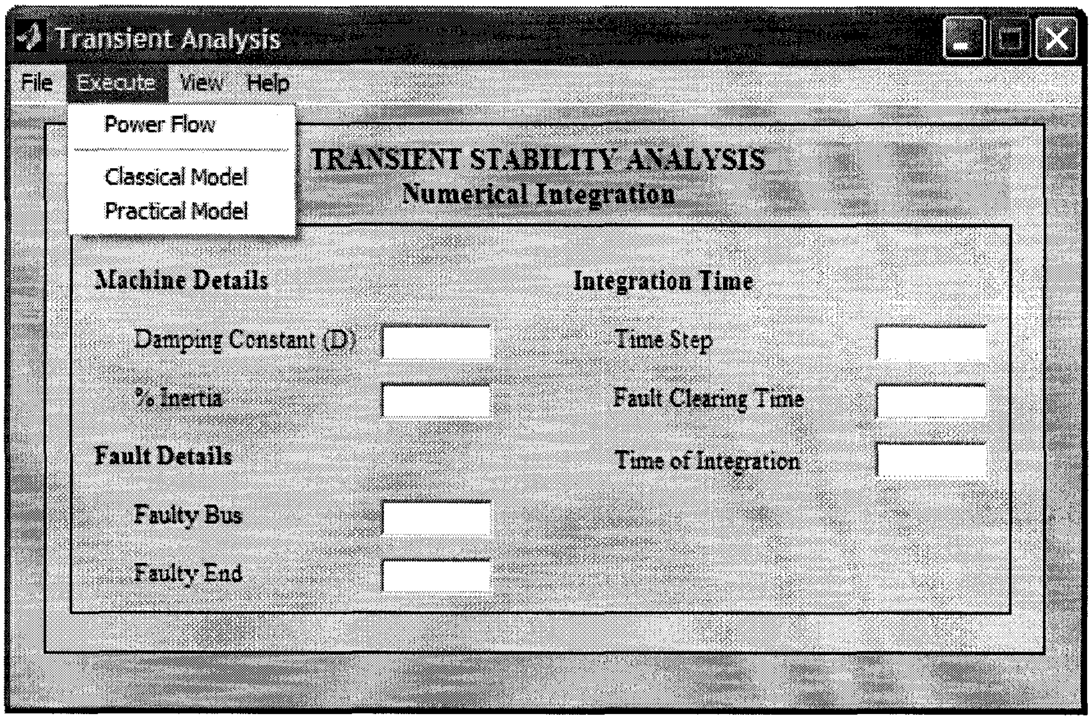


Figure 3.7 – GUI for Transient Stability Analysis

The description of the Data Fields:

Damping Constant	- considering that D is same for all machines
% inertia	- to check the effect of machine inertia on stability
Fault Bus	- considering the fault occurs at the end of transmission line
Fault End	- bus to which the other end of the transmission line is connected
Time step	- time step of integration in seconds
Fault clearing time	- time at which the fault is cleared (in seconds)
Time of integration	- length of time for which integration is to be performed

The description of the Menu Editor:

File-Open	- to load system data required for load flow and stability analysis. The files to be loaded are 'gendata.mat', 'transfdata.mat', 'transldata.mat', 'loaddata.mat' and 'NoofBuses.mat'. These files give the whole set of data required.
File-Save	- to save Y-matrix and power flow results
File-Open	- to delete the set of data in edit boxes in GUI to access new set of data
File-Exit	- to exit from GUI

Execute-Power Flow	- to execute the power flow program
Execute-Classical Model	- to execute the stability analysis using classical representation
Execute-Practical Model	- to execute the stability analysis using two axis model representation

Using 'View' menu gives the facility to view the system data, system admittance matrix, jacoban matrix, power flow results and transient stability results. The idea behind it is to give the facility to check the data if the program runs incorrectly.

'Help' provides the links to the details of utilization of current GUI and to theory.

The programs connected with GUI implementation are described in next chapter in pseudo format. The .m file format of the programs is written in appendix.

CHAPTER 4

FLOW OF SIMULATION PROGRAM

The simulation program is developed according to the flow of GUI implementation. The pseudo code of the program involved with each GUI implementation is described here to understand the program flow.

4.1 Program : 'maingui.m'

The program connected with the 'MainGUI' figure

a.) Function - **'MainGUI_listbox_Callback'**

Selects the next step by tallying with the string in the list box.

b.) Function – **'Exit_pushbutton_Callback'**

Close the current figure.

4.2 Program : 'systemdata.m'

The program connected with the 'SystemData' figure

a.) Function - **'systemdata_listbox_Callback'**

Selects the corresponding GUI by tallying with the string in the list box.

b.) Function – **'Exit_pushbutton_Callback'**

Opens the 'NoofBuses.mat' file and stores new data while transferring back to mainGUI.

c.) Function – **'Close_pushbutton_Callback'**

Close the current figure without moving back to 'Maingui'.

4.3 Program : 'generatordata.m'

The program connected with the 'SystemData-Generator' figure

a.) Function - **'Enter_pushbutton_Callback'**

1. Reads the data in the data fields of the GUI.
2. Starts the indexing of number of data set from 1.
3. If there is field named 'GeneratorData' fill the matrix with the data read, else generate the 'GeneratorData' field and fill the matrix.

4. Set the data field in GUI empty.
5. Store ‘GeneratorData’ and ‘index’ on handle structure for future use.

b.) Function - **‘Previous_pushbutton_Callback’**

1. Read current ‘index’ and ‘GeneratorData’ from handle structure.
2. Set handles of ‘Enter’ pushbutton ‘off’ and ‘Next’ push button ‘on’.
3. Set the data field of the GUI with set of GeneratorData corresponding to current ‘index’.
4. If ‘index ≠ 1’, Set ‘index = index-1’
5. Store new ‘index’ in handle structure.

c.) Function - **‘Next_pushbutton_Callback’**

1. Read current ‘index’ and ‘GeneratorData’ from handle structure.
2. Read the size of ‘GeneratorData’ (gives the number of set of data)
3. If the row size is ‘r’ and ‘index<= r’, set index=index+1.
4. If ‘index<= r’, set the data field of the GUI with set of GeneratorData corresponding to current ‘index’, else set handles of ‘Enter’ push button ‘on’ and ‘Next’ push button ‘off’ while setting the data file of GUI empty.
5. Store new ‘index’ in handle structure.

d.) Function - **‘Save_Callback’**

1. Read ‘GeneratorData’ from handle structure.
2. Open ‘gendata.mat’ file and make it empty.
3. Load ‘gendata.mat’ file with new ‘GeneratorData’.
4. Save ‘gendata.mat’ file

e.) Function - **‘ViewGenData_Callback’**

1. Open the ‘gendata.mat’ file
2. Load gendata variable field from gendata.mat.
3. Open gendata. (matrix form)

f.) Function - **'DeleteSelect_Callback'**

1. Read current 'index' and 'GeneratorData' from handle structure.
2. Read the size of 'GeneratorData' (r)
2. Set the set of 'GeneratorData' corresponding to the current 'index' empty.
3. If current 'index' is '*i*', then fill GeneratorData(*i*,:) by Generatordata(*i*+1,:)
4. Repeat procedure until $i = r\text{-}1$
5. Store new 'index' in handle structure.

g.) Function - **'HelpGUI_Callback'**

1. Links the file 'program.pdf' which tell about utilization of GUI

h.) Function - **'Exit_Callback'**

1. Delete current figure
2. Find object which has the name 'SystemData'

i.) Function - **'Close_Callback'**

1. Close current figure

Note: The flow of the programs of **'transmissiondata.m'**, **'transformerdata.m'** and **'loaddata.m'** is similar to the **'generatordata.m'** file except the data fields.

4.4 Program : 'transient.m'

The program connected with the 'Transient Analysis' figure

a.) Function - **'Open_Callback'**

1. Opens the data files of generators, transformers, transmission lines, loads and number of buses and read them
2. Store them in handle structure

b.) Function - **'PowerFlow_Callback'**

The program of the function is arranged in the following order.

Step 1. To prepare system bus admittance matrix using the data of transformers and transmission lines

1. Read data 'number of buses (N)' and 'transformer data'
2. Calculate 'complex turns ratio' of each transformer using the data fields 'phase' and 'tap position'
3. Prepare transformer bus admittance matrix using function **'Ytransf'**

   Function – **'Ytransf'**

   - Calculate transformer admittance of each transformer using the data, transformer impedance and inductance to resistance ratio
   - Set the matrix to NxN
   - Calculate the bus admittances of each transformer connected buses using the equation

$$\begin{bmatrix} |n|^2 Y & -n^* Y \\ -nY & Y \end{bmatrix} \begin{bmatrix} V_S \\ V_R \end{bmatrix} = \begin{bmatrix} I_S \\ I_R \end{bmatrix} \qquad (4.1)$$

   where

   n - complex turns ratio

   Y - transformer admittance

4. Read 'transmission line data' and calculate transformer series impedance
5. Prepare transformer bus admittance matrix using function **'Ytransm'**

   Function – **'Ytransm'**

   - Set the matrix to NxN
   - Calculate series admittances of each line and set the admittance matrix (NxN) off diagonal elements considering series admittances
   - Add the bus susceptances to their diagonal elements to complete the admittance matrix
6. To get system admittance matrix add two matrices 'Ytransf' and "Ytransm'

Step 2. Arranging system data for power flow calculation

1. Read generator data and size of the 'gendata' file
2. Read load data and size of the 'loadata' file
3. Prepare the 'Data' file considering the bus numbers which the generators and loads are connected

   Data = ['Bus_No.' 'Bus_type' 'Bus_Voltage' 'V_Angle' 'P_gen' 'Q_gen' 'P_load' 'Q_load']
4. Prepare 'machine_data' file considering generator bus numbers

   machine_data = ['Bus_No' 'Xd' 'Transient Xd' 'Xq' 'Transient Xq' 'Leakage X' 'Inertia constant'];
5. Modifying 'Data' file for calculation subtracting P_load from P_gen and Q_load from Q_gen (matrix –'spec')

Step 3. Power flow calculation using Newton Raphson method

The method of power flow calculation is described in Chapter 2.

1. For the first iteration, the modified data from Step2 is to be used.
2. Calculate the power P and Q at each bus using power flow equations and thereby calculate the power mismatches at each bus
3. Separate the active power mismatches (m) considering the bus type PV and PQ (2&3) and reactive power mismatches (n) by the bus type PQ (3)
4. Arrange the mismatches in a matrix form ((m+n)x1)
5. Calculate Jacobian matrix using the function **'Jacobian'**

   Function – **'Jacobian'**
   - Partial derivatives of P against angle – daba_PDelta (mxm)
   - Partial derivatives of P against voltage – daba_PU (mxn)
   - Partial derivatives of Q against angle – daba_QDelta (nxm)
   - Partial derivatives of Q against voltage – daba_QU (nxn)
   - Jacobian = [daba_PDelta Vdaba_PU;daba_QDelta Vdaba_QU]
6. Compute difference of voltage and angle using matrix of mismatches and jacobian matrix and then separate them using the size 'm' and 'n'
7. Calculate the new values of voltage and angle and assign them to the modified data matrix 'spec'.

8. Repeat the iterative process from 1 until the minimum error of mismatches is less than some specified value (eg : error<0.000001) and update the 'spec' to get the final results of power flow.

Step 4. Calculation of generator internal voltages at steady state

1 Calculate bus loads using bus voltage and Y matrix and separate out generator bus voltage and bus loads

2 Compute the steady state internal voltage of each generator by using the equation

$$E = V_t + I_a R_a + jI_d X_d + jI_q X_q \tag{4.2}$$

Reactance data is in 'machine_data' file. To compute internal voltage function **'emf'** is used.

c.) Function - **'ClassicalModel_Callback'**

1 Calculate the transient state internal voltage of generators using transient reactance data. The function **'emf'** will calculate the internal voltage.

2 Represent loads as constant impedance loads connected between relevant bus and reference point.

3 Prepare new system admittance matrix considering the impedance of the loads and transient reactance of the generators.

4 Get the data about faulty bus and faulty end from the edit fields of 'Transient Analysis' GUI

5 Prepare the Y-matrix for faulty condition by removing the row and column corresponding to faulty bus

6 Prepare Y-matrix for post fault condition by removing the faulty line from the Y-matrix

7 Reduce all three matrices by Kron-Elimination method keeping only generator buses in the Y-matrices.

Function for Kron elimination - **'kron'**.

8 Analyze the transient stability of the system using the function **'transient'**

The function **'transient'** analyzes the transient stability of the system by numerical integration of swing equation. The numerical integration is done using 'Euler' method. The two equations used are

$$\Delta(\Delta\omega) = \frac{\omega_S}{2H}((P_m - P_e) - D(\Delta\omega))\Delta t \tag{4.3}$$

$$\Delta(\Delta\delta) = \Delta\omega.\Delta t + \frac{\omega_S}{2H}((P_m - P_e) - D(\Delta\omega)).\frac{\Delta t^2}{2} \tag{4.4}$$

9 Plot the graph of Pe, Δω and Δt w.r.t time to analyze the transient stability of the system.

d.) Function - **'Save_Callback'**

1. To save Y-matrix and power flow results in Y_matrix.mat and powerflow.mat files. The algorithm is same as in data files.

e.) Function - **'Exit_Callback'**

1. To close the figure and return back to maingui.

CHAPTER 5

CASE STUDY

This chapter covers the study of transient stability response of a nine bus system and a fifteen bus system under three phase symmetrical faults. The study was carried out using the developed software. Software is developed to analyze the three phase symmetrical faults considering the severity of the fault and its usefulness in elimination of symmetrical component calculation.

5.1 Nine Bus System

The configuration and the system data of the nine bus system were taken from reference [18]. Validity of the developed software is checked with some of the results given in the reference.

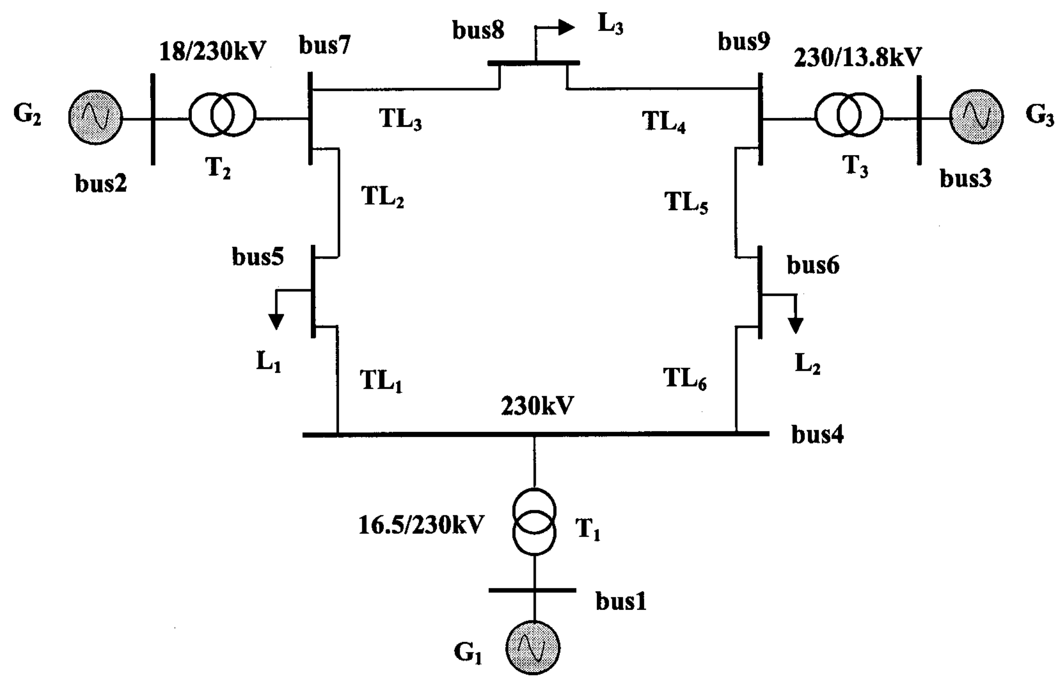


Figure 5.1 – Configuration of Nine Bus System

In case of nine bus system, transient stability response for two cases- system fault at bus 7 and system fault at bus 5 in transmission line TL2 is analyzed. The factors which affect the transient stability response, machine damping , machine inertia, fault position and fault clearing time are analyzed in the study. The fault at the middle point of the transmission line is omitted considering the difficulty of network representation at fault and post fault conditions.

5.1.1 System Data

a.) Table 5.1 : Generator Data

Generator	G_1	G_2	G_3
Rated MVA	247.5	192.0	128.0
kV	16.5	18.0	13.8
Power factor	1.0	0.85	0.85
Type	hydro	steam	steam
Speed (r.p.m)	180	3600	3600
X_d	0.1460	0.8958	1.3125
X'_d	0.0608	0.1198	0.1813
X_q	0.0969	0.8645	1.2578
X'_q	0.0969	0.1969	0.25
X_l	0.0336	0.0521	0.0742
T'_{do}	8.96	6.00	5.89
T'_{qo}	0	0.535	0.600
H (MJ/MVA)	9.55	3.33	2.35

b.) Table 5.2 : Known Loading Data reference to base 100MVA

Bus No.	Bus Type	Voltage p.u.	Power(P) p.u.	Power(Q) p.u.
1	swing	1.04		
2	PV	1.025	1.63	
3	PV	1.025	0.85	
4	PQ			
5	PQ		-1.25	-0.5
6	PQ		-0.9	-0.3
7	PQ			
8	PQ		-1.0	-0.35
9	PQ			

c.) Table 5.3 : Impedance Data of Transmission Lines

Transmission Line	Impedance (Z) p.u.	Susceptance (B/2) p.u.
TL_1	0.0100+j0.0850	j0.0880
TL_2	0.0320+j0.1610	j0.1530
TL_3	0.0085+j0.0720	j0.0745

Table 5.3 (contd.)

TL_4	0.0119+j0.1008	j0.1045
TL_5	0.0390+j0.1700	j0.1790
TL_6	0.0170+j0.0920	j0.0790

d.) **Table 5.4 : Impedance Data of Transformers**

Transformers	Impedance (Z) p.u.
T_1	j0.0576
T_2	j0.0625
T_3	j0.0586

5.1.2 Simulation Results

5.1.2.1 Steady State Operating Condition of the System

The power flow results gives the initial operating condition of the system just prior to the transient.

a.) **Table 5.5 : Power flow results**

Bus	Terminal Voltage (p.u.)	Voltage Angle (deg)	Active Power P (p.u.)	Reactive Power Q (p.u.)
1	1.0400	0	0.7164	0.2705
2	1.0250	9.2800	1.6300	0.0665
3	1.0250	4.6648	0.8500	-0.1086
4	1.0258	-2.2168	0	0
5	0.9956	-3.9888	-1.2500	-0.5000
6	1.0127	-3.6874	-0.9000	-0.3000
7	1.0258	3.7197	0	0
8	1.0159	0.7275	-1.0000	-0.3500
9	1.0324	1.9667	0	0

Note : + sign indicates the power flow towards the bus and – sign indicates the power flow from the bus.

b.) **Table 5.6 : Initial operating condition of generators**

Generator	Terminal Voltage		Armature Current		Power factor	Internal Voltage	
	Mag./p.u.	Ang./deg.	Mag./p.u.	Ang./deg.		Mag./p.u.	Ang./deg.
1	1.0400	0	0.7363	-20.6825	0.9355	1.0932	4.7849
2	1.0250	9.2800	1.5916	6.9425	0.9992	1.8580	62.6299
3	1.0250	4.6648	0.8360	11.9455	0.9919	1.4465	55.9984

5.1.2.2 System Network Representation

a.) Table 5.7 : Y-matrix of the 9-bus system (without considering generator synchronous reactance)

Bus	1	2	3	4	5	6	7	8	9
1	0 -17.3611i	0	0	0 +17.3611i	0	0	0	0	0
2	0	0 -16.0000i	0	0	0	0	0 +16.0000i	0	0
3	0	0	0 -17.0648i	0	0	0	0	0	0 +17.0648i
4	0 +17.3611i	0	0	3.3074 -39.3089i	-1.3652 +11.6041i	-1.9422 +10.5107i	0	0	0
5	0	0	0	-1.3652 +11.6041i	2.5528 -17.3382i	0	-1.1876 + 5.9751i	0	0
6	0	0	0	-1.9422 +10.5107i	0	3.2242 -15.8409i	0	0	-1.2820 + 5.5882i
7	0	0 +16.0000i	0	0	-1.1876 + 5.9751i	0	2.8047 -35.4456i	-1.6171 +13.6980i	0
8	0	0	0	0	0	0	-1.6171 +13.6980i	2.7722 -23.3032i	-1.1551 + 9.7843i
9	0	0	0 +17.0648i	0	0	-1.2820 + 5.5882i	0	-1.1551 + 9.7843i	2.4371 -32.1539i

5.1.2.3 Transient Stability Analysis of the System (Fault at bus 7 in TL2)

a.) Reduced Y-matrices

Table 5.8 : Reduced Y-matrix for pre-fault network
(including generator transient reactance)

Bus	1	2	3
1	0.8455 - 2.9883i	0.2871 + 1.5129i	0.2096 + 1.2256i
2	0.2871 + 1.5129i	0.4200 - 2.7239i	0.2133 + 1.0879i
3	0.2096 + 1.2256i	0.2133 + 1.0879i	0.2770 - 2.3681i

Table 5.9 : Reduced Y-matrix for faulty network
(with fault at bus 7 and section 7-5)

Bus	1	2	3
1	0.6568 - 3.8160i	0	0.0701 + 0.6306i
2	0	0 - 5.4855i	0
3	0.0701 + 0.6306i	0	0.1740 - 2.7959i

Table 5.10 : Reduced Y-matrix for post-fault network

Bus	1	2	3
1	1.1386 - 2.2966i	0.1290 + 0.7063i	0.1824 + 1.0637i
2	0.1290 + 0.7063i	0.3744 - 2.0151i	0.1921 + 1.2067i
3	0.1824 + 1.0637i	0.1921 + 1.2067i	0.2691 - 2.3516i

b.) Transient response of the Faulty system

The transient response of electrical power generated, speed, load angle of each generator is analyzed for following cases.

1. without considering machine damping
2. considering machine damping
3. by increasing machine inertia

1.) Transient Response without considering damping

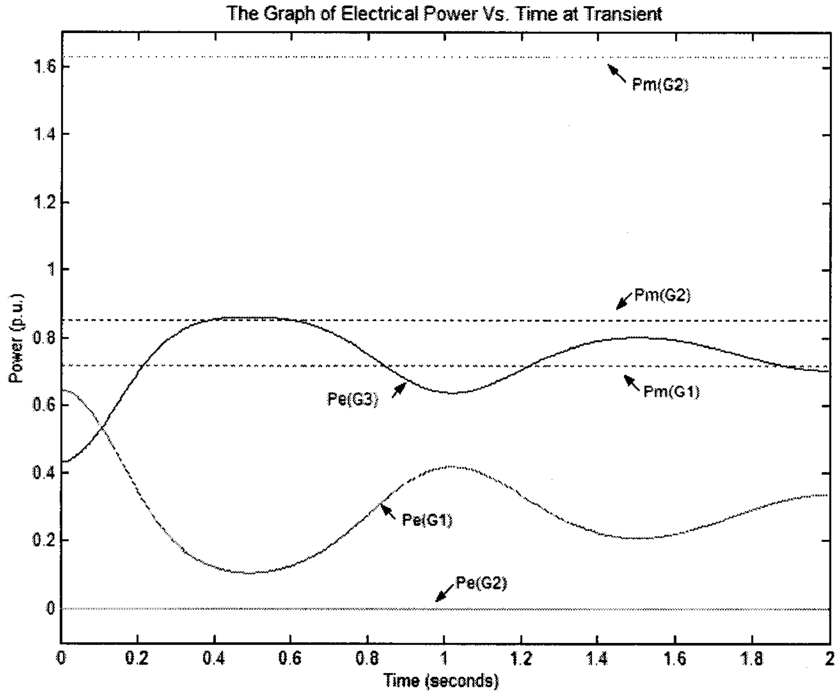


Figure 5.2 : Transient response of electrical power generated

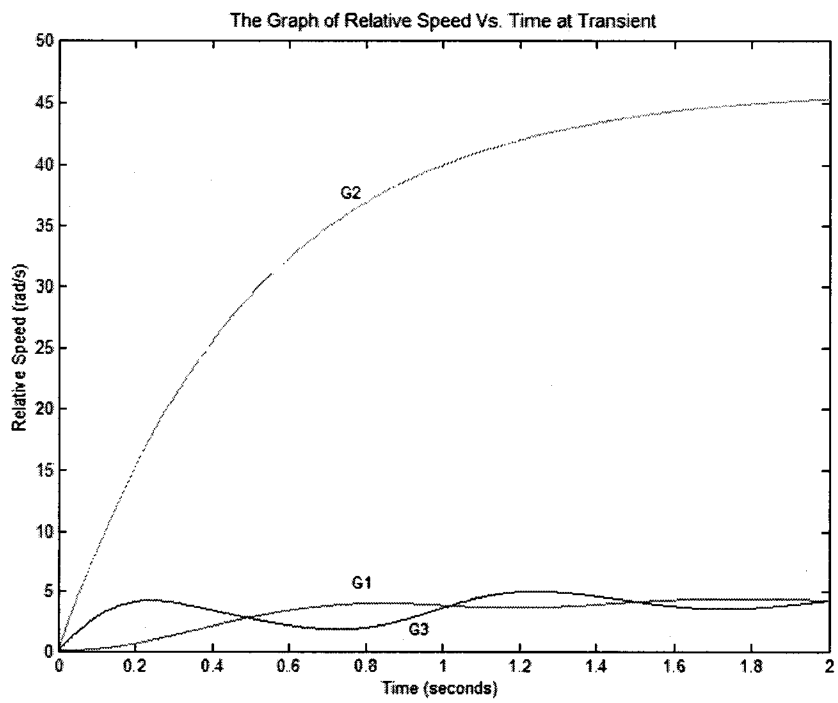


Figure 5.3 : Transient response of generator speed (faulty system)

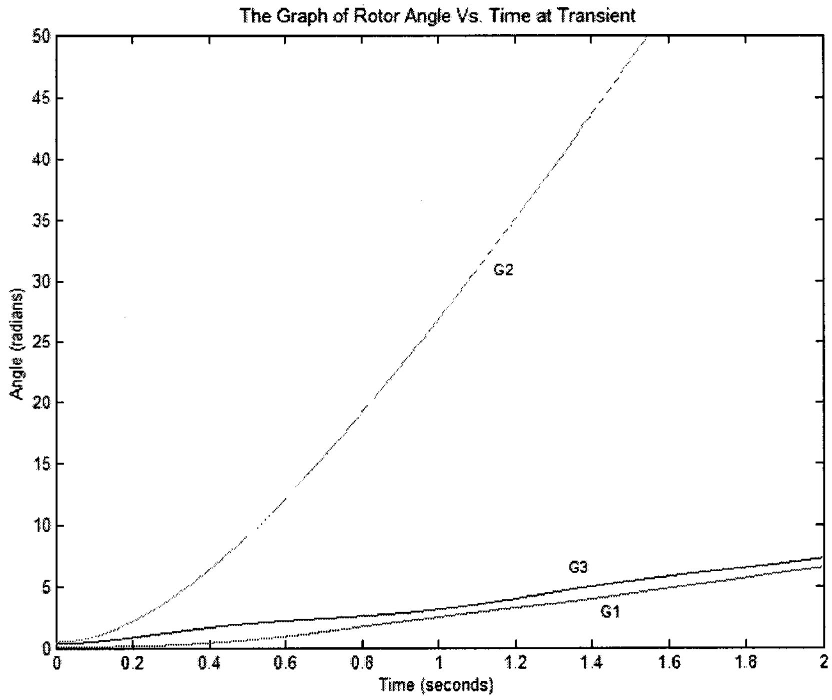


Figure 5.4 : Transient response of generator load angle (faulty system)

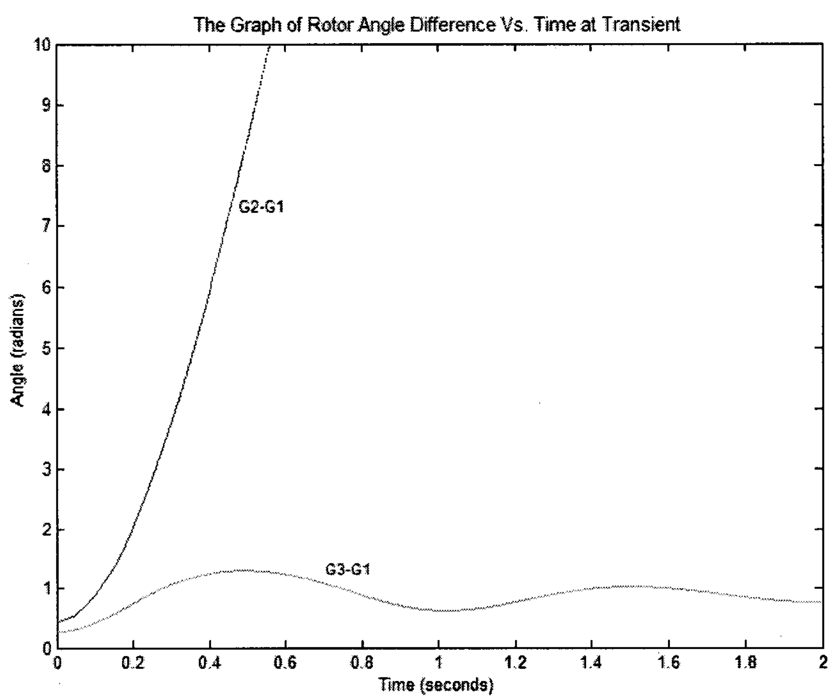


Figure 5.5 : Transient response of generator load angle difference (faulty system)

2.) Transient Response considering damping (D=2)

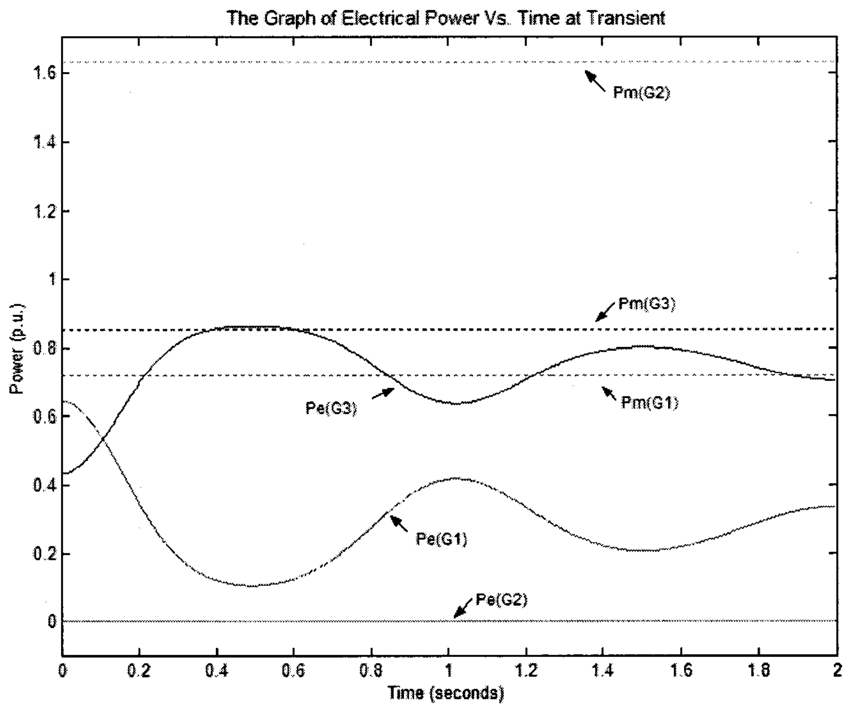


Figure 5.6 : Transient response of electrical power generated

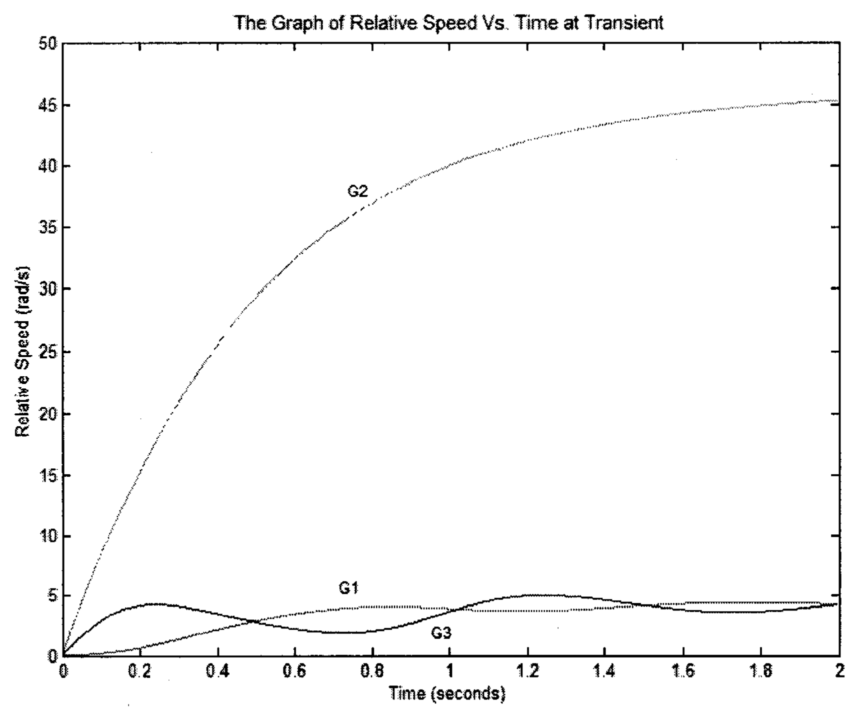


Figure 5.7 : Transient response of generator speed (faulty system)

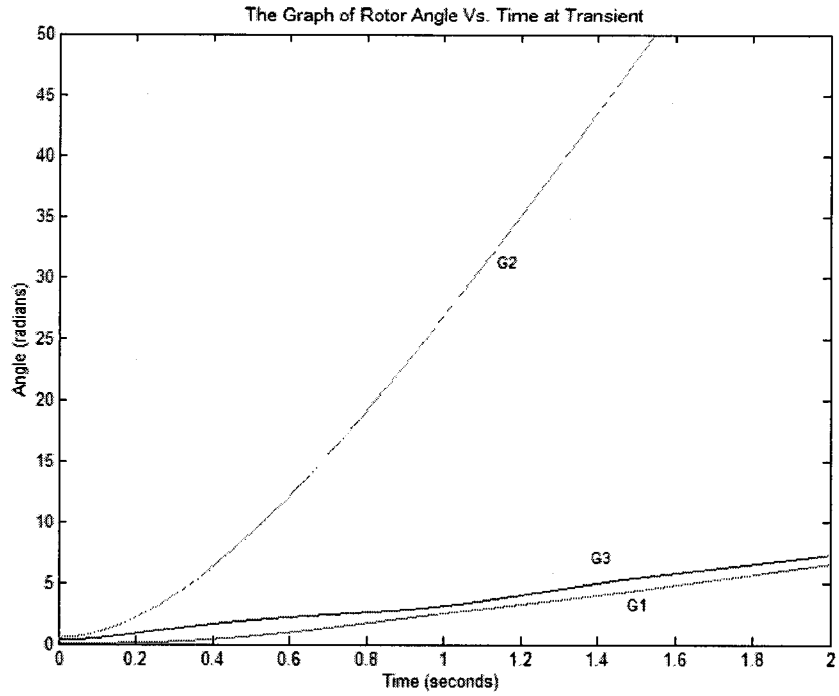


Figure 5.8 : Transient response of generator load angle (faulty system)

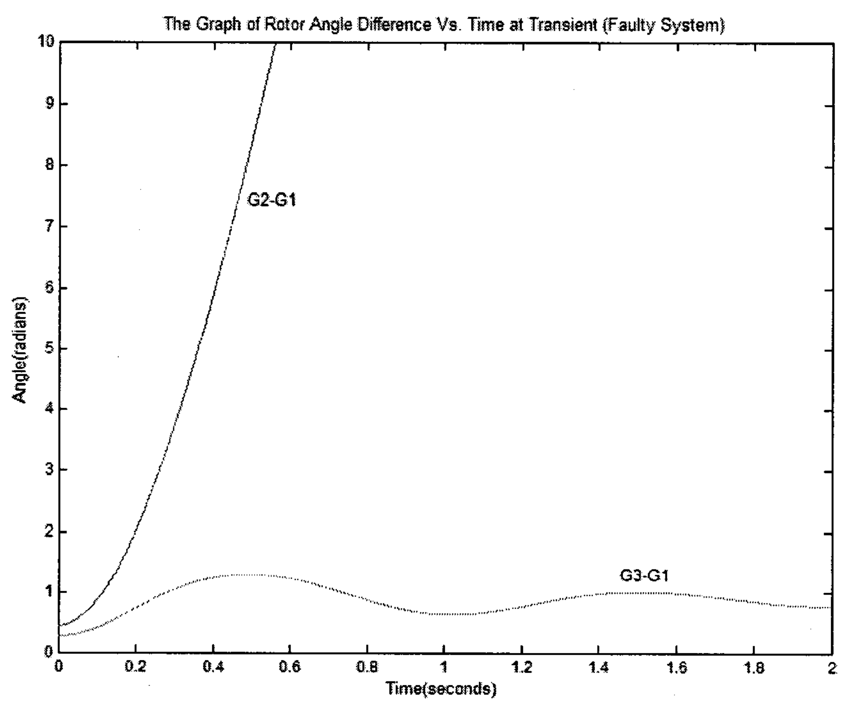


Figure 5.9 : Transient response of generator load angle difference (faulty system)

3.) Transient Response with increased inertia (by 10%)

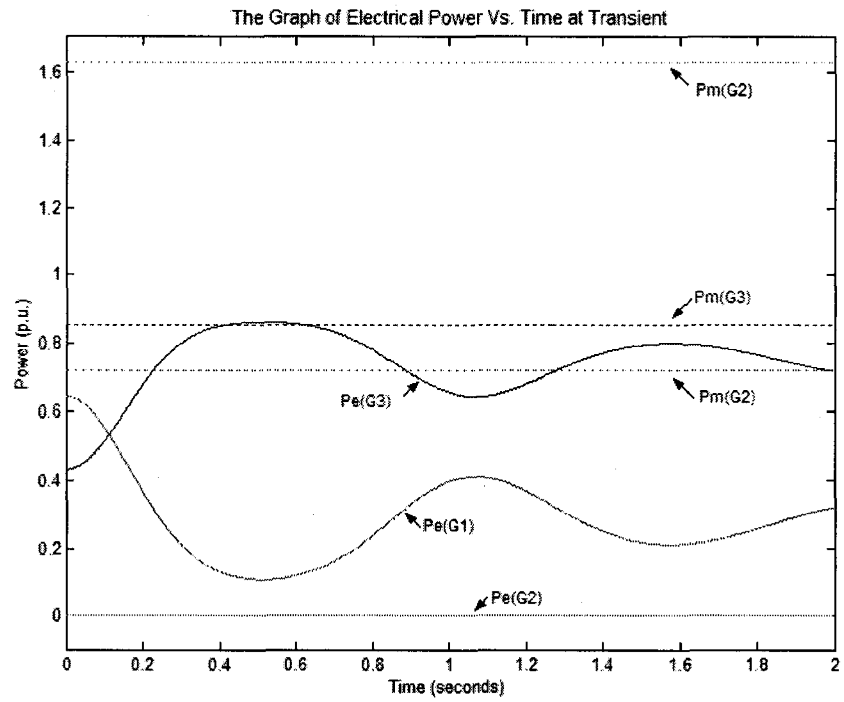


Figure 5.10 : Transient response of electrical power generated

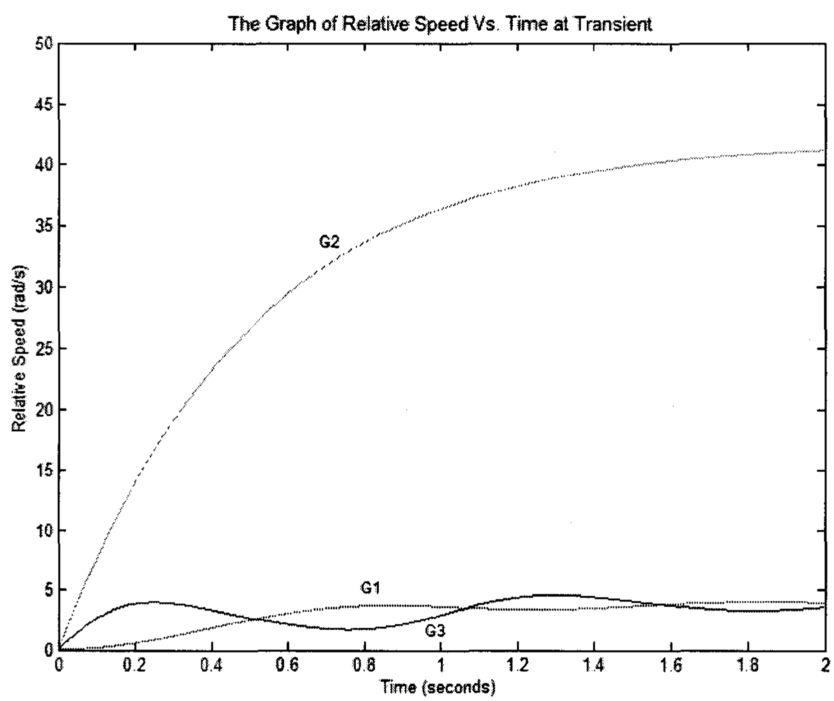


Figure 5.11 : Transient response of generator speed (faulty system)

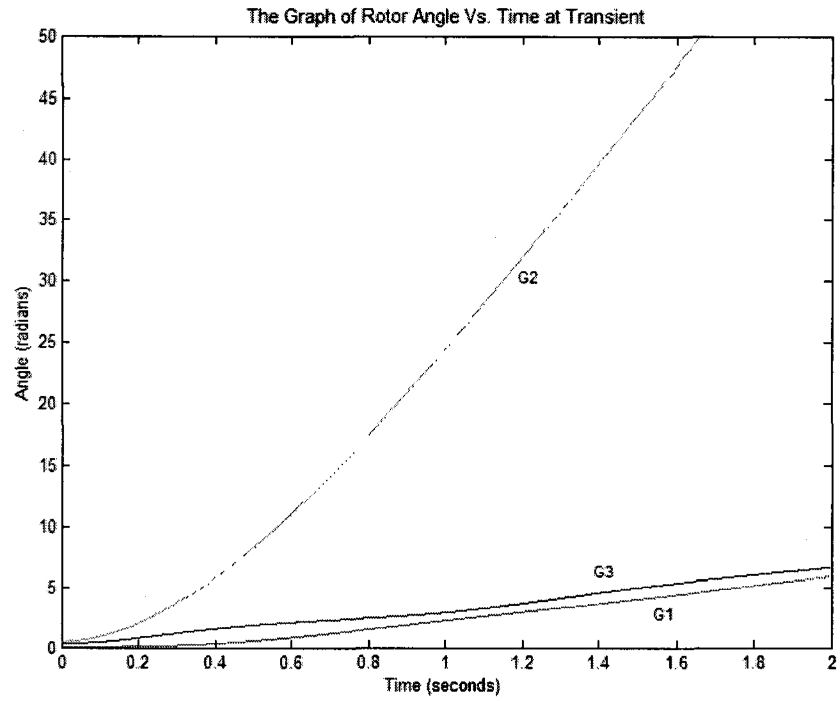


Figure 5.12 : Transient response of generator load angle (faulty system)

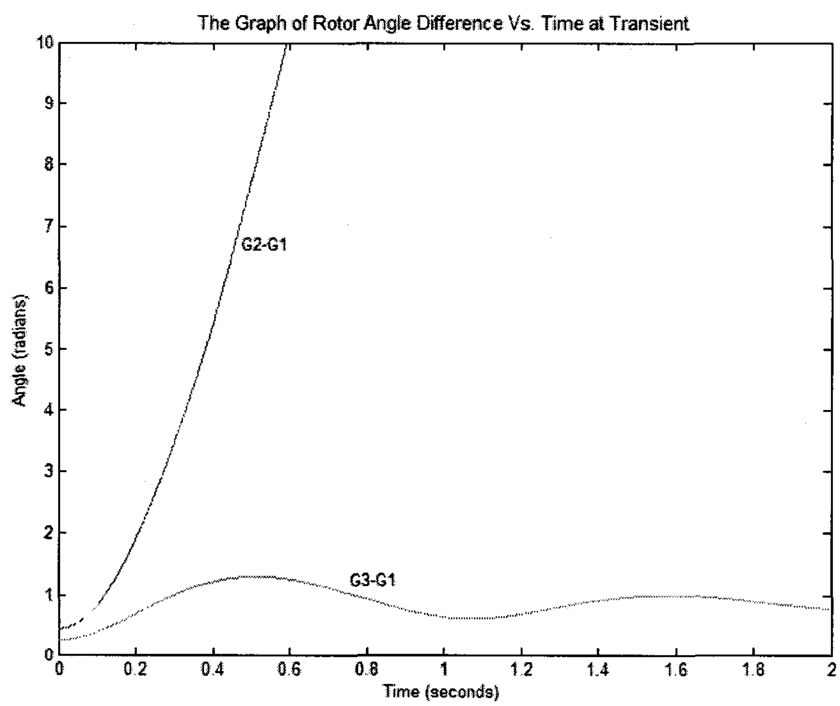


Figure 5.13 : Transient response of generator load angle difference (faulty system)

c.) **Transient response of the system (Fault Cleared)**

The transient response of the system with fault clearing at 3 cycles (0.1s) is analyzed for all three cases.

1.) **Transient Response without considering damping (D=0)**

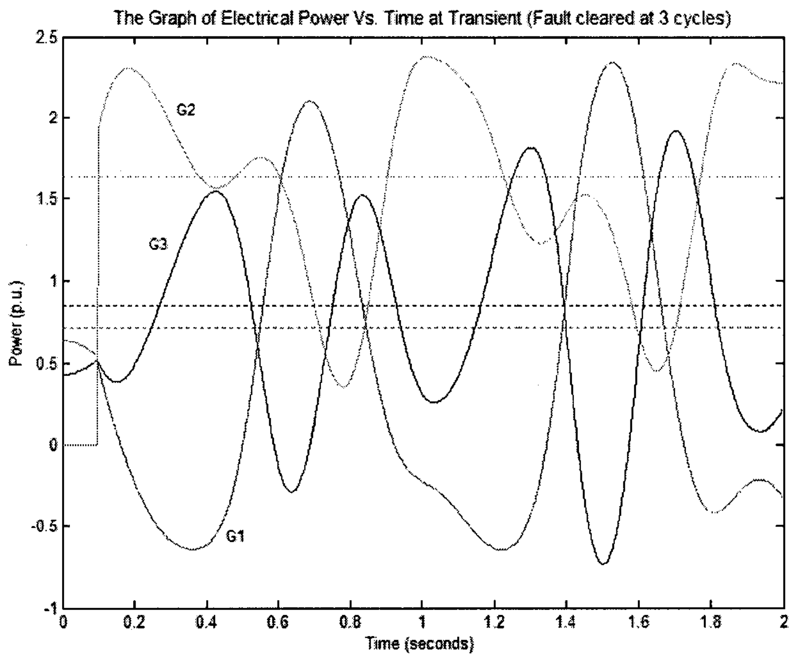


Figure 5.14 : Transient response of electrical power generated

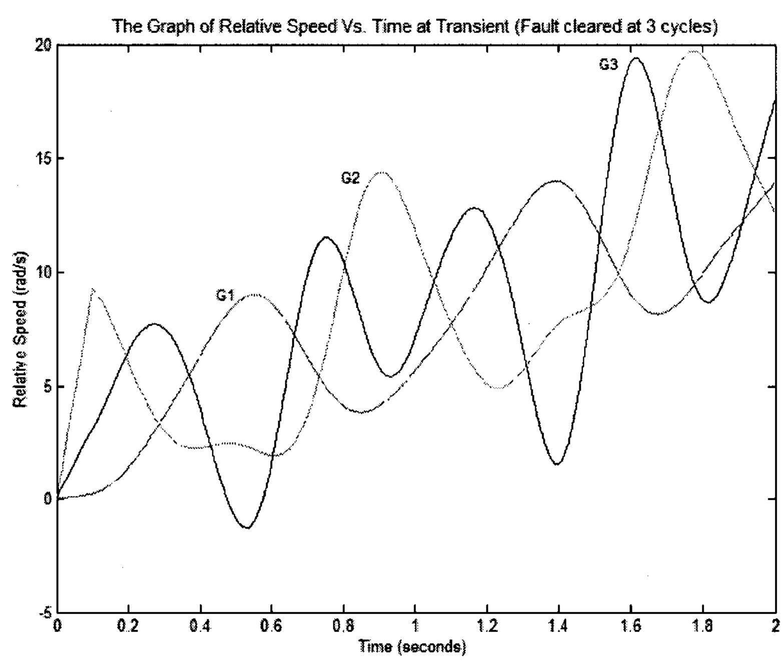


Figure 5.15 : Transient response of generator speed (fault cleared)

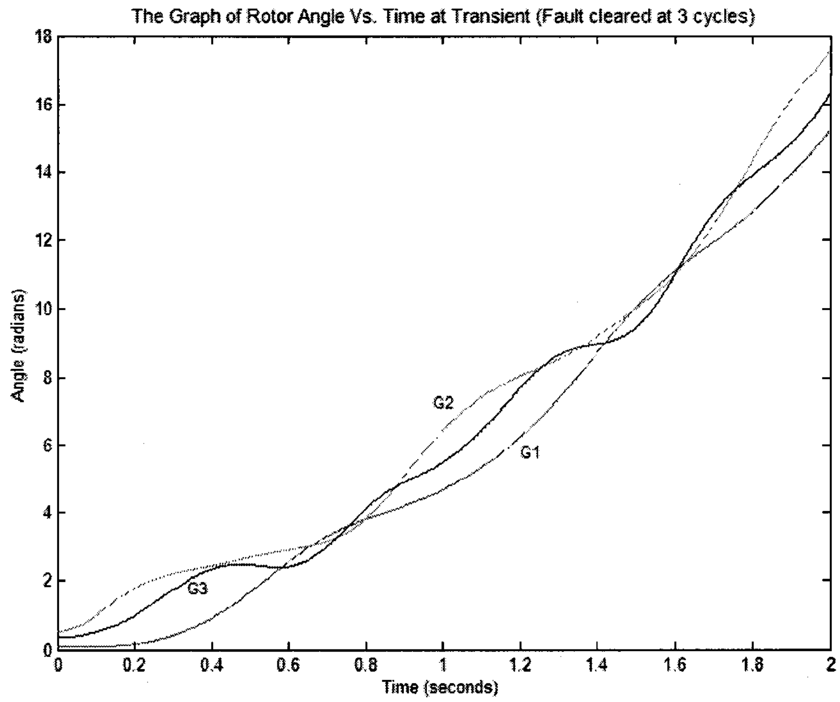


Figure 5.16 : Transient response of generator load angle (fault cleared after 3 cycles)

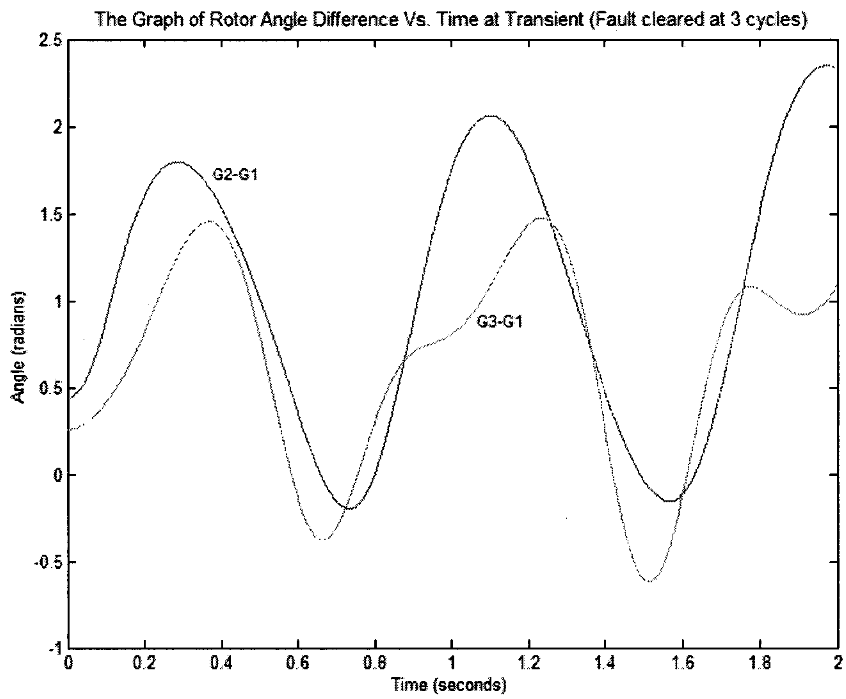


Figure 5.17 : Transient response of generator load angle difference (fault cleared after 3 cycles)

2.) Transient Response considering damping (D=2)

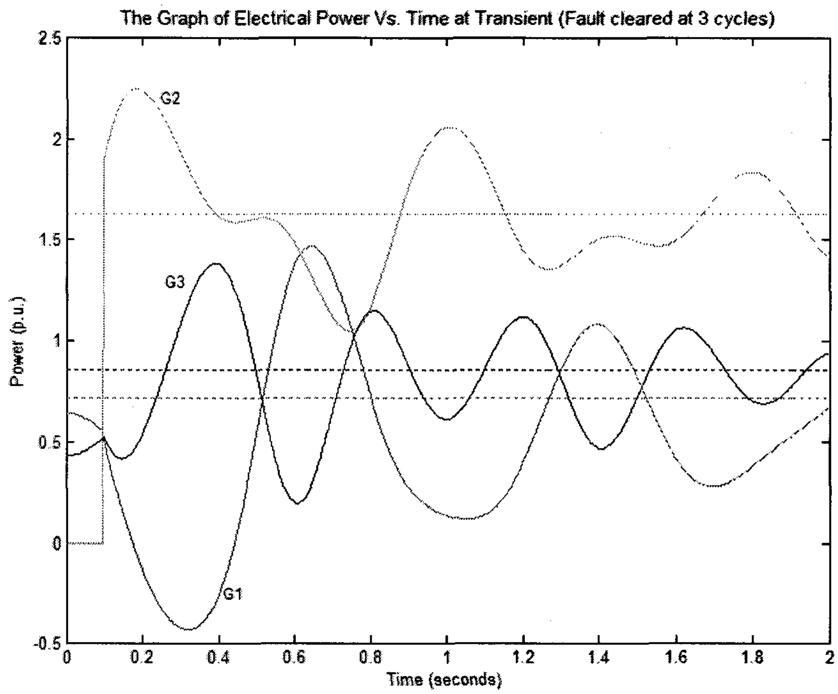


Figure 5.18 : Transient response of electrical power generated

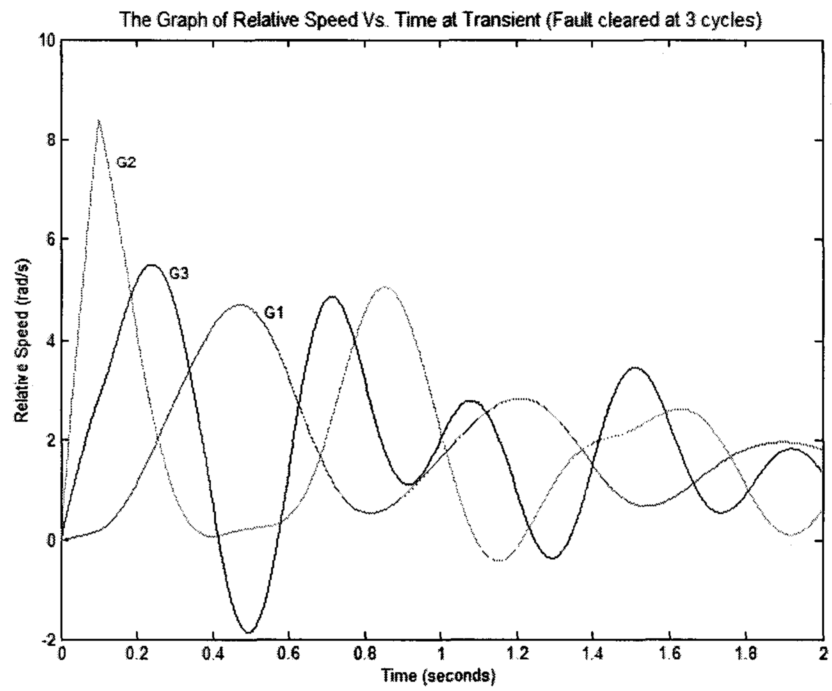


Figure 5.19 : Transient response of generator speed (fault cleared)

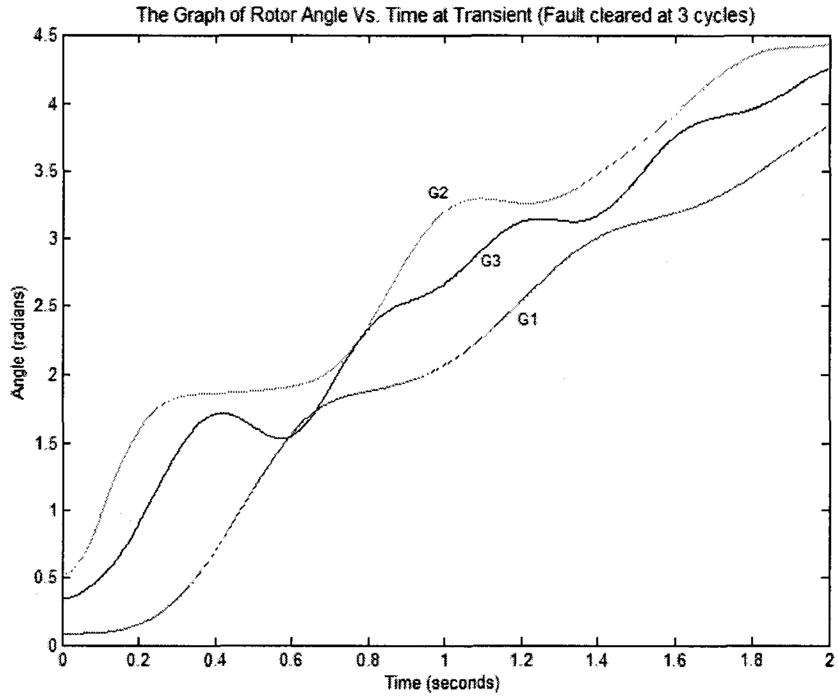


Figure 5.20 : Transient response of generator load angle (fault cleared after 3 cycles)

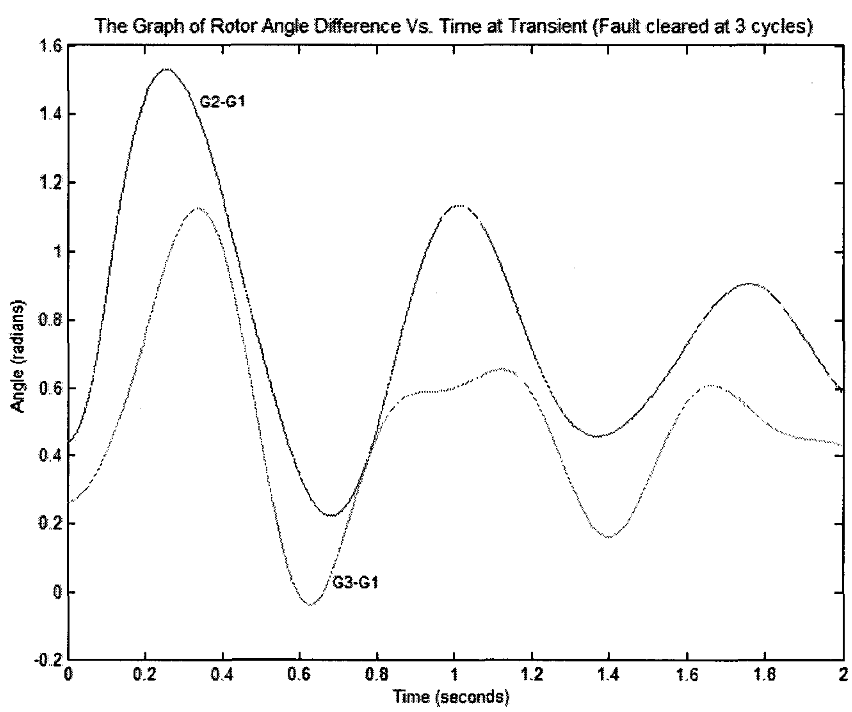


Figure 5.21 : Transient response of generator load angle difference (fault cleared after 3 cycles)

5.1.2.4 Transient Stability Analysis of the System (Fault at bus 5 in TL2)

The transient response under the fault at a load bus is analyzed to see the behavior of the generators when the fault is away from them.

a.) Reduced Y-matrices

Table 5.11 : Reduced Y-matrix for Pre-fault network
(including generator transient reactance)

Bus	1	2	3
1	0.8455 - 2.9883i	0.2871 + 1.5129i	0.2096 + 1.2256i
2	0.2871 + 1.5129i	0.4200 - 2.7239i	0.2133 + 1.0879i
3	0.2096 + 1.2256i	0.2133 + 1.0879i	0.2770 - 2.3681i

Table 5.12 : Reduced Y-matrix for faulty network
(with fault at bus 5 and section 7-5)

Bus	1	2	3
1	0.3241 - 5.2974i	0.0300 + 0.2154i	0.0443 + 0.4971i
2	0.0300 + 0.2154i	0.2956 - 3.4525i	0.1317 + 0.6785i
3	0.0443 + 0.4971i	0.1317 + 0.6785i	0.2246 - 2.5980i

Table 5.13 : Reduced Y-matrix for post fault network
(with section 5-7 cleared)

Bus	1	2	3
1	1.1386 - 2.2966i	0.1290 + 0.7063i	0.1824 + 1.0637i
2	0.1290 + 0.7063i	0.3744 - 2.0151i	0.1921 + 1.2067i
3	0.1824 + 1.0637i	0.1921 + 1.2067i	0.2691 - 2.3516i

The Y-matrix of the post-fault network is similar for both cases which the fault is at bus 5 and fault is at bus 7. In both cases section 5-7 is clared.

b.) Transient response of the Faulty system

The transient response of electrical power generated, speed, load angle of each generator is analyzed for following cases.

1. without considering machine damping
2. considering machine damping

1.) Transient Response without considering damping (D=0)

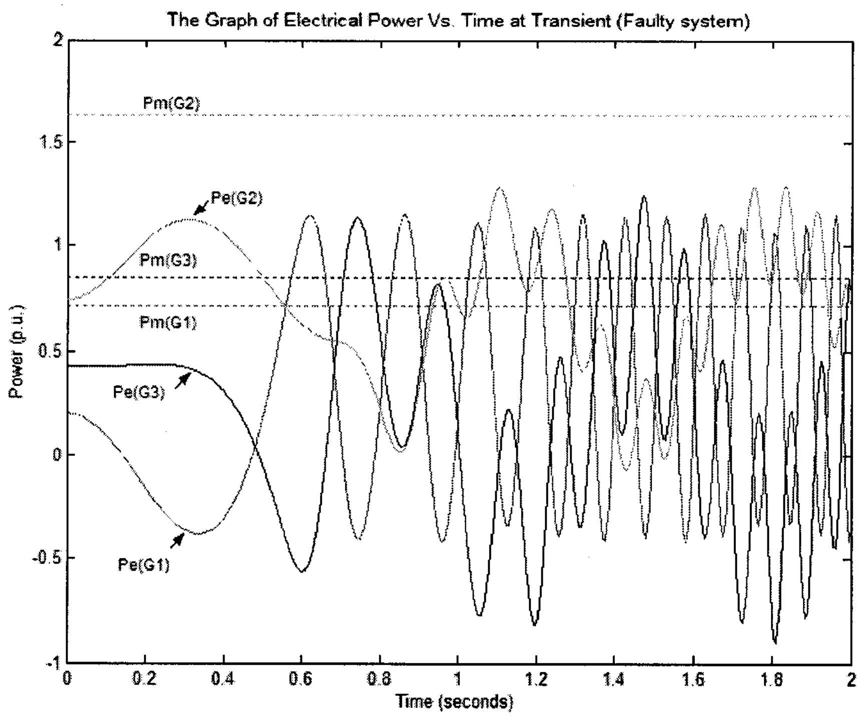


Figure 5.22 : Transient response of electrical power generated

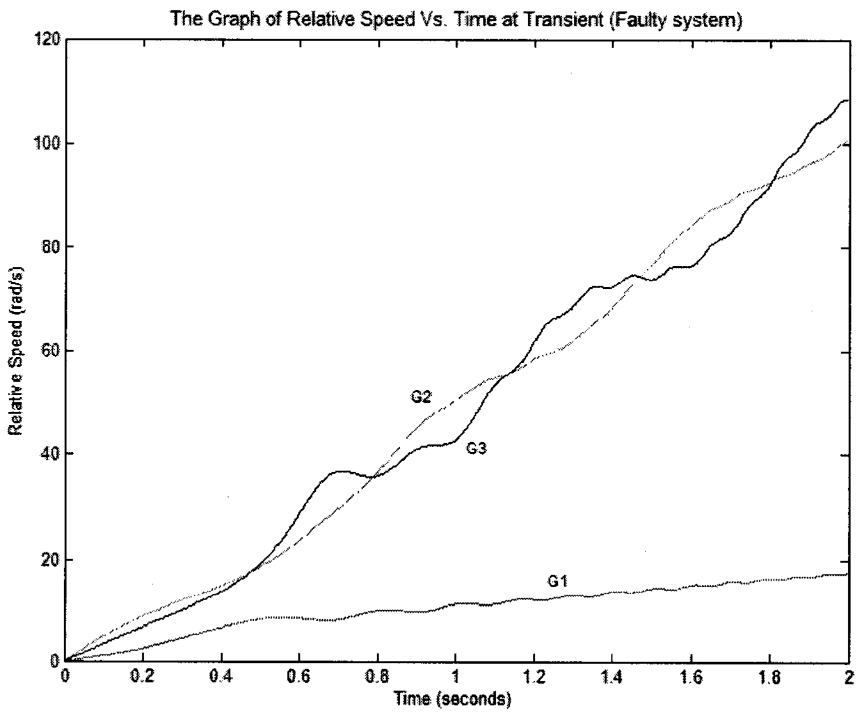


Figure 5.23 : Transient response of generator speed

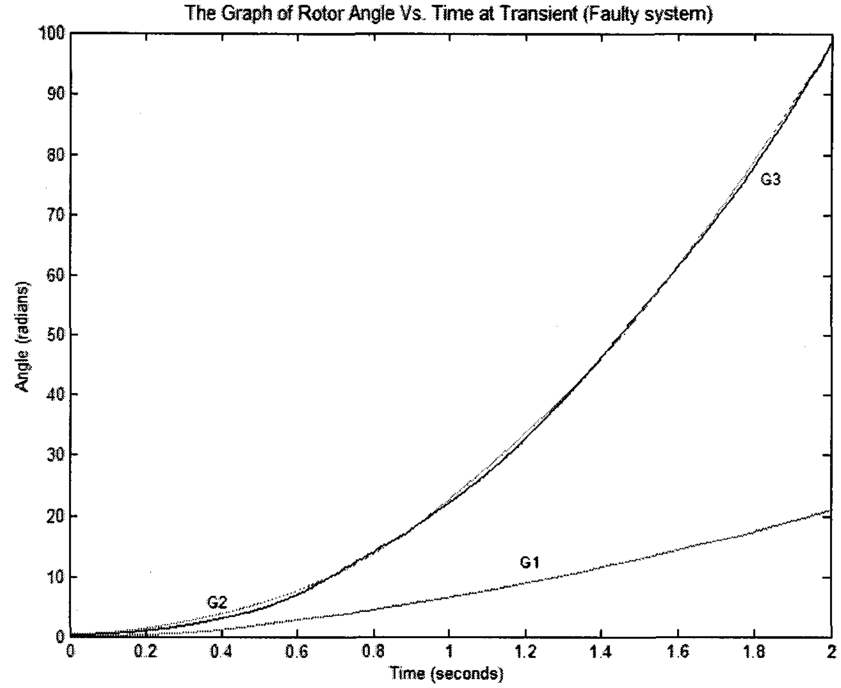


Figure 5.24 : Transient response of generator load angle

2.) Transient Response considering damping (D=2)

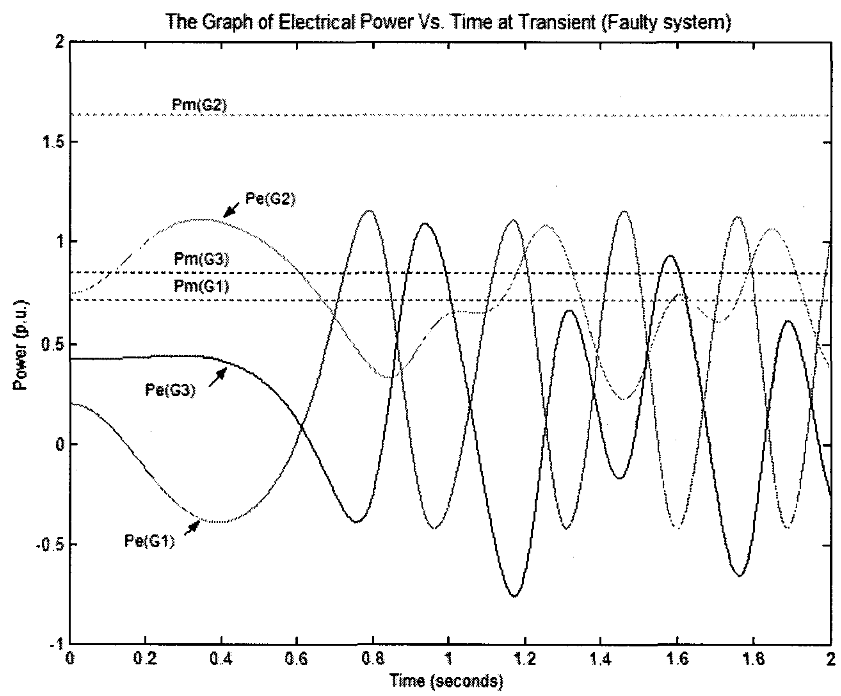


Figure 5.25 : Transient response of electrical power generated

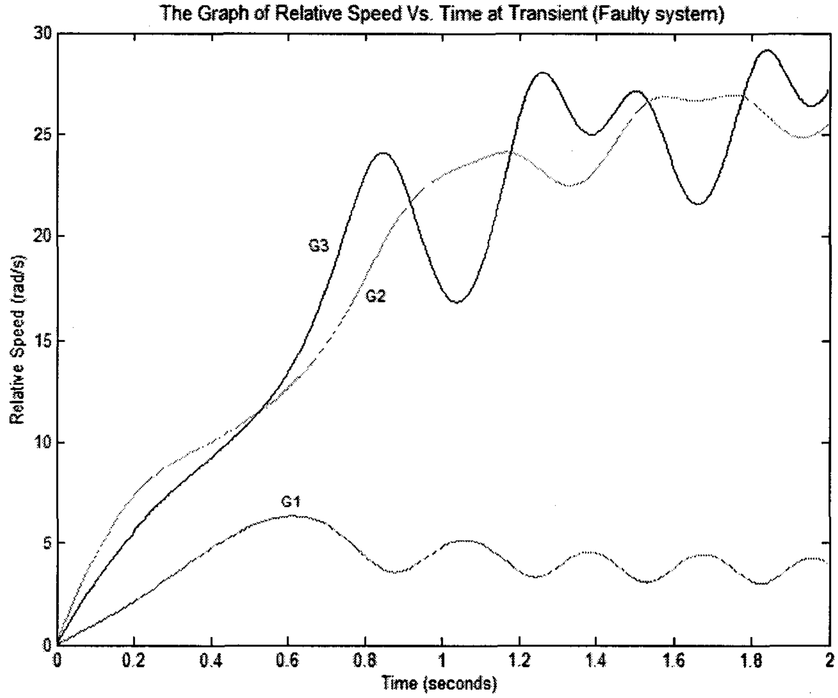


Figure 5.26 : Transient response of generator speed

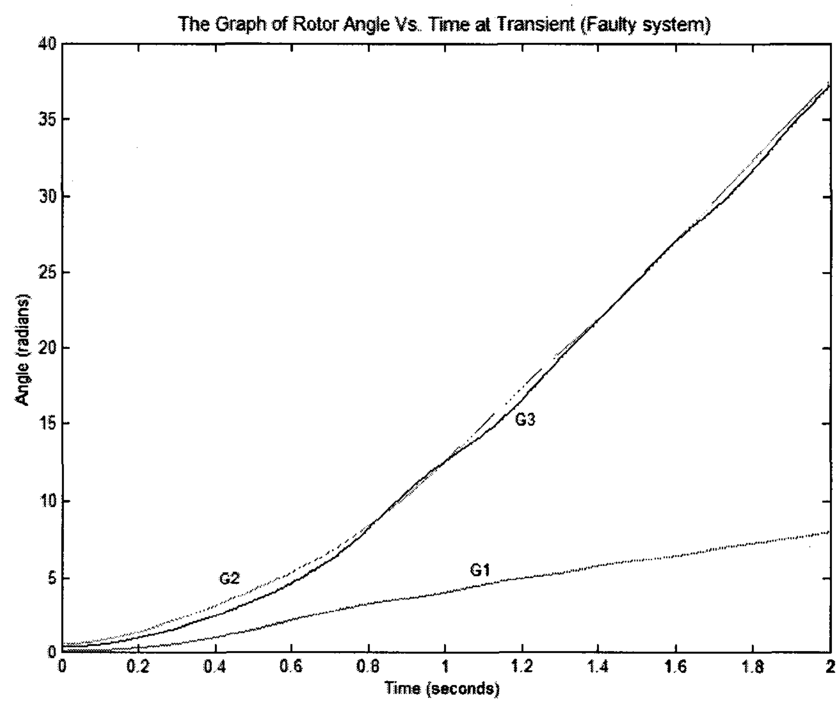


Figure 5.27 : Transient response of generator load angle

c.) **Transient response of the system (Fault Cleared)**

The transient response of the system with fault clearing at 3 cycles (0.1s)

1.) **Transient Response without considering damping (D=0)**

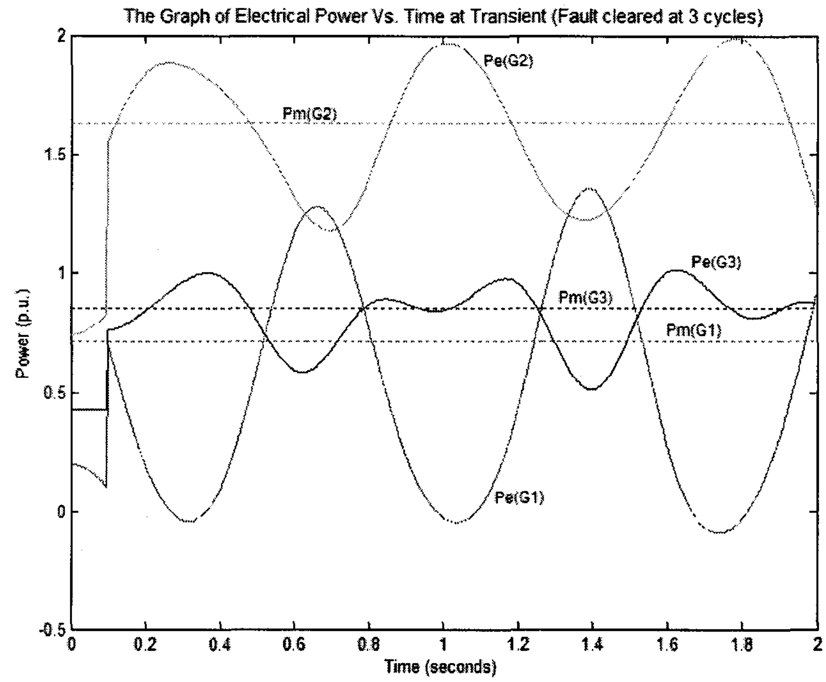


Figure 5.28 : Transient response of electrical power generated

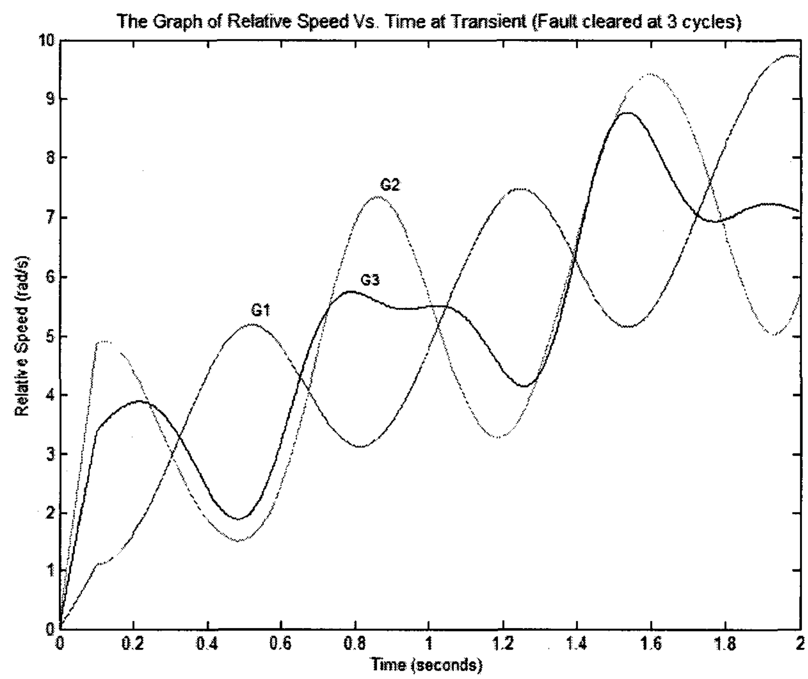


Figure 5.29 : Transient response of generator speed

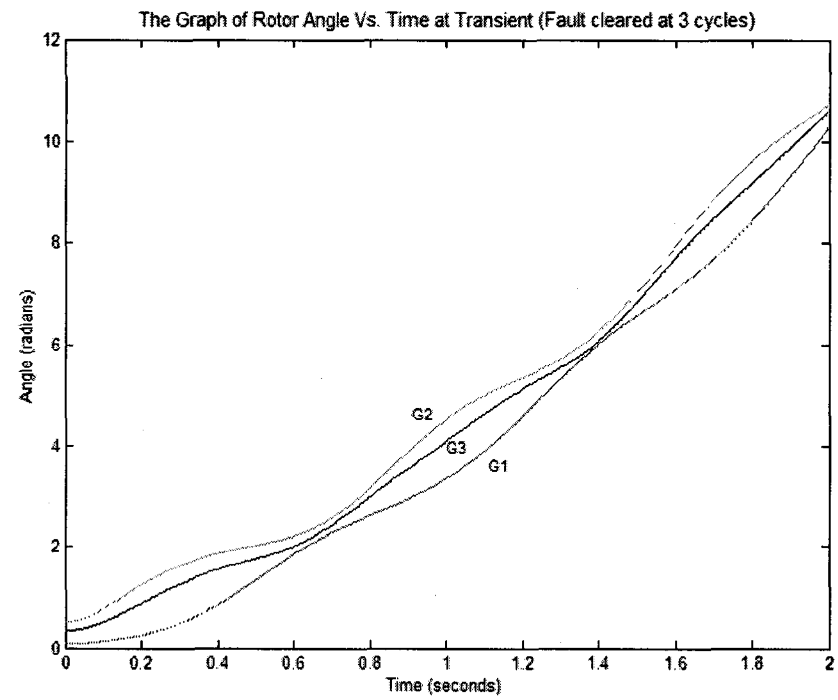


Figure 5.30 : Transient response of generator load angle

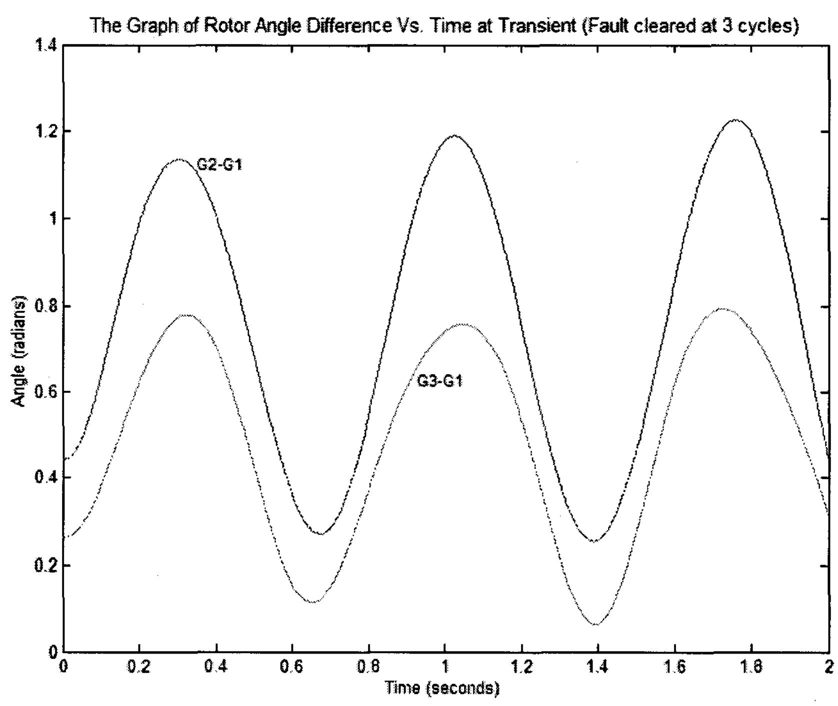


Figure 5.31 : Transient response of generator load angle difference

2.) Transient Response considering damping (D=2)

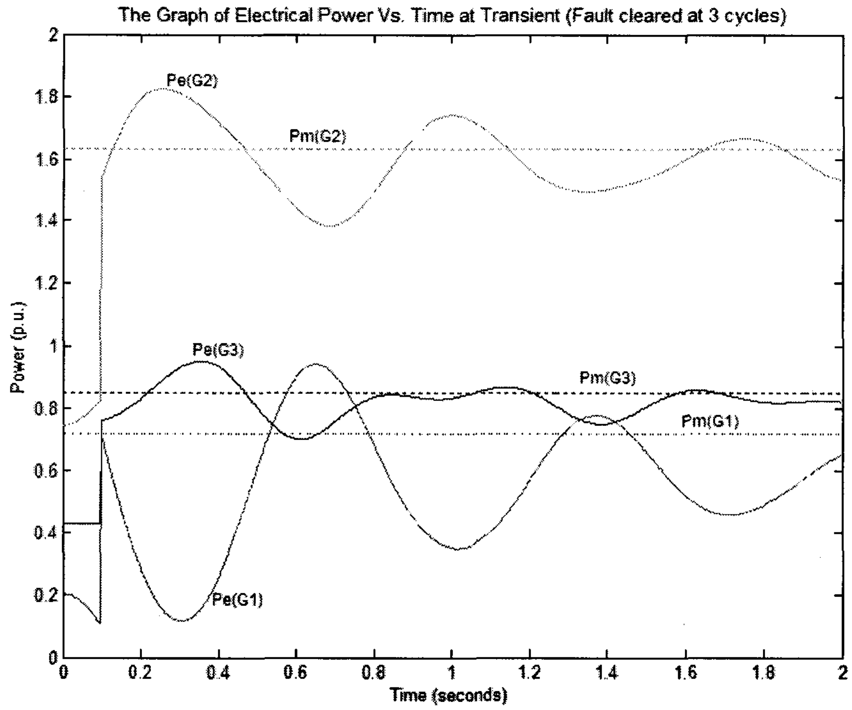


Figure 5.32: Transient response of electrical power generated

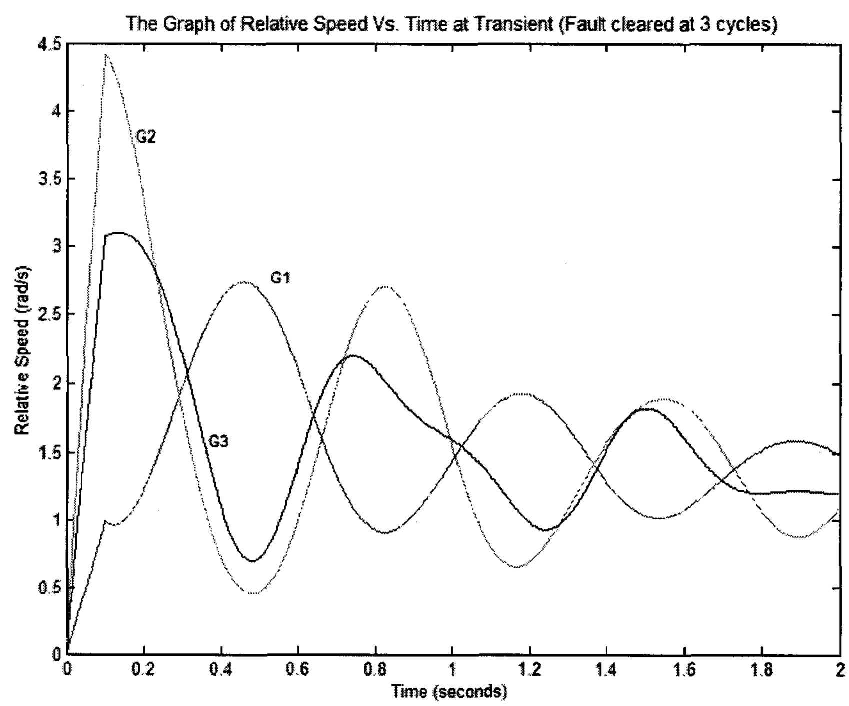


Figure 5.33 : Transient response of generator speed

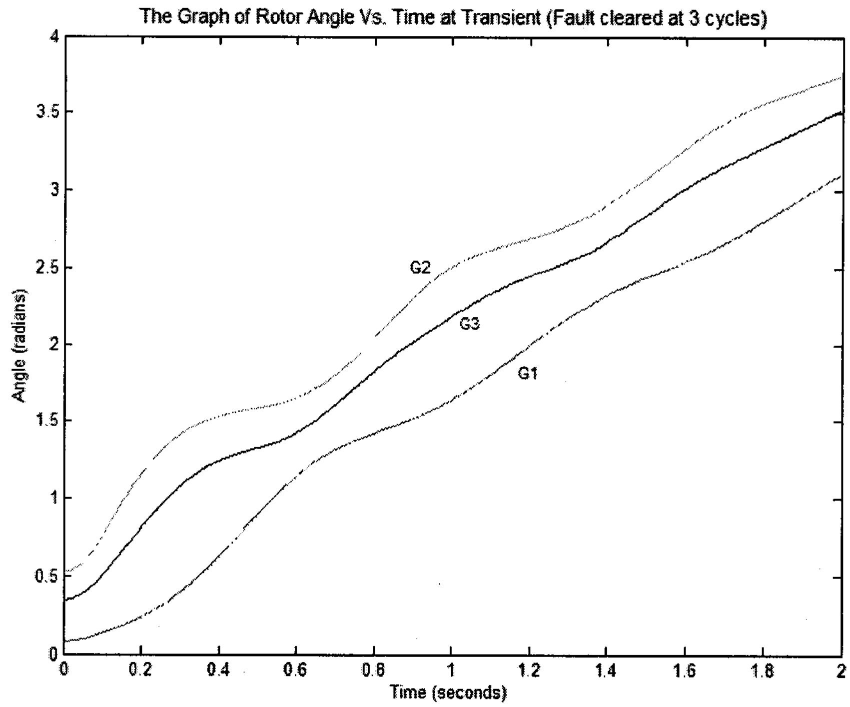


Figure 5.34 : Transient response of generator load angle

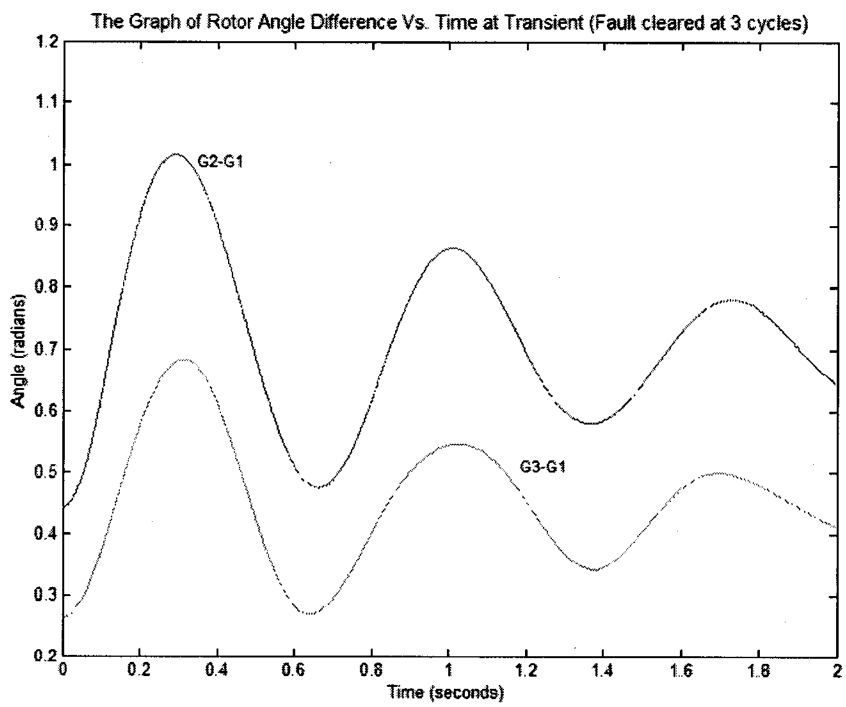


Figure 5.35 : Transient response of generator load angle difference

5.1.2.5 Critical Clearing Time

If the system fault is cleared beyond the critical clearing time, the system tends to instability. The critical clearing time for above each case is calculated to compare their degree of instability.

Table 5.14 : Critical Clearing Time

Position of Fault	Description	Critical clearing time (t_{cr})
Bus 7 in TL2	D = 0	0.130 seconds
	D = 2	0.1635 seconds
	D = 2 and with increase of inertia	0.1730 seconds
Bus 5 in TL2	D = 0	0.3105 seconds
	D = 2	0.4330 seconds

5.2 15-bus System

The configuration of the 15-bus system were taken from reference [13].

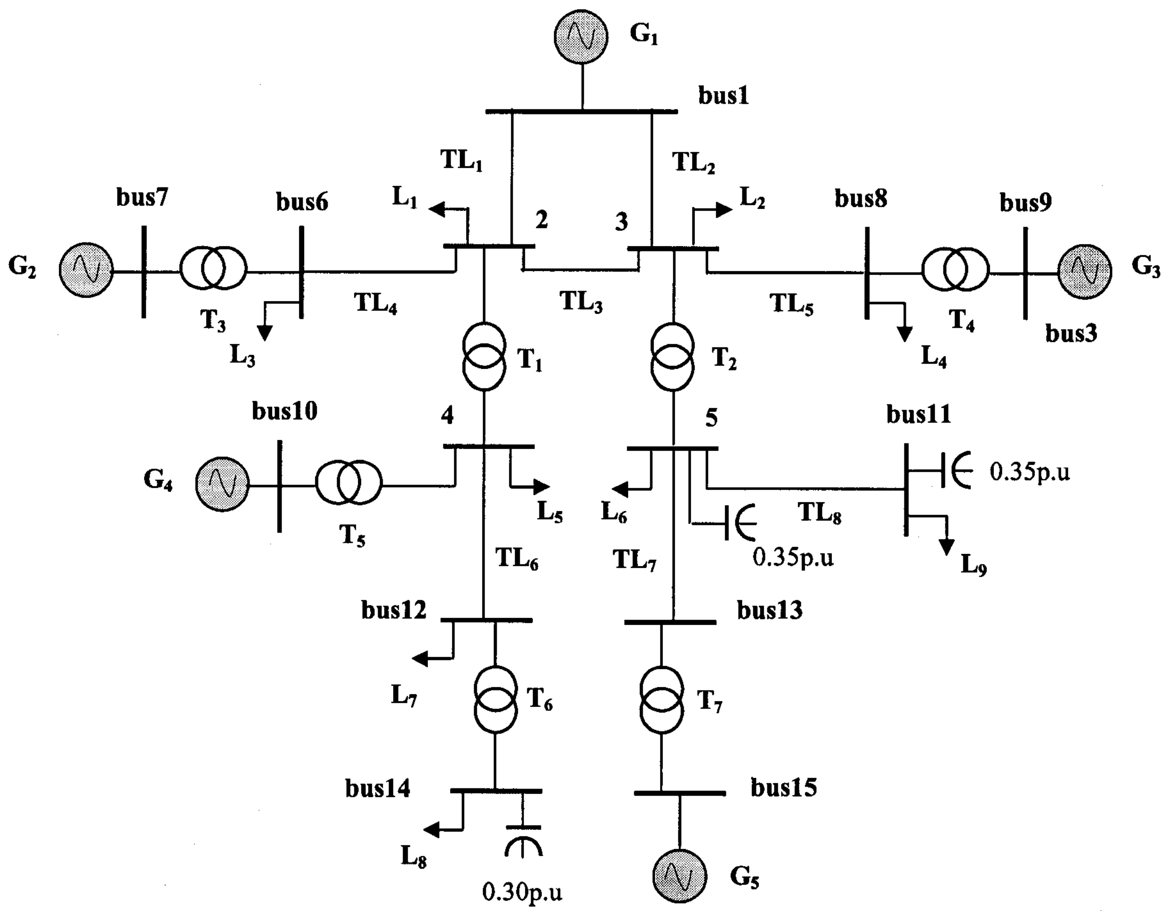


Figure 5.36 – Configuration of Fifteen Bus System

5.2.1 System Data

a.) Table 5.15 : Generator Data

Generator	G_1	G_2	G_3	G_4	G_5
Rated MVA	125	75	100	125	75
kV	15.5	13.8	13.8	15.5	13.8
Power factor	0.85	0.80	0.80	0.85	0.80
Type	steam	steam	steam	steam	steam
Speed (r.p.m)	3600	3600	3600	3600	3600
X_d	1.220	1.050	1.180	1.220	1.050
X'_d	0.174	0.185	0.220	0.174	0.185
X_q	1.160	0.980	1.050	1.160	0.980

Table 5.15 (contd.)

X'_q	0.250	0.360	0.380	0.250	0.360
X_l	0.078	0.070	0.075	0.078	0.070
T'_{do}	8.97	6.10	5.90	8.97	6.10
T'_{qo}	0.50	0.30	0.30	0.50	0.30
H (MJ/MVA)	4.75	6.15	4.98	4.75	6.15

b.) Table 5.16 : Transformer Data

Transformer	**T_1**	**T_2**	**T_3**	**T_4**	**T_5**	**T_6**	**T_7**
Rated MVA	100	100	60	100	120	40	80
LV voltage (kV)	138	138	13.8	13.8	13.8	44	13.8
HV voltage (kV)	230	230	230	230	138	138	138
Phase	0	0	0	0	0	0	0
Z	0.1	0.1	0.1	0.1	0.11	0.08	0.09
X/R ratio	34.1	34.1	34.1	34.1	42	27.3	34.1
Tap position	1.02	1.02	1.03	1.04	1.02	1.01	1.05

c.) Table 5.17 : Loading Data

S (reference) = 100MVA

Bus No.	**Bus Type**	**Voltage p.u.**	**Power(P) p.u.**	**Power(Q) p.u.**
1	swing	1.04		
2	PV		-0.32	-0.39
3	PQ		-0.40	-0.30
4	PQ		-0.40	-0.30
5	PQ		-0.60	-0.15
6	PQ		-0.30	-0.25
7	PV	1.01	0.6	
8	PQ		-0.60	-0.40
9	PV	1.01	0.8	
10	PV	1.04	1.0	
11	PQ		-0.60	-0.05
12	PQ		-0.40	-0.30
13	PQ			
14	PQ		-0.40	0.10
15	PV	1.04	0.60	

d.) Table 5.18 : Transmission Line Impedance Data

Transmission Line	Impedance (Z) p.u.	Susceptance (B/2) p.u.
TL_1	0.0275+j0.1512	j0.13965
TL_2	0.0275+j0.1512	j0.13965
TL_3	0.0413+j0.2268	j0.2094
TL_4	0.0157+j0.0080	j0.0698
TL_5	0.0157+j0.0080	j0.0698
TL_6	0.0224+j0.1181	j0.01505
TL_7	0.0224+j0.1181	j0.01505
TL_8	0.0373+j0.1968	j0.02515

5.2.2 Simulation Results

The transient response of the 15-bus system is also analyzed as 9-bus system and most significant results for the 15-bus system are mentioned in this thesis.

5.2.2.1 Transient Response of the System (Fault at bus 8)

In this case, it is considered that the fault is at bus 8 in transmission line between bus 8 and bus 3.

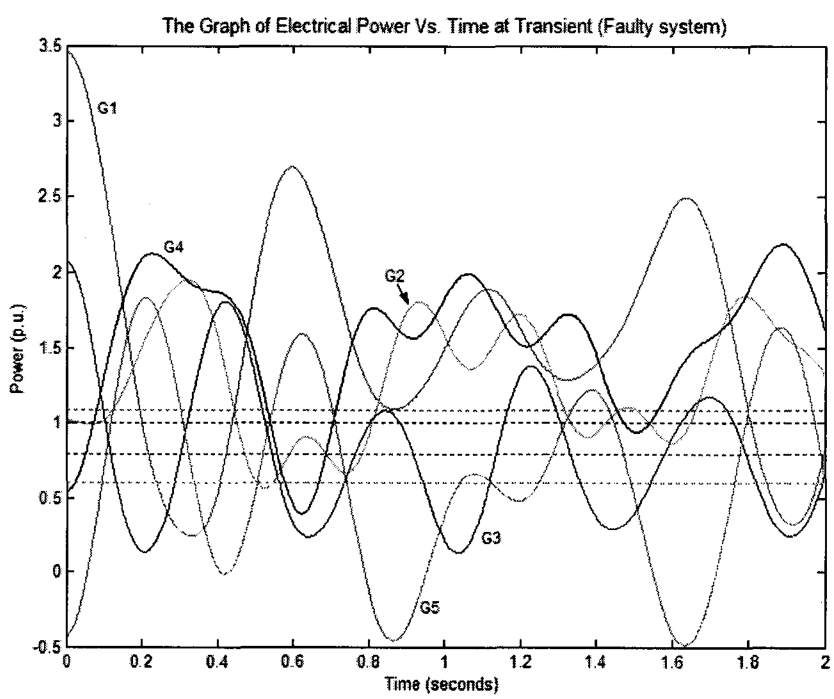


Figure 5.37 : Transient response of electrical power generated

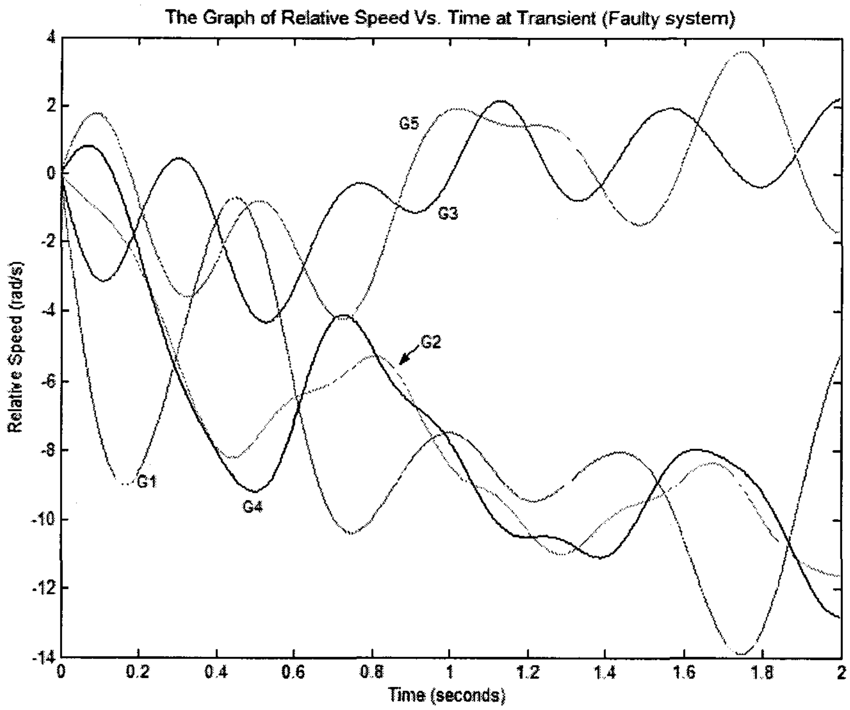


Figure 5.38 : Transient response of generator speed

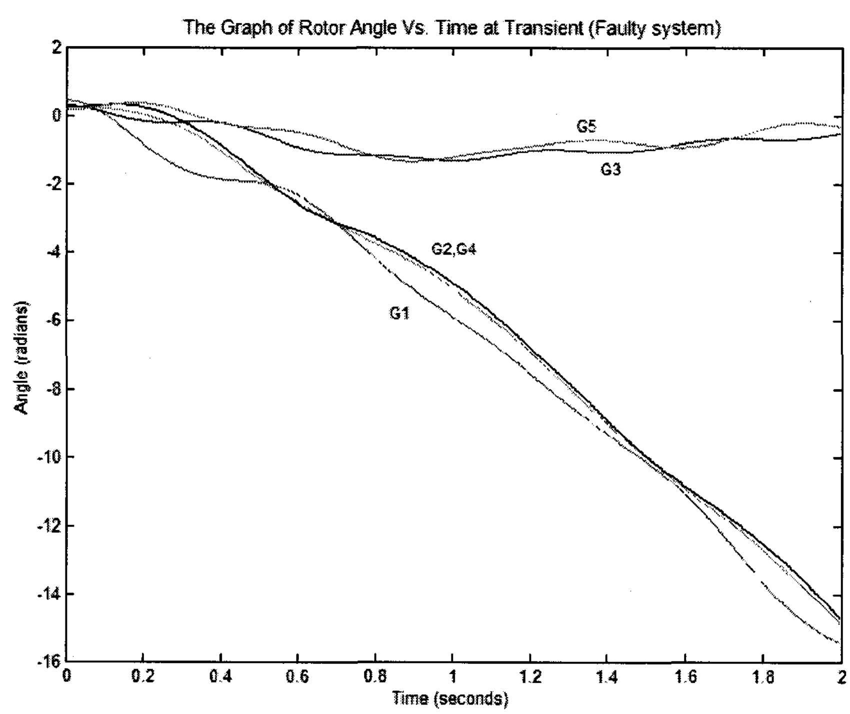


Figure 5.39 : Transient response of rotor angle

5.2.2.2 Transient Response of the System (Fault at bus 4)

Considering the fault is at bus 4 in transmission line section 4 and 12.

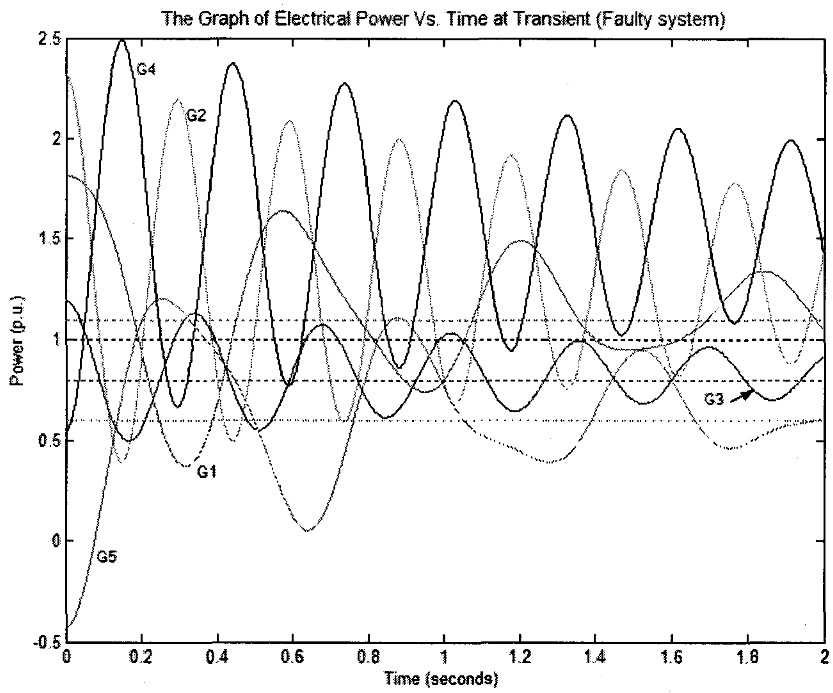


Figure 5.40 : Transient response of electrical power generated

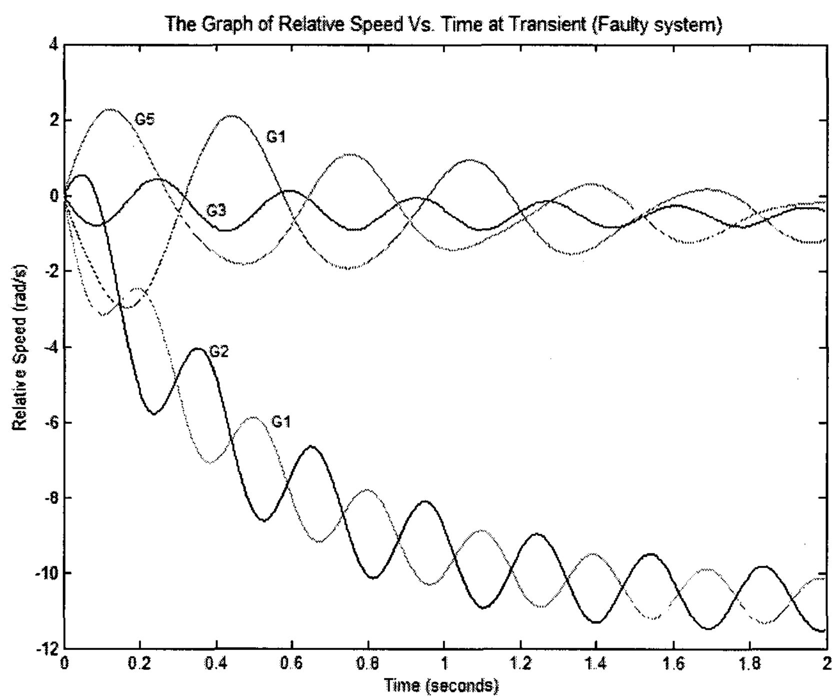


Figure 5.41 : Transient response of relative speed

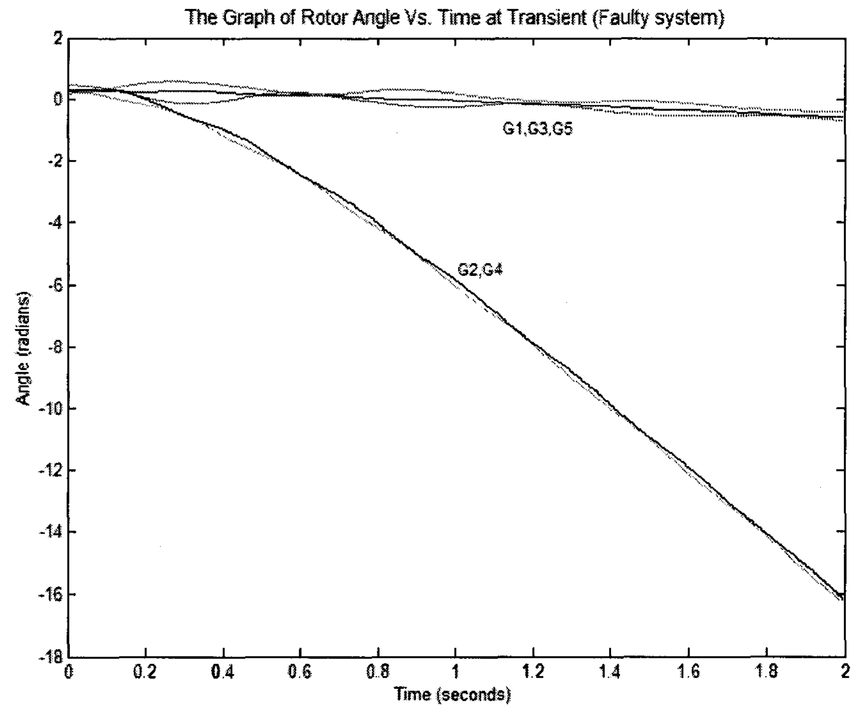


Figure 5.42 : Transient response of rotor angle

CHAPTER 6
DISCUSSION AND CONCLUSION

This project mainly focused on implementation of computer simulation package to analyze the transient stability of multi-machine power system while understanding the theories and methodologies behind the analysis.

The software package was developed in MATLAB environment using its version 6.1. To make the software user friendly, several graphical user interfaces were developed. The GUI "maingui" links two .pdf files which describes how to use the program and theories and methodologies while link the other GUIs for system data entry and transient stability analysis. GUI 'SystemData" links the other GUI which are developed for data entering of system components - generators, transformers, transmission lines and loads. All four GUIs were programmed in the similar manner to enter the data, to store the data in .mat files and view the .mat files if required. The programs were written in the way that once set of data was saved, it will delete the earlier set of data.

The GUI for transient stability analysis was programmed to retrieve all sets of data stored in .mat files, to prepare system Y-matrix, execute power flow and transient stability response of the system. Provisions were made in the GUI to change the machine damping constant, to change the inertia of the machines, faulty bus and faulty section, time step of integration, clearing time and total time of integration to see their effects on transient stability response of the system. Facility of entering fault clearing time provides the possibility of determining critical clearing time for each case depending on the time step of integration.

Once the software was developed, it was used to analyze the transient stability response of a 9-bus system and 15-bus system for their classical representation. The effect of machine damping, machine inertia, distance to the fault from the machine to the system transient response were studied.

6.1 Transient Response of 9-bus System

In analyzing transient stability response of the 9-bus system, two cases were considered. In the first case, it is considered that the system was subjected to three phase symmetrical fault at bus 7 in transmission line between bus 5 and bus 7 such that the fault is near to the generator 2. In second case, the fault is at bus 5 in the same section.

6.1.1 Fault is near to a Generator Bus

When the fault is near to a generator bus, the electrical power generated by the corresponding generator becomes zero or near to zero as the fault is fed by affected generator. This tends high imbalance between the mechanical power in to the generator and electrical power out from the generator causing rapid acceleration of the machine and rapid change of rotor angle making the system become unstable and tending the machine out of step with the other two. The other two generators of the system have the oscillatory nature of power exchange between them. Inter-machine oscillations occur in these two machines while machines are accelerated in much lesser speed than machine near to the fault.

When the machine damping is considered in the above system, these oscillations will be damped out after few seconds while they are accelerating at a lesser speed than the case when the machine damping is not considered. As the relative speed of the machine increases, the damping has a remarkable effect on the machine acceleration. Machine inertia also has a similar effect on machine acceleration. With higher machine inertia, the machine tends to accelerate slower.

When the fault is cleared within few cycles, the machines try to synchronize together tending the system to be stable. In the case without damping, the system tries to maintain its oscillatory nature. In actual situation, machine damping is present and these oscillations will be damped out and system will become stable in an operating state different from the pre fault operating state. The effect of machine damping and machine inertia could be clearly seen from the improved critical clearing time.

6.1.2 Fault is far from a Generator Bus

When the fault is at a load bus far from the generator bus, inter-machine oscillations occur in all 3 machines while accelerating speed and change of rotor angle depends on the machine inertia and distance to the fault. We could clearly see the difference between accelerating speed of machine 1 which has higher inertia and the other two machines having lower inertia with respect to the machine 1. With this argument, generator 3 which has lowest inertia should tend to accelerate rapidly compared to the other two machines. But, it could be clearly seen from the transient response of the system the machine 2 and 3 try to accelerate at the same speed. Machine 3 is much farthest to the fault than machine 2 and that will increase the impedance between the fault and machine. Further, power oscillations are much rapid than the case with the fault near to a generator bus as all three machine try to exchange power between them.

6.2 Transient Response of the 15-bus System

The complexity of the 15-bus system is rather higher than the 9-bus system. The loads are fed near to the generator buses. In realistic situation, the transmission system is fed by the generators in the system and transmission grid feeds the loads connected to it. In analyzing the transient response of the 15-bus system, it is considered that shunt reactors will compensate the Var requirement of loads connected to the same bus. There is not much difference in the generator inertia constants.

Even in the analysis of 15-bus system, the two cases, fault closer to a generator bus and a fault far from the generator buses were considered. When the fault is closer to the generator bus, power oscillations occur in each machine. Though the machine damping is considered, the oscillations will not be damped out quickly like 9-bus system as the complexity of the system is high. From the transient response of the system, it could be clearly seen that the machines closer to the fault swing together separately from the machines far from the fault. When the fault is far from the generators, although the power oscillations occur, they will be damped out within few seconds. Damping effect of most farthest generator is higher. As in the case of fault closer to the generators, generators in the different zones swing separately. The transient response of the complex system does not totally depend on the characteristics of the machine itself. It depends on the whole system.

6.3 Future work to be done

This software package is only developed to understand the transient behavior of the system if the 3-phase symmetrical fault occurs closer to a bus in a power system. In transient analysis, even if the 3-pahse symmetrical fault is considered it has to analyze the fault in the middle part of the transmission line. That part was omitted in this software considering the difficulty of preparing the reduced Y matrix as the admittance to the fault depends on the distance to the fault. Further, this software should be developed to analyze the unsymmetrical fault in the system as this is prepared to understand the transient behavior of the system.

Fault is not the only cause for a transient instability of the system. Loss of generation, loss of transmission line or loss of considerable load are some examples for the causes of transient instability. Further, in analyzing transient response it should be considered practical behavior of the generators and their controls rather than its classical representation. Transient behavior of loads and reactors should also be considered. In preparing a software for educational purposes, these should be included step by step as the students can understand the theories and methodologies behind the analysis unlike the software used for commercial purposes. The inclusion of those areas for the software is suggested as future work.

APPENDIX A

LIBRARY FILES USED IN GUI IMPLEMENTATION

Appendix is dedicated for the MATLAB programs written in the implementation of GUI. For this purpose, GUI implementation tools and handle structure environment of MATLAB is utilized.

A.1 Matlab Program : 'maingui'

The 'maingui' is for linking other files and for opening other GUIs.

```
% This program is to open the files 'Introduction' and 'Theory' in .pdf format
% and to open the gui 'system data' and 'transient analysis'
clear
% ------------------------------------------------------------------
function varargout = MainGUI_listbox_Callback(h, eventdata, handles, varargin)

if strcmp(get(handles.fig_maingui,'SelectionType'),'open')
   Val = get(handles.MainGUI_listbox,'Value');
   Str = get(handles.MainGUI_listbox,'String');

   switch Str{Val}
   case 'Introduction'
      delete (gcf)
      uigetfile('C:\MATLAB6p1\work\GUI\Introduction.pdf')
   case 'Thoery'
      delete (gcf)
      uigetfile ('C:\MATLAB6p1\work\GUI\Theory.pdf')
   case 'System Data Entry'
      delete(gcf)
      findobj ('Name',SystemData)
   case 'Steady State Power Flow'
      delete (gcf)
      findobj ('Name',Transient)
```

```
  case 'Classical Stability Analysis'
    delete (gcf)
    findobj ('Name',Transient)
  case 'Practical Model Stability Analysis'
    delete (gcf)
    findobj ('Name',Transient)
  end
end
% ---------------------------------------------------------------------
function varargout = Exit_pushbutton_Callback(h, eventdata, handles, varargin)
close(gcf)
```

A.2 Matlab Program : for GUI 'System Data'

The GUI 'System Data' is to open the other GUIs which are used for entering data.

```
clear
% ---------------------------------------------------------------------
function varargout = SystemData_listbox_Callback(h, eventdata, handles, varargin)

% to link the other GUIs
if strcmp(get(handles.fig_System,'SelectionType'),'open')
  Val = get(handles.SystemData_listbox,'Value');
  Str = get(handles.SystemData_listbox,'String');

  % to get the next screen
  switch Str{Val}
  case 'Generator Data'
    close (gcf)
    guide (generatordata)
  case 'Transformer Data'
    close (gcf)
    guide (transformerdata)
  case 'Transmission Line Data'
    close (gcf)
```

```
      guide (transmissiondata)
    case 'Load Data'
      close (gcf)
      guide (loaddata)
    end
end

% ----------------------------------------------------------------------
function varargout = Exit_pushbutton_Callback(h, eventdata, handles, varargin)

% to read number of buses
TotalBuses = str2num(get(handles.NoBus_edit,'string'));
% to save data in NoofBuses.mat file
open ('C:\MATLAB6p1\work\GUI\NoofBuses.mat');
load NoofBuses.mat NoofBuses;
NoofBuses;
[r,c] = size(NoofBuses);
if isequal ([r,c],[0,0])
  NoofBuses = TotalBuses;
else
  NoofBuses =[ ];
  NoofBuses = TotalBuses;
end
save NoofBuses;

% to close the figure
delete(gcf)
% back to Main GUI
findobj('Name',MainGUI)

% ----------------------------------------------------------------------
function varargout = Close_pushbutton_Callback(h, eventdata, handles, varargin)
% to close the figure
close(gcf)
```

A.3 Matlab Program : GUIs for Entering Data

Four different GUIs and MATLAB programs were written for entering system data of generators, transformers, transmission lines and loads. The coding of all four programs are similar except the data fields. Therefore, the program for generator data is written in this report while mentioning the data field of the other three programs.

A.3.1 Program for entering Generator Data (generatordata.m)

```
% This program is prepared for entering generator data needed for system calculation
% The data will be stored in 'gendata.mat' file
% data will be stored in the same order as in the .m file
% Once the data is stored and next time you enter new set of data and save it , the earlier
% -set of data will be replaced by new set of data

clear                                       % to clear data in the memory
GeneratorData = [ ];                        % to develop gendata matrix
handles.GeneratorData = GeneratorData;      % the program is written in handle structure
% -----------------------------------------------------------------
% function for taking the data into memory when the 'Enter' push button is pressed
function varargout = Enter_pushbutton_Callback(h, eventdata, handles, varargin)

% to get the generator data into numeric format
GenBus_No = str2num(get(handles.GenBusNo_edit,'string'));              % bus number
GenBus_Type = str2num(get(handles.GenBusType_edit,'string'));          % bus type
Bus_Voltage = str2double(get(handles.BusVolt_edit,'string'));          % bus voltage
Voltage_Angle = str2double(get(handles.AngleVolt_edit,'string'));      % voltage angle
P_Gen = str2double(get(handles.PGen_edit,'string'));           % active power generated
Q_Gen = str2double(get(handles.QGen_edit,'string'));           % reactive power generated
Xd = str2double(get(handles.Xd_edit,'string'));                % direct-axis reactance
Xq = str2double(get(handles.Xq_edit,'string'));                % quadrature-axis reactance
XdT = str2double(get(handles.XdT_edit,'string'));              % D-axis transient reactance
XqT = str2double(get(handles.XqT_edit,'string'));              % Q-axis transient reactance
Xl = str2double(get(handles.Xl_edit,'string'));                % leakage reactance
```

```
% to retrive generator data structure
if   isfield(handles,'GeneratorData')&~isempty(handles.GeneratorData)
     GeneratorData = handles.GeneratorData;
     % to determine number of generator data exist
     [r,c] = size(GeneratorData);
     Index = r;
     i = Index+1;
else
     GeneratorData = [ ];
     i =1;
end

% to assign generator data
GeneratorData(i,1) = GenBus_No;
GeneratorData(i,2) = GenBus_Type;
GeneratorData(i,3) = Bus_Voltage;
GeneratorData(i,4) = Voltage_Angle;
GeneratorData(i,5) = P_Gen;
GeneratorData(i,6) = Q_Gen;
GeneratorData(i,7) = Xd;
GeneratorData(i,8) = Xq;
GeneratorData(i,9) = XdT;
GeneratorData(i,10) = XqT;
GeneratorData(i,11) = Xl;

% remove data from GUI for next entry
set(handles.GenBusNo_edit,'string',[ ])
set(handles.GenBusType_edit,'string',[ ])
set(handles.BusVolt_edit,'string',[ ])
set(handles.AngleVolt_edit,'string',[ ])
set(handles.PGen_edit,'string',[ ])
set(handles.QGen_edit,'string',[ ])
set(handles.Xd_edit,'string',[ ])
```

```
set(handles.Xq_edit,'string',[ ])
set(handles.XdT_edit,'string',[ ])
set(handles.XqT_edit,'string',[ ])
set(handles.Xl_edit,'string',[ ])

% to store new generator data in handle structure
handles.GeneratorData = GeneratorData;
handles.Index = i;
guidata(h,handles)

% ------------------------------------------------------------------
% function to check previously entered data using 'Previous' push button
function varargout = Previous_pushbutton_Callback(h, eventdata, handles, varargin)

% to retrive index and the generator data
Index = handles.Index;
GeneratorData = handles.GeneratorData;
% to disable 'Enter' pushbutton & enable 'Next' pushbutton while paging back and forth
set(handles.Enter_pushbutton,'Enable','off')
set(handles.Next_pushbutton,'Enable','on')
% to set the required position
i = Index;

% to set the data corresponding to the previous case
set(handles.GenBusNo_edit,'string',num2str(GeneratorData(i,1)))
set(handles.GenBusType_edit,'string',num2str(GeneratorData(i,2)))
set(handles.BusVolt_edit,'string',num2str(GeneratorData(i,3)))
set(handles.AngleVolt_edit,'string',num2str(GeneratorData(i,4)))
set(handles.PGen_edit,'string',num2str(GeneratorData(i,5)))
set(handles.QGen_edit,'string',num2str(GeneratorData(i,6)))
set(handles.Xd_edit,'string',num2str(GeneratorData(i,7)))
set(handles.Xq_edit,'string',num2str(GeneratorData(i,8)))
set(handles.XdT_edit,'string',num2str(GeneratorData(i,9)))
set(handles.XqT_edit,'string',num2str(GeneratorData(i,10)))
```

```
set(handles.Xl_edit,'string',num2str(GeneratorData(i,11)))

% to end it up at first set of data
if  i>1
    i=i-1;
end

% to store data in handle structure
handles.Index = i;
guidata(h,handles)

% ------------------------------------------------------------------
% when the 'previous' push button is used, to go back to data entering position
% using 'Next' push button
function varargout = Next_pushbutton_Callback(h, eventdata, handles, varargin)

% to retrive index and the generator data
Index = handles.Index;
GeneratorData = handles.GeneratorData;
[r,c] = size(GeneratorData);
% to set the required position
if  Index<=r
    i = Index+1;
else
    i = Index;
end
if  i<=r
    % to set the data corresponding to next case
    set(handles.GenBusNo_edit,'string',num2str(GeneratorData(i,1)))
    set(handles.GenBusType_edit,'string',num2str(GeneratorData(i,2)))
    set(handles.BusVolt_edit,'string',num2str(GeneratorData(i,3)))
    set(handles.AngleVolt_edit,'string',num2str(GeneratorData(i,4)))
    set(handles.PGen_edit,'string',num2str(GeneratorData(i,5)))
    set(handles.QGen_edit,'string',num2str(GeneratorData(i,6)))
```

```
    set(handles.Xd_edit,'string',num2str(GeneratorData(i,7)))
    set(handles.Xq_edit,'string',num2str(GeneratorData(i,8)))
    set(handles.XdT_edit,'string',num2str(GeneratorData(i,9)))
    set(handles.XqT_edit,'string',num2str(GeneratorData(i,10)))
    set(handles.Xl_edit,'string',num2str(GeneratorData(i,11)))
else
    % to enable 'Enter' pushbutton
    set(handles.Enter_pushbutton,'Enable','on')
    % to disable 'Next' pushbutton while data entering
    set(handles.Next_pushbutton,'Enable','off')
    % to set GUI for data entry
    set(handles.GenBusNo_edit,'string',[ ])
    set(handles.GenBusType_edit,'string',[ ])
    set(handles.BusVolt_edit,'string',[ ])
    set(handles.AngleVolt_edit,'string',[ ])
    set(handles.PGen_edit,'string',[ ])
    set(handles.QGen_edit,'string',[ ])
    set(handles.Xd_edit,'string',[ ])
    set(handles.Xq_edit,'string',[ ])
    set(handles.XdT_edit,'string',[ ])
    set(handles.XqT_edit,'string',[ ])
    set(handles.Xl_edit,'string',[ ])
end

handles.Index = i;
guidata(h,handles)
% ------------------------------------------------------------------
% function to save data in 'gendata.mat' file
function varargout = Save_Callback(h, eventdata, handles, varargin)

GeneratorData = handles.GeneratorData
% to save data in gendata.mat file
open('C:\MATLAB6p1\work\GUI\gendata.mat');
load gendata.mat gendata;
```

```
gendata;
[r,c] = size(gendata)
if isequal([r,c],[0,0])
   gendata = GeneratorData
else
   gendata =[ ];
   gendata = GeneratorData
end
uiputfile('gendata.mat')
save('C:\MATLAB6p1\work\GUI\gendata.mat');

% ----------------------------------------------------------------
% function to view 'gendata.mat' file
function varargout = ViewGenData_Callback(h, eventdata, handles, varargin)

uigetfile('gendata.mat')
open('C:\MATLAB6p1\work\GUI\gendata.mat')
load gendata.mat gendata;
gendata;
assignin('base','gendata',gendata);
open('gendata')

% ----------------------------------------------------------------
% function to go back to 'system data' GUI using 'Exit' button
function varargout = Exit_pushbutton_Callback(h, eventdata, handles, varargin)
delete(gcf)
findobj('Name',SystemData)
% ----------------------------------------------------------------
% to close the GUI completely
function varargout = Close_pushbutton_Callback(h, eventdata, handles, varargin)
close(gcf)
```

A.3.2 Program for entering Transformer Data (transformerdata.m)

a.) The data fields used in entering transformer data

Transformer_No	**% transformer identification number**
Rated_MVA	**% rated power in MVA**
Max_MVA	**% maximum power in MVA**
LV_Bus	**% bus number which LV side connected**
LV_Voltage	**% LV side rated voltage**
HV_Bus	**% bus number which HV side connected**
HV_Voltage	**% HV side rated voltage**
Phase_shift	**% phase shift from primary to secondary**
Tap	**% tap position**
Impedence	**% transformer impedance in p.u.**
ImpR_ratio	**% transformer reactance to resistance ratio**

b.) The .mat file used for data storing : **'transfdata.mat'**

A.3.3 Program for entering Transmission Line Data (transmissiondata.m)

a.) The data fields used in entering transmission line data

LineLength	**% transmission line length**
Resistance	**% total series resistance for Π model**
Reactance	**% total series reactance for Π model**
Susceptance	**% shunt susceptance (B/2)**
Bus1	**% line connected bus number 1**
Bus2	**% line connected bus number 2**

b.) The .mat file used for data storing : **'transldata.mat'**

A.3.4 Program for entering Load Data (loaddata.m)

a.) The data fields used in entering transmission line data

LoadBus_No	**% bus number which load is connected**
BusType	**% type of bus (load bus – bus type 3)**
P_load	**% active power utilized**
Q_load	**% reactive power utilized**

b.) The .mat file used for data storing : **'loadata.mat'**

A.4 Matlab Program : GUI for Power Flow & transient stability (transient.m)

The 'transient.m' program file is developed to analyze the pre-transient state and transient response of the system. The description of the function is in chapter4.

A.4.1 Main program of power flow & transient stability analysis

```
function varargout = File_Callback(h, eventdata, handles, varargin)
% ------------------------------------------------------------------
% function for retrieving System Data using 'File -Open' menu
function varargout = Open_Callback(h, eventdata, handles, varargin)
[x,y] = uigetfile('*.mat');
switch x
case 'NoofBuses.mat'
    load NoofBuses.mat NoofBuses;
    NoofBuses;
    handles.NoofBuses = NoofBuses;
case 'gendata.mat'
    load gendata.mat gendata;
    gendata;
    handles.gendata = gendata;
case 'transfdata.mat'
    load transfdata.mat transfdata;
    transfdata;
    handles.transfdata = transfdata;
case 'transldata.mat'
    load transldata.mat transldata;
    transldata;
    handles.transldata = transldata;
case 'loadata.mat'
    load loadata.mat loadata;
    loadata;
    handles.loadata = loadata;
end
```

```
% to store handles
guidata(h,handles)

% ------------------------------------------------------------------
% function for saving power flow results
function varargout = Save_Callback(h, eventdata, handles, varargin)

Y = handles.Y;
powerflow_results = handles.powerflow_results;

% to save admittance matrix in Y_matrix.mat file
[x,y] = uigetfile('*.mat');
switch x
case  'Y_matrix.mat'
   load Y_matrix.mat Y_matrix;
   Y_matrix;
   [r,c] = size(Y_matrix);
   if isequal ([r,c],[0,0])
      Y_matrix = Y;
   else
      Y_matrix =[];
      Y_matrix =Y;
   end
   uiputfile ('Y_matrix.mat')
   save('C:\MATLAB6p1\work\GUI\Y_matrix.mat');

case 'powerflow.mat'
   load powerflow.mat powerflow;
   powerflow;
   [r,c] = size(powerflow);
   if  isequal([r,c],[0,0])
      powerflow = powerflow_results;
   else
```

```
        powerflow =[];
        powerflow = powerflow_results;
    end
    uiputfile ('powerflow.mat')
    save ('C:\MATLAB6p1\work\GUI\powerflow.mat');
end

% ------------------------------------------------------------------
% function to clear contents in data fields of GUI 'transient'
function varargout = Clear_Callback(h, eventdata, handles, varargin)
set(handles.FaultyBus_edit,'string',[])
set(handles.FaultyEnd_edit,'string',[])
set(handles.D_edit,'string',[])
set(handles.perH_edit,'string',[])
set(handles.TimeStep_edit,'string',[])
set(handles.ClearTime_edit,'string',[])
set(handles.TotalTime_edit,'string',[])

guidata(h,handles)

% ------------------------------------------------------------------
% function to exit from the GUI
function varargout = Exit_Callback(h, eventdata, handles, varargin)
delete(gcf)
findobj('Name',maingui)

% ------------------------------------------------------------------
function varargout = Execute_Callback(h, eventdata, handles, varargin)
% ------------------------------------------------------------------
function varargout = PowerFlow_Callback(h, eventdata, handles, varargin)
% main function for power flow analysis
% function includes programs of pre-transient stability studies
% Y-matrix preparation and Power flow analysis using Newton-Raphson method
% function runs when 'execute-powerflow' menu is selected
```

```
% Number of buses in the system
N = handles.NoofBuses;

% to prepare transformer bus admittance matrix
% Transformer data as a matrix
% Transfdata = ['Transformer_No.' 'Rated_MVA' 'Max_MVA'  'LV_Bus' 'LV_Voltage'...
%                   'HV_Bus'  'HV_Voltage'  'Phase'  'Tap_position' 'Impedence'  'X/R Ratio']
Transfdata = handles.transfdata;

% to calculate complex turns ratio
a = (cos(Transfdata(:,8))+i*sin(Transfdata(:,8))).*Transfdata(:,9);
% transformer bus admittance matrix
Y_Transf =Ytransf(Transfdata,a,N);                    % 'Ytransf' is a separate function

% to prepare transmission line admittance matrix
Transldata = handles.transldata;
[h1,w1] = size(Transldata);
% Bus impedence matrix & susceptance matrix
imp = zeros(N);
B = zeros(N);
for m = 1:h1
   imp(Transldata(m,5),Transldata(m,6)) = Transldata(m,2)+i*Transldata(m,3);
   B(Transldata(m,5),Transldata(m,6)) = 0+i*Transldata(m,4);
end
imp;
B;
Y_Transm = Ytransm(imp,B,N);                          % 'Ytransm' is a separate function

% Nodel admittance matrix of the system
Y = Y_Transf+Y_Transm
abs_Y = abs(Y);
angle_Y =angle(Y);

% -------------------------------------------
```

```
% arranging system data for power flow calculation
% gendata = ['Bus_No.' 'Bus_Type' 'Bus_Voltage' 'Voltage_Angle' 'P_gen' 'Q_gen'...
%            'Xd' 'Xq' 'Xd_Transient' 'Xq_Transient' 'Xl']
gendata = handles.gendata;
[r,c] = size(gendata);

% Loaddata = ['Bus_No.' 'Bus_Type' 'P_load' 'Q_load']
loaddata = handles.loadata;
[r1,c1] = size(loaddata);

% Loading data of the system
% Data = ['Bus_No.' 'Bus_type' 'Bus_Voltage' 'V_Angle' 'P_gen' 'Q_gen' 'P_load' 'Q_load'];
Data(N,8) = 0;
for m =1:r
  Data(gendata(m,1),1:6) = gendata(m,1:6);
end
for n =1:r1
  Data(loaddata(n,1),1:2) = loaddata(n,1:2);
  Data(loaddata(n,1),7:8) = loaddata(n,3:4);
end
for k =1:N
    if isequal(Data(k,1),0)
        Data(k,1) =k;
        Data(k,2) =3;
    end
end
Data

% Machine data (reactance data of the machines)
% machine_data = ['Bus_No' 'Xd' 'Transient Xd' 'Xq' 'Transient Xq' ' Leakage X'];
machine_data(:,1) = gendata(:,1);
machine_data(:,2:6) = gendata(:,7:11);
machine_data = sortrows(machine_data,1)
```

```
% -------------------------------------------
% arranging loading data matrix for calculation
for n = 1:N
    spec(n,1:4) = Data(n,1:4);
    spec(n,5) = Data(n,5)- Data(n,7);
    spec(n,6) = Data(n,6)- Data(n,8);
end
% to set bus voltage to 1.0 p.u.
for n = 1:N
    if spec(n,2)==3 & spec(n,3)==0
        spec(n,3)=1;
    end
end
spec;                                   % spec is the data matrix arranged for power flow

% Load Flow using Newton Raphson method
error = 1;                              % initial error is set to 1
k=1;
% iteration loop starts
while error > 0.000001
% to find active and reactive power of buses
    for i=1:N
        x = 0;
        y = 0;
        for j=1:N
            p = spec(i,3)*abs_Y(i,j)*spec(j,3)*cos(spec(i,4)-spec(j,4)-angle_Y(i,j));
            q = spec(i,3)*abs_Y(i,j)*spec(j,3)*sin(spec(i,4)-spec(j,4)-angle_Y(i,j));
            P(i,1) = x+p;
            Q(i,1) = y+q;
            x = P(i,1);
            y = Q(i,1);
        end
        Diff_P(i,1) = spec(i,5)-P(i,1);         % to find active power mismatches
        Diff_Q(i,1) = spec(i,6)-Q(i,1);         % to find reactive power mismatches
```

```
end

% to seperate required mismatches
m = find(spec(:,2)~=1);                % separation of PV and PQ buses for P
p = P(m,1);
diff_p =Diff_P(m,1);
Power_PM = [m spec(m,5) p diff_p];
[r1 c] = size(diff_p);

n = find(spec(:,2)==3);                % separation of PQ busses for reactive power
q = Q(n,1);
diff_q =Diff_Q(n,1);
Power_QM = [n spec(n,6) q diff_q];

Diff_power = [diff_p;diff_q];          % power mismatch
[r2 c] = size(Diff_power);
err = abs(Diff_power);
error = max(err);                      % error to check mismatches

% Calculation of Jacobian matrix using function 'Jacobian'
jacobian = Jacobian(spec,Y,N);         % function to calculate 'Jacobian' matrix

% to calculate difference in voltage and angle
diff = jacobian\Diff_power;
diff_angle = diff(1:r1,1:c);
diff_v = diff(r1+1:r2,1:c);

% to claculate new values of voltage and angle
diffangle_V(m,1) = diff_angle;
[h w] =size(diffangle_V);
if h~=N
    diffangle_V(N,1) =0;
end
diffangle_V;
```

```
    angleV = spec(:,4) +diffangle_V;
    spec(:,4) = angleV;

    diff_V(n,1) =diff_v;
    [h1,w1] = size(diff_V);
    if h1~=N
        diff_V(N,1) =0;
    end
    diff_V;
    V = spec(:,3).*(1+diff_V);
    spec(:,3) = V;

    spec;
    k=k+1;
end

% substitution of calculated active and reactive power values to data matrix
P;                          % calculated values of active power (from last iteration)
Q;                          % calculated values of reactive power (from last iteration)
x = find(spec(:,2)==1);     % separation of slack bus
y = find(spec(:,2)~=3);     % separation of slack and PV Buses
spec(x,5) = P(x,1);
spec(y,6) = Q(y,1);
spec;

% power flow results
powerflow_results= [spec(:,1:3) spec(:,4)*180/pi spec(:,5:6)]

% -------------------------------------------
% calculation of bus voltages in complex form
for i=1:N
    BusVoltage(i,1) = ComplexV(spec(i,3),spec(i,4));
end
BusVoltage;
```

```
Current = Y*BusVoltage;

% calculation of loading of lines
A(1,1:1:5) =1;
for i=1:N
    for j=1:N
        if imp(i,j)~=0
        A1 = ((1+(imp(i,j)*B(i,j)))*BusVoltage(i,1))-BusVoltage(j,1);
        BusCurrent = A1/imp(i,j);
        power = conj(BusCurrent)*BusVoltage(i,1);
        data= [i j BusCurrent real(power) imag(power)];
        BusLoad = [A;data];
        A = BusLoad;
        end
    end
end

for i=1:N
    for j=1:N
        if Y_Transf(i,j)~=0 & i~=j
        A2 = Y_Transf(i,i)*BusVoltage(i,1)+Y_Transf(i,j)*BusVoltage(j,1);
        BusCurrent = A2;
        power = conj(BusCurrent)*BusVoltage(i,1);
        data= [i j BusCurrent real(power) imag(power)];
        BusLoad = [A;data];
        A = BusLoad;
        end
    end
end

[r,c] = size(A);
BusLoad = A(2:r,:)                          % load flow between buses
BusCurrent = BusLoad(:,3);
abs(BusCurrent);
```

```
angle(BusCurrent)*180/pi;

% Calculation of steady state internal voltages of generators
L = find(spec(:,2)~=3);
V_Terminal = BusVoltage(L,1)
I_GenLoad = Current(L,1)                          % generator loads
abs(I_GenLoad);
angle(I_GenLoad)*180/pi;
PF_angle = angle(V_Terminal)-angle(I_GenLoad);

% Generator steady state synchronous reactance data
Xd = machine_data(:,2);
Xq = machine_data(:,3);
Xl = machine_data(:,6);

%  Generator steady state internal votages
E = emf(V_Terminal,I_GenLoad,Xd,Xq,Xl)
abs_E =abs(E);
angle_E = angle(E)*180/pi;

% to store data in handle structure for later retrieval
handles.Data =Data;                               % loading data of the system
handles.machine_data = machine_data;              % generator reactance data
handles.imp = imp;                                % transmission line impedance data
handles.B = B;                                    % transmission line susceptance data
handles.a =a;                                     % complex turns ratio of transformers
handles.Y_Transf = Y_Transf;                      % admittance matrix for transformers
handles.Y_Transm = Y_Transm;                      % admittance matrix for transmission lines
handles.Y = Y;                                    % nodal admittance matrix
handles.spec = spec;                              % power flow results for calculation
handles.powerflow_results = powerflow_results; % power flow results
handles.V_Terminal = V_Terminal;                  % generator terminal voltage
handles.I_GenLoad = I_GenLoad;                    % generator current
handles.E = E;                                    % generator internal voltages
```

```
guidata(gcbo,handles)

% -------------------------------------------------------------------
% function to analyze transient response
function varargout = ClassicalModel_Callback(h, eventdata, handles, varargin)

N = handles.NoofBuses;
Transfdata = handles.transfdata;
a = handles.a;
imp =handles.imp;
B = handles.B;
Y_Transf = handles.Y_Transf;
Y_Transm = handles.Y_Transm;
spec = handles.spec;
V_Terminal = handles.V_Terminal;
I_GenLoad = handles.I_GenLoad;
machine_data = handles.machine_data;
Xd_T = machine_data(:,4);
Xq_T = machine_data(:,5);
Xl = machine_data(:,6);
H = machine_data(:,7);

per_H = str2num(get(handles.perH_edit,'string'));                 % percentage increase of H
D = str2num(get(handles.D_edit,'string'));                        % damping constant
FaultyBus = str2num(get(handles.FaultyBus_edit,'string'));        % Faulty Bus
FaultyEnd = str2num(get(handles.FaultyEnd_edit,'string'));        % Faulty end
delta_t = str2num(get(handles.TimeStep_edit,'string'));           % Time step of integration
tc = str2num(get(handles.ClearTime_edit,'string'));               % clearing time
T = str2num(get(handles.TotalTime_edit,'string'));                % Total time of integration

% internal voltages of generators at transient state
E1 = emf(V_Terminal,I_GenLoad,Xd_T,Xq_T,Xl);
internal_E1 = [abs(E1) angle(E1)*180/pi]
```

```
% impedences of the connected loads
adml = admittance(spec,N)
Y1 = Y_Transm+adml;

% considering transient reactance of generators
% Y matrix of the pre-fault network
L = find(spec(:,2)~=3);
[d,c] = size(L);
Gen = [L Xd_T];
[b,c1] = size(Transfdata);
X1=0;
for n=1:b
   for l=1:d
      if Transfdata(n,4)==Gen(l,1)
      X1 = Transfdata(n,10)+(abs(a(n,1)))^2*Gen(l,2);
      Transfdata(n,10)= X1;
      X1 = 0;
      end
   end
end
Y_Trans = Ytransf(Transfdata,a,N);
Y_pre = Y1+Y_Trans

% Y matrix of the faulty network
a1 = Y_pre(1:(FaultyBus-1),1:(FaultyBus-1));
b1 = Y_pre(1:(FaultyBus-1),(FaultyBus+1):N);
c1 = Y_pre((FaultyBus+1):N,1:(FaultyBus-1));
d1 = Y_pre((FaultyBus+1):N,(FaultyBus+1):N);
Y_fault =[a1 b1; c1 d1]

%Y matrix after fault cleared
if imp(FaultyBus,FaultyEnd)~=0;
   imp(FaultyBus,FaultyEnd)=0;
else
```

```
  imp(FaultyEnd,FaultyBus)=0;;
end
imp;
if B(FaultyBus,FaultyEnd)~=0;
  B(FaultyBus,FaultyEnd)=0;
else
  B(FaultyEnd,FaultyBus)=0;
end
B;
Y_Transm = Ytransm(imp,B,N);
Y3 = Y_Transm+adm1
Y_post = Y3+Y_Trans

% Reduced Y-Matrices using node elimination method
Reduced_Y_pre = kron(Y_pre,spec)
Reduced_Y_fault = kron(Y_fault,spec)
Reduced_Y_post = kron(Y_post,spec)

Pm = spec(L,5);                         % mechanical power input of the machine
H = H*(per_H)/100;                      % machine inertia constant

% function for transient analysis
TransientData = Transient(E1,Pm,Reduced_Y_fault,Reduced_Y_post,D,H,delta_t,tc,T);

time = TransientData(:,1);
mech_power = TransientData(:,2:4);
elec_power = TransientData(:,5:7);
relative_speed = TransientData(:,8:10);
load_angle = TransientData(:,11:13);
angle_difference1 = load_angle(:,2)-load_angle(:,1);
angle_difference2 = load_angle(:,3)-load_angle(:,1);

% to plot the graph of power vs. time
fig1 = figure('position', [120 100 670 500])
```

```
plot(time(:,1),elec_power(:,1),'r-')
hold on
plot(time(:,1),mech_power(:,1),'r:')
plot(time(:,1),elec_power(:,2),'g-')
plot(time(:,1),mech_power(:,2),'g:')
plot(time(:,1),elec_power(:,3),'b-')
plot(time(:,1),mech_power(:,3),'b:')
plot(time(:,1),elec_power(:,4),'k-')
plot(time(:,1),mech_power(:,4),'k:')
plot(time(:,1),elec_power(:,5),'m-')
plot(time(:,1),mech_power(:,5),'m:')
title('The Graph of Electrical Power Vs. Time at Transient')
Xlabel('Time(seconds)')
Ylabel('Power(p.u.)')

% to plot the graph of relative speed vs. time
fig2 = figure('position',[120 100 670 500])
plot(time(:,1),relative_speed(:,1),'r-')
hold on
plot(time(:,1),relative_speed(:,2),'g-')
plot(time(:,1),relative_speed(:,3),'b-')
plot(time(:,1),relative_speed(:,4),'k-')
plot(time(:,1),relative_speed(:,5),'m-')
title('The Graph of Relative Speed Vs. Time at Transient')
Xlabel('Time(seconds)')
Ylabel('Relative Speed(rad/s)')

% to plot the graph of rotor angle vs. time
fig3 = figure('position',[120 100 670 500])
plot(time(:,1),load_angle(:,1),'r-')
hold on
plot(time(:,1),load_angle(:,2),'g-')
plot(time(:,1),load_angle(:,3),'b-')
plot(time(:,1),load_angle(:,4),'k-')
```

```
plot(time(:,1),load_angle(:,5),'m-')
title('The Graph of Rotor Angle Vs. Time at Transient')
Xlabel('Time(seconds)')
Ylabel('Angle(radians)')

% -----------------------------------------------------------------
function varargout = View_Callback(h, eventdata, handles, varargin)
% -----------------------------------------------------------------
function varargout = SystemData_Callback(h, eventdata, handles, varargin)
% System Data
[x,y] = uigetfile('*.mat');

switch x
case 'NoofBuses.mat'
   load NoofBuses.mat NoofBuses;
   NoofBuses;
   assignin('base','NoofBuses',NoofBuses);
   open('NoofBuses')
case 'gendata.mat'
   load gendata.mat gendata;
   gendata;
   assignin('base','gendata',gendata);
   open('gendata')
case 'transfdata.mat'
   load transfdata.mat transfdata;
   transfdata;
   assignin('base','transfdata',transfdata);
   open('transfdata')
case 'transldata.mat'
   load transldata.mat transldata;
   transldata;
   assignin('base','transldata',transldata);
   open('transldata')
case 'loadata.mat'
```

```
   load loadata.mat loadata;
   loadata;
   assignin('base','loadata',loadata);
   open('loadata')
end

% ----------------------------------------------------------------
function varargout = Admittance_Callback(h, eventdata, handles, varargin)

% to view admittance matrix
uigetfile('Y_matrix.mat')
open('C:\MATLAB6p1\work\GUI\Y_matrix.mat')
load Y_matrix.mat Y_matrix;
Y_matrix;
assignin('base','Y_matrix',Y_matrix);
open('Y_matrix')

% ----------------------------------------------------------------
```

A.2.2 Extra functions used in power flow program& transient stability analysis

```
% ----------------------------------------------------------------
% EXTRA FUNCTIONS
% ----------------------------------------------------------------
% function to prepare transformer bus admittance matrix
function Y_Transf = Ytransf(Transfdata,a,N)

[h,w] = size(Transfdata);
% to calculate transformer impedence
% Transfdata = ['Transformer_No.' 'Rated_MVA' 'Max_MVA' 'LV_Bus' 'LV_Voltage'...
%               'HV_Bus' 'HV_Voltage' 'Phase' 'Tap_position' 'Impedence' 'X/R Ratio']
% to calculate transformer impedance
z = (Transfdata(:,10)./Transfdata(:,2))*100;          % z in common base

for n=1:h
```

```
    if Transfdata(n,11)==1
        Z(n,1) = i*z(n,1);
    else
        r = 1+Transfdata(n,11).^2;
        R = z(n,1)/sqrt(r);
        X= Transfdata(n,11).*R;
        Z(n,1) = R+i*X;
    end
end
Y_tr = Z.\1;

% to calculate bus admittances of the transformer connected buses
Y_Transf=zeros(N);
for n=1:h
    Y1_Transf = zeros(N);
    Y1_Transf(Transfdata(n,4),Transfdata(n,4)) = Y_tr(n,1)*(abs(a(n,1)))^2;
    Y1_Transf(Transfdata(n,4),Transfdata(n,6)) = -Y_tr(n,1)*conj(a(n,1));
    Y1_Transf(Transfdata(n,6),Transfdata(n,4)) = -Y_tr(n,1)*a(n,1);
    Y1_Transf(Transfdata(n,6),Transfdata(n,6)) = Y_tr(n,1);
    x = Y_Transf + Y1_Transf;
    Y_Transf =x;
end
Y_Transf;

% -------------------------------------------------------------------
% function to prepare transmission line admittance matrix
function Y_Transm = Ytransm(imp,B,N)

% Bus impedence matrix
for i=1:N
    for j=1:N
        if imp(i,j)~=0
        imp(j,i)=imp(i,j);
        end
```

```
    end
end
imp;

% Bus susceptance matrix
for i=1:N
    for j=1:N
        if B(i,j)~=0
        B(j,i)= B(i,j);
        end
    end
end
B;

% to prepare bus admittance matrix
for i=1:N
    for j=1:N
        if imp(i,j)~=0
        adm(i,j)=1/imp(i,j);
        end
    end
end

% considering Bus susceptance
for  i=1:N
    x(i,i)=0;
    for j=1:N
        if B(i,j)~=0
        x(i,i) = adm(i,i)+B(i,j);
        end
        adm(i,i) = x(i,i);
    end
end
adm;
```

```
% the Y matrix
Y_Transm(i,i)=0;
for i=1:N
    for j=1:N
        x(i,i) = Y_Transm(i,i)+adm(i,j);
        Y_Transm(i,i) = x(i,i);
        if  i<j|i>j
        Y_Transm(i,j)=-adm(i,j);
        end
    end
end
Y_Transm;

% -------------------------------------------------------------------
%function to calculate Jacobian matrix
function jacobian = Jacobian(spec,Y,N)

% to get magnitude and angle of admittance
Y;
abs_Y = abs(Y);
angle_Y =angle(Y);

spec;
m = find(spec(:,2)~=1);                          % separation of PV and PQ buses
n = find(spec(:,2)==3);                          % separation of PQ buses

N;
% partial derivatives of active power against angle
for i=1:N
    x = 0;
    if spec(i,2)~=1
        for j=1:N
            if i~=j
```

```
                daba_pdelta = -spec(i,3)*abs_Y(i,j)*spec(j,3)*sin(spec(i,4)-spec(j,4)-angle_Y(i,j));
                if spec(j,2)~=1
                daba_Pdelta(i,j) = -daba_pdelta;
                end
                daba_Pdelta(i,i) = x+daba_pdelta;
                x = daba_Pdelta(i,i);
                end
            end
        end
    end
    daba_Pdelta;
    daba_PDelta = daba_Pdelta(m,m);

    % partial derivatives of active power against voltage
    for  i=1:N
        x = 0;
        if spec(i,2)~=1
            for j=1:N
                if i~=j
                Vdaba_pu = spec(i,3)*spec(j,3)*abs_Y(i,j)*cos(spec(i,4)-spec(j,4)-angle_Y(i,j));
                if spec(j,2)==3
                Vdaba_Pu(i,j) = Vdaba_pu;
                end
                else
                Vdaba_pu = 2*abs_Y(i,j)*spec(i,3)^2*cos(angle_Y(i,j));
                end
                Vdaba_Pu(i,i) = x+Vdaba_pu;
                x = Vdaba_Pu(i,i);
            end
        end
    end
    Vdaba_Pu;
    Vdaba_PU = Vdaba_Pu(m,n);
```

```
% partial derivatives of reactive power against voltage
for  i=1:N
    x = 0;
    if spec(i,2)==3
        for j=1:N
            if i~=j
            Vdaba_qu = spec(i,3)*spec(j,3)*abs_Y(i,j)*sin(spec(i,4)-spec(j,4)-angle_Y(i,j));
            if spec(j,2)==3
            Vdaba_Qu(i,j) = Vdaba_qu;
            end
            else
            Vdaba_qu = -2*abs_Y(i,j)*spec(i,3)^2*sin(angle_Y(i,j));
            end
            Vdaba_Qu(i,i) = x+Vdaba_qu;
            x = Vdaba_Qu(i,i);
        end
    end
end
Vdaba_Qu;
Vdaba_QU = Vdaba_Qu(n,n);

% partial derivatives of reactive power against angle
for i=1:N
  x = 0;
  if spec(i,2)==3
        for j=1:N
            if i~=j
            daba_qdelta = spec(i,3)*abs_Y(i,j)*spec(j,3)*cos(spec(i,4)-spec(j,4)-angle_Y(i,j));
            if spec(j,5)~=1
            daba_Qdelta(i,j) = -daba_qdelta;
            end
            daba_Qdelta(i,i) = x+daba_qdelta;
            x = daba_Qdelta(i,i);
            end
```

```
        end
    end
end
daba_Qdelta;
daba_QDelta = daba_Qdelta(n,m);

% Jacobian matrix
jacobian = [daba_PDelta Vdaba_PU;daba_QDelta Vdaba_QU];

% -------------------------------------------------------------------
% to calculate complex form of voltage
function V = ComplexV(x,teta)
p = x*cos(teta);
q = x*sin(teta);
V = p + i*q;
% -------------------------------------------------------------------
%function to calculate internal voltage of generators
function E = emf(V_Terminal,I_GenLoad,Xd,Xq,Xl)
xd = Xd+Xl;
xq = Xq+Xl;
Eq = V_Terminal+ i*xq.*I_GenLoad;
abs(Eq);
angle(Eq);                                                    % angle of internal volatge
Iaq = abs(I_GenLoad).*cos(-angle(I_GenLoad)+angle(Eq));       % q-axis component
Iad = abs(I_GenLoad).*sin(-angle(I_GenLoad)+angle(Eq));       % d-axis component
real_Iq = Iaq.*cos(angle(Eq));
imag_Iq = Iaq.*sin(angle(Eq));
real_Id = Iad.*sin(angle(Eq));
imag_Id = -Iad.*cos(angle(Eq));
Iq = real_Iq + i*imag_Iq;                            % q-axis component in complex form
Id = real_Id + i*imag_Id;                            % d-axis component in complex form
E = V_Terminal + i*xd.*Id +i*xq.*Iq;                 % internal voltage

% -------------------------------------------------------------------
```

```
function adm = admittance(spec,N)
% function to calculate load admittance
x = find(spec(:,2)==3&(spec(:,5)&spec(:,6))~=0);
M=zeros(N,1);
M(x,1) = -((spec(x,5)-i*spec(x,6)))./(spec(x,3).^2);
for m=1:N
adm(m,m) = M(m,1);
end
adm;

% ------------------------------------------------------------------
% function for node elimination using Kron Method
function Reduced_Y = kron(Y,spec)

x = find(spec(:,2)~=3);
y =length(x);
k = 1;
[r c] = size(Y);
while r>y
if k~=x
   for a=1:r
     for n=1:c
     Y_new(a,n) = Y(a,n)-(Y(a,k)*Y(k,n)/Y(k,k));
     end
   end
   A = Y_new(1:(k-1),1:(k-1));
   B = Y_new(1:(k-1),(k+1):c);
   C = Y_new((k+1):r,1:(k-1));
   D = Y_new((k+1):r,(k+1):c);
   Y = [A B;C D];
   [r1 c1] =size(Y);
   r=r1;
   c=c1;
else
```

```
k=k+1;
end
end
Reduced_Y =Y;

% -------------------------------------------------------------------
%function for transient analysis
function TransientData = Transient(E1,Pm,Reduced_Y_fault,Reduced_Y_post,D,H,delta_t,tc,T)

abs_E1 =abs(E1);
delta_angle = angle(E1);         % initial load angle of generators
Pm;                              % mechanical power output
H;                               % machine inertia constant

% using Taylor Series expansion and neglecting higher order terms

f = 60;
[r,c] = size(H);
delta_omega = zeros(r,1);
for i=1:r
    K(i,1) = (pi*f/H(i,1));
end
K;

t =0;
delta_t;
D;
tc;
T;
x =1;
while t<T
for i=1:5
    k=0;
    t1(x,1) =t;
```

```
    for j=1:5
        if t1<tc
        p = abs_E1(i,1)*abs(Reduced_Y_fault(i,j))*abs_E1(j,1)*cos(delta_angle(i,1)-
            delta_angle(j,1)-angle(Reduced_Y_fault(i,j)));
        else
        p = abs_E1(i,1)*abs(Reduced_Y_post(i,j))*abs_E1(j,1)*cos(delta_angle(i,1)-
            delta_angle(j,1)-angle(Reduced_Y_post(i,j)));
        end
    Pe(i,1) = k+p;
    k = Pe(i,1);
    end
Pa(i,1) = Pm(i,1)-Pe(i,1);
Pelect(x,i) = Pe(i,1);
Pmech(x,i) = Pm(i,1);
end

delta_omega1= delta_omega+ (K.*Pa-delta_omega.*D).*delta_t;
DeltaOmega(x,:) = delta_omega';
delta_angle1 = delta_angle+delta_omega.*delta_t+(K.*Pa-delta_omega.*D).*(delta_t^2/2);
DeltaAngle(x,:) =delta_angle';
delta_omega = delta_omega1;
delta_angle =delta_angle1;

x=x+1;
t=t+delta_t;
end

TransientData = [t1,Pmech,Pelect,DeltaOmega,DeltaAngle];
```

REFERENCES

[1] John J. Grainger, William D. Stevenson. "*Power System Analysis*" by McGraw-Hill Book Company – Singapore, 1994.

[2] Richard T. Byerly, Edward W. Kimbark. "*Stability of Large Electric Power Systems*" IEEE press selected Reprint Series, 1974. (Introduction)

[3] P. Kundur, J. Parseba, S. Vitet. "*Overview on Definition and Classification of Power System Stability*" Quality and Security of Electric Power Delivery Systems, CIGRE/IEEE PES International Symposium, 8-10 Oct. 2003.

[4] P. Kundur. "*Definition of Power System Stability*" Draft2, 14 January 2002 (www.yahoo.com)

[5] IEEE Standard 1110-1991. "*IEEE Guide for Synchronous Generator Modeling Practices in Stability Analyses*"

[6] M. Moechtar, T. C. Cheng, L. Hu. "*Transient Stability of Power System - A Survey*" WESCON/'95, Conference record, 7-9 Nov. 1995, pages 166-171 (IEEE web site)

[7] G. G. Karady, G. T. Heydt, P. Crossley, Manfred Michel, Hugh Rudnick, Shinichi Iwamoto. "*Review of Electric Power Engineering Education Worldwide*". Power Engineering Society Summer Meeting, 1999. IEEE , Volume: 2 , 18-22 July 1999, pages 906-915

[8] Si-Leong Chan, Chu-San Lei. "*Study on Building Digital Power System Simulator-Matlab/Simulink based Transient Analysis and Simulation of Synchronous Generators*" Power System Technology Proceedings. 2002. Volume: 3 , Pages:1558 – 1562

[9] S. Muknahallipatna, S. Logowski, Sadrul Ula, Jason Kopas. "*Power System Transient Stability Analysis Software Tool foe an Undergraduate Curriculum*" John Wiley & Sons, Inc. Computer Applications in Engineering Education, Volume 9: pages 37-48, 2001

[10] B. M. Weedy, B. J. Cory. "*Electric Power Systems*" Forth Edition by John Wiley & Sons Ltd. England. 1998

[11] Arne T. Holen. *"Lecture notes-Power System Analysis"* Post Graduate Diploma in Electric Power Systems, NTNU, Trondheim, Norway (unpublished).

[12] L. P. Singh. *"Advanced Power System Analysis and Dynamics"* by John Wiley & Sons Eastern Ltd. New Delhi, India. 1983.

[13] J. C. Das. *"Power System Analysis, Short-Circuit, Load Flow and Harmonics"* by Marcel Dekker, Inc. New York, 2002.

[14] Maurice H. Kent, Wayne R. Schmus, Francis A. McCrackin, Luther M. Wheeler. *"Dynamic Modeling of Loads in Stability Studies"* IEEE Transactions on PAS, May 1969, pages 139-146

[15] IEEE Task Force on Load Representation for Dynamic Performance. "Standard Load Models for Power Flow and Dynamic Performance Simulation" IEEE Transactions on Power Systems, Vol. 10, No. 3, August 1995, pages 1302-1313.

[16] Prabha Kundur. *"Power System Stability and Control"* EPRI Power System Engineering Series, McGraw-Hill, Inc. Publication U.S.A. 1994

[17] W. W. Price, K. A. Wiragu, A. Murdoch, J. V. Mitschi, E. Vaahedi, M. A. El-Kady. "Load Modeling for Power Flow and Transient Stability Computer Studies" IEEE Transactions on Power Systems, Vol. 3, No.1, Feb. 1988, pages 180-187

[18] P. M. Anderson, A. A. Fouad. *"Power System Control and Stability"* IEEE Press Power Engineering Series, John Wiley & Sons, Inc. Publication, 2003

[19] Vincent Del Toro. *"Electric Power Systems"* by Prentice Hall, Inc. New Jersey. 1992

[20] Mariesa Crow. *"Computational Methods for Electric Power Systems"* by CRC Press. 2003

[21] www.mathworks.com. *"Support/Documentation/ Matlab"*

VITA AUCTORIS

NAME	:	N. Sandya Kumari Jayasekara Menike	
DATE OF BIRTH	:	19th April 1964	
PLACE OF BIRTH	:	Kegalle, Sri Lanka	
EDUCATION	:	1973-1982	Pinnawala Central College Pinnawala, Kegalle, Sri Lanka
		1984-1990	University of Peradeniya Peradeniya, Sri Lanka (B.Sc. in Electrical & Electronic Engineering)
		1998-1999	Norwegian University of Science & Technology Trondheim, Norway (Post Graduate Diploma in Electrical Power Systems)
		2002-2004	University of Windsor Windsor, Ontario, Canada (M.A.Sc. in Electrical Engineering)
WORK HISTORY	:	1990-1991	Instructor, Electrical & Electronic Laboratory University of Peradeniya Peradeniya, Sri Lanka
		1991-2001	Electrical Engineer Lanka Electricity Company (pvt)Ltd. Colombo, Sri Lanka