January 2015

Visualization And Mining Of Phasor Data From Optimally Placed Synchrophasors In A Smart-Grid

Ranganath Vallakati

VISUALIZATION AND MINING OF PHASOR DATA FROM OPTIMALLY PLACED SYNCHROPHASORS IN A SMART-GRID

by

Ranganath Vallakati
Bachelor of Technology, Jawaharlal Nehru Technological University, 2012

A Thesis

Submitted to the Graduate Faculty

of the

University of North Dakota

in partial fulfillment of the requirements

for the degree of

Master of Science

Grand Forks, North Dakota
August
2015

This thesis, submitted by Ranganath Vallakati in partial fulfillment of the requirements for the Degree of Master of Science from the University of North Dakota, has been read by the Faculty Advisory Committee under whom the work has been done and is hereby approved.

Prakash Ranganathan, Ph.D., Chairperson

Saleh Faruque, Ph.D.

Arash Nejadpak, Ph.D.

This thesis meets the standards for appearance, conforms to the style and format requirements of the Graduate School of the University of North Dakota, and is hereby approved.

Wayne Swisher,
Dean of the Graduate School

Date

PERMISSION

Title	Visualization and Mining of Phasor Data from Optimally Placed Synchrophasors in a Smart-Grid
Department	Electrical Engineering
Degree	Master of Science

Ranganath Vallakati
July 2015

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

First of all, I want to thank my parents, Bhaskar and Umadevi Vallakati and my sister Swetha for supporting me throughout my life for my happiness. I am forever indebted to them.

I would like to express my sincere gratitude to my advisor, Dr. Prakash Ranganathan for his continuous guidance, support and advice which he had provided from the day 1. I'm very fortunate to have him as my advisor which happened accidentally. This is where I felt *"Everything Happens for a Reason…"*

I am thankful to the members of my committee, Saleh Faruque and Arash Nejadpak for their acceptance of being in my committee as well as their guidance. I would also like to acknowledge the role of the Department of Electrical Engineering at University of North Dakota (UND), Grand Forks, North Dakota for providing me the opportunity to study and conduct research.

My thanks also goes to my friends and colleagues here in Grand Forks for supporting me both academically and socially. They gave me a whole new life.

Above all, I am grateful to the Almighty God for giving me the capability and the chance to finish my thesis.

ABBREVIATIONS

RES	Renewable Energy Sources
SG	Smart Grid
KVL	Kirchhoff’s Voltage Law
KCL	Kirchhoff’s Current Law
DG	Distributed Generation
SE	State Estimation
WAMS	Wide Area Monitoring System
UTC	Universal Time Frame
SCADA	Supervisory Control and Data Acquisition
GPS	Global Positioning System
ISO	Independent System Operator
RTU	Remote Terminal Units
PMU	Phasor Measurement Unit
PDC	Phasor Data Concentrator
IEEE	Institute of Electrical and Electronics Engineers
AEP	American Electric Power
OPP	Optimal placement Problem
SORI	System Observability Redundancy Criteria
ORC	Optimal Redundancy Criterion

SQL	Structured Query Language
DBSCAN	Density-based spatial clustering of applications with noise
DENCLUE	Density based Clustering
CURE	Clustering using representatives
OPTICS	Ordering Points to Identify the Clustering Structure
CLARA	Clustering Large Applications
CLARANS	Clustering Algorithm based on Randomized Search
GRIDCLUS	Grid based clustering
STING	Statistical information grid based
CLIQUE	Clustering in quest
ENCLUS	Entropy clustering
OPTIGRID	Optimal Grid Clustering
db	Data base
SRIPG	Southern Region Indian Power Grid
AMPL	Advanced Modelling and Programming Language

ABSTRACT

Synchrophasors, or also known as Phasor Measurement Units (PMUs), are the state-of-the-art measurement sensor that gather key sensor parameters such as voltage, frequency (f), current (i), and phase angle (ϕ) to monitor the state of an electric grid. The significant feature of a synchrophasor is in its ability to provide real-time streaming data from smart grid. The sampling rate of PMUs ranges from 30 samples to a maximum of 120 samples per second. With such large date-rate, the operations of the power-grid is known with high granularity. However, utilities face certain challenges with synchrophasor measurements. One of the common challenge with synchrophasor is the selection of location to place them in the grid. A synchrophasor placed on a bus is capable of measuring currents, voltages, phasor and frequency information on the entire transmission line incident to that bus. Furthermore, neighboring buses also become observable (i.e. adjacent bus voltage equations are solvable) using Ohm's law, Kirchhoff's Voltage Law (KVL) and Kirchhoff's Current Law (KCL). Thus, it is not necessary to place PMUs on every single bus of the power-grid.

Synchrophasors are expensive units and depending on vendor type, the number of measurement channels and features, the cost per unit can increase. There are several optimal solutions proposed to minimize the cost function to place the synchrophasors. Studies often ignored other metrics such as reliability, and security. This can jeopardize the reliability of the power-grid.

Thus, this thesis work focus on a multi-objective problem that include reliability, cost, energy, and distance. This research proposes a criteria called as Optimal Redundancy Criterion (ORC) based on Linear Programming (LP) methods to find an optimal solution for the placement problem. Although, synchrophasors provide real-time information about the grid, the system operators need to identify, classify and analyze fault or anomalies in the power-grid. Such detection of the faults will improve the situational awareness of the power-grid. This research addresses such challenges by developing data mining algorithms for effective visualization and control of data. The secondary goal is accomplished by implementing a Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm to IEEE test system and phasor data from openPDC framework. The scalability and decision making process for large scale utility test systems using DBSCAN is also investigated.

CHAPTER 1

INTRODUCTION

The dependency of human lives on electrical power is increasing exponentially. Our lives have become so much reliant on electricity that a brownout situation in the grid could mean life and death conditions. Thus, a stable power-grid is important. Electricity possess two important qualities. They are reliability and continuity. The goal of an electric utility is to supply power continuously and reliably. We need a reliable electricity rather than electricity without reliance. Continuity and reliability go hand-in-hand, one can be achieved only, if the other is achieved. With increased injection of Renewable Energy Sources (RESs) or Distributed Generations (DGs), various problems are showing up more often than normal. The problems include deterioration in grid characteristics such as power quality, supply efficiencies, and reliability. Re-routing of excess power becomes a concern due to lack of transporting the energy infrastructure. This is due to the fact that existing transmission and distribution infrastructures are designed only for uni-directional power flows rather than bi-directional power flows. This means that power flow starts from high voltage lines to medium voltage lines and then to low voltages distribution lines. As penetration of DGs increases, the current unidirectional flow infrastructure doesn’t work due to the power following into the low voltage lines from the DGs. Thus, an effective management, monitoring and forecasting methods are needed in order to maintain the stability of the grid. A Smarter-Grid provides management solutions for a stable operation.

A Smart-grid (SG) is simply a conventional power grid that is integrated with intelligent sensing, and software technologies. SG contains, automated sub-stations, switch gears, sensors for intelligent monitoring and control, technologies for improvising generator, and load efficiencies that ultimately keep the power system running in a stable conditions. Figure 1 show the complete framework for a Smart-grid [1]. It contains conventional sources such as coal fired power plants and DGs such as wind farms and solar panels. There are intelligent monitoring sensors on transmission lines, buildings and homes at DG locations. Sensors can take decisions on their own and are needed to be well-connected for better monitoring and situational awareness. Data sensed in a SG are collected for various functionalities such as real-time monitoring, off-line analysis, load flow studies, generation and load shedding schemes.

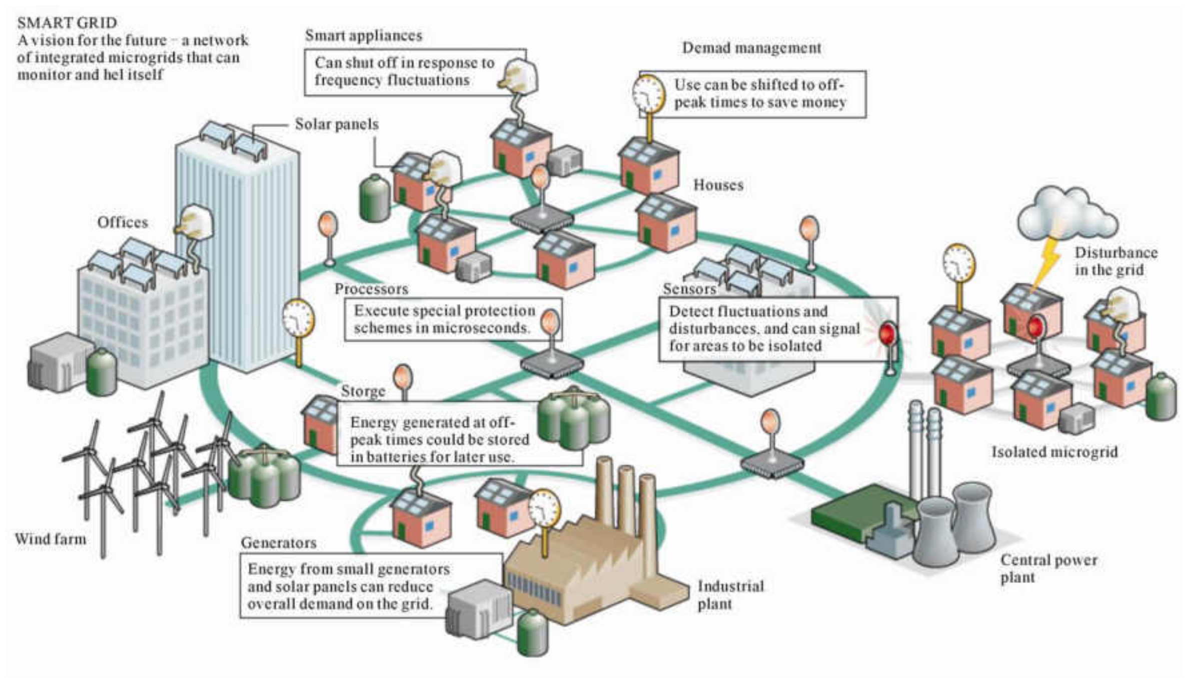

SMART GRID
A vision for the future – a network of integrated microgrids that can monitor and hel itself
Smart appliances
Can shut off in response to frequency fluctuations
Demad management
Use can be shifted to off-peak times to save money
Solar panels
Houses
Offices
Disturbance in the grid
Processors
Execute special protection schemes in microseconds.
Sensors
Detect fluctuations and disturbances, and can signal for areas to be isolated
Storge
Energy generated at off-peak times could be stored in batteries for later use.
Isolated microgrid
Wind farm
Generators
Energy from small generators and solar panels can reduce overall demand on the grid.
Industrial plant
Central power plant

Figure 1 Smart-Grid Network [1]

The electric utilities currently use Supervisory Control and Data Acquisition (SCADA) for controlling and monitoring operations in power systems. There are several drawbacks with existing SCADA systems. In the SCADA system, data samples of voltage, current magnitudes, and phase angle parameters are estimated than measured. The magnitudes were not time-synchronized, so difficult to relate to timing and occurring of an event. The limitations of SCADA system are overcome with the invention of Phasor Measurement Units by Dr. Arun G. Phadke and Dr. James S. Thorp in 1988. Synchrophasors are the modern-day measurement tools that gather key sensor parameters such as voltage, frequency and phase angle information which can be utilized to monitor the power grid [2]. With these sensors, each and every detail of the grid is observed which would assist system operators to supply reliable electricity. Synchrophasors provide time-synchronized measurements of the power system such as voltage and current phasors used to calculate active power, reactive power, and frequency. The data then utilized for applications such as State Estimation (SE), transient analysis, capacitor bank monitoring, load shedding schemes, Inter-area oscillations, and micro-grid operations [3], [4]. Typical micro grid operations include islanding and anti-islanding, and identifying fault locations on transmission lines [5]–[7].

1.1 Synchrophasors

To avoid blackouts, parameters such as frequency, voltages, currents, and phase angles are analyzed with respect to time and location of occurrence. The key advantage of synchrophasor is its ability to time stamp the measured data with a high data rate. Synchrophasors utilize Global Positioning System (GPS) time clocks in order to time stamp their measurements. Synchrophasors helped Independent System Operators (ISO) in realizing the Wide Area Measurement Systems (WAMS). WAMS are an essential an upgrade for SCADA systems by using synchrophasors to monitor power system conditions over large geographical areas with an aim to stabilize the grid. The recent model of synchrophasors has a sampling rate of around 240 samples per second. This level of high-granularity is much required because the quantities such as frequency, and phasor values vary at a sub-seconds time-intervals. ISO utilize synchrophasor data collected from different geographical locations at the same time and take critical decisions in real-time to maintain the grid. Figure 2 represents the WAMS framework.

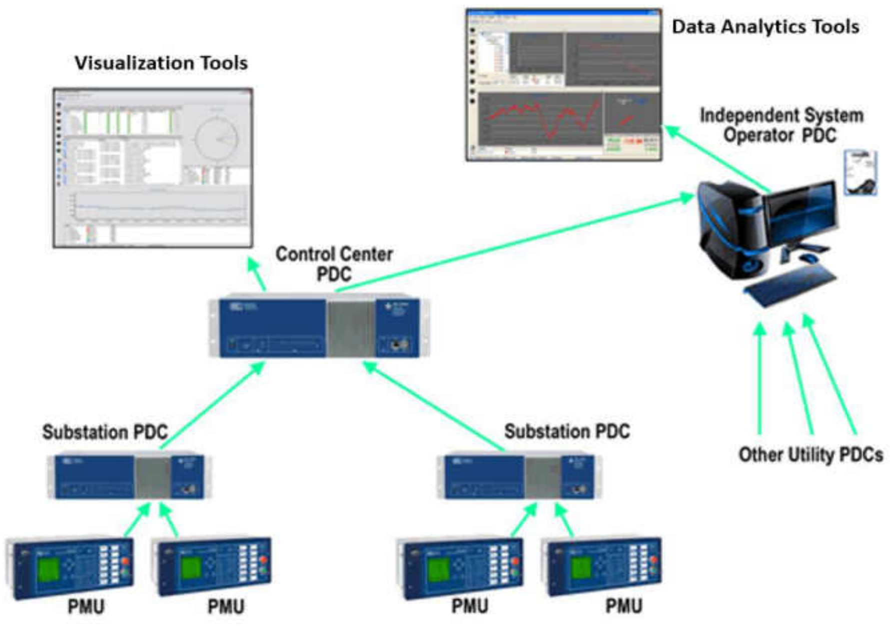


Figure 2 Wide Area Monitoring System (WAMS)

There are different components which form a WAMS network as shown in the Figure 2. They are as follows:

1. Phasor Measurement Unit (PMU)
2. Phasor Data Concentrator (PDC)
3. GPS Clock

PMU uses the most common technique for determining the phasor representation of a signal i.e. by taking the samples from the waveforms and then applying the Discrete Fourier Transform (DFT) to compute the phasors [8]. The AC waveforms are digitized by an Analog to Digital converter for each phase. A Phase-lock oscillator along with GPS source provides the needed synchronization for the samples.

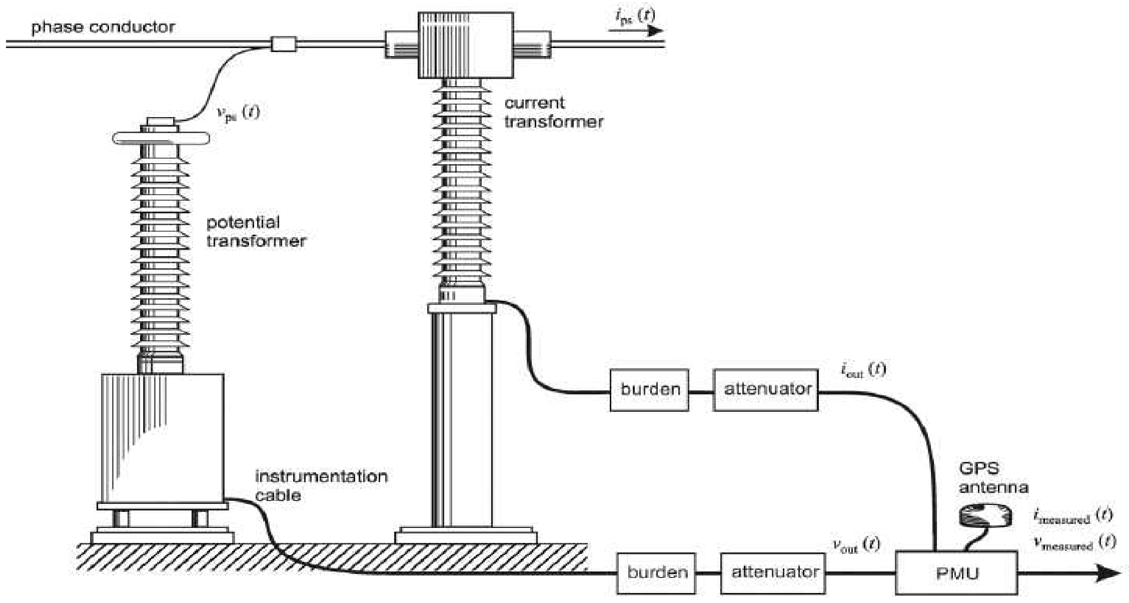


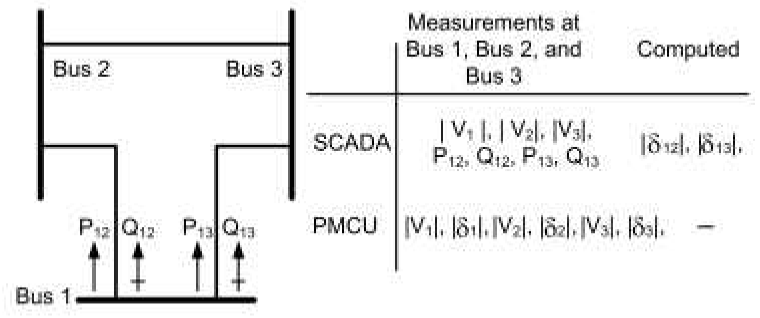
Figure 3 Synchrophasor Hardware at Ground Level

Waveforms are obtained through a line of components such as transformers and filters as shown in Figure 3. These measured signals are time synchronized using the GPS clock. The GPS clocks allow synchronization of signals into an Universal Time Frame (UTC) to assist the wide area system operations. The time synchronized phasor measurements are aggregated in Phasor Data Concentrator (PDC) as per IEEE Standard C37.118 [9]. PDC acts as a centralized main-stream data server that collects and stores the streaming information from various synchrophasors across the grid as per IEEE Standard C37.244 [10]. The data stored can then be extracted and used for various applications.

1.2 Advantages of Synchrophasors over SCADA Measurements

SCADA provides information about voltage magnitude, active and reactive power flows, load measurements, generator outputs, the breaker status, and performances of capacitor banks and transformers as calculated estimates from SE. When power system state is changing quickly, conventional SE fail to capture certain situations, as SCADA scan is inefficient.

In addition, analog measurements are proportional to the time differences between the measurements and the rate at which states of system change [11], [12]. With synchronized measurements, variables such as phase angles need not be estimated as PMU capture all measurements that are synchronous to universal time frame. Also, the quickly changing states can be captured for accurate SE. Figure 4 provides a comparative analysis of SCADA and PMU measurements [11].



	Measurements at Bus 1, Bus 2, and Bus 3	Computed
SCADA	$\|V_1\|$, $\|V_2\|$, $\|V_3\|$, P_{12}, Q_{12}, P_{13}, Q_{13}	$\|\delta_{12}\|$, $\|\delta_{13}\|$,
PMCU	$\|V_1\|$, $\|\delta_1\|$, $\|V_2\|$, $\|\delta_2\|$, $\|V_3\|$, $\|\delta_3\|$,	–

Figure 4 SCADA Vs PMU Measurements

1.3 Challenges with Synchrophasors Measurements

1.3.1 Economics of Synchrophasor Units

PMUs deployed across utilities to improve the SE of power system. However, the major constraint for utilities is the cost of PMUs. In real-world applications, utilities need to setup a number of things before PMUs can be placed on the grid. These include but are not limited to construction of communication infrastructure, procurement of PMUs, phasor data concentrators (PDC) for data aggregation, and synchronization clocks. Thus, it is an expensive issue for utilities to upgrade their monitoring systems. Table 1 shows the cost of installing a Synchrophasor [13]. The total cost is approximately US$30,000 excluding any infrastructure, operational, and labor costs. However, devices such as PDC may collect data from a number of

PMUs. Therefore, a small or mid-size utility can install only a few PDCs, since a typical PDC can handle data up to 40 PMUs.

Table 1 Typical Approximation of Synchrophasor Costs

Type of Equipment	**Cost (US$)**
PMU with protection, automation & controls	~15,000
PDC (up to 40 PMU's)	~ 8,000
Synchronization clocks	~ 2,000
Digital equipment (firewalls, cables, and routers)	~ 5,000

However, there's a solution. By applying the fundamental Ohm's law on the transmission lines, a PMU is placed on a bus allows for observation of the neighboring buses; i.e., adjacent bus voltages are easily computable. Thus, the placement of PMUs in a grid became a challenge for the utilities so as to reduce the cost as well as improve the power system situation to maintain reliability and continuity in the system. This problem of reducing the cost factor as well as to increase the reliability of the grid using PMU placement is addressed in Chapter 2 by using the method of Linear Programming (LP).

1.3.2 Reliability of Synchrophasor Data

With the objective of PMU placement problems is to minimize cost, optimization methods s would try to make the buses measurable or observable as less as possible number of times thus, the system remains without redundancy. During contingency situations such as failure of a PMU or a communication line the system loses its observability and the system becomes unobservable affecting operators ability in decision making. Thus, optimal methods for PMU placement problem should include the concept of redundancy in its measurements to keep the system stable during contingencies. This research proposes a criterion for including such a concept of redundancy into the PMU placement algorithm.

1.3.3 Utilization of Synchrophasor Data

Synchrophasors generate around 30 to 120 samples per second that contain information about the state of power system. Thus, ISO are bombarded with nearly 108,000 samples/hour, which equates to 25,92,000 samples/a day or 7,76,60,000 samples a month. These numbers grow exponentially and approximately around 1.5 Tera Bytes (TB) of data is accumulated in a month's span. With all such big data accumulated, ISO make use of the data for meaningful applications. It is neither possible to monitor every data point nor unmonitored. A visualization technique had been developed using an unit-circle representation of phasor data. Unit-circle visualization type will help ISO to monitor the incoming data with ease of understanding. The data representation is stored in the database (db) prior to data mining tasks. ISO uses synchrophasor data for predictive analytics to forecast the system behavior. Any forecasting task require knowledge of past history of data. Data Mining (DM) is the process of extracting useful information to understand more meaningful interpretation about the system. A density based clustering algorithm called as Density-Based Spatial Clustering of Applications with Noise, (DBSCAN) is used to cluster the synchrophasor data. The results of clustering is stored in the db after the initial stage of visualization.

1.4 Thesis Contributions

The following are the key contributions of this research work.

1. Identifying a fastest and an efficient way to solve the OPP of PMUs by including a reliability constraint in terms of Optimal Redundancy Criterion (ORC). The ORC is a novel metric that assists ISO in keeping the grid reliable.
2. Test the scalability of LP formulations in large test systems such as IEEE 14, 30, 57, 118, 300 and SRIPG 208 bus systems.
3. Development an Unit Circle Representation for the phasor data for visualizing SG data.
4. Implementation a clustering technique to detect anomalies in the grid. This technique will capture different types of fault scenarios such as steady state, heavy-load, light-load conditions.

I strongly believe that my contributions will assist ISO in making critical decisions that increases situational awareness and the reliability of modern power-grids alias Smart-grids.

1.5 Thesis Organization

The thesis is organized as follows. Chapter 2 introduces the optimal placement problem followed with a review of available solutions to the problem. A solution based on Linear Programming (LP) is modeled in the thesis using Advanced Modeling and Programming Language (AMPL) with an aim to keep the power system completely observable enforcing a reliability constraint called as Optimal Redundancy Criteria. Chapter 3 describes recent work published on literatures on the various visualization and clustering techniques. The development of an unit-circle representation of the phasor data and its application of DBSCAN clustering technique on synchrophasor data is also described. Chapter 4 discusses solutions obtained from optimal placement problem with various contingency situations taken into consideration. It also discusses the benefits obtained from the unit-circle representations and the DBSCAN clustering algorithm followed by the Conclusion.

CHAPTER 2

OPTIMAL PLACEMENT OF PHASOR MEASUREMENT UNITS (PMU)

PMUs provide time-synchronized voltage and current measurements when placed on a bus. These time-synchronized values are critical for continuous operation of Wide Area Monitoring Systems (WAMS) in the smart-grids. As discussed in Chapter 1, it is not required to place a PMU on every bus in the system to get full observability. Using Ohm's law, a PMU placed on specific bus can be used to measure the voltage and current phasors at that point. It can also be used to calculate the voltage and current phasors on the lines that are connected to the placed bus. Also, it is not feasible for the utilities to afford a PMU for every bus in their grid due to its costs. Thus, there is a need to find a solution for Optimal Placement Problem (OPP) for PMUs.

2.1 Need for Optimal Placement Problem for Synchrophasors

Figure 5 4-Bus System show various cases of placing the PMUs on a 4-bus system.

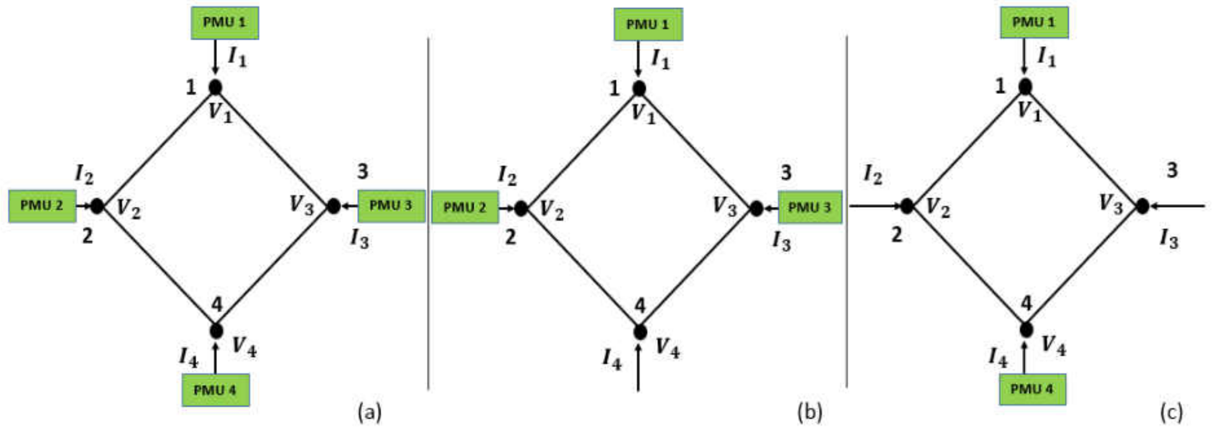


Figure 5 4-Bus System

Case 1: In Figure 5(a), all four bus points are accompanied with PMUs for complete observability. The situation is shown in Table 2. There exist many solutions of redundant observabilities than required. Thus, it would be useless to have a PMU on every bus.

Case 2: The Figure 5(b) show how three PMUs on buses 1, 2, and 3 are sufficient to cover the full-observability of system. This situation is given in Table 2. There's a reduced redundancy in the measurements but the system is yet fully-observable.

Case 3: Figure 5(c) show a 4 bus system with only two PMUs placed on the buses 1 and 4. By analyzing case 3 column of Table 2, we can say that the values at the buses 2 and 3 without the PMU can be calculated using the line parameters (R, X) and measured parameters at buses 1 and 4. In this way, even with two PMUs, the system remains fully-observable.

Table 2 Case Studies of PMU Placement on Four Bus System

		Case 1	**Case 2**	**Case 3**
Bus 1	Measured	PMU 1	PMU 1	PMU 1
	Calculated	PMU 2, PMU 3	PMU 2, PMU 3	-
Bus 2	Measured	PMU 2	PMU 2	-
	Calculated	PMU 1, PMU 3	PMU 1, PMU 3	PMU 1, PMU 4
Bus 3	Measured	PMU 3	PMU 3	-
	Calculated	PMU 2, PMU 4	PMU 2	PMU 1, PMU 4
Bus 4	Measured	PMU 4	-	PMU 4
	Calculated	PMU 2, PMU 3	PMU 2, PMU 3	-

Thus, we conclude that the placing the PMUs using known topology in a certain manner could provide complete observability of system with cost savings that include operational costs and hardware (equipment and infrastructure) costs. In addition, this research includes the zero-injection buses of the power systems to further reduce the number of PMU required for full observability. Zero-injection buses are similar to transshipment nodes in the OPP algorithm that reduce the minimum number of PMUs. Details about zero-injection buses are explained in later section of the thesis. An index called as Optimal Redundancy Criterion (ORC) is proposed to address the issue of reliability that is very important for the power systems. ORC assist in identifying the best solution for OPP, while enforcing some conditions to maintain redundancy and full-observability of critical buses.

2.2 Literature Review

Various approaches have been proposed to find optimal solutions for the OPP of PMUs [14-34]. In [14], [15], the solution for OPP are found through a dual search algorithm which uses both a modified bisecting search and a simulated-annealing-based method. The bisecting algorithm fixes the number of PMUs, while the latter looks for an optimal solution for the fixed number of PMUs. [16] provides a solution based on Simulated Annealing technique where a novel concept of depth of un-observability is introduced and its impact on the number of PMU placements is explained by drawing the spanning trees for the power systems. A Graph theory approach is proposed to solve the problem of placement in [17]. Reference [18] proposed an easy way to find the solution for placing the PMUs using a generalized Integer Linear Programming (ILP). The problem of minimizing PMUs was considered as an objective using topology based constraints were formed to obtain an ILP based solution for the OPP. A specially tailored non-dominated sorting genetic algorithm for the OPP is proposed with two conflicting objective functions such as maximization of measurement redundancies and minimization of PMUs are considered in [19]. [20] proposes a solution for OPP using immunity genetic algorithm. A modified binary particle swarm optimization (BPSO) is used as an optimization tool for obtaining the minimal number of PMUs with the help of topological observability rules in [21]. Using similar BPSO technique, a solution for OPP is proposed by considering the factor of cost for the installation of PMU as an additional objective to minimize the number of PMUs in [22]. In [5], a scheme is proposed to detect the fault locations on the transmission line in addition to the complete observability of the system with PMUs. [23] proposes a novel method for a power system that is suffering from random outages. The minimal set of PMUs using this novel method is obtained with an aim to minimize the state estimation error covariance. Using the similar ILP method proposed in [17], a method is proposed in [24] to further reduce the minimal set of PMUs by considering zero-injection buses as constraints. Also, two indices namely, Bus Observability Index (BOI) and System Observability Redundancy Index (SORI) were proposed to choose the best solutions, if multiple solutions are available. For utilities with economic limitations, a multi-stage placement problem has been formulated in [24], [25]. In [26], a ILP methodology is proposed for multistaging of PMU placement in a given time horizon, while in [25], a mixed-integer programming is proposed for PMU placement problem in a multi-year planning horizon along with network expansion scenarios. [27] proposed a pre-processing method that includes where real power system conditions. A national inter-connected power system is used to formulate the practical conditions and to find the solution for OPP. An Iterated Local Search (ILS) metaheuristic approach is proposed in [28] to minimize the size of the PMU configuration required for a given system. [29] presents an unified approach using Binary ILP approach to find solution for OPP by integrating the impacts of conventional measurements present in the system. Using similar ILP methodology and by considering conventional measurements, an approach is developed in [30] to find the optimal placements during all possible line and measurement contingencies. [31] proposes an information-theory approach that presents a mutual information criterion between PMU measurements and its power system states. An unconstrained non-linear weighted least squares (WLS) approach is proposed in [32]. A solution for OPP is proposed in [7] by considering a system that needs to be observable both under normal conditions and islanding conditions. In [33], a reliability based incremental OPP problem is proposed by evaluating the proposed reliability indices of node and system observability. A placement scheme that ensures the observability of critical buses is proposed in [34]. Here, critical buses are differentiated from the non-critical buses by considering high voltages, and topological connectivity.

2.3 PMU Placement Problem Formulation using Linear Programming (LP)

In this research, the OPP of PMU is solved using the Linear Programming (LP) techniques. In any LP method, the problem is defined by having an objective function that maximizes or minimizes a linear function subject to set of linear constraints. The constraints include combination of equalities or inequality constraints. LP models are commonly used for its reliability and finding the optimized solution for the placement problem as our goal is to reduce the minimum number of PMUs without losing the complete observability of the system.

2.3.1 Problem Formulation

For a system with N buses, the OPP of synchrophasors can be formulated as shown in the equation (1).

$$\begin{gathered} minimize \sum_{i}^{N} W_i * X_i \\ Subject\ to. f(X) \geq \hat{1} \end{gathered} \tag{1}$$

Where $\boldsymbol{X_i}$ is a binary decision variable vector, whole entries are defined using the binary variables as follows:

$$X_i = \begin{Bmatrix} 1 & if\ a\ PMU\ is\ installed\ at\ bus\ i \\ 0 & otherwise \end{Bmatrix} \tag{2}$$

$\hat{1}$ is a vector whose entries are all ones

W_i is the cost associated with the installation of PMU at that bus i. Figure 6 shows a single line diagram of an IEEE 14 bus system. There are 5 generation points at buses 1, 2, 3, 6 and 8. Of these, only generators at bus 1 and 2 generate both active and reactive power, while generators at 3, 6, and 8 generate reactive power. There are 11 load points connected at bus 2, 3, 4, 5, 6, 9, 10, 11, 12, 13, and 14 that consume the active and reactive power. Using the LP approach, the system is evaluated to find the PMU placement locations.

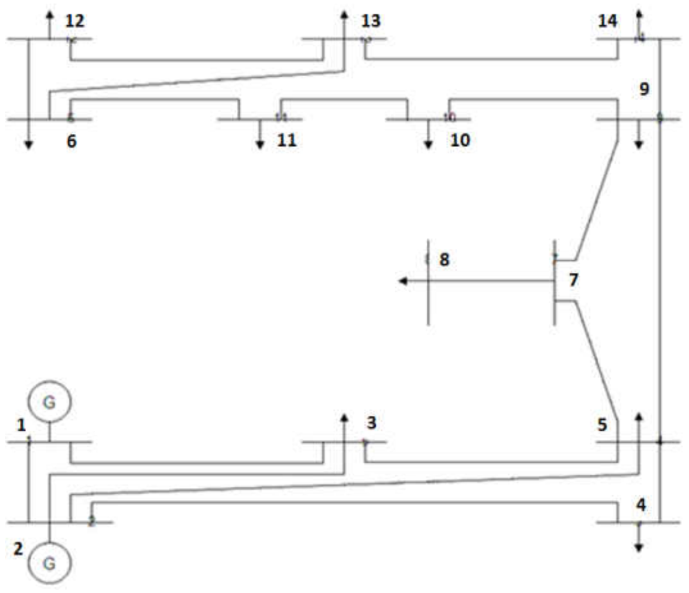


Figure 6 IEEE 14 Bus System

The two cases of the system that is considered to determine the OPP solution are as follows:

A. By considering regular test-bus system with any zero-injection buses

B. By considering zero-injection buses as special constraints.

2.3.1.1 System with No Zero-Injection Buses

Assuming all of the fourteen buses in the IEEE 14 bus system are regular ones having at least a load or a generation point. There is a requirement to measure or observe all the buses because of the changes that happen at each of them due to the generation and load points. A binary connectivity matrix is formed to indicate links between the buses. The entries of $A_{i,j}$ are defined as follows in equation (3):

$$A_{i,j} = \begin{Bmatrix} 1 & if\ i = j \\ 1 & if\ i\ and\ j\ are\ connected \\ 0 & if\ otherwise \end{Bmatrix} \tag{3}$$

Matrix $A_{i,j}$ can be directly obtained from the bus admittance matrix by transforming its entries into binary form as shown in (4):

$$A_{i,j} = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \tag{4}$$

Here, a zero in the connectivity matrix derived from the admittance matrix implies that there is no connection between buses i and j. While one (1) in the matrix implies that there is a connection. Once the binary connectivity matrix is formed, the LP constraints for all the 14 buses from the 14 connections is shown in equations from (5) to (18):

$$Bus\ 1: x_1 + x_2 + x_5 \geq 1 \tag{5}$$

$$Bus\ 2: x_1 + x_2 + x_3 + x_4 + x_5 \geq 1 \tag{6}$$

$$Bus\ 3: x_2 + x_3 + x_4 \geq 1 \tag{7}$$

$$Bus\ 4: x_2 + x_3 + x_4 + x_5 + x_7 + x_9 \geq 1 \tag{8}$$

$$Bus\ 5: x_1 + x_2 + x_4 + x_5 \geq 1 \tag{9}$$

$$Bus\ 6: x_6 + x_{11} + x_{12} + x_{13} \geq 1 \tag{10}$$

$$Bus\ 7: x_4 + x_7 + x_8 + x_9 \geq 1 \tag{11}$$

$$Bus\ 8: x_7 + x_8 \geq 1 \tag{12}$$

$$Bus\ 9: x_4 + x_7 + x_9 + x_{10} + x_{14} \geq 1 \tag{13}$$

$$Bus\ 10: x_9 + x_{10} + x_{11} \geq 1 \tag{14}$$

$$Bus\ 11: x_6 + x_{10} + x_{11} \geq 1 \tag{15}$$

$$Bus\ 12: x_6 + x_{12} + x_{13} \geq 1 \tag{16}$$

$$Bus\ 13: x_6 + x_{12} + x_{13} + x_{14} \geq 1 \tag{17}$$

$$Bus\ 14: x_9 + x_{13} + x_{14} \geq 1 \tag{18}$$

The operator $(+)$ in the formulation is a logical "OR" operation, while the inequality operator $(\geq)$ ensures that at least one of the variables appearing in the sum will be non-zero. For example, consider the following constraints (5) and (6) associated with bus 1 and bus 2:

$$Bus\ 1: x_1 + x_2 + x_5 \geq 1 \tag{19}$$

$$Bus\ 2: x_1 + x_2 + x_3 + x_4 + x_5 \geq 1 \tag{20}$$

The constraint $f_1 \geq 1$, equation (19) indicate that at least one PMU must be placed on at least one of the three buses bus 1 (x_1), bus 2 (x_2), or bus 5 (x_5). Similarly, $f_2 \geq 1$ i.e. the equation (20) implies that at least one PMU should be installed on any one of the buses 1, 2, 3, 4, or 5 in order to make bus 2 observable (i.e., to satisfy f_2 constraint).

2.3.1.2 System with Zero-Injection Measurements

Zero-injection buses are buses where no currents are neither injected nor withdrawn into/from the system (i.e., a buses which have neither load nor generation units). In the power systems, there are few bus locations where there is no requirement for measuring. The use of Kirchhoff's current law (KCL) to include zero-injection buses in modeling the OPP further reduces the number of PMUs required. Each zero-injection node in a system creates an additional constraint. Thus, the number of PMUs required for complete observability of the system can be reduced. For a zero injection bus i, let A_i indicate the set of adjacent buses to bus i. Let $B_i = A_i \cup \{i\}$. Let the number of zero injections be Z. Since there is no change in current (i) values at a zero-injection bus, the variables related to the zero-injection buses are eliminated from all the constraining equations. The model formulation in an ILP framework, therefore, selectively allow some of the buses to be un-observable. However, there are additional constraints that need to be included for a fully-observable system. They are:

- The un-observable buses must be adjacent to zero-injection buses.
- The number of un-observable buses in the set (zero-injection bus and its adjacent buses) is at mostly as one (1) in the matrix.

In the IEEE 14-bus system shown in Figure 6, bus 7 is a zero-injection bus. Thus, and the new constraining equation can be written as,

$$Bus\ 1: x_1 + x_2 + x_5 \geq 1 \tag{21}$$

$$Bus\ 2: x_1 + x_2 + x_3 + x_4 + x_5 \geq 1 \tag{22}$$

$$Bus\ 3: x_2 + x_3 + x_4 \geq 1 \tag{23}$$

$$Bus\ 4: x_2 + x_3 + x_4 + x_5 + x_8 + x_9 \geq 1 \tag{24}$$

$$Bus\ 5: x_1 + x_2 + x_4 + x_5 \geq 1 \tag{25}$$

$$Bus\ 6: x_6 + x_{11} + x_{12} + x_{13} \geq 1 \tag{26}$$

$$Bus\ 8: x_4 + x_8 + x_9 \geq 1 \tag{27}$$

$$Bus\ 9: x_4 + x_8 + x_9 + x_{10} + x_{14} \geq 1 \tag{28}$$

$$Bus\ 10: x_9 + x_{10} + x_{11} \geq 1 \tag{29}$$

$$Bus\ 11: x_6 + x_{10} + x_{11} \geq 1 \tag{30}$$

$$Bus\ 12: x_6 + x_{12} + x_{13} \geq 1 \tag{31}$$

$$Bus\ 13: x_6 + x_{12} + x_{13} + x_{14} \geq 1 \tag{32}$$

$$Bus\ 14: x_9 + x_{13} + x_{14} \geq 1 \tag{33}$$

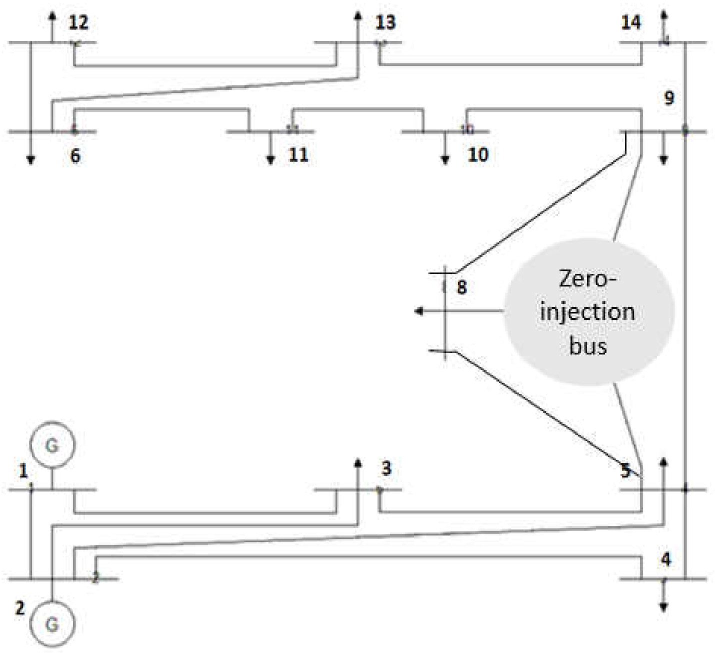


Figure 7 IEEE 14 Bus System with Zero-Injection Bus

The constraints (21) – (33) are the updated (or modified) connections obtained from the bus connectivity matrix formed from the modified 14 bus system as shown in Figure 7. There are only 13 constraints, as the bus 7 which is a zero-injection bus is removed and thereby reduce the number of PMUs. In this way, the number of PMUs required is thus reduced, if a system has zero-injection nodes.

2.4 System Observability Redundancy Index (SORI)

If there are multiple solutions for an OPP placement problem, then identifying an optimal solution is often a challenge. To solve this problem, a SORI index is used [24]. If a bus i is observed by a PMU 'n' number of times, then the SORI (γ) is given by equation (34):

$$\gamma_i = \sum n_i \tag{34}$$

Choosing an optimal solution will provide a maximum redundancy enabling complete observability rather than choosing an optimal solution that might be less redundant. The SORI index help in choosing the best solution. In general, larger redundancy values yield higher SORI values. Thus, maximizing the SORI index is a good measure of success for the OPP placement problem, as the approach keeps system observable during any contingency situations. For example, let us consider the IEEE 14 bus system in Figure 6. The system is made completely observable with 4 PMUs either being placed at 2, 6, 7, and 9 bus locations or at 2, 6, 8, and 9 bus locations. In both the cases, all 14 buses are fully-observed. Table 3 provide a comparison of bus-observabilities for both the solutions. Using equation (34), the SORI index is calculated by simply adding all bus observabilities in a system. In the first case (2nd column of Table 3) when the PMUs are placed at 2, 6, 7, and 9 locations, the SORI index is calculated as follows:

$$SORI = 1+1+1+3+1+1+2+1+2+1+1+1+1+1 = 18 \tag{35}$$

While in the second case (3rd column of Table 3), the SORI index will be:

$$SORI = 1+1+1+2+1+1+2+1+1+1+1+1+1+1 = 16 \tag{36}$$

Thus, even though with same number of PMUs, full-observability is obtained in both the cases, It is recommended that placing the PMUs at buses 2, 6, 7, and 9 would yield better results and better observability. In addition, a complete redundancy is attained which is critical for contingency scenarios or events.

Table 3 Observability Index for Various Placement Solutions in IEEE 14 Bus System

	PMU Placement at Buses : 2, 6, 7, 9	**PMU Placement at Buses : 2, 6, 8, 9**
Bus 1	PMU at bus 2	PMU at bus 2
Bus 2	PMU at bus 2	PMU at bus 2
Bus 3	PMU at bus 2	PMU at bus 2
Bus 4	PMU at buses 2, 7, and 9	PMU at buses 2 and 9
Bus 5	PMU at bus 2	PMU at bus 2
Bus 6	PMU at bus 6	PMU at bus 6
Bus 7	PMU at buses 7 and 9	PMU at buses 8 and 9
Bus 8	PMU at bus 7	PMU at bus 8
Bus 9	PMU at buses 7 and 9	PMU at bus 9
Bus 10	PMU at bus 9	PMU at bus 9
Bus 11	PMU at bus 6	PMU at bus 6
Bus 12	PMU at bus 6	PMU at bus 6
Bus 13	PMU at bus 6	PMU at bus 6
Bus 14	PMU at bus 9	PMU at bus 9

2.5 Optimal Redundancy Criterion (ORC)

Although the OPP problem yields solutions for full-observability, there is a likelihood of uncertainty during contingency situations which may result in partial or zero observability of the system. Contingency here, refers to a situation in the power system where,

- There is a loss of PMU measurements from the observing bus. This would result in the loss of PMU measurements from all the connected buses.

- There is a loss in communication channel that is transmitting information from the observing bus.
- There is a loss of PMU measurements from the observable bus.

The above contingency cases can be caused through physical damage of an equipment, infrastructure, and cyber-physical damages. This means, in certain cases, it is better to include a certain number of buses to be observed by multiple PMUs. Thus, the loss of PMU measurements from one device will not affect the observability of any other bus. The factor of percentage or observability weights of buses can be considered in different ways. Some literature studies consider high voltage buses to be more critical than other buses, while others consider all generating buses to be critical buses [34]. We propose a new index called "Optimal Redundancy Criterion" (ORC) which is based on the number of connections to each bus. ORC is defined as the ratio of the total number of connections between buses to the total number of buses in the system. We can select the buses that have more connections or links by adapting the following criterion (37).

$$ORC = \frac{\sum_{i=1}^{n} \sum_{j=1, i \neq j}^{n} B_{i,j}}{n} \tag{37}$$

Where i and j represent the buses, $B_{i,j}$ represent connections between i and j, and n is the total number of buses in the system. The ORC conditions are enforced as follows:

Step 1: ORC is calculated using the formula provided in (24).

Step 2: Based on ORC, the critical bus constraints are selected from all the constraints.

Step 3: Increase the minimum observability requirement of the selected critical buses.

Let us consider constraints (5) to (18), where each constraint represent a bus in the IEEE 14 bus system. Here, the total number of connections for all 14 buses is the sum of their connections.

$$B_1 + B_2 + \cdots B_{14} = 56 \tag{38}$$

The number of buses are in the test system is. Thus, ORC for the 14 bus system is given by,

$$ORC = \frac{56}{14} = 4 \tag{39}$$

Therefore, from the constraints formed earlier, we consider equations (6), (8), and (13), and increase the weights, as these buses have a greater number of connections than ORC values (five, six, and five connections respectively). The weights in constraints appear on the right hand side of the equation. Now, the updated constraints would be as shown in (40) to (42).

$$Bus\ 2: x_1 + \ x_2 + x_3 + \ x_4 + x_5 \geq 2 \tag{40}$$

$$Bus\ 4: x_2 + \ x_3 + x_4 + \ x_5 + x_7 + \ x_9 \geq 2 \tag{41}$$

$$Bus\ 9: x_4 + \ x_7 + x_9 + \ x_{10} + x_{14} \geq 2 \tag{42}$$

For full-observability, a ”1” is placed on the right-hand side of the constraint equation of each critical bus, indicating that each bus is observed at least once. Based on system and individual utility need or requirements, one can adjust the observability weights. For our test system, we chose to make the critical buses observable twice. This will provide sufficient redundancy, and help increase the reliability of the system. Chapter 4 discusses the various advantages gained in terms of grid reliability while enforcing ORC by using the test case systems.

CHAPTER 3

VISUALIZATION AND CLUSTERING OF SYNCHROPHASOR DATA

This chapter discusses the need of Data Visualization (DV) and Data Mining (DM) for synchrophasor data. The first part of this chapter is dedicated for modeling of a system that mimics synchrophasor data. The second part discusses the circle representation for the synchrophasor data thus obtained from the system and the final part would discuss the data clustering technique, DBSCAN.

3.1 Need for Visualizing and Analyzing the Synchrophasor Data

A basic PMU provide a minimum of 30 samples of data per second. With so many samples per second, the state of the power system at any given time is studied and observed easily. The ISO can actively observe any system anomalies, or outages in the grid, and respond to them in the event of malfunctioning or problems in the power system. Although PMUs provide Situational Awareness (SA) of the system using large number of its samples, the ISO's need to gather, classify, and analyze the streaming PMU data. PMUs with high-sampling data rate, flood the data concentrators with streaming data every sub-seconds. Neither it is possible to monitor each and every data point (~120 samples/second) nor leave the data unmonitored. Thus, visualization forms an important aspect for quick analysis of data. There also needs to be some mechanism which will assist system operators in decision making process to analyze the data from the PMUs thereby keeping the grid online without any un-stable conditions.

Data Mining (DM) or data analysis will assist ISO in performing off analysis of the power system parameters which will impact the future decision-making of the ISO. For my research, to perform the DV and DM, I need to have a synchrophasor data which was practically impossible without having a PMU. Thus, I was in pursuit of synchrophasor data. I was in search of acquiring the source of synchrophasor data with-out having an actual PMU installed in our Data Sciences laboratory. The two sources that I have considered for providing synchrophasor data are:

A. A MATLAB based IEEE 14 bus system model.

A MATLAB based simulink model can provide me data that resembles as real synchrophasor data. Even though, it might not actually represent a PMU, the model provides me the voltage and current phasor values which in general represents the synchrophasor data.

B. A openPDC data source

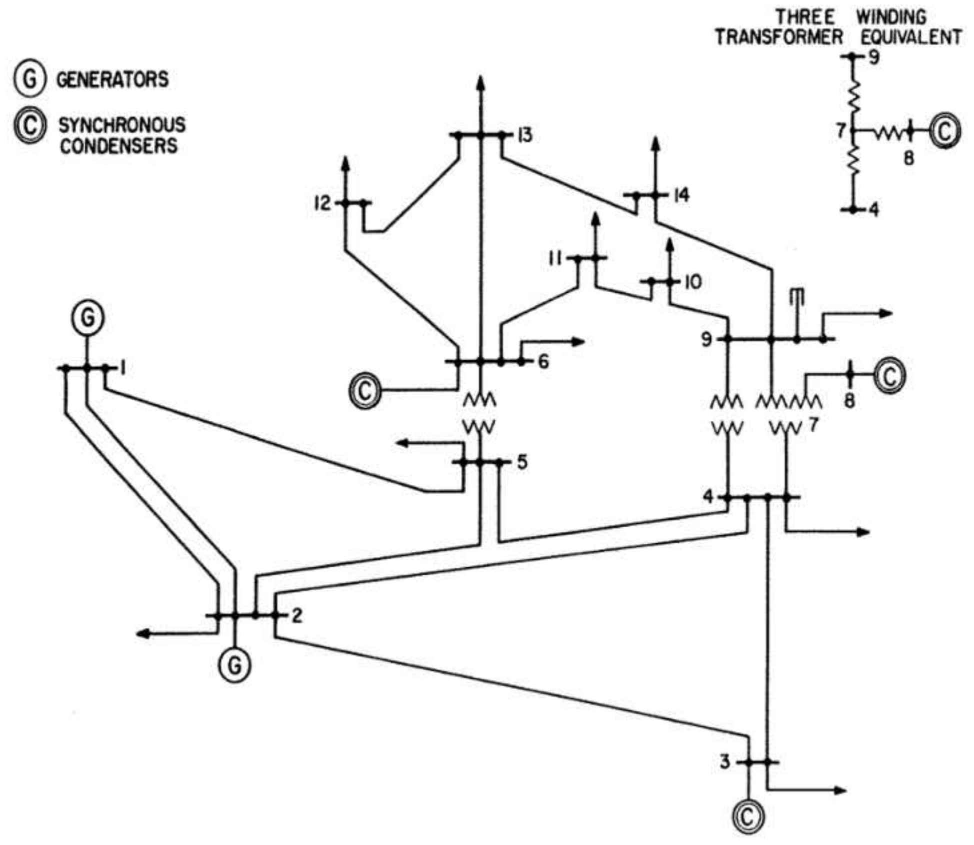


Figure 8 IEEE 14 Bus System

3.2 Modeling of IEEE 14 Bus Test System

The IEEE 14-bus test case represents a portion of the American Electric Power (AEP) System (in the Midwestern US) as of February, 1962. A much-xeroxed paper version of the data was kindly provided by Iraj Dabbagchi of AEP and entered in IEEE Common Data Format by Rich Christie at the University of Washington in August 1993 [35]. Once this was accepted as an IEEE Test Case, it became a standard system for all power system experiments such load flow studies, and economic dispatch problems and is now widely accepted. Basically, the 14 bus system has 14 buses, 5 generators and 11 loads. All generators, loads and its branches are shown in Figure 8. The complete 14 bus system has been modeled in MATLAB/SIMULINK. Also, fault conditions are modeled in the IEEE 14 bus system realize the ideal power system conditions.

3.2.1 IEEE 14 Bus System

Figure 8 show the simulink model of an ideal fourteen bus model in MATLAB using the bus and branch parameters [35]. This model depicts the ideal conditions where there is sufficient supply available to meet the demand units. The model shown in Figure 8 uses the three-phase constant voltage sources as generators. The voltage sources provide power at the specified voltage levels all the time. So, they vary current in order to achieve the required power output. All parameters such generation and load levels, line parameters ($R, C,$ and X), transformer tap ratios are simulated in the 14 bus system.

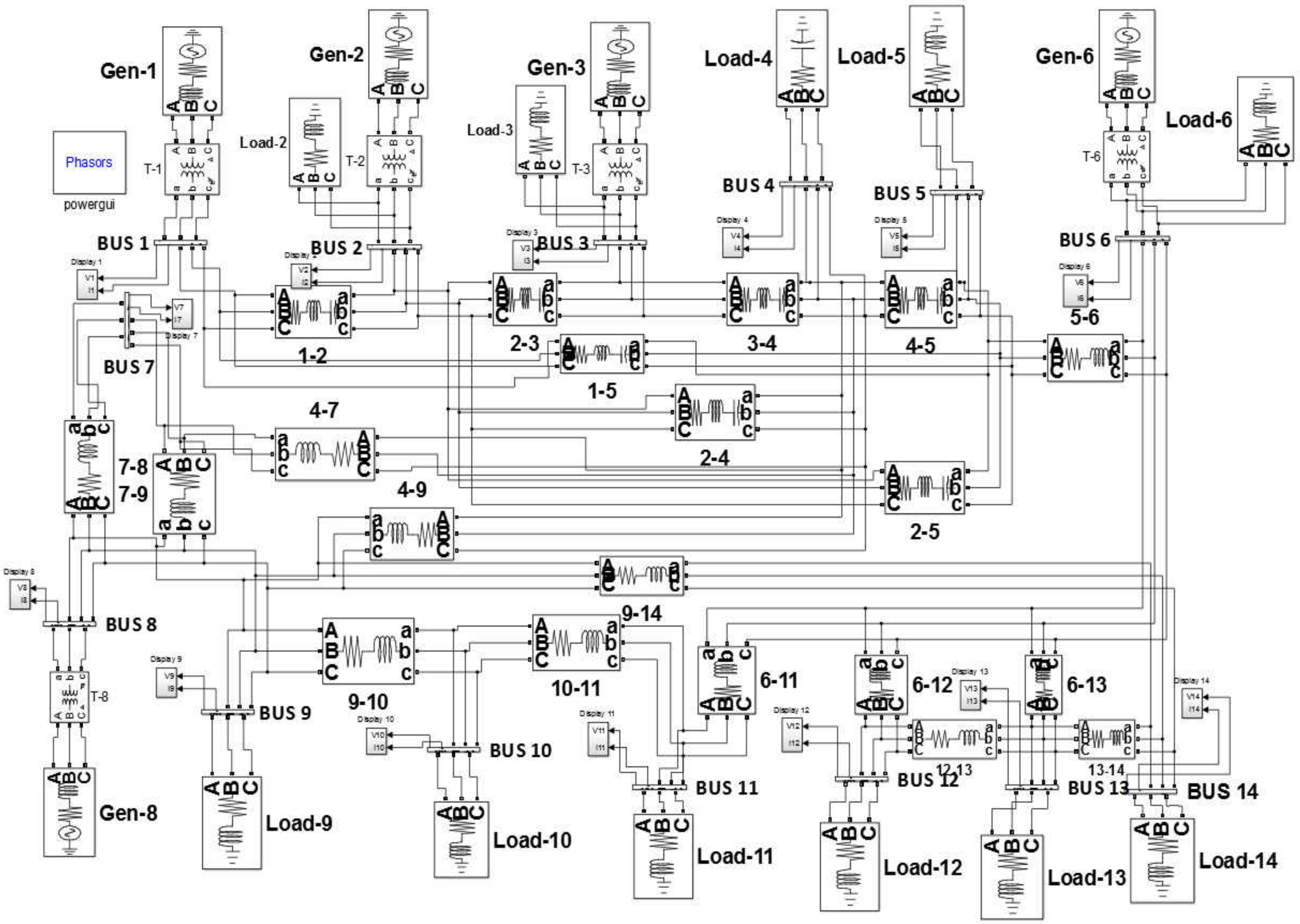


Figure 9 Simulink Model of IEEE 14 Bus System

The voltage and current waveforms at all the buses are observed individually using scope block in MATLAB. The two waveforms in Figure 10 represents voltage and the current plots at a zoomed in level. The three phase voltages and currents at bus 1 are represented with three colors, they are: red, yellow, and green. The X-axis is shown in time (in seconds) and the Y-axis shown in voltage and current magnitudes respectively. As this system is constructed in per unit (p.u.) system representing an ideal case scenario, the voltage always remains to be constant at 1 p.u. and the currents are at very low-level with respect to the voltage values.

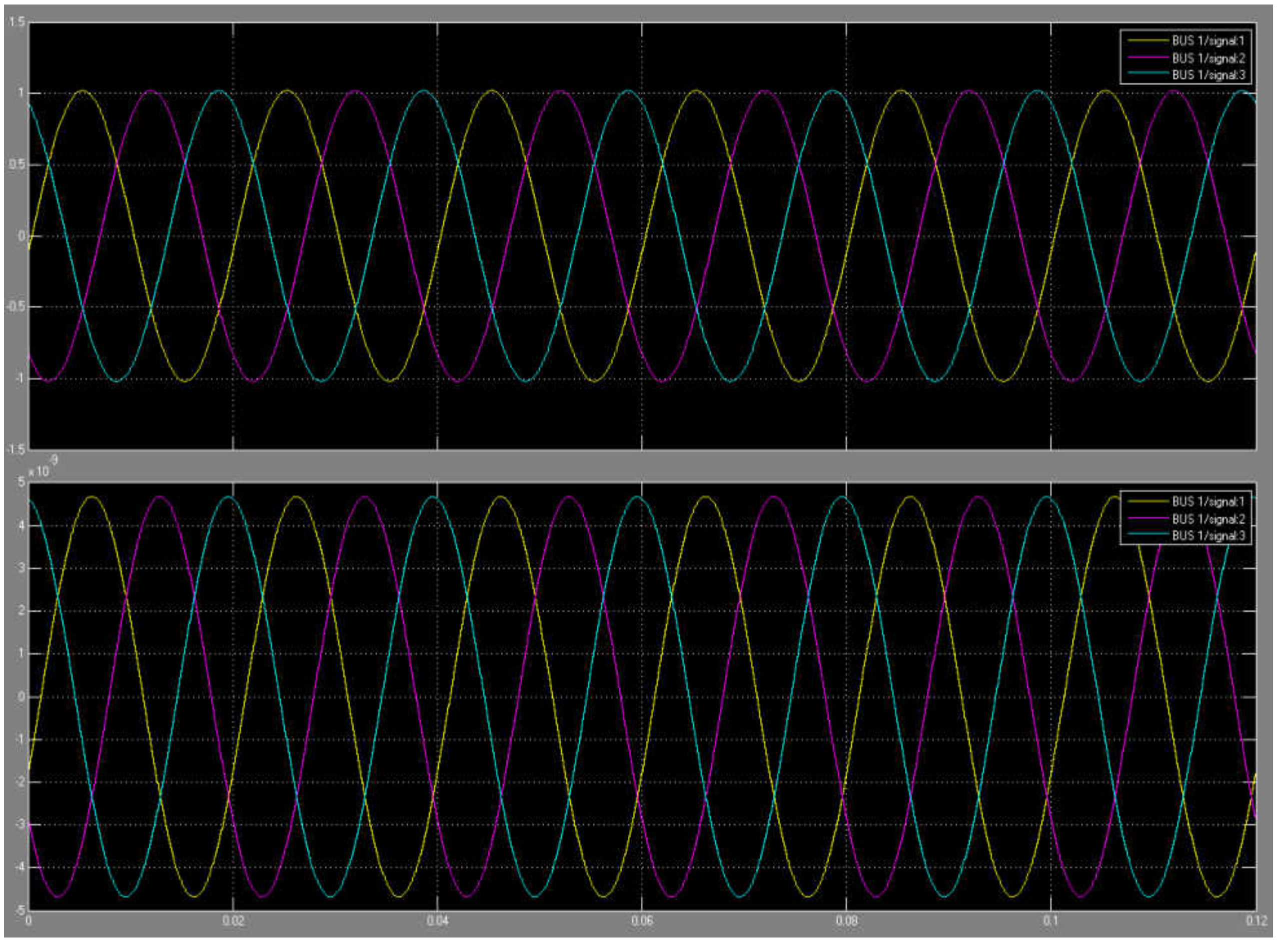


Figure 10 Voltage and Current Waveforms at Bus 4

Table 4 3-ϕ Voltage Magnitudes and Phase Angles

A - phase magnitude	**B - phase magnitude**	**C- phase magnitude**	**A - phase angle**	**B - phase angle**	**C - phase angle**
1.020223859	1.02022386	1.020223859	-5.908535586	-125.9085356	114.0914644
1.020223859	1.02022386	1.020223859	-5.908535586	-125.9085356	114.0914644
1.020223859	1.02022386	1.020223859	-5.908535586	-125.9085356	114.0914644
1.020223859	1.02022386	1.020223859	-5.908535586	-125.9085356	114.0914644

3.2.2 IEEE 14 Bus System with Faults

This case represents a 14 bus system with fault conditions. The realistic model is built using few generator blocks and load points that are added and removed to some specific buses during the simulation time. Figure 11 represents the 14 bus system with faults. The model described is similar to original IEEE fourteen bus system except with the following additions

1. A random generator included at bus 4 in addition to the existing load.
2. A random load attached at bus 3 in addition to the existing load.
3. A three-phase fault modeled on a branch between bus 4 and bus 9.

This fault is a three-phase-to-ground fault that happens to occur during the time frame of 0.6 to 0.8 seconds. Figure 12 shows the scope of waveforms from the bus 4. In comparison with the Figure 8, the differences can be easily seen. The changes from ideal waveforms are due to the additional conditions modeled in this case. During the ideal case, the voltages were always at 1 p.u, while during the case of fault (Figure 12), voltage reduces while the current maximizes during 0.6 to 0.8 seconds of fault condition.

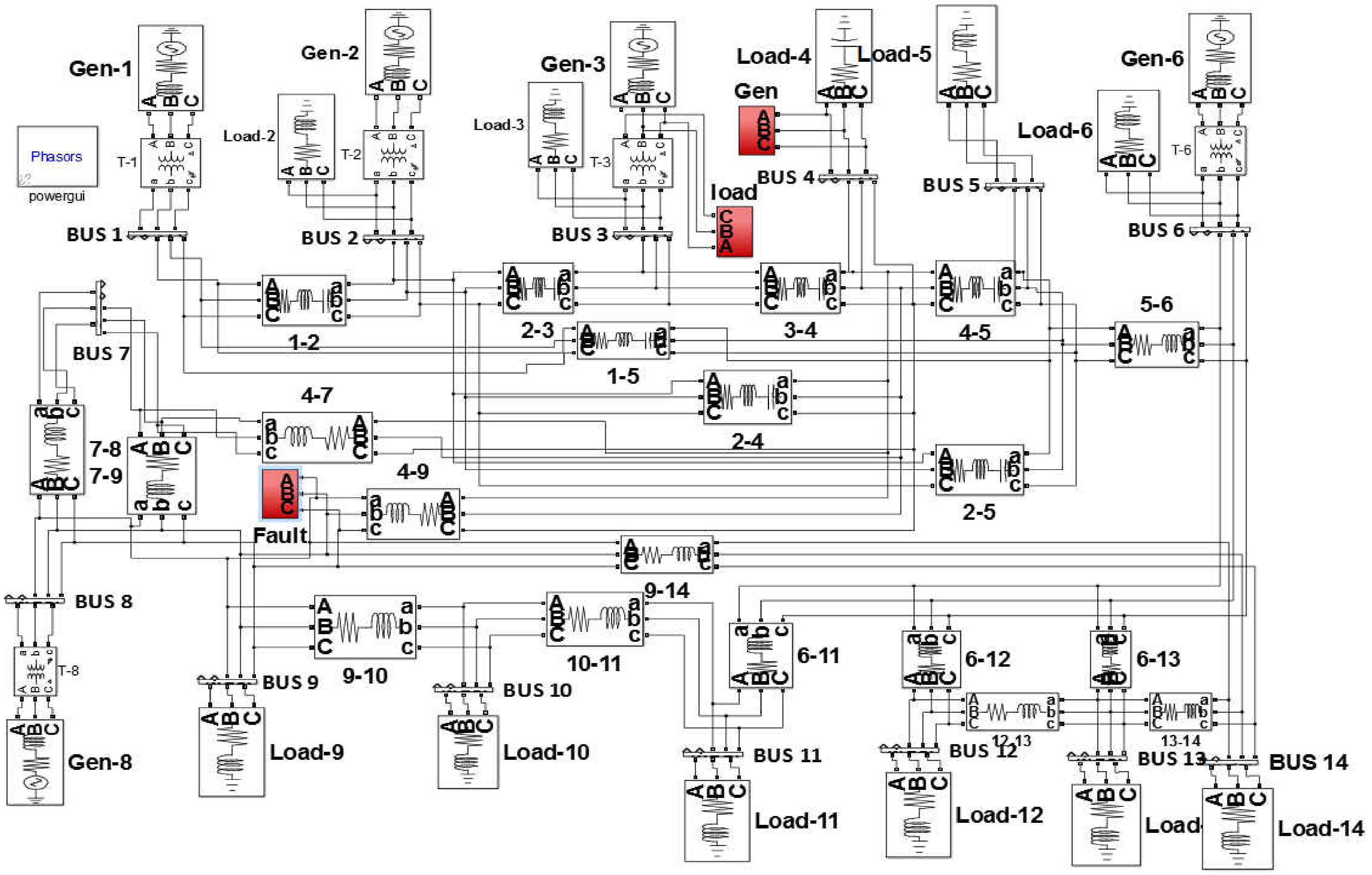


Figure 11 14 Bus System with Faults

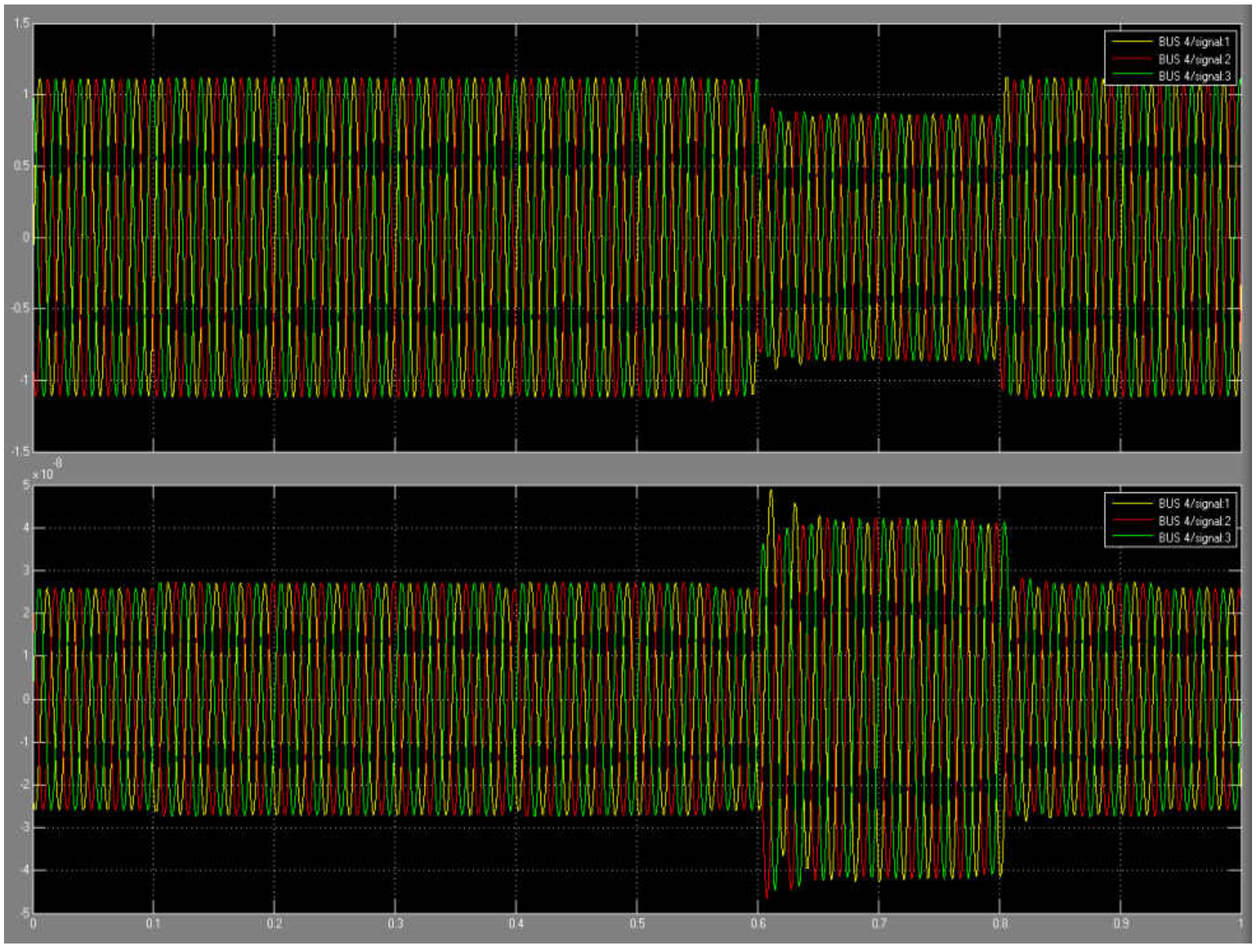


Figure 12 Voltage and Current Waveforms at Bus 4 During Fault Conditions

As explained earlier, PMU can measure electrical signals such as voltage and current at a data-rate of 30, 60, 120, 240 samples/second. My initial goal was to build a model that will simulate a realistic power system model that allow bus measurements. Our simulink model will provide similar values as an actual PMU. Each of the models will provide us the values of magnitude and phase angle of both current and voltage at all the 14 buses. A PMU placed at a bus does include the time-stamped phasor values. The measured phasor values and unit-circle representation is discussed in Chapter 4. This representation will enable a sub-station operators to visualize the parameters in the power system in near real-time.

3.3 openPDC Data Source

openPDC, an open-source software, developed by the Tennessee Valley Authority (TVA) and administered by the Grid Protection Alliance (GPA), is a complete Phasor Data Concentrator software system that is designed to process streaming time-series data in real time[36]. openPDC functions by receiving data broadcast by a PMU and concentrating it, enabling archiving, rebroadcasting, and analyzing of the phasor data [37]. Generally, openPDC archives the data as they are received. Synchrophasor data that are stored in openPDC can be mined by extraction. For this task, a C# code using Microsoft Visual Studio [38] has been developed. OpenPDC is a virtual software that has a PMU data generator developed with the Microsoft Visual C# platform. An Structured Query Language (SQL) database has been set up for the Open PDC environment; the database produces PMU data using a PDC connection tester file. The data are produced according to the Grid Protection Alliance directives. The generated data are stored in an Archives folder as ".dat" files: "ppa_archive.dat". There are separate virtual ports to access frequency, magnitude, and phase-angle data. The streaming data are stored in a ".dat" format file with a port number as an identifier. To record the data for a post-event analysis or a near real-time analysis, the port numbers corresponding to the fields can be utilized. A data-extraction code is required to extract the data from the ".dat" files and to transfer them to an ".xls" or ".csv" file format.

3.4 Data Visualization

Visualization is a useful medium for examining, understanding, and transmitting information because,

- It leverages the incredible capabilities and bandwidth of the visual system to move a huge amount of information into the brain quickly.
- It takes advantage of our brains that has the capability to identify patterns and communicate relationships and meaning.
- Data Visualization will identifying trends and outliers, discovering or searching for interesting or specific data points in a larger field, etc.

Data visualization is the study of the visual representation of data that is,

1. Data points are drawn algorithmically
2. Easy to represent complex-data form
3. Often aesthetically barren
4. Relatively data-rich

Data representation in any form need to communicate the information of data with visual cues accurately. There are two types of data visualization: *exploration and explanation*. Exploration is generally best done at a high level of granularity. There may be a whole lot of noise in your data, but if we oversimplify or strip out too much information, we could end up missing something important. This type of visualization is typically part of the data analysis phase, and is used to find the story the data has to tell you. Explanatory data visualization is appropriate when we already know what the data has to say, and you are trying to tell that story to somebody else.

There are a number of forms such as maps, flow-animation, graphical alarms, area-tie diagrams, contour plots, 3-D displays, and GreenGrid etc [39], [40] are used to represent power system data. Data generated from the models represents PMU data needs to be visually best-represented for system operators to make decisions. This will helps ISO can monitor the changes in the grid. This research chose to use a circle representation of the streaming PMU data.

3.4.1 Unit Circle Representation of Phasor Data

The voltage in an AC circuit is oscillating and represented as follows,

$$V = V_o * (\cos(\theta) + j * \sin(\theta))$$

$$where, \qquad \theta = \omega * t + \Phi \tag{43}$$

$\cos(\theta)$ represents the real part and $\sin(\theta)$ represents the imaginary part of the signal.

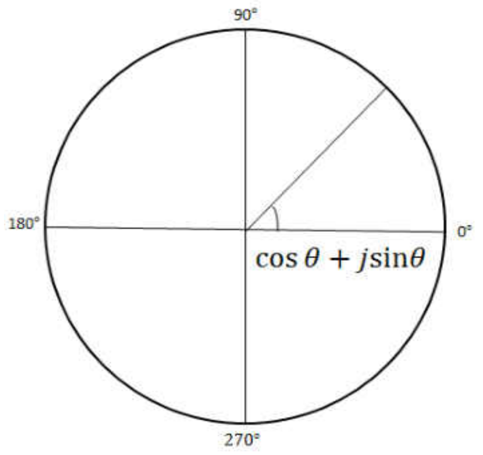


Figure 13 Unit Circle Representation

As phase angle varies between -π to+π (in radians) or between 0° to 360° (in degrees) and the magnitudes are always above 0 for the electric signals, the “unit-circle diagram” as our visualization tool. The trigonometric representations are functions of angle θ. Therefore, any value of radius can be chosen, and the simplest is a circle of radius 1, the unit circle. Figure 13 represents the basic unit circle diagram. The magnitude and phase angle information is represented better visually using unit-circle where the radius of the line and the angle θ it makes with the horizontal axis X. Therefore, this is a perfect match for our type of data and applications.

3.5 Data Mining

Databases are always rich in hidden information that is used for intelligent decision making. Data Mining (DM) is the discovery of "models" for data. Some commonly used models are statistical modelling, machine learning, artificial intelligence, neural-networks, pattern recognition and computational approaches. The other approaches include markov chains, graphical models, cluster analysis, factor analysis, linear and non-linear regression analyses, time-series analysis, and classification trees. Classification and prediction are two forms of data analysis used to extract model properties describing important data classes or to predict future trends. Such analysis can provide a better understanding of the data at large. The classification approaches predicts categorical (discrete, unordered) labels, prediction models continuous valued functions. Few methods of data classification include: neural networks, SVM classifiers, decision trees, and probabilistic methods [41], [42]. Imagine that you are given a set of data objects for analysis where the class label of each object is not known. This is quite common in large databases, because assigning class labels to a large number of objects is an expensive process. Classification techniques need class labels for classifying the data objects but unsupervised learning techniques do not require such inputs. Clustering is the process of grouping data into classes or clusters, so that objects within a cluster have high degree of similarity in comparison to another but they are dissimilar to objects in other clusters [43]. Dissimilarities are assessed based on the attribute values describing the objects. Often, distance measures are used [44], [45]. There are a number of clustering algorithms that have been developed in the past. All such algorithms are classified into different types [46]. They are:

1. Partitioning methods: k-Means [47], Minimization of Sum of Squared Errors, K-Mediods, K-Medians, K-Modes [48], Fuzzy k-Means [49], Kernel k-Means [50], Weighted k-means, Genetic k-means clustering.
2. Hierarchical methods: Agglomerative clustering and Divisive clustering [44].
3. Density based methods: DBSCAN[51], DENCLUE [52], OPTICS [53], CLARANS [54], BIRCH [55], and CURE [56].
4. Grid methods: GRIDCLUS [57], BANG, WaveCluster, STING, CLIQUE, ENCLUS, and OptiGrid clustering.
5. Spectral methods: Spectral clustering, normalized spectral, and graph cut view.

Synchrophasor data consists of voltage magnitude and its corresponding phase-angle, current magnitude and it's corresponding phase-angle. To analyze this phasor data, various DM techniques are deployed. However, each algorithm has its own advantages and dis-advantages. It is not certain that every algorithm suits all data types. Some work on large sets of data and some work with small data-sets. Some DM techniques can work well with large monotonic data and very few on multi-data variations. To mine the data and obtain meaningful information, a correct algorithm selection is very important. Synchrophasor data require large database sizes to store, and mine large number of samples. Clustering algorithms are attractive for the task of class identification in spatial databases. However, the application to large spatial databases rises the following requirements for clustering algorithms:

- Minimal requirements of domain knowledge to determine the input parameters, because appropriate values are often not known in advance when dealing with large databases.

- Discovery of clusters with arbitrary shape, because the shape of clusters in spatial databases may be spherical, drawn-out, linear, elongated etc.
- Good efficiency on large databases, i.e. on databases of significantly more than just a few thousand objects.

To achieve the following requirements, a Density Based Spatial Clustering of Applications with Noise, DBSCAN can be used [51]. DBSCAN is a density based clustering approach which is designed to discover the clusters and the noise in a spatial db. The advantages of the DBSCAN are:

- Its requires only two parameters as inputs to cluster the entire-data.
- It can discover clusters of arbitrary shapes.

3.5.1 Density based Clustering Technique (DBSCAN)

DBSCAN basically deal with the density or concentration of points. The fundamental principle of DBSCAN is to expect each cluster is defined through a probability distribution that is darker in center and lighter near the edges of cluster. In other words, the probability of selecting a sample near the central core of the distribution should be much higher. If we randomly select a large number of samples from the distribution, we'll get points from both the central core and the outer edges, but we should be able to tell the difference between them because the outer edge points will be farther apart from each other. In other words, we can tell the difference by how dense the points are. So the idea behind DBSCAN is to try and ignore the data points that are more spread-out and focus on the dense parts, which should be the central cores of the clusters. The output cluster of the DBSCAN can contain three type of points that form three clusters. They are: Core points, Border points, and Noise points.

There are few definitions which define DBSCAN. They are as follows:

- DBSCAN requires two parameters as its input values to cluster the data. They are: ε (Eps) and $MinPts$. ε: Eps is the neighborhood of a point p, denoted by $N_{Eps}(p)$, defined by

$$N_{Eps}(p) = \{q \in D | dist(p, q) \leq Eps\}. \tag{43}$$

- A point p is $directly\ density - reachable$ from a point $q$wrt. $Eps, MinPts$ if
    a. $p \in N_{Eps}(q)$
    b. $|N_{Eps}(p)| \geq MinPts$ (core point condition)
- A point p is $density - reachable$ from a point q wrt. $Eps, MinPts$ if there is a chain of points $p_1, \ldots p_n, p_1 = q, p_n = p$ such that p_{i+1} is directly density-reachable from p_i.
- Let $C_1, \ldots, C_k$ be the cluster centers of the db Dwrt. parameters $Eps_i, MinPts_i, i = 1, \ldots, k$. Then, noise can be defined as the set of points in the db D that do not belong to any of the cluster C_i i.e. $noise = p \in D | \forall i: p \notin C_i$.

A point p become a $core$ point if it has more than a specified number of points ($MinPts$) within a radius Eps (these points are the interior of a cluster). A $border$ point p has fewer than $MinPts$ within Eps, but is in the neighborhood of a core point. A $noise$ point p is any point that is not a $core$ point or a $border$ point. In general, an Eps - neighborhood of a $border$ point contains significantly less points than an Eps-neighborhood of a $core$ point. Therefore, we would have to set the minimum number of points to a relatively low value in order to include all points belonging to the same cluster. Figure 14 visually defines the different points in the DBSCAN and their definitions. The following are the steps on how DBSCAN forms clusters:

Step 1: Initially, select a point p arbitrarily.

Step 2: Observe all the points that are density reachable from w.r.t ε (Eps) and $Minpts$

Step 3: If point p is a $core$ point, a cluster is formed.

Step 4: If point p is a $border$ point and no points are density reachable from p , then DBSCAN visits the next point of the db.

Step 4: This process from step 1 to step 4 is continued till all the points have been processed or when no new point can be added to any cluster.

All points are clustered into three types that are called as $Core, Border, and\ noise$ points.

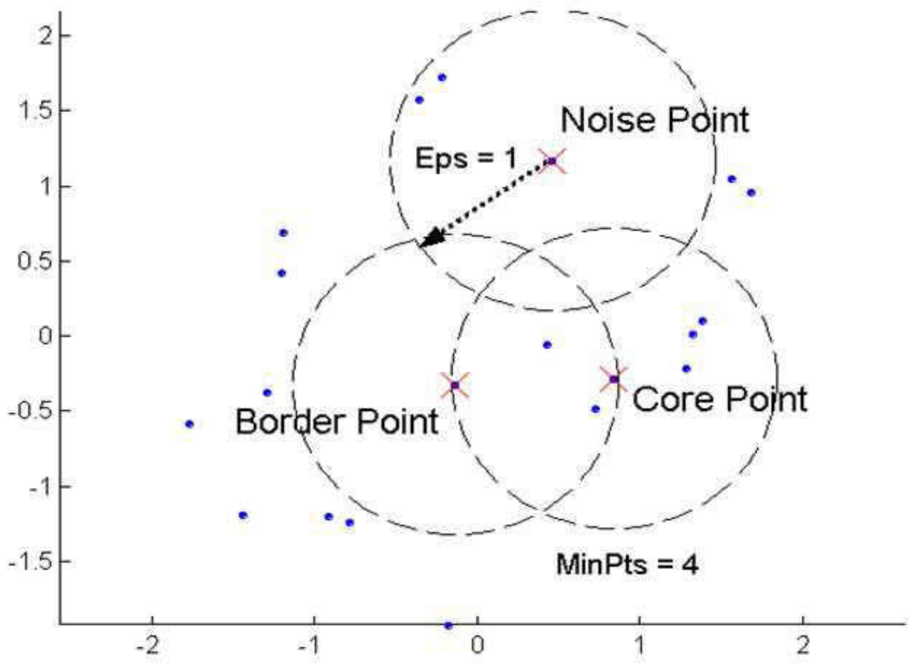


Figure 14 DBSCAN and its Points

3.5.2 Pseudo code for DBSCAN clustering

The pseudo code for DBSCAN can be written as follows:

Input: $X = \{x_1, x_2, \ldots, x_n\}$ (Set of entities to be clustered)

eps = Minimum distance between two points to be clustered (D). $MinPts$ = Minimum number of points that should be in a cluster to be considered as a border group (Pt).

```
Output: L= { $1(x)|x = 1,2, \dots n$} (Set of cluster labels of $x$)

DBSCAN (Input Set (X), Pt, D)
    foreach $x_i$ in the Input set do
      If ($x_i$ is not in any cluster) then
        If ($x_i$ is a core point) then
           Generate a new clusterID.
            Label $x_i$ with clusterID.
              expandCluster ($x_i$, X, Pt, D, clusterID)
            else
             label ($x_i$, NOISE)
            end
        end
       end
end
expandCluster ($x_i$, X, Pt, D, clusterID)
    put $x_i$ in seed queue
    while the queue is not empty
      extract c from the queue (where $c \in X$ and $c \neq x_i$)
        retrieve the neighborhood (eps) of c.
        If there are atleast minPts neighbors
          foreach neighbor n
            If n is labeled NOISE
              Label n with clusterID
                If n is not labeled
                 Label n with clusterID
                   Put n in the queue
                 end
              end
            end
end
```

Figure 15 shows an example of clustering of different type of points through DBSCAN.

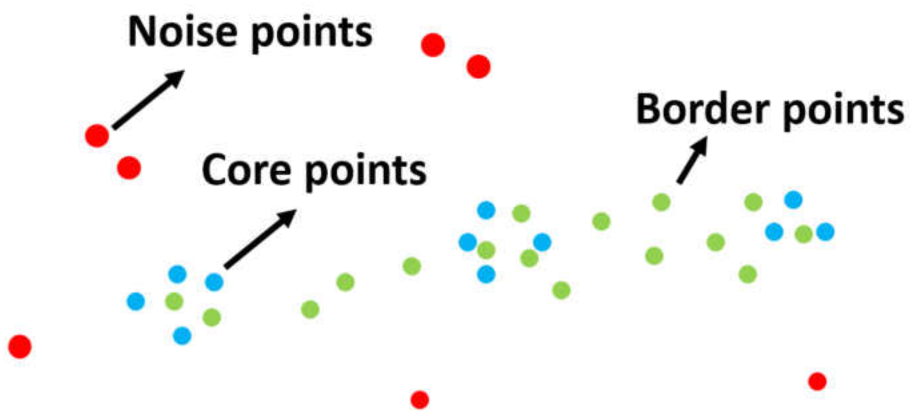


Figure 15 Cluster Formations of DBSCAN

The advantages of DBSCAN are that this method is resistant to $noise$ and can handle clusters of different sizes and shapes. The limitation lie with right selection of Eps and $MinPts$ parameters.

CHAPTER 4

RESULTS AND DISCUSSIONS

This chapter discusses the results obtained for the OPP problem using LP approach. A list of IEEE test cases are used to determine the reliability of system by applying the ORC condition. The latter part of this chapter show an the unit-circle representation obtained for synchrophasor data visualization. It discusses the advantages of such representations for monitoring of power systems. The last section of this chapter discusses the results from DBSCAN method. Various test cases are considered to analyze the output from data clustering by modeling the IEEE 14 bus system for different power system conditions.

4.1 Optimal Placement of Phasor Measurement Units (PMU) Problem

4.1.1 Test Systems (IEEE and other grid systems)

To validate the approach for solving the optimal placement problem for PMUs, the following test cases have been considered:

1. IEEE 14 bus system
2. IEEE 30 bus system
3. IEEE 57 bus system
4. IEEE 118 bus system
5. SRIPG 208 bus system
6. IEEE 300 bus system

All the required data for IEEE test cases have been taken from the University of Washington webpage [35]. The data for the SRIPG test system has been considered from [58]. The Advanced Modeling and Programming Language (AMPL) IDE platform is used to develop the LP models. IBM ILOG CPLEX Optimization Studio solver is used to solve the formulated linear constraints obtained from the test cases.

4.1.1.1 IEEE 14 bus system

The IEEE 14-bus case represents a simple approximation of the American Electric Power system as of February 1962. It has 14 buses, 5 generators, and 11 loads as shown in Figure 16. It has also one zero-injection bus at bus 7. The AMPL code is provided in Appendix A.

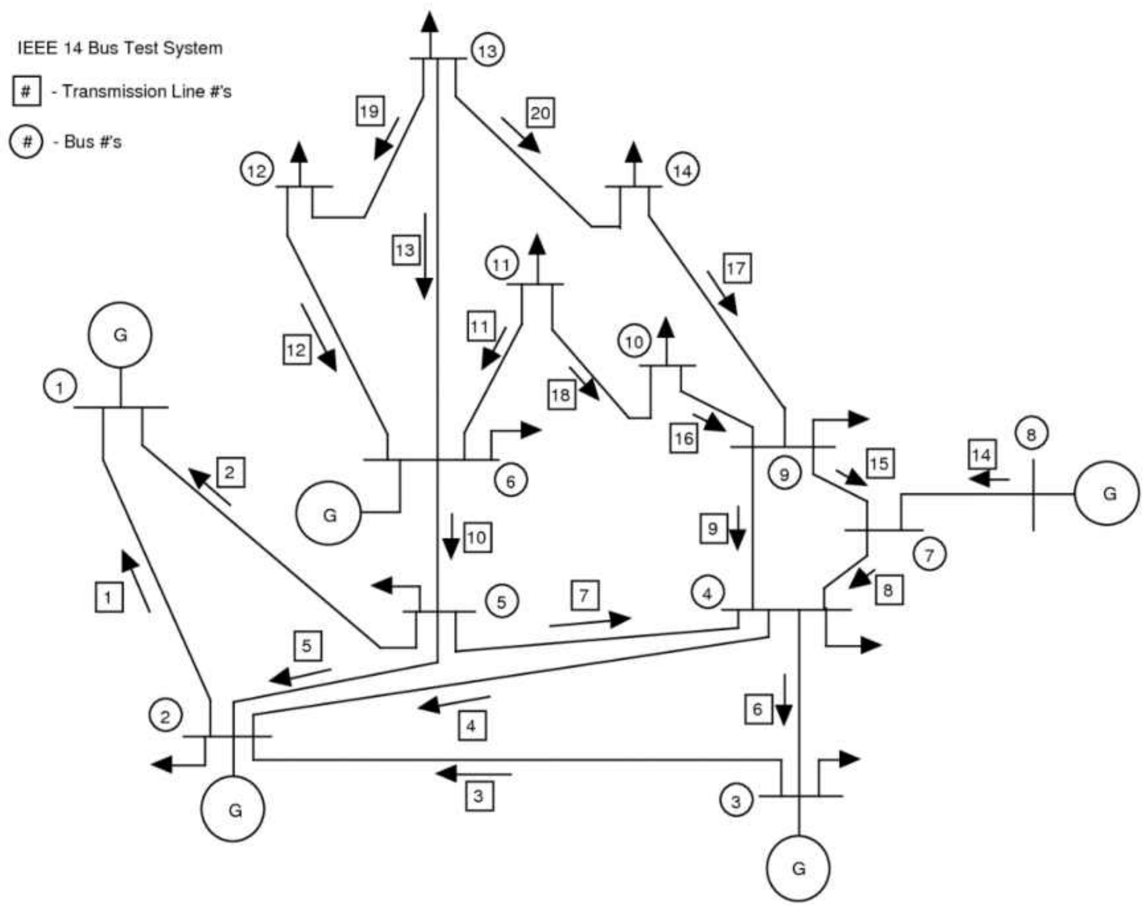


Figure 16 IEEE 14 Bus System

4.1.1.2 IEEE 30 Bus System

The IEEE 30-bus test case represents an approximation of AEP system. The system has 15 buses, 7 generators at buses 1, 2, 4, 5, 8, 11, and 13 as shown in Figure 17. It has 5 zero-injection buses at 6, 9, 11, 25, and 28. The AMPL code is provided in Appendix B.

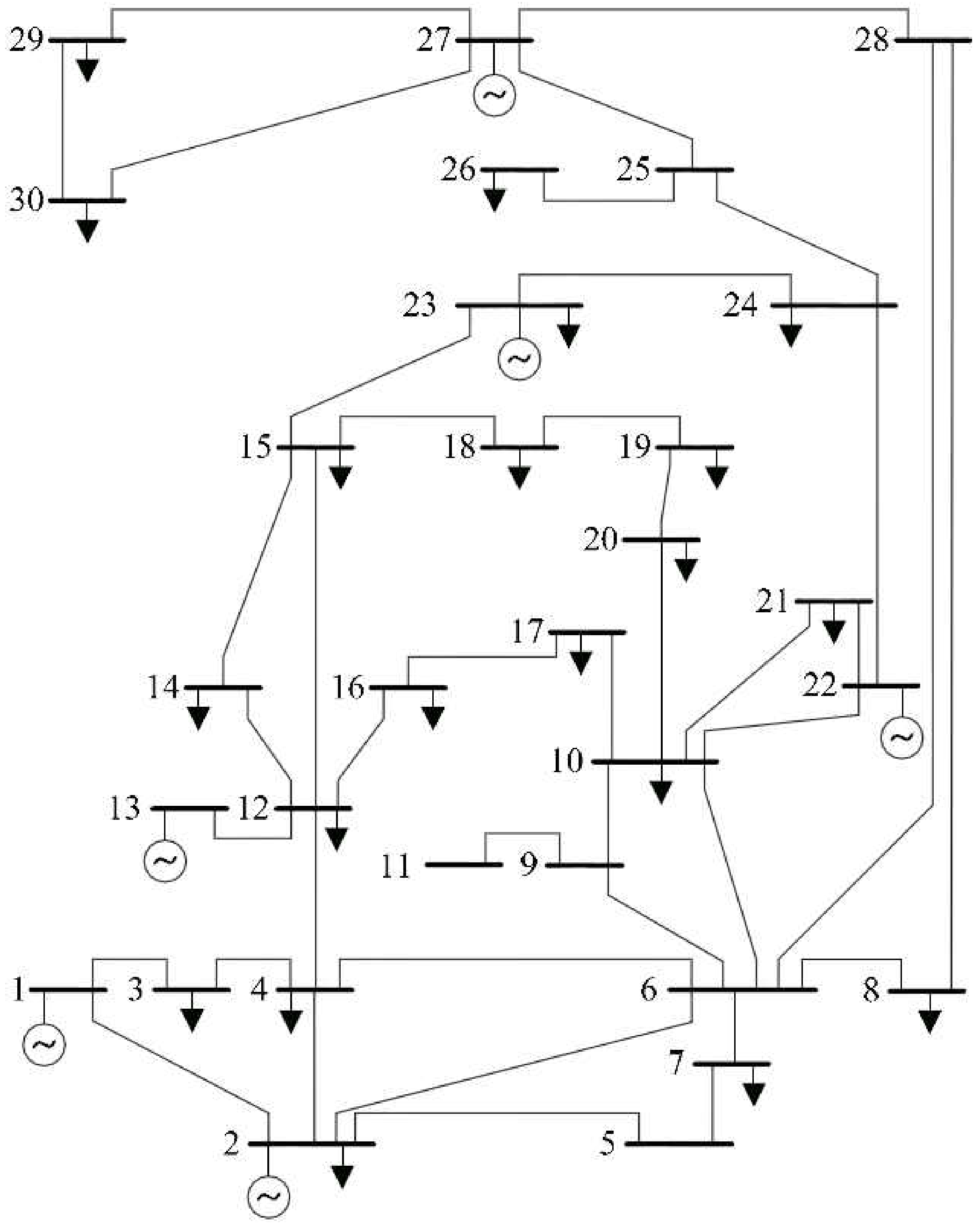


Figure 17 IEEE 30 Bus System

4.1.1.3 IEEE 57 Bus System

The IEEE 57-bus test case represent a simple approximation of the American Electric Power system (in the U.S. Midwest) developed in 1960. From Figure 18, the IEEE 57-bus test case system has 57 buses, 7 generators at buses 1, 2, 3, 6, 8, 9, and 12. It also got 42 loads. It has got 15 zero-injections buses at 4, 7, 11, 21, 22, 24, 26, 34, 36, 37, 39, 40, 45, 46, and 48. The AMPL code is provided in Appendix C.

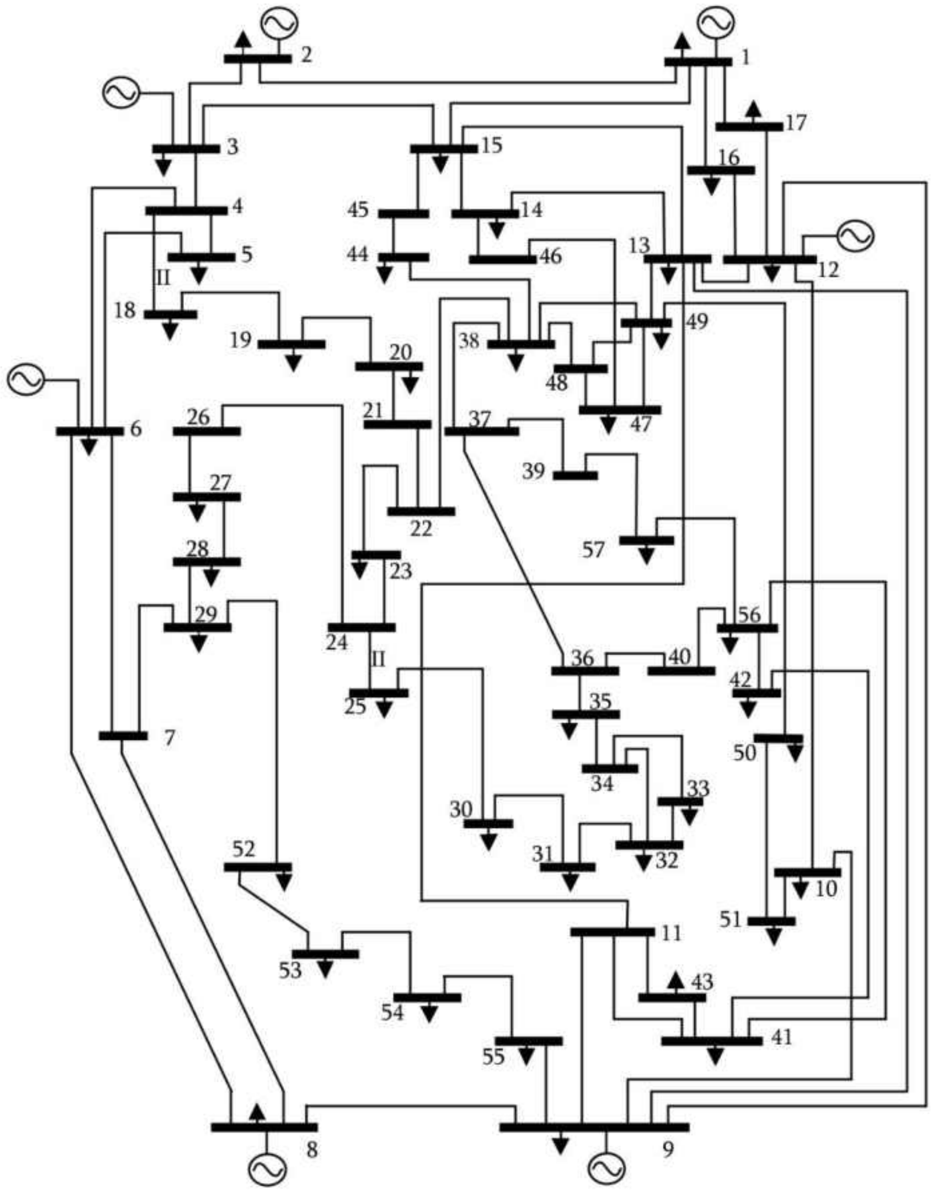


Figure 18 IEEE 57 Bus System

4.1.1.4 IEEE 118 Bus System

This IEEE 118-bus test case represents a simple approximation of the American Electric Power system (in the U.S. Midwest).. This IEEE 118 bus system is shown in Figure 19. The system contains 19 generators, 35 synchronous condensers, 177 lines, 9 transformers, and 91 loads. There are 10 zero-injection buses at 5, 9, 30, 37, 38, 63, 64, 68, 71, and 81. The AMPL code is provided in Appendix D.

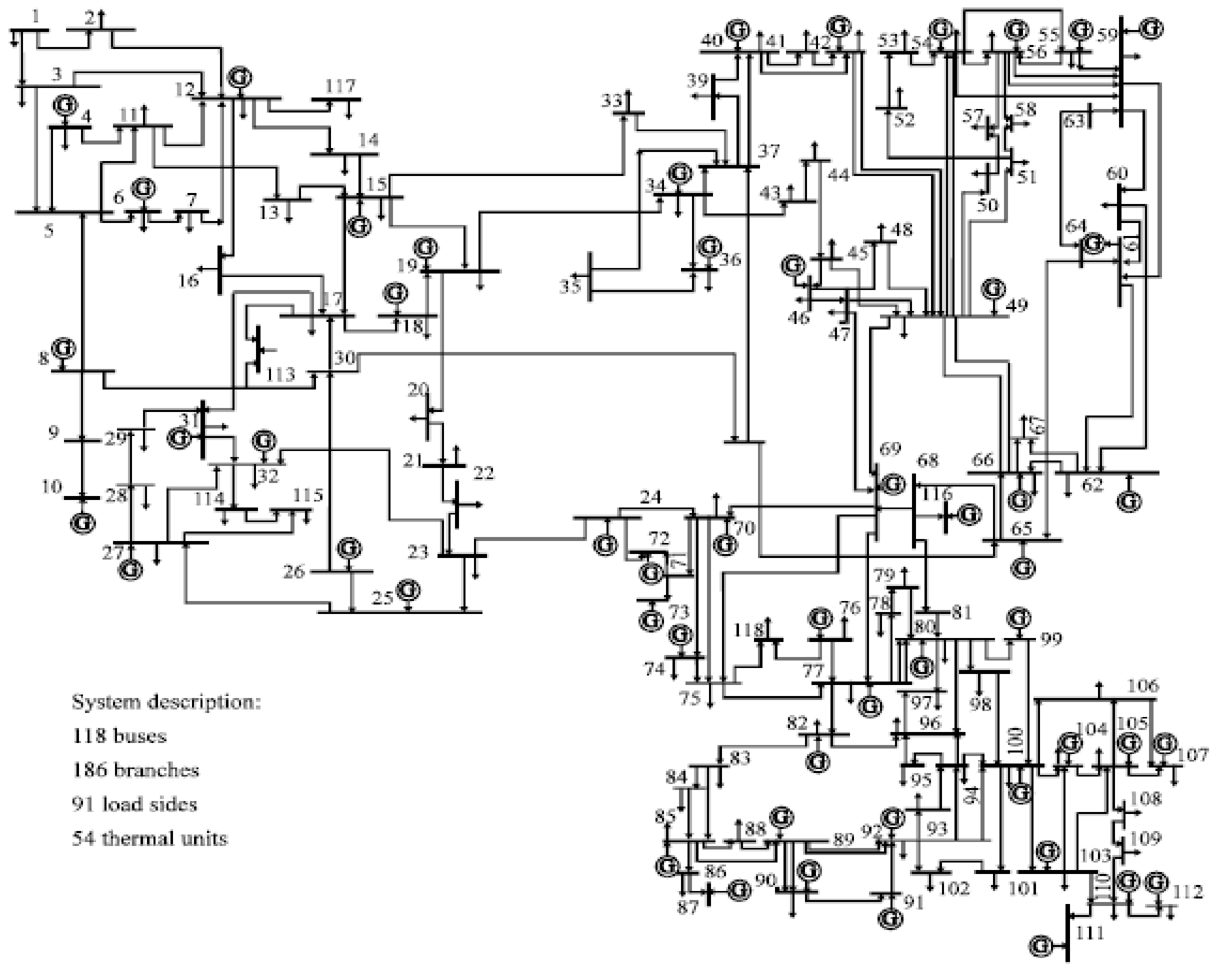


Figure 19 IEEE 118 Bus System

4.1.1.5 SRIPG 208 Bus System

The SRIPG 208 bus system is a part of complex Indian power grid. Figure 20 show the single line diagram of the system. The AMPL code is provided in Appendix E.

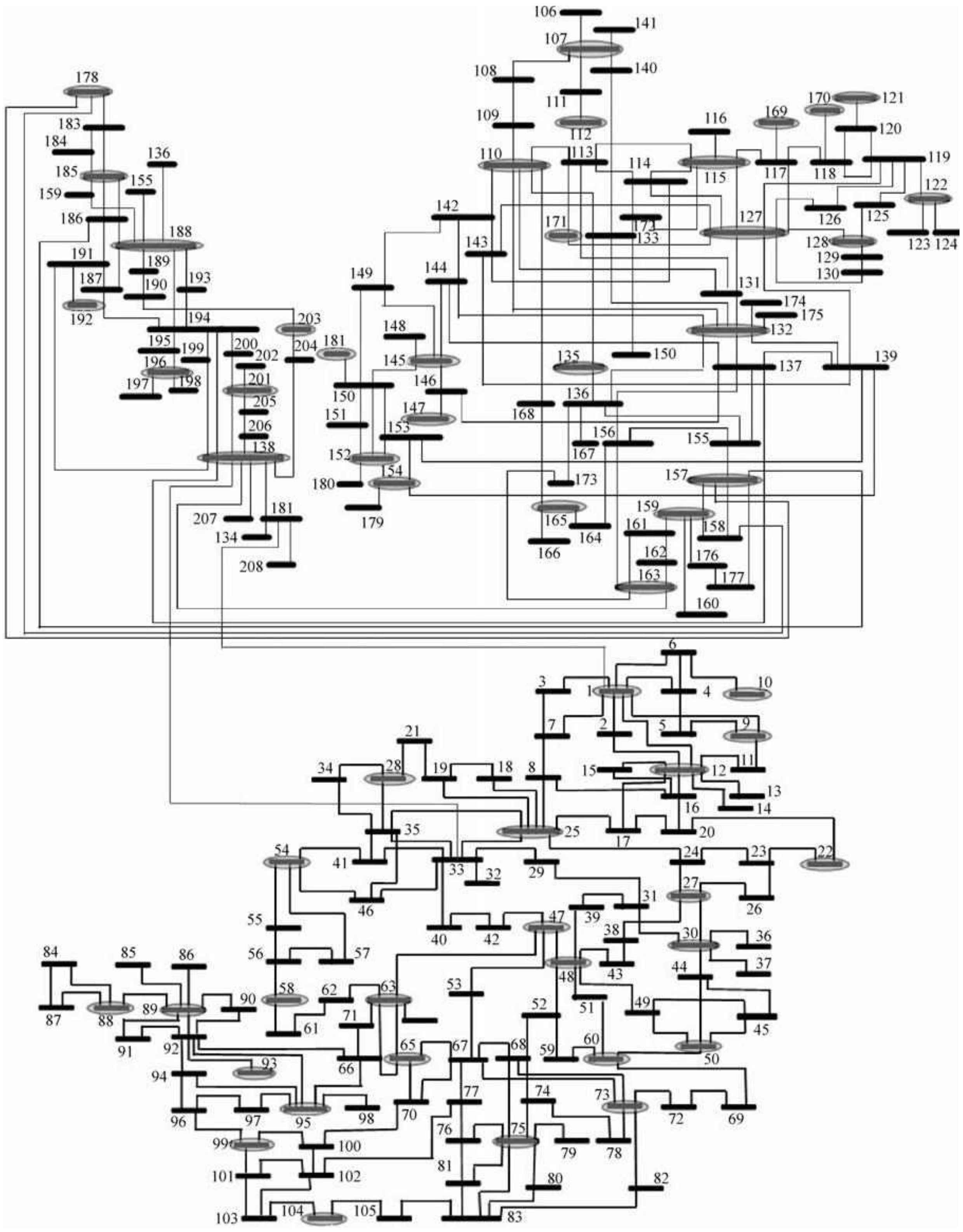


Figure 20 SRIPG 208 Bus System

4.1.1.6 IEEE 300 Bus System

This IEEE 300-bus test case was “developed by the IEEE Test Systems Task Force under the direction of Mike Adibi in 1993” . This IEEE 300-bus system contains 69 generators, 60 LTCs, 304 transmission lines, and 195 loads as shown in Figure 21. The single line diagram is considered from [59]. It has 65 zero-injection buses. The AMPL code is provided in Appendix F.

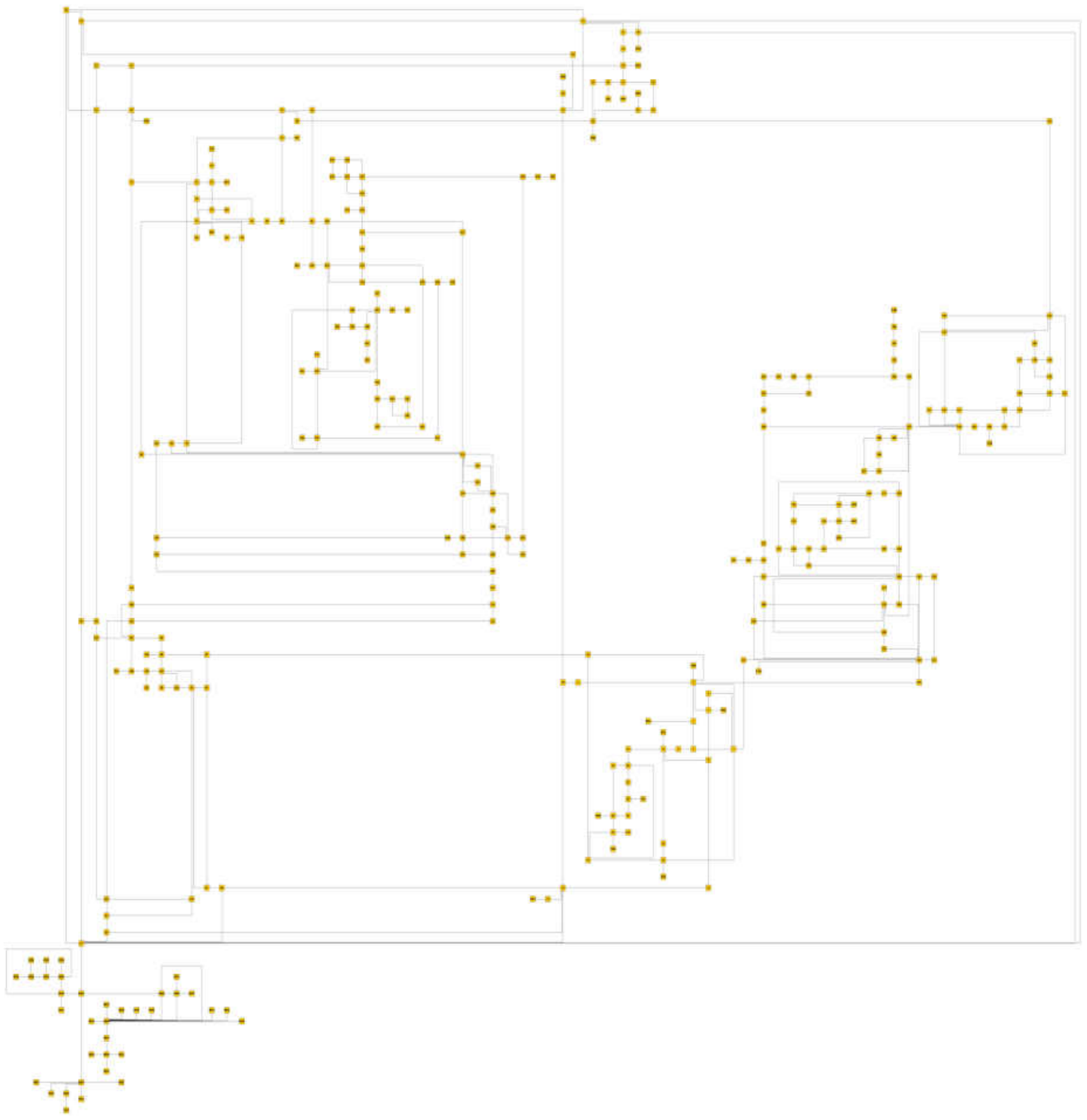


Figure 21 IEEE 300 Bus System

4.1.2 Advanced Modeling and Programming Language (AMPL)

AMPL is a algebraic modeling language for mathematical programming. It was designed and implemented in 1985 and has been evolving ever since [60]. This is a sophisticated modeling tool that supports the entire optimization modeling lifecycle: development, testing, deployment, and maintenance. AMPL integrates

1. Modeling language for describing optimization data, variables, objectives, and constraints.
2. A command language for browsing models and analyzing results; and a scripting language for gathering and manipulating data and for implementing iterative optimization schemes. All use the same concepts and syntax for streamlined application-building.

The main advantages using AMPL are as follows:

1. It has a general and easy to use syntax for arithmetic, logical, and conditional expressions. It used familiar conventions for summations and other iterated operators. It can handle linear and convex quadratic problems in continuous and integer variables, nonlinear programming.
2. A wide range of solvers such as CPLEX, Gurobi, Xpress (Linear and convex quadratic solvers), CONOPT, Ipopt, KNITRO, MINOS, SNOPT (Nonlinear solvers), Bonmin, Couenne, KNITRO (mixed-integer solvers) are available. In addition, open-source solvers can also hooked to AMPL.

AMPL is notable for the similarity of its arithmetic expressions to customary algebraic notation, and for the generality and power of its set and subscripting expressions. AMPL also extends algebraic notation to express common mathematical programming structures such as network flow constraints and piecewise linearties. AMPL has a flexible interface that enables several solvers to be available at once so a user can switch among solvers and select options that may improve and optimize the solution. All of the general set and arithmetic expressions of the AMPL modeling language can also be used for displaying data and results; a variety of options are available to format data for browsing, printing reports, or preparing input to other programs. Using AMPL, mathematical algebraic problems can be directly converted into a small system of inequalities and an objective function. The availability of an algebraic modeling language makes it possible to emphasize the kinds of general models that can be used to describe large-scale optimization problems. A practical diagram of how AMPL works is shown in Figure 22 below.

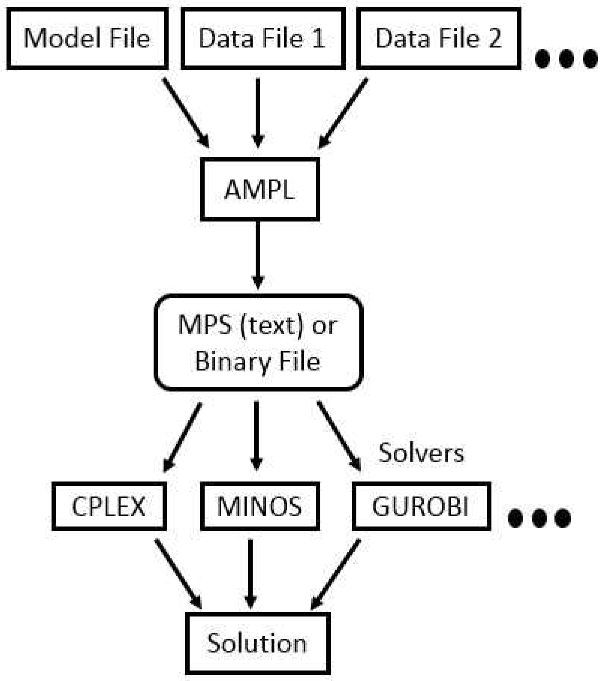


Figure 22 Functional Diagram of AMPL

The model (.mod) files are fed into the AMPL program which works like a compiler. The model and the input data are put into a format that is understood buy the solver. The solver actually finds an optimal solution to the problem by reading the intermediate file produced by AMPL and applying the appropriate algorithm, The OPP for PMUs is solved using the technique of LP. Based on the topology information, the constraint formulations can be directly embedded into the AMPL. I followed the methodology of formulating the algebraic topological constraints in AMPL and then using the CPLEX linear optimization solver to solve them.

4.1.3 Zero-Injection Buses

Table 5 shows the various zero-injection buses in the test cases considered. It provides various bus numbers that act as zero-injection buses in the test systems to model the zero-injection constraints for the OPP.

Table 5 Zero-Injection Buses

Bus Systems	**Zero Injection Buses**	**Number of Zero Injections**
14	{7}	1
30	{6,9,11,25,28}	5
57	{4,7,11,21,22,24,26,34,36,37,39,40,45,46,48}	15
118	{5,9,30,37,38,63,64,68,71,81}	10
300	{4,7,12,16,19,24,34,35,36,39,42, 45,46,60,62,64, 69, 74,78,81,85,86,87,88,100,115,116,117,128,129, 130, 131,132,133,134,144,150,151, 158,160,164, 165, 166, 168,169,174,193,194,195,210,212,219,226,237,240,244,1201,2040,9001,9005,9006,9007,9012,9023,9044}	65

The LP formulations are carried for all the test systems in AMPL. Firstly, solutions are obtained with normal test cases without considering any zero-injection buses. Secondly, zero-injection constraints are added as additional constraints to study the number of PMUs required.

Then, an ORC condition is enforced on both the scenarios (i.e. with and without zero-injection) bus constraints to determine the reliability of the model. Table 6 show the number of PMUs required for the test case systems without zero-injection constraints. The column two in Table 6 identify the number of PMUs required for each of the test case system. It also shows the bus locations, where the PMUs needed to be placed. Column three show an optimal number of PMUs obtained in various literatures.

Table 6 Optimal Number of PMUs without Zero-Injections

Test Systems	**ILP Method**	**Other Methods**
IEEE 14 Bus	4 {2,6,7,9}	4
IEEE 30 Bus	10 {1,5,6,9,10,12, 15,18,25,27}	10
IEEE 57 Bus	17 {2,6,12,19,22,25,27,32,36,39,41,45,46,49,51,52,55}	17
IEEE 118 Bus	32{1,5,9,12,15,17,21,23,29,30,36,40,44,46,50,51,54,62,64,71,75,77,80, 85,86,90,94,101,105,110,115,116}	32
SRIPG 208 Bus	55{5,6,7,12,22,25,27,28,30,33,47,48,50,54,58,60,63,67,73,75,88,89,92, 95,99,104,107,110,111,115,117,118,120,122,127,128,131,132,136,138, 145,146,152,154,157,159,163,165,181,183,188,190,191,196,201}	58
IEEE 300 Bus	87{1,2,3,11,12,15,17,20,23,24,26,33,36,39,43,44,49,55,57,61,62,63,70, 71,72,74,77,81,84,86,97,102,104,105,108,109,114,119,120,122,130,13 2,133,134,137,139,140,143,153,154,159,160,164,166,173,178,181,184, 189,194,204,208,210,211,214,217,221,225,229,231,233,234,237,238,2 40,244,245,249,9003,9004,9005,9006,9006,9006,9006,9006,9533}	87

By comparison of column two and three, the minimal set of PMUs required for test cases yielded the similar results that have been previously proposed in the literatures [61]. However, the SRIPG system seem to provide a better result of 55 number of PMUs when compared to the results provided in [62]. Table 7 provide the results obtained for the OPP when the zero-injection buses are considered. Generally, with zero-injection considerations, the number of PMUs required will decrease due to the increase in the constraints in the LP approach. This can is seen in Table 7.

Table 7 Optimum Number of PMUs while Considering Zero-Injection Constraints

Test systems	**ILP method**	**Other Methods**
IEEE 14 Bus	3 {2,6,9}	3
IEEE 30 Bus	7 {2,3,10,12,19,24,27}	7
IEEE 57 Bus	10 {1,13,18,25,29,32,38,41,51,54}	11
IEEE 118 Bus	27{2,8,11,12,19,22,27,28,32,34,40,45,49,53,56, 62,70,76,77,80,85,87,91,94,101,105,110}	28
SRIPG 208 Bus	N/A	N/A
IEEE 300 Bus	68{1,2,3,11,15,21,23,26,33,43,44,49,55,57,61,63,664,70,72,73,77,7071,97,102,104,105,108,109,7166,114,119,120,122,126,137,139,140,143,152,155,167,175,178,183,184,188,198,204,205,214,217,223,225,229,231,233,234,238,245,249,7017,7039,9002,9003,9004,9006,9006,9006}	N/A

While considering zero-injection buses for solving the OPP for PMUs, there is a significant decrease in the number of PMUs required. This can be seen in by comparing the Table 6 and Table 7. Using the LP formulations, the number of PMUs required for complete observability are smaller than the solutions provided in the literatures. (IEEE 57 and 118 bus systems). To the best of authors knowledge, there is no study that has solved the optimal placement problem for the IEEE 300 bus system by considering zero-injection buses. This is carried as an additional task in this research. This research has successfully implemented LP formulations in AMPL on IEEE 300 bus system to obtain the minimal set of PMUs required.

4.1.4 Enforcing Optimal Redundancy Criterion (ORC)

Another case was to find optimal number of PMUs with ORC condition is enforced. Table 8 show the result with ORC enforced. This is done simply by modifying the right hand-side of the inequality constraint. Here, the redundancy of the critical buses are set to 2. By enforcing ORC, all critical buses are observed at least twice. This means that any failure in PMU measurements will not affect the system's observability. For more clear observation on ORC's performance, consider the Table 9. Column two from the Table 9 show the minimum number of PMUs required and column three show the number of PMUs required with ORC is enforced. Column four gives us an idea of percentage increase in number of PMUs when ORC is enforced. It is observed that with around 10% increase in number of PMUs, more redundancy in measurements are seen.

Table 8 Optimal Number of PMUs Enforcing ORC

Test Systems	**Without Zero-Injections With ORC**	**With Zero-Injections With ORC**
IEEE 14 Bus	5 {1,4,8,10,13}	4 {4,5,10,13}
IEEE 30 Bus	10 {1,2,6,9,10,12,15,18,25,27}	8{2,3,10,12,13,19,23}
IEEE 57 Bus	19{1,4,6,9,10,11,15,20,24,25,28,32,36,38,41,46,49,53,57}	13{3,6,13,16,17,20,25,29,32,38,41,51,54}
IEEE 118 Bus	37{3,5,9,12,15,17,19,22,23,26,28,34,37,40,45,49,52,56,59,61,66,68,71,75,77,80,85,86,90,93,94,100,104,107,109,110,114}	29{3.8.12.15.19.21.27.29.32.34.42.45.49.53.56.57.62.70.75.77.80.85.86.90.94.100.101.105.110}
SRIPG 208 Bus	60{1,6,9,12,20,24,25,27,28,30,33,47,48,50,55,57,58,60,63,67,73,75,83,88,89,92,95,99,102,103,107,110,111,115,117,118,120,122,127,128,132,136,138,142,145,146,152,153,154,157,159,163,165,181,184,188,190,191,196,201}	N/A
IEEE 300 Bus	96{1,2,3,11,12,13,15,17,21,23,24,26,323,39,40,41,42,43,44,55,57,61,62,63,69,70,71,72,74,77,84,88,94,102,104,105,108,109,110,117,119,120,122,127,130,132,133,134,137,139,145,146,148,153,154,160,164,166,173,175,177,184,188,189,194,196,205,206,210,211,214,217,219,221,225,229,231,232,236,237,238,240,245,246,249,7049,9003,9004,9005,9006,9006,9006,9006,9006,9006,9006}	76{1,2,3,11,13,15,17,21,23,26,37,41,43,44,55,57,61,63,664,70,71,72,73,77,94,102,104,105,108,109,110,7166,119,120,121,125,126,137,139,140,145,148,153,156,163,167,173,176,180,184,188,196,198,201,206,214,217,221,225,229,231,233,234,238,245,247,249,7039,7049,9002,9003,9004,9006,9006,9006,9006}

Table 9 Number of PMUs with ORC Enforced without Zero-Injection Buses

Bus System	Number of PMU's for Full Observability	Number of PMU's with ORC	% of Increase in PMU's with ORC
IEEE 14 Bus System	4	5	25%
IEEE 30 Bus System	10	10	0
IEEE 57 Bus System	17	19	11.764%
IEEE 118 Bus System	32	37	15.625%
SRIPG 208 Bus System	55	60	7.272%
IEEE 300 Bus System	87	96	10.344%

A similar table is prepared for test cases while considering zero-injection buses. Table 10 show the percentage of increase in number of PMUs considering zero-injection constraints.

Table 10 Number of PMUs with ORC Enforced with Zero-Injection Buses

Bus System	Number of PMU's for Full Observability	Number of PMU's with ORC	% of Increase in PMU's with ORC
IEEE 14 Bus	3	4	25%
IEEE 30 Bus	7	8	14.29%
IEEE 57 Bus	10	13	30%
IEEE 118 Bus	27	29	7.41%
SRIPG 208 Bus	N/A	N/A	N/A
IEEE 300 Bus	68	76	11.76%

4.1.5 SORI Index

Table 11 show a SORI index obtained for various test systems considering both with and without zero-injection constraints. It also shows a comparison of the results obtained through proposed approach and approaches in literatures [26]. It is seen that the LP formulated SORI indices are higher than the ones in the existing literatures. This indicate that using the LP approach, the test case systems are much more redundant and the full-observability of the buses are higher when compared to the values indicated in other literatures.

Table 11 System Observability Redundancy Index

Bus System	**SORI**			
	Without Zero-Injections	**With Zero-Injections**	**Without Zero-Injections**	**With Zero-Injections**
	ILP Method		**Other Methods**	
IEEE 14 Bus	18	14	19	15
IEEE 30 Bus	51	46	N/A	N/A
IEEE 57 Bus	72	52	72	61
IEEE 118 Bus	167	182	164	152
SRIPG 208 Bus	320	NA	N/A	N/A
IEEE 300 Bus	427	355	N/A	N/A

4.1.6 PMU Placement Costs

In Chapter 1, we discussed about the requirement of the PMU placement problem. While reliability is one of the reasons for PMU placement problem, the other parameter is of interest is the cost. Table 1 show the approximate cost of placing a PMU in the system. Based on the approximate costs, the costs associated with the PMU placement for test systems are estimated. Table 12 provide cost information for an utility to place the PMUs in grids. Column two show the dollar amount required for placing PMUs without considering zero-injections, while the third column indicate the amount required for without considering zero-injections and ORC. Fourth column show the dollar amount required for systems considering zero-injections. By observation, using zero-injection constraints in the LP formulations, the total cost could be reduced significantly. The ORC increases the cost of PMUs by a fraction 9 (for larger test systems). Nevertheless, the cost increase is judiciously justified by trading reliability for full observability for critical infra-structure such as a reliable power-grid for tomorrow.

Table 12 Costs Associated with PMU Placement

Test System	**Without Zero-Injections (in US $)**	**Without Zero-Injections with ORC (in US $)**	**With Zero-Injections (in US $)**
IEEE 14 Bus	96,000	96,000	74,000
IEEE 30 Bus	2,28,000	2,28,000	1,62,000
IEEE 57 Bus	3,82,000	4,26,000	2,28,000
IEEE 118 Bus	7,12,000	8,22,000	6,02,000
SRIPG 208 Bus	12,26,000	13,36,000	NA
IEEE 300 Bus	19,38,000	21,36,000	15,12,000

4.1.7 Justification of ORC Index

ORC increases total costs for an electric utility. However, it also increases the redundancy of the critical buses required to keep a stable system. Table 13 show an increase in SORI index values with ORC enforced. For example, IEEE 57 bus system requires 17 and 19 PMUs for full observability with and without enforcing ORC respectively as seen in Table 9. Comparing to

Table 13, the SORI index rose from 72 to 85 in the same system with ORC is enforced. This indicate that with just two PMUs increase in the system, the full-observability is increased to around 13 buses in the 57 bus system. Thus, increases the confidence of redundancy for critical buses. The other test cases also show positive results with increased values in SORI. Table 9 show the number of PMUs required for full-observability with and without ORC index are 32 and 37 respectively. Now, let us consider Table 14 for some contingency conditions in IEEE 118 bus system.

Table 13 System Observability Redundancy Index with ORC

Test Bus Systems	**Without ORC**	**With ORC**
IEEE 14 Bus System	18	18
IEEE 30 Bus System	51	51
IEEE 57 Bus System	72	85
IEEE 118 Bus System	167	215
SRIPG 208 Bus System	320	358
IEEE 300 Bus System	427	490

Scenario 1: During normal operating conditions, a PMU at bus location 40 in IEEE 118 bus system will observe buses 37, 39, 40, 41, and 42. If the PMU at bus 40 fails for some reason (weather, planned or unplanned failure) to observe the bus 37, then this failure will affect the observability of 7 other buses that are inter-connected to the bus 37. This can be visually understood with the help of case 1 in Figure 24. This is the main reason why bus 37 is selected as a critical bus when ORC is enforced. On application of ORC, the critical bus 37 will be observed by 3 other PMUs as shown in Figure 24 and Table 14.

Table 14 Contingency Cases in IEEE 118 Bus System with and without ORC

Scenario	**Critical buses**	**PMU failures (number of un-observed buses)**	**# of Unobservable Buses**	**% of un-observability without ORC**	**PMU placed buses with ORC enforced (critical buses are observed at least twice)**	**% of un-observability with ORC**
1	37	Case 1: PMU at bus **40**: {37,39,40,41,42}	5	4.237%	Bus 37 is observed thrice :{34, 37, 40}	0 %

2	37, 59	Case 1: PMU at bus **40**: {37,39,40,41,42}
		Case 2: PMU at bus **54**: {49,53,54,55,56,59}
			11	9.32%	Bus 37 is observed thrice :{34,37,40}
					Bus 59 is observed thrice :{56, 59,61}
						0 %

3	37,59, 105	Case 1: PMU at bus **40**: {37,39,40,41,42}
		Case 2: PMU at bus **54**: {49,53,54,55,56,59}
		Case 3: PMU at bus **105**:{104,105,106,107,108}
			16	13.55%	Bus 37 is observed thrice :{34,37,40}
					Bus 59 is observed thrice :{56,59,61}
					Bus 105 is observed twice :{104,108}
						0 %

4	37,59, 92,105	Case 1: PMU at bus **40**: {37,39,40,41,42}
		Case 2: PMU at bus **54**: {49,53,54,55,56,59)
		Case 3: PMU at bus **105**:{104,105,106,107,108}
		Case 4: PMU at bus **94**:{92,93,94,95,96,100}
			22	18.6%	Bus 37 is observed thrice :{34,37,40}
					Bus 59 is observed thrice :{56, 59, 61}
					Bus 105 is observed twice :{104,108}
					Bus 92 is observed twice :{93,94}
						0 %

5	5,37,59, 92,105	Case 1: PMU at bus **5**:{3,4,5,6,8,11}
		Case 2: PMU at bus **40**: {37,39,40,41,42}
		Case 3: PMU at bus **54**: {49,53,54,55,56,59}
		Case 4: PMU at bus **105**:{104,105,106,107,108}
		Case 5: PMU at bus **94**:{92,93,94,95,96,100}
			28	23.72%	Bus 5 is observed twice :{3,5}
					Bus 37 is observed thrice :{34,37, 40}
					Bus 59 is observed thrice :{56, 59,61}
					Bus 105 is observed twice :{104,108}
					Bus 92 is observed twice :{93,94}
						0 %

Table 14 Cont

6	5,12,37,
	59,92,
	105
		Case 1: PMU at bus **5**:{3,4,5,6,8,11}
		Case 2: PMU at bus **12**:{2,3,7,11,12,14,16,117}
		Case 3: PMU at bus **40**: {37,39,40,41,42}
		Case 4: PMU at bus **54**: {49,53,54,55,56,59}
		Case 5: PMU at bus **105**:{104,105,106,107,108}
		Case 6: PMU at bus **94**:{92,93,94,95,96,100}
			36	30.5%	Bus 5 is observed twice :{3,5}
					Bus 12 is observed twice :{3,12}
					Bus 37 is observed thrice :{34,37,40}
					Bus 59 is observed thrice :{56, 59,61}
					Bus 105 is observed twice :{104,108}
					Bus 92 is observed twice :{93,94}
						0 %

7	5,12,17,
	37,59,
	92,105
		Case 1: PMU at bus **5**:{3,4,5,6,8,11}
		Case 2: PMU at bus **12**: 2,3,7,11,12,14,16,117}
		Case 3: PMU at bus **17**:{15,16,17,18,30,31,113}
		Case 4: PMU at bus **40**: {37,39,40,41,42}
		Case 5: PMU at bus **54**: {49,53,54,55,56,59}
		Case 6: PMU at bus **105**: 104,105,106,107,108}
		Case 7: PMU at bus **94**:{92,93,94,95,96,100}
			43	36.4%	Bus 5 is observed twice :{3,5}
					Bus 12 is observed twice :{3,12}
					Bus 17 is observed twice :{15,17}
					Bus 37 is observed thrice :{34,37,40}
					Bus 59 is observed thrice :{56, 59,61}
					Bus 105 is observed twice :{104,108}
					Bus 92 is observed twice :{93,94}
						0 %

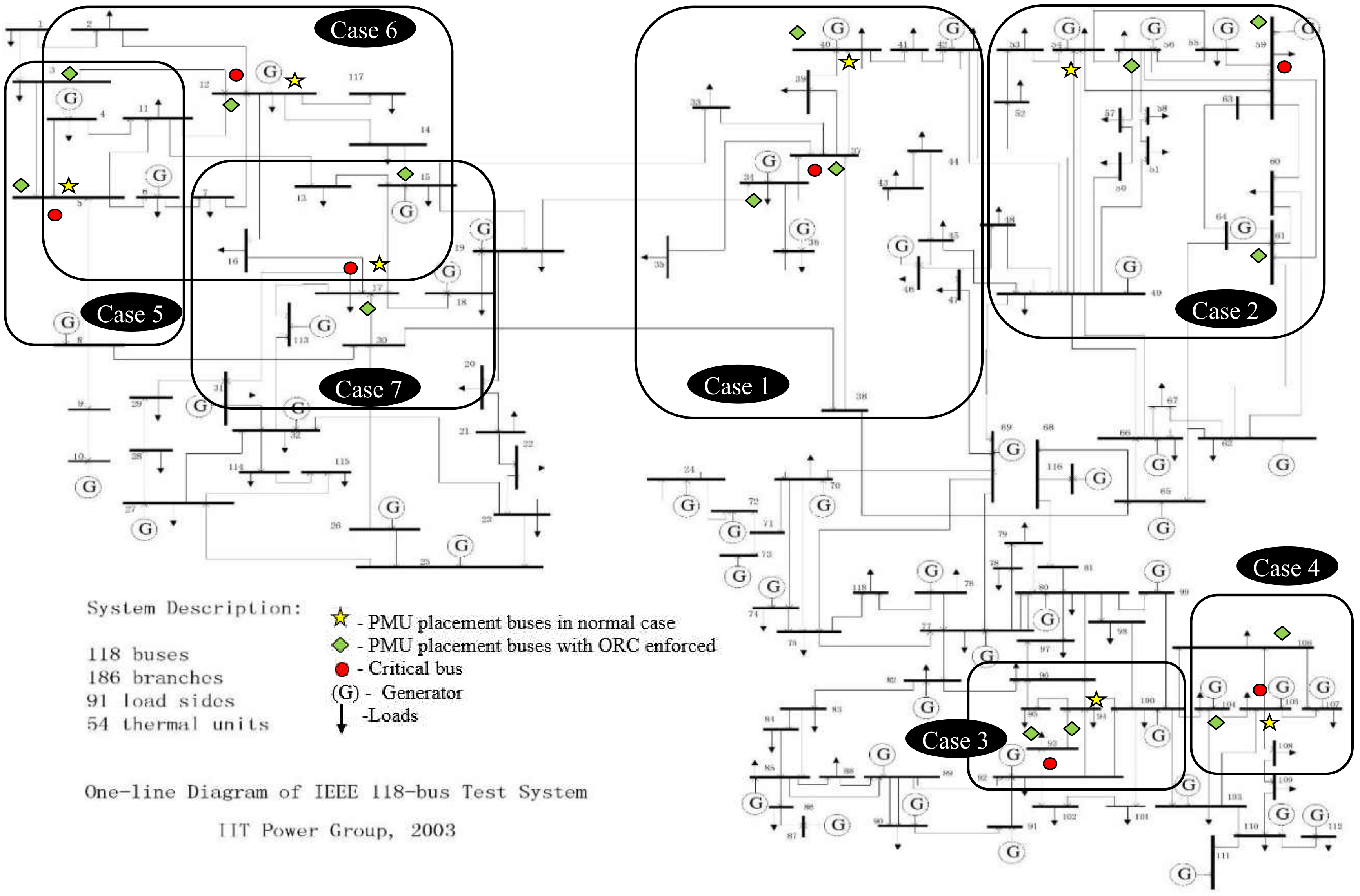


Figure 23 IEEE 118 Bus System with Contingency Cases

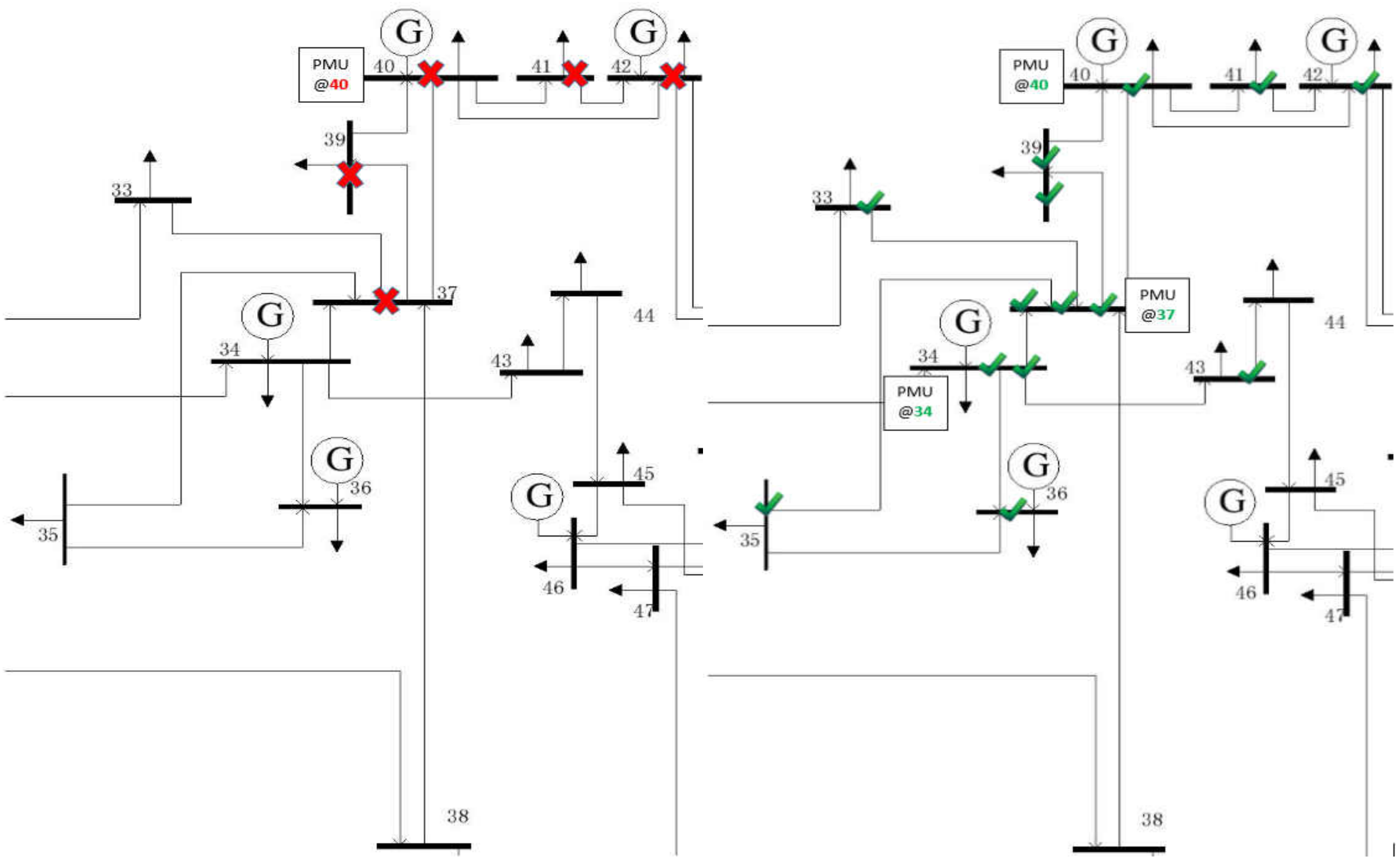


Figure 24 Case (1): Loss of PMU at Bus 40 (left) and with ORC Enforced

☆ - PMU placement buses in normal case
◆ - PMU placement buses with ORC enforced
● - Critical bus
(G) - Generator
↓ -Loads

Scenario 2: Similarly, with normal conditions, PMUs at bus 40 and 57 will observe buses 37, 39, 40, 41, 42 and 49, 53, 54, 55, 56, 59 respectively. However, if any of these PMUs fail, there will be a un-observability in IEEE 118 bus system including two critical buses 37 and 59 that are inter-connected to 14 other buses. This counts towards 9.32% un-observability in the total system. But when ORC is enforced, this un-observability is prevented. Both the PMUs at 37 and 59 will observe the critical buses redundantly zeroing down the un-observability from 9.32%. There are 7 case studies that are conducted using the IEEE 118 bus system as shown in Figure 23 where the reliability of 100% is obtained by enforcing ORC in our formulations. There are many other scenarios where the similar contingency cases can happen, and applying ORC rule will prove useful, and reliable. The contingency scenarios are listed in Table 14 with 1, 2, 3, 4, 5, 6, and 7 number of PMU failures. Table 14 provides the following information:

1. What if certain PMU(s) are lost, and which buses are getting affected?
2. If there is a loss of critical bus and which buses are unobservable?
3. Percentage of un-observability due to the loss of PMU
4. Reliability Results with ORC is enforced.
5. Percentage of un-observability during the loss of PMU(s) and when ORC is enforced.

This is an another example which demonstrate the effectiveness of ORC index in assisting the utilities to keep the full-observability of critical buses.

4.1.8 Time Computation Using LP

In addition, the LP run-time is calculated by measuring the computational (CPU) run time for each of the test cases. Table 15 show the time taken by the LP method using AMPL and CPLEX solver. The computational results of LP approach with and without ORC were compared with existing literatures [24]. From the Table 15, it is seen that it takes virtually zero-seconds to compute the optimal solutions for all test cases. The other methods takes some measureable amount of time to find the optimal solutions. All the test cases ran on the following configured PC system:

Processor: Intel Xeon W3503 @2.4 GHz

Installed Memory: 8 GB

Hard Drive: 465 GB

Table 15 Computation Time for LP Method

Test Bus Systems	**ILP Method (milli seconds)**	**ILP Method with ORC (milli seconds)**	**Other Methods** [63] **(milli seconds)**
IEEE 14 bus	0.00	0.00	660
IEEE 30 bus	15.6	15.6	830
IEEE 57 bus	15.6	31.2	870
IEEE 118 bus	15.6	46.8	1340
SRIPG 208 bus	15.6	46.8	-
IEEE 300 bus	15.6	46.8	-

4.1.9 Forecasting the Scalability of OPP Using Linear Regression

In [15], the authors have made a conclusion that only one fourth to one third of the system buses are need to be provided with PMUs in order to make a system fully-observable. Based on our LP results, the estimation of the number of PMUs required for random bus systems was done by utilizing the well-known forecasting Linear Regression (LR) method. LR is an statistical approach for modeling the relationship between the dependent and independent variables. It is a simple, yet powerful approach to predict the value of a variable based on the value of another variable.

Table 16 Estimation of Number of PMUs Required

No. of Buses in a System (X)	Minimum Observability		Percentage of Observability (No. of PMU placed Buses in the System)
	Estimated no of PMUs Using Linear Regression (Y)	Through ILP Approach	
7	2	2	28.5
14	4	4	28.5
30	9	10	30
57	16	17	28.07
100*	28	*	28
118	33	32	27.11
200*	56	*	28
208	58	55	26.44
300	84	87	29
500*	140	*	28
1000*	280	*	28
1500*	421	*	28.06
2000*	561	*	28.05
3000*	842	*	28.06

* No bus systems are practically available so there are the no solutions

Using the technique of linear regression with least squares method, the number of PMUs required for different bus systems are estimated using the equations (44) to (46).

$$Y = \alpha + \beta * X \tag{44}$$

Where,

$$\alpha = \bar{Y} - \beta * \bar{X} \tag{45}$$

$$\beta = \frac{\sum_{i=1}^{n}(X_i - \bar{X}) * (Y_i - \bar{Y})}{\sum_{i=1}^{n}(X_i - \bar{X})^2} \tag{46}$$

$\bar{X}$ and $\bar{Y}$ are the averages of X and Y variables respectively. The values of X and Y have been considered from our results in Table 6 and Table 7. X refers to the number of buses in the systems while Y refers to the number of PMUs required. Table 16 provides the estimation of the PMUs required w.r.t the number of buses in the system.

Column 2 show the number of PMUs estimated through linear regression while column 3 provides the actual results obtained through ILP approach. From column 4, the percentage of full-observability in any system is around 30% which indicate that approximately 30% of the buses in a system needs to be equipped with PMUs to make the system fully-observable.

4.2 Visualization of Synchrophasor Data

Using an unit-circle representation, the developed IEEE 14 bus system is ran for 1, 5, and 10 second intervals in MATLAB. The voltage and current phasor data from all the 14 buses are visualized. A MATLAB code is written for the synchrophasor data visualization which is detailed in appendix A. Figure 25 represents a voltage phasor measurements at t=1 second. Each of the 14 circles represent 14 buses in the system thereby assisting the ISO in monitoring the system at the ground level.

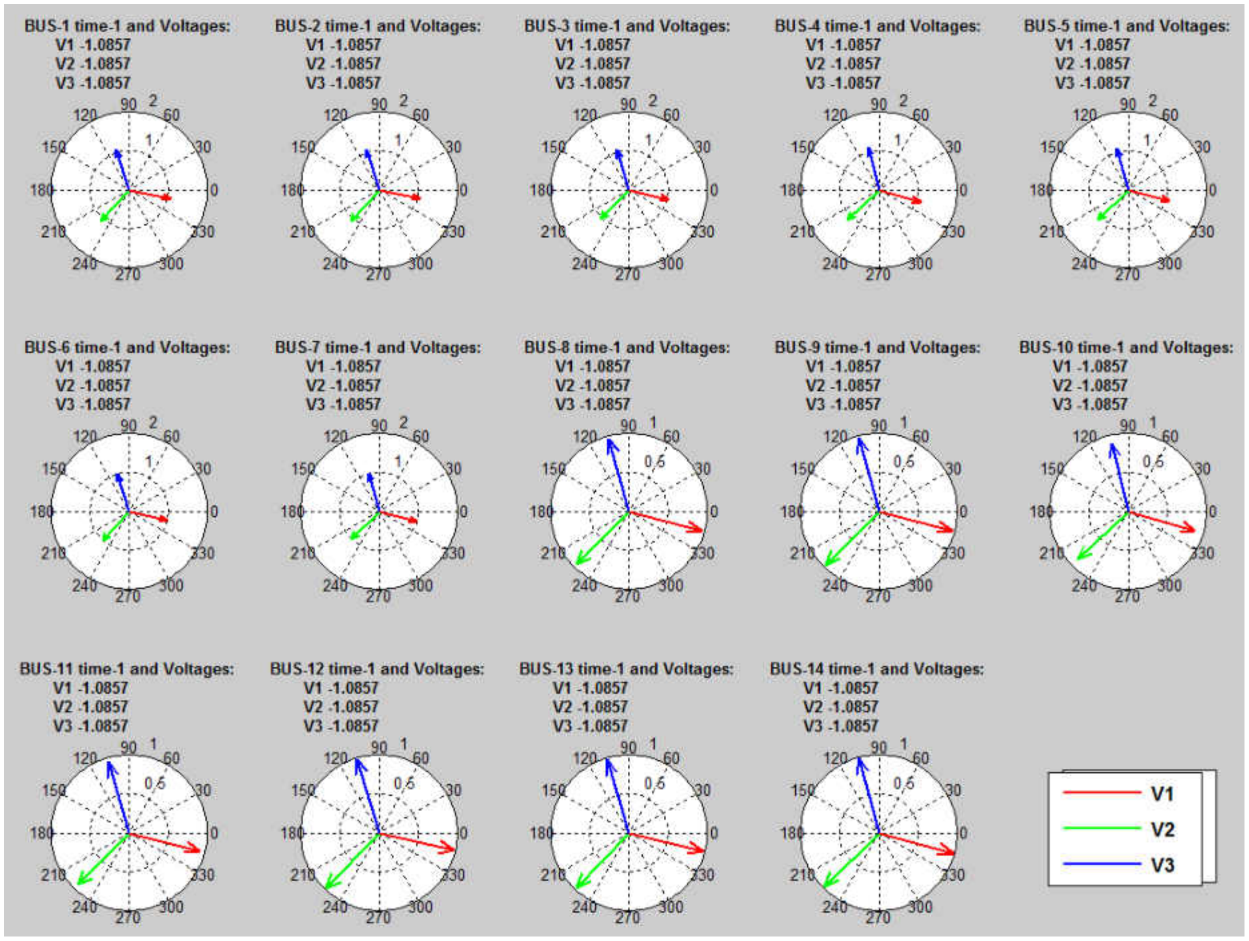


Figure 25 Unit Circle Representation of 14 bus 3-ϕ Voltage Phasors

Each of the circles show three phase voltage phasors represented by red, green, blue colored arrows. Each of the arrows collectively represent the voltage magnitude and the phase angle on each line. A similar representation is obtained for three phase current phasors at all 14 buses respectively. Figure 26 show screenshot of current phasors at 14 buses at t =1 second. Figure 27 shows the snapshot of the visualization of voltage phasor from openPDC. As the data streams in to the openPDC, the visualization tool runs to provide a quick overview on the voltage profile of the grid. The ISO will have voltage thresholds already fixed for the system.

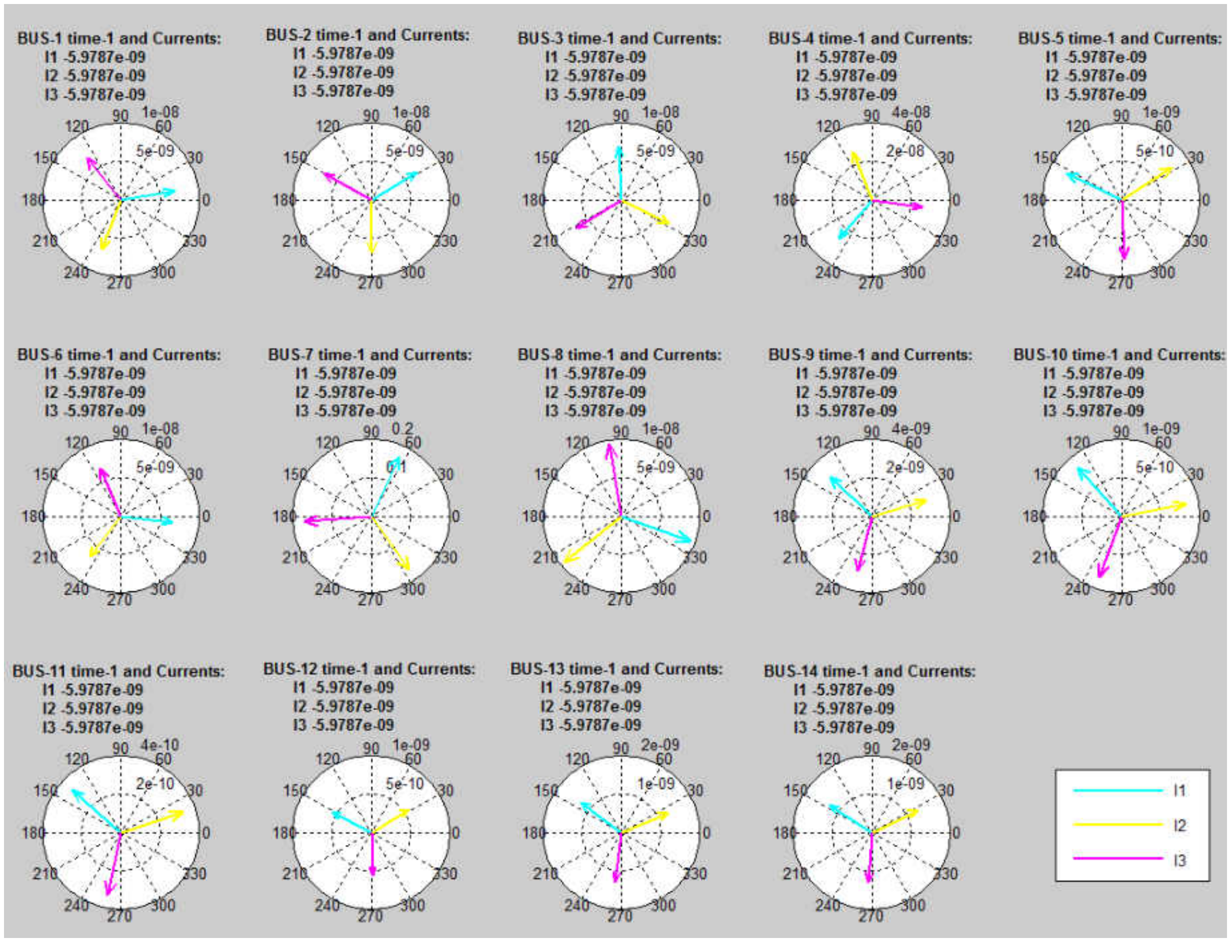


Figure 26 Unit Circle Representation of 14 Bus 3-ϕ Current Phasors

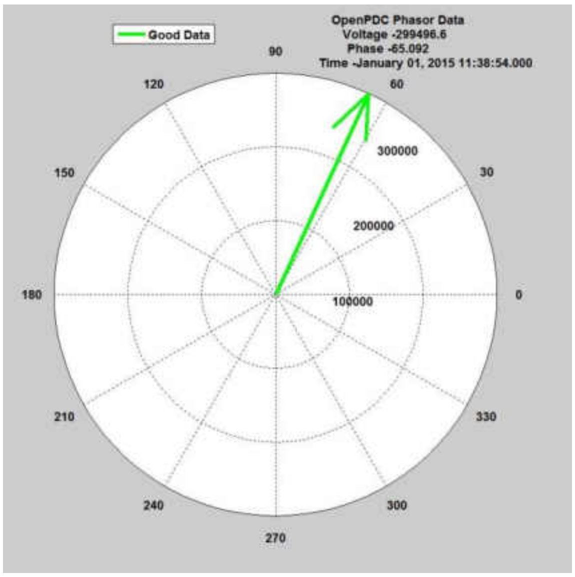


Figure 27 Unit Circle Representation of Voltage from openPDC Data Source

4.3 Synchrophasor Data Mining Using DBSCAN

This section details the various DM case studies studied using DBSCAN from the data generated from two sources, IEEE 14 bus system model and the openPDC. The different cases summarize how DBSCAN method clusters. The four case studies considered here are:

1. Steady state condition
2. High load condition
3. Light load condition
4. Fault condition

While considering the IEEE model, the system is ran for 1 second for each case at a sampling rate of 0.02 seconds while the openPDC data is a continuous data set and I have considered 20 minutes of data. Therefore, the granularity of the data points is very high. The data generated from the simulation model mimics the synchrophasor data and is visualized through unit-circle representation. It is also then provided as an input to the DBSCAN clustering algorithm [64].

4.3.1 Case 1: Under steady State Conditions

Under normal conditions, all the systems is running normally. All the three phase voltage phasors from the 14 buses are provided as input to DBSCAN. DBSCAN then clusters based on its principles of core (green), border (yellow), and noise (red) data points. Figure 28 show the clustered output of a normal condition using DBSCAN. Data from all fourteen buses is clustered together into a core cluster. This is because the voltage remains constant and the density of points around that value form a core cluster. The voltage magnitude, phase angle, and time represent the X,Y, and Z axes respectively. it can be observed that all data points from 14 buses mostly lie within the ranges of 0.98 p.u to 1.02 p.u. The three phase voltages from all 14 buses are separated by 120 ° approximately on the Y axis.

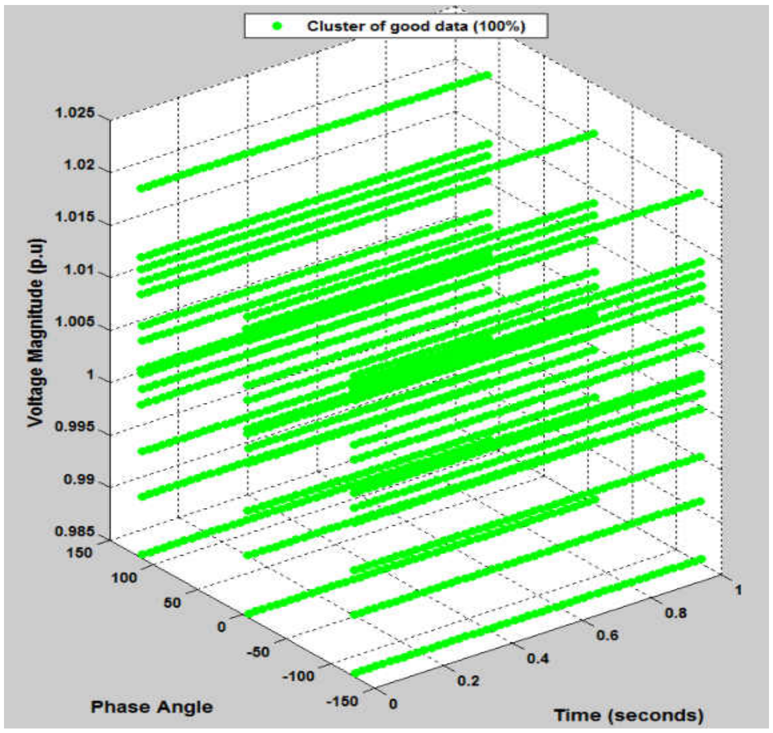


Figure 28 3-ϕ Voltage Data from IEEE Test System during Steady-State

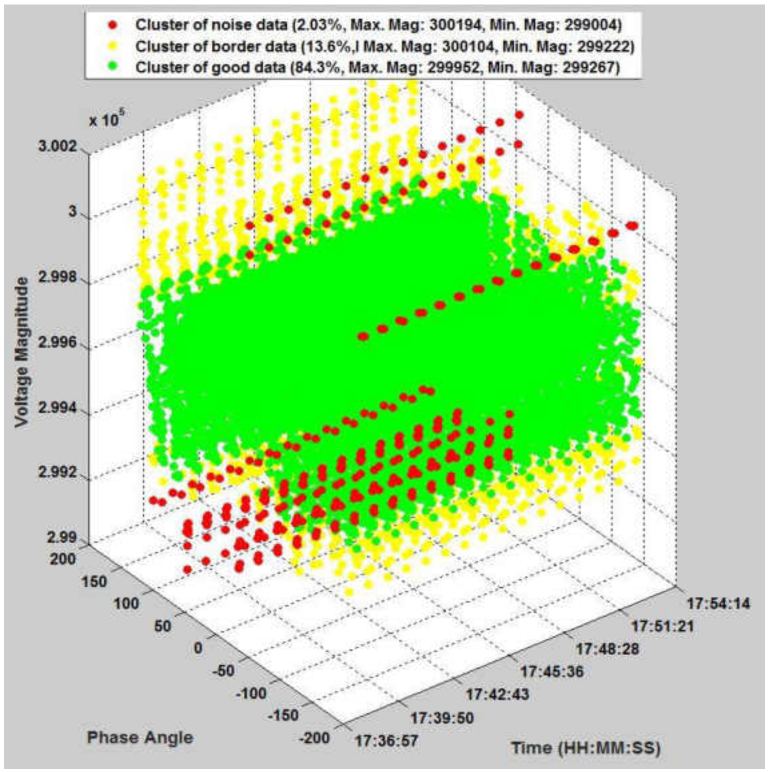


Figure 29 Voltage Phasor Data from openPDC during Steady-State

While the Figure 29 show the clustered output of DBSCAN using openPDC data source. DBSCAN clusters the data points that are equal or nearer to the ideal transmission-line voltage (300 kV) as core points and the values that are slightly away from nominal as border points. The farther points are clustered as noise points.

4.3.2 Case 2: Heavy-Load Condition

In real-world scenario, systems never tend to be match the ideal conditions. Heavy-load conditions occur due to the differences between the generation and consumption capacities. The power grids are often overloaded with demand requests due to shortage of power. Figure 30 represents clustered output from DBSCAN of a heavy-load scenario. It is to be noted that the line voltages are dropped from normal level to around 0.87 p.u. In comparison with Figure 28, the shift in density from 0.9 p.u. to 0.87 p.u.

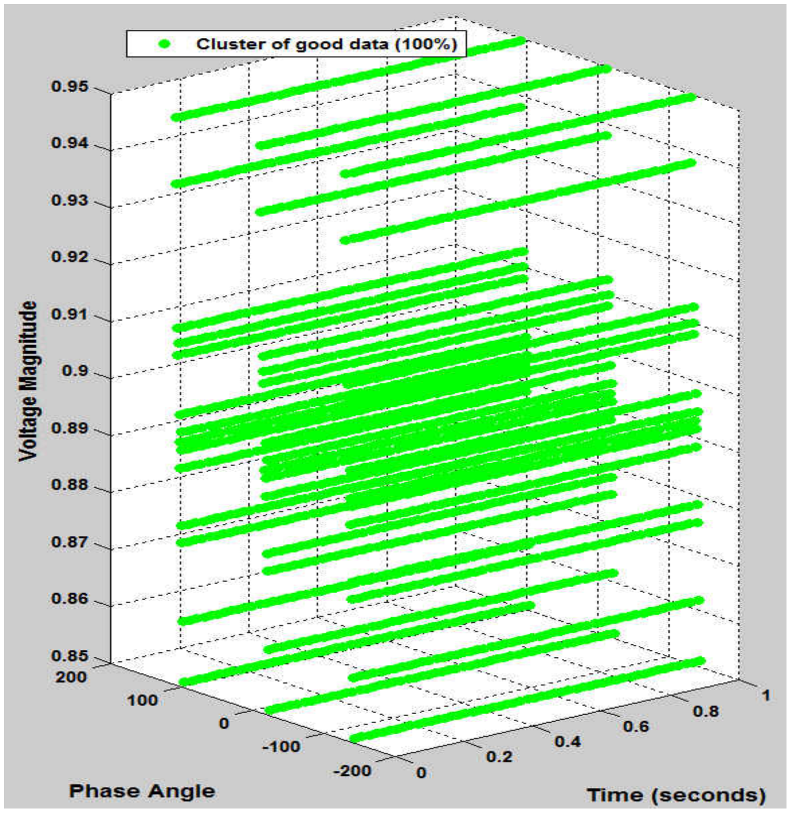


Figure 30 3-ϕ Voltage Data from IEEE Test System during Heavy-Loads

Similarly, the comparison can be done between the openPDC data sets in Figure 29 and Figure 31 where the readers can see the shift in the core points from 299.800 kV to 299.500 kV. Operators can easily identify the densely populated areas (green) and the sparsely populated data points (yellow) in the plot.

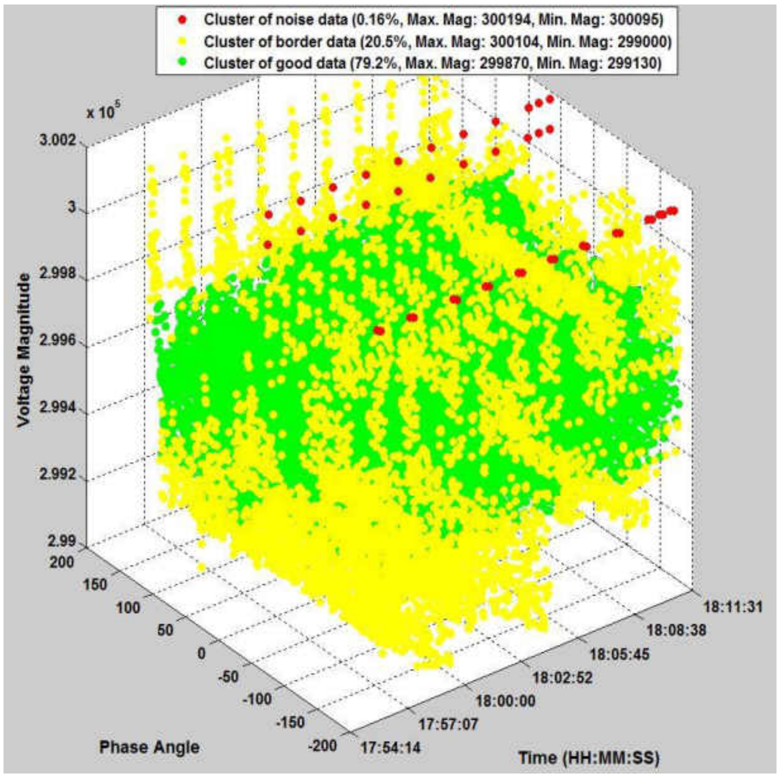


Figure 31 Voltage Phasor Data from openPDC during Heavy-Loads

4.3.3 Case 3: Light-Load Condition

Light loaded situations occur when demand decreases and generation levels remain high. This type of situation generally occurs during the night. This is because the energy usage is at very low-level. Figure 32 show lightly loaded condition of the test system clustered using DBSCAN. The density of points in this light-load condition are shifted towards higher voltages. In comparing Figure 28 and Figure 32, the level shift between the densities of points from 0.98 p.u to 1.07 p.u is visibly seen.

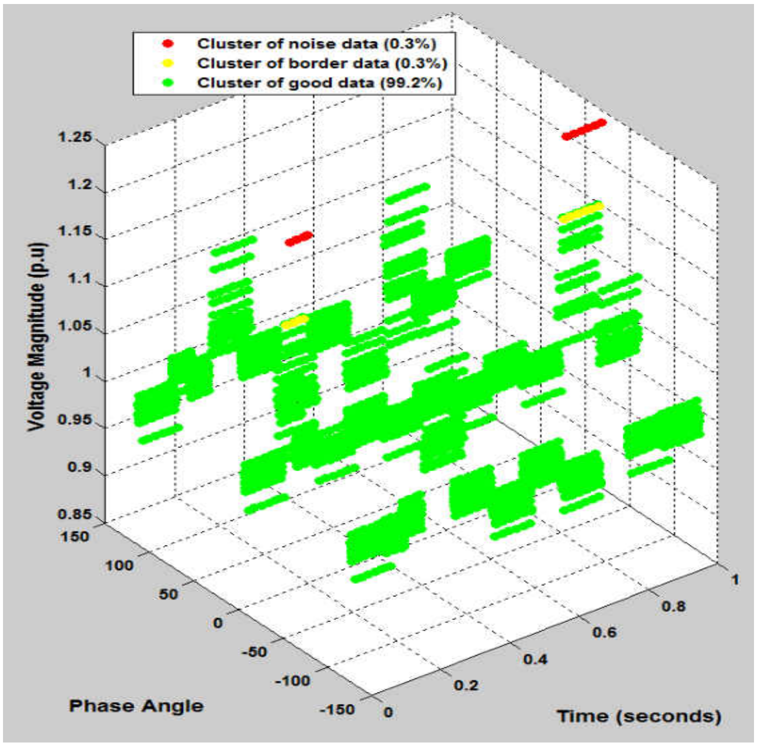


Figure 32 3-ϕ Voltage Data from IEEE Test System during Light-Loads

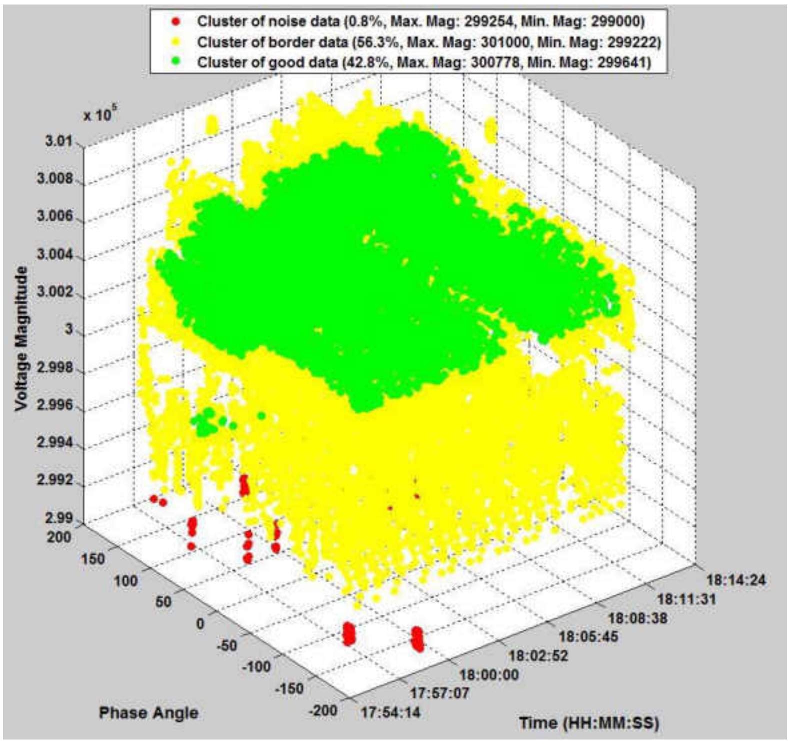


Figure 33 Voltage Phasor Data from openPDC during Light-Loads

By observing Figure 29 and Figure 33, readers can spot the shift in the core points from 299.800 kV to 300.600 kV for the openPDC data sets.

4.3.4 *Case 4: Phase-to-Ground Fault Condition*

A fault condition refers to a phase to ground fault. In IEEE model, it occurs during the time span of 0.6 and 0.7 seconds while in the openPDC data source, the fault happens and then gets cleared. Figure 34 display the DBSCAN output of the fault condition for the simulink model, wherein at the time of fault, the voltage drops from 1 p.u to 0 p.u. DBSCAN successfully captures the fault and groups them into a noise (red) cluster.

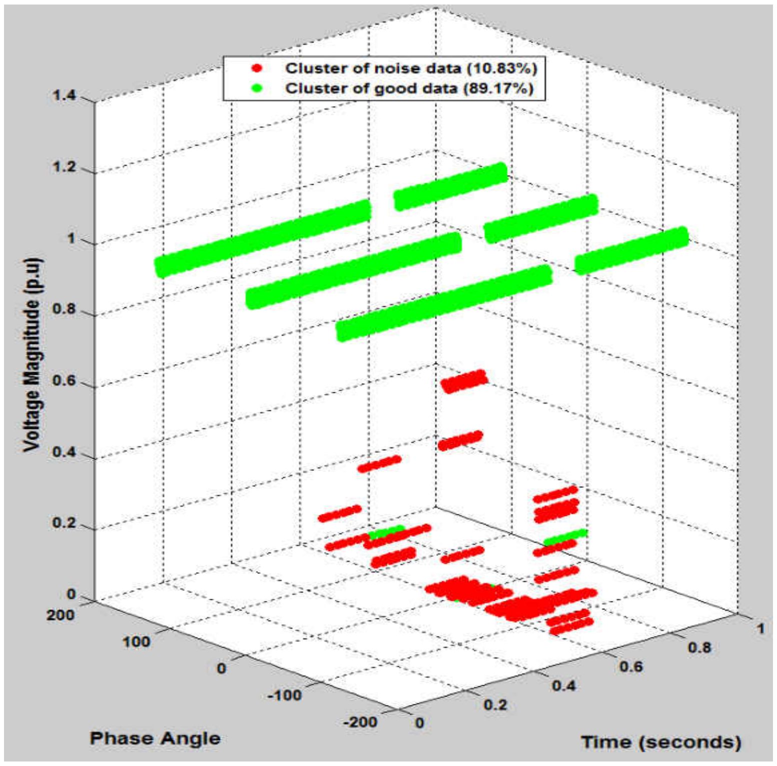


Figure 34 3-ϕ Voltage Data from IEEE Test System during Fault

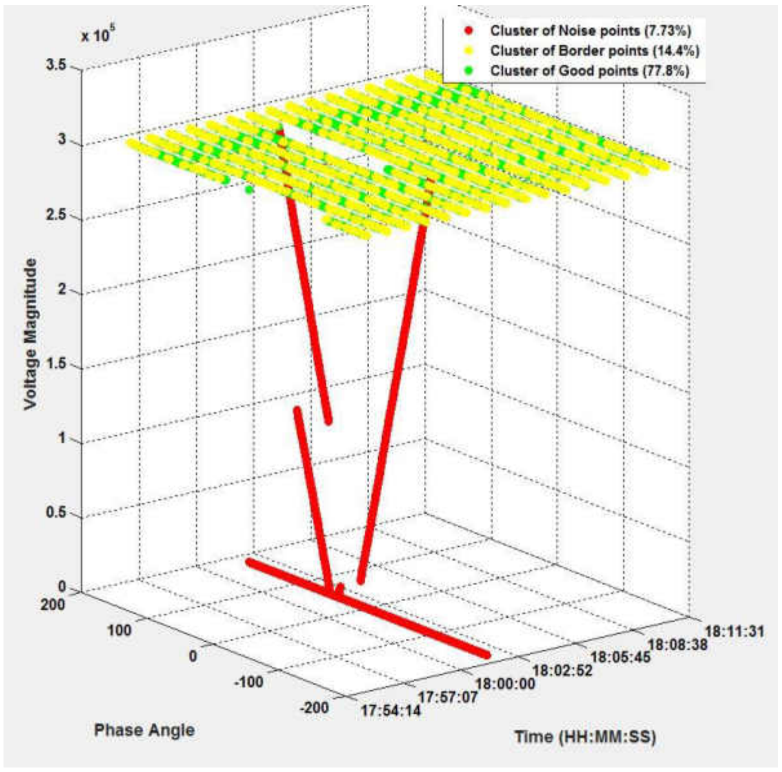


Figure 35 Voltage Phasor Data from openPDC during Fault

The Figure 35 represents the fault data from openPDC source. From the figure, it can be seen how distinctly the DBSCAN clusters the good and fault data. As the voltage decreases, the data points away from the normal ranges are clustered into noise. From all the 4 cases, it can be said that DBSCAN performs well in differentiating the good and bad data points. It also assists the ISO in approximating the voltage values using the density of core, border, and noise points. Thus, different anomalies in the smart-grid parameters are captured using proposed DBSCAN technique.

CHAPTER 5

CONCLUSION & FUTURE WORK

An approach to solve the Phasor Measurement Unit (PMU) placement problem using Linear Programming (LP) formulations is investigated. Several test system such as IEEE and Southern Regional Indian Power Grid (SRIPG) were modeled and studied. The proposed approach uses zero-injection buses to further reduce the number of PMUs required for complete observability of the system. In addition, an ORC index is implemented to keep the system fully-observable for high reliability. The results are promising and proven effective for the test cases tested [65]. The economic benefits in the placement problem of synchrophasors are also realized for smaller utilities using the LP approach. The results indicate that proposed approach is 100% reliable, observable and fully-scalable. An estimation technique to determine the number of PMUs required for planning is proposed using a well-known Linear regression model.

A simulink model of IEEE 14 bus system is developed to generate the voltage and current phasor data [66]. The problem of visualizing the synchrophasor data is solved using an unit-circle representation. A data-mining technique known as DBSCAN is applied for visualizing and representing the complex-phasor data. Several fault conditions are modeled to study the meaningful application of data-mining algorithms for decision-making process [67]. In a nut shell, my contributions can be summarized as follows:

1. The optimal placement problem of PMUs has been successfully solved using the linear programming formulations.

2. An ORC index has been proposed to include the grid’s reliability as well as the economic aspects while solving the optimal placement problem of PMUs.
3. A design of IEEE 14 bus system model and an open source software called as openPDC have been developed and integrated to MATLAB environment for generating phasor data.
4. An unit circle representation has been developed to visualize the phasor data.
5. A density based clustering algorithm called as DBSCAN is implemented to analyze the phasor data.

My work started with finding an optimal solutions for PMU placement followed by synchrophasor data visualization using a unit circle representation and a clustering technique. The future work can investigate other data clustering techniques for classification and mining of phasor data sets using neural network, structured prediction model, regression, fuzzy logics using our IEEE 14 bus system and openPDC platform. Furthermore, issues concerned with the security of synchrophasor data are also needed to be addressed to transform the existing power grids into smarter grids.

RESEARCH PUBLICATIONS

Transactions/Journals:

1. Mukherjee, **R. Vallakati**, and P. Ranganathan, "**A Framework for Situational Awareness Using openPDC**," *IEEE Trans. Smart Grid.* (Accepted)
2. **R. Vallakati**, A. Mukherjee, and P. Ranganathan (2015) "**Situational Awareness using DBSCAN in Smart-Grid**", *Smart Grid and Renewable Energy, Vol.6 No.5 2015*

   Web link: http://dx.doi.org/10.4236/sgre.2015.65011

Conference Publications:

1. **R. Vallakati**, A. Mukherjee and P. Ranganathan**, "Preserving Observability in Smart Grid using Phasor Measurement Unit by applying Optimal Redundancy Criteria (ORC)"** *in IEEE First ICDCM Conference, Atlanta, June 7-10, 2015.*
2. A. Mukherjee, **R. Vallakati** and P. Ranganathan, "**Using Phasor data for Visualization and Data Mining Purposes in Power Systems** ", in *IEEE First ICDCM Conference, Atlanta, June 7-10, 2015.*
3. R. Mahmud, **R. Vallakati**, A. Mukherjee, P. Ranganathan, and A. Nejadpak, "**A Survey on Attacks on Smart Grid Metering Infra-Structures: Threats and Solutions***", in IEEE EIT Conference, Illinois, May 23- 25 2015.*
4. N. Gellerman, P. Ranganathan, **R. Vallakati**, and A. Mukherjee**, "User interface for situational awareness of openPDC,"** in 2014 *North American Power Symposium (NAPS),* pp. 1–6, 2014.

   Web link: http://dx.doi.org/10.1109/NAPS.2014.6965418

APPENDICES

APPENDIX A

AMPL CODE FOR IEEE 14 BUS SYSTEM

```
var a{1..14} binary;
#Number of buses in the system
param pmu,default 14;
minimize z:sum{i in 1..14}(a[i]);
# objective function of the problem. Minimizing the number of PMUs required.
s.t. c1:a[1]+a[2]+a[5]>=1;
s.t. c2:a[1]+a[2]+a[3]+a[4]+a[5]>=1;
s.t. c3:a[2]+a[3]+a[4]>=1;
s.t. c4:a[2]+a[3]+a[4]+a[5]+a[8]+a[9]>=1;
s.t. c5:a[1]+a[2]+a[4]+a[5]>=1;
s.t. c6:a[6]+a[11]+a[12]+a[13]>=1;
s.t. c7:a[4]+a[7]+a[8]+a[9]>=1;
s.t. c8:a[4]+a[8]+a[9]>=1;
s.t. c9:a[4]+a[8]+a[9]+a[10]+a[14]>=1;
s.t. c10:a[9]+a[10]+a[11]>=1;
s.t. c11:a[6]+a[10]+a[11]>=1;
s.t. c12:a[6]+a[12]+a[13]>=1;
s.t. c13:a[6]+a[12]+a[13]+a[14]>=1;
s.t. c14:a[9]+a[13]+a[14]>=1;
option solver cplex;
solve;
display a,z;
printf "CompletionTime %s.\n", ctime();
display _total_solve_time;
```

APPENDIX B

AMPL CODE FOR IEEE 30 BUS SYSTEM

```
var a{1..30} binary;
param pmu,default 30;
minimize z:sum{i in 1..30}(a[i]);
# objective function of the problem. Minimizing the number of PMUs required.
s.t. c1:a[1]+a[2]+a[3]>=1;
s.t. c2:a[1]+a[2]+a[4]+a[5]+a[6]>=1;
s.t. c3:a[1]+a[3]+a[4]>=1;
s.t. c4:a[2]+a[3]+a[4]+a[6]+a[12]>=1;
s.t. c5:a[2]+a[5]+a[7]>=1;
s.t. c6:a[2]+a[4]+a[6]+a[7]+a[8]+a[9]+a[10]+a[28]>=1;
s.t. c7:a[5]+a[6]+a[7]>=1;
s.t. c8:a[6]+a[8]+a[28]>=1;
s.t. c9:a[6]+a[9]+a[10]+a[11]>=1;
s.t. c10:a[9]+a[10]+a[17]+a[20]+a[21]>=1;
s.t. c11:a[9]+a[11]>=1;
s.t. c12:a[12]+a[13]+a[14]+a[15]+a[16]>=1;
s.t. c13:a[12]+a[13]>=1;
s.t. c14:a[12]+a[14]+a[15]>=1;
s.t. c15:a[12]+a[14]+a[15]+a[18]+a[23]>=1;
s.t. c16:a[12]+a[16]+a[17]>=1;
s.t. c17:a[10]+a[16]+a[17]>=1;
s.t. c18:a[15]+a[18]+a[19]>=1;
s.t. c19:a[18]+a[19]+a[20]>=1;
s.t. c20:a[10]+a[19]+a[20]>=1;
s.t. c21:a[10]+a[21]+a[22]>=1;
s.t. c22:a[10]+a[21]+a[22]>=1;
s.t. c23:a[23]+a[24]+a[15]>=1;
s.t. c24:a[23]+a[24]+a[25]>=1;
s.t. c25:a[24]+a[25]+a[26]+a[27]>=1;
s.t. c26:a[25]+a[26]>=1;
s.t. c27:a[25]+a[27]+a[28]+a[29]+a[30]>=1;
s.t. c28:a[6]+a[8]+a[27]+a[28]>=1;
s.t. c29:a[27]+a[29]+a[30]>=1;
s.t. c30:a[27]+a[29]+a[30]>=1;
option solver cplex;
solve;
display a,z;
printf "CompletionTime %s.\n", ctime();
display _total_solve_time;
```

APPENDIX C

AMPL CODE FOR IEEE 57 BUS SYSTEM

```
var a{1..57} binary; #1
#param pmu,default 14;
minimize z:sum{i in 1..57}(a[i]);

s.t. c1:a[1]+a[2]+a[15]+a[16]+a[17]>=1;
s.t. c2:a[1]+a[2]+a[3]>=1;
s.t. c3:a[2]+a[3]+a[4]+a[15]>=1;
s.t. c4:a[3]+a[4]+a[5]+a[6]+a[18]>=1;
s.t. c5:a[4]+a[5]+a[6]>=1;
s.t. c6:a[4]+a[5]+a[6]+a[7]+a[8]>=1;
s.t. c7:a[6]+a[7]+a[8]+a[29]>=1;
s.t. c8:a[6]+a[7]+a[8]+a[9]>=1;
s.t. c9:a[8]+a[9]+a[10]+a[11]+a[12]+a[13]+a[55]>=1;

s.t. c10:a[9]+a[10]+a[12]+a[51]>=1;
s.t. c11:a[9]+a[11]+a[13]+a[41]+a[43]>=1;
s.t. c12:a[9]+a[10]+a[12]+a[13]+a[16]+a[17]>=1;
s.t. c13:a[9]+a[11]+a[12]+a[13]+a[14]+a[15]+a[49]>=1;
s.t. c14:a[13]+a[14]+a[15]+a[46]>=1;
s.t. c15:a[13]+a[14]+a[15]+a[1]+a[3]+a[45]>=1;
s.t. c16:a[1]+a[12]+a[16]>=1;
s.t. c17:a[1]+a[12]+a[17]>=1;
s.t. c18:a[4]+a[18]+a[19]>=1;
s.t. c19:a[18]+a[19]+a[20]>=1;
s.t. c20:a[19]+a[20]+a[21]>=1;

s.t. c21:a[20]+a[21]+a[22]>=1;
s.t. c22:a[21]+a[22]+a[23]+a[38]>=1;
s.t. c23:a[22]+a[23]+a[24]>=1;
s.t. c24:a[23]+a[24]+a[25]+a[26]>=1;
s.t. c25:a[24]+a[25]+a[30]>=1;
s.t. c26:a[24]+a[26]+a[27]>=1;
s.t. c27:a[26]+a[27]+a[28]>=1;
s.t. c28:a[27]+a[28]+a[29]>=1;
s.t. c29:a[7]+a[28]+a[29]+a[52]>=1;
s.t. c30:a[25]+a[30]+a[31]>=1;

s.t. c31:a[30]+a[31]+a[32]>=1;
s.t. c32:a[31]+a[32]+a[33]+a[34]>=1;
s.t. c33:a[32]+a[33]>=1;
```

```
s.t. c34:a[32]+a[34]+a[35]>=1;
s.t. c35:a[34]+a[35]+a[36]>=1;
s.t. c36:a[35]+a[36]+a[37]+a[40]>=1;
s.t. c37:a[36]+a[37]+a[38]+a[39]>=1;
s.t. c38:a[37]+a[38]+a[22]+a[44]+a[48]+a[49]>=1;
s.t. c39:a[37]+a[39]+a[57]>=1;
s.t. c40:a[36]+a[40]+a[56]>=1;
s.t. c41:a[11]+a[41]+a[42]+a[43]+a[56]>=1;
s.t. c42:a[41]+a[42]+a[56]>=1;
s.t. c43:a[11]+a[41]+a[43]>=1;
s.t. c44:a[38]+a[44]+a[45]>=1;
s.t. c45:a[15]+a[44]+a[45]>=1;
s.t. c46:a[14]+a[46]+a[47]>=1;
s.t. c47:a[46]+a[47]+a[48]>=1;
s.t. c48:a[38]+a[47]+a[48]+a[49]>=1;
s.t. c49:a[13]+a[38]+a[48]+a[49]+a[50]>=1;
s.t. c50:a[49]+a[50]+a[51]>=1;
s.t. c51:a[10]+a[50]+a[51]>=1;
s.t. c52:a[29]+a[52]+a[53]>=1;
s.t. c53:a[52]+a[53]+a[54]>=1;
s.t. c54:a[53]+a[54]+a[55]>=1;
s.t. c55:a[9]+a[54]+a[55]>=1;
s.t. c56:a[40]+a[41]+a[42]+a[56]+a[57]>=1;
s.t. c57:a[39]+a[56]+a[57]>=1;

option solver cplex;
solve;
display a,z;
printf "CompletionTime %s.\n", ctime();
display _total_solve_time;
```

APPENDIX D

AMPL CODE FOR IEEE 118 BUS SYSTEM

```
var a{1..118} binary; #1
#var u{1..14} binary;
#param pmu,default 14;
minimize z:sum{i in 1..118}(a[i]);

s.t. c1:a[1]+a[2]+a[3]>=1;
s.t. c2:a[1]+a[2]+a[12]>=1;
s.t. c3:a[1]+a[3]+a[12]+a[5]>=1;
s.t. c4:a[4]+a[5]+a[11]>=1;
s.t. c5:a[3]+a[4]+a[5]+a[6]+a[8]+a[11]>=1;
s.t. c6:a[5]+a[6]+a[7]>=1;
s.t. c7:a[6]+a[7]+a[12]>=1;
s.t. c8:a[5]+a[8]+a[9]+a[30]>=1;
s.t. c9:a[8]+a[9]+a[10]>=1;
s.t. c10:a[9]+a[10]>=1;

s.t. c11:a[4]+a[5]+a[11]+a[12]+a[13]>=1;
s.t. c12:a[2]+a[3]+a[7]+a[11]+a[12]+a[14]+a[16]+a[117]>=1;
s.t. c13:a[11]+a[13]+a[15]>=1;
s.t. c14:a[12]+a[14]+a[15]>=1;
s.t. c15:a[13]+a[14]+a[15]+a[17]+a[19]>=1;
s.t. c16:a[12]+a[16]+a[17]>=1;
s.t. c17:a[15]+a[16]+a[17]+a[18]+a[30]+a[31]+a[113]>=1;
s.t. c18:a[17]+a[18]+a[19]>=1;
s.t. c19:a[15]+a[18]+a[19]+a[20]+a[34]>=1;
s.t. c20:a[19]+a[20]+a[21]>=1;

s.t. c21:a[20]+a[21]+a[22]>=1;
s.t. c22:a[21]+a[22]+a[23]>=1;
s.t. c23:a[22]+a[23]+a[24]+a[25]+a[32]>=1;
s.t. c24:a[23]+a[24]+a[70]+a[72]>=1;
s.t. c25:a[23]+a[25]+a[26]+a[27]>=1;
s.t. c26:a[25]+a[26]+a[30]>=1;
s.t. c27:a[26]+a[27]+a[28]+a[32]+a[115]>=1;
s.t. c28:a[27]+a[28]+a[29]>=1;
s.t. c29:a[28]+a[29]+a[31]>=1;
s.t. c30:a[8]+a[17]+a[26]+a[30]+a[38]>=1;

s.t. c31:a[17]+a[29]+a[31]+a[32]>=1;
s.t. c32:a[23]+a[27]+a[31]+a[32]+a[113]+a[114]>=1;
s.t. c33:a[15]+a[33]+a[37]>=1;
s.t. c34:a[19]+a[34]+a[36]+a[37]+a[43]>=1;
```

```
s.t. c35:a[35]+a[36]+a[37]>=1;
s.t. c36:a[34]+a[35]+a[36]>=1;
s.t. c37:a[33]+a[34]+a[35]+a[37]+a[38]+a[39]+a[40]>=1;
s.t. c38:a[30]+a[37]+a[38]+a[65]>=1;
s.t. c39:a[37]+a[39]+a[40]>=1;
s.t. c40:a[37]+a[39]+a[40]+a[41]+a[42]>=1;

s.t. c41:a[40]+a[41]+a[42]>=1;
s.t. c42:a[40]+a[41]+a[42]+a[49]+a[49]>=1;
s.t. c43:a[34]+a[43]+a[44]>=1;
s.t. c44:a[43]+a[44]+a[45]>=1;
s.t. c45:a[44]+a[45]+a[46]+a[49]>=1;
s.t. c46:a[45]+a[46]+a[47]+a[48]>=1;
s.t. c47:a[46]+a[47]+a[49]+a[69]>=1;
s.t. c48:a[46]+a[48]+a[49]>=1;
s.t. c49:a[42]+a[45]+a[47]+a[48]+a[49]+a[50]+a[51]+a[54]+a[54]+a[66]+a[66]>=1;
s.t. c50:a[49]+a[50]+a[57]>=1;

s.t. c51:a[49]+a[51]+a[52]+a[58]>=1;
s.t. c52:a[51]+a[52]+a[53]>=1;
s.t. c53:a[52]+a[53]+a[54]>=1;
s.t. c54:a[49]+a[53]+a[54]+a[55]+a[56]+a[59]>=1;
s.t. c55:a[54]+a[55]+a[56]+a[59]>=1;
s.t. c56:a[54]+a[55]+a[56]+a[57]+a[58]+a[59]>=1;
s.t. c57:a[50]+a[56]+a[57]>=1;
s.t. c58:a[51]+a[56]+a[58]>=1;
s.t. c59:a[54]+a[55]+a[56]+a[59]+a[60]+a[61]+a[63]>=1;
s.t. c60:a[59]+a[60]+a[61]+a[62]>=1;

s.t. c61:a[59]+a[60]+a[61]+a[62]+a[64]>=1;
s.t. c62:a[60]+a[61]+a[62]+a[66]+a[67]>=1;
s.t. c63:a[59]+a[63]+a[64]>=1;
s.t. c64:a[61]+a[63]+a[64]+a[65]>=1;
s.t. c65:a[38]+a[64]+a[65]+a[66]+a[68]>=1;
s.t. c66:a[49]+a[49]+a[62]+a[65]+a[66]+a[67]>=1;
s.t. c67:a[62]+a[66]+a[67]>=1;
s.t. c68:a[65]+a[68]+a[69]+a[116]>=1;
s.t. c69:a[47]+a[49]+a[68]+a[69]+a[70]+a[75]+a[77]>=1;
s.t. c70:a[24]+a[69]+a[70]+a[71]+a[74]+a[75]>=1;

s.t. c71:a[70]+a[71]+a[72]+a[73]>=1;
s.t. c72:a[24]+a[71]+a[72]>=1;
s.t. c73:a[71]+a[73]>=1;
s.t. c74:a[70]+a[74]+a[75]>=1;
s.t. c75:a[70]+a[74]+a[75]+a[77]+a[118]>=1;
s.t. c76:a[76]+a[77]+a[118]>=1;
```

s.t. c77:a[69]+a[75]+a[76]+a[77]+a[78]+a[80]+a[80]>=1;
s.t. c78:a[77]+a[78]+a[79]>=1;
s.t. c79:a[78]+a[79]+a[80]>=1;
s.t. c80:a[77]+a[77]+a[79]+a[80]+a[81]+a[96]+a[97]+a[98]+a[99]>=1;

s.t. c81:a[80]+a[81]+a[116]>=1;
s.t. c82:a[77]+a[82]+a[83]+a[96]>=1;
s.t. c83:a[82]+a[83]+a[84]+a[85]>=1;
s.t. c84:a[83]+a[84]+a[85]>=1;
s.t. c85:a[83]+a[84]+a[85]+a[86]+a[88]+a[89]>=1;
s.t. c86:a[85]+a[86]+a[87]>=1;
s.t. c87:a[86]+a[87]>=1;
s.t. c88:a[85]+a[88]+a[89]>=1;
s.t. c89:a[85]+a[88]+a[89]+a[90]+a[92]+a[92]>=1;
s.t. c90:a[89]+a[89]+a[90]+a[91]>=1;

s.t. c91:a[90]+a[91]+a[92]>=1;
s.t. c92:a[89]+a[89]+a[91]+a[92]+a[93]+a[94]+a[100]+a[102]>=1;
s.t. c93:a[92]+a[93]+a[94]>=1;
s.t. c94:a[92]+a[93]+a[94]+a[95]+a[96]+a[100]>=1;
s.t. c95:a[94]+a[95]+a[96]>=1;
s.t. c96:a[80]+a[82]+a[94]+a[95]+a[96]+a[97]>=1;
s.t. c97:a[80]+a[96]+a[97]>=1;
s.t. c98:a[80]+a[98]+a[100]>=1;
s.t. c99:a[80]+a[99]+a[100]>=1;
s.t. c100:a[92]+a[94]+a[98]+a[99]+a[100]+a[101]+a[103]+a[104]+a[106]>=1;

s.t. c101:a[100]+a[101]+a[102]>=1;
s.t. c102:a[93]+a[101]+a[102]>=1;
s.t. c103:a[100]+a[103]+a[104]+a[105]+a[110]>=1;
s.t. c104:a[100]+a[103]+a[104]+a[105]>=1;
s.t. c105:a[104]+a[105]+a[106]+a[107]+a[108]>=1;
s.t. c106:a[100]+a[105]+a[106]+a[107]>=1;
s.t. c107:a[105]+a[106]+a[107]>=1;
s.t. c108:a[105]+a[108]+a[109]>=1;
s.t. c109:a[108]+a[109]+a[110]>=1;
s.t. c110:a[103]+a[109]+a[110]+a[111]+a[112]>=1;

s.t. c111:a[110]+a[111]>=1;
s.t. c112:a[110]+a[112]>=1;
s.t. c113:a[17]+a[32]+a[113]>=1;
s.t. c114:a[32]+a[114]+a[115]>=1;
s.t. c115:a[27]+a[114]+a[115]>=1;
s.t. c116:a[68]+a[116]>=1;
s.t. c117:a[12]+a[117]>=1;
s.t. c118:a[75]+a[76]+a[118]>=1;

```
option solver gurobi;
solve;
display a,z;
printf "CompletionTime %s.\n", ctime();
display _total_solve_time;
```

APPENDIX E

AMPL CODE FOR SRIPG 208 BUS SYSTEM

```
var a{1..208} binary; #1
#number of buses in the system
minimize z:sum{i in 1..208}(a[i]);
#Minimizing the number of PMUs required

#Tamil Nadu
s.t. c1:a[1]+a[2]+a[3]+a[4]+a[6]+a[7]+a[9]+a[12]>=1;
s.t. c2:a[1]+a[2]+a[12]>=1;
s.t. c3:a[1]+a[3]+a[7]>=1;
s.t. c4:a[1]+a[4]+a[5]+a[6]>=1;
s.t. c5:a[4]+a[5]+a[9]>=1;
s.t. c6:a[1]+a[4]+a[6]+a[10]>=1;
s.t. c7:a[1]+a[3]+a[7]+a[8]>=1;
s.t. c8:a[7]+a[8]+a[16]+a[25]>=1;
s.t. c9:a[1]+a[5]+a[9]+a[11]>=1;
s.t. c10:a[6]+a[10]>=1;

s.t. c11:a[9]+a[11]+a[12]>=1;
s.t. c12:a[1]+a[2]+a[11]+a[12]+a[13]+a[14]+a[15]+a[16]+a[17]>=1;
s.t. c13:a[12]+a[13]>=1;
s.t. c14:a[12]+a[14]>=1;
s.t. c15:a[12]+a[15]+a[16]>=1;
s.t. c16:a[8]+a[12]+a[15]+a[16]+a[20]>=1;
s.t. c17:a[12]+a[17]+a[20]+a[25]>=1;
s.t. c18:a[18]+a[19]+a[25]>=1;
s.t. c19:a[18]+a[19]+a[21]+a[25]>=1;
s.t. c20:a[16]+a[17]+a[20]+a[22]>=1;

s.t. c21:a[19]+a[21]+a[28]>=1;
s.t. c22:a[20]+a[22]+a[23]>=1;
s.t. c23:a[22]+a[23]+a[24]+a[26]>=1;
s.t. c24:a[23]+a[24]+a[25]+a[27]>=1;
s.t. c25:a[8]+a[17]+a[18]+a[19]+a[24]+a[25]+a[33]+a[35]>=1;
s.t. c26:a[23]+a[26]+a[27]>=1;
s.t. c27:a[24]+a[26]+a[27]+a[30]+a[38]>=1;
s.t. c28:a[21]+a[28]+a[34]+a[35]>=1;
s.t. c29:a[29]+a[31]+a[33]>=1;
s.t. c30:a[27]+a[30]+a[31]+a[36]+a[37]+a[44]>=1;

s.t. c31:a[29]+a[30]+a[31]+a[39]>=1;
s.t. c32:a[32]+a[33]>=1;
s.t. c33:a[25]+a[29]+a[32]+a[33]+a[35]+a[40]+a[41]+a[46]>=1;
```

```
s.t. c34:a[28]+a[34]+a[35]>=1;
s.t. c35:a[25]+a[28]+a[33]+a[34]+a[35]+a[41]+a[46]>=1;
s.t. c36:a[30]+a[36]>=1;
s.t. c37:a[30]+a[37]>=1;
s.t. c38:a[27]+a[38]+a[43]>=1;
s.t. c39:a[31]+a[39]+a[48]>=1;
s.t. c40:a[33]+a[40]+a[42]>=1;

s.t. c41:a[33]+a[35]+a[41]+a[54]>=1;
s.t. c42:a[40]+a[42]+a[47]>=1;
s.t. c43:a[38]+a[43]+a[48]>=1;
s.t. c44:a[30]+a[44]+a[45]+a[50]>=1;
s.t. c45:a[44]+a[45]+a[49]+a[50]>=1;
s.t. c46:a[33]+a[35]+a[46]+a[54]>=1;
s.t. c47:a[42]+a[47]+a[48]+a[53]+a[63]>=1;
s.t. c48:a[39]+a[43]+a[47]+a[48]+a[49]+a[51]+a[52]>=1;
s.t. c49:a[45]+a[48]+a[49]+a[50]>=1;
s.t. c50:a[44]+a[45]+a[49]+a[50]+a[60]>=1;

s.t. c51:a[48]+a[51]+a[60]>=1;
s.t. c52:a[48]+a[52]+a[59]+a[68]>=1;
s.t. c53:a[47]+a[53]+a[67]>=1;
s.t. c54:a[41]+a[46]+a[54]+a[55]+a[57]>=1;
s.t. c55:a[54]+a[55]+a[56]>=1;
s.t. c56:a[55]+a[56]+a[57]+a[58]>=1;
s.t. c57:a[54]+a[56]+a[57]>=1;
s.t. c58:a[56]+a[58]+a[61]>=1;
s.t. c59:a[52]+a[59]+a[60]>=1;
s.t. c60:a[50]+a[51]+a[59]+a[60]+a[69]>=1;

s.t. c61:a[58]+a[61]+a[62]>=1;
s.t. c62:a[61]+a[62]+a[63]>=1;
s.t. c63:a[47]+a[62]+a[63]+a[64]+a[65]+a[66]+a[71]>=1;
s.t. c64:a[63]+a[64]>=1;
s.t. c65:a[63]+a[65]+a[66]+a[67]+a[70]>=1;
s.t. c66:a[63]+a[65]+a[66]+a[71]+a[92]+a[95]>=1;
s.t. c67:a[53]+a[65]+a[67]+a[68]+a[70]+a[73]+a[77]>=1;
s.t. c68:a[52]+a[67]+a[68]+a[73]+a[74]+a[75]>=1;
s.t. c69:a[60]+a[69]+a[72]>=1;
s.t. c70:a[65]+a[67]+a[70]+a[100]>=1;

s.t. c71:a[63]+a[66]+a[71]>=1;
s.t. c72:a[69]+a[72]+a[73]>=1;
s.t. c73:a[67]+a[68]+a[72]+a[73]+a[78]+a[82]>=1;
s.t. c74:a[68]+a[74]+a[75]+a[78]>=1;
s.t. c75:a[68]+a[74]+a[75]+a[76]+a[79]+a[80]+a[81]+a[83]>=1;
```

```
s.t. c76:a[75]+a[76]+a[77]+a[81]>=1;
s.t. c77:a[67]+a[76]+a[77]+a[102]>=1;
s.t. c78:a[73]+a[74]+a[78]>=1;
s.t. c79:a[75]+a[79]>=1;
s.t. c80:a[75]+a[80]+a[83]>=1;

s.t. c81:a[75]+a[76]+a[81]+a[83]>=1;
s.t. c82:a[73]+a[82]+a[83]>=1;
s.t. c83:a[80]+a[81]+a[82]+a[83]+a[75]+a[105]>=1;

#Kerala

s.t. c84:a[84]+a[87]+a[88]>=1;
s.t. c85:a[85]+a[89]>=1;
s.t. c86:a[86]+a[89]>=1;
s.t. c87:a[84]+a[87]+a[88]>=1;
s.t. c88:a[84]+a[87]+a[88]+a[89]>=1;
s.t. c89:a[85]+a[86]+a[88]+a[89]+a[90]+a[91]+a[92]>=1;
s.t. c90:a[89]+a[90]+a[92]>=1;

s.t. c91:a[89]+a[91]+a[92]>=1;
s.t. c92:a[89]+a[90]+a[91]+a[92]+a[93]+a[94]+a[66]+a[95]>=1;
s.t. c93:a[92]+a[93]>=1;
s.t. c94:a[92]+a[94]+a[95]+a[96]>=1;
s.t. c95:a[94]+a[95]+a[97]+a[98]+a[92]+a[66]>=1;
s.t. c96:a[94]+a[96]+a[97]+a[99]>=1;
s.t. c97:a[95]+a[96]+a[97]>=1;
s.t. c98:a[95]+a[98]>=1;
s.t. c99:a[96]+a[99]+a[100]+a[101]>=1;
s.t. c100:a[70]+a[99]+a[100]+a[102]>=1;

s.t. c101:a[99]+a[101]+a[102]+a[103]>=1;
s.t. c102:a[100]+a[101]+a[102]+a[103]+a[77]>=1;
s.t. c103:a[101]+a[102]+a[103]+a[104]>=1;
s.t. c104:a[103]+a[104]+a[105]>=1;
s.t. c105:a[104]+a[105]+a[83]>=1;

#Andhra Pradesh

s.t. c106:a[106]+a[107]>=1;
s.t. c107:a[106]+a[107]+a[108]+a[111]+a[140]+a[141]>=1;
s.t. c108:a[107]+a[108]+a[109]>=1;
s.t. c109:a[108]+a[109]+a[110]>=1;
s.t. c110:a[109]+a[110]+a[113]+a[131]+a[132]+a[133]+a[142]+a[168]>=1;

s.t. c111:a[107]+a[111]+a[112]>=1;
```

```
s.t. c112:a[111]+a[112]>=1;
s.t. c113:a[110]+a[113]+a[114]+a[115]+a[131]+a[171]>=1;
s.t. c114:a[113]+a[114]+a[115]+a[127]+a[143]+a[172]>=1;
s.t. c115:a[113]+a[114]+a[115]+a[116]+a[117]+a[127]+a[172]>=1;
s.t. c116:a[115]+a[116]>=1;
s.t. c117:a[115]+a[117]+a[118]+a[128]+a[169]>=1;
s.t. c118:a[117]+a[118]+a[119]+a[120]+a[170]>=1;

s.t. c119:a[118]+a[119]+a[120]+a[122]+a[125]+a[126]+a[127]>=1;
s.t. c120:a[118]+a[119]+a[120]+a[121]>=1;
s.t. c121:a[120]+a[121]>=1;
s.t. c122:a[119]+a[122]+a[123]+a[124]>=1;
s.t. c123:a[122]+a[123]>=1;

s.t. c124:a[122]+a[124]>=1;
s.t. c125:a[119]+a[125]+a[128]>=1;
s.t. c126:a[119]+a[126]+a[127]>=1;
s.t. c127:a[114]+a[115]+a[119]+a[126]+a[127]+a[130]+a[131]+a[139]+a[143]+a[171]>=1;
s.t. c128:a[117]+a[125]+a[128]+a[119]>=1;
s.t. c129:a[128]+a[129]+a[130]>=1;
s.t. c130:a[127]+a[129]+a[130]>=1;
s.t. c131:a[110]+a[113]+a[127]+a[131]+a[132]>=1;
s.t. c132:a[110]+a[131]+a[132]+a[133]+a[136]+a[137]+a[139]+a[174]+a[175]>=1;
s.t. c133:a[110]+a[132]+a[133]+a[135]+a[140]+a[150]+a[172]>=1;
s.t. c134:a[134]+a[181]>=1; # not in AP Grid
s.t. c135:a[133]+a[135]+a[136]>=1;
s.t. c136:a[135]+a[136]+a[132]+a[137]+a[155]+a[156]+a[167]+a[173]+a[188]>=1;

s.t. c137:a[132]+a[136]+a[137]+a[138]+a[139]+a[144]+a[146]+a[155]>=1;
s.t.
c138:a[137]+a[138]+a[33]++a[163]+a[181]+a[191]+a[194]+a[199]+a[201]+a[206]+a[207]>=1;
s.t. c139:a[127]+a[132]+a[137]+a[139]+a[143]+a[153]+a[154]>=1;
s.t. c140:a[107]+a[133]+a[140]>=1;
s.t. c141:a[107]+a[141]>=1;
s.t. c142:a[110]+a[142]+a[143]+a[144]+a[149]>=1;
s.t. c143:a[142]+a[143]+a[127]+a[114]+a[139]>=1;
s.t. c144:a[132]+a[137]+a[142]+a[144]+a[145]>=1;
s.t. c145:a[144]+a[145]+a[146]+a[148]+a[149]+a[150]>=1;
s.t. c146:a[137]+a[145]+a[146]+a[147]>=1;
s.t. c147:a[146]+a[147]>=1;
s.t. c148:a[145]+a[148]>=1;
s.t. c149:a[142]+a[145]+a[149]+a[150]>=1;

s.t. c150:a[145]+a[149]+a[150]+a[151]+a[152]+a[153]+a[181]>=1;
s.t. c151:a[150]+a[151]+a[152]>=1;
s.t. c152:a[150]+a[151]+a[152]+a[153]+a[180]>=1;
```

```
s.t. c153:a[139]+a[150]+a[152]+a[153]+a[154]>=1;
s.t. c154:a[139]+a[153]+a[154]+a[179]>=1;
s.t. c155:a[136]+a[137]+a[155]+a[156]+a[157]+a[188]>=1;
s.t. c156:a[136]+a[155]+a[156]+a[163]+a[164]>=1;
s.t. c157:a[155]+a[157]+a[158]+a[159]+a[177]+a[178]+a[186]>=1;

s.t. c158:a[157]+a[158]+a[159]+a[178]>=1;
s.t. c159:a[157]+a[158]+a[159]+a[160]+a[161]+a[176]>=1;

s.t. c160:a[159]+a[160]>=1;
s.t. c161:a[159]+a[161]+a[162]+a[163]>=1;
s.t. c162:a[161]+a[162]+a[163]>=1;
s.t. c163:a[138]+a[156]+a[173]+a[161]+a[162]+a[163]>=1;
s.t. c164:a[156]+a[164]+a[165]>=1;
s.t. c165:a[164]+a[165]+a[166]+a[168]>=1;
s.t. c166:a[165]+a[166]>=1;
s.t. c167:a[136]+a[167]>=1;
s.t. c168:a[110]+a[165]+a[168]>=1;
s.t. c169:a[117]+a[169]>=1;
s.t. c170:a[118]+a[170]>=1;

s.t. c171:a[113]+a[127]+a[171]>=1;
s.t. c172:a[114]+a[115]+a[133]+a[172]>=1;
s.t. c173:a[136]+a[163]+a[173]>=1;
s.t. c174:a[132]+a[174]>=1;
s.t. c175:a[132]+a[175]>=1;

s.t. c176:a[159]+a[176]+a[177]>=1;
s.t. c177:a[157]+a[176]+a[177]>=1;
s.t. c178:a[157]+a[158]+a[178]+a[183]>=1;
s.t. c179:a[154]+a[179]>=1;
s.t. c180:a[152]+a[180]>=1;
s.t. c181:a[134]+a[138]+a[150]+a[181]+a[208]>=1;

# Karnataka
s.t. c182:a[127]+a[129]+a[130]>=1;
s.t. c183:a[178]+a[183]+a[184]+a[185]>=1;
s.t. c184:a[183]+a[184]+a[185]>=1;
s.t. c185:a[159]+a[183]+a[184]+a[185]+a[186]>=1;
s.t. c186:a[157]+a[185]+a[186]+a[188]+a[191]>=1;
s.t. c187:a[187]+a[188]+a[191]+a[195]>=1;
s.t. c188:a[136]+a[155]+a[159]+a[186]+a[187]+a[188]+a[189]+a[193]+a[194]>=1;
s.t. c189:a[188]+a[189]+a[190]>=1;
s.t. c190:a[189]+a[190]+a[203]>=1;

s.t. c191:a[138]+a[186]+a[187]+a[191]+a[192]>=1;
```

```
s.t. c192:a[191]+a[192]>=1;
s.t. c193:a[188]+a[193]+a[194]>=1;
s.t. c194:a[138]+a[187]+a[188]+a[193]+a[194]+a[195]+a[199]+a[200]>=1;
s.t. c195:a[194]+a[195]+a[196]>=1;
s.t. c196:a[195]+a[196]+a[197]+a[198]>=1;
s.t. c197:a[196]+a[197]>=1;
s.t. c198:a[196]+a[198]>=1;
s.t. c199:a[138]+a[194]+a[199]>=1;
s.t. c200:a[194]+a[200]+a[201]>=1;
s.t. c201:a[200]+a[201]+a[202]+a[205]>=1;

s.t. c202:a[201]+a[202]>=1;
s.t. c203:a[190]+a[203]+a[204]>=1;
s.t. c204:a[138]+a[203]+a[204]>=1;
s.t. c205:a[201]+a[205]+a[206]>=1;
s.t. c206:a[138]+a[205]+a[206]>=1;
s.t. c207:a[138]+a[207]>=1;
s.t. c208:a[181]+a[208]>=1;


option solver gurobi;
solve;
display a,z;
printf "CompletionTime %s.\n", ctime();
display _total_solve_time;
```

APPENDIX F

AMPL CODE FOR IEEE 300 BUS SYSTEM

```
var a{1..300} binary; #1
#number of buses in the system
minimize z:sum{i in 1..300}(a[i]);
# objective function of the problem. Minimizing the number of PMUs required.

s.t. c1:a[1]+a[3]+a[5]+a[251]>=1;
s.t. c2:a[2]+a[3]+a[6]+a[8]+a[252]>=1;
s.t. c3:a[1]+a[2]+a[3]+a[4]+a[7]+a[19]+a[150]+a[253]>=1;
s.t. c4:a[3]+a[4]+a[16]>=1;
s.t. c5:a[1]+a[5]+a[7]+a[9]>=1;
s.t. c6:a[2]+a[6]+a[7]>=1;
s.t. c7:a[3]+a[5]+a[6]+a[7]+a[12]+a[151]>=1;
s.t. c8:a[2]+a[8]+a[11]+a[14]>=1;
s.t. c9:a[5]+a[9]+a[11]>=1;
s.t. c10:a[10]+a[11]+a[12]>=1;

s.t. c11:a[8]+a[9]+a[10]+a[11]+a[13]+a[254]>=1;
s.t. c12:a[7]+a[10]+a[12]+a[21]+a[255]>=1;
s.t. c13:a[11]+a[13]+a[20]>=1;
s.t. c14:a[8]+a[14]+a[15]>=1;
s.t. c15:a[14]+a[15]+a[16]+a[17]+a[37]+a[89]+a[90]>=1;
s.t. c16:a[4]+a[15]+a[16]+a[42]>=1;
s.t. c17:a[15]+a[17]+a[256]>=1;
s.t. c18:a[24]+a[18]>=1;
s.t. c19:a[3]+a[19]+a[21]+a[87]>=1;
s.t. c20:a[13]+a[20]+a[21]+a[22]+a[27]>=1;

s.t. c21:a[12]+a[19]+a[20]+a[21]+a[24]>=1;
s.t. c22:a[20]+a[22]+a[23]>=1;
s.t. c23:a[22]+a[23]+a[24]+a[25]+a[257]>=1;
s.t. c24:a[21]+a[23]+a[24]+a[18]+a[258]>=1;
s.t. c25:a[23]+a[25]+a[26]>=1;
s.t. c26:a[25]+a[26]+a[27]+a[28]>=1;
s.t. c27:a[20]+a[26]+a[27]>=1;
s.t. c28:a[26]+a[28]>=1;
s.t. c29:a[104]+a[29]>=1;
s.t. c30:a[86]+a[30]>=1;

s.t. c31:a[31]+a[108]>=1;
s.t. c32:a[32]+a[63]>=1;
s.t. c33:a[33]+a[34]+a[38]+a[40]+a[41]>=1;
s.t. c34:a[33]+a[34]+a[42]>=1;
```

s.t. c35:a[35]+a[36]+a[72]+a[76]+a[77]>=1;
s.t. c36:a[35]+a[36]+a[88]>=1;
s.t. c37:a[15]+a[37]+a[38]+a[40]+a[41]+a[49]+a[89]+a[90]+a[265]>=1;
s.t. c38:a[33]+a[37]+a[38]+a[41]+a[43]>=1;
s.t. c39:a[39]+a[42]+a[259]>=1;
s.t. c40:a[33]+a[40]+a[58]+a[37]>=1;

s.t. c41:a[33]+a[37]+a[38]+a[41]+a[42]+a[49]+a[51]>=1;
s.t. c42:a[16]+a[34]+a[39]+a[41]+a[42]+a[46]>=1;
s.t. c43:a[38]+a[43]+a[44]+a[48]+a[53]>=1;
s.t. c44:a[43]+a[44]+a[45]+a[47]+a[54]+a[260]>=1;
s.t. c45:a[44]+a[45]+a[46]+a[60]+a[74]>=1;
s.t. c46:a[42]+a[45]+a[46]+a[81]>=1;
s.t. c47:a[44]+a[47]+a[73]+a[113]>=1;
s.t. c48:a[40]+a[43]+a[48]+a[103]>=1;
s.t. c49:a[37]+a[41]+a[49]+a[51]+a[261]>=1;
s.t. c50:a[50]+a[70]>=1;

s.t. c51:a[41]+a[49]+a[51]+a[52]>=1;
s.t. c52:a[51]+a[52]+a[55]>=1;
s.t. c53:a[43]+a[53]+a[54]>=1;
s.t. c54:a[44]+a[53]+a[54]+a[55]>=1;
s.t. c55:a[52]+a[54]+a[55]+a[57]+a[262]>=1;
s.t. c56:a[56]+a[72]>=1;
s.t. c57:a[55]+a[57]+a[58]+a[63]+a[263]>=1;
s.t. c58:a[57]+a[58]+a[59]>=1;
s.t. c59:a[58]+a[59]+a[61]>=1;
s.t. c60:a[45]+a[60]+a[62]>=1;

s.t. c61:a[59]+a[61]+a[62]+a[264]>=1;
s.t. c62:a[60]+a[61]+a[62]+a[64]+a[144]+a[95]>=1;
s.t. c63:a[57]+a[63]+a[64]+a[32]>=1;
s.t. c64:a[62]+a[63]+a[64]>=1;
s.t. c65:a[65]+a[77]>=1;
s.t. c66:a[66]+a[74]>=1;
s.t. c67:a[67]+a[77]>=1;
s.t. c68:a[68]+a[194]>=1;
s.t. c69:a[69]+a[79]+a[201]+a[211]>=1;
s.t. c70:a[70]+a[71]+a[50]>=1;

s.t. c71:a[70]+a[71]+a[72]+a[73]+a[96]>=1;
s.t. c72:a[35]+a[71]+a[72]+a[77]+a[56]>=1;
s.t. c73:a[47]+a[73]+a[71]+a[74]+a[76]+a[79]>=1;
s.t. c74:a[45]+a[73]+a[74]+a[66]+a[88]>=1;
s.t. c75:a[75]+a[119]>=1;
s.t. c76:a[35]+a[73]+a[76]+a[77]>=1;

s.t. c77:a[35]+a[72]+a[76]+a[77]+a[78]+a[80]+a[65]+a[67]>=1;
s.t. c78:a[77]+a[78]+a[79]+a[84]>=1;
s.t. c79:a[69]+a[73]+a[78]+a[79]+a[211]>=1;
s.t. c80:a[77]+a[80]+a[211]>=1;

s.t. c81:a[46]+a[81]+a[88]+a[194]+a[195]>=1;
s.t. c82:a[82]+a[120]>=1;
s.t. c83:a[83]+a[118]+a[120]>=1;
s.t. c84:a[83]+a[84]+a[85]>=1;
s.t. c85:a[78]+a[84]>=1;
s.t. c86:a[85]+a[86]+a[87]++a[102]+a[30]>=1;
s.t. c87:a[19]+a[86]+a[87]+a[94]>=1;
s.t. c88:a[15]+a[36]+a[89]+a[91]>=1;
s.t. c89:a[15]+a[37]+a[89]+a[91]>=1;
s.t. c90:a[15]+a[90]+a[37]+a[92]>=1;

s.t. c91:a[89]+a[91]+a[94]+a[97]>=1;
s.t. c92:a[90]+a[92]+a[103]+a[105]>=1;
s.t. c93:a[92]+a[196]+a[204]>=1;
s.t. c94:a[87]+a[94]+a[91]+a[97]>=1;
s.t. c95:a[62]+a[95]>=1;
s.t. c96:a[71]+a[96]>=1;
s.t. c97:a[91]+a[94]+a[97]+a[100]+a[102]+a[103]>=1;
s.t. c98:a[98]+a[100]+a[102]>=1;
s.t. c99:a[85]+a[99]+a[107]+a[108]+a[109]+a[110]>=1;
s.t. c100:a[95]+a[98]+a[100]+a[102]>=1;

s.t. c101:a[101]+a[130]>=1;
s.t. c102:a[86]+a[97]+a[98]+a[102]+a[104]+a[100]>=1;
s.t. c103:a[92]+a[97]+a[103]+a[105]>=1;
s.t. c104:a[102]+a[104]+a[108]+a[29]>=1;
s.t. c105:a[92]+a[103]+a[105]+a[107]+a[110]>=1;
s.t. c106:a[106]+a[139]>=1;
s.t. c107:a[48]+a[99]+a[105]+a[107]>=1;
s.t. c108:a[99]+a[104]+a[108]+a[31]>=1;
s.t. c109:a[99]+a[109]+a[110]+a[113]+a[114]>=1;
s.t. c110:a[99]+a[105]+a[109]+a[110]+a[112]>=1;

s.t. c111:a[111]+a[166]>=1;
s.t. c112:a[110]+a[112]+a[114]>=1;
s.t. c113:a[47]+a[113]+a[109]>=1;
s.t. c114:a[112]+a[114]+a[109]+a[207]>=1;
s.t. c115:a[115]+a[121]+a[122]>=1;
s.t. c116:a[116]+a[120]+a[124]>=1;
s.t. c117:a[117]+a[118]+a[159]>=1;
s.t. c118:a[117]+a[118]+a[119]+a[83]+a[121]>=1;

s.t. c119:a[118]+a[119]+a[120]+a[121]+a[75]>=1;

s.t. c120:a[116]+a[119]+a[120]+a[82]+a[83]>=1;
s.t. c121:a[115]+a[118]+a[119]+a[121]>=1;
s.t. c122:a[115]+a[122]+a[123]+a[125]+a[127]>=1;
s.t. c123:a[122]+a[123]+a[124]+a[125]>=1;
s.t. c124:a[116]+a[123]+a[124]+a[160]>=1;
s.t. c125:a[122]+a[123]+a[125]+a[126]>=1;
s.t. c126:a[125]+a[126]+a[127]+a[129]+a[132]+a[157]+a[158]+a[169]>=1;
s.t. c127:a[126]+a[127]+a[128]+a[134]+a[168]>=1;
s.t. c128:a[127]+a[128]+a[130]+a[133]>=1;
s.t. c129:a[126]+a[129]+a[130]+a[133]>=1;

s.t. c130:a[128]+a[129]+a[130]+a[131]+a[132]+a[150]+a[151]+a[167]+a[168]+a[101]>=1;
s.t. c131:a[7]+a[130]+a[131]>=1;
s.t. c132:a[126]+a[130]+a[132]+a[170]>=1;
s.t. c133:a[128]+a[129]+a[133]+a[137]+a[168]+a[169]+a[171]>=1;
s.t. c134:a[127]+a[134]+a[135]+a[184]>=1;
s.t. c135:a[134]+a[135]+a[136]>=1;
s.t. c136:a[135]+a[136]+a[137]+a[152]>=1;
s.t. c137:a[133]+a[136]+a[137]+a[140]+a[163]+a[181]+a[186]+a[188]>=1;
s.t. c138:a[138]+a[181]+a[188]>=1;
s.t. c139:a[106]+a[139]+a[172]+a[182]>=1;

s.t. c140:a[137]+a[140]+a[141]+a[145]+a[146]+a[147]+a[142]+a[182]>=1;
s.t. c141:a[140]+a[141]+a[146]+a[174]>=1;
s.t. c142:a[140]+a[142]+a[143]+a[175]>=1;
s.t. c143:a[142]+a[143]+a[144]+a[145]+a[148]+a[149]>=1;
s.t. c144:a[62]+a[143]+a[144]>=1;
s.t. c145:a[143]+a[145]+a[146]+a[140]+a[149]+a[180]>=1;
s.t. c146:a[140]+a[141]+a[145]+a[146]+a[147]>=1;
s.t. c147:a[140]+a[146]+a[147]>=1;
s.t. c148:a[143]+a[148]+a[178]+a[179]>=1;
s.t. c149:a[143]+a[145]+a[149]>=1;

s.t. c150:a[3]+a[130]+a[150]>=1;
s.t. c151:a[130]+a[151]+a[170]>=1;
s.t. c152:a[136]+a[152]+a[153]>=1;
s.t. c153:a[152]+a[153]+a[161]+a[183]>=1;
s.t. c154:a[154]+a[156]+a[183]>=1;
s.t. c155:a[155]+a[156]+a[161]+a[164]>=1;
s.t. c156:a[154]+a[155]+a[156]>=1;
s.t. c157:a[122]+a[126]+a[157]+a[159]>=1;
s.t. c158:a[126]+a[158]+a[159]+a[160]>=1;
s.t. c159:a[117]+a[157]+a[158]+a[159]>=1;

s.t. c160:a[124]+a[158]+a[160]>=1;
s.t. c161:a[153]+a[155]+a[161]>=1;
s.t. c162:a[162]+a[164]+a[165]>=1;
s.t. c163:a[137]+a[163]+a[164]>=1;
s.t. c164:a[155]+a[162]+a[163]+a[164]>=1;
s.t. c165:a[162]+a[165]+a[166]>=1;
s.t. c166:a[165]+a[166]+a[111]>=1;
s.t. c167:a[130]+a[167]+a[169]>=1;
s.t. c168:a[127]+a[130]+a[133]+a[168]>=1;
s.t. c169:a[126]+a[133]+a[167]+a[169]>=1;

s.t. c170:a[132]+a[151]+a[170]>=1;
s.t. c171:a[133]+a[171]>=1;
s.t. c172:a[139]+a[172]+a[173]+a[174]>=1;
s.t. c173:a[172]+a[173]+a[174]+a[175]+a[176]>=1;
s.t. c174:a[141]+a[172]+a[173]+a[174]>=1;
s.t. c175:a[173]+a[175]+a[176]+a[179]+a[142]>=1;
s.t. c176:a[173]+a[175]+a[176]+a[177]>=1;
s.t. c177:a[176]+a[177]+a[178]>=1;
s.t. c178:a[148]+a[177]+a[178]+a[179]+a[180]>=1;
s.t. c179:a[148]+a[178]+a[179]+a[175]>=1;

s.t. c180:a[145]+a[178]+a[180]>=1;
s.t. c181:a[137]+a[138]+a[181]+a[187]>=1;
s.t. c182:a[139]+a[140]+a[182]>=1;
s.t. c183:a[153]+a[154]+a[183]>=1;
s.t. c184:a[134]+a[184]+a[185]>=1;
s.t. c185:a[184]+a[185]>=1;
s.t. c186:a[137]+a[186]+a[188]>=1;
s.t. c187:a[181]+a[187]+a[188]>=1;
s.t. c188:a[137]+a[138]+a[186]+a[187]+a[188]>=1;
s.t. c189:a[189]+a[208]+a[209]+a[210]>=1;

s.t. c190:a[190]+a[229]+a[231]+a[240]>=1;
s.t. c191:a[191]+a[192]+a[225]>=1;
s.t. c192:a[191]+a[192]+a[225]>=1;
s.t. c193:a[193]+a[205]+a[208]+a[196]>=1;
s.t. c194:a[68]+a[81]+a[194]+a[219]>=1;
s.t. c195:a[81]+a[195]+a[212]+a[219]>=1;
s.t. c196:a[193]+a[196]+a[197]+a[210]+a[93]>=1;
s.t. c197:a[196]+a[197]+a[198]+a[211]>=1;
s.t. c198:a[197]+a[198]+a[202]+a[203]+a[209]+a[210]+a[211]>=1;
s.t. c199:a[199]+a[210]+a[200]>=1;

s.t. c200:a[199]+a[200]+a[210]+a[248]>=1;
s.t. c201:a[69]+a[201]+a[204]>=1;

s.t. c202:a[198]+a[202]+a[211]>=1;
s.t. c203:a[198]+a[203]+a[211]>=1;
s.t. c204:a[201]+a[204]+a[205]+a[93]>=1;
s.t. c205:a[204]+a[205]+a[206]+a[193]>=1;
s.t. c206:a[205]+a[206]+a[207]+a[208]>=1;
s.t. c207:a[114]+a[206]+a[207]>=1;
s.t. c208:a[189]+a[193]+a[206]+a[208]>=1;
s.t. c209:a[189]+a[198]+a[209]>=1;

s.t. c210:a[189]+a[196]+a[198]+a[199]+a[200]+a[210]>=1;
s.t. c211:a[69]+a[79]+a[80]+a[197]+a[198]+a[202]+a[203]+a[211]+a[212]>=1;
s.t. c212:a[195]+a[211]+a[212]+a[215]>=1;
s.t. c213:a[213]+a[214]>=1;
s.t. c214:a[213]+a[214]+a[215]+a[242]>=1;
s.t. c215:a[212]+a[214]+a[215]+a[216]>=1;
s.t. c216:a[215]+a[216]+a[217]>=1;
s.t. c217:a[216]+a[217]+a[218]+a[219]+a[220]>=1;
s.t. c218:a[217]+a[218]+a[219]+a[220]>=1;
s.t. c219:a[194]+a[195]+a[217]+a[218]+a[219]+a[237]>=1;

s.t. c220:a[217]+a[218]+a[220]+a[221]+a[238]>=1;
s.t. c221:a[220]+a[221]+a[223]>=1;
s.t. c222:a[222]+a[237]>=1;
s.t. c223:a[221]+a[223]+a[224]>=1;
s.t. c224:a[223]+a[224]+a[225]+a[226]>=1;
s.t. c225:a[191]+a[192]+a[224]+a[225]>=1;
s.t. c226:a[224]+a[226]+a[231]>=1;
s.t. c227:a[227]+a[231]>=1;
s.t. c228:a[228]+a[229]+a[231]+a[234]>=1;
s.t. c229:a[190]+a[228]+a[229]+a[230]>=1;

s.t. c230:a[229]+a[230]>=1;
s.t. c231:a[190]+a[226]+a[227]+a[228]+a[231]+a[232]+a[237]>=1;
s.t. c232:a[231]+a[232]+a[233]>=1;
s.t. c233:a[232]+a[233]>=1;
s.t. c234:a[228]+a[234]+a[235]+a[236]+a[237]>=1;
s.t. c235:a[234]+a[235]+a[238]>=1;
s.t. c236:a[234]+a[236]>=1;
s.t. c237:a[219]+a[222]+a[231]+a[234]+a[237]+a[241]>=1;
s.t. c238:a[220]+a[235]+a[238]+a[239]>=1;
s.t. c239:a[238]+a[239]>=1;

s.t. c240:a[190]+a[240]+a[281]>=1;
s.t. c241:a[237]+a[241]>=1;
s.t. c242:a[214]+a[242]+a[245]+a[247]>=1;
s.t. c243:a[243]+a[244]+a[245]>=1;

s.t. c244:a[243]+a[244]+a[246]>=1;
s.t. c245:a[242]+a[243]+a[245]+a[246]+a[247]>=1;
s.t. c246:a[244]+a[245]+a[246]+a[247]>=1;
s.t. c247:a[242]+a[245]+a[246]+a[247]+a[248]>=1;
s.t. c248:a[200]+a[247]+a[248]+a[249]>=1;
s.t. c249:a[248]+a[249]+a[250]>=1;

s.t. c250:a[249]+a[250]>=1;
s.t. c251:a[1]+a[251]>=1;
s.t. c252:a[2]+a[252]>=1;
s.t. c253:a[3]+a[253]>=1;
s.t. c254:a[11]+a[254]>=1;
s.t. c255:a[12]+a[255]>=1;
s.t. c256:a[17]+a[256]>=1;
s.t. c257:a[23]+a[257]>=1;
s.t. c258:a[258]+a[24]>=1;
s.t. c259:a[39]+a[259]>=1;

s.t. c260:a[44]+a[260]>=1;
s.t. c261:a[49]+a[261]>=1;
s.t. c262:a[55]+a[262]>=1;
s.t. c263:a[57]+a[263]>=1;
s.t. c264:a[61]+a[264]>=1;
s.t. c265:a[37]+a[265]+a[269]+a[270]+a[272]>=1;
s.t. c266:a[266]+a[272]+a[273]+a[276]>=1;
s.t. c267:a[267]+a[270]+a[271]+a[279]+a[280]+a[282]+a[283]+a[284]+a[285]+a[286]+a[287]+
a[291]>=1;
s.t. c268:a[268]+a[288]+a[289]+a[290]+a[291]>=1;
s.t. c269:a[265]+a[269]+a[292]+a[293]+a[294]+a[295]+a[296]>=1;

s.t. c270:a[270]+a[265]+a[267]+a[271]>=1;
s.t. c271:a[267]+a[270]+a[271]+a[297]+a[298]>=1;
s.t. c272:a[265]+a[266]+a[272]+a[299]>=1;
s.t. c273:a[266]+a[273]+a[274]+a[275]>=1;
s.t. c274:a[273]+a[274]>=1;
s.t. c275:a[273]+a[274]+a[277]+a[278]>=1;
s.t. c276:a[266]+a[276]>=1;
s.t. c277:a[275]+a[277]>=1;
s.t. c278:a[275]+a[278]>=1;
s.t. c279:a[267]+a[279]>=1;

s.t. c280:a[267]+a[280]>=1;
s.t. c281:a[240]+a[281]>=1;
s.t. c282:a[267]+a[282]>=1;
s.t. c283:a[267]+a[283]>=1;
s.t. c284:a[267]+a[284]>=1;

```
s.t. c285:a[267]+a[285]>=1;
s.t. c286:a[267]+a[286]>=1;
s.t. c287:a[267]+a[287]>=1;
s.t. c288:a[268]+a[288]>=1;
s.t. c289:a[268]+a[289]>=1;

s.t. c290:a[268]+a[290]>=1;
s.t. c291:a[267]+a[268]+a[291]>=1;
s.t. c292:a[269]+a[292]>=1;
s.t. c293:a[269]+a[293]>=1;
s.t. c294:a[269]+a[294]+a[300]>=1;
s.t. c295:a[269]+a[295]>=1;
s.t. c296:a[269]+a[296]>=1;
s.t. c297:a[271]+a[297]>=1;
s.t. c298:a[271]+a[298]>=1;
s.t. c299:a[272]+a[299]>=1;

s.t. c300:a[294]+a[300]>=1;

option solver cplex;
solve;
display a,z;
printf "CompletionTime %s.\n", ctime();
```

APPENDIX G

DETAILED PARAMETERS OF IEEE 14 BUS SYSTEM

Tables 17 and 18 show the detailed parameters of IEEE 14 bus system that have been referred while modeling the simulink version of the system. The Table 17 shows the bus data that includes final voltages and angles, generation and load limits including the active and reactive power limits etc…

While the Table 18 shows the branch parameters such as resistance, reactance, capacitance, transformer tap ratios etc…

Table 17 Bus Data of IEEE 14 Bus System

Bus Connections	Voltage Type	Type of Bus	Final Voltage	Final Angle	Load		Generation		Base K.V	Desired K.V	Max MVAR	Min MVAR
					MW	MVAR	MW	MVAR				
1 Bus 1	HV	3	1.060	0.0	0.0	0.0	232.4	-16.9	0.0	1.060	0.0	0.0
2 Bus 2	HV	2	1.045	-4.98	21.7	12.7	40.0	42.4	0.0	1.045	50.0	-40.0
3 Bus 3	HV	2	1.010	-12.72	94.2	19.0	0.0	23.4	0.0	1.010	40.0	0.0
4 Bus 4	HV	0	1.019	-10.33	47.8	-3.9	0.0	0.0	0.0	0.0	0.0	0.0
5 Bus 5	HV	0	1.020	-8.78	7.6	1.6	0.0	0.0	0.0	0.0	0.0	0.0
6 Bus 6	LV	2	1.070	-14.22	11.2	7.5	0.0	12.2	0.0	1.070	24.0	-6.0
7 Bus 7	ZV	0	1.062	-13.37	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
8 Bus 8	TV	2	1.090	-13.36	0.0	0.0	0.0	17.4	0.0	1.090	24.0	-6.0
9 Bus 9	LV	0	1.056	-14.94	29.5	16.6	0.0	0.0	0.0	0.0	0.0	0.0
10 Bus 10	LV	0	1.051	-15.10	9.0	5.8	0.0	0.0	0.0	0.0	0.0	0.0
11 Bus 11	LV	0	1.057	-14.79	3.5	1.8	0.0	0.0	0.0	0.0	0.0	0.0
12 Bus 12	LV	0	1.055	-15.07	6.1	1.6	0.0	0.0	0.0	0.0	0.0	0.0
13 Bus 13	LV	0	1.050	-15.16	13.5	5.8	0.0	0.0	0.0	0.0	0.0	0.0
14 Bus 14	LV	0	1.036	-16.04	14.9	5.0	0.0	0.0	0.0	0.0	0.0	0.0

Table 18 Branch Data of IEEE 14 Bus System

Branch Data	Resistance (R)	Reactance (X)	Line Charging (B)	Line 1 MVA	Line 2 MVA	Line 3 MVA	Transformer Ratio
1 -2	0.01938	0.05917	0.0528	0.0	0.0	0.0	0.0
1 – 5	0.05403	0.22304	0.0492	0.0	0.0	0.0	0.0
2 – 3	0.04699	0.19797	0.0438	0.0	0.0	0.0	0.0
2 – 4	0.05811	0.17632	0.034	0.0	0.0	0.0	0.0
2 – 5	0.05695	0.17388	0.0346	0.0	0.0	0.0	0.0
3 – 4	0.06701	0.17103	0.0128	0.0	0.0	0.0	0.0
4 – 5	0.01335	0.04211	0.0	0.0	0.0	0.0	0.0
4 – 7	0.0	0.20912	0.0	0.0	0.0	0.0	0.978
4 – 9	0.0	0.55618	0.0	0.0	0.0	0.0	0.969
5 – 6	0.0	0.25202	0.0	0.0	0.0	0.0	0.932
6 – 11	0.09498	0.19890	0.0	0.0	0.0	0.0	0.0
6 – 12	0.12291	0.25202	0.0	0.0	0.0	0.0	0.0
6 – 13	0.06615	0.13027	0.0	0.0	0.0	0.0	0.0
7 – 8	0.0	0.17615	0.0	0.0	0.0	0.0	0.0

Table 18 Cont

7 – 9	0.0	0.11001	0.0	0.0	0.0	0.0	0.0
9 – 10	0.09181	0.08450	0.0	0.0	0.0	0.0	0.0
9 – 14	0.12711	0.27038	0.0	0.0	0.0	0.0	0.0
10 – 11	0.08205	0.19207	0.0	0.0	0.0	0.0	0.0
12 – 13	0.22092	0.19988	0.0	0.0	0.0	0.0	0.0
13 – 14	0.17093	0.34802	0.0	0.0	0.0	0.0	0.0

APPENDIX H

MATLAB CODE FOR CIRCLE REPRESENTATION OF 14 BUSES

```
model = 'ieee14bus_phasor_modified';
load_system(model);
sim(model);

t1 = Vone.time;
t2 = Vone.time;
t3 = Vone.time;
t4 = Vone.time;
t5 = Vone.time;
t6 = Vone.time;
t7 = Vone.time;
t8 = Vone.time;
t9 = Vone.time;
t10 = Vone.time;
t11 = Vone.time;
t12 = Vone.time;
t13 = Vone.time;
t14 = Vone.time;

V1_one(:,1) = Vone.signals(1).values(:,1);
V2_one(:,1) = Vone.signals(1).values(:,2);
V3_one(:,1) = Vone.signals(1).values(:,3);
P1_one(:,1) = Vone.signals(2).values(:,1);
P2_one(:,1) = Vone.signals(2).values(:,2);
P3_one(:,1) = Vone.signals(2).values(:,3);

I1_one(:,1) = Ione.signals(1).values(:,1);
I2_one(:,1) = Ione.signals(1).values(:,2);
I3_one(:,1) = Ione.signals(1).values(:,3);
Ip1_one(:,1) = Ione.signals(2).values(:,1);
Ip2_one(:,1) = Ione.signals(2).values(:,2);
Ip3_one(:,1) = Ione.signals(2).values(:,3);

V1_two(:,1) = Vtwo.signals(1).values(:,1);
V2_two(:,1) = Vtwo.signals(1).values(:,2);
V3_two(:,1) = Vtwo.signals(1).values(:,3);
P1_two(:,1) = Vtwo.signals(2).values(:,1);
P2_two(:,1) = Vtwo.signals(2).values(:,2);
P3_two(:,1) = Vtwo.signals(2).values(:,3);
```

```
I1_two(:,1) = Itwo.signals(1).values(:,1);
I2_two(:,1) = Itwo.signals(1).values(:,2);
I3_two(:,1) = Itwo.signals(1).values(:,3);
Ip1_two(:,1) = Itwo.signals(2).values(:,1);
Ip2_two(:,1) = Itwo.signals(2).values(:,2);
Ip3_two(:,1) = Itwo.signals(2).values(:,3);

V1_three(:,1) = Vthree.signals(1).values(:,1);
V2_three(:,1) = Vthree.signals(1).values(:,2);
V3_three(:,1) = Vthree.signals(1).values(:,3);
P1_three(:,1) = Vthree.signals(2).values(:,1);
P2_three(:,1) = Vthree.signals(2).values(:,2);
P3_three(:,1) = Vthree.signals(2).values(:,3);

I1_three(:,1) = Ithree.signals(1).values(:,1);
I2_three(:,1) = Ithree.signals(1).values(:,2);
I3_three(:,1) = Ithree.signals(1).values(:,3);
Ip1_three(:,1) = Ithree.signals(2).values(:,1);
Ip2_three(:,1) = Ithree.signals(2).values(:,2);
Ip3_three(:,1) = Ithree.signals(2).values(:,3);

V1_four(:,1) = Vfour.signals(1).values(:,1);
V2_four(:,1) = Vfour.signals(1).values(:,2);
V3_four(:,1) = Vfour.signals(1).values(:,3);
P1_four(:,1) = Vfour.signals(2).values(:,1);
P2_four(:,1) = Vfour.signals(2).values(:,2);
P3_four(:,1) = Vfour.signals(2).values(:,3);

I1_four(:,1) = Ifour.signals(1).values(:,1);
I2_four(:,1) = Ifour.signals(1).values(:,2);
I3_four(:,1) = Ifour.signals(1).values(:,3);
Ip1_four(:,1) = Ifour.signals(2).values(:,1);
Ip2_four(:,1) = Ifour.signals(2).values(:,2);
Ip3_four(:,1) = Ifour.signals(2).values(:,3);

V1_five(:,1) = Vfive.signals(1).values(:,1);
V2_five(:,1) = Vfive.signals(1).values(:,2);
V3_five(:,1) = Vfive.signals(1).values(:,3);
P1_five(:,1) = Vfive.signals(2).values(:,1);
P2_five(:,1) = Vfive.signals(2).values(:,2);
P3_five(:,1) = Vfive.signals(2).values(:,3);

I1_five(:,1) = Ifive.signals(1).values(:,1);
```

```
I2_five(:,1) = Ifive.signals(1).values(:,2);
I3_five(:,1) = Ifive.signals(1).values(:,3);
Ip1_five(:,1) = Ifive.signals(2).values(:,1);
Ip2_five(:,1) = Ifive.signals(2).values(:,2);
Ip3_five(:,1) = Ifive.signals(2).values(:,3);

V1_six(:,1) = Vsix.signals(1).values(:,1);
V2_six(:,1) = Vsix.signals(1).values(:,2);
V3_six(:,1) = Vsix.signals(1).values(:,3);
P1_six(:,1) = Vsix.signals(2).values(:,1);
P2_six(:,1) = Vsix.signals(2).values(:,2);
P3_six(:,1) = Vsix.signals(2).values(:,3);

I1_six(:,1) = Isix.signals(1).values(:,1);
I2_six(:,1) = Isix.signals(1).values(:,2);
I3_six(:,1) = Isix.signals(1).values(:,3);
Ip1_six(:,1) = Isix.signals(2).values(:,1);
Ip2_six(:,1) = Isix.signals(2).values(:,2);
Ip3_six(:,1) = Isix.signals(2).values(:,3);

V1_seven(:,1) = Vseven.signals(1).values(:,1);
V2_seven(:,1) = Vseven.signals(1).values(:,2);
V3_seven(:,1) = Vseven.signals(1).values(:,3);
P1_seven(:,1) = Vseven.signals(2).values(:,1);
P2_seven(:,1) = Vseven.signals(2).values(:,2);
P3_seven(:,1) = Vseven.signals(2).values(:,3);

I1_seven(:,1) = Iseven.signals(1).values(:,1);
I2_seven(:,1) = Iseven.signals(1).values(:,2);
I3_seven(:,1) = Iseven.signals(1).values(:,3);
Ip1_seven(:,1) = Iseven.signals(2).values(:,1);
Ip2_seven(:,1) = Iseven.signals(2).values(:,2);
Ip3_seven(:,1) = Iseven.signals(2).values(:,3);

V1_eight(:,1) = Veight.signals(1).values(:,1);
V2_eight(:,1) = Veight.signals(1).values(:,2);
V3_eight(:,1) = Veight.signals(1).values(:,3);
P1_eight(:,1) = Veight.signals(2).values(:,1);
P2_eight(:,1) = Veight.signals(2).values(:,2);
P3_eight(:,1) = Veight.signals(2).values(:,3);

I1_eight(:,1) = Ieight.signals(1).values(:,1);
I2_eight(:,1) = Ieight.signals(1).values(:,2);
I3_eight(:,1) = Ieight.signals(1).values(:,3);
```

```
Ip1_eight(:,1) = Ieight.signals(2).values(:,1);
Ip2_eight(:,1) = Ieight.signals(2).values(:,2);
Ip3_eight(:,1) = Ieight.signals(2).values(:,3);

V1_nine(:,1) = Vnine.signals(1).values(:,1);
V2_nine(:,1) = Vnine.signals(1).values(:,2);
V3_nine(:,1) = Vnine.signals(1).values(:,3);
P1_nine(:,1) = Vnine.signals(2).values(:,1);
P2_nine(:,1) = Vnine.signals(2).values(:,2);
P3_nine(:,1) = Vnine.signals(2).values(:,3);

I1_nine(:,1) = Inine.signals(1).values(:,1);
I2_nine(:,1) = Inine.signals(1).values(:,2);
I3_nine(:,1) = Inine.signals(1).values(:,3);
Ip1_nine(:,1) = Inine.signals(2).values(:,1);
Ip2_nine(:,1) = Inine.signals(2).values(:,2);
Ip3_nine(:,1) = Inine.signals(2).values(:,3);

V1_ten(:,1) = Vten.signals(1).values(:,1);
V2_ten(:,1) = Vten.signals(1).values(:,2);
V3_ten(:,1) = Vten.signals(1).values(:,3);
P1_ten(:,1) = Vten.signals(2).values(:,1);
P2_ten(:,1) = Vten.signals(2).values(:,2);
P3_ten(:,1) = Vten.signals(2).values(:,3);

I1_ten(:,1) = Iten.signals(1).values(:,1);
I2_ten(:,1) = Iten.signals(1).values(:,2);
I3_ten(:,1) = Iten.signals(1).values(:,3);
Ip1_ten(:,1) = Iten.signals(2).values(:,1);
Ip2_ten(:,1) = Iten.signals(2).values(:,2);
Ip3_ten(:,1) = Iten.signals(2).values(:,3);



V1_eleven(:,1) = Veleven.signals(1).values(:,1);
V2_eleven(:,1) = Veleven.signals(1).values(:,2);
V3_eleven(:,1) = Veleven.signals(1).values(:,3);
P1_eleven(:,1) = Veleven.signals(2).values(:,1);
P2_eleven(:,1) = Veleven.signals(2).values(:,2);
P3_eleven(:,1) = Veleven.signals(2).values(:,3);

I1_eleven(:,1) = Ieleven.signals(1).values(:,1);
I2_eleven(:,1) = Ieleven.signals(1).values(:,2);
I3_eleven(:,1) = Ieleven.signals(1).values(:,3);
Ip1_eleven(:,1) = Ieleven.signals(2).values(:,1);
```

```
Ip2_eleven(:,1) = Ieleven.signals(2).values(:,2);
Ip3_eleven(:,1) = Ieleven.signals(2).values(:,3);

V1_twelve(:,1) = Vtwelve.signals(1).values(:,1);
V2_twelve(:,1) = Vtwelve.signals(1).values(:,2);
V3_twelve(:,1) = Vtwelve.signals(1).values(:,3);
P1_twelve(:,1) = Vtwelve.signals(2).values(:,1);
P2_twelve(:,1) = Vtwelve.signals(2).values(:,2);
P3_twelve(:,1) = Vtwelve.signals(2).values(:,3);

I1_twelve(:,1) = Itwelve.signals(1).values(:,1);
I2_twelve(:,1) = Itwelve.signals(1).values(:,2);
I3_twelve(:,1) = Itwelve.signals(1).values(:,3);
Ip1_twelve(:,1) = Itwelve.signals(2).values(:,1);
Ip2_twelve(:,1) = Itwelve.signals(2).values(:,2);
Ip3_twelve(:,1) = Itwelve.signals(2).values(:,3);

V1_thirteen(:,1) = Vthirteen.signals(1).values(:,1);
V2_thirteen(:,1) = Vthirteen.signals(1).values(:,2);
V3_thirteen(:,1) = Vthirteen.signals(1).values(:,3);
P1_thirteen(:,1) = Vthirteen.signals(2).values(:,1);
P2_thirteen(:,1) = Vthirteen.signals(2).values(:,2);
P3_thirteen(:,1) = Vthirteen.signals(2).values(:,3);

I1_thirteen(:,1) = Ithirteen.signals(1).values(:,1);
I2_thirteen(:,1) = Ithirteen.signals(1).values(:,2);
I3_thirteen(:,1) = Ithirteen.signals(1).values(:,3);
Ip1_thirteen(:,1) = Ithirteen.signals(2).values(:,1);
Ip2_thirteen(:,1) = Ithirteen.signals(2).values(:,2);
Ip3_thirteen(:,1) = Ithirteen.signals(2).values(:,3);



V1_fourteen(:,1) = Vfourteen.signals(1).values(:,1);
V2_fourteen(:,1) = Vfourteen.signals(1).values(:,2);
V3_fourteen(:,1) = Vfourteen.signals(1).values(:,3);
P1_fourteen(:,1) = Vfourteen.signals(2).values(:,1);
P2_fourteen(:,1) = Vfourteen.signals(2).values(:,2);
P3_fourteen(:,1) = Vfourteen.signals(2).values(:,3);

I1_fourteen(:,1) = Ifourteen.signals(1).values(:,1);
I2_fourteen(:,1) = Ifourteen.signals(1).values(:,2);
I3_fourteen(:,1) = Ifourteen.signals(1).values(:,3);
Ip1_fourteen(:,1) = Ifourteen.signals(2).values(:,1);
Ip2_fourteen(:,1) = Ifourteen.signals(2).values(:,2);
```

```
Ip3_fourteen(:,1) = Ifourteen.signals(2).values(:,3);


for i = 1:length(V1_one)
   V_1_one(i,1) = V1_one(i,1).*exp(1i * degtorad(P1_one(i,1)));
   V_2_one(i,1) = V2_one(i,1).*exp(1i * degtorad(P2_one(i,1)));
   V_3_one(i,1) = V3_one(i,1).*exp(1i * degtorad(P3_one(i,1)));

   I_1_one(i,1) = I1_one(i,1).*exp(1i * degtorad(Ip1_one(i,1)));
   I_2_one(i,1) = I2_one(i,1).*exp(1i * degtorad(Ip2_one(i,1)));
   I_3_one(i,1) = I3_one(i,1).*exp(1i * degtorad(Ip3_one(i,1)));
end

for i = 1:length(V1_two)
   V_1_two(i,1) = V1_two(i,1).*exp(1i * degtorad(P1_two(i,1)));
   V_2_two(i,1) = V2_two(i,1).*exp(1i * degtorad(P2_two(i,1)));
   V_3_two(i,1) = V3_two(i,1).*exp(1i * degtorad(P3_two(i,1)));

   I_1_two(i,1) = I1_one(i,1).*exp(1i * degtorad(Ip1_two(i,1)));
   I_2_two(i,1) = I2_one(i,1).*exp(1i * degtorad(Ip2_two(i,1)));
   I_3_two(i,1) = I3_one(i,1).*exp(1i * degtorad(Ip3_two(i,1)));
end

for i = 1:length(V1_three)
   V_1_three(i,1) = V1_three(i,1).*exp(1i * degtorad(P1_three(i,1)));
   V_2_three(i,1) = V2_three(i,1).*exp(1i * degtorad(P2_three(i,1)));
   V_3_three(i,1) = V3_three(i,1).*exp(1i * degtorad(P3_three(i,1)));

   I_1_three(i,1) = I1_one(i,1).*exp(1i * degtorad(Ip1_three(i,1)));
   I_2_three(i,1) = I2_one(i,1).*exp(1i * degtorad(Ip2_three(i,1)));
   I_3_three(i,1) = I3_one(i,1).*exp(1i * degtorad(Ip3_three(i,1)));
end

for i = 1:length(V1_four)
   V_1_four(i,1) = V1_four(i,1).*exp(1i * degtorad(P1_four(i,1)));
   V_2_four(i,1) = V2_four(i,1).*exp(1i * degtorad(P2_four(i,1)));
   V_3_four(i,1) = V3_four(i,1).*exp(1i * degtorad(P3_four(i,1)));

   I_1_four(i,1) = I1_four(i,1).*exp(1i * degtorad(Ip1_four(i,1)));
   I_2_four(i,1) = I2_four(i,1).*exp(1i * degtorad(Ip2_four(i,1)));
   I_3_four(i,1) = I3_four(i,1).*exp(1i * degtorad(Ip3_four(i,1)));
end
```

```
for i = 1:length(V1_five)
   V_1_five(i,1) = V1_five(i,1).*exp(1i * degtorad(P1_five(i,1)));
   V_2_five(i,1) = V2_five(i,1).*exp(1i * degtorad(P2_five(i,1)));
   V_3_five(i,1) = V3_five(i,1).*exp(1i * degtorad(P3_five(i,1)));

   I_1_five(i,1) = I1_five(i,1).*exp(1i * degtorad(Ip1_five(i,1)));
   I_2_five(i,1) = I2_five(i,1).*exp(1i * degtorad(Ip2_five(i,1)));
   I_3_five(i,1) = I3_five(i,1).*exp(1i * degtorad(Ip3_five(i,1)));
end

for i = 1:length(V1_six)
   V_1_six(i,1) = V1_six(i,1).*exp(1i * degtorad(P1_six(i,1)));
   V_2_six(i,1) = V2_six(i,1).*exp(1i * degtorad(P2_six(i,1)));
   V_3_six(i,1) = V3_six(i,1).*exp(1i * degtorad(P3_six(i,1)));

   I_1_six(i,1) = I1_six(i,1).*exp(1i * degtorad(Ip1_six(i,1)));
   I_2_six(i,1) = I2_six(i,1).*exp(1i * degtorad(Ip2_six(i,1)));
   I_3_six(i,1) = I3_six(i,1).*exp(1i * degtorad(Ip3_six(i,1)));
end

for i = 1:length(V1_seven)
   V_1_seven(i,1) = V1_seven(i,1).*exp(1i * degtorad(P1_seven(i,1)));
   V_2_seven(i,1) = V2_seven(i,1).*exp(1i * degtorad(P2_seven(i,1)));
   V_3_seven(i,1) = V3_seven(i,1).*exp(1i * degtorad(P3_seven(i,1)));

   I_1_seven(i,1) = I1_seven(i,1).*exp(1i * degtorad(Ip1_seven(i,1)));
   I_2_seven(i,1) = I2_seven(i,1).*exp(1i * degtorad(Ip2_seven(i,1)));
   I_3_seven(i,1) = I3_seven(i,1).*exp(1i * degtorad(Ip3_seven(i,1)));
end

for i = 1:length(V1_eight)
   V_1_eight(i,1) = V1_eight(i,1).*exp(1i * degtorad(P1_eight(i,1)));
   V_2_eight(i,1) = V2_eight(i,1).*exp(1i * degtorad(P2_eight(i,1)));
   V_3_eight(i,1) = V3_eight(i,1).*exp(1i * degtorad(P3_eight(i,1)));

   I_1_eight(i,1) = I1_eight(i,1).*exp(1i * degtorad(Ip1_eight(i,1)));
   I_2_eight(i,1) = I2_eight(i,1).*exp(1i * degtorad(Ip2_eight(i,1)));
   I_3_eight(i,1) = I3_eight(i,1).*exp(1i * degtorad(Ip3_eight(i,1)));
end

for i = 1:length(V1_nine)
   V_1_nine(i,1) = V1_nine(i,1).*exp(1i * degtorad(P1_nine(i,1)));
   V_2_nine(i,1) = V2_nine(i,1).*exp(1i * degtorad(P2_nine(i,1)));
   V_3_nine(i,1) = V3_nine(i,1).*exp(1i * degtorad(P3_nine(i,1)));
```

```
    I_1_nine(i,1) = I1_nine(i,1).*exp(1i * degtorad(Ip1_nine(i,1)));
    I_2_nine(i,1) = I2_nine(i,1).*exp(1i * degtorad(Ip2_nine(i,1)));
    I_3_nine(i,1) = I3_nine(i,1).*exp(1i * degtorad(Ip3_nine(i,1)));
end

for i = 1:length(V1_ten)
    V_1_ten(i,1) = V1_ten(i,1).*exp(1i * degtorad(P1_ten(i,1)));
    V_2_ten(i,1) = V2_ten(i,1).*exp(1i * degtorad(P2_ten(i,1)));
    V_3_ten(i,1) = V3_ten(i,1).*exp(1i * degtorad(P3_ten(i,1)));

    I_1_ten(i,1) = I1_ten(i,1).*exp(1i * degtorad(Ip1_ten(i,1)));
    I_2_ten(i,1) = I2_ten(i,1).*exp(1i * degtorad(Ip2_ten(i,1)));
    I_3_ten(i,1) = I3_ten(i,1).*exp(1i * degtorad(Ip3_ten(i,1)));
end

for i = 1:length(V1_eleven)
    V_1_eleven(i,1) = V1_eleven(i,1).*exp(1i * degtorad(P1_eleven(i,1)));
    V_2_eleven(i,1) = V2_eleven(i,1).*exp(1i * degtorad(P2_eleven(i,1)));
    V_3_eleven(i,1) = V3_eleven(i,1).*exp(1i * degtorad(P3_eleven(i,1)));

    I_1_eleven(i,1) = I1_eleven(i,1).*exp(1i * degtorad(Ip1_eleven(i,1)));
    I_2_eleven(i,1) = I2_eleven(i,1).*exp(1i * degtorad(Ip2_eleven(i,1)));
    I_3_eleven(i,1) = I3_eleven(i,1).*exp(1i * degtorad(Ip3_eleven(i,1)));
end

for i = 1:length(V1_twelve)
    V_1_twelve(i,1) = V1_twelve(i,1).*exp(1i * degtorad(P1_twelve(i,1)));
    V_2_twelve(i,1) = V2_twelve(i,1).*exp(1i * degtorad(P2_twelve(i,1)));
    V_3_twelve(i,1) = V3_twelve(i,1).*exp(1i * degtorad(P3_twelve(i,1)));

    I_1_twelve(i,1) = I1_twelve(i,1).*exp(1i * degtorad(Ip1_twelve(i,1)));
    I_2_twelve(i,1) = I2_twelve(i,1).*exp(1i * degtorad(Ip2_twelve(i,1)));
    I_3_twelve(i,1) = I3_twelve(i,1).*exp(1i * degtorad(Ip3_twelve(i,1)));
end

for i = 1:length(V1_thirteen)
    V_1_thirteen(i,1) = V1_thirteen(i,1).*exp(1i * degtorad(P1_thirteen(i,1)));
    V_2_thirteen(i,1) = V2_thirteen(i,1).*exp(1i * degtorad(P2_thirteen(i,1)));
    V_3_thirteen(i,1) = V3_thirteen(i,1).*exp(1i * degtorad(P3_thirteen(i,1)));

    I_1_thirteen(i,1) = I1_thirteen(i,1).*exp(1i * degtorad(Ip1_thirteen(i,1)));
    I_2_thirteen(i,1) = I2_thirteen(i,1).*exp(1i * degtorad(Ip2_thirteen(i,1)));
```

```
   I_3_thirteen(i,1) = I3_thirteen(i,1).*exp(1i * degtorad(Ip3_thirteen(i,1)));
end

for i = 1:length(V1_fourteen)
   V_1_fourteen(i,1) = V1_fourteen(i,1).*exp(1i * degtorad(P1_fourteen(i,1)));
   V_2_fourteen(i,1) = V2_fourteen(i,1).*exp(1i * degtorad(P2_fourteen(i,1)));
   V_3_fourteen(i,1) = V3_fourteen(i,1).*exp(1i * degtorad(P3_fourteen(i,1)));

   I_1_fourteen(i,1) = I1_fourteen(i,1).*exp(1i * degtorad(Ip1_fourteen(i,1)));
   I_2_fourteen(i,1) = I2_fourteen(i,1).*exp(1i * degtorad(Ip2_fourteen(i,1)));
   I_3_fourteen(i,1) = I3_fourteen(i,1).*exp(1i * degtorad(Ip3_fourteen(i,1)));
end

sz = get(0,'ScreenSize');
figure('Position',[1 1 sz(3) sz(4)],'name','Phasor for IEEE 14 Buses Voltages','NumberTitle','off');
bcount = 1;
time = 1;

 for j = 1:length(V1_one)
   z(bcount,:) = [V_1_one(j,1), V_2_one(j,1), V_3_one(j,1), I_1_one(j,1), I_2_one(j,1),
I_3_one(j,1)];
   z(bcount+1,:) = [V_1_two(j,1), V_2_two(j,1), V_3_two(j,1), I_1_two(j,1), I_2_two(j,1),
I_3_two(j,1)];
   z(bcount+2,:) = [V_1_three(j,1), V_2_three(j,1), V_3_three(j,1), I_1_three(j,1), I_2_three(j,1),
I_3_three(j,1)];
   z(bcount+3,:) = [V_1_four(j,1), V_2_four(j,1), V_3_four(j,1), I_1_four(j,1), I_2_four(j,1),
I_3_four(j,1)];
   z(bcount+4,:) = [V_1_five(j,1), V_2_five(j,1), V_3_five(j,1), I_1_five(j,1), I_2_five(j,1),
I_3_five(j,1)];
   z(bcount+5,:) = [V_1_six(j,1), V_2_six(j,1), V_3_six(j,1), I_1_six(j,1), I_2_six(j,1),
I_3_six(j,1)];
   z(bcount+6,:) = [V_1_seven(j,1), V_2_seven(j,1), V_3_seven(j,1), I_1_seven(j,1),
I_2_seven(j,1), I_3_seven(j,1)];
   z(bcount+7,:) = [V_1_eight(j,1), V_2_eight(j,1), V_3_eight(j,1), I_1_eight(j,1), I_2_eight(j,1),
I_3_eight(j,1)];
   z(bcount+8,:) = [V_1_nine(j,1), V_2_nine(j,1), V_3_nine(j,1), I_1_nine(j,1), I_2_nine(j,1),
I_3_nine(j,1)];
   z(bcount+9,:) = [V_1_ten(j,1), V_2_ten(j,1), V_3_ten(j,1), I_1_ten(j,1), I_2_ten(j,1),
I_3_ten(j,1)];
   z(bcount+10,:) = [V_1_eleven(j,1), V_2_eleven(j,1), V_3_eleven(j,1), I_1_eleven(j,1),
I_2_eleven(j,1), I_3_eleven(j,1)];
   z(bcount+11,:) = [V_1_twelve(j,1), V_2_twelve(j,1), V_3_twelve(j,1), I_1_twelve(j,1),
I_2_twelve(j,1), I_3_twelve(j,1)];
   z(bcount+12,:) = [V_1_thirteen(j,1), V_2_thirteen(j,1), V_3_thirteen(j,1), I_1_thirteen(j,1),
I_2_thirteen(j,1), I_3_thirteen(j,1)];
```

```
   z(bcount+13,:) = [V_1_fourteen(j,1), V_2_fourteen(j,1), V_3_fourteen(j,1), I_1_fourteen(j,1),
I_2_fourteen(j,1), I_3_fourteen(j,1)];

   string = ' and Voltages: ';
   count = 1;
   for i = 1:14
           axesHandles(i) = subplot(3,5,count);
           count = count+1;
           c = compass(z(i,1:3));
           bus = strcat('BUS-',num2str(i));
           bus = strcat(bus, ' time-');
           bus = strcat(bus, num2str(t1(time,1)));
           bus = strcat(bus,string);
           str1 = strcat(' V1 - ',num2str(V1_two(time,1)));
           str2 = strcat(' V2 - ',num2str(V2_two(time,1)));
           str3 = strcat(' V3 - ',num2str(V3_two(time,1)));
           bus = strvcat(bus,str1,str2,str3);
           title(bus);
           hold on
           for index = 1:length(c)
               red = [1 0 0];
               green = [0 1 0];
               blue = [0 0 1];
               cyan = [0 1 1];
               yellow = [1 1 0];
               pink = [1 0 1];
               colours = [red; green; blue; cyan; yellow; pink];
               set(c(index,:),'color',colours(index,:))
               set(c(index,:),'LineWidth',2)
           end
       hold off
       legend({'V1','V2','V3','I1','I2','I3'}, 'Position',[0.8 0.15 0.1 0.1], 'Units', 'normalized');
   end
   time = time + 1;
  pause(0.05);
 end



sz = get(0,'ScreenSize');
figure('Position',[1 1 sz(3) sz(4)],'name','Phasor for IEEE 14 Buses Currents','NumberTitle','off');
bcount = 1;
time = 1;


 for j = 1:length(V1_one)
```

```
   z(bcount,:) = [V_1_one(j,1), V_2_one(j,1), V_3_one(j,1), I_1_one(j,1), I_2_one(j,1),
I_3_one(j,1)];
   z(bcount+1,:) = [V_1_two(j,1), V_2_two(j,1), V_3_two(j,1), I_1_two(j,1), I_2_two(j,1),
I_3_two(j,1)];
   z(bcount+2,:) = [V_1_three(j,1), V_2_three(j,1), V_3_three(j,1), I_1_three(j,1), I_2_three(j,1),
I_3_three(j,1)];
   z(bcount+3,:) = [V_1_four(j,1), V_2_four(j,1), V_3_four(j,1), I_1_four(j,1), I_2_four(j,1),
I_3_four(j,1)];
   z(bcount+4,:) = [V_1_five(j,1), V_2_five(j,1), V_3_five(j,1), I_1_five(j,1), I_2_five(j,1),
I_3_five(j,1)];
   z(bcount+5,:) = [V_1_six(j,1), V_2_six(j,1), V_3_six(j,1), I_1_six(j,1), I_2_six(j,1),
I_3_six(j,1)];
   z(bcount+6,:) = [V_1_seven(j,1), V_2_seven(j,1), V_3_seven(j,1), I_1_seven(j,1),
I_2_seven(j,1), I_3_seven(j,1)];
   z(bcount+7,:) = [V_1_eight(j,1), V_2_eight(j,1), V_3_eight(j,1), I_1_eight(j,1), I_2_eight(j,1),
I_3_eight(j,1)];
   z(bcount+8,:) = [V_1_nine(j,1), V_2_nine(j,1), V_3_nine(j,1), I_1_nine(j,1), I_2_nine(j,1),
I_3_nine(j,1)];
   z(bcount+9,:) = [V_1_ten(j,1), V_2_ten(j,1), V_3_ten(j,1), I_1_ten(j,1), I_2_ten(j,1),
I_3_ten(j,1)];
   z(bcount+10,:) = [V_1_eleven(j,1), V_2_eleven(j,1), V_3_eleven(j,1), I_1_eleven(j,1),
I_2_eleven(j,1), I_3_eleven(j,1)];
   z(bcount+11,:) = [V_1_twelve(j,1), V_2_twelve(j,1), V_3_twelve(j,1), I_1_twelve(j,1),
I_2_twelve(j,1), I_3_twelve(j,1)];
   z(bcount+12,:) = [V_1_thirteen(j,1), V_2_thirteen(j,1), V_3_thirteen(j,1), I_1_thirteen(j,1),
I_2_thirteen(j,1), I_3_thirteen(j,1)];
   z(bcount+13,:) = [V_1_fourteen(j,1), V_2_fourteen(j,1), V_3_fourteen(j,1), I_1_fourteen(j,1),
I_2_fourteen(j,1), I_3_fourteen(j,1)];

   string = ' and Currents:';
   count = 1;
   for i = 1:14
        axesHandles(i) = subplot(3,5,count);
        count = count+1;
        c = compass(z(i,4:6));
        bus = strcat('BUS-',num2str(i));
        bus = strcat(bus, ' time-');
        bus = strcat(bus, num2str(t1(time,1)));
        bus = strcat(bus,string);
        str1 = strcat(' I1 - ',num2str(I1_two(time,1)));
        str2 = strcat(' I2 - ',num2str(I2_two(time,1)));
        str3 = strcat(' I3 - ',num2str(I3_two(time,1)));
        bus = strvcat(bus,str1,str2,str3);
        title(bus);
        hold on
        for index = 1:length(c)
```

```
            red = [1 0 0];
            green = [0 1 0];
            blue = [0 0 1];
            cyan = [0 1 1];
            yellow = [1 1 0];
            pink = [1 0 1];
            colours = [cyan; yellow; pink; red; green; blue];
            set(c(index,:),'color',colours(index,:))
            set(c(index,:),'LineWidth',2)
          end
        hold off
        legend({'I1','I2','I3','V1','V2','V3'}, 'Position',[0.8 0.15 0.1 0.1], 'Units', 'normalized');
    end
    time = time + 1;
%     pause(0.05);
 end
```

APPENDIX I

MATLAB CODE FOR DBSCAN ALGORITHM

```
% Function: [class,type]=dbscan(x,k,Eps)
% -------------------------------------------------------------------------
% Aim:
% Clustering the data with Density-Based Scan Algorithm with Noise (DBSCAN)
% -------------------------------------------------------------------------
% Input:
% x - data set (m,n); m-objects, n-variables
% k - number of objects in a neighborhood of an object
% (minimal number of objects considered as a cluster)
% Eps - neighborhood radius, if not known avoid this parameter or put []
% -------------------------------------------------------------------------
% Output:
% class - vector specifying assignment of the i-th object to certain
% cluster (m,1)
% type - vector specifying type of the i-th object
% (core: 1, border: 0, outlier: -1)

        function [class,type]=dbscan(x,k,Eps)
         [m,n]=size(x);
         if nargin<3 | isempty(Eps)
           [Eps]=epsilon(x,k);
        end
         x=[[1:m]' x];
        [m,n]=size(x);
        type=zeros(1,m);
        no=1;
        touched=zeros(m,1);
         for i=1:m
           if touched(i)==0;
              ob=x(i,:);
              D=dist(ob(2:n),x(:,2:n));
              ind=find(D<=Eps);
                if length(ind)>1 & length(ind)<k+1
                 type(i)=0;
                 class(i)=0;
              end
              if length(ind)==1
                 type(i)=-1;
                 class(i)=-1;
                 touched(i)=1;
              end
```

```
                if length(ind)>=k+1;
                  type(i)=1;
                  class(ind)=ones(length(ind),1)*max(no);
                  while ~isempty(ind)
                       ob=x(ind(1),:);
                       touched(ind(1))=1;
                       ind(1)=[];
                       D=dist(ob(2:n),x(:,2:n));
                       i1=find(D<=Eps);
                           if length(i1)>1
                         class(i1)=no;
                         if length(i1)>=k+1;
                            type(ob(1))=1;
                         else
                            type(ob(1))=0;
                         end
                          for i=1:length(i1)
                             if touched(i1(i))==0
                                touched(i1(i))=1;
                                ind=[ind i1(i)];
                                class(i1(i))=no;
                             end
                         end
                       end
                  end
                  no=no+1;
               end
            end
          end
           i1=find(class==0);
          class(i1)=-1;
          type(i1)=-1;
%..........................................
function [Eps]=epsilon(x,k)
% Function: [Eps]=epsilon(x,k)
%
% Aim:
% Analytical way of estimating neighborhood radius for DBSCAN
%
% Input:
% x - data matrix (m,n); m-objects, n-variables
% k - number of objects in a neighborhood of an object
% (minimal number of objects considered as a cluster)
  [m,n]=size(x);
Eps=((prod(max(x)-min(x))*k*gamma(.5*n+1))/(m*sqrt(pi.^n))).^(1/n);
%...........................................
```

```
function [D]=dist(i,x)
 % function: [D]=dist(i,x)
% Aim: Calculates the Euclidean distances between the i-th object and all objects in x
% Input:
% i - an object (1,n)
% x - data matrix (m,n); m-objects, n-variables
%
% Output:
% D - Euclidean distance (m,1)
 [m,n]=size(x);
D=sqrt(sum((((ones(m,1)*i)-x).^2)'));

if n==1
   D=abs((ones(m,1)*i-x))';
end
```

REFERENCES

[1] "How Smart Grid Works." [Online]. Available: http://www.nature.com/news/2008/080730/images/454570a-6.jpg. [Accessed: 01-Jan-2015].

[2] A. G. Phadke, "Synchronized phasor measurements in power systems," *IEEE Comput. Appl. Power*, vol. 6, no. 2, pp. 10–15, Apr. 1993.

[3] B. Mohammadi, A. Rabiei, and S. Mobayen, "Online inter-area oscillation monitoring in power systems using PMU data and Prony analysis," vol. 6, no. 22, pp. 5267–5272, 2011.

[4] K. S. Shim, S. T. Kim, J. H. Lee, E. J. Choi, and J. H. Choi, "Detection of low-frequency oscillation using synchrophasor in wide-area rolling blackouts," *Int. J. Electr. Power Energy Syst.*, vol. 63, pp. 1015–1022, Dec. 2014.

[5] K. P. Lien, C. W. Liu, C. S. Yu, and J. A. Jiang, "Transmission network fault location observability with minimal PMU placement," *IEEE Trans. Power Deliv.*, vol. 21, no. 3, pp. 1128–1136, 2006.

[6] A. Reddy, "PMU based Real-Time Short Term Voltage Stability Monitoring – Analysis and Implementation on a Real- Time Test Bed," 2007.

[7] L. Huang, Y. Sun, J. Xu, W. Gao, J. Zhang, and Z. Wu, "Optimal PMU Placement Considering Controlled Islanding of Power System," *IEEE Trans. Power Syst.*, vol. 29, no. 2, pp. 742–755, Mar. 2014.

[8] J. D. La Ree, S. Member, V. Centeno, J. S. Thorp, L. Fellow, and A. G. Phadke, "Synchronized Phasor Measurement Applications in Power Systems," vol. 1, no. 1, pp. 20–27, 2010.

[9] K. Martin and G. Brunello, "An overview of the IEEE Standard C37.118.2 — Synchrophasor Data Transfer for Power Systems," in *2014 IEEE PES General Meeting | Conference & Exposition*, pp. 1–1, 2014.

[10] "IEEE Guide for Phasor Data Concentrator Requirements for Power System Protection, Control, and Monitoring," 2013.

[11] E. O. Schweitzer and D. E. Whitehead, "Real-time power system control using synchrophasors," in *2008 61st Annual Conference for Protective Relay Engineers*, pp. 78–88, 2008.

[12] M. V Mynam, A. Harikrishna, and V. Singh, "Synchrophasors Redefining SCADA Systems," 2011.

[13] "SynchroPhasor Products." [Online]. Available: https://www.selinc.com/synchrophasors/products/.

[14] L. Mili, T. Baldwin, and R. Adapa, "Phasor measurement placement for voltage stability analysis of power systems," in *29th IEEE Conference on Decision and Control*, pp. 3033–3038 vol.6, 1990.

[15] T. L. Baldwin, L. Mili, M. B. Boisen, and R. Adapa, "Power system observability with minimal phasor measurement placement," *IEEE Trans. Power Syst.*, vol. 8, no. 2, pp. 707–715, May 1993.

[16] R. F. Nuqui and A. G. Phadke, "Phasor measurement unit placement techniques for complete and incomplete observability," *IEEE Trans. Power Deliv.*, vol. 20, no. 4, pp. 2381–2388, 2005.

[17] X. Bei, Y. J. Yoon, and A. Abur, "Optimal Placement and Utilization of Phasor Measurements for State Estimation," *PSERC*, 2005.

[18] B. Gou, "Generalized Integer Linear Programming Formulation for Optimal PMU Placement," *IEEE Trans. Power Syst.*, vol. 23, no. 3, pp. 1099–1104, Aug. 2008.

[19] B. Milosevic and M. Begovic, "Nondominated sorting genetic algorithm for optimal phasor measurement placement," *Power Syst. IEEE Trans.*, vol. 18, no. 1, pp. 69–75, 2003.

[20] F. Aminifar, C. Lucas, A. Khodaei, and M. Fotuhi-Firuzabad, "Optimal Placement of Phasor Measurement Units Using Immunity Genetic Algorithm," *IEEE Trans. Power Deliv.*, vol. 24, no. 3, pp. 1014–1020, Jul. 2009.

[21] M. Hajian, A. M. Ranjbar, T. Amraee, and A. R. Shirani, "Optimal Placement of Phasor Measurement Units: Particle Swarm Optimization Approach," *2007 Int. Conf. Intell. Syst. Appl. to Power Syst.*, no. 1, pp. 1–6, Nov. 2007.

[22] C. S. C. Su and Z. C. Z. Chen, "Optimal Placement of Phasor Measurement Units with New Considerations," *Power Energy Eng. Conf. (APPEEC), 2010 Asia-Pacific*, pp. 1–4, 2010.

[23] X. Tai, D. Marelli, E. Rohr, and M. Fu, "Optimal PMU placement for power system state estimation with random component outages," *Int. J. Electr. Power Energy Syst.*, vol. 51, pp. 35–42, Oct. 2013.

[24] S. Dambhare, D. Dua, R. Gajbhiye, and S. Soman, "Optimal Zero Injection considerations in PMU placement: An ILP approach," *16th PSCC, Glas. July 14*, pp. 14–19, 2008.

[25] F. Aminifar, M. Fotuhi-Firuzabad, M. Shahidehpour, and A. Khodaei, "Probabilistic multistage PMU placement in electric power systems," *IEEE Trans. Power Deliv.*, vol. 26, no. 2, pp. 841–849, 2011.

[26] D. Dua, S. Dambhare, R. K. Gajbhiye, and S. A. Soman, "Optimal multistage scheduling of PMU placement: An ILP approach," *IEEE Trans. Power Deliv.*, vol. 23, no. 4, pp. 1812–1820, 2008.

[27] V. A. Centeno, A. G. Phadke, D. Novosel, and H. A. R. Volskis, "A preprocessing method for effective PMU placement studies," in *2008 Third International Conference on Electric Utility Deregulation and Restructuring and Power Technologies*, pp. 2862–2867, 2008.

[28] M. Hurtgen and J. C. Maun, "Optimal PMU placement using Iterated Local Search," *Int. J. Electr. Power Energy Syst.*, vol. 32, no. 8, pp. 857–860, Oct. 2010.

[29] N. H. Abbasy and H. M. Ismail, "A unified approach for the optimal PMU location for power system state estimation," *IEEE Trans. Power Syst.*, vol. 24, no. 2, pp. 806–813, May 2009.

[30] S. Azizi, G. B. Gharehpetian, and A. S. Dobakhshari, "Optimal integration of phasor measurement units in power systems considering conventional measurements," *IEEE Trans. Smart Grid*, vol. 4, no. 2, pp. 1113–1121, 2013.

[31] Q. Li, S. Member, T. Cui, and Y. Weng, "An Information-Theoretic Approach to PMU Placement in Electric Power Systems," vol. 4, no. 1, pp. 446–456, 2013.

[32] G. N. Korres, P. S. Georgilakis, N. C. Koutsoukis, and N. M. Manousakis, "Numerical observability method for optimal phasor measurement units placement using recursive Tabu search method," *IET Gener. Transm. Distrib.*, vol. 7, no. 4, pp. 347–356, Apr. 2013.

[33] Y. Wang, C. Wang, W. Li, J. Li, and F. Lin, "Reliability-Based Incremental PMU Placement," *IEEE Trans. Power Syst.*, vol. 29, no. 6, pp. 2744–2752, 2014.

[34] A. Pal, G. Sanchez-Ayala, V. Centeno, and J. Thorp, "A PMU Placement Scheme Ensuring Real-Time Monitoring of Critical Buses of the Network," vol. 29, no. 2, pp. 510–517, 2014.

[35] "IEEE 14 bus system." [Online]. Available: http://www.ee.washington.edu/research/pstca/pf14/pg_tca14bus.htm. [Accessed: 08-Aug-2014].

[36] "OpenPDC." [Online]. Available: http://openpdc.codeplex.com/. [Accessed: 08-Aug-2014].

[37] "Documentation of openPDC." [Online]. Available: http://openpdc.codeplex.com/documentation. [Accessed: 15-Jul-2014].

[38] "Microsoft Developer Network." [Online]. Available: http://msdn.microsoft.com/en-us/library/.

[39] J. D. Weber and T. J. Overbye, "Voltage contours for power system visualization," *IEEE Trans. Power Syst.*, vol. 15, no. 1, pp. 404–409, 2000.

[40] T. J. Overbye, D. A. Wiegmann, and A. M. Rich, "Human factors aspects of power system voltage contour visualizations," *IEEE Trans. Power Syst.*, vol. 18, no. 1, pp. 76–82, Feb. 2003.

[41] C. Cortes, V. Vapnik, and L. Saitta, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.

[42] J. Stutz and W. Taylor, "L., J. H. Friedman, R. A. Olshen, and C. J. Stone. Classification and Regression Trees. Monterey, CA: Wads worth. Cheesman P., J. Kelly, M. Self, 1984"

[43] C. C. Aggarwal and C. K. Reddy, "Data Clustering: Algorithms and Applications," Aug. 2013.

[44] J. H. W. Jr., "Hierarchical Grouping to Optimize an Objective Function," Apr. 2012.

[45] M. M. Deza and E. Deza, *Encyclopedia of Distances*, vol. 2006. 2009.

[46] M. K. Jiawei Han, *Data Mining: Concepts and Techniques: Concepts and Techniques*, 3rd ed. Elsevier, 2011.

[47] J. Hartigan and M. Wong, "Algorithm AS 136: A k-means clustering algorithm," *Appl. Stat.*, vol. 28, no. 1, pp. 100–108, 1979.

[48] A. Chaturvedi, P. E. Green, and J. D. Caroll, "K-modes Clustering," *J. Classif.*, vol. 18, no. 1, pp. 35–55, Jul. 2014.

[49] R. Nock and F. Nielsen, "On weighting clustering.," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 8, pp. 1223–35, Aug. 2006.

[50] I. S. Dhillon, Y. Guan, and B. Kulis, “Kernel k-means,” in *Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining - KDD ’04*, p. 551, 2004.

[51] X. X. Martin Ester, Hans-peter Kriegel, Jörg S, “A density-based algorithm for discovering clusters in large spatial databases with noise,” *Kdd*, 1996.

[52] A. Hinneburg and H. Gabriel, “Denclue 2.0: Fast clustering based on kernel density estimation,” *Adv. Intell. Data Anal. VII*, 2007.

[53] M. M. B. H. K. J. S. Mihael Ankerst, “OPTICS: Ordering Points To Identify the Clustering Structure.”

[54] R. T. Ng, “CLARANS: a method for clustering objects for spatial data mining,” *IEEE Trans. Knowl. Data Eng.*, vol. 14, no. 5, pp. 1003–1016, Sep. 2002.

[55] T. Zhang, R. Ramakrishnan, and M. Livny, “BIRCH: An Efficient Data Clustering Databases Method for Very Large,” *ACM SIGMOD Int. Conf. Manag. Data*, vol. 1, pp. 103–114, 1996.

[56] S. Guha, R. Rastogi, and K. Shim, “CURE : An efficient clustering algorithm for large databases,” *SIGMOD ’98 Proceedings of the 1998 ACM SIGMOD international conference on Management of data*. pp. 73–84, 1998.

[57] E. Schikuta, “Grid-clustering: an efficient hierarchical clustering method for very large data sets,” in *Proceedings of 13th International Conference on Pattern Recognition*, vol. 2, pp. 101–105 vol.2, 1996.

[58] P. Gopakumar, G. S. Chandra, M. J. B. Reddy, and D. K. Mohanta, “Optimal redundant placement of PMUs in Indian power grid — northern, eastern and north-eastern regions,” *Front. Energy*, vol. 7, no. 3, pp. 413–428, May 2013.

[59] The University of Edinburgh, “IEEE 300 bus system.” [Online]. Available: http://www.maths.ed.ac.uk/optenergy/LocalOpt/300busnetwork_other.html. [Accessed: 01-Jan-2015].

[60] R. Fourer, D. M. Gay, M. Hill, B. W. Kernighan, and T. B. Laboratories, “AMPL : A Mathematical Programming Language,” *Manage. Sci.*, vol. 36, pp. 519–554, 1990.

[61] M. J. Alvarez, F. S. Sellschopp, and E. Vazquez, “A PMUs placement methodology based on inverse of connectivity and critical measurements,” *Int. J. Electr. Power Energy Syst.*, vol. 68, pp. 336–344, Jun. 2015.

[62] P. Gopakumar., G. Surya Chandra, and M. J. B. Reddy, “Optimal placement of phasor measurement units for Tamil Nadu state of Indian power grid,” *2012 11th Int. Conf. Environ. Electr. Eng.*, pp. 80–83, May 2012.

[63] B. K. Saha Roy, a. K. Sinha, and a. K. Pradhan, “An optimal PMU placement technique for power system observability,” *Int. J. Electr. Power Energy Syst.*, vol. 42, no. 1, pp. 71–77, Nov. 2012.

[64] M. Daszykowski, S. Serneels, K. Kaczmarek, P. Van Espen, C. Croux, and B. Walczak, “TOMCAT: A MATLAB toolbox for multivariate calibration techniques,” *Chemom. Intell. Lab. Syst.*, vol. 85, no. 2, pp. 269–277, Feb. 2007.

[65] R. Vallakati, A. Mukherjee, and P. Ranganathan, “Preserving Observability in Smart Grid using Phasor Measurement Unit by applying Optimal Redundancy Criteria (ORC),” in *2015 1st IEEE International Conference on DC Micro-Grids*, 2015.

[66] R. Vallakati, A. Mukherjee, and P. Ranganathan, “Situational Awareness Using DBSCAN in Smart-Grid,” *Smart Grid Renew. Energy*, vol. 06, no. 05, pp. 120–127, May 2015.

[67] A. Mukherjee, R. Vallakati, and P. Ranganathan, “Using Phasor data for Visualization and Data Mining Purposes in Power Systems,” in *2015 1st IEEE International Conference on DC Micro-Grids*, 2015.