1-1-2007

Intelligent placement of meters/sensors for shipboard power system analysis

Sandhya Sankar

INTELLIGENT PLACEMENT OF METERS / SENSORS

FOR SHIPBOARD POWER SYSTEM ANALYSIS

By

Sandhya Sankar

A Thesis
Submitted to the Faculty of
Mississippi State University
in Partial Fulfillment of the Requirements
for the Degree of Master of Science
in Electrical Engineering
in the Department of Electrical and Computer Engineering

Mississippi State, Mississippi

December 2007

INTELLIGENT PLACEMENT OF METERS / SENSORS

FOR SHIPBOARD POWER SYSTEM ANALYSIS

By

Sandhya Sankar

Approved:

Noel N. Schulz Associate Professor of Electrical and Computer Engineering (Major Advisor and Director of Thesis)	Stanislaw Grzybowski Professor of Electrical and Computer Engineering (Committee Member)
Herbert L. Ginn III Assistant Professor of Electrical and Computer Engineering (Committee Member)	Nicolas H. Younan Professor of Electrical and Computer Engineering (Graduate Program Director)
Roger L. King Associate Dean of College of Engineering	

Name: Sandhya Sankar

Date of Degree: August 23, 2007

Institution: Mississippi State University

Major Field: Electrical Engineering

Major Professor: Dr. Noel N. Schulz

Title of Study: INTELLIGENT PLACEMENT OF METERS / SENSORS FOR SHIPBOARD POWER SYSTEM ANALYSIS

Pages in Study: 152

Candidate for Degree of Master of Science

Unlike the terrestrial power system, the Shipboard power system (SPS) is a comparatively smaller system with more complexity in terms of its system operation. This requires the power system to be continuously monitored to detect any type of fluctuations or disturbances. Planning metering systems for the SPS is a challenging task not only due to the dimensionality of the problem, but also due to the need for reducing redundancy while improving network observability and efficient data collection for a reliable state estimation process. This research is geared towards the use of a Genetic Algorithm for intelligent placement of meters in the SPS for real time power system monitoring taking into account different system topologies and critical parameters to be measured from the system. The algorithm predicts the type and location of meters for identification and collection of measurements from the system. The algorithm has been tested with several system topologies.

Key words: Power system monitoring, shipboard power system, genetic algorithm, system observability, meter placement, state estimation.

DEDICATION

To my loving parents, Mrs. Lakshmi Sankar and Late Mr. S. Sankar, for their blessings, unfaltering support and encouragement.

ACKNOWLEDGEMENTS

I take this opportunity to sincerely express my gratitude towards my advisor, Dr. Noel N. Schulz for her invaluable guidance and support throughout my thesis work and during my Masters program at Mississippi State University. Also, I am grateful to my committee members Dr. Stanislaw Grzybowski, and Dr. Herbert Ginn III for their valuable suggestions and guidance.

I gratefully acknowledge the Office of Naval Research (ONR) for providing financial support during my graduate program.

I would like to thank my parents for providing me with the opportunity to be where I am. Without them, none of this would even be possible. I would also like to thank my husband, Jayasuryan Iyer, for his unparalleled support, motivation and invaluable advice, throughout my research. I would like to acknowledge my uncle, Dr. Narayanaswami for his guidance and advice.

Warm thanks to my colleagues and friends for all their encouragement.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

In recent years the evolution of the electric ship in the U.S Navy has increased the importance of electrical systems. The electric power system on the ship provides power to the entire ship, including the propulsion motors, power conversion and distribution networks, service loads as well as the sophisticated systems for weapons, communication, navigation and other operations of warships. As a result, modern naval vessels have developed a strong need for devices capable of monitoring numerous points in electrical power systems and on loads. The ideal tool for this task would be a system that can automate the analysis of meter data while minimizing the number of meters required. Continuous monitoring of Shipboard electric power systems (SPS) before, during or after battle can help predict the fitness of the electric system to continue its operation. It may also benefit in improving the performance in tasks related to protection, reconfiguration and improved survivability.

This chapter presents a general overview on power systems monitoring, shipboard power system, monitoring of SPS, and concludes with a brief description of the research accomplished and an outline of the contents in subsequent chapters.

1.1 Terrestrial Power System Monitoring

Power system substations today are operated and controlled from control centers. The operators in these control centers are required to maintain stable operation. This involves continuous monitoring of system conditions, identifying operating states, and determining and preventing sudden fluctuations and abnormal conditions.

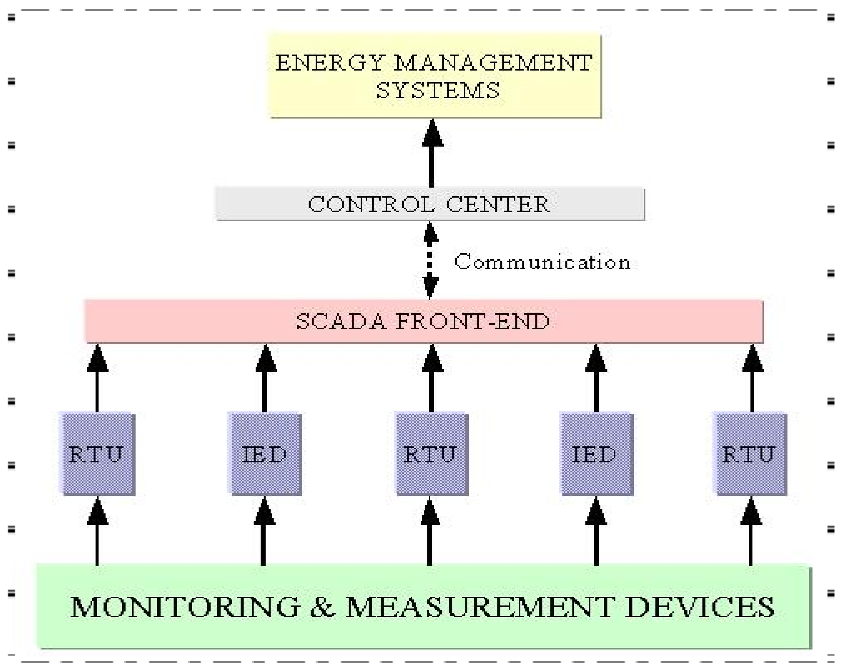

Figure 1.1 Typical Power System Monitoring [1]

The substations are equipped with remote terminal units (RTU) and intelligent electronic devices (IED) that collect all the required measurements (power load on the lines, voltage magnitudes, line current magnitudes, loads, generator output status of circuit breakers etc.) from measurement devices connected to the power system and transmit them to the control center. The information from the RTUs and IEDs is communicated with the help of a Supervisory Control and Data Acquisition (SCADA) front-end computer to the SCADA host computer at the control center [1]. The state estimator at the control center should be able to evaluate the states of the system. During the state estimation process statistically small metering errors are filtered out and bad data is correctly identified and removed. Also, as part of the state estimation process, system topology and observability are determined. During real-time power system monitoring, state estimation plays an important role in providing a complete and reliable database that can be used by security assessment programs in an Energy Management System. Data redundancy is crucial for the success of state estimation. Adequate redundancy levels enable state estimation to efficiently process bad data and also to achieve reliable estimates, even in case of temporary data loss. Data redundancy may be evaluated considering the number, type, and topological distribution of metering devices.

1.2 Shipboard Power System

1.2.1 General Overview

A shipboard power system consists of a scenario quite dissimilar to a terrestrial power system with respect to its configuration and load characteristics. Due to the size constraint of the ship, the size of equipment and the use of redundant equipment have to be reduced considerably. The generating capacity and rotational inertia on a SPS is much smaller relative to the system load than in terrestrial systems. The loads on the electric ship are classified as vital and non-vital loads, thereby prioritizing the supply requirement. Some of the loads like the propulsion motors draw on a significant fraction of the generating capacity. Also, loads may change rapidly and unpredictably during battle and emergency conditions, injecting harmonics and disturbances into the system and affecting its stability. Thus, the shipboard power system includes many application components for managing electrical networks on the ship. Some of these components are monitoring and control of equipment for power delivery, management processes to ensure system reliability, voltage management and outage management.

1.2.2 Shipboard Power System Monitoring

Real time monitoring of the shipboard power system is a complex task to address. Compared to the terrestrial power system, the shipboard power system is a smaller system but with more complexity in terms of its system operation. This requires the power system to be continuously monitored to detect any type of fluctuations or disturbances. For this purpose, meters have to be placed throughout the power system and at the right places to measure and transmit the values that will help estimate other parameters to determine if the system is completely observable or not. A system is said to be observable if, by knowing the external outputs (meter measurements), the internal state of the system can be predicted. Taking into account the need to reduce redundancy and redundant equipment, an optimal number of meters need to be placed considering all the constraints. Another factor to be considered for meter placement on the electric ship is that the number of meters required and the meter location needs to be determined such that even if part of the system is lost, information about most of the system should still be available. This would mean that redundancy would be allowed to a certain extent and the meters are placed in such a way that vital loads remain observable even if one or more meters close to them are lost.

1.3 Thesis Statement

Placement of adequate number of meters throughout the system is a prerequisite for power system monitoring. Although highly redundant metering systems would be advantageous for monitoring a power system, they are seldom installed in terrestrial power systems due to financial constraints [16]. In contrast, given that the size of the shipboard power systems compared to the terrestrial power system is quite small, placing a sufficient number of metering devices is not a major concern from a financial aspect.

During power system operation, topology changes or temporary malfunction of the data acquisition system may reduce data redundancy for state estimation. Planning metering systems in the power system of a ship is a challenging task, not only due to the dimensionality of the problem, but also due to the need for reducing redundancy while improving network observability and efficient data collection for reliable state estimation process.

For a system to be completely observable a certain minimum number of meters need to be placed for continuous monitoring. Efficient placement of the meters also improves the reliability of the state estimation process. For a ship system we need to take into account an important aspect, which is the vulnerability of the ship power system to damage resulting in loss of power to vital loads during battle. Therefore while keeping in mind the need to reduce redundancy, it is necessary that enough meters be placed such that the system remains observable even if a part of it loses communication or is damaged. This research is geared towards the use of a genetic algorithm for intelligent placement of meters in a shipboard system for real time power system monitoring taking into account different system topologies and critical parameters to be measured from the system.

1.4 Research Activities

As a system grows larger, the number of possible combinations of meters grows quickly. In order to determine the best combinations a method called Genetic Algorithms has been used. Genetic Algorithms provide a user defined fitness function for the given meter configuration that best fits the power system monitoring needs.

As a part of this thesis a mathematical framework is provided within which various constraints like minimum number of meters, types of meters, presence of critical measurements, contingency analyses has been considered to come up with an intelligent meter placement scheme which satisfies all these constraints. This research project demonstrates the possibility of two types of meters, power injection meters at the buses and power flow meters at the branches on a shipboard power system. These two types of meters were used to start with, because the equations for state-estimation for power measurements can be decoupled and are hence easier to solve [1]. This is not the case for voltage and current meters, making state estimation quite difficult. The algorithm has been successfully developed to perform 'N-2' contingency analysis on different meter arrangement schemes before selecting the optimal meter placement scheme from the solution space. This algorithm can be used for planning the meter placement scheme during design of the power system as well as to determine meter arrangement in the power system during ship power system operation for system observability and stability. The algorithm has also been tested successfully with three (large as well as small) test systems to demonstrate the outcome of meter placement. The results are promising for advanced design applications.

1.5 Thesis Organization

In this chapter a brief description of power system monitoring, shipboard system and its monitoring and an outline of the research accomplished are provided. The second chapter gives a detailed description of the research background and previous work done by other researchers in this field. Chapter III introduces the power system monitoring problems in the ship, offers the need for a solution and provides details about the toolbox used and the algorithm developed. Chapter IV discusses the test cases used to validate the algorithm and the working of the toolbox. Chapter V discusses the results. In the conclusion, Chapter VI cites benefits of the work performed and presented, and offers suggestions on the scope for improvement and areas of potential future work.

CHAPTER II

BACKGROUND AND PREVIOUS WORK

2.1 Introduction

Shipboard power system monitoring is a critical process, as protection and reconfiguration depend on it. In the SPS, many types of faults can occur due to equipment failure, over-voltages caused by switching surges or battle damage. A power system protection scheme is required to continuously monitor the system and detect any abnormal conditions or a fault as quickly as possible. The power system then removes a small part of the electric system to isolate the faulted part from the system so as to maintain power balance and system stability. The reconfiguration of the power system topology is performed so that energy is always available to maintain vital loads. This is done by altering the system topology by changing the status of devices like circuit breakers and switches. Protection and reconfiguration continuously require information about the state of the system and parameters like voltage, current and real and reactive power flow through the system. Thus, there is a need for an efficient metering scheme that will monitor the ship power system status and provide relevant data that can be used in state estimation to estimate the necessary information required to perform other operations.

While monitoring of the SPS forms an essential operation, the weight and space issues in the ship must also be considered. Figure 2.1 shows the presence of a meter on every bus (power injection meter - dark grey) and branch (power flow meter – light grey) in the 3-bus test system. This would result in not only too much redundant equipment, but also extra data to be processed. Instead, the presence of power meters M4 and M5 at branches and M1 and M2 at buses 1 and 2 can help estimate the power flow throughout the system at bus 3. Hence, the meters M3 at bus 3 and M6 on the branch shown in white are unnecessary. Including power meters M3 and M6 will not only increase the time required to process and store the data from all the meters but will also require more memory. They will also add to the weight of the ship and occupy space. This is an unacceptable luxury as the system built should be efficient and perform well during normal conditions or in fight or flight conditions during battle.

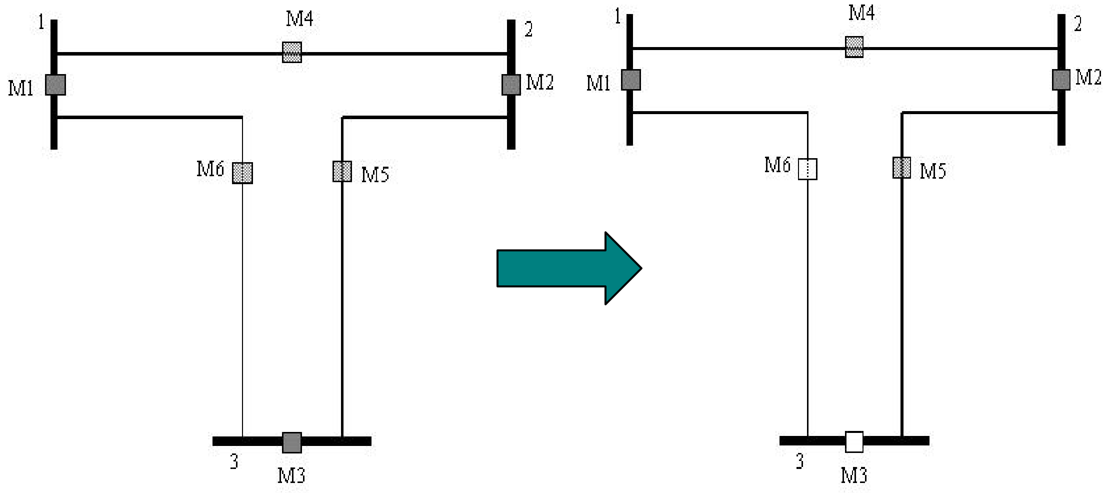

Figure 2.1   3-bus system with meter at buses and branches

Another aspect to be considered for meter placement on the electric ship is that the number of meters required and their location need to be determined such that even if part of the system is lost, information about most of the system should still be available. This would mean that redundancy would be allowed to a certain extent and that the meters are placed in a way that vital loads remain observable even if one or more meters in close proximity to them are lost.

2.2 System Observability

The meters in the ship power system are the ‘eyes’ for monitoring the power system. They obtain the best possible snapshot of the real-time status of the shipboard power system. System observability is maintained when the correct number and type of meters are placed in the right locations in the system topology. Power injection meters at the buses and the power flow meters on the branches are the two types of meters considered in this research.

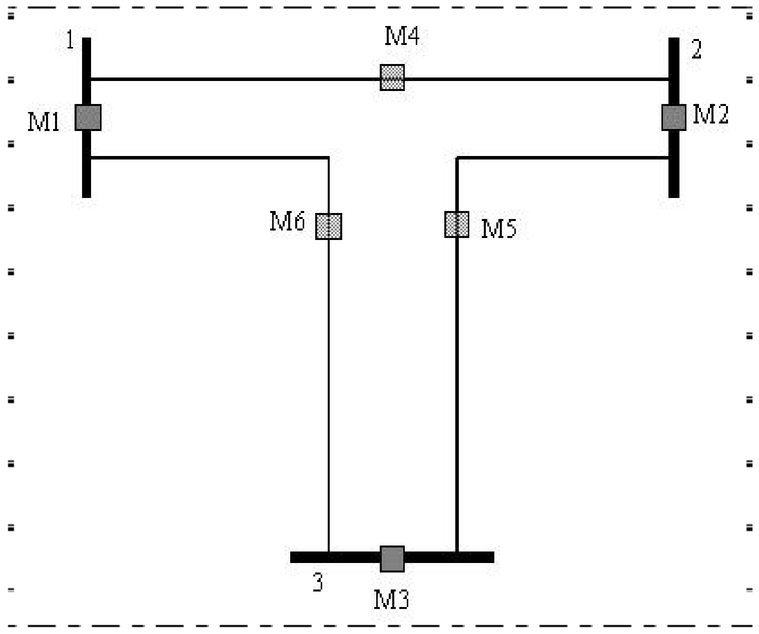

Figure 2.2 Test system with power injection meter at buses and power flow meter at branches

Consider the 3-bus, 3-branch system shown in Figure 2.2. Power injection meters are placed at all the buses and power flow meters are placed at all the branches. There are a total of 6 meters, 3 power injection meters and 3 power flow meters placed in the test system. This system will be observable as meters are placed at all possible locations. In order to find the minimum number of meters and their location to maintain observability, the bus incidence matrix and Jacobian matrix are built first. Then the states of the system are evaluated. The method to determine system observability and identify critical measurements is explained in detail in reference [1].

The following equations indicate the conditions for the system observability [1]:

For a system to be observable,

$$\text{if } H \times \theta_t = 0, \tag{2.1}$$

$$\text{then } I \times \theta_t = 0 \tag{2.2}$$

where H is the Jacobian matrix.

θ_t is the estimated state

I is the bus incidence matrix

$$\text{if } I \times \theta_t \neq 0, \tag{2.3}$$

then the system is said to be unobservable.

Critical measurements can be evaluated only if the system is observable.

2.3 Critical measurements

Measurements are said to be critical if the absence of these measurements causes the system to lose observability. Figure 2.2 shows power injection meters on each bus and power flow meters on each branch. From all these meters, a few meters are selected such that the system remains observable with the presence of the selected meters. The measurements from these meters are called critical measurements [1].

Work related to critical measurements is detailed in [1]. Peter-Wilkinson decomposition is applied to the Jacobian matrix (H) to determine the matrix L. The matrix L is then split into two matrices, L_1 and L_2, such that:

L_1 is a $n \times n$ lower triangular matrix,

L_2 is a $(m\text{-}n) \times n$ rectangular matrix.

where m represents the number of rows of the matrix and,

n represents the number of columns of the matrix.

Matrix C is then evaluated to identify the critical measurements.

$$C = L_2 \cdot L_1^{-1} \tag{2.4}$$

The column, which is null in matrix C, indicates that the measurement corresponding to the same row in H is critical.

2.4 Contingency Analysis

Contingency is an uncertainty or a problem that occurs in a system. Contingency analysis is a study that explores the effect of these uncertainties in the system. In load dispatch centers contingency analyses are performed to study and determine the variations and uncertainties that occur during power flow in the electric power network. Contingencies such as one or more line outages, branch outages, loss of a generator, or variation in loads are applied to the system. The effects are studied by running power flow analysis algorithms. This study provides knowledge to the staff in advance for them to take corrective measures during occurrence of overloads and violations in the future.

This analysis is also important in power system monitoring as loss of meters in the system can have an adverse effect on devices and processes dependent on data from these meters. The analysis helps to estimate which combination of meter location and type gives the least number of unobservable buses and branches.

2.5 Brief Overview of Genetics

The Genetic Algorithm has been developed based on the principle of genetics in various organisms. Genetics in biological organisms involves variation and inheritance of characteristic traits by offspring from its parent. Every organism develops depending on certain set of rules or scheme describing how that organism is built up from the tiny building blocks of life [23]. These rules are encoded in the *genes* of an organism, which in turn are connected together into long strings called *chromosome* [2, 17]. Each gene represents a specific trait of the organism, like skin color or eye color, and has several different settings. For example, the characteristic for an eye color gene may be black, brown, blue or green. These genes and their characteristics are usually referred to as an organism's *genotype* [2, 17].

When two organisms mate they share their genes. The resultant offspring will end up having a part of the genes from one parent and the rest from the other. This process is called recombination [23]. Sometimes a gene may undergo mutation whereby some traits or characteristics in the gene of the offspring maybe altered. Normally this mutated gene will not affect the development of the organism but very occasionally it will be expressed in the organism as a completely new trait [23].

2.5.1 Introduction to Genetic Algorithms

A *Genetic Algorithm (GA)* is a method of solving problems by imitating the process used by nature to create various organisms [2]. It is inspired by and uses the same combination of selection, recombination and mutation to evolve a solution to a problem. The Genetic Algorithm is a powerful stochastic search and optimization technique that uses the principles of evolution theory [2]. It helps in computing true or approximate solutions to optimization and search problems. It is based on the principle of the *Survival of the fittest*, where the fitter individuals have a higher opportunity to be selected to participate in the population of the next generation [16]. The advantage of using the genetic algorithm technique is that it is less susceptible to getting stuck at a local optimum [2, 16, 17]. This means that the algorithm is less likely to finding a solution within a neighboring set of solutions instead of obtaining the best solution within the whole solution space.

GA differs from conventional search techniques and starts with an initial set of random solutions called a *population* [2]. As shown in Figure 2.3, each individual in the population is called a *chromosome*, which represents a solution to the given problem. The chromosomes change through successive iterations, called *generations* [2, 17]. During each generation the chromosomes are evaluated through some measure of fitness using the fitness function [2]. The next generation is created with new chromosomes called offspring, which are formed, either by integrating two chromosomes from the current generation using the crossover operator or by modifying a chromosome using a mutation operator.

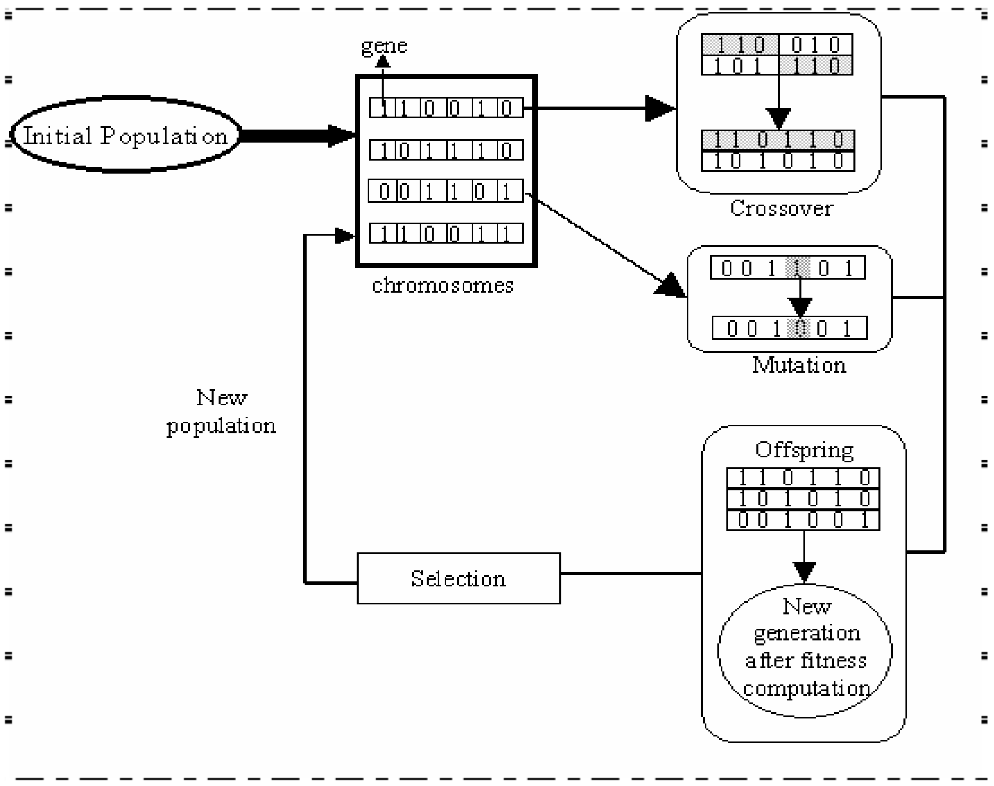

Figure 2.3 General representation of GA [2].

The new generation is formed by selecting from the offspring and some of the parents with the help of the fitness function. The rest are rejected in order to maintain a constant population size [2].

2.6 Previous Work

In terrestrial systems the cost of the meters and remote terminal units (RTUs) is the major concern for power system monitoring. This is due to the fact that several meters and RTUs need to be placed to cover the large terrestrial power system. Meter placement related to terrestrial power systems in general has been covered in references [3-16]. However, many of the references have used different approaches to solve the meter placement and the system observability problem. In reference [3], the proposed method incorporates cost, accuracy, reliability and bad data processing capability. A rule-based meter placement scheme has been proposed in reference [4], to identify the data requirements for real-time power monitoring and control of distribution. Reference [5] presents a simple root-based algorithm with efficient data structure to determine the topological observability from the available measurement data set. In references [6-10], a measurement system is designed considering the occurrence of topological changes and measurement losses. The methods in [11-13] consider intelligent systems techniques, such as Tabu search based and simulated annealing techniques, to obtain optimal metering systems. Only the observability requirement is taken into account in [6-13].

According to references [2, 17], the working of genetic algorithms can be summarized as the following:

Genetic algorithms (GAs) search through a set of solution using a fitness function to find the best possible solution from the solution set. The fitness function is an objective function that contains all the constraints applied to a given system. GAs minimize or maximize the fitness function depending on the requirement to determine the solution that satisfies these constraints.

In references [14-16], a genetic algorithm is employed to design a metering system on terrestrial systems. In [14] and [15], a meter placement scheme is developed without considering critical measurements. In reference [16], however, critical measurements and critical sets are determined and the genetic algorithm is mainly used to achieve a trade-off between investment cost and reliability of state estimation.

This thesis is inclined towards looking at another important aspect of electric ship power system, which is the SPS monitoring. The constraints applied to the design of a metering system for the ship power system are more and different than those applied for terrestrial system. Also traditionally, most terrestrial power systems have enough redundancy; so performing an ‘N-1’ contingency analysis is not really necessary. Due to the tighter operational constraints for the electric ship, high quality and accurate knowledge of the states of the electric power system is the key for survivability and fight-through capabilities. But the possibility of damage leads to the need for reconfigurable metering systems to help provide the best picture. This has resulted in selecting Genetic Algorithm as the optimization technique for this research. Also a study and comparison of different algorithms used for metering of the power system in references [2-17] has shown that the GA is one of the most accurate, reliable, efficient and robust algorithm which also yields quick results. It is also easy to understand. With the help of the MATLAB GA toolbox the fitness function (objective function of GA) was developed and tested. Genetic Algorithm as the search technique serves as a stepping stone for furthering research in the field of SPS monitoring.

2.7 Summary

This chapter discussed in detail the various concepts in power system monitoring and the previous work done in this field. In the electric power system of the ship, considering its small size, the financial aspect is not taken into account. However as research activities focus on different configurations for the electric ship, developing an algorithm to evaluate the various metering options for each configuration is another component in designing the best ship for survivability and fight-through capabilities. Important aspects like weight and space issues are considered. Thus, there is a need to minimize the number of measurements to avoid unnecessary equipment that occupies space and adds weight to the ship.

CHAPTER III

PROBLEM DESCRIPTION

The naval ship architecture has undergone considerable change in the recent years. The DD(X) architecture was one of the Navy's proposed ship system models. This typical DD(X) ship power system model shown in Figure 3.1 is used as one of the test systems in the research.

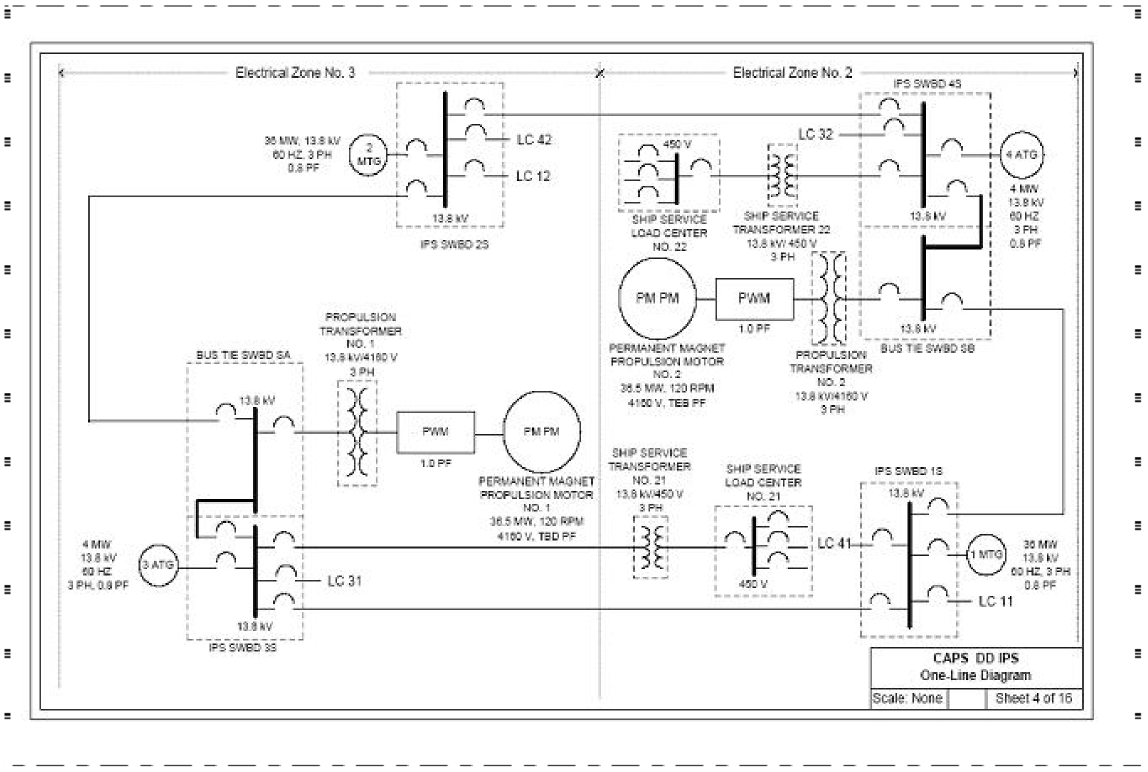

Figure 3.1 DD(X) Ship model [21]

The system model consists of two main turbine generators (MTGs) rated at 36 MW each and two auxiliary turbine generators (ATGs) rated at 4 MW each. Permanent magnet propulsion motors rated at 36.5 MW each support the propeller system. There are seven main buses that serve as load centers for the ship's load. The load centers are connected to auxiliary power units (APUs) rated at 0.5 MW each.

3.2 Problem Statement

The main points of concern in the research are, to reduce redundancy and to improve observability, reliability and robustness of the meter placement scheme. Placement of an adequate number of meters at different locations is taken as a prerequisite for system monitoring. The meter placement scheme not only takes into account the minimum number of meters required to make the system observable but also takes into account the vulnerability aspect such that even if a part of the system is lost or damaged, the rest of the system continues to be observable with the help of the existing meters. Hence, meter placement scheme is of primary importance in a shipboard power system monitoring not only due to its combinatorial nature but also due to the need to establish a trade-off between state estimator performances and the observability and vulnerability aspect of the ship power system. The genetic algorithm in this research is trained to predict the type and location of meters for a given system topology. It takes into consideration the power flow measurements at the branches and the power injection measurements at buses, checks for system observability and identifies critical measurements. The algorithm also ensures that the predicted meter types and meter location not only satisfy the first order (N-1) contingency analysis but also the second order (N-2) contingency analysis. The algorithm has been successfully tested with different system topologies.

3.3 Software Package and Tools

3.3.1 MATLAB

MATLAB is a high level language that can be used to perform intensive computation. It consists of several toolboxes and functions that help in technical computing. Some of its uses include mathematical computation, algorithm development, data acquisition and analysis, modeling and simulation, and visualization [20]. MATLAB is used at many universities and companies for research and design purposes.

3.3.2 OPTIMIZATION TOOLBOX

The Optimization Toolbox extends the capability of MATLAB in a technical computing environment [18, 19]. It consists of collection of functions, that are used for algorithm development and helps to define models, gather data, manage model formulations, perform tradeoff analysis and analyze results. The toolbox also facilitates incorporation of optimization methods in user-developed algorithms and models.

The algorithms in the optimization toolbox assist solving constrained and unconstrained continuous and discrete problems [19]. The toolbox is comprised of functions for linear programming, quadratic programming, nonlinear optimization, nonlinear least squares, nonlinear equations, multi-objective optimization, and binary integer programming [20].

3.3.3 *GENETIC ALGORITHMS*

The genetic algorithm toolbox is an extension of the optimization toolbox in MATLAB as shown in Figure 3.2.

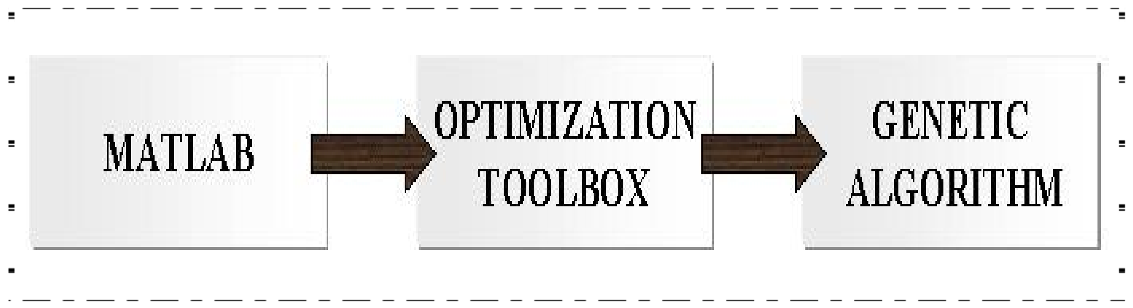

Figure 3.2 Block Diagram of Tools

According to reference [20], the GA is used to solve problems that are difficult to solve with traditional optimization techniques. These include problems that are not well defined or are difficult to model mathematically. It is also used when computation of the objective function is discontinuous, highly nonlinear or stochastic, or has unreliable or undefined derivatives [20]. The GA Toolbox can be accessed through a graphical user interface (GUI) or the MATLAB command line. But for this research the inbuilt functions of the GA are used, bypassing the GUI to create custom codes.

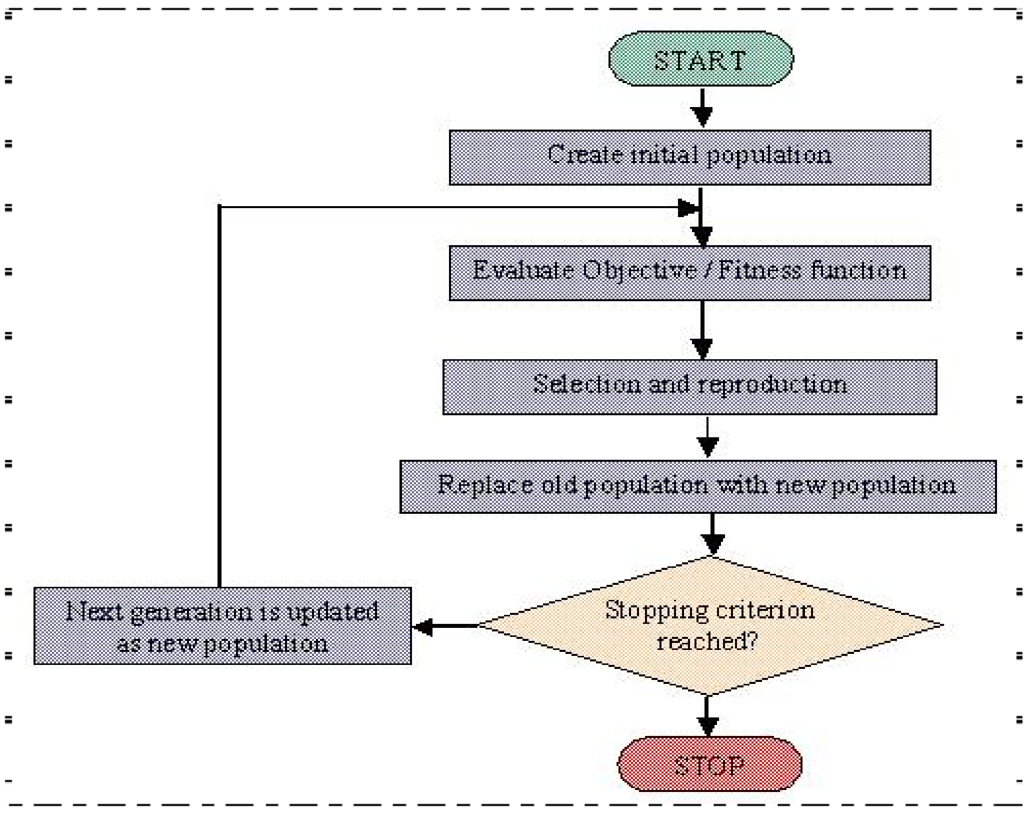
START
Create initial population
Evaluate Objective / Fitness function
Selection and reproduction
Replace old population with new population
Stopping criterion reached?
Next generation is updated as new population
STOP

Figure 3.3 Flowchart of Genetic Algorithms

The flowchart in Figure 3.3 shows the working of the genetic algorithms. The solution of the optimization problem using the genetic algorithm technique requires the generation of successive populations of individuals where each of the newly generated population is better than the previous generation. The initial population is evaluated each time using the fitness function. A new population is generated through selection and reproduction process. The new population to be evaluated by the fitness function then replaces the old population. The best individual is found by searching within the population. At the end of several iterations the algorithms converge to the best solution or the chromosome.

The three most important aspects of using genetic algorithms are: definition of the objective function, definition and implementation of the genetic representation, and definition and implementation of the genetic operators [2, 17]. A typical genetic algorithm requires a genetic representation of the solution domain and a fitness function to evaluate the solution domain. The success of the optimization process depends on the design of the fitness function for the problem.

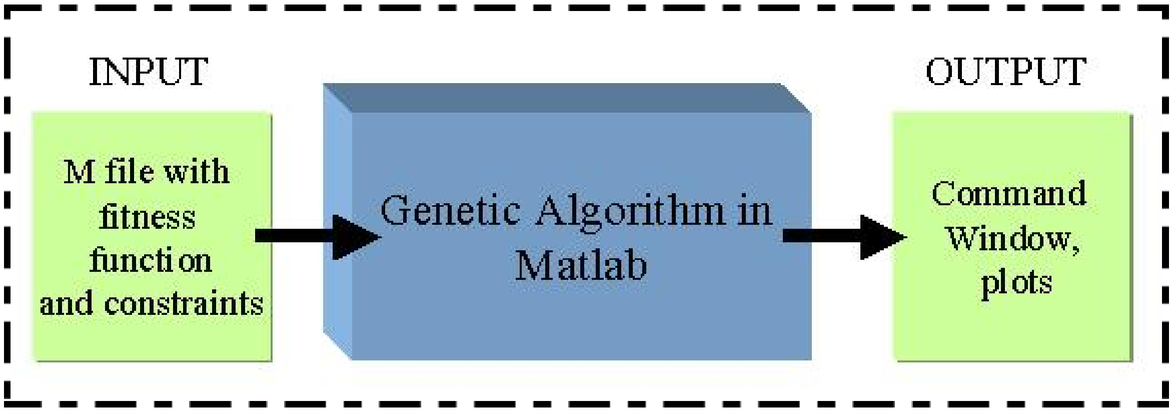
INPUT
M file with fitness function and constraints
Genetic Algorithm in Matlab
OUTPUT
Command Window, plots

Figure 3.4 GA in MATLAB

In MATLAB, the genetic algorithm receives input from m-file containing the fitness function and constraints for the given problem, and the output is displayed in the command window and in the form of plots.

3.3.3.1 Terminologies

Some genetic algorithm terminologies are [2, 20]:

1. **Population:** It consists of individuals or chromosomes, which represent the solution set for the problem.

2. **Initialization**: Many individual solutions are randomly generated initially to form an initial population. The population size depends on the nature of the problem.
3. **Selection:** Fitter individual solutions are chosen through a fitness-based process to breed the new generation.
4. **Crossover operator**: It is the main genetic operator. It operates on two individuals and recombines their genetic material to generate new individuals in the next population.
5. **Mutation:** It is a background operator, which produces spontaneous random changes in various chromosomes by altering one or more genes. It provides genetic diversity and enables the Genetic Algorithm to search a broader space.
6. **Reproduction**: Offspring are created for the next generation population of solutions through genetic operators: crossover and/or mutation.
7. **Termination:** The generation of individuals is continued until the stopping criterion is reached.
8. **Elitist strategy:** It is usually used to make sure that the best-fitted individual in a generation appears in the population of the next generation.

3.4 Summary

This chapter gives an outline of the problem that has been solved in the research. It also provides an overview of the tools and algorithms used to arrive at the solution. The working of genetic algorithms and the terminology used are discussed.

CHAPTER IV

METER PLACEMENT SCHEME

4.1 Introduction

This chapter explains in detail the working of the algorithm for the meter placement scheme for the shipboard power system. The algorithm consists of four parts namely: System observability, identification of critical measurements, first order contingency analysis and second order contingency analysis.

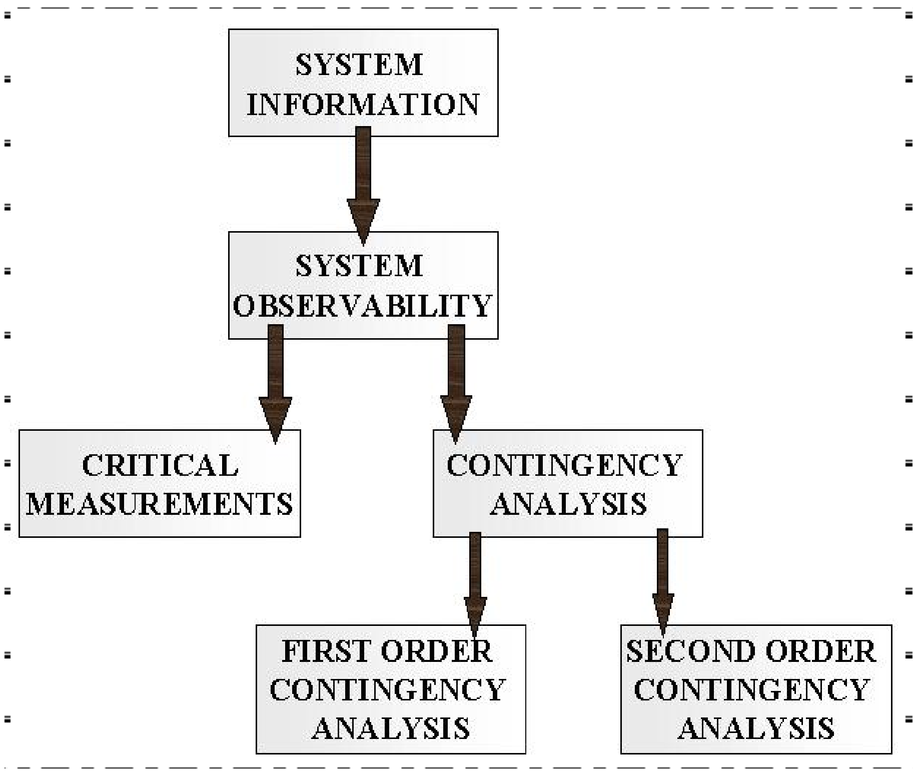

Figure 4.1 Block Diagram Representation of Algorithms

The algorithms shown in Figure 4.1 were coded and tested in MATLAB. The genetic algorithm used many inbuilt functions of the MATLAB GA Toolbox. The fitness function fed to the genetic algorithm incorporated various constraints like the system observability, meter type, critical measurements and the contingency analyses which were written as MATLAB scripts. The system observability algorithm was used to verify if a particular meter placement scheme maintained system observability or not. The critical measurements algorithm determined the presence of critical measurements in the system topology. The first and second order contingency analysis algorithms removed one or two meters as required each time to determine the unobservable branches in the system. The fitness function of the genetic algorithm was evaluated for many meter arrangements, based on the outputs from all the above-mentioned algorithms and the weights applied to each function. The best solution given by the algorithm had the minimum fitness function value.

4.2 Computer System Configuration and Software Version

This research was performed using a computer with 3.40GHz Intel® Pentium® IV processor and 1GB RAM. The operating system used was Microsoft Windows 2000 Professional, version 5.0.2195 Service pack 4 build 2195. The software used was MATLAB, version 7.0.1.267 (R2006b). Many inbuilt functions from Optimization Toolbox, version 3.1 and Genetic Algorithm Toolbox, version 1.0.2 in MATLAB were used.

4.3 Details of the Algorithm

4.3.1 Input Parameters

The genetic algorithm took the system topology, meter types (power injection meters at the buses and power flow meter at the branches) and the location of the meters on the system as inputs. The input m-file contained information about the number of buses, the number of branches connecting the buses, how the branches are connected between the buses and also all possible meter locations for both power flow meters and power injection meters. The power injection meters were considered first as input followed by the power flow meters in the numerical order. The number of meters varied with the topology considered. Only one meter was placed on each bus and branch. In the meter arrangement scheme, meters were placed randomly on the buses and branches initially. A value of 1 or 0 was assigned to the meter status depending on its presence or absence of the meter.

Examples of the input file for the three test systems (3-bus, 6-bus and the 18-bus ship systems) are shown in Appendix C.

4.3.2 GA Options

The GA included several settings and options like population size, population type, stall generation limit etc which were set according to the requirement before running the algorithms. The ‘PopulationType’ option was set to ‘bitString’. Mutation function was chosen as uniform. The stall generation limit, stall time limit, generations, and population size options were set according to user requirement for the genetic algorithm. The stall generation limit acted like a stopping criterion for the genetic algorithm. The *stall generation* is a count of the number of successive generations when the same minimum fitness value appears, when the algorithm is executed. The *population size* is the number of individuals or meter arrangements considered during each generation. The initial starting conditions were set such that the GA started evaluating a random meter arrangement from its solution set, each time it was run.

4.3.3 System Observability

In order to perform state estimation, a minimum number of measurements were required, to predict the rest of the system values. The system observability algorithm determined whether the currently available set of measurements from the power system network provided sufficient information to allow computation of the state estimation. If all the required measurements throughout the system network can be estimated by processing the given set of measurements in the system, then the network is said to be *observable*; otherwise it is said to be *unobservable*.

The input to the system observability algorithm was the power system topology, meter type and location. The algorithm was developed based on the following steps:

4.3.3.1 Build bus incidence matrix (I)

The bus incidence matrix was built based on the equations explained in [1]. For each branch of the given system, the start bus location was given a value of 1, and the end bus was given a value of –1.

$$I(i,j) = \begin{cases} 1 \text{ if bus } j \text{ is the sending end of branch } i. \\ -1 \text{ if bus } j \text{ is the receiving end of branch } i \\ 0 \text{ otherwise} \end{cases} \qquad [1]$$

For example, consider the system in Figure 4.2.

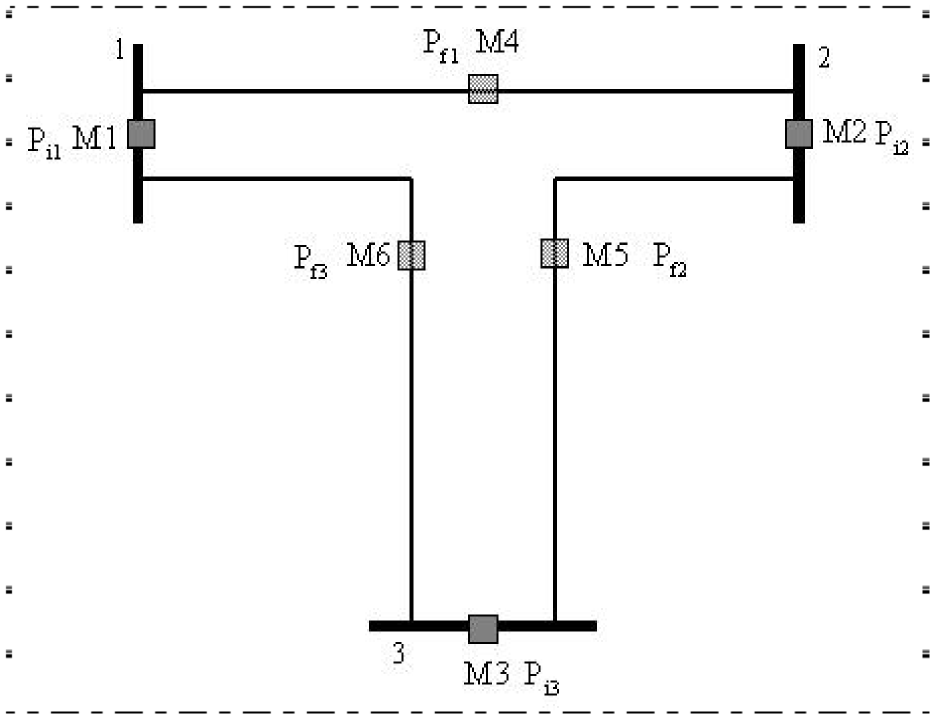

Figure 4.2   3-bus system with meters at all branches and buses.

In matrix I, the branches represent the rows and the buses represent the columns. The matrix I shown below is calculated for the 3-bus system shown in Figure 4.2.

$$I = \begin{array}{c|ccc} & B1 & B2 & B3 \\ \hline Br1 & 1 & -1 & 0 \\ Br2 & 0 & 1 & -1 \\ Br3 & 1 & 0 & -1 \end{array}$$

where *B1, B2* and *B3* are the buses and *Br1, Br2* and *Br3* are the branches

4.3.3.2 Build Jacobian matrix (H)

The Jacobian matrix was built based on the equations explained in [1]. The number of rows equaled the total meters present, and the number of columns equaled the number of buses. The total power flow at the bus is equal to the sum of the power flow at the branches connected to that bus as shown. For example, consider the system in Figure 4.2.

$$P_{i1} = P_{12} + P_{13} \tag{4.1}$$

$$P_{i1} = [(\theta_1 - \theta_2) + (\theta_1 - \theta_3)] = 2\,\theta_1 - \theta_2 - \theta_3 \tag{4.2}$$

$$P_{f1} = \theta_1 - \theta_2 \tag{4.3}$$

where P_{i1} is the power flow through injection meter 1 at the bus 1

P_{f1} is the power flow through flow meter 1 at the branch 1-2.

θ_1, θ_2 and θ_3 are the phase angles at each bus respectively.

The power flow equations for the system in Figure 4.2 are represented in matrix form. The coefficients of θ_2 and θ_3 are shown in matrix H. The H matrix below is developed for the meter arrangement scheme shown in Figure 4.2.

$$H = \begin{array}{c} \\ P_{i1} \\ P_{i2} \\ P_{i3} \\ P_{f1} \\ P_{f2} \\ P_{f3} \end{array} \begin{array}{c} \begin{array}{cc} \theta_2 & \theta_3 \end{array} \\ \begin{bmatrix} -1 & -1 \\ -1 & 0 \\ -1 & 2 \\ -1 & 0 \\ 1 & -1 \\ 0 & 1 \end{bmatrix} \end{array}$$

Note: Since bus 1 was taken as reference bus.

$$\theta_1 = 0 \tag{4.4}$$

The column representing θ_1 is a null column.

4.3.3.3 Build states of the system (θ_t)

θ_t was determined by solving the equation 2.1 in Chapter 2, Section 2.2.

The observability of the meter arrangement scheme is evaluated using the equations 2.1, 2.2 and 2.3 in Chapter 2, Section 2.2.

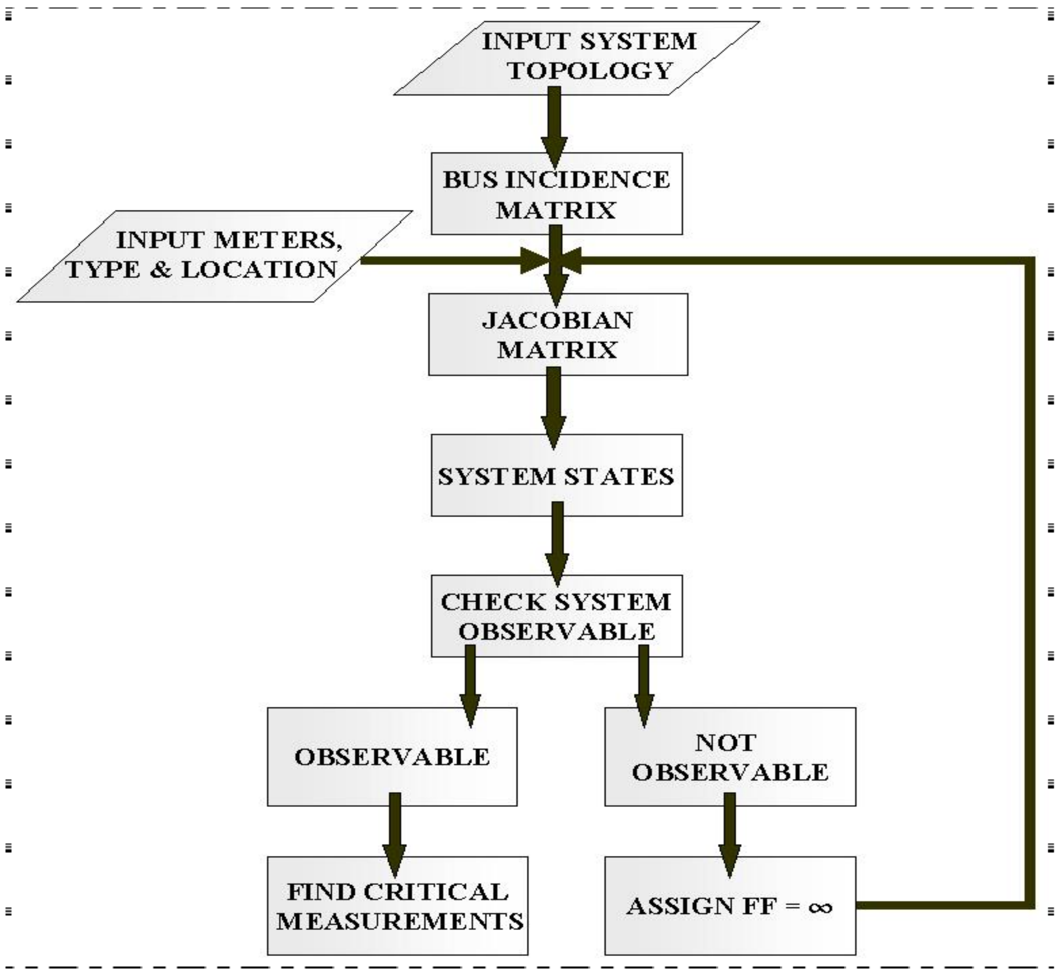

Figure 4.3 Block Diagram Representation of Working of System Observability Algorithm.

The block diagram in Figure 4.3 shows the working of the system observability algorithm. The system observability algorithm was executed each time the system observability function is called to evaluate a meter arrangement scheme or chromosome. If the particular meter arrangement scheme or chromosome was observable, then the algorithm continued to find the critical measurements. If the system was non-observable, the algorithm set the fitness function value to be infinite, and then started again for a different meter arrangement scheme.

4.3.4 Critical measurements

The critical measurements were determined as explained in Section 2.3 in Chapter 2 and in [1]. If no critical measurement was found, the algorithm continued to the next step in the genetic algorithm. Figure 4.4 shows a block diagram representation to determine critical measurements. The critical measurements were found in a meter arrangement scheme only if the system is observable. Peter-Wilkinson decomposition was applied to the Jacobian matrix to get matrix L. The matrix L was then divided into two matrices L1 and L2 as explained in Chapter 2 Section 2.3. The matrix C was evaluated. The null column number in matrix C indicated that the corresponding row in Jacobian matrix (H), was a critical measurement. There can be more than one critical measurement based on the number of null columns present in matrix C. After determining the critical measurements the algorithm proceeded to the next step to perform first and second order contingency analysis.

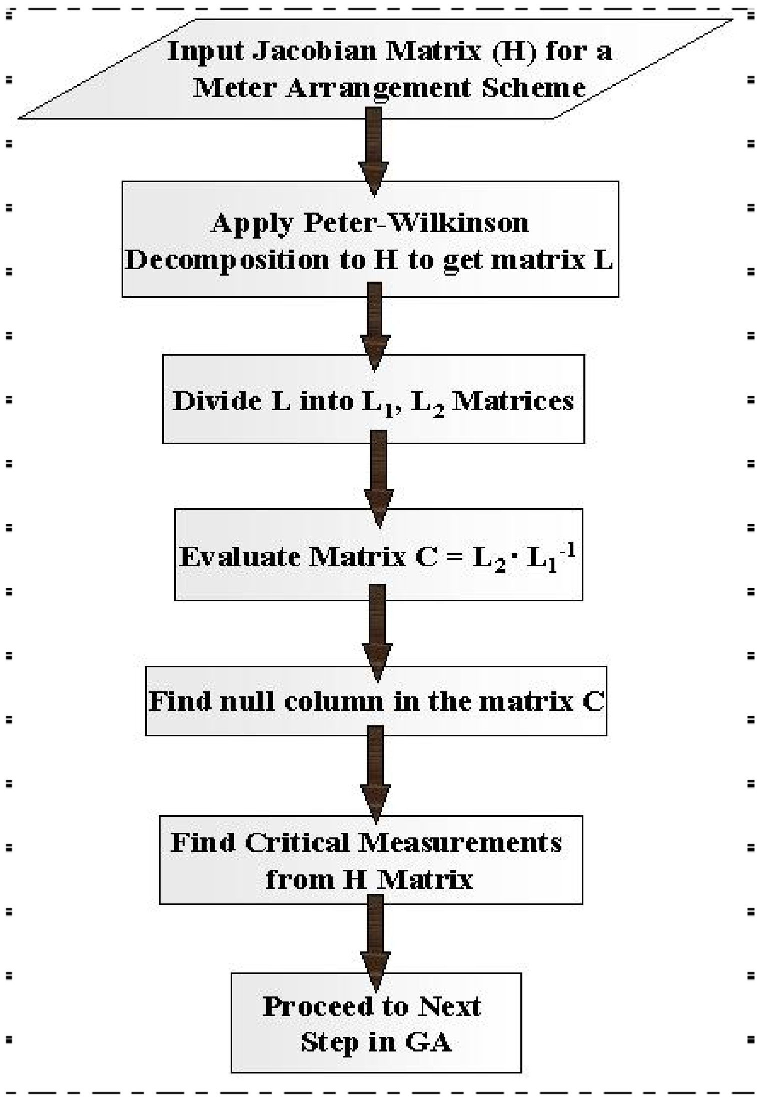
Input Jacobian Matrix (H) for a Meter Arrangement Scheme
Apply Peter-Wilkinson Decomposition to H to get matrix L
Divide L into L_1, L_2 Matrices
Evaluate Matrix $C = L_2 \cdot L_1^{-1}$
Find null column in the matrix C
Find Critical Measurements from H Matrix
Proceed to Next Step in GA

Figure 4.4 Block Diagram Representation for Determining Critical Measurements

4.3.5 First Order Contingency Analysis

While performing the first order contingency analysis, the algorithm initially removed one meter from the current meter placement scheme that was being evaluated. It checked to see if the new arrangement was observable. For this purpose, the system observability algorithm was executed each time a meter was removed and the number of unobservable branches was calculated. The meter was then replaced and the same procedure repeated with another meter. The number of unobservable branches was thus calculated for every meter removed from the meter placement scheme, and was averaged over the number of meters for every meter type. These averaged values served as first order contingency constraints to be minimized in the fitness function.

For example, let there be two power injection meters and three power flow meters in a particular meter arrangement scheme. Also, suppose the total number of unobservable branches obtained after removal of each of the power injection meters at a time is six. Hence, the first order contingency analysis function value for power injection meter will be 3, which is the average number of non-observable branches. The same is repeated for unobservable branches obtained after removal of each of the power flow meters.

4.3.6 Second Order Contingency Analysis

The second order contingency analysis is quite similar to first contingency analysis, except in the former two meters were removed at a time from a possible arrangement of meters. The meters removed could be - two power injection meters, two power flow meters or one power injection meter and one power flow meter. The algorithm evaluated the average number of unobservable branches for removal of each of the above-mentioned three possible cases of two meters. The average number of unobservable branches became another constraint to be minimized in the fitness function.

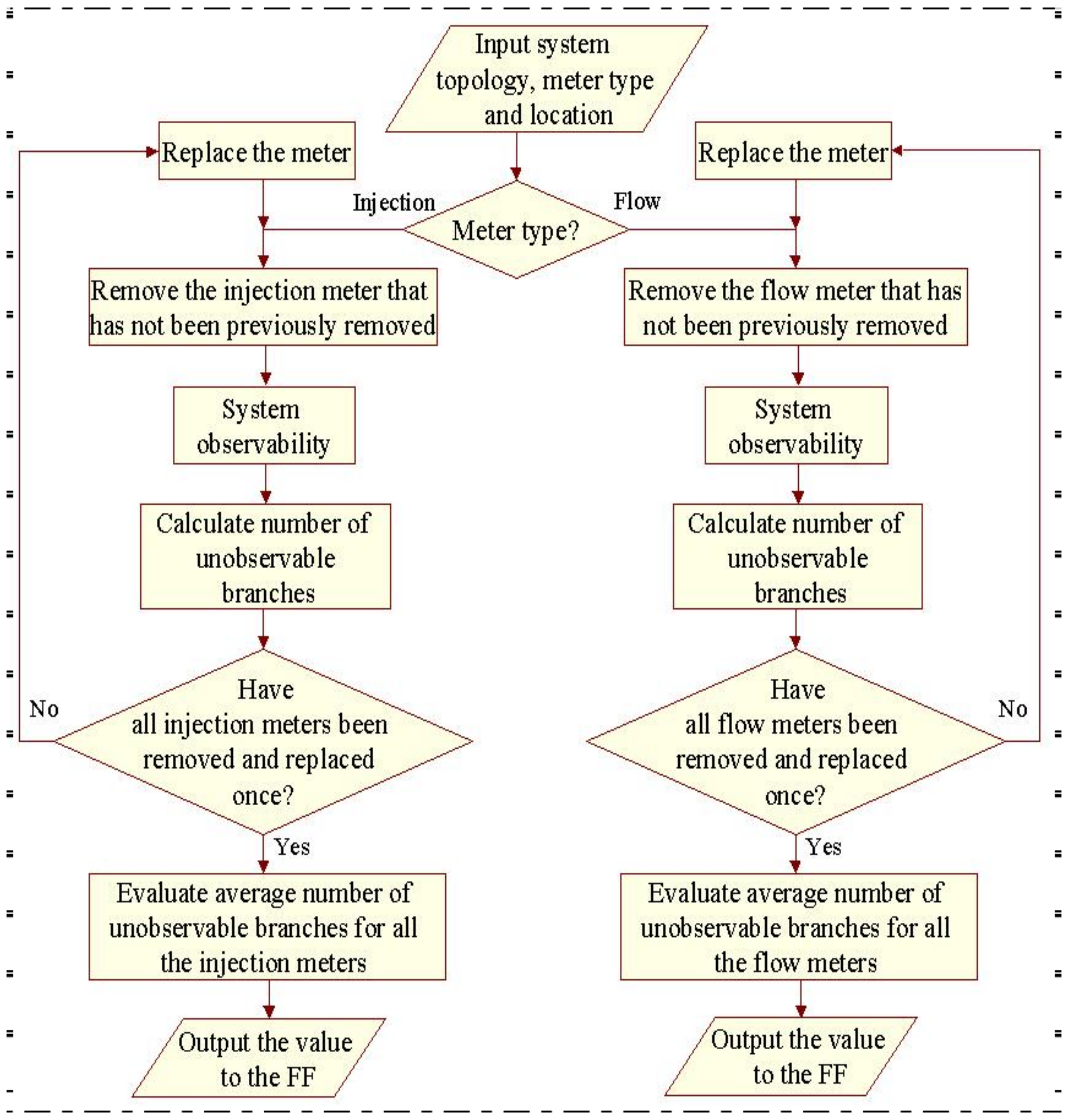
Input system topology, meter type and location
Replace the meter
Replace the meter
Injection
Flow
Meter type?
Remove the injection meter that has not been previously removed
Remove the flow meter that has not been previously removed
System observability
System observability
Calculate number of unobservable branches
Calculate number of unobservable branches
No
Have all injection meters been removed and replaced once?
Have all flow meters been removed and replaced once?
No
Yes
Yes
Evaluate average number of unobservable branches for all the injection meters
Evaluate average number of unobservable branches for all the flow meters
Output the value to the FF
Output the value to the FF

Figure 4.5 Block Diagram Representation of Working of First order Contingency Analysis Algorithm

4.3.7 Fitness Function (FF)

Fitness function is also known as the objective function that is minimized or maximized in the genetic algorithm to obtain an optimal solution from the solution set [2, 17, 18, 22]. In this genetic algorithm, the fitness function developed was minimized considering all constraints applied to it, to obtain minimum number of meters that need to be placed in a particular ship power system. The fitness function for the electric ship was formulated considering the following problem constraints:

- System observability
- Measurement type (Power flow meter & power injection meter)
- Critical measurements
- First order contingency analysis
- Second order contingency analysis

The proposed fitness function for the electric ship is as follows:

$$\begin{aligned} FF = \text{Min } [&P1 \times num\_flow\_meas + P2 \times num\_inj\_meas + P3 \times crit\_meas + \\ &P4 \times FO\_cont\_anal\_flow + P5 \times FO\_cont\_anal\_inj + \\ &P6 \times SO\_cont\_anal\_(2)\,flow + P7 \times SO\_cont\_anal\_(2)\,inj + \\ &P8 \times SO\_cont\_anal\_(1)\,flow\_(1)\,inj] \end{aligned} \tag{4.5}$$

where *P1, P2 , P3* etc. are the weights assigned to each individual constraint depending on their priority.

num_flow_meas is the number of flow measurements in the meter arrangement scheme being evaluated.

num_inj_meas is the number of injection measurements in the meter arrangement scheme being evaluated.

crit_meas is the critical measurements in the meter arrangement scheme being evaluated.

FO_cont_anal_flow is the output of the first order contingency analysis based on removal of one flow meter at a time.

FO_cont_anal_inj is the output of the first order contingency analysis based on removal of one injection meter at a time.

SO_cont_anal_(2)flow is the output of the second order contingency analysis based on removal of two flow meters at a time.

SO_cont_anal_(2)inj is the output of the second order contingency analysis based on removal of two injection meters at a time.

SO_cont_anal_(1)flow_(1)inj is the output of the second order contingency analysis based on removal of one flow meter and one injection meter at a time.

4.3.8 Weights

Weights play a significant role in the development of the fitness function. The value of the weights could be based on any scale (0-1, 0-10, 0-100 etc). They were applied to each constraint in the fitness function based on their importance. If the need for minimization of a particular constraint is higher than all the other constraints in the FF, the value of the weight assigned to it will be the highest. Higher weights bias the FF in the direction of that constraint. The following were the weights given to each constraint in the fitness function:

1. Weight applied to number of flow meters = P1
2. Weight applied to number of injection meters = P2
3. Weight applied to critical measurements = P3
4. Weight applied to first order contingency analysis for loss of one flow meter = P4
5. Weight applied to first order contingency analysis for loss of one injection meter = P5

6. Weight applied to second order contingency analysis for loss of (2) flow meters = P6
7. Weight applied to second order contingency analysis for loss of (2) injection meters = P7
8. Weight applied to second order contingency analysis for loss of (1) flow meter and (1) injection meter = P8

Chapter 5 shows different meter placement schemes as output for a 6-bus system as weights applied to each constraint in the FF were varied in each case.

4.3.9 Assumptions

The following are a list of assumptions made while developing the algorithms:

- Only one meter could be placed on each branch or bus.
- GA started with a random initial population each time it ran.
- Bus 1 for any system topology was considered to be the reference bus. Therefore $\theta_1 = 0^\circ$ always (where θ_1 is the phase angle).

4.4 Meter Placement Scheme Flowchart

The flowchart of the GA based meter placement scheme shown in Fig 4.6 gives a clear picture of the flow of the algorithm. The input parameters considered and the various steps involved to determine an intelligent meter placement algorithm are explained in detail in this chapter. A brief summary scheme is as follows. The GA took system topology, the meter types considered and their location in the system as the input. The GA options were set and it started with a random initial condition. The output of the system observability algorithm was fed to the fitness function. The bus incidence matrix, and the Jacobian matrix were developed and the states of the system were determined in the system observability algorithm. The observability check was then performed to determine if the meter arrangement scheme being evaluated maintains system observability or not.

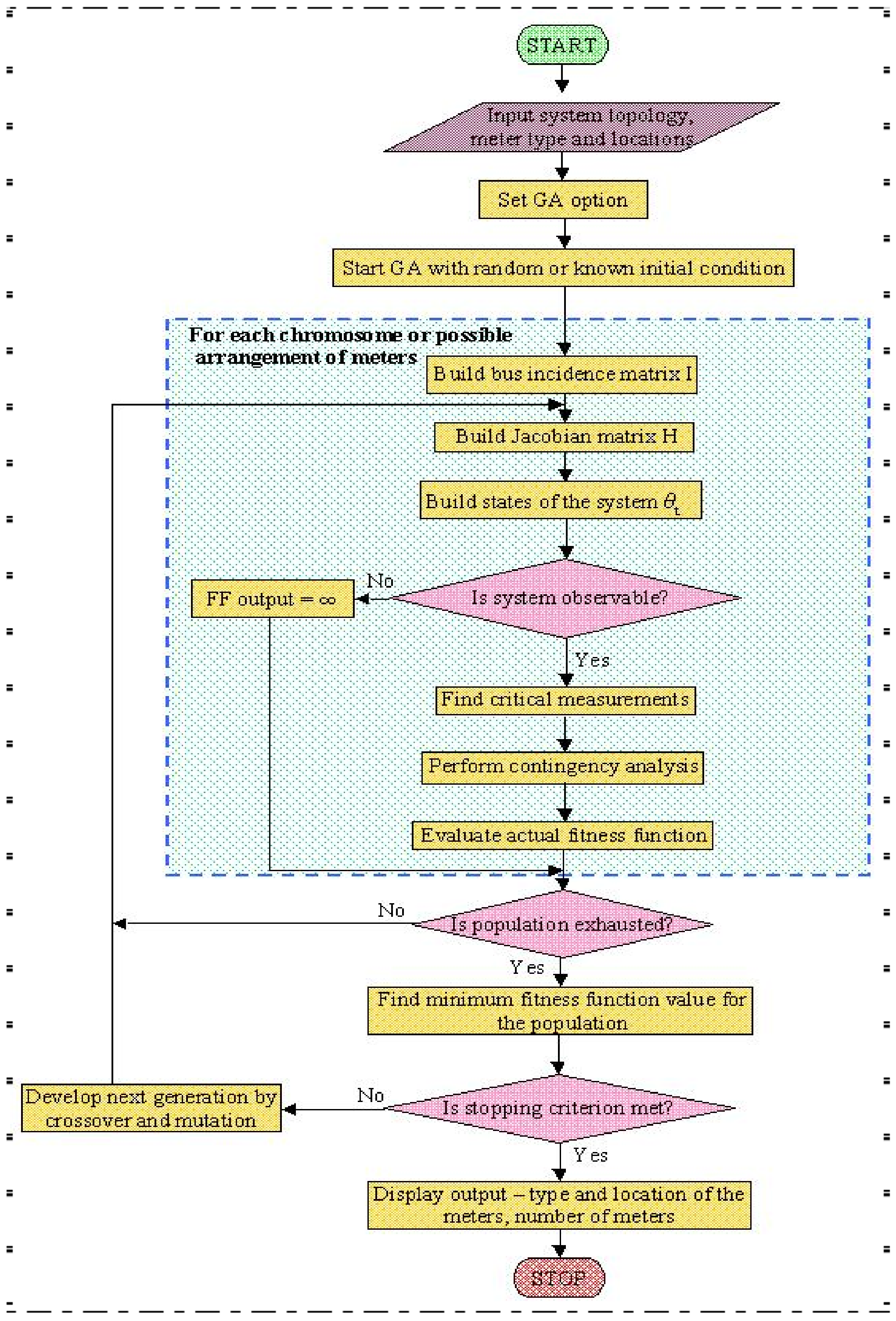
START
Input system topology, meter type and locations
Set GA option
Start GA with random or known initial condition
For each chromosome or possible arrangement of meters
Build bus incidence matrix I
Build Jacobian matrix H
Build states of the system θ_t
Is system observable?
No
FF output = ∞
Yes
Find critical measurements
Perform contingency analysis
Evaluate actual fitness function
Is population exhausted?
No
Yes
Find minimum fitness function value for the population
Is stopping criterion met?
No
Develop next generation by crossover and mutation
Yes
Display output – type and location of the meters, number of meters
STOP

Figure 4.6 Flowchart of Meter Placement Scheme

If the system was not observable the fitness function output was set to an infinite value. The algorithm proceeded to check if all the population had been exhausted. If the population was not exhausted the next individual or meter arrangement was evaluated. If the population was exhausted the algorithm found the minimum fitness value for the population and checked if the stopping criterion was met which was the stall generation limit. If the stopping criterion was met, the algorithm displayed the output and exited. If the stopping criterion was not met it developed the population for the next generation of meter arrangements by crossover and mutation. The GA then proceeded to evaluate each of the meter arrangement schemes in the population for that generation.

If the system was observable, the critical measurements were determined and the first and second order contingency analyses were performed. The fitness function output was determined based on the output from the system observability algorithm, the critical measurements, and the first and second order contingency analyses algorithms. The behavior of the fitness function depended on how the different component algorithm outputs were weighted. The fitness function was minimized to obtain the minimum number of meters, their type and location for a particular system topology that satisfy all the constraints applied. The fitness function output was evaluated for all the individuals in a population and the best and mean fitness values were determined for each generation.

4.5 Overview of the Test cases

The algorithm was successfully tested with three test systems namely:

1. 3-bus test system

2. 6-bus DDX test system
3. 18-bus ship system

The figures below show the three-test systems used in the research.

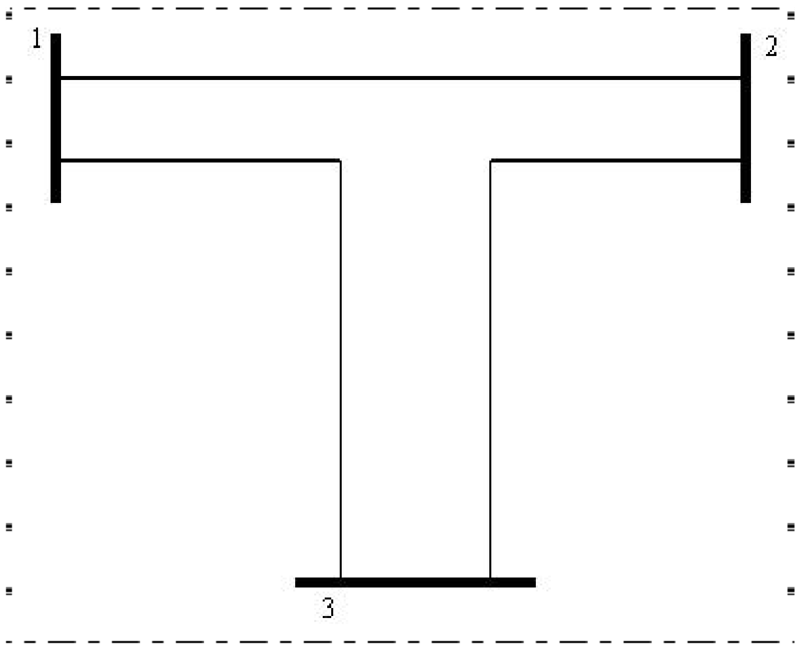

Figure 4.7 3-bus test system

The 3-bus test system has 3 branches. If a meter was placed on every bus and branch the maximum number of meters that could be placed was 6.

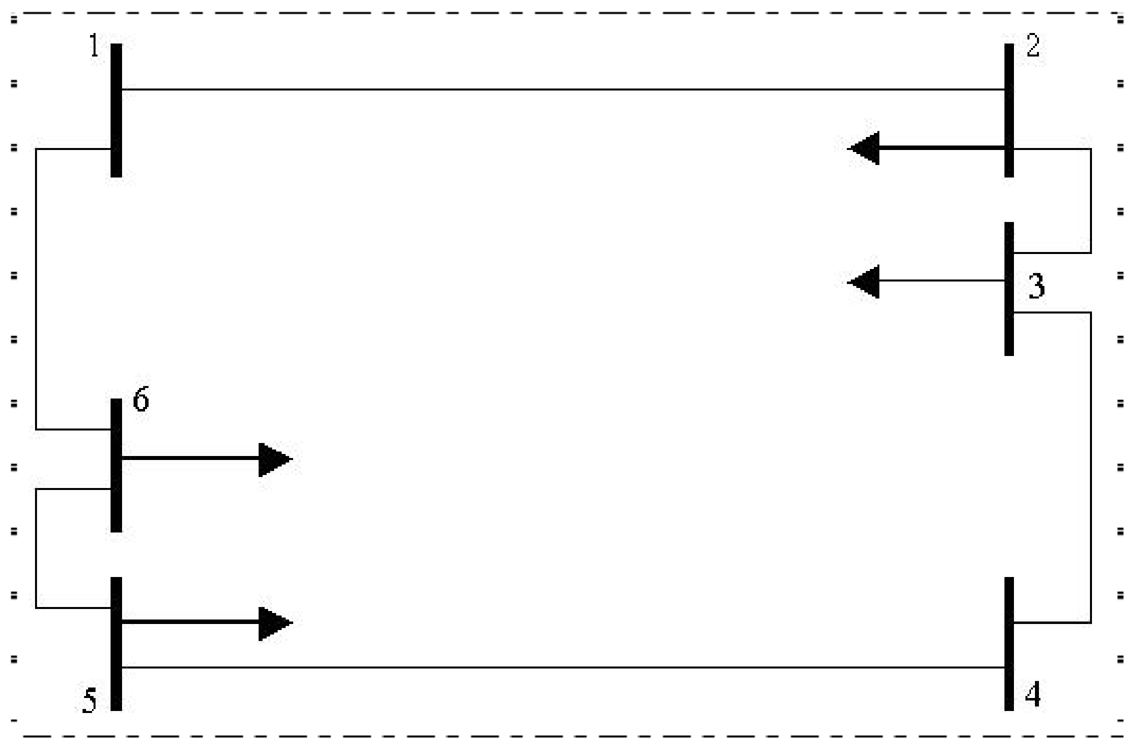

Figure 4.8 6-bus DDX test system [21]

Similarly for the 6-bus system with 6 branches, the maximum number of meters that can be placed was 12, and for the 18-bus, 17-branches system the maximum number of meters possible was 35.

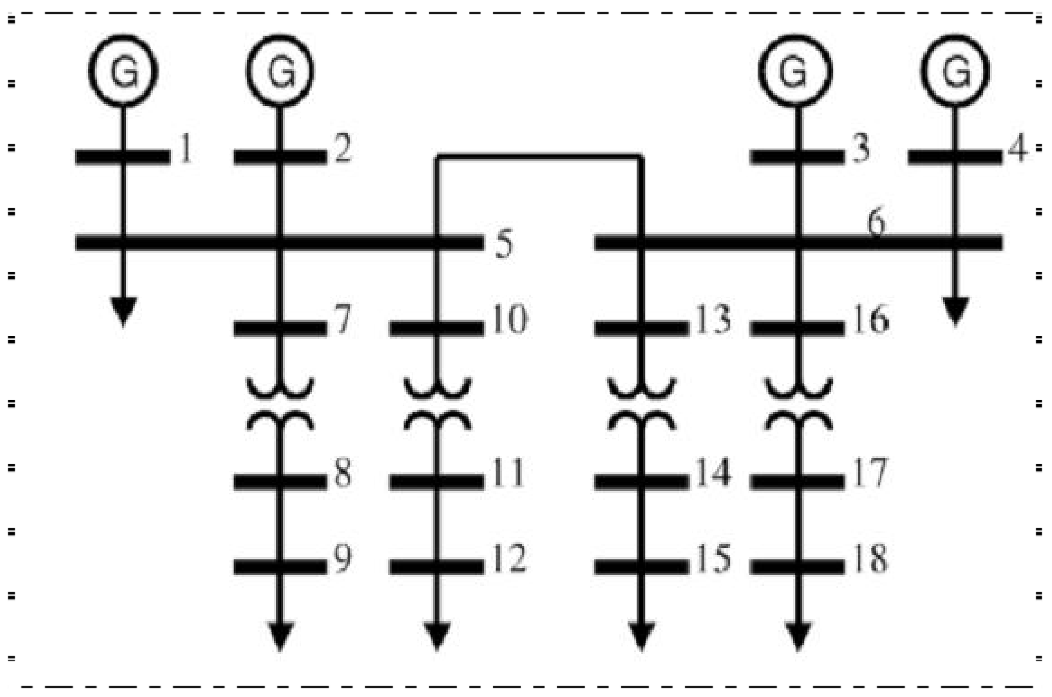

Figure 4.9 18-bus Ship system [22]

For each test system the algorithm was run to determine the meter arrangement scheme with minimum number of meters with constraints like: minimum number of meters without any other constraint, minimum number of flow meters, minimum number of injection meters, first order contingency analysis with loss of flow meters, first order contingency analysis with loss of injection meter, second order contingency analysis with loss of two flow meters, second order contingency analysis with loss of two injection meters and second order contingency analysis with loss of one flow and one injection meter. The genetic algorithm gave the location of the meters and also the minimum number of meters for each test case for a particular test system. The number of meters given as output each time at the end of the execution of the algorithm was less than the maximum number of meters that can be placed in the particular test system.

4.6 Summary

This chapter explains in detail the algorithms developed for intelligent placement of meters in the ship power system using genetic algorithm. The chapter also discusses the version of the software and the computer system configuration used to develop the algorithms. It presents the GA options and the assumptions made to execute the algorithm. The chapter also outlines the test cases used to successfully demonstrate the working of the algorithm and the expected results from the GA for each test system.

CHAPTER V

RESULTS AND DISCUSSION

The algorithms were successfully tested on the three test systems (3-bus, 6-bus and 18-bus test systems). The results for the 6-bus test system are discussed in detail in this chapter. The results for the 3-bus system and the 18-bus system are summarized later in the chapter. More results from the 3-bus system and the 18-bus system are attached in Appendices A and B. A summary of the results from the three test systems is also presented in the end of this chapter.

5.1 6 - BUS TEST SYSTEM

Figure 5.1 shows the 6-bus test system considered to test the algorithms and the fitness function. The figure also shows all possible locations where meters can be placed in the system. The meters are numbered, starting with all the injection meters followed by the flow meters in the clockwise direction.

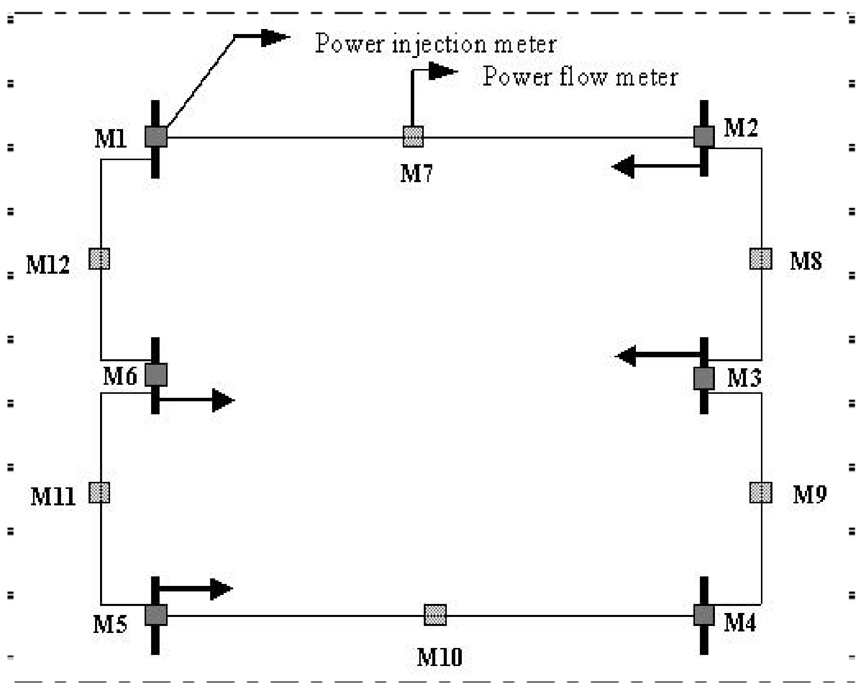

Figure 5.1 6-bus test system with meters at all buses and branches

An explanation about the weights is given in detail in Chapter 4, Section 4.3.8. A brief outline of the weights is given below.

Weight applied to number of flow meters = *P1*

Weight applied to number of injection meters = *P2*

Weight applied to critical measurements = *P3*

Generic weight factor for FO contingency = pen_first

Generic weight factor for SO contingency = pen_second

Fail probability for loss of 1 flow meter = prob_fail_flow

Fail probability for loss of 1 injection meter = prob_fail_inj

Weight applied to loss of 1 flow meter in FO contingency, *P4* = pen_first × prob_fail_flow

Weight applied to loss of 1 injection meter in FO contingency, *P5* = pen_first × prob_fail_inj

Weight applied to loss of 2 flow meter in SO contingency, *P6* = pen_second × prob_fail_flow × prob_fail_flow

Weight applied to loss of 2 injection meter in SO contingency, *P7* = pen_second × prob_fail_inj × prob_fail_inj

Weight applied to loss of 1 flow and 1 injection meter in SO contingency, *P8* = pen_second × prob_fail_inj × prob_fail_flow

Table 5.1 Weights for Constraints

Case No.	P1	P2	P3	pen_ first	pen_ second	prob_fa il_flow	prob_ fail_i nj	P4	P5	P6	P7	P8
1	10	10	0	0	0	0	0	0	0	0	0	0
2a	100	10	0	0	0	0	0	0	0	0	0	0
2b	10	100	0	0	0	0	0	0	0	0	0	0
3	1	2	10	0	0	0	0	0	0	0	0	0
4a	4	1	1	50	0	0.95	0.05	47.5	2.5	0	0	0
4b	1	4	1	50	0	0.05	0.95	2.5	47.5	0	0	0
5a	4	1	1	5	124	0.9	0.1	4.5	0.5	100.44	1.24	11.16
5b	1	4	1	5	124	0.1	0.9	0.5	4.5	1.24	100.44	11.16
5c	1	4	1	3	124	0.5	0.4	1.5	1.2	31.25	19.84	24.8

In general, the value of weights applied to different constraints, depends on the number and type of constraints within the fitness function, as they determine the behavior of the fitness function. The selection of weights depends on the system topology as well.

Table 5.2 Fitness Function Constraint Ranges for 6-bus Test case

Constraint Type	**Range**
Number of power flow measurements	0 – 6
Number of power injection measurements	0 – 6
Number of critical measurements	0 – 12
FO contingency analysis for loss of 1 flow meter	0 – 6
FO contingency analysis for loss of 1 injection meter	0 – 6
SO contingency analysis for loss of 2 flow meters	0 – 6
SO contingency analysis for loss of 2 injection meters	0 – 6
SO contingency analysis for loss of 1 flow and 1 injection meter	0 – 6

Table 5.2 shows the scale for various constraints in the fitness function. As only one power flow meter can be placed on each branch the possible number of power flow meters ranges between 0 - 6 meters. The same holds for power injection meters at buses. Critical measurements can range up to the total number of possible meters in the power system. First and second order contingency analyses provide the average number of unobservable branches as inputs to the fitness function. These range as shown in Table 5.2. The fitness function constraint ranges depend on the system topology, i.e. the number of buses and branches in the system. Hence it varies accordingly for the 3 - bus test system and the 18 – bus test system.

5.1.2 CASE 1: Minimum number of meters without considering other factors

As shown in Table 5.1 weights were equally applied to the number of flow meters and injection meters such that the fitness function is minimized to find the minimum number of meters, irrespective of the meter type and without considering other factors or constraints. All other weights were zero.

Profile time is the time taken by the GA to execute the algorithm considering all the constraints and display the outputs.

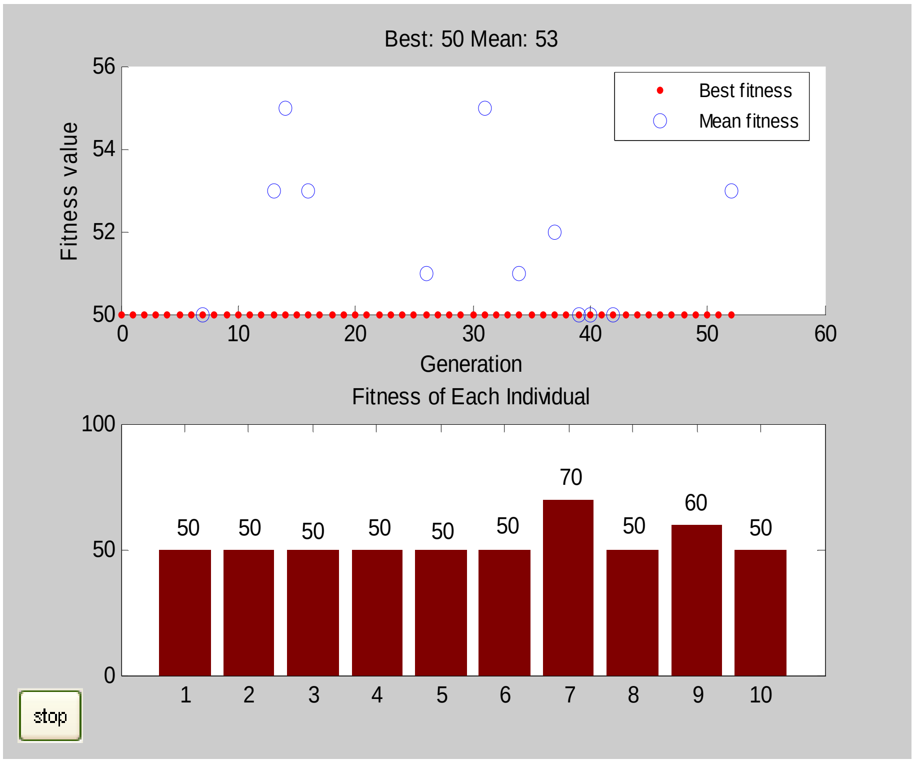

Figure 5.2 MATLAB Plot for Genetic Algorithm – case 1

Figure 5.2 is the MATLAB plot showing the output of the Genetic Algorithm for case 1. The subplot 1 in the graph above shows the fitness value versus generation. It shows the best and the mean fitness value appearing in each generation. Whenever the mean fitness value for a particular generation tends to infinity then the value is not shown in the plot.

The subplot 2 in the graph shows the fitness value for each individual or meter arrangement scheme. The first set of individuals appearing in the subplot 2 is the elite, the middle ones are from crossover and the individuals in the end are from mutation. The best fitness value is the minimum fitness value of an individual appearing in the population. Subplot 2 displays the population of individuals in the last generation, before the stopping criterion is reached. The best fitness value from plot 2 is 50.

The plots in all the subsequent cases (Case 2a - Case 5c) and other test systems (3-bus system and 18-bus system) show the best and the mean fitness values for all the generation and the fitness value of each individual meter arrangement scheme in the last generation for the respective cases.

Each generation consists of 10 meter arrangement schemes (individuals). The first 10 meter arrangement schemes were chosen randomly by the GA. The subsequent meter arrangement schemes were developed by crossover and mutation. The mean fitness value was found for each generation. Table 5.3 indicates that at the end of 52 generations the algorithm evaluated 520 meter arrangements.

Table 5.3 Genetic Algorithm Output for Case 1

Generation	Best f(x)	Mean f(x)	Stall Generations
1	50	Inf	0
2	50	Inf	1
3	50	Inf	2
4	50	Inf	3
5	50	Inf	4
6	50	Inf	5

Generation	Best f(x)	Mean f(x)	Stall Generations
7	50	50	6
8	50	Inf	7
9	50	Inf	8
10	50	Inf	9
11	50	Inf	10
12	50	Inf	11
13	50	53	12
14	50	55	13
15	50	Inf	14
16	50	53	15
17	50	Inf	16
18	50	Inf	17
19	50	Inf	18
20	50	Inf	19
21	50	Inf	20
22	50	Inf	21
23	50	Inf	22
24	50	Inf	23
25	50	Inf	24
26	50	51	25
27	50	Inf	26
28	50	Inf	27
29	50	Inf	28
30	50	Inf	29
31	50	55	30
32	50	Inf	31
33	50	Inf	32
34	50	51	33
35	50	Inf	34
36	50	Inf	35
37	50	52	36
38	50	Inf	37
39	50	50	38
40	50	50	39
41	50	Inf	40
42	50	50	41
43	50	Inf	42
44	50	Inf	43
45	50	Inf	44
46	50	Inf	45
47	50	Inf	46
48	50	Inf	47
49	50	Inf	48
50	50	Inf	49
51	50	Inf	50
52	50	53	51

Tables 5.3 - 5.11 display the generation number, best fitness value (Best f(x)), and mean fitness value (Mean f(x)) for each case tested on the 6-bus system. The stall generation shown in Table 5.3 counts the number of times the same best fitness value is repeated. The algorithm always exits after displaying the results when the stall generation count becomes 50. The stall generation is shown only in Table 5.3 in order to show how it acts as a stopping criterion for the algorithm when the same minimum fitness value appears in at least 50 generations continuously.

The other outputs from the genetic algorithm like different meter locations, meter arrangement, total number of meters, minimum fitness values and profile time were tabulated in Table 5.12 along with results from Cases 2a – 5c.

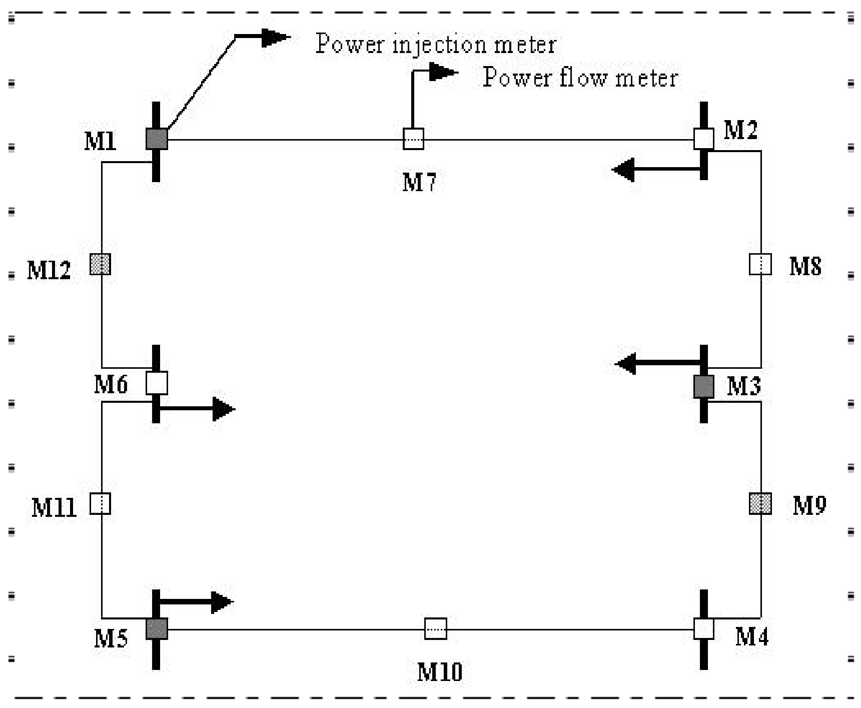

Figure 5.3 Meter placements on 6-bus test system for case 1

Figure 5.3 shows the 5 meters, M1, M3, M5, M9 and M12 locations (output of the genetic algorithm) on the system. The darkened meters represent the meters required and the meters in white represent the ones not required.

5.1.3 *CASE 2: Minimum number of meters with emphasis on type of meter*

5.1.3.1 CASE 2a: Minimum number of flow meters

Weight applied to the number of flow meters (P1) was higher that the weight applied to the number of injection meters (P2). Weights for all other constraints were 0. In case 2a, due to the higher weight applied to number of flow meters, the fitness function (FF) was minimized so that the output of the meter placement scheme has fewer numbers of flow meters than injection meters.

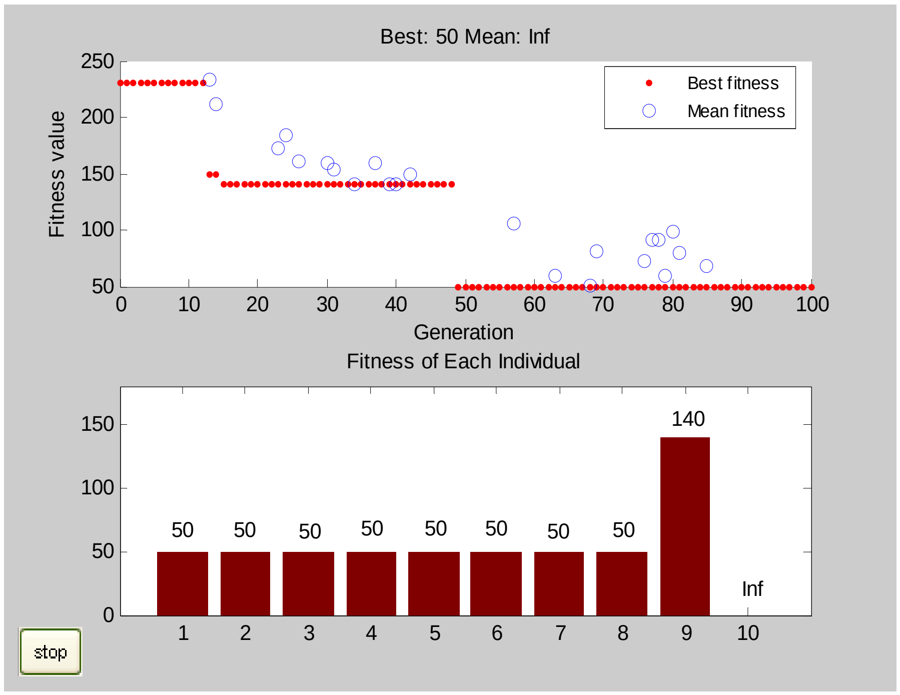

Figure 5.4 MATLAB Plot for Genetic Algorithm – case 2a

The explanation about the plots is given in Section 5.1.1. Figure 5.4 is the MATLAB plot showing the output of the Genetic Algorithm for case 2a. The best fitness value from plot 2 is 50.

Table 5.4 Genetic Algorithm Output for Case2a

Generation	Best f(x)	Mean f(x)
1	230	Inf
2	230	Inf
3	230	Inf
4	230	Inf
5	230	Inf
6	230	Inf
7	230	Inf
8	230	Inf
9	230	Inf
10	230	Inf
11	230	Inf
12	230	Inf
13	150	233
14	150	212
15	140	Inf
16	140	Inf
17	140	Inf
18	140	Inf
19	140	Inf
20	140	Inf
21	140	Inf
22	140	Inf
23	140	173
24	140	184
25	140	Inf
26	140	161
27	140	Inf
28	140	Inf
29	140	Inf
30	140	160
31	140	153
32	140	Inf
33	140	Inf
34	140	141
35	140	Inf
36	140	Inf
37	140	160
38	140	Inf
39	140	140
40	140	140
41	140	Inf
42	140	149
43	140	Inf
44	140	Inf
45	140	Inf
46	140	Inf
47	140	Inf
48	140	Inf
49	50	Inf
50	50	Inf
51	50	Inf
52	50	Inf
53	50	Inf
54	50	Inf
55	50	Inf
56	50	Inf
57	50	106
58	50	Inf
59	50	Inf
60	50	Inf
61	50	Inf
62	50	Inf
63	50	60
64	50	Inf
65	50	Inf
66	50	Inf
67	50	Inf
68	50	51
69	50	81
70	50	Inf
71	50	Inf
72	50	Inf
73	50	Inf
74	50	Inf
75	50	Inf
76	50	72
77	50	92
78	50	92
79	50	59
80	50	99
81	50	79
82	50	Inf
83	50	Inf
84	50	Inf
85	50	68
86	50	Inf

Generation	Best f(x)	Mean f(x)
87	50	Inf
88	50	Inf
89	50	Inf
90	50	Inf
91	50	Inf
92	50	Inf
93	50	Inf

Generation	Best f(x)	Mean f(x)
94	50	Inf
95	50	Inf
96	50	Inf
97	50	Inf
98	50	Inf
99	50	Inf
100	50	Inf

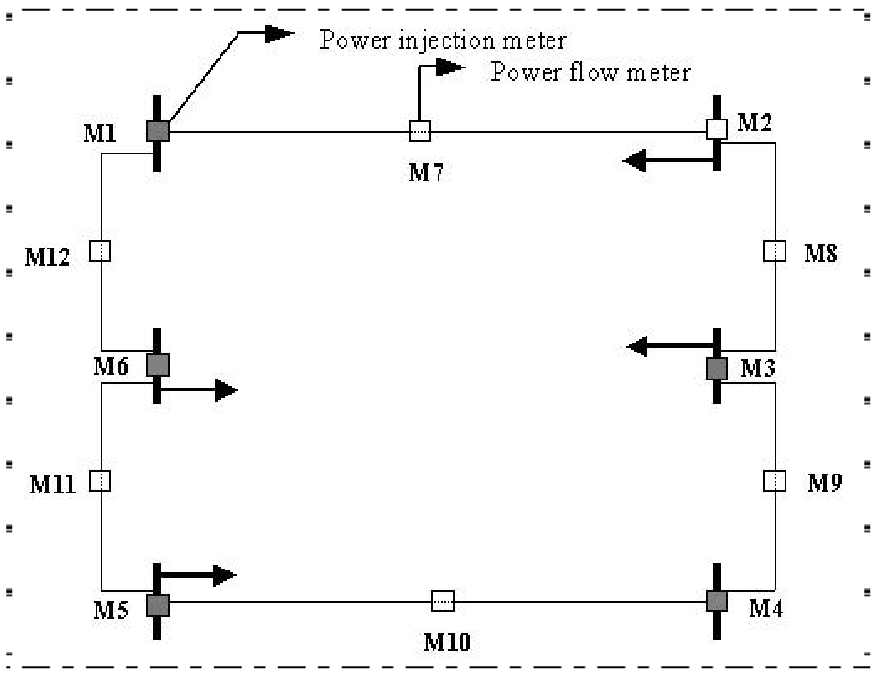

Figure 5.5 Meter placements on 6-bus test system for case 2a

Figure 5.5 shows the 5 meters, M1, M3, M4, M5 and M6 locations (according to the algorithm) on the system. The darkened meters represent the meters required and the meters in white represent the ones not required.

5.1.3.2 CASE 2b: Minimum number of injection meters

Weight applied to the number of injection meters (P2) was higher than the weight applied to the number of flow meters (P1). Weights for all other constraints were 0. In case 2b, due to the higher weight applied to number of injection meters, the fitness function (FF) was minimized so that the output of the meter placement scheme has fewer numbers of injection meters than flow meters.

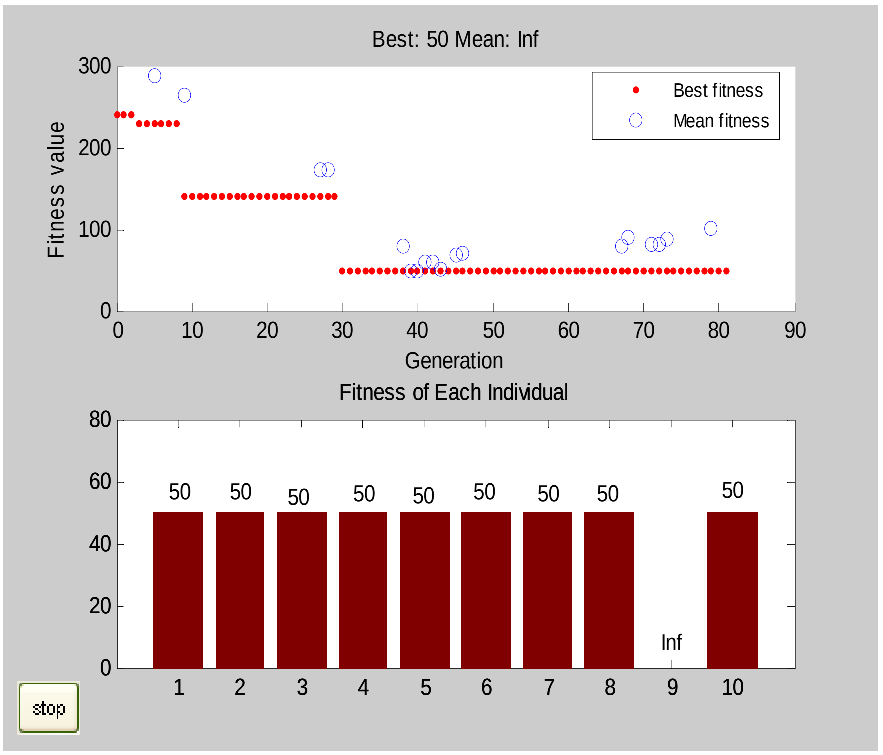

Figure 5.6 Plot for Genetic Algorithm – case 2b

Figure 5.6 is the MATLAB plot showing the output of the Genetic Algorithm for case 2b. The best fitness value from plot 2 is 50. The solution in Table 5.12 shows the presence of only flow meters in the meter arrangement.

Table 5.5 Genetic Algorithm Output for Case 2b

Generation	Best f(x)	Mean f(x)
1	240	Inf
2	240	Inf
3	230	Inf
4	230	Inf
5	230	287
6	230	Inf
7	230	Inf
8	230	Inf
9	140	264
10	140	Inf
11	140	Inf
12	140	Inf
13	140	Inf
14	140	Inf
15	140	Inf
16	140	Inf
17	140	Inf
18	140	Inf
19	140	Inf
20	140	Inf
21	140	Inf
22	140	Inf
23	140	Inf
24	140	Inf
25	140	Inf
26	140	Inf
27	140	172
28	140	173
29	140	Inf
30	50	Inf
31	50	Inf
32	50	Inf
33	50	Inf
34	50	Inf
35	50	Inf
36	50	Inf
37	50	Inf
38	50	80
39	50	50
40	50	50
41	50	60
42	50	59
43	50	51
44	50	Inf
45	50	69
46	50	71
47	50	Inf
48	50	Inf
49	50	Inf
50	50	Inf
51	50	Inf
52	50	Inf
53	50	Inf
54	50	Inf
55	50	Inf
56	50	Inf
57	50	Inf
58	50	Inf
59	50	Inf
60	50	Inf
61	50	Inf
62	50	Inf
63	50	Inf
64	50	Inf
65	50	Inf
66	50	Inf
67	50	80
68	50	90
69	50	Inf
70	50	Inf
71	50	81
72	50	81
73	50	88
74	50	Inf
75	50	Inf
76	50	Inf
77	50	Inf
78	50	Inf
79	50	101
80	50	Inf
81	50	Inf

5.1.4 CASE 3: Minimum number of meters with emphasis to eliminate critical measurements

In this case a higher priority is given to minimizing the number of critical measurements for the 6-bus system. Therefore the weight applied to the critical measurements is high. The meter arrangement scheme output of the Genetic Algorithm has meters placed in locations such that all the critical measurements are measured.

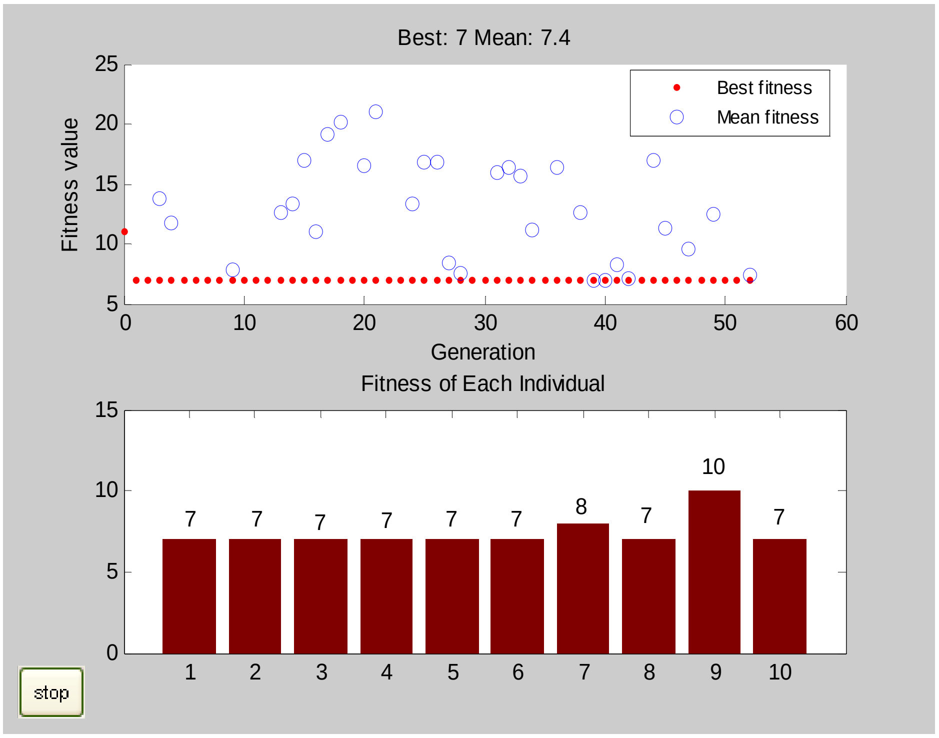

Figure 5.7 MATLAB Plot for Genetic Algorithm – case 3

Figure 5.7 is the MATLAB plot showing the output of the Genetic Algorithm for case 3. The best fitness value from plot 2 is 7.

Table 5.6 Genetic Algorithm Output for Case 3

Generation	Best f(x)	Mean f(x)
1	7	Inf
2	7	Inf
3	7	13.7
4	7	11.8
5	7	Inf
6	7	Inf
7	7	Inf
8	7	Inf
9	7	7.8
10	7	Inf
11	7	Inf
12	7	Inf
13	7	12.6
14	7	13.3
15	7	16.9
16	7	11
17	7	19.2
18	7	20.1
19	7	Inf
20	7	16.5
21	7	21
22	7	Inf
23	7	Inf
24	7	13.4
25	7	16.8
26	7	16.8
27	7	8.4
28	7	7.5
29	7	Inf
30	7	Inf
31	7	15.9
32	7	16.4
33	7	15.6
34	7	11.2
35	7	Inf
36	7	16.4
37	7	Inf
38	7	12.6
39	7	7
40	7	7
41	7	8.2
42	7	7.1
43	7	Inf
44	7	16.9
45	7	11.3
46	7	Inf
47	7	9.5
48	7	Inf
49	7	12.5
50	7	Inf
51	7	Inf
52	7	7.4

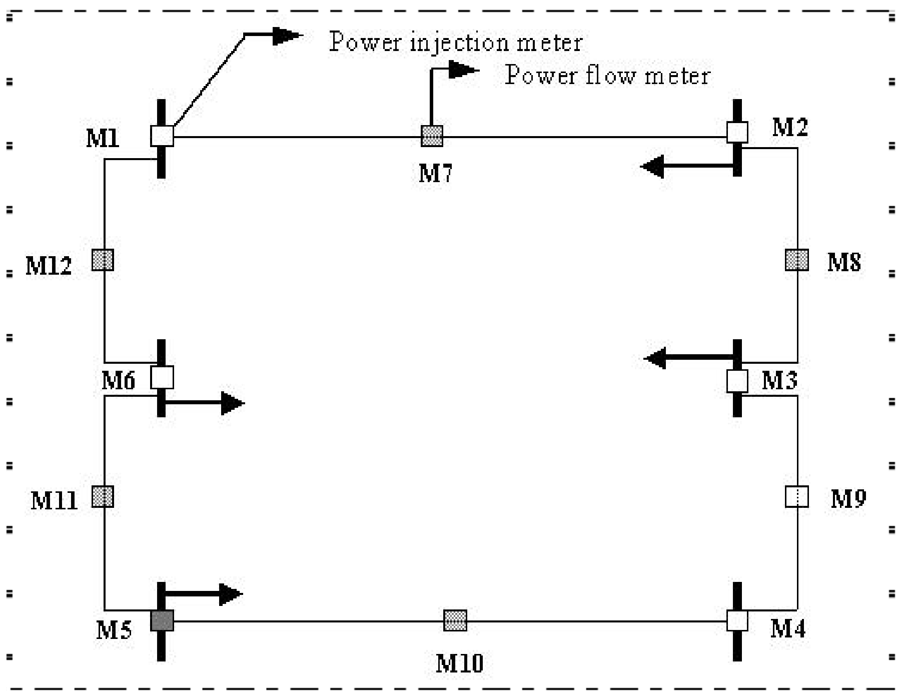

Figure 5.8 Meter placements on 6-bus test system for case 3

Figure 5.8 shows the 6 meters, M5, M7, M8, M10, M11 and M12 locations (according to the algorithm) on the system.

5.1.5 CASE 4: Minimum number of meters with emphasis to perform first order contingency analysis

5.1.5.1 CASE 4a: Loss of one flow meter

Higher weight was applied to the first order contingency analysis. Also the value of the probability of loss of flow meter was assumed to be higher than that of the injection meter. Weight was not applied to the second order contingency analysis. Therefore the output from second order contingency analysis algorithm to the fitness function was zero. Thus the GA performed first order contingency analysis on all meter arrangement schemes considered. Since the weight applied to number of flow meters was higher than the number of injection meters as mentioned in Table 5.1, the output of the GA has lower number of flow meters and more injection meters. Also the loss of any of the flow meter in the output meter arrangement would not have affected system observability.

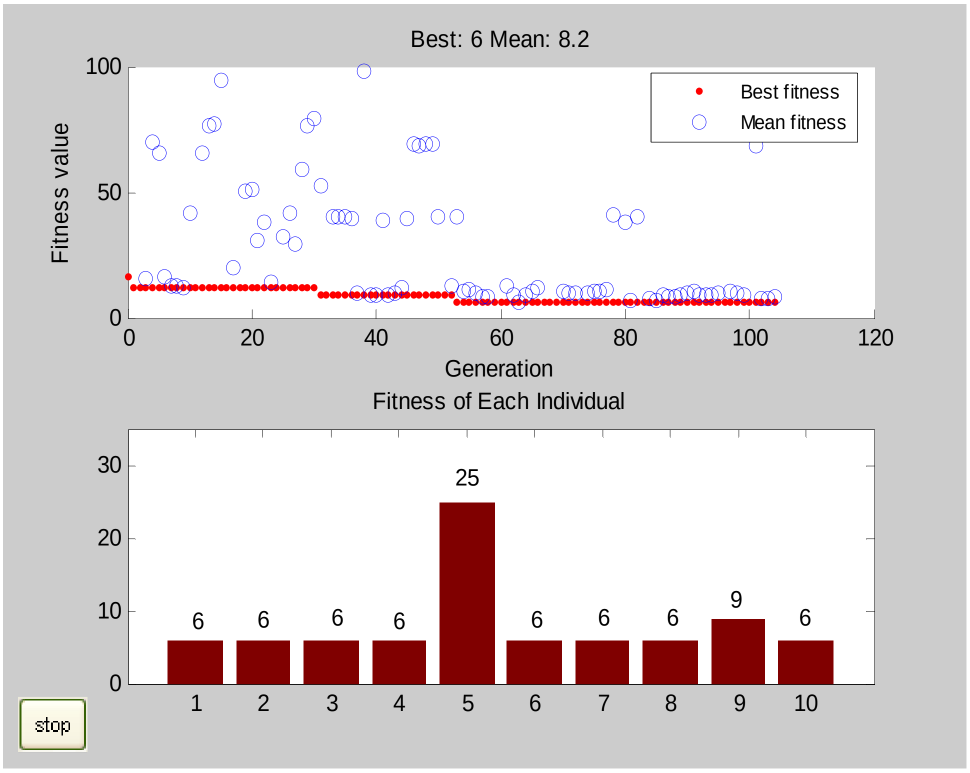

Figure 5.9 MATLAB Plot for Genetic Algorithm – case 4a

Figure 5.9 is the MATLAB plot showing the output of the Genetic Algorithm for case 4a.

The solution in Table 5.12 shows the meter arrangement having only injection meters and no flow meters as high weight was applied to minimize the number of flow meters. Due to the absence of flow meters in the meter arrangement the injection meters maintain system observability.

Table 5.7 Genetic Algorithm Output for Case 4a

Generation	Best f(x)	Mean f(x)
1	12	Inf
2	12	Inf
3	12	15.66
4	12	69.71
5	12	65.47
6	12	16.43
7	12	12.9
8	12	12.9
9	12	12.3
10	12	42.02
11	12	Inf
12	12	65.88
13	12	76.62
14	12	77.22
15	12	94.43
16	12	Inf
17	12	19.85
18	12	Inf
19	12	50.48
20	12	51.36
21	12	31.11
22	12	37.83
23	12	14
24	12	Inf
25	12	32.55
26	12	41.85
27	12	29.61
28	12	59.08
29	12	76.44
30	12	79.18
31	9	52.2
32	9	Inf
33	9	40.52
34	9	39.94
35	9	40.24
36	9	39.64
37	9	9.9
38	9	98.01
39	9	9
40	9	9
41	9	38.65
42	9	9.3
43	9	9.8
44	9	12.01
45	9	39.75
46	9	69.1
47	9	68.7
48	9	69.5
49	9	69.5
50	9	40.25
51	9	Inf
52	9	12.66
53	6	40.29
54	6	10.4
55	6	11.1
56	6	9.9
57	6	8.3
58	6	8
59	6	Inf
60	6	Inf
61	6	12.8
62	6	9.1
63	6	6.4
64	6	8.7
65	6	10.8
66	6	12.1
67	6	Inf
68	6	Inf
69	6	Inf
70	6	10.46
71	6	10.1
72	6	10.1
73	6	Inf
74	6	10.1
75	6	10.4
76	6	10.6

Generation	Best f(x)	Mean f(x)
77	6	11.4
78	6	41.08
79	6	Inf
80	6	37.95
81	6	7.1
82	6	40.06
83	6	Inf
84	6	7.9
85	6	6.6
86	6	8.725
87	6	8.3
88	6	8.6
89	6	9.4
90	6	10
91	6	10.5

Generation	Best f(x)	Mean f(x)
92	6	8.7
93	6	8.7
94	6	9
95	6	9.8
96	6	Inf
97	6	10.6
98	6	9.5
99	6	9
100	6	Inf
101	6	68.3
102	6	7.6
103	6	7.9
104	6	8.2

5.1.5.2 CASE 4b: Loss of one injection meter

Higher weight was applied to the first order contingency analysis. Also the value of the probability of loss of injection meter was assumed to be higher than that of the flow meter. Weight was not applied to the second order contingency analysis. The output of the GA has lower number of injection meters and more flow meters due to high weight applied to minimize injection meters.

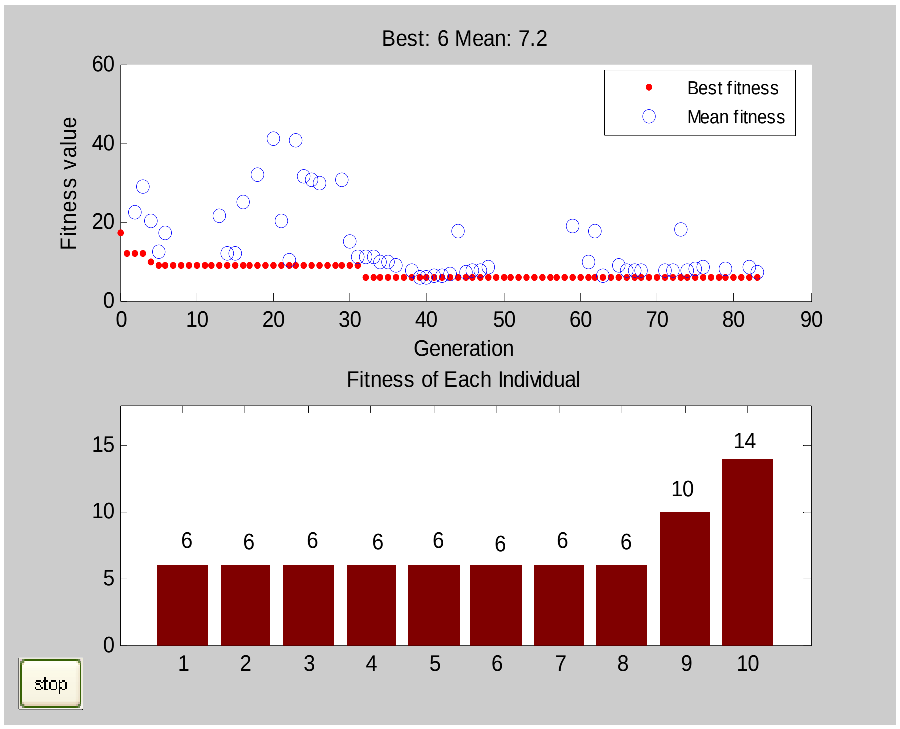

Figure 5.10 MATLAB Plot for Genetic Algorithm – case 4b

Figure 5.10 is the MATLAB plot showing the output of the Genetic Algorithm for case 4b. The solution shows the meter arrangement scheme having only flow meters as high weight was applied to minimize the number of injection meters to compensate the loss of any injection meter, and maintain system observability.

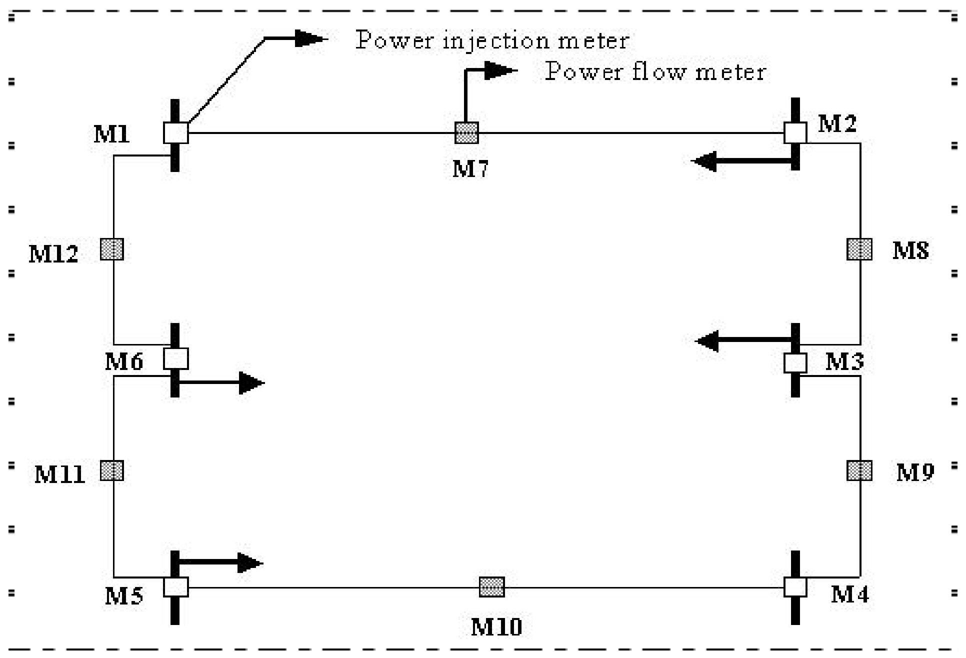

Figure 5.11 Meter placements on 6-bus test system for case 4b

Figure 5.11 shows the 6 meters, M7, M8, M9, M10, M11 and M12 locations (according to the algorithm) on the system.

Table 5.8 Genetic Algorithm Output for Case 4b

Generation	Best f(x)	Mean f(x)
1	12	Inf
2	12	22.36
3	12	28.76
4	10	20.23
5	9	12.2
6	9	17.23
7	9	Inf
8	9	Inf
9	9	Inf
10	9	Inf
11	9	Inf
12	9	Inf
13	9	21.45
14	9	12.1
15	9	11.8
16	9	25.05
17	9	Inf
18	9	32.01
19	9	Inf
20	9	41.17
21	9	20.26
22	9	10.1
23	9	40.58
24	9	31.31
25	9	30.51
26	9	29.93
27	9	Inf
28	9	Inf
29	9	30.7
30	9	14.93

Generation	Best f(x)	Mean f(x)
31	9	11.1
32	6	11
33	6	11.2
34	6	9.6
35	6	9.9
36	6	9
37	6	Inf
38	6	7.6
39	6	6
40	6	6
41	6	6.4
42	6	6.3
43	6	6.9
44	6	17.66
45	6	7.3
46	6	7.7
47	6	7.7
48	6	8.6
49	6	Inf
50	6	Inf
51	6	Inf
52	6	Inf
53	6	Inf
54	6	Inf
55	6	Inf
56	6	Inf
57	6	Inf

Generation	Best f(x)	Mean f(x)
58	6	Inf
59	6	19.06
60	6	Inf
61	6	9.7
62	6	17.56
63	6	6.4
64	6	Inf
65	6	9.1
66	6	7.8
67	6	7.6
68	6	7.6
69	6	Inf
70	6	Inf
71	6	7.5
72	6	7.5
73	6	17.96
74	6	7.4
75	6	8
76	6	8.6
77	6	Inf
78	6	Inf
79	6	7.9
80	6	Inf
81	6	Inf
82	6	8.3
83	6	7.2

5.1.6 CASE 5: Minimum number of meters with emphasis to perform second order contingency analysis

5.1.6.1 CASE 5a: Loss of two flow meters

Higher weight was applied to the second order contingency analysis. Also the value of the probability of loss of flow meter was assumed to be higher than that of the injection meter. Thus the GA found a meter arrangement, in which the loss of any two flow meters will continue to maintain system observability.

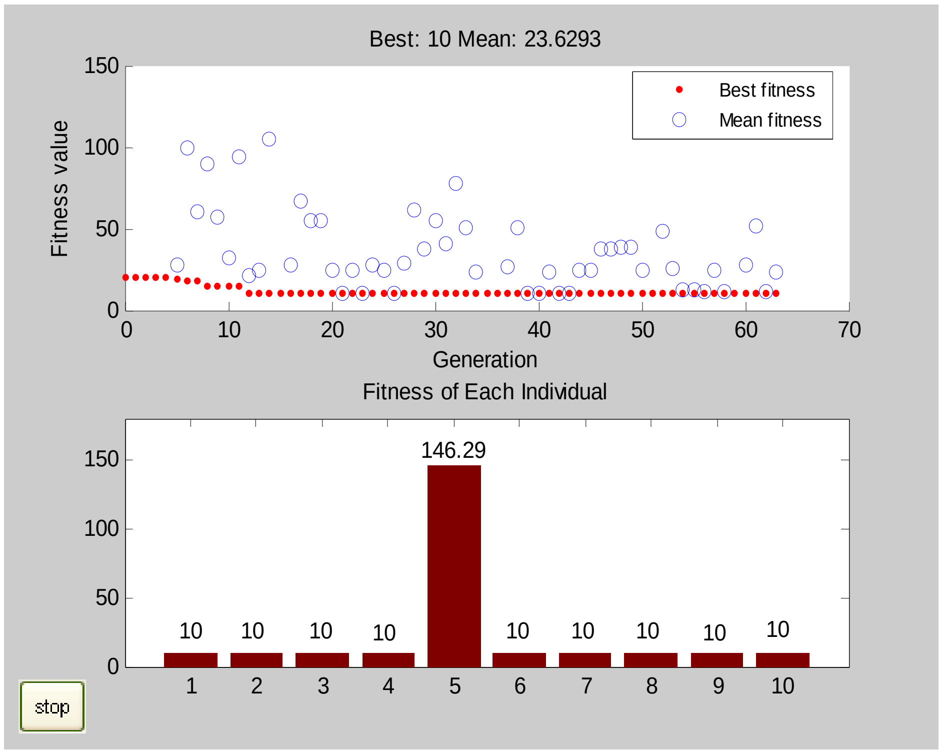

Figure 5.12 MATLAB Plot for Genetic Algorithm – case 5a

Figure 5.12 is the MATLAB plot showing the output of the Genetic Algorithm for case 5a.

Table 5.9 Genetic Algorithm Output for Case 5a

Generation	Best f(x)	Mean F(x)
1	20	Inf
2	20	Inf
3	20	Inf
4	20	Inf
5	18.59	28.05
6	17.69	99.55
7	17.69	60.17
8	15	89.2
9	14.69	57.46
10	14.69	32.23
11	14.69	93.49
12	10	21.63
13	10	24.47
14	10	105.1
15	10	Inf
16	10	27.86
17	10	66.72
18	10	54.79
19	10	54.74
20	10	24.72
21	10	10.84
22	10	24.8

Generation	Best f(x)	Mean F(x)
23	10	10.84
24	10	27.51
25	10	24.31
26	10	10.72
27	10	28.96
28	10	61.4
29	10	37.69
30	10	54.51
31	10	41.26
32	10	77.77
33	10	50.89
34	10	23.63
35	10	Inf
36	10	Inf
37	10	26.94
38	10	50.83
39	10	10
40	10	10
41	10	23.6
42	10	10.32
43	10	10.5

Generation	Best f(x)	Mean F(x)
44	10	24.19
45	10	24.49
46	10	37.7
47	10	37.6
48	10	38.1
49	10	38.3
50	10	24.97
51	10	Inf
52	10	48.54
53	10	25.66
54	10	12.55
55	10	12.15
56	10	11.3
57	10	24.32
58	10	11.27
59	10	Inf
60	10	27.75
61	10	52.15
62	10	11.02
63	10	23.63

5.1.6.2 CASE 5b: Loss of two injection meters

Higher weight was applied to the second order contingency analysis. Also the value of the probability of loss of injection meter was assumed to be higher than that of the flow meter.

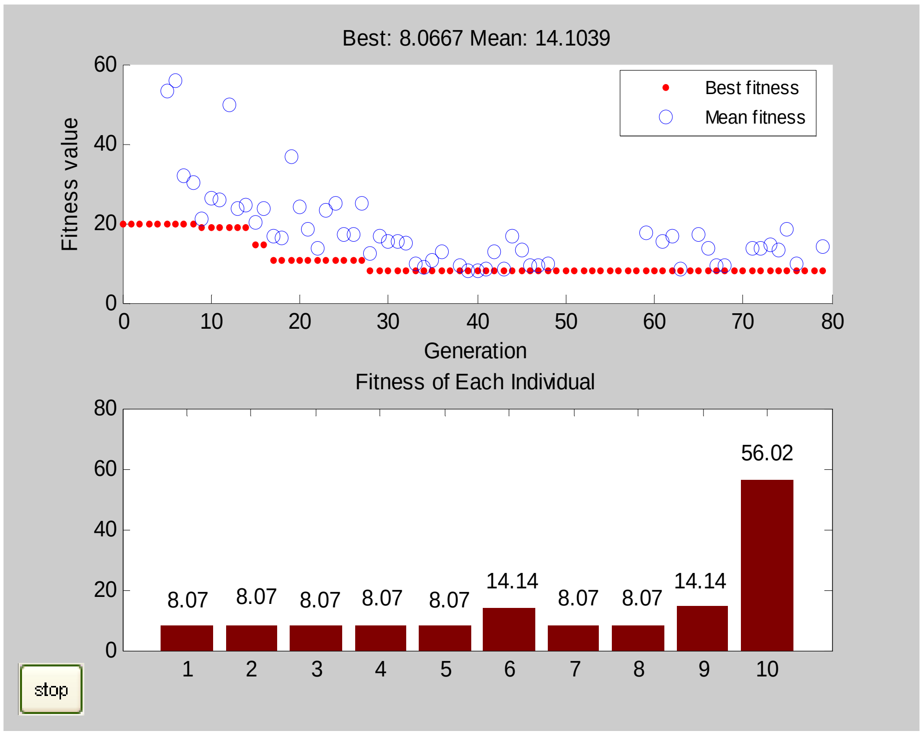

Figure 5.13 MATLAB Plot for Genetic Algorithm – case 5b

Figure 5.13 is the MATLAB plot showing the output of the Genetic Algorithm for case 5b. The best fitness value from plot 2 is 8.07. Due to the high weight applied to minimize number of injection meters the output has all the flow meters and no injection meters.

Table 5.10 Genetic Algorithm Output for Case 5b

Generation	Best f(x)	Mean f(x)	Generation	Best f(x)	Mean f(x)
1	20	Inf	9	19	21.19
2	20	Inf	10	19	26.48
3	20	Inf	11	19	25.74
4	20	Inf	12	19	49.94
5	20	53.46	13	19	23.85
6	19.98	56.06	14	19	24.67
7	19.98	31.74	15	14.41	20.43
8	19.98	30.06	16	14.41	23.58

Generation	Best f(x)	Mean f(x)
17	10.83	16.6
18	10.83	16.18
19	10.83	36.75
20	10.83	24.3
21	10.83	18.39
22	10.83	13.72
23	10.83	23.1
24	10.83	25.1
25	10.83	17.32
26	10.83	17.02
27	10.83	24.91
28	8.067	12.39
29	8.067	16.69
30	8.067	15.34
31	8.067	15.53
32	8.067	15.13
33	8.067	9.957
34	8.067	8.743
35	8.067	10.62
36	8.067	12.63
37	8.067	Inf
38	8.067	9.295
39	8.067	8.067
40	8.067	8.067
41	8.067	8.343
42	8.067	12.86
43	8.067	8.648
44	8.067	16.58
45	8.067	13.48
46	8.067	9.2
47	8.067	9.19
48	8.067	9.771
49	8.067	Inf
50	8.067	Inf
51	8.067	Inf
52	8.067	Inf
53	8.067	Inf
54	8.067	Inf
55	8.067	Inf
56	8.067	Inf
57	8.067	Inf
58	8.067	Inf
59	8.067	17.81
60	8.067	Inf
61	8.067	15.23
62	8.067	16.59
63	8.067	8.343
64	8.067	Inf
65	8.067	16.97
66	8.067	13.74
67	8.067	9.308
68	8.067	9.253
69	8.067	Inf
70	8.067	Inf
71	8.067	13.81
72	8.067	13.81
73	8.067	14.51
74	8.067	13.45
75	8.067	18.3
76	8.067	9.781
77	8.067	Inf
78	8.067	Inf
79	8.067	14.1

5.1.6.3 CASE 5c: Loss of one flow and one injection meter

Weight was applied to all the constraints (number of flow meters, number of injection meters, critical measurements, and first and second order contingency analyses depending on the type of meters lost). Higher priority was given to performing second order contingency analysis on all meter arrangements schemes considered. The probability of loss of one flow meter or one injection meter was almost equal. This enabled the GA to also consider the loss of flow meter and injection meter equally. The final meter arrangement scheme has enough meters so that loss of any two flow meters or two injection meters or one flow and one injection meter will not affect system observability.

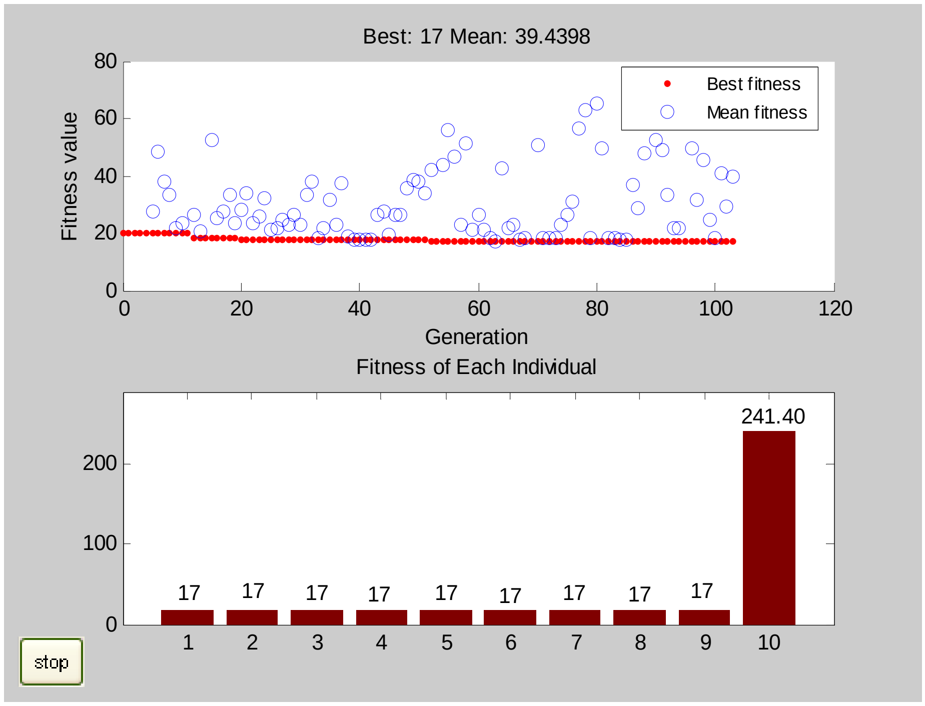

Figure 5.14 MATLAB Plot for Genetic Algorithm – case 5c

Figure 5.14 is the MATLAB plot showing the output of the Genetic Algorithm for case 5c. The meter arrangement scheme selected has enough number of flow as well as injection meters. The number of meters required has increased to 8 due to the increase in the number of constraints applied to the fitness function.

Table 5.11 Genetic Algorithm Output for Case 5c

Generation	Best f(x)	Mean f(x)
1	20	Inf
2	20	Inf
3	20	Inf
4	20	Inf
5	20	27.32
6	20	48.55
7	20	38.25
8	20	33.46
9	20	21.6
10	20	23.68
11	20	Inf
12	18	26.37
13	18	20.3
14	18	Inf
15	18	52.37
16	18	25.23
17	18	27.39
18	18	33.52
19	18	23.66
20	17.44	28.22
21	17.44	34.15
22	17.44	23.54
23	17.44	25.59
24	17.44	32.32
25	17.44	20.9
26	17.44	21.49
27	17.44	24.42
28	17.44	22.87
29	17.44	26.59
30	17.44	22.63
31	17.44	33.32
32	17.44	37.84
33	17.44	18.17
34	17.44	22.02
35	17.44	31.84
36	17.44	22.87
37	17.44	37.47
38	17.44	18.88
39	17.44	17.44
40	17.44	17.44
41	17.44	17.5
42	17.44	17.5
43	17.44	26.48
44	17.44	27.58
45	17.44	19.52
46	17.44	26.6
47	17.44	26.6
48	17.44	35.63
49	17.44	38.29
50	17.44	38.08
51	17.44	33.95
52	17	42.03
53	17	Inf
54	17	43.52
55	17	56.22
56	17	46.43
57	17	22.94
58	17	51.43
59	17	21.21
60	17	26.45
61	17	21.06
62	17	18.53
63	17	17.1
64	17	42.88
65	17	21.87
66	17	23.15
67	17	17.4
68	17	18.09
69	17	Inf
70	17	50.99
71	17	18.1
72	17	18.1
73	17	18.32
74	17	22.69
75	17	26.35
76	17	31.17
77	17	56.75
78	17	62.99
79	17	18.49
80	17	65.37
81	17	49.84
82	17	18.5
83	17	18.2
84	17	17.4
85	17	17.66
86	17	36.81

Generation	Best f(x)	Mean f(x)
87	17	28.44
88	17	48.06
89	17	59.1
90	17	52.38
91	17	48.89
92	17	33.56
93	17	21.96
94	17	21.96
95	17	69.47

Generation	Best f(x)	Mean f(x)
96	17	49.75
97	17	31.56
98	17	45.6
99	17	24.58
100	17	18.13
101	17	40.72
102	17	29.25
103	17	39.44

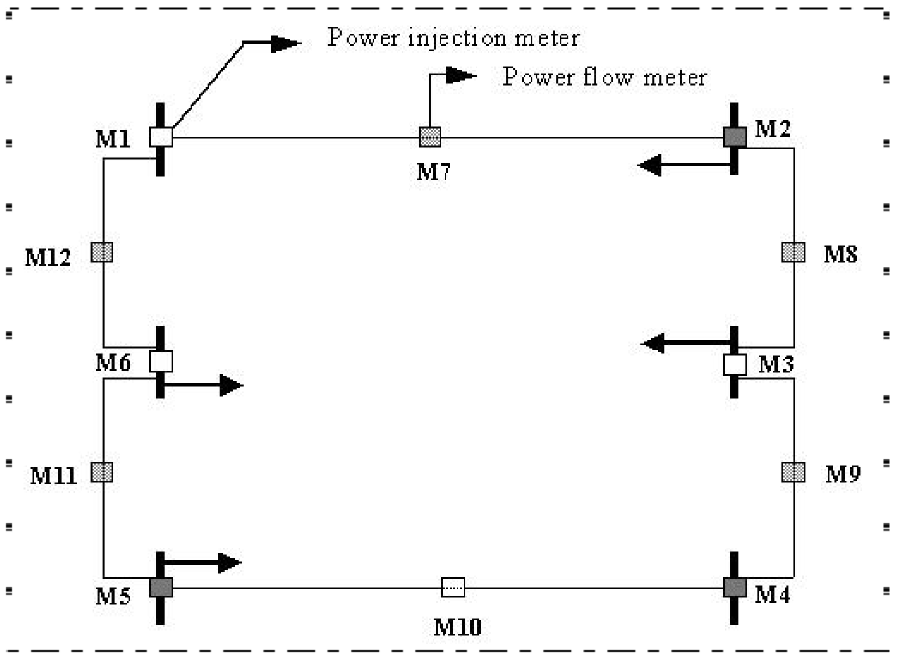

Figure 5.15 Meter placement on 6-bus test system for case 5c

Figure 5.15 shows the 8 meters, M2, M4, M5, M7, M8, M9, M11 and M12 locations (according to the algorithm) on the system.

The output meter arrangement schemes for all the cases from the Genetic Algorithm indicating the meter status is shown in Table 5.12. ‘1’ indicates that the meter is required and ‘0’ indicates that the meter is not required. The first six values are the status of the 6 injection meters on the buses and the rest is the status of the flow meters on the branches. The minimum fitness value of the final solution for each case was found after evaluating all the individuals at the end of all the generations and is also shown in Table 5.12.

The table below also indicates the meters, their location and the total number of meters in the 6-bus test system.

Table 5.12   Solution for all Cases

Case No.	Meter Arrangement	Meter Location	Total No. of Meters	Min. FF Value	Profile Time (s)
1	101010001001	M1, M3, M5, M9, M12	5	50	3.578
2a	101111000000	M1, M3, M4, M5, M6	5	50	5.594
2b	000000111110	M7, M8, M9, M10, M11	5	50	4.532
3	000010110111	M5, M7, M8, M10, M11, M12	6	7	3.953
4a	111111000000	M1, M2, M3, M4, M5, M6	6	6	8.344
4b	000000111111	M7, M8, M9, M10, M11, M12	6	6	6.484
5a	111111000100	M1, M2, M3, M4, M5, M6, M10	7	10	11.453
5b	000000111111	M7, M8, M9, M10, M11, M12	6	8.07	13.031
5c	010110111011	M2, M4, M5, M7, M8, M9, M11, M12	8	17	20.641

Table 5.13 summarizes the results achieved by the genetic algorithm for different constraints applied to the fitness function. The results for the three test systems (3-bus system, 6-bus system and 18-bus system) are also compared. As the size of the test system becomes larger (the number of buses and branches), the number of possible meter arrangements increases exponentially. This increases the computatation time required for the algorithm to find the optimal solution satisfying the given constraints. In each case,

as the number of constraints applied to the test system increases, the number of meters required in the solution also increase to maintain system observability. This can be clearly seen in the results for all the three test systems. The fitness function for the GA depends on the constraints considered and the weights applied to them. The location of the meters in each case can be found from figures 5.1, A.1, B.1 which show meters in all possible locations for the three test systems.

Table 5.13 Summary of Results

CASE		WEIGHTS	OUTPUT	3-BUS SYSTEM	6-BUS SYSTEM	18-BUS SYSTEM
Number	Type					
1	Min. number of meters with no constraints	P1 = 10, P2 = 10	Meter Arrangement	000011	101010001001	1001111011101001000100 0100010101011
			Meter Location	M5, M6	M1, M3, M5, M9, M12	M1, M4, M5, M6, M7, M9, M10, M11, M13, M16, M20, M24, M28, M30, M32, M34, M35
			Number of Meters	2	5	17
			Min. FF Value	20	50	170
			Profile Time (s)	3.297	3.578	5.750
2a	Min. number of meters with emphasis on min. number of flow meters	P1 = 100, P2 = 10	Meter Arrangement	101000	101111000000	10111111011101001100100 0100000101010
			Meter Location	M1, M3	M1, M3, M4, M5, M6	M1, M3, M4, M5, M6, M7, M9, M10, M11, M13, M16, M17, M20, M24, M30, M32, M34
			Number of Meters	2	5	17
			Min. FF Value	20	50	620
			Profile Time (s)	4.469	5.594	8.468
2b	Min. number of meters with emphasis on min. number of injection meters	P1 = 10, P2 = 100	Meter Arrangement	000011	000000111110	10001000000001000000111 0111111101111
			Meter Location	M5, M6	M7, M8, M9, M10, M11	M1, M5, M13, M20, M21, M22, M24, M25, M26, M27, M28, M29, M30, M32, M33, M34, M35
			Number of Meters	2	5	17
			Min. FF Value	20	50	440
			Profile Time (s)	2.656	4.532	7.453

CASE		WEIGHTS	OUTPUT	3-BUS SYSTEM	6-BUS SYSTEM	18-BUS SYSTEM
Number	Type					
3	Min. number of meters with emphasis on min. number of critical measurements	P1 = 1, P2 = 2, P3 = 10	Meter Arrangement	000111	000010110111	10001000000000000001111 111111111111
			Meter Location	M4, M5, M6	M5, M7, M8, M10, M11, M12	M1, M5, M20, M21, M22, M23, M24, M25, M26, M27, M28, M29, M30, M31, M32, M33, M34, M35
			Number of Meters	3	6	18
			Min. FF Value	3	7	20
			Profile Time (s)	3.235	3.953	10.047
4a	After FO contingency analysis (loss of 1 flow meter)	P1 = 4, P2 = 1, P3 = 1, P4 = 47.5, P5 = 2.5	Meter Arrangement	111000	111111000000	11111111111111010100000 000000001010
			Meter Location	M1, M2, M3	M1, M2, M3, M4, M5, M6	M1, M2, M3, M4, M5, M6, M7, M8, M9, M10, M11, M12, M13, M14, M16, M18, M32, M34
			Number of Meters	3	6	18
			Min. FF Value	3	6	25.16
			Profile Time (s)	3.859	8.344	30.343
4b	After FO contingency analysis (loss of 1 injection meter)	P1 = 1, P2 = 4, P3 = 1, P4 = 2.5, P5 = 47.5	Meter Arrangement	000111	000000111111	10001110000000010001110 011111111011
			Meter Location	M4, M5, M6	M7, M8, M9, M10, M11, M12	M1, M5, M6, M7, M16, M20, M21, M22, M25, M26, M27, M28, M29, M30, M31, M32, M34, M35
			Number of Meters	3	6	18
			Min. FF Value	3	6	34.15
			Profile Time (s)	3.641	6.484	12.328
5a	After SO contingency analysis (loss of 2 flow meters)	P1 = 4, P2 = 1, P3 = 1, P4 = 4.5,	Meter Arrangement	111000	111111000100	11111111111111011100000 000000011000

CASE		WEIGHTS	OUTPUT	3-BUS SYSTEM	6-BUS SYSTEM	18-BUS SYSTEM
Number	Type					
		P5 = 0.5, P6 = 100.44, P7 = 1.24, P8 = 11.16	Meter Location	M1, M2, M3	M1, M2, M3, M4, M5, M6, M10	M1, M2, M3, M4, M5, M6, M7, M8, M9, M10, M11, M12, M13, M14 M16, M17, M18, M31, M32
			Number of Meters	3	7	19
			Min. FF Value	5.48	10	28.95
			Profile Time (s)	4.938	11.453	514.282
5b	After SO contingency analysis (loss of 2 injection meters)	P1 = 1, P2 = 4, P3 = 1, P4 = 0.5, P5 = 4.5, P6 = 1.24, P7 = 100.44, P8 = 11.16	Meter Arrangement	000111	000000111111	00001101001001000011111 111111111111
			Meter Location	M4, M5, M6	M7, M8, M9, M10, M11, M12	M5, M6, M8, M11, M14, M19, M20, M21, M22, M23, M24, M25, M26, M27, M28, M29, M30, M31, M32, M33, M34, M35
			Number of Meters	3	6	22
			Min. FF Value	4.653	8.07	43.89
			Profile Time (s)	5.032	13.031	301.156
5c	After SO contingency analysis (loss of 1 flow meter & 1 injection meter)	P1 = 1, P2 = 4, P3 = 1, P4 = 1.5, P5 = 1.2, P6 = 31, P7 = 19.84, P8 = 24.8	Meter Arrangement	100110	010110111011	00001101001001001011111 111111111111
			Meter Location	M1, M4, M5	M2, M4, M5, M7, M8, M9, M11, M12	M5, M6, M8, M11, M14, M17, M19, M20, M21, M22, M23, M24, M25, M26, M27, M28, M29, M30, M31, M32, M33, M34, M35
			Number of Meters	4	8	23
			Min. FF Value	8	17	55.65
			Profile Time (s)	5.578	20.641	535.828

5.2 Summary

This chapter discusses in detail the results for the 6-bus (DDX) test system. It gives an explanation about the weights applied to each constraint and the output of the genetic algorithm in each case. The results for the 3-bus test system, 6-bus test system and the 18-bus test system are compared in Table 5.10. More results with figures and tables for the 3-bus system and the 18-bus system are given in the appendices A and B.

CHAPTER VI

CONCLUSIONS AND FUTURE WORK

6.1 Conclusions

The aim of this thesis was to develop an algorithm that provides an intelligent shipboard power system monitoring scheme. A genetic algorithm based meter placement scheme was developed for this purpose and the goal was met as presented in previous chapters and as shown in the results. The ship power system monitoring conditions are different from those of a terrestrial system. The shipboard power system involves additional constraints (like better system observability, reliability and higher order contingency analysis) which have to be taken into consideration when designing the SPS monitoring system.

The algorithm developed for monitoring a SPS was based on the inputs about the system topology, meter types considered and all possible locations of these meters. Two types of meters were considered; namely, power flow meters at the lines and the power injection meters at the buses. A fitness function was developed based on the constraints like minimum number of flow meters, minimum number of injection meters, critical measurements, first order contingency analysis and second order contingency analysis and the weights applied to each of these constraints. Weights are applied to the constraints in the fitness function based on the priority of the constraint to be minimized. Depending on the inputs and the constraints applied to it the GA evaluates different configurations of meter arrangements and identifies an intelligent solution or meter arrangement scheme that satisfies all the conditions. The GA based meter placement technique not only helps in planning of meter locations during SPS design, but also helps to predict meter locations that are required during operation to maintain system observability when certain meters have to be disconnected during maintenance or are lost due to damage.

The work presented in this thesis identifies the need for a reliable and efficient monitoring system for the SPS, introduces the development of a GA based meter placement scheme as an intelligent method for SPS monitoring and lays the groundwork for further improvements to the algorithm and future work in monitoring of SPS. The thesis also presents in detail three test systems used to test the working of the genetic algorithm based meter placement scheme and the successful results achieved therefrom. The thesis serves as a reference for further work in SPS monitoring. It also identifies some key areas where research can be further enhanced.

6.2 Future Work

There are a few areas in the algorithm that can be improved. The algorithms developed for power system monitoring in general have mostly looked at only two types of meters, the power injection meters at the buses and the power flow meters on the branches. The existing algorithm can be further developed to incorporate other types of meters like voltage meter and current meters and look at their impact on the fitness function and the additional constraints applied to it. Additionally temperature and pressure sensors on the system can also be considered.

Developing the genetic algorithm for systems containing vital and semi-vital loads would be a more realistic representation of the power system. The location of vital, semi-vital and non-vital loads will impose additional constraints on the fitness function and impact the meter placement scheme output. Future work can include development of algorithms using other techniques like Particle Swarm Optimization (PSO). The GA and PSO algorithms can be compared to study their efficiency, reliability and robustness.

REFERENCES

[1] A. Abur and A. G. Expósito, Power System State Estimation, Theory and Implementation, New York: Marcel Dekker, Inc 2004, pp. 59-98.

[2] M. Gen and R. Cheng, Genetic Algorithms and Engineering Design, New York: John Wiley & Sons, 1997, pp: 1-7.

[3] M. E. Baran, Jinxiang Zhu, Hongbo Zhu and K.E. Garren, "A meter placement method for state estimation," *IEEE Transactions on Power Systems*, vol.10, pp: 1704 -1710, Aug.1995.

[4] M. E. Baran, Jinxiang Zhu and A. W. Kelley, "Meter placement for real-time monitoring of distribution feeders," *IEEE Transactions on Power Systems*, vol 11, pp: 332 – 337, Feb. 1996.

[5] S. Maheshwarapu, "An effective root based algorithm for power system topological observability," *Proceedings of IEEE Region 10 International Conference on Global Connectivity in Energy, Computer, Communication and Control, TENCON '98.* vol: 2, pp: 596-600, 1998.

[6] A. Abur and F. H. Magnago, "Optimal meter placement for maintaining observability during single branch outages," *IEEE Transactions on Power Systems*, vol: 14, pp: 1273-1278, Nov 1999.

[7] F. H. Magnago and A. Abur, "A unified approach to robust meter placement against loss of measurements and branch outages," *IEEE Transactions on Power Systems*, vol: 15, pp: 945-949, Aug 2000.

[8] A. Abur and F. H. Magnago,"Optimal meter placement against contingencies," *Proceedings of IEEE Power Engineering Society Summer Meeting,* vol. 1, pp: 424-428, 2001.

[9] Qifeng Ding and A. Abur, "An improved measurement placement method against loss of multiple measurements and branches," *Proceedings of IEEE Power Engineering Society Winter Meeting*, vol 1, pp: 234- 237, 2002.

[10] B. Xu and A. Abur, "Observability analysis and measurement placement for systems with PMUs," *Proceedings of IEEE PES Power Systems Conference and Exposition,* vol.2, pp: 943- 946, Oct. 2004.

[11] A. B. Antonio, J. R. A. Torreao and M.B. Do Coutto Filho, "Meter placement for power system state estimation using simulated annealing," *Proceedings of IEEE Porto PowerTech*, vol. 3, p. 5, 2001.

[12] H. Mori and Y. Sone, "Tabu search based meter placement for topological observability in power system state estimation," *Proceedings of IEEE Transmission and Distribution Conference*, vol. 1, pp: 172-177, Apr 1999.

[13] H. Mori and M. A. Saito, "Hybrid approach of parallel tabu search and ordinal optimization to meter placement for improving topological observability," *Proceedings of International Conference on Power System Technology*, vol. 1, pp: 963- 968, Nov. 2004.

[14] H. Mori and S. Iida, "Application of a genetic algorithm to meter allocation in electric power systems," *Proceedings of International Joint Conference on Neural Networks*, *IJCNN '93-Nagoya,* vol 2, pp: 1594 – 1597, Oct. 1993.

[15] A. El-Zonkoly, "A novel algorithm for optimal meter placement to maintain network observability," *Proceedings of IEEE PES Power Systems Conference and Exposition,* vol.1, pp: 93- 99, Oct. 2004.

[16] J.C.S. de Souza, M.B. Do Coutto Filho, M.T. Schilling and C. de Capdeville, "Optimal metering systems for monitoring power networks under multiple topological scenarios," *IEEE Transactions on Power Systems,* vol. 20, pp: 1700 – 1708, Nov. 2005.

[17] D.E. Goldberg**,** Genetic Algorithms in Search, Optimization and Machine, Boston: Addison-Wesley Longman Publishing Co., Inc. 1989.

[18] Sandhya Sankar and Noel. N. Schulz, "Intelligent Placement of Meters / Sensors for Shipboard Power System Analysis," *Proceedings of IEEE Electric Ship Technologies Symposium, 2007. ESTS '07*, pp: 504 – 509, 21-23 May 2007.

[19] http://www-ccs.ucsd.edu/MATLAB/pdf_doc/optim/optim_tb.pdf.

[20] MATLAB - http://www.mathworks.com

[21] DD(X)_Notional_One_Line_Dwg1.pdf

[22] T.L. Baldwin and S.A. Lewis, "Distribution load flow methods for shipboard power systems," *IEEE Transactions on Industry Applications*, vol. 40, pp: 1183 – 1190, Sept.-Oct. 2004.

[23] A.J.F. Griffiths, J.H. Miller, D.T. Suzuki, R.C. Lewontin, and W.M. Gelbart, An Introduction to Genetic Analysis**.** New York: W.H. Freeman and Company, 2000.

APPENDIX A

TEST RESULTS FOR 3 - BUS SYSTEM

The algorithms were tested with the 3-bus system. System topology, meter types (power flow and power injection meters) and all possible meter locations were given as input to the algorithms. Figure A.1 shows the 3-bus test system considered to test the algorithms and the fitness function. The figure also shows all possible locations where meters can be placed in the system. The meters are numbered, starting with all the injection meters followed by the flow meters in the clockwise direction.

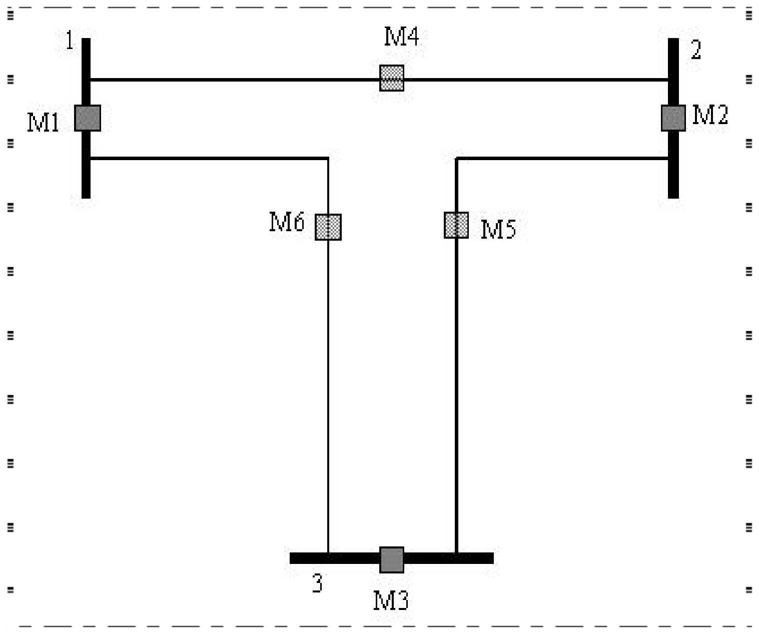

Figure A.1 3- Bus Test System with Meters

The system is tested for different cases as shown in the subsequent sections. In each case different weights were applied to each constraint in the fitness function depending on their priority. The following list shows the weight applied to each constraint in the FF.

Weight applied to number of flow meters, P1

Weight applied to number of injection meters, P2

Weight applied to critical measurements, P3

Generic weight factor for FO contingency, pen_first

Generic weight factor for SO contingency, pen_second

Fail probability for loss of 1 flow meter, prob_fail_flow

Fail probability for loss of 1 injection meter, prob_fail_inj

Weight applied to loss of 1 flow meter in FO contingency, P4 = pen_first × prob_fail_flow

Weight applied to loss of 1 injection meter in FO contingency, P5 = pen_first × prob_fail_inj

Weight applied to loss of 2 flow meter in SO contingency, P6 = pen_second × prob_fail_flow × prob_fail_flow

Weight applied to loss of 2 injection meter in SO contingency, P7 = pen_second × prob_fail_inj × prob_fail_inj

Weight applied to loss of 1 flow and 1 injection meter in SO contingency, P8 = pen_second × prob_fail_inj × prob_fail_flow

Table A.1 shows the value of the weights applied in each case.

Table A.1 Weights for Constraints

Case No.	P1	P2	P3	pen_first	pen_second	prob_fail_flow	prob_fail_inj	P4	P5	P6	P7	P8
1	10	10	0	0	0	0	0	0	0	0	0	0
2a	100	10	0	0	0	0	0	0	0	0	0	0
2b	10	100	0	0	0	0	0	0	0	0	0	0
3	1	2	10	0	0	0	0	0	0	0	0	0
4a	4	1	1	50	0	0.95	0.05	47.5	2.5	0	0	0
4b	1	4	1	50	0	0.05	0.95	2.5	47.5	0	0	0
5a	4	1	1	5	124	0.9	0.1	4.5	0.5	100.44	1.24	11.16
5b	1	4	1	5	124	0.1	0.9	0.5	4.5	1.24	100.44	11.16
5c	1	4	1	3	124	0.5	0.4	1.5	1.2	31.25	19.84	24.8

Table A.2 Solution for all Cases

Case No.	Meter Arrangement	Meter Location	Total No. of Meters	Min. FF Value	Profile Time (s)
1	000011	M5, M6	2	20	3.297
2a	101000	M1, M3	2	20	4.469
2b	000011	M5, M6	2	20	2.656
3	000111	M4, M5, M6	3	3	3.235
4a	111000	M1, M2, M3	3	3	3.859
4b	000111	M4, M5, M6	3	3	3.641
5a	111000	M1, M2, M3	3	5.48	4.938
5b	000111	M4, M5, M6	3	4.653	5.032
5c	100110	M1, M4, M5	4	8	5.578

A.1 CASE 1: Minimum number of meters without considering other factors

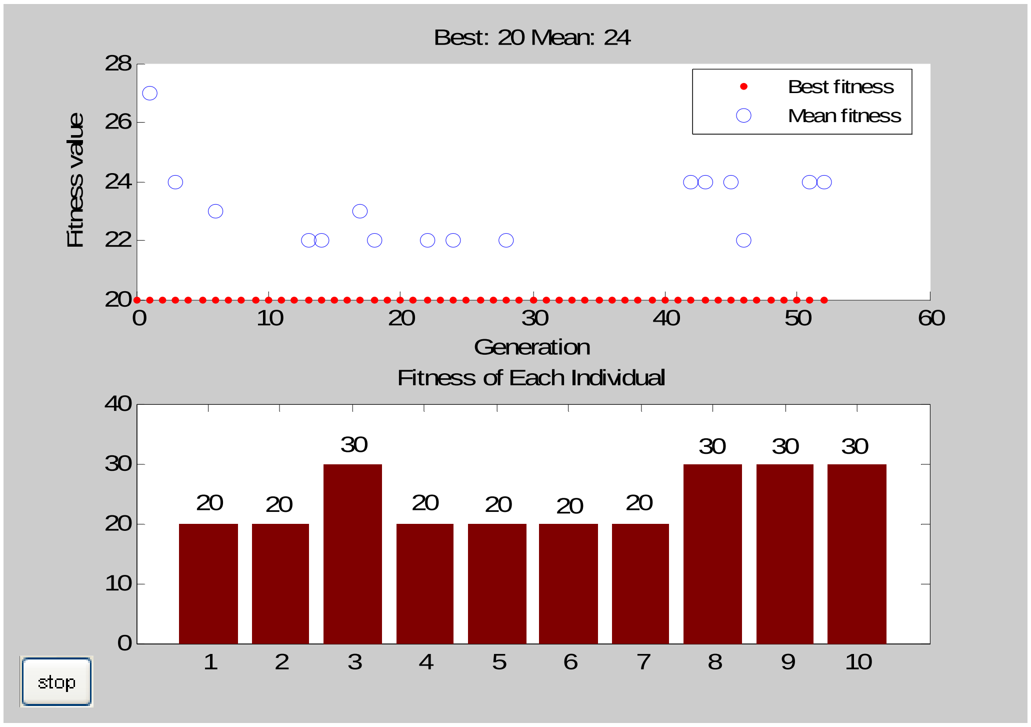

Figure A.2 MATLAB Plot of Fitness Function Values - Case 1

Table A.3 Genetic Algorithm Output for Case – 1

Generation	Best f(x)	Mean f(x)	Stall Generations	Generation	Best f(x)	Mean f(x)	Stall Generations
1	20	27	0	16	20	Inf	15
2	20	Inf	1	17	20	23	16
3	20	24	2	18	20	22	17
4	20	Inf	3	19	20	Inf	18
5	20	Inf	4	20	20	Inf	19
6	20	23	5	21	20	Inf	20
7	20	Inf	6	22	20	22	21
8	20	Inf	7	23	20	Inf	22
9	20	Inf	8	24	20	22	23
10	20	Inf	9	25	20	Inf	24
11	20	Inf	10	26	20	Inf	25
12	20	Inf	11	27	20	Inf	26
13	20	22	12	28	20	22	27
14	20	22	13	29	20	Inf	28
15	20	Inf	14	30	20	Inf	29

Generation	Best f(x)	Mean f(x)	Stall Generations
31	20	Inf	30
32	20	Inf	31
33	20	Inf	32
34	20	Inf	33
35	20	Inf	34
36	20	Inf	35
37	20	Inf	36
38	20	Inf	37
39	20	Inf	38
40	20	Inf	39
41	20	Inf	40
42	20	24	41

Generation	Best f(x)	Mean f(x)	Stall Generations
43	20	24	42
44	20	26	43
45	20	24	44
46	20	22	45
47	20	Inf	46
48	20	Inf	47
49	20	Inf	48
50	20	Inf	49
51	20	24	50
52	20	24	51

A.2 CASE 2: Minimum number of meters with emphasis on type of meter

A.2.1 CASE 2a: Minimum number of flow meters

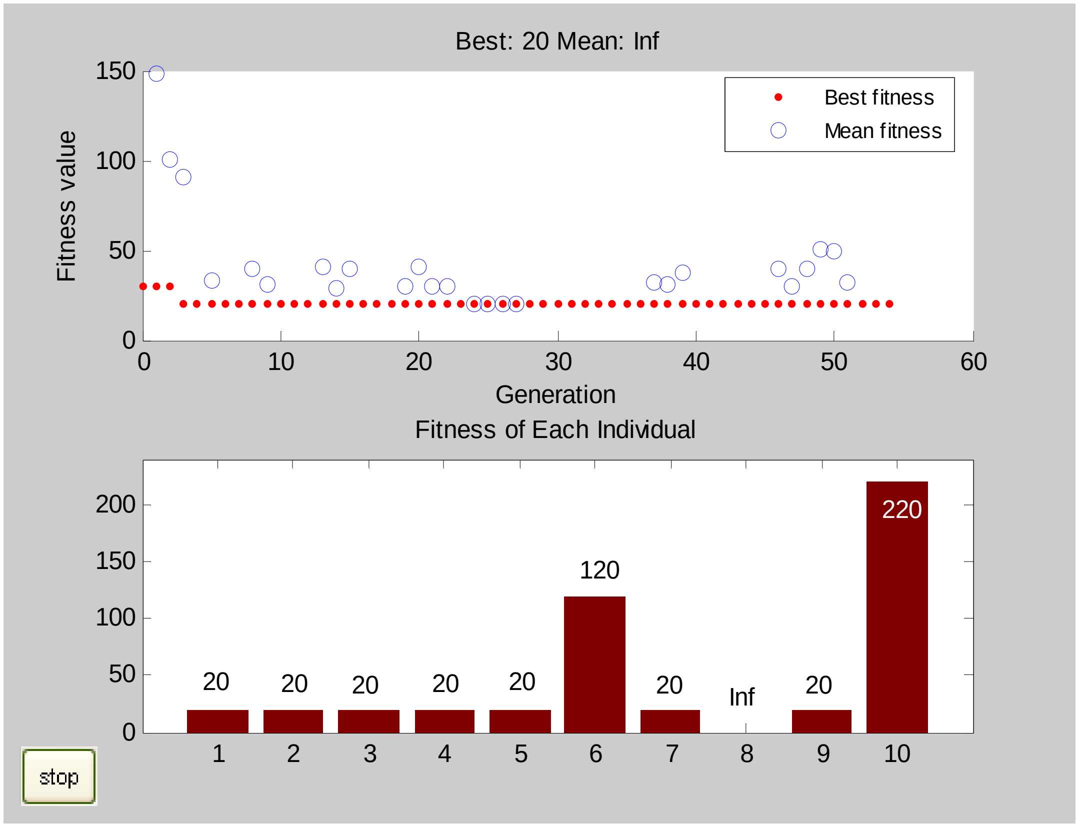

Figure A.3 MATLAB Plot of Fitness Function Values - Case 2a

Table A.4 Genetic Algorithm Output for Case - 2a

Generation	Best f(x)	Mean f(x)
1	30	148
2	30	101
3	20	91
4	20	Inf
5	20	33
6	20	Inf
7	20	Inf
8	20	40
9	20	31
10	20	Inf
11	20	Inf
12	20	Inf
13	20	41
14	20	29

Generation	Best f(x)	Mean f(x)
15	20	40
16	20	Inf
17	20	Inf
18	20	Inf
19	20	30
20	20	41
21	20	30
22	20	30
23	20	Inf
24	20	20
25	20	20
26	20	20
27	20	20
28	20	Inf
29	20	Inf
30	20	Inf
31	20	Inf
32	20	Inf
33	20	Inf
34	20	Inf
35	20	Inf

Generation	Best f(x)	Mean f(x)
36	20	Inf
37	20	32
38	20	31
39	20	38
40	20	Inf
41	20	Inf
42	20	Inf
43	20	Inf
44	20	Inf
45	20	Inf
46	20	40
47	20	30
48	20	40
49	20	51
50	20	50
51	20	32
52	20	Inf
53	20	Inf
54	20	Inf

A.2.2 CASE 2b: Minimum number of injection meters

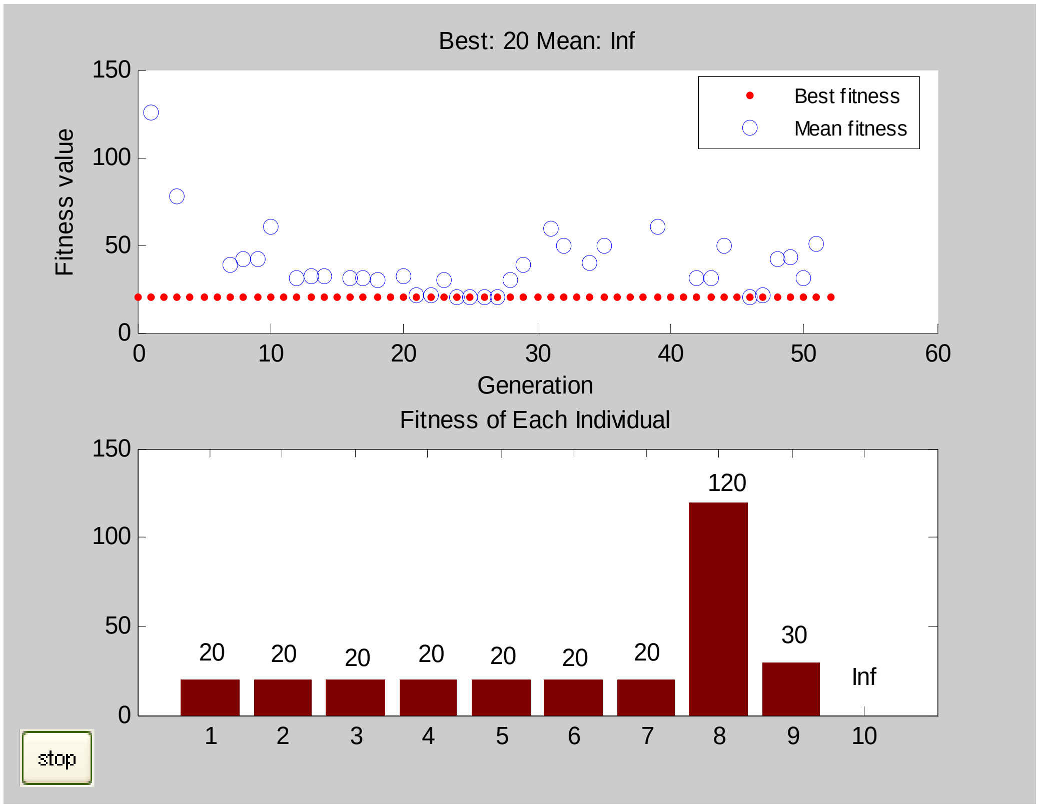

Figure A.4 MATLAB Plot of Fitness Function Values - Case 2b

Table A.5 Genetic Algorithm Output for Case - 2b

Generation	Best f(x)	Mean f(x)
1	20	126
2	20	Inf
3	20	78
4	20	Inf
5	20	Inf
6	20	Inf
7	20	39
8	20	42
9	20	42
10	20	60
11	20	Inf
12	20	31
13	20	32
14	20	32
15	20	Inf
16	20	31
17	20	31
18	20	30
19	20	Inf
20	20	32
21	20	21
22	20	21

Generation	Best f(x)	Mean f(x)
23	20	30
24	20	20
25	20	20
26	20	20
27	20	20
28	20	30
29	20	39
30	20	Inf
31	20	59
32	20	49
33	20	Inf
34	20	40
35	20	50
36	20	Inf
37	20	Inf
38	20	Inf

Generation	Best f(x)	Mean f(x)
39	20	60
40	20	Inf
41	20	Inf
42	20	31
43	20	31
44	20	50
45	20	Inf
46	20	20
47	20	21
48	20	42
49	20	43
50	20	31
51	20	51
52	20	Inf

A.3 CASE 3: Minimum number of meters with emphasis to eliminate critical measurements

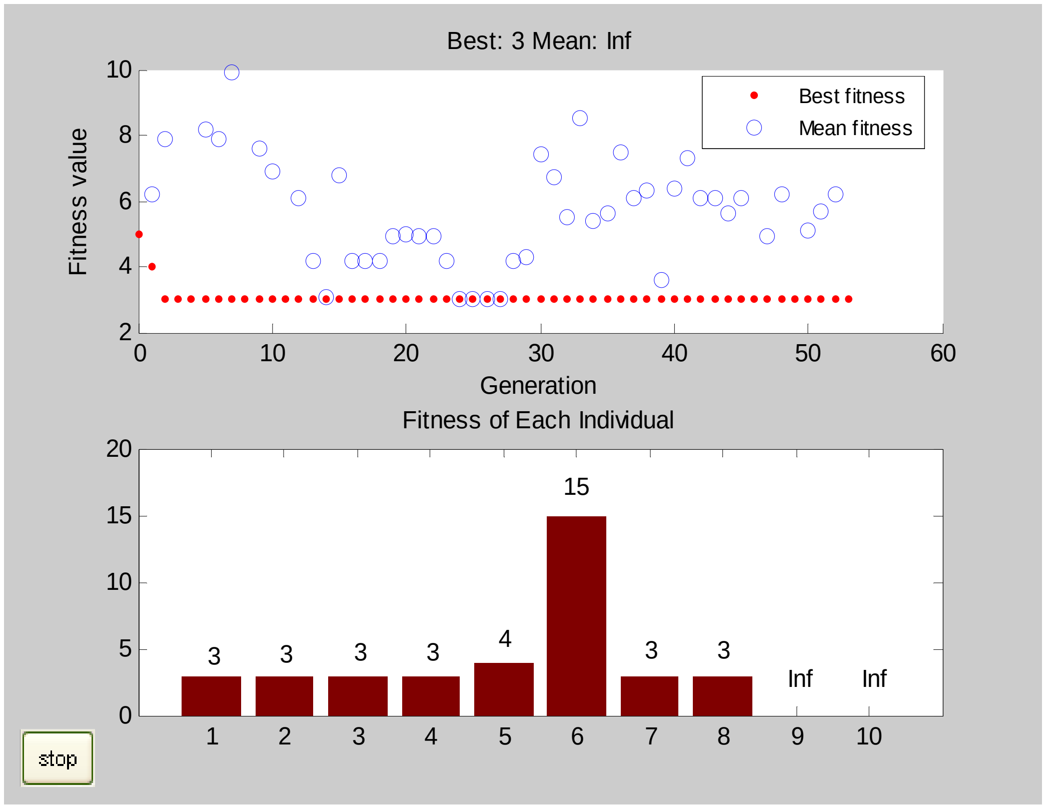

Figure A.5 MATLAB Plot of Fitness Function Values - Case 3

Table A.6 Genetic Algorithm Output for Case - 3

Generation	Best f(x)	Mean f(x)
1	4	6.2
2	3	7.9
3	3	Inf
4	3	Inf
5	3	8.2
6	3	7.9
7	3	9.9
8	3	Inf
9	3	7.6
10	3	6.9
11	3	Inf
12	3	6.1
13	3	4.2
14	3	3.1
15	3	6.8
16	3	4.2
17	3	4.2
18	3	4.2
19	3	4.9
20	3	5

Generation	Best f(x)	Mean f(x)
21	3	4.9
22	3	4.9
23	3	4.2
24	3	3
25	3	3
26	3	3
27	3	3
28	3	4.2
29	3	4.3
30	3	7.4
31	3	6.7
32	3	5.5
33	3	8.5
34	3	5.4
35	3	5.6
36	3	7.5
37	3	6.1

Generation	Best f(x)	Mean f(x)
38	3	6.3
39	3	3.6
40	3	6.4
41	3	7.3
42	3	6.1
43	3	6.1
44	3	5.6
45	3	6.1
46	3	Inf
47	3	4.9
48	3	6.2
49	3	9.3
50	3	5.1
51	3	5.7
52	3	6.2
53	3	Inf

A.4 CASE 4: Minimum number of meters with emphasis to perform first order contingency analysis

A.4.1 CASE 4a: Loss of one flow meter

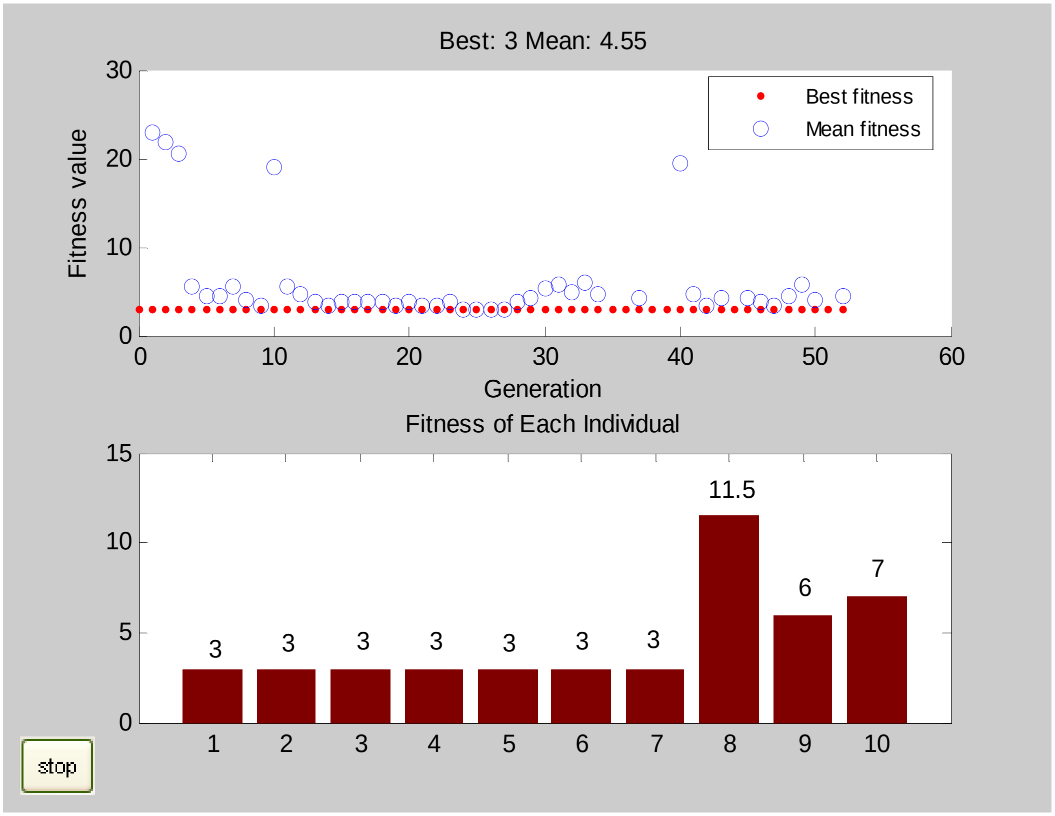

Figure A.6 MATLAB Plot of Fitness Function Values - Case 4a

Table A.7 Genetic Algorithm Output for Case - 4a

Generation	Best f(x)	Mean F(x)
1	3	22.95
2	3	21.95
3	3	20.55
4	3	5.65
5	3	4.55
6	3	4.55

Generation	Best f(x)	Mean F(x)
7	3	5.5
8	3	4.1
9	3	3.4
10	3	19
11	3	5.45
12	3	4.65

Generation	Best f(x)	Mean F(x)
13	3	3.85
14	3	3.3
15	3	3.8
16	3	3.85
17	3	3.85
18	3	3.85
19	3	3.4
20	3	3.7
21	3	3.4
22	3	3.4
23	3	3.85
24	3	3
25	3	3
26	3	3
27	3	3
28	3	3.85
29	3	4.15
30	3	5.4
31	3	5.85
32	3	5
33	3	5.95

Generation	Best f(x)	Mean F(x)
34	3	4.7
35	3	Inf
36	3	Inf
37	3	4.25
38	3	Inf
39	3	Inf
40	3	19.4
41	3	4.65
42	3	3.4
43	3	4.25
44	3	Inf
45	3	4.25
46	3	3.8
47	3	3.4
48	3	4.55
49	3	5.8
50	3	4
51	3	Inf
52	3	4.55

A.4.2 CASE 4b: Loss of one injection meter

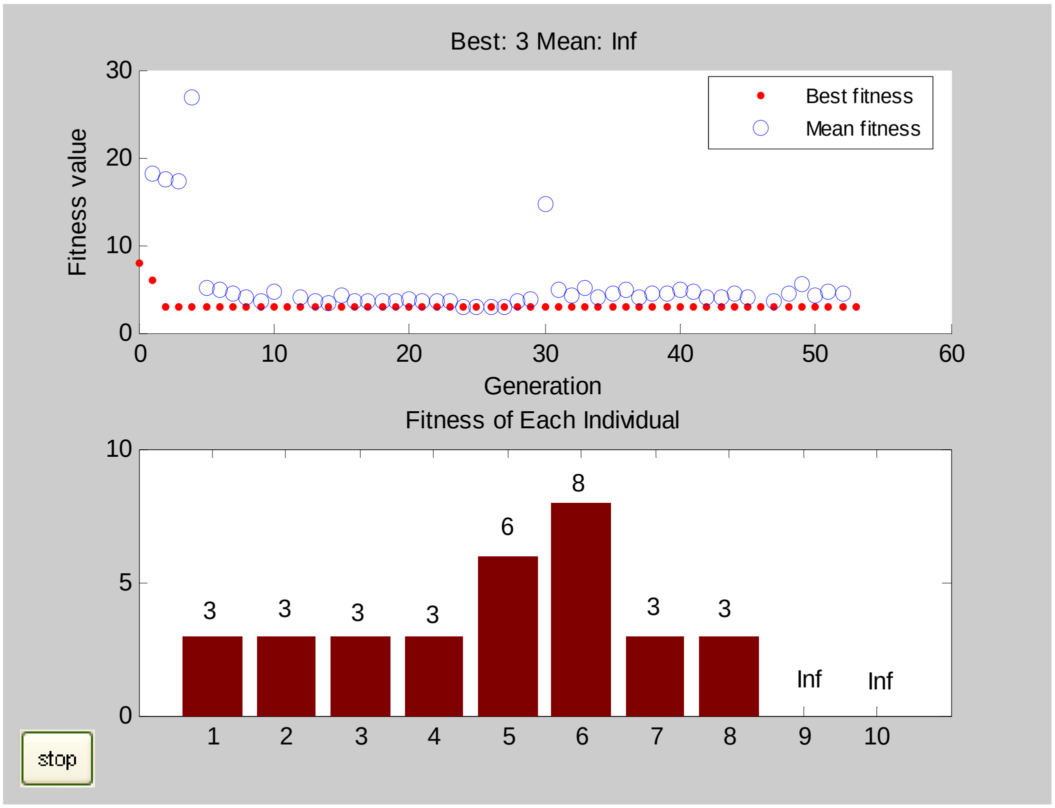

Figure A.7 MATLAB Plot of Fitness Function Values - Case 4b

Table A.8 Genetic Algorithm Output for Case - 4b

Generation	Best f(x)	Mean f(x)
1	6	18.15
2	3	17.45
3	3	17.35
4	3	26.8
5	3	5.1
6	3	5
7	3	4.5
8	3	4.1
9	3	3.6
10	3	4.7
11	3	Inf
12	3	4.1
13	3	3.5
14	3	3.3
15	3	4.2
16	3	3.5
17	3	3.5
18	3	3.5
19	3	3.6
20	3	3.9
21	3	3.6
22	3	3.6
23	3	3.5
24	3	3

Generation	Best f(x)	Mean f(x)
25	3	3
26	3	3
27	3	3
28	3	3.5
29	3	3.8
30	3	14.65
31	3	4.8
32	3	4.3
33	3	5.1
34	3	4
35	3	4.4
36	3	5
37	3	4.1
38	3	4.5
39	3	4.4

Generation	Best f(x)	Mean f(x)
40	3	4.8
41	3	4.6
42	3	4.1
43	3	4.1
44	3	4.4
45	3	4.1
46	3	Inf
47	3	3.6
48	3	4.4
49	3	5.5
50	3	4.2
51	3	4.7
52	3	4.4
53	3	Inf

A.5 CASE 5: Minimum number of meters with emphasis to perform second order contingency analysis

A.5.1 CASE 5a: Loss of two flow meters

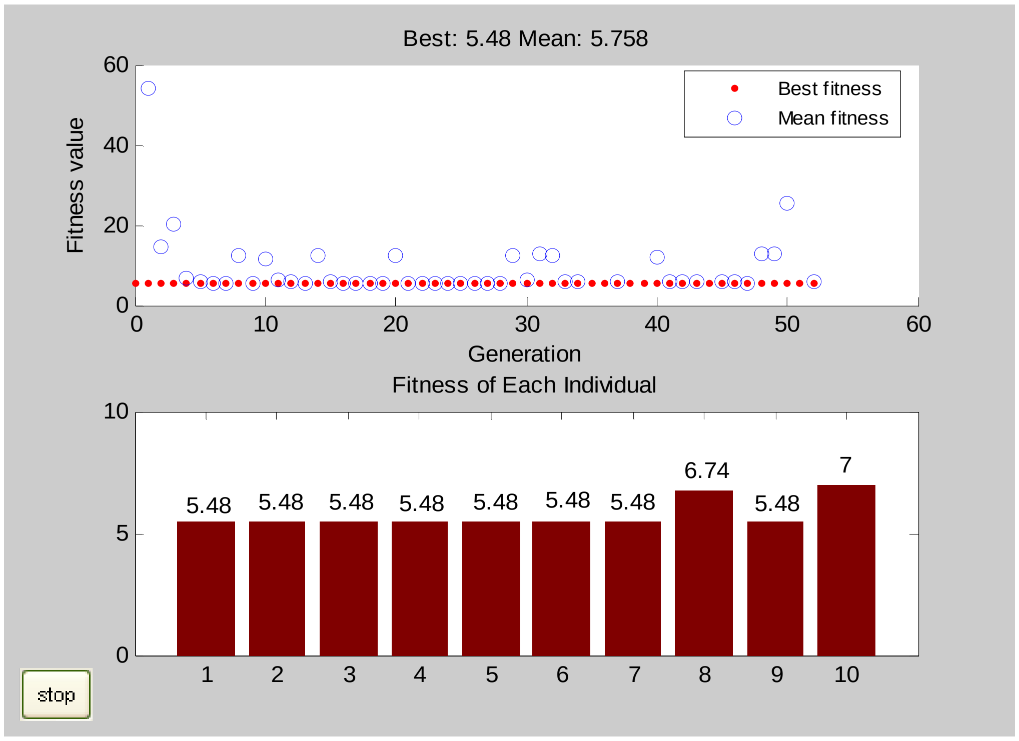

Figure A.8 MATLAB Plot of Fitness Function Values – Case 5a

Table A.9 Genetic Algorithm Output for Case - 5a

Generation	Best f(x)	Mean f(x)
1	5.48	53.92
2	5.48	14.61
3	5.48	20.07
4	5.48	6.64
5	5.48	5.784
6	5.48	5.632
7	5.48	5.606
8	5.48	12.5
9	5.48	5.632

Generation	Best f(x)	Mean f(x)
10	5.48	11.67
11	5.48	6.462
12	5.48	5.758
13	5.48	5.606
14	5.48	12.35
15	5.48	5.784
16	5.48	5.606
17	5.48	5.606
18	5.48	5.606

Generation	Best f(x)	Mean f(x)
19	5.48	5.632
20	5.48	12.5
21	5.48	5.632
22	5.48	5.632
23	5.48	5.606
24	5.48	5.48
25	5.48	5.48
26	5.48	5.48
27	5.48	5.48
28	5.48	5.606
29	5.48	12.48
30	5.48	6.184
31	5.48	12.73
32	5.48	12.6
33	5.48	5.858
34	5.48	5.732
35	5.48	Inf
36	5.48	Inf

Generation	Best f(x)	Mean f(x)
37	5.48	5.758
38	5.48	Inf
39	5.48	Inf
40	5.48	11.82
41	5.48	5.884
42	5.48	5.758
43	5.48	5.758
44	5.48	Inf
45	5.48	5.758
46	5.48	6.032
47	5.48	5.632
48	5.48	12.63
49	5.48	12.91
50	5.48	25.51
51	5.48	Inf
52	5.48	5.758

A.5.2   CASE 5b: Loss of two injection meters

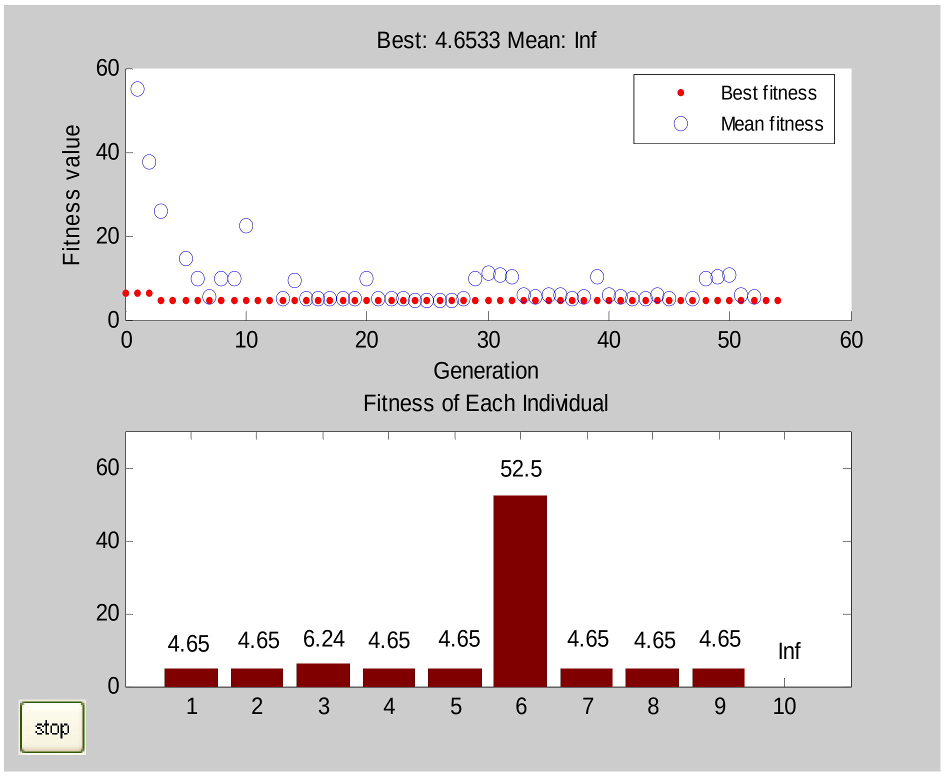

Figure A.9   MATLAB Plot of Fitness Function Values - Case 5b

Table A.10   Genetic Algorithm Output for Case - 5b

Generation	Best f(x)	Mean f(x)
1	6.24	54.88
2	6.24	37.49
3	4.653	25.73
4	4.653	Inf
5	4.653	14.7
6	4.653	9.931
7	4.653	5.323
8	4.653	9.931
9	4.653	9.931
10	4.653	22.5
11	4.653	Inf
12	4.653	Inf
13	4.653	5.147

Generation	Best f(x)	Mean f(x)
14	4.653	9.438
15	4.653	4.971
16	4.653	4.988
17	4.653	4.988
18	4.653	4.988
19	4.653	4.812
20	4.653	9.597
21	4.653	4.812
22	4.653	4.812
23	4.653	4.988
24	4.653	4.653
25	4.653	4.653
26	4.653	4.653

Generation	Best f(x)	Mean f(x)
27	4.653	4.653
28	4.653	4.988
29	4.653	9.773
30	4.653	11.07
31	4.653	10.44
32	4.653	10.11
33	4.653	5.657
34	4.653	5.323
35	4.653	5.723
36	4.653	5.881
37	4.653	5.147
38	4.653	5.547
39	4.653	10.37
40	4.653	5.681
41	4.653	5.305

Generation	Best f(x)	Mean f(x)
42	4.653	4.812
43	4.653	5.147
44	4.653	5.723
45	4.653	5.147
46	4.653	Inf
47	4.653	4.812
48	4.653	9.931
49	4.653	10.09
50	4.653	10.56
51	4.653	5.881
52	4.653	5.305
53	4.653	Inf
54	4.653	Inf

A.5.3 CASE 5c: Loss of one flow meter and one injection meter

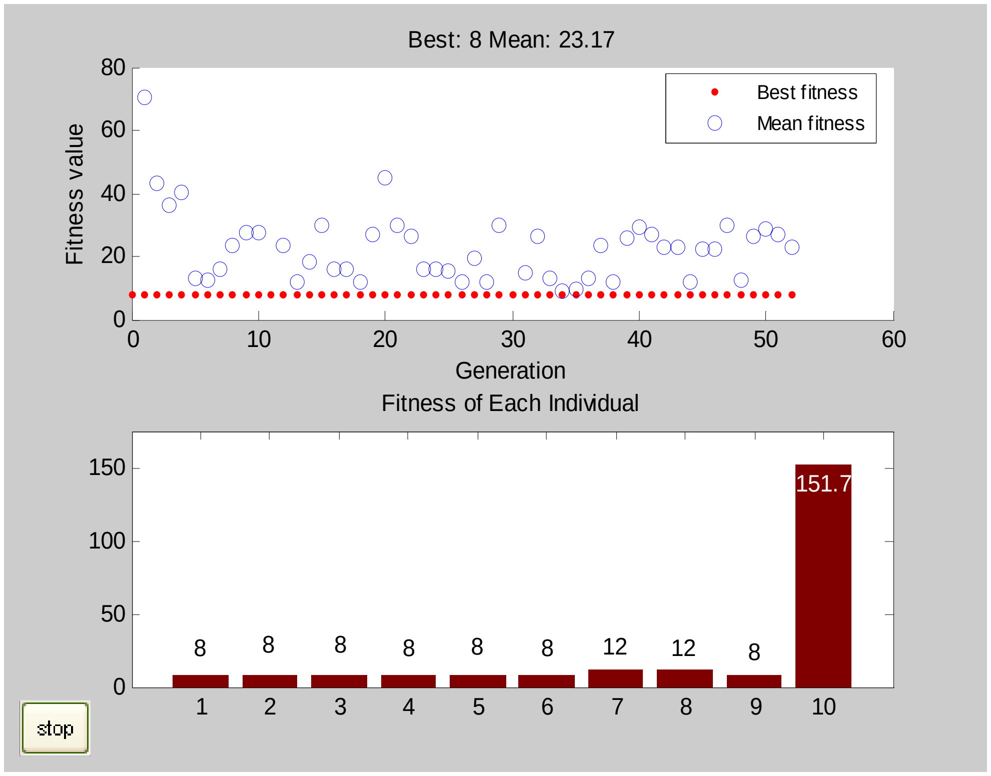

Figure A.9 MATLAB Plot of Fitness Function Values - Case 5c

Table A.10 Genetic Algorithm Output for Case – 5c

Generation	Best f(x)	Mean f(x)
1	8	70.71
2	8	43.41
3	8	36.46
4	8	40.57
5	8	12.83
6	8	12.43
7	8	16.07
8	8	23.37
9	8	27.4
10	8	27.28
11	8	Inf
12	8	23.34
13	8	12.03
14	8	18.27
15	8	29.8
16	8	15.67
17	8	15.67
18	8	12.03
19	8	26.8
20	8	44.81
21	8	30.04
22	8	26.4
23	8	15.67
24	8	15.67
25	8	15.27
26	8	11.63
27	8	19.3
28	8	11.63
29	8	30.04
30	8	Inf
31	8	15.03
32	8	26.4
33	8	13.23
34	8	9.2
35	8	9.5
36	8	13.33
37	8	23.57
38	8	11.8
39	8	25.54
40	8	29.4
41	8	27.2
42	8	22.77
43	8	23.17
44	8	11.63
45	8	22.37
46	8	22.14
47	8	30.04
48	8	12.2
49	8	26.17
50	8	28.6
51	8	26.7
52	8	23.17

APPENDIX B

TEST RESULTS FOR 18-BUS SYSTEM

The algorithms were tested with the 18-bus system. System topology, meter types (power flow and power injection meters) and all possible meter locations were given as input to the algorithms. Figure B.1 shows the 18-bus test system considered while testing the algorithms and the fitness function. The figure also shows all possible meter locations. The meters are numbered, starting with all the injection meters followed by the flow meters in the clockwise direction.

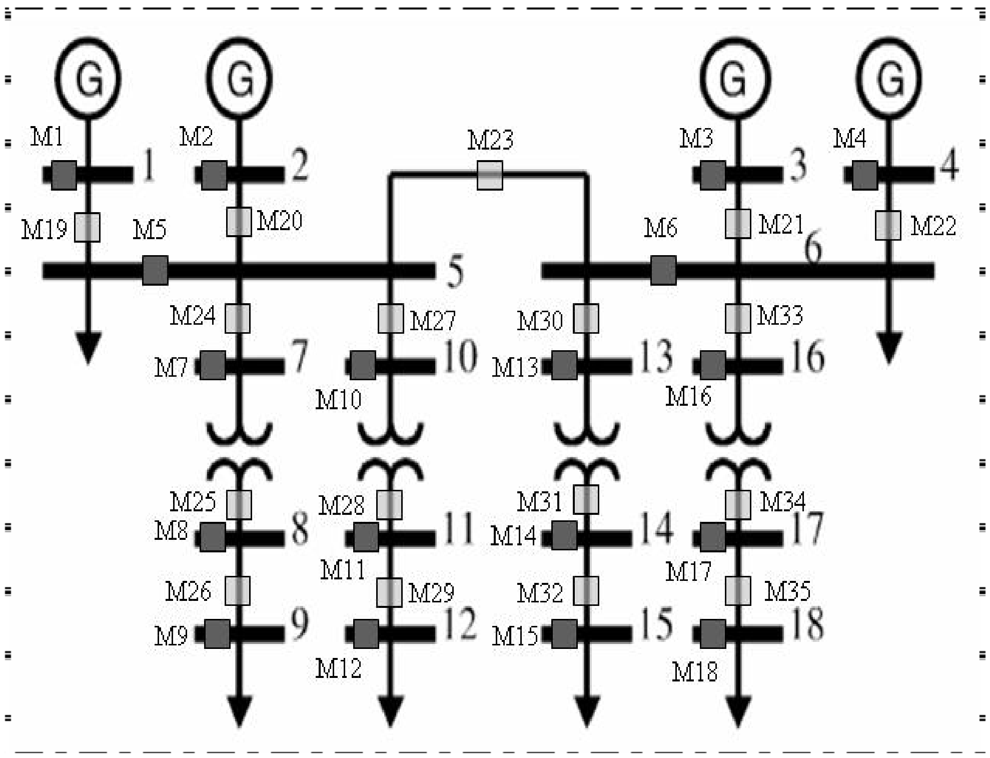

Figure B.1 18-Bus System with Meters [22]

The system is tested for different cases as shown in the subsequent sections. In each case different weights were applied to each constraint in the fitness function depending on their priority. The following list shows the weight applied to each constraint in the FF.

Weight applied to number of flow meters, P1

Weight applied to number of injection meters, P2

Weight applied to critical measurements, P3

Generic weight factor for FO contingency, pen_first

Generic weight factor for SO contingency, pen_second

Fail probability for loss of 1 flow meter, prob_fail_flow

Fail probability for loss of 1 injection meter, prob_fail_inj

Weight applied to loss of 1 flow meter in FO contingency, P4 = pen_first × prob_fail_flow

Weight applied to loss of 1 injection meter in FO contingency, P5 = pen_first × prob_fail_inj

Weight applied to loss of 2 flow meter in SO contingency, P6 = pen_second × prob_fail_flow × prob_fail_flow

Weight applied to loss of 2 injection meter in SO contingency, P7 = pen_second × prob_fail_inj × prob_fail_inj

Weight applied to loss of 1 flow and 1 injection meter in SO contingency, P8 = pen_second × prob_fail_inj × prob_fail_flow

Table B.1 shows the value of the weights applied in each case.

Table B.1 Weights for Constraints

Case No.	P1	P2	P3	pen_first	pen_second	Prob_fail_flow	Prob_fail_inj	P4	P5	P6	P7	P8
1	10	10	0	0	0	0	0	0	0	0	0	0
2a	100	10	0	0	0	0	0	0	0	0	0	0
2b	10	100	0	0	0	0	0	0	0	0	0	0
3	1	2	10	0	0	0	0	0	0	0	0	0
4a	4	1	1	50	0	0.95	0.05	47.5	2.5	0	0	0
4b	1	4	1	50	0	0.05	0.95	2.5	47.5	0	0	0
5a	4	1	1	5	124	0.9	0.1	4.5	0.5	100.44	1.24	11.16
5b	1	4	1	5	124	0.1	0.9	0.5	4.5	1.24	100.44	11.16
5c	1	4	1	3	124	0.5	0.4	1.5	1.2	31.25	19.84	24.8

Table B.2 Solution for all Cases

Case No.	Meter Arrangement	Meter Location	Total No. of Meters	Min. FF Value	Profile Time (s)
1	1001111011101001000100010 0010101011	M1, M4, M5, M6, M7, M9, M10, M11, M13, M16, M20, M24, M28, M30, M32, M34, M35	17	170	5.750
2a	1011111011101001100100010 0000101010	M1, M3, M4, M5, M6, M7, M9, M10, M11, M13, M16, M17, M20, M24, M30, M32, M34	17	620	8.468
2b	1000100000001000000111011 1111101111	M1, M5, M13, M20, M21, M22, M24, M25, M26, M27, M28, M29, M30, M32, M33, M34, M35	17	440	7.453
3	1000100000000000000111111 1111111111	M1, M5, M20, M21, M22, M23, M24, M25, M26, M27, M28, M29, M30, M31, M32, M33, M34, M35	18	20	10.047

Case No.	Meter Arrangement	Meter Location	Total No. of Meters	Min. FF Value	Profile Time (s)
4a	11111111111111101010000000 0000001010	M1, M2, M3, M4, M5, M6, M7, M8, M9, M10, M11, M12, M13, M14, M16, M18, M32, M34	18	25.16	30.343
4b	10001110000000001000111001 1111111011	M1, M5, M6, M7, M16, M20, M21, M22, M25, M26, M27, M28, M29, M30, M31, M32, M34, M35	18	34.15	12.328
5a	11111111111111101110000000 0000011000	M1, M2, M3, M4, M5, M6, M7, M8, M9, M10, M11, M12, M13, M14 M16, M17, M18, M31, M32	19	28.95	514.282
5b	00001101001001000011111111 1111111111	M5, M6, M8, M11, M14, M19, M20, M21, M22, M23, M24, M25, M26, M27, M28, M29, M30, M31, M32, M33, M34, M35	22	43.89	301.156
5c	00001101001001001011111111 1111111111	M5, M6, M8, M11, M14, M17, M19, M20, M21, M22, M23, M24, M25, M26, M27, M28, M29, M30, M31, M32, M33, M34, M35	23	55.65	535.828

The explanation about the MATLAB plots, tables and outputs from the Genetic Algorithm is given in Chapter 5, Section 5.1.2. A summary and comparison between the three test systems is also given in Chapter 5, Section 5.2.

B.1 CASE 1: Minimum number of meters without considering other factors

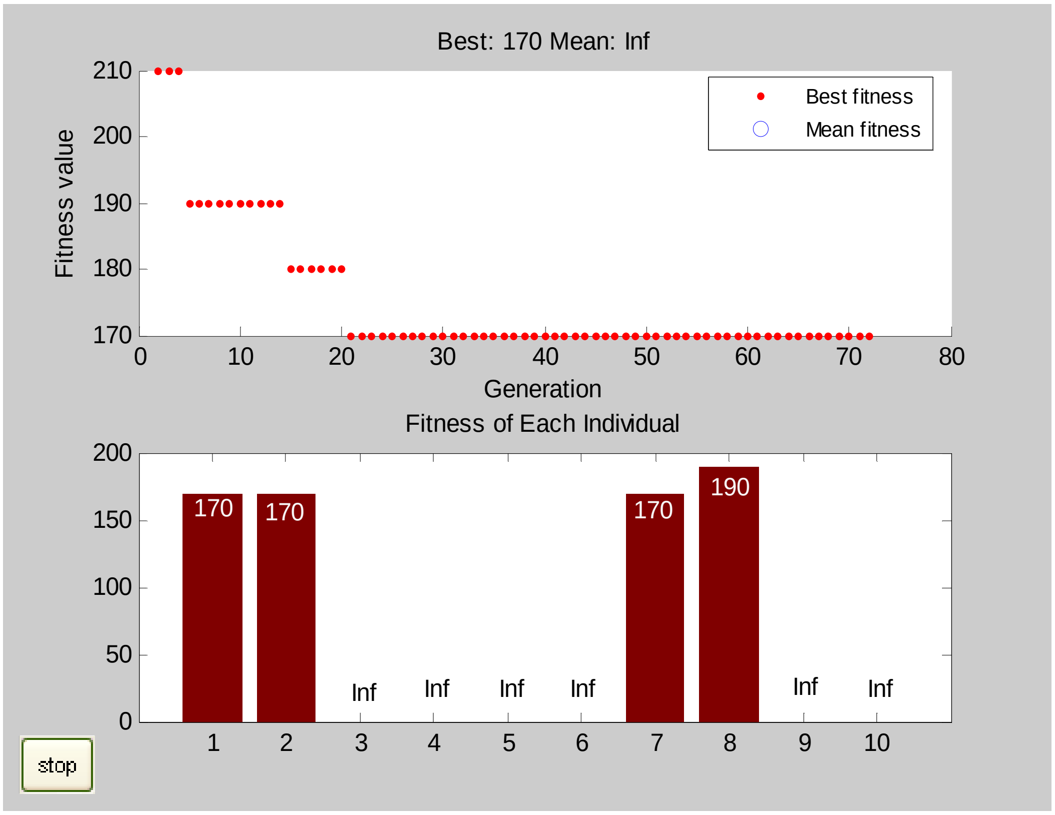

Figure B.2 MATLAB Plot of Fitness Function Values – Case 1

Table B.3 Genetic Algorithm Output for Case 1

Generation	Best f(x)	Mean f(x)
1	Inf	Inf
2	210	Inf
3	210	Inf
4	210	Inf
5	190	Inf
6	190	Inf
7	190	Inf
8	190	Inf
9	190	Inf
10	190	Inf
11	190	Inf
12	190	Inf
13	190	Inf
14	190	Inf
15	180	Inf
16	180	Inf
17	180	Inf
18	180	Inf
19	180	Inf
20	180	Inf
21	170	Inf
22	170	Inf
23	170	Inf
24	170	Inf

Generation	Best f(x)	Mean f(x)
25	170	Inf
26	170	Inf
27	170	Inf
28	170	Inf
29	170	Inf
30	170	Inf
31	170	Inf
32	170	Inf
33	170	Inf
34	170	Inf
35	170	Inf
36	170	Inf
37	170	Inf
38	170	Inf
39	170	Inf
40	170	Inf
41	170	Inf
42	170	Inf
43	170	Inf
44	170	Inf
45	170	Inf
46	170	Inf
47	170	Inf
48	170	Inf
49	170	Inf

Generation	Best f(x)	Mean f(x)
50	170	Inf
51	170	Inf
52	170	Inf
53	170	Inf
54	170	Inf
55	170	Inf
56	170	Inf
57	170	Inf
58	170	Inf
59	170	Inf
60	170	Inf
61	170	Inf
62	170	Inf
63	170	Inf
64	170	Inf
65	170	Inf
66	170	Inf
67	170	Inf
68	170	Inf
69	170	Inf
70	170	Inf
71	170	Inf
72	170	Inf

B.2 CASE 2: Minimum number of meters with emphasis on type of meter

B.2.1 CASE 2a: Minimum number of flow meters

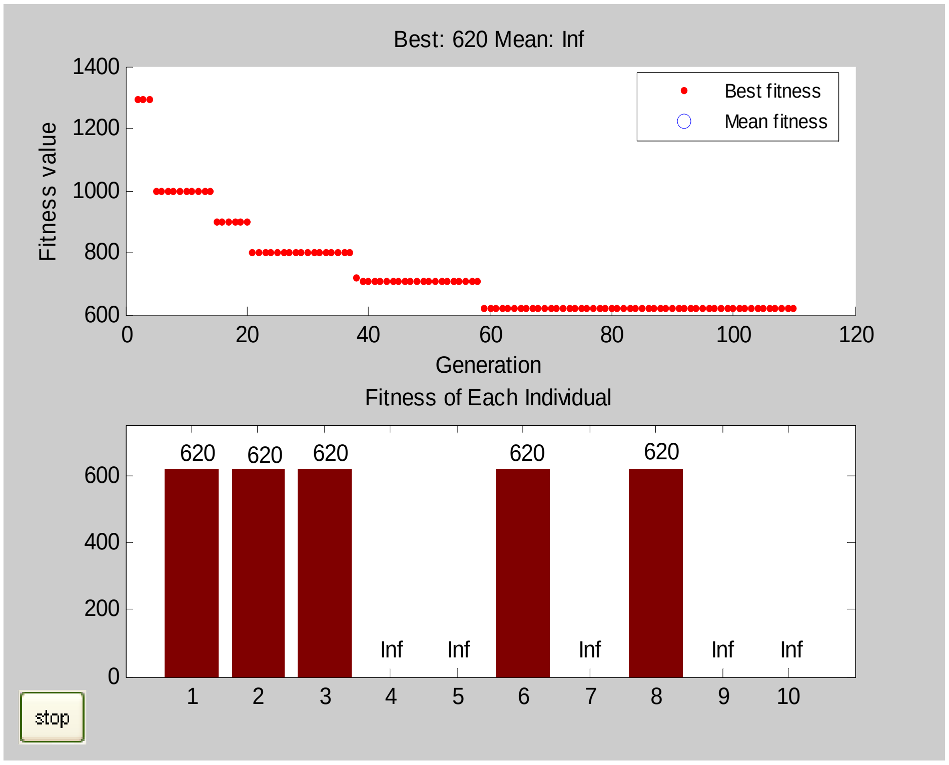

Figure B.3 MATLAB Plot of Fitness Function Values – Case 2a

Table B.4 Genetic Algorithm Output for Case 2a

Generation	Best f(x)	Mean f(x)
1	Inf	Inf
2	1290	Inf
3	1290	Inf
4	1290	Inf
5	1000	Inf
6	1000	Inf
7	1000	Inf
8	1000	Inf
9	1000	Inf
10	1000	Inf

Generation	Best f(x)	Mean f(x)
11	1000	Inf
12	1000	Inf
13	1000	Inf
14	1000	Inf
15	900	Inf
16	900	Inf
17	900	Inf
18	900	Inf
19	900	Inf
20	900	Inf

Generation	Best f(x)	Mean f(x)
21	800	Inf
22	800	Inf
23	800	Inf
24	800	Inf
25	800	Inf
26	800	Inf
27	800	Inf
28	800	Inf
29	800	Inf
30	800	Inf
31	800	Inf
32	800	Inf
33	800	Inf
34	800	Inf
35	800	Inf
36	800	Inf
37	800	Inf
38	720	Inf
39	710	Inf
40	710	Inf
41	710	Inf
42	710	Inf
43	710	Inf
44	710	Inf
45	710	Inf
46	710	Inf
47	710	Inf
48	710	Inf
49	710	Inf
50	710	Inf
51	710	Inf
52	710	Inf
53	710	Inf
54	710	Inf
55	710	Inf
56	710	Inf
57	710	Inf
58	710	Inf
59	620	Inf
60	620	Inf
61	620	Inf
62	620	Inf
63	620	Inf
64	620	Inf
65	620	Inf

Generation	Best f(x)	Mean f(x)
66	620	Inf
67	620	Inf
68	620	Inf
69	620	Inf
70	620	Inf
71	620	Inf
72	620	Inf
73	620	Inf
74	620	Inf
75	620	Inf
76	620	Inf
77	620	Inf
78	620	Inf
79	620	Inf
80	620	Inf
81	620	Inf
82	620	Inf
83	620	Inf
84	620	Inf
85	620	Inf
86	620	Inf
87	620	Inf
88	620	Inf
89	620	Inf
90	620	Inf
91	620	Inf
92	620	Inf
93	620	Inf
94	620	Inf
95	620	Inf
96	620	Inf
97	620	Inf
98	620	Inf
99	620	Inf
100	620	Inf
101	620	Inf
102	620	Inf
103	620	Inf
104	620	Inf
105	620	Inf
106	620	Inf
107	620	Inf
108	620	Inf
109	620	Inf
110	620	Inf

B.2.2 CASE 2b: Minimum number of injection meters

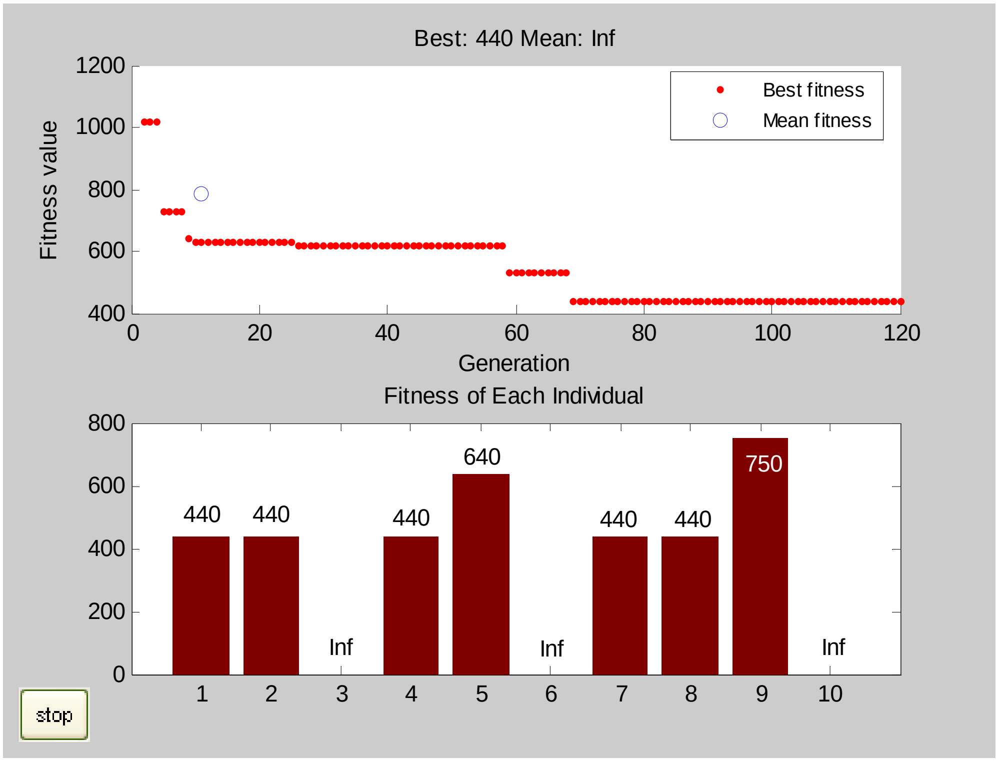

Figure B.4 MATLAB Plot of Fitness Function Values – Case 2b

Table B.5 Genetic Algorithm Output for Case 2b

Generation	Best f(x)	Mean f(x)
1	Inf	Inf
2	1020	Inf
3	1020	Inf
4	1020	Inf
5	730	Inf
6	730	Inf
7	730	Inf
8	730	Inf
9	640	Inf
10	630	Inf
11	630	788
12	630	Inf
13	630	Inf
14	630	Inf
15	630	Inf
16	630	Inf
17	630	Inf
18	630	Inf
19	630	Inf
20	630	Inf
21	630	Inf
22	630	Inf
23	630	Inf
24	630	Inf
25	630	Inf
26	620	Inf

Generation	Best f(x)	Mean f(x)
27	620	Inf
28	620	Inf
29	620	Inf
30	620	Inf
31	620	Inf
32	620	Inf
33	620	Inf
34	620	Inf
35	620	Inf
36	620	Inf
37	620	Inf
38	620	Inf
39	620	Inf
40	620	Inf
41	620	Inf
42	620	Inf
43	620	Inf
44	620	Inf
45	620	Inf
46	620	Inf
47	620	Inf
48	620	Inf
49	620	Inf
50	620	Inf
51	620	Inf
52	620	Inf
53	620	Inf
54	620	Inf
55	620	Inf
56	620	Inf
57	620	Inf
58	620	Inf
59	530	Inf
60	530	Inf
61	530	Inf
62	530	Inf
63	530	Inf
64	530	Inf
65	530	Inf
66	530	Inf
67	530	Inf
68	530	Inf
69	440	Inf
70	440	Inf
71	440	Inf

Generation	Best f(x)	Mean f(x)
72	440	Inf
73	440	Inf
74	440	Inf
75	440	Inf
76	440	Inf
77	440	Inf
78	440	Inf
79	440	Inf
80	440	Inf
81	440	Inf
82	440	Inf
83	440	Inf
84	440	Inf
85	440	Inf
86	440	Inf
87	440	Inf
88	440	Inf
89	440	Inf
90	440	Inf
91	440	Inf
92	440	Inf
93	440	Inf
94	440	Inf
95	440	Inf
96	440	Inf
97	440	Inf
98	440	Inf
99	440	Inf
100	440	Inf
101	440	Inf
102	440	Inf
103	440	Inf
104	440	Inf
105	440	Inf
106	440	Inf
107	440	Inf
108	440	Inf
109	440	Inf
110	440	Inf
111	440	Inf
112	440	Inf
113	440	Inf
114	440	Inf
115	440	Inf
116	440	Inf

Generation	Best f(x)	Mean f(x)
117	440	Inf
118	440	Inf
119	440	Inf
120	440	Inf

B.3 CASE 3: Minimum number of meters with emphasis to eliminate critical measurements

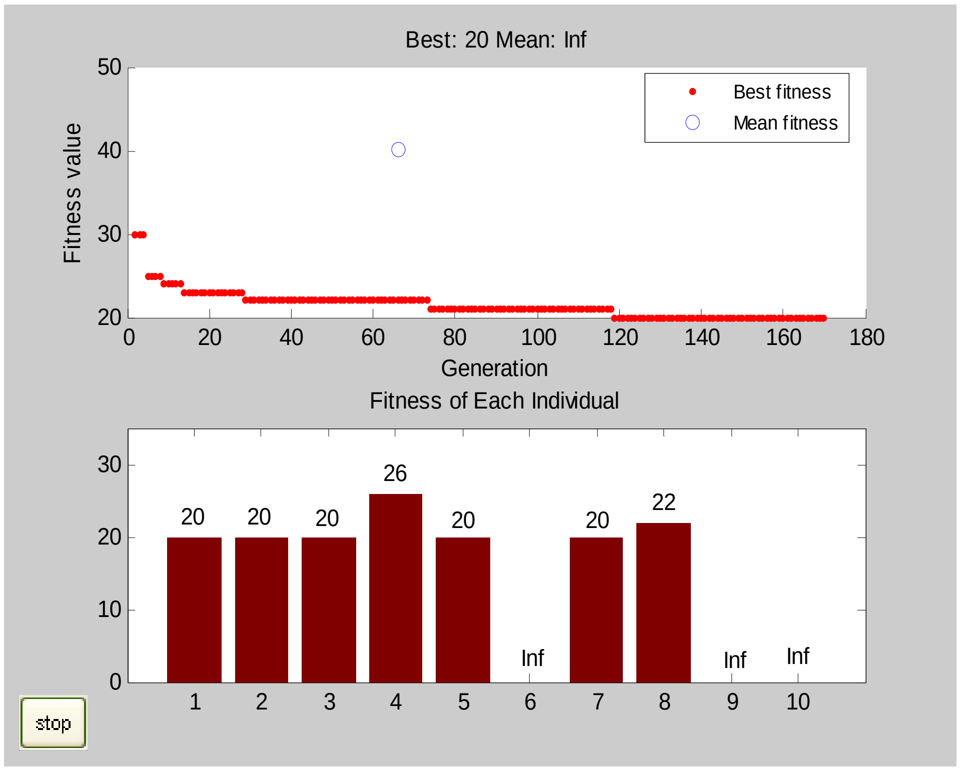

Figure B.5 MATLAB Plot of Fitness Function Values – Case 3

Table B.6 Genetic Algorithm Output for Case 3

Generation	Best f(x)	Mean f(x)
1	Inf	Inf
2	30	Inf
3	30	Inf
4	30	Inf
5	25	Inf
6	25	Inf
7	25	Inf
8	25	Inf

Generation	Best f(x)	Mean f(x)
9	24	Inf
10	24	Inf
11	24	Inf
12	24	Inf
13	24	Inf
14	23	Inf
15	23	Inf
16	23	Inf
17	23	Inf
18	23	Inf
19	23	Inf
20	23	Inf
21	23	Inf
22	23	Inf
23	23	Inf
24	23	Inf
25	23	Inf
26	23	Inf
27	23	Inf
28	23	Inf
29	22	Inf
30	22	Inf
31	22	Inf
32	22	Inf
33	22	Inf
34	22	Inf
35	22	Inf
36	22	Inf
37	22	Inf
38	22	Inf
39	22	Inf
40	22	Inf
41	22	Inf
42	22	Inf
43	22	Inf
44	22	Inf
45	22	Inf
46	22	Inf
47	22	Inf
48	22	Inf
49	22	Inf
50	22	Inf
51	22	Inf
52	22	Inf
53	22	Inf

Generation	Best f(x)	Mean f(x)
54	22	Inf
55	22	Inf
56	22	Inf
57	22	Inf
58	22	Inf
59	22	Inf
60	22	Inf
61	22	Inf
62	22	Inf
63	22	Inf
64	22	Inf
65	22	Inf
66	22	40.1
67	22	Inf
68	22	Inf
69	22	Inf
70	22	Inf
71	22	Inf
72	22	Inf
73	22	Inf
74	21	Inf
75	21	Inf
76	21	Inf
77	21	Inf
78	21	Inf
79	21	Inf
80	21	Inf
81	21	Inf
82	21	Inf
83	21	Inf
84	21	Inf
85	21	Inf
86	21	Inf
87	21	Inf
88	21	Inf
89	21	Inf
90	21	Inf
91	21	Inf
92	21	Inf
93	21	Inf
94	21	Inf
95	21	Inf
96	21	Inf
97	21	Inf
98	21	Inf

Generation	Best f(x)	Mean f(x)
99	21	Inf
100	21	Inf
101	21	Inf
102	21	Inf
103	21	Inf
104	21	Inf
105	21	Inf
106	21	Inf
107	21	Inf
108	21	Inf
109	21	Inf
110	21	Inf
111	21	Inf
112	21	Inf
113	21	Inf
114	21	Inf
115	21	Inf
116	21	Inf
117	21	Inf
118	21	Inf
119	20	Inf
120	20	Inf
121	20	Inf
122	20	Inf
123	20	Inf
124	20	Inf
125	20	Inf
126	20	Inf
127	20	Inf
128	20	Inf
129	20	Inf
130	20	Inf
131	20	Inf
132	20	Inf
133	20	Inf
134	20	Inf
135	20	Inf

Generation	Best f(x)	Mean f(x)
136	20	Inf
137	20	Inf
138	20	Inf
139	20	Inf
140	20	Inf
141	20	Inf
142	20	Inf
143	20	Inf
144	20	Inf
145	20	Inf
146	20	Inf
147	20	Inf
148	20	Inf
149	20	Inf
150	20	Inf
151	20	Inf
152	20	Inf
153	20	Inf
154	20	Inf
155	20	Inf
156	20	Inf
157	20	Inf
158	20	Inf
159	20	Inf
160	20	Inf
161	20	Inf
162	20	Inf
163	20	Inf
164	20	Inf
165	20	Inf
166	20	Inf
167	20	Inf
168	20	Inf
169	20	Inf
170	20	Inf

B.4 CASE 4: Minimum number of meters with emphasis to perform first order contingency analysis

B.4.1 CASE 4a: Loss of one flow meter

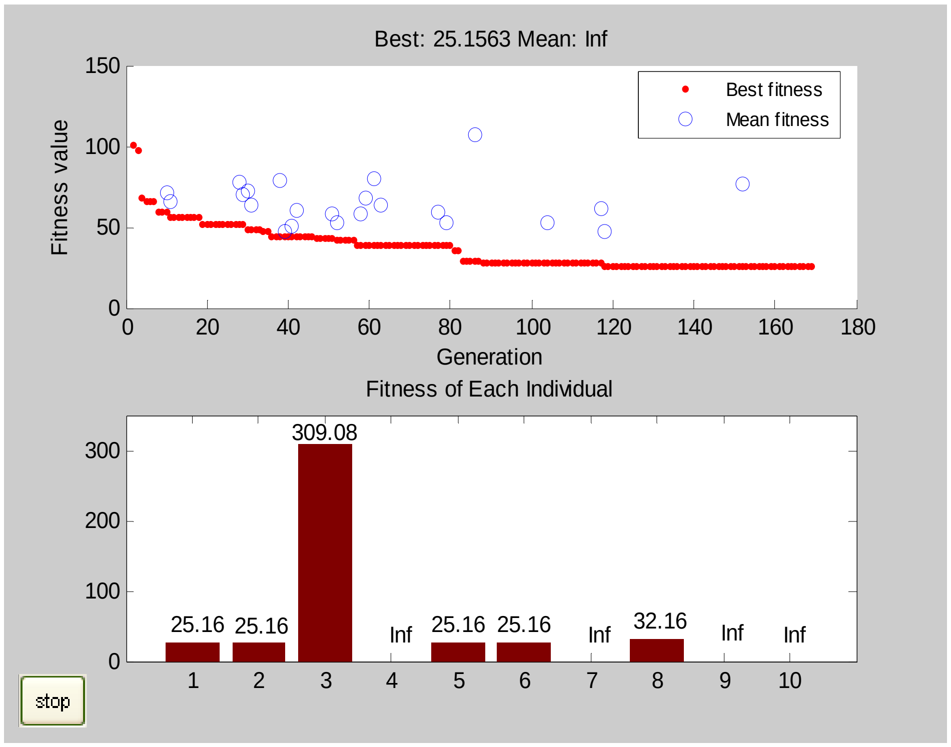

Figure B.6 MATLAB Plot of Fitness Function Values – Case 4a

Table B.7 Genetic Algorithm Output for Case 4a

Generation	Best f(x)	Mean f(x)	Generation	Best f(x)	Mean f(x)
1	Inf	Inf	10	59	70.86
2	100.2	Inf	11	56.21	65.44
3	97.13	Inf	12	56.21	Inf
4	68.31	Inf	13	56.21	Inf
5	65.65	Inf	14	56.21	Inf
6	65.65	Inf	15	56.21	Inf
7	65.65	Inf	16	56.21	Inf
8	59	Inf	17	56.21	Inf
9	59	Inf	18	56.21	Inf

Generation	Best f(x)	Mean f(x)
19	51.23	Inf
20	51.23	Inf
21	51.23	Inf
22	51.23	Inf
23	51.23	Inf
24	51.23	Inf
25	51.23	Inf
26	51.23	Inf
27	51.23	Inf
28	51.23	77.22
29	51.23	70.18
30	48.21	72.46
31	48.21	63.14
32	48.21	Inf
33	48.21	Inf
34	47.23	Inf
35	47.23	Inf
36	44.21	Inf
37	44.21	Inf
38	44.21	79
39	44.21	47.67
40	44.21	Inf
41	44.21	50.96
42	44.21	60.48
43	44.21	Inf
44	44.21	Inf
45	44.21	Inf
46	44.21	Inf
47	43.23	Inf
48	43.23	Inf
49	43.23	Inf
50	43.23	Inf
51	43.23	57.86
52	42.18	52.52
53	42	Inf
54	42	Inf
55	41.38	Inf
56	41.38	Inf
57	39	Inf
58	39	57.67
59	39	67.49
60	39	Inf
61	39	79.88
62	39	Inf
63	39	63.7

Generation	Best f(x)	Mean f(x)
64	39	Inf
65	39	Inf
66	39	Inf
67	39	Inf
68	39	Inf
69	39	Inf
70	39	Inf
71	39	Inf
72	39	Inf
73	39	Inf
74	39	Inf
75	39	Inf
76	39	Inf
77	39	59.21
78	39	Inf
79	39	52.46
80	39	Inf
81	35	Inf
82	35	Inf
83	29	Inf
84	29	Inf
85	29	Inf
86	29	106.6
87	29	Inf
88	28.17	Inf
89	28.17	Inf
90	28.17	Inf
91	28.17	Inf
92	28.17	Inf
93	28.17	Inf
94	28.17	Inf
95	28.17	Inf
96	28.17	Inf
97	28.17	Inf
98	28.17	Inf
99	28.17	Inf
100	28.17	Inf
101	28.17	Inf
102	28.17	Inf
103	28.17	Inf
104	28.17	53.1
105	28.17	Inf
106	28.17	Inf
107	28.17	Inf
108	28.17	Inf

Generation	Best f(x)	Mean f(x)
109	28.17	Inf
110	28.17	Inf
111	28.17	Inf
112	28.17	Inf
113	28.17	Inf
114	28.17	Inf
115	28.17	Inf
116	28.17	Inf
117	28.16	61.76
118	25.16	46.91
119	25.16	Inf
120	25.16	Inf
121	25.16	Inf
122	25.16	Inf
123	25.16	Inf
124	25.16	Inf
125	25.16	Inf
126	25.16	Inf
127	25.16	Inf
128	25.16	Inf
129	25.16	Inf
130	25.16	Inf
131	25.16	Inf
132	25.16	Inf
133	25.16	Inf
134	25.16	Inf
135	25.16	Inf
136	25.16	Inf
137	25.16	Inf
138	25.16	Inf
139	25.16	Inf

Generation	Best f(x)	Mean f(x)
140	25.16	Inf
141	25.16	Inf
142	25.16	Inf
143	25.16	Inf
144	25.16	Inf
145	25.16	Inf
146	25.16	Inf
147	25.16	Inf
148	25.16	Inf
149	25.16	Inf
150	25.16	Inf
151	25.16	Inf
152	25.16	76.53
153	25.16	Inf
154	25.16	Inf
155	25.16	Inf
156	25.16	Inf
157	25.16	Inf
158	25.16	Inf
159	25.16	Inf
160	25.16	Inf
161	25.16	Inf
162	25.16	Inf
163	25.16	Inf
164	25.16	Inf
165	25.16	Inf
166	25.16	Inf
167	25.16	Inf
168	25.16	Inf
169	25.16	Inf

B.4.2 CASE 4b: Loss of one injection meter

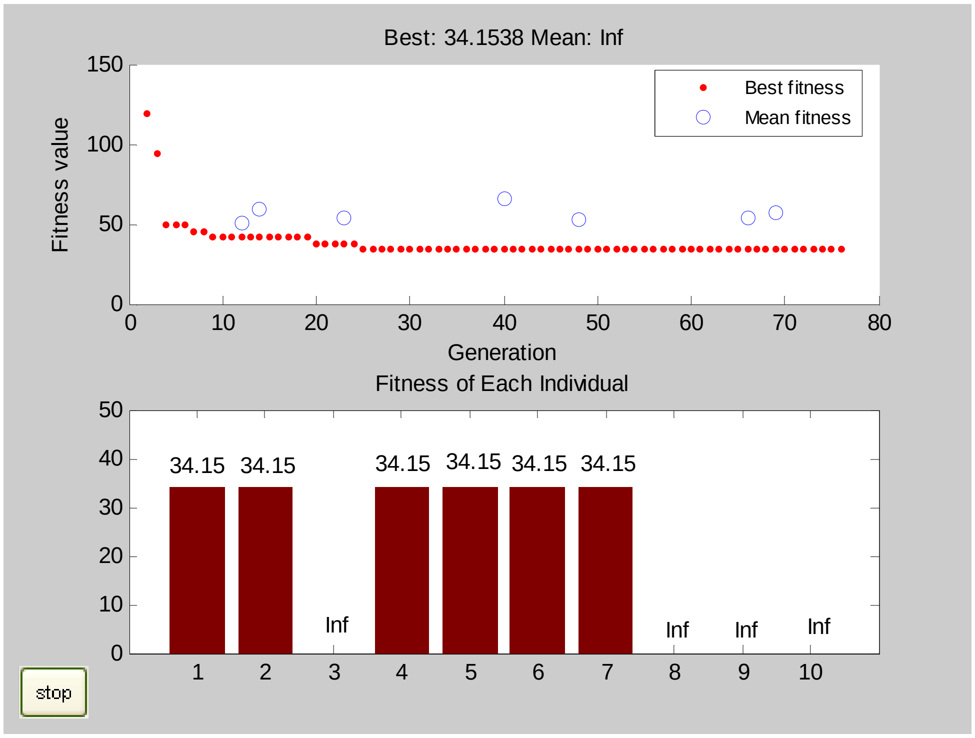

Figure B.7 MATLAB Plot of Fitness Function Values – Case 4b

Table B.8 Genetic Algorithm Output for Case 4b

Generation	Best f(x)	Mean f(x)	Generation	Best f(x)	Mean f(x)
1	Inf	Inf	16	41.58	Inf
2	118.7	Inf	17	41.58	Inf
3	94.11	Inf	18	41.58	Inf
4	49.38	Inf	19	41.58	Inf
5	49.38	Inf	20	37.96	Inf
6	49.38	Inf	21	37.96	Inf
7	45.58	Inf	22	37.96	Inf
8	45.58	Inf	23	37.96	54.23
9	41.58	Inf	24	37.77	Inf
10	41.58	Inf	25	34.15	Inf
11	41.58	Inf	26	34.15	Inf
12	41.58	50.5	27	34.15	Inf
13	41.58	Inf	28	34.15	Inf
14	41.58	59.06	29	34.15	Inf
15	41.58	Inf	30	34.15	Inf

Generation	Best f(x)	Mean f(x)
31	34.15	Inf
32	34.15	Inf
33	34.15	Inf
34	34.15	Inf
35	34.15	Inf
36	34.15	Inf
37	34.15	Inf
38	34.15	Inf
39	34.15	Inf
40	34.15	65.43
41	34.15	Inf
42	34.15	Inf
43	34.15	Inf
44	34.15	Inf
45	34.15	Inf
46	34.15	Inf
47	34.15	Inf
48	34.15	52.26
49	34.15	Inf
50	34.15	Inf
51	34.15	Inf
52	34.15	Inf
53	34.15	Inf
54	34.15	Inf

Generation	Best f(x)	Mean f(x)
55	34.15	Inf
56	34.15	Inf
57	34.15	Inf
58	34.15	Inf
59	34.15	Inf
60	34.15	Inf
61	34.15	Inf
62	34.15	Inf
63	34.15	Inf
64	34.15	Inf
65	34.15	Inf
66	34.15	53.38
67	34.15	Inf
68	34.15	Inf
69	34.15	57.02
70	34.15	Inf
71	34.15	Inf
72	34.15	Inf
73	34.15	Inf
74	34.15	Inf
75	34.15	Inf
76	34.15	Inf

B.5 CASE 5: Minimum number of meters with emphasis to perform second order contingency analysis

B.5.1 CASE 5a: Loss of two flow meters

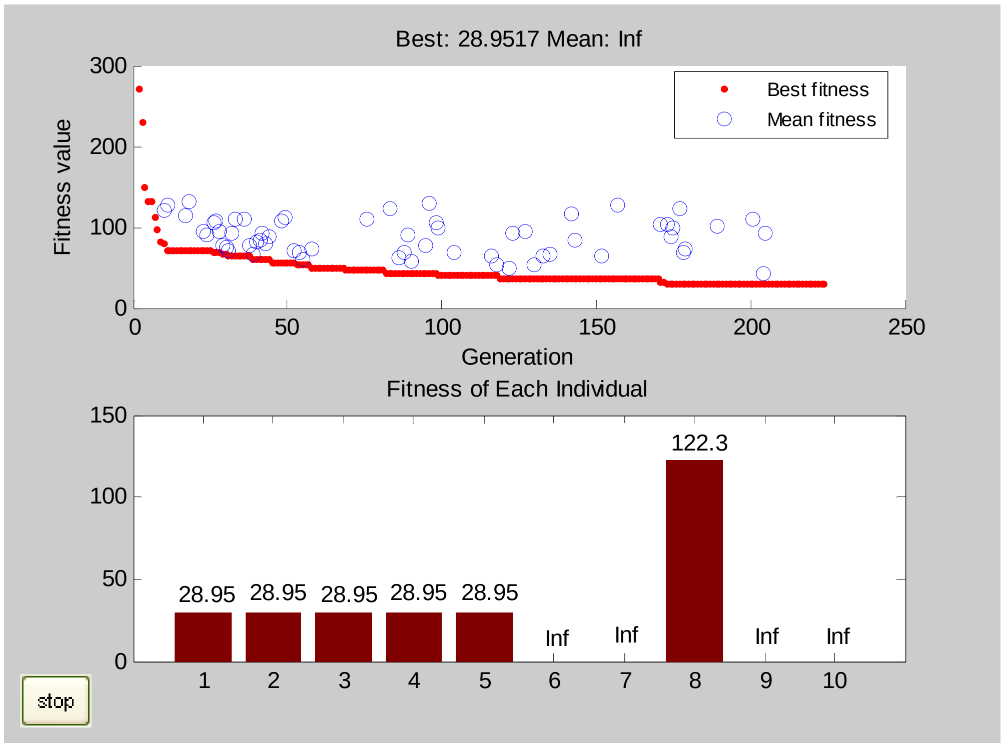

Figure B.8 MATLAB Plot of Fitness Function Values – Case 5a

Table B.9 Genetic Algorithm Output for Case 5a

Generation	Best f(x)	Mean f(x)
1	Inf	Inf
2	270	Inf
3	229.6	Inf
4	149	Inf
5	132.4	Inf
6	132.4	Inf
7	111.1	Inf
8	97.48	Inf
9	81.8	Inf
10	80.11	120.3
11	70.74	127.4
12	69.86	Inf
13	69.86	Inf
14	69.86	Inf
15	69.86	Inf
16	69.86	Inf
17	69.86	114.4
18	69.86	131.9

Generation	Best f(x)	Mean f(x)
19	69.86	Inf
20	69.86	Inf
21	69.86	Inf
22	69.86	Inf
23	69.86	93.82
24	69.86	90.55
25	69.86	Inf
26	67.61	106.1
27	67.61	108.2
28	67.61	93.51
29	67.26	77.59
30	67.26	74.27
31	63.85	71.58
32	63.85	91.77
33	63.85	110.3
34	63.85	Inf
35	63.85	Inf
36	63.85	110.6
37	63.85	Inf
38	63.85	76.89
39	59.56	66.95
40	59.56	80.46
41	59.56	83.94
42	59.56	93.35
43	59.56	79.8
44	59.56	87.77
45	55.83	Inf
46	55.83	Inf
47	55.83	Inf
48	55.83	107.4
49	55.83	111.5
50	55.83	Inf
51	55.83	Inf
52	55.83	70.32
53	52.37	Inf
54	52.37	67.55
55	52.37	59.46
56	52.37	Inf
57	52.37	Inf
58	49.12	73.28
59	49.12	Inf
60	49.12	Inf
61	49.12	Inf
62	49.12	Inf
63	49.12	Inf

Generation	Best f(x)	Mean f(x)
64	49.12	Inf
65	49.12	Inf
66	49.12	Inf
67	49.12	Inf
68	49.12	Inf
69	45.89	Inf
70	45.89	Inf
71	45.89	Inf
72	45.89	Inf
73	45.89	Inf
74	45.89	Inf
75	45.89	Inf
76	45.89	110.4
77	45.89	Inf
78	45.89	Inf
79	45.89	Inf
80	45.89	Inf
81	45.89	Inf
82	43.39	Inf
83	43.39	123.3
84	43.39	Inf
85	43.39	Inf
86	43.39	62.73
87	43.39	Inf
88	43.39	68.34
89	43.39	90.12
90	43.39	58.16
91	43.39	Inf
92	43.39	Inf
93	43.39	Inf
94	43.39	Inf
95	43.39	77.14
96	43.39	129.8
97	42.45	Inf
98	42.45	106.2
99	39.21	99.76
100	39.21	Inf
101	39.21	Inf
102	39.21	Inf
103	39.21	Inf
104	39.21	67.74
105	39.21	Inf
106	39.21	Inf
107	39.21	Inf
108	39.21	Inf

Generation	Best f(x)	Mean f(x)
109	39.21	Inf
110	39.21	Inf
111	39.21	Inf
112	39.21	Inf
113	39.21	Inf
114	39.21	Inf
115	39.21	Inf
116	39.21	64.3
117	39.21	Inf
118	39.21	53.14
119	35.68	Inf
120	35.68	Inf
121	35.68	Inf
122	35.68	49.91
123	35.68	92.3
124	35.31	Inf
125	35.31	Inf
126	35.31	Inf
127	35.31	95.19
128	35.31	Inf
129	35.31	Inf
130	35.31	52.23
131	35.31	Inf
132	35.31	Inf
133	35.31	63.82
134	35.31	Inf
135	35.31	66.9
136	35.31	Inf
137	35.31	Inf
138	35.31	Inf
139	35.31	Inf
140	35.31	Inf
141	35.31	Inf
142	35.31	115.4
143	35.31	83.19
144	35.31	Inf
145	35.31	Inf
146	35.31	Inf
147	35.31	Inf
148	35.31	Inf
149	35.31	Inf
150	35.31	Inf
151	35.31	Inf
152	35.31	63.64
153	35.31	Inf

Generation	Best f(x)	Mean f(x)
154	35.31	Inf
155	35.31	Inf
156	35.31	Inf
157	35.31	126.3
158	35.31	Inf
159	35.31	Inf
160	35.31	Inf
161	35.31	Inf
162	35.31	Inf
163	35.31	Inf
164	35.31	Inf
165	35.31	Inf
166	35.31	Inf
167	35.31	Inf
168	35.31	Inf
169	35.31	Inf
170	35.31	Inf
171	32.36	102.9
172	31.74	Inf
173	28.95	103.6
174	28.95	88.3
175	28.95	99.43
176	28.95	Inf
177	28.95	121.9
178	28.95	69.03
179	28.95	73.8
180	28.95	Inf
181	28.95	Inf
182	28.95	Inf
183	28.95	Inf
184	28.95	Inf
185	28.95	Inf
186	28.95	Inf
187	28.95	Inf
188	28.95	Inf
189	28.95	101.1
190	28.95	Inf
191	28.95	Inf
192	28.95	Inf
193	28.95	Inf
194	28.95	Inf
195	28.95	Inf
196	28.95	Inf
197	28.95	Inf
198	28.95	Inf

Generation	Best f(x)	Mean f(x)
199	28.95	Inf
200	28.95	Inf
201	28.95	109.9
202	28.95	Inf
203	28.95	Inf
204	28.95	41.78
205	28.95	92.98
206	28.95	Inf
207	28.95	Inf
208	28.95	Inf
209	28.95	Inf
210	28.95	Inf
211	28.95	Inf
212	28.95	Inf
213	28.95	Inf
214	28.95	Inf
215	28.95	Inf
216	28.95	Inf
217	28.95	Inf
218	28.95	Inf
219	28.95	Inf
220	28.95	Inf
221	28.95	Inf
222	28.95	Inf
223	28.95	Inf
224	28.95	Inf

B.5.2 CASE 5b: Loss of two injection meters

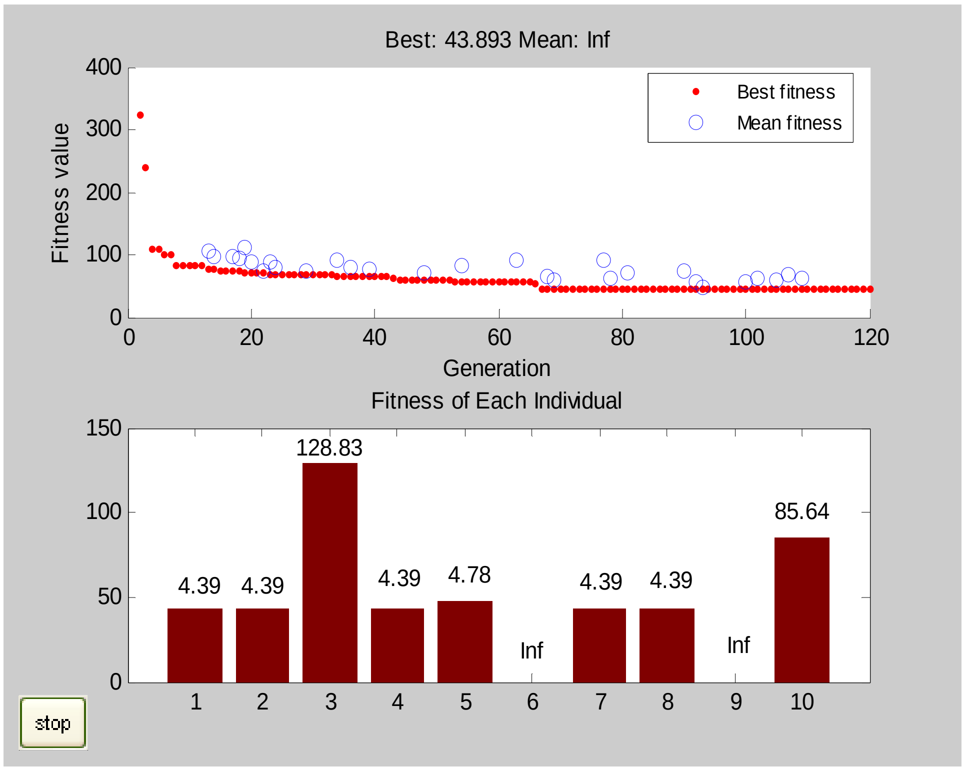

Figure B.9 MATLAB Plot of Fitness Function Values – Case 5b

Table B.10 Genetic Algorithm Output for Case 5b

Generation	Best f(x)	Mean f(x)
1	Inf	Inf
2	322.2	Inf
3	238.6	Inf
4	110	Inf
5	110	Inf
6	99.98	Inf
7	99.98	Inf
8	83.15	Inf
9	83.15	Inf
10	83.15	Inf
11	83.15	Inf
12	83.15	Inf
13	75.86	106.7
14	75.86	98.1
15	74.38	Inf
16	74.38	Inf
17	74.38	97.22
18	74.28	94.02
19	71.59	112.2
20	70.57	87.47
21	70.57	Inf
22	70.03	73.44
23	69.26	89.78
24	69.26	80.78
25	69.26	Inf
26	69.26	Inf
27	69.26	Inf
28	69.26	Inf
29	69.09	75.13
30	69.09	Inf
31	69.09	Inf
32	69.09	Inf
33	69.09	Inf
34	66.42	91.29
35	66.42	Inf
36	66.15	78.31
37	66.15	Inf
38	66.15	Inf
39	66.15	76.78
40	64.13	Inf
41	64.13	Inf
42	64.13	Inf
43	61.12	Inf
44	59.22	Inf
45	59.22	Inf
46	59.22	Inf
47	59.22	Inf
48	59.22	71
49	59.22	Inf
50	59.22	Inf
51	59.22	Inf
52	59.22	Inf
53	55.25	Inf
54	55.25	83.18
55	55.25	Inf
56	55.25	Inf
57	55.25	Inf
58	55.25	Inf
59	55.25	Inf
60	55.25	Inf
61	55.25	Inf
62	55.25	Inf
63	55.25	91.07
64	55.25	Inf
65	55.25	Inf
66	52.21	Inf
67	45.09	Inf
68	45.09	65.09
69	43.89	60.44
70	43.89	Inf
71	43.89	Inf
72	43.89	Inf
73	43.89	Inf
74	43.89	Inf
75	43.89	Inf
76	43.89	Inf
77	43.89	90.4
78	43.89	62.55
79	43.89	Inf
80	43.89	Inf
81	43.89	70.35
82	43.89	Inf
83	43.89	Inf
84	43.89	Inf
85	43.89	Inf
86	43.89	Inf

Generation	Best f(x)	Mean f(x)
87	43.89	Inf
88	43.89	Inf
89	43.89	Inf
90	43.89	73.41
91	43.89	Inf
92	43.89	57.57
93	43.89	48.75
94	43.89	Inf
95	43.89	Inf
96	43.89	Inf
97	43.89	Inf
98	43.89	Inf
99	43.89	Inf
100	43.89	57.2
101	43.89	Inf
102	43.89	61.56
103	43.89	Inf
104	43.89	Inf

Generation	Best f(x)	Mean f(x)
105	43.89	58.31
106	43.89	Inf
107	43.89	69.16
108	43.89	Inf
109	43.89	63.21
110	43.89	Inf
111	43.89	Inf
112	43.89	Inf
113	43.89	Inf
114	43.89	Inf
115	43.89	Inf
116	43.89	Inf
117	43.89	Inf
118	43.89	Inf
119	43.89	Inf
120	43.89	Inf

B.5.3 CASE 5c: Loss of one flow and one injection meters

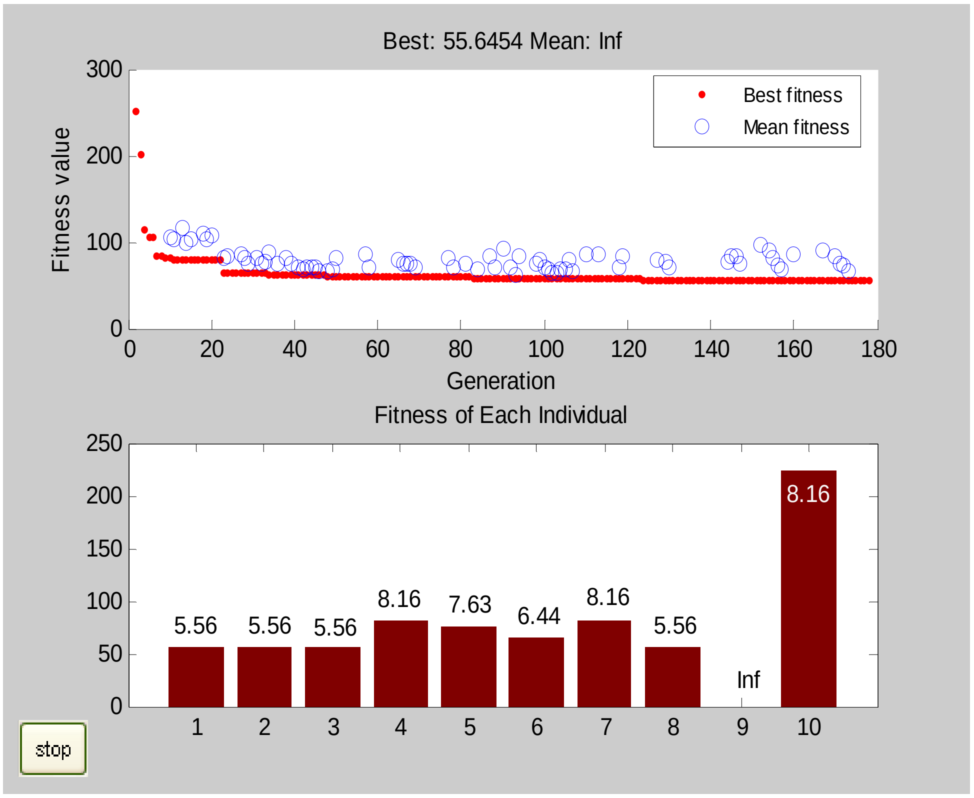

Figure B.10 MATLAB Plot of Fitness Function Values – Case 5c

Table B.11 Genetic Algorithm Output for Case 5c

Generation	Best f(x)	Mean f(x)	Generation	Best f(x)	Mean f(x)
1	Inf	Inf	15	79.07	103.2
2	252	Inf	16	79.07	Inf
3	200.8	Inf	17	79.07	Inf
4	114.4	Inf	18	79.07	109.6
5	105.7	Inf	19	79.07	102.2
6	105.7	Inf	20	79.07	107.1
7	83.03	Inf	21	79.07	Inf
8	83.03	Inf	22	79.07	Inf
9	81.22	Inf	23	65.08	82.04
10	81.22	105.3	24	65.08	84.16
11	79.15	103	25	65.08	Inf
12	79.07	Inf	26	65.08	Inf
13	79.07	115.6	27	65.08	85.36
14	79.07	99.46	28	65.08	80.84

Generation	Best f(x)	Mean f(x)
29	65.08	74.84
30	65.08	Inf
31	65.08	82.37
32	65.08	75.64
33	65.08	78.11
34	61.74	87.77
35	61.74	Inf
36	61.74	75.2
37	61.74	Inf
38	61.74	80.96
39	61.74	74.74
40	61.74	Inf
41	61.74	69.85
42	61.74	69.04
43	61.74	69.57
44	61.74	70.02
45	61.74	71.24
46	61.74	66.4
47	61.74	Inf
48	60.56	67.09
49	60.56	69.11
50	60.56	82.58
51	60.56	Inf
52	60.56	Inf
53	60.56	Inf
54	60.56	Inf
55	60.56	Inf
56	59.87	Inf
57	59.87	85.41
58	59.14	70.43
59	59.14	Inf
60	59.14	Inf
61	59.14	Inf
62	59.14	Inf
63	59.14	Inf
64	59.14	Inf
65	59.14	80.34
66	59.14	74.7
67	59.14	74.4
68	59.14	74.16
69	59.14	70.39
70	59.14	Inf
71	59.14	Inf
72	59.14	Inf
73	59.14	Inf
74	59.14	Inf
75	59.14	Inf
76	59.14	Inf
77	59.14	82.6
78	59.14	70.15
79	59.14	Inf
80	59.14	Inf
81	59.14	74.95
82	59.14	Inf
83	57.6	Inf
84	57.6	69.48
85	57.6	Inf
86	57.6	Inf
87	57.6	84.6
88	57.6	71.52
89	57.6	Inf
90	57.6	92.1
91	57.6	Inf
92	57.6	70.81
93	57.6	60.9
94	57.6	84.21
95	57.6	Inf
96	57.6	Inf
97	57.6	Inf
98	57.6	76.04
99	57.6	79.12
100	57.6	70.4
101	57.6	69.38
102	57.6	63.74
103	57.6	64.87
104	57.6	67.78
105	57.6	67.45
106	57.6	79.79
107	57.6	65.45
108	57.6	Inf
109	56.83	Inf
110	56.83	86.59
111	56.83	Inf
112	56.83	Inf
113	56.83	85.94
114	56.83	Inf
115	56.83	Inf
116	56.83	Inf
117	56.83	Inf
118	56.83	71.12

Generation	Best f(x)	Mean f(x)
119	56.83	84.63
120	56.83	Inf
121	56.83	Inf
122	56.83	Inf
123	56.83	Inf
124	55.98	Inf
125	55.98	Inf
126	55.98	Inf
127	55.65	80.42
128	55.65	Inf
129	55.65	76.45
130	55.65	70.38
131	55.65	Inf
132	55.65	Inf
133	55.65	Inf
134	55.65	Inf
135	55.65	Inf
136	55.65	Inf
137	55.65	Inf
138	55.65	Inf
139	55.65	Inf
140	55.65	Inf
141	55.65	Inf
142	55.65	Inf
143	55.65	Inf
144	55.65	77.86
145	55.65	82.68
146	55.65	83.91
147	55.65	74.04
148	55.65	Inf
149	55.65	Inf
150	55.65	Inf
151	55.65	Inf
152	55.65	96.15
153	55.65	Inf
154	55.65	90.26
155	55.65	81.14
156	55.65	72.24
157	55.65	68
158	55.65	Inf
159	55.65	Inf
160	55.65	85.15
161	55.65	Inf
162	55.65	Inf
163	55.65	Inf
164	55.65	Inf
165	55.65	Inf
166	55.65	Inf
167	55.65	91.23
168	55.65	Inf
169	55.65	Inf
170	55.65	83.32
171	55.65	74.75
172	55.65	72.91
173	55.65	67.27
174	55.65	Inf
175	55.65	Inf
176	55.65	Inf
177	55.65	Inf
178	55.65	Inf

APPENDIX C

MATLAB INPUT FILES FOR 3-BUS, 6-BUS AND 18-BUS TEST SYSTEMS

C.1 3-Bus Test System

```
% 3-BUS, 3-BRANCH TEST SYSTEM

clear all;
dbstop if error;

profile on;
% Circuit description.  circ_top is a structure that contains all the
% information pertaining to a specific circuit topology.  It contains
% information about the number of buses, the number of branches connecting
% buses, how the branches are connected between buses, and also all
% possible meter locations - both Power flow meters and Power injection
% meters - in the circuit.
circ_top.num_bus = 3;
circ_top.num_branch = 3;
circ_top.branch(1).bus1 = 1;
circ_top.branch(1).bus2 = 2;
circ_top.branch(2).bus1 = 2;
circ_top.branch(2).bus2 = 3;
circ_top.branch(3).bus1 = 3;
circ_top.branch(3).bus2 = 1;

% All possible measurements.  There are 3 Power injection measurements and 3
power
% flow measurements possible for this topology.  This will vary from
% topology to topology.

circ_top.Power_inj(1).bus = 1;
circ_top.Power_inj(2).bus = 2;
circ_top.Power_inj(3).bus = 3;

circ_top.Power_flow(1).bus1 = 1;
circ_top.Power_flow(1).bus2 = 2;
circ_top.Power_flow(2).bus1 = 2;
circ_top.Power_flow(2).bus2 = 3;
circ_top.Power_flow(3).bus1 = 3;
circ_top.Power_flow(3).bus2 = 1;

% Inputs above.  All the code below is derived from the inputs above.
% Maximum number of power injection measurements and power flow
% measurements possible.
circ_top.max_pwr_inj_meas = numel(circ_top.Power_inj);
circ_top.max_pwr_flow_meas = numel(circ_top.Power_flow);

% Maximum measurements possible.  This dictates the number of genes in the
% chromosome (in the genetic algorithm).  Each gene in the chromosome
% corresponds to one of these 6 meters mentioned above.  So there will be
% a total of 2^6 chromosomes, since each of the genes can either be 0 or 1
% corresponding to whether the meter represented by that particular gene is
% absent or present.
max_meas_possible = circ_top.max_pwr_inj_meas + circ_top.max_pwr_flow_meas;
num_genes = max_meas_possible;
num_chromosomes = 2^(num_genes);

% The chromosomes are expressed as logical vectors of size equal to the
% number of genes.  chrom_log_str(1) corresponds to the first Power
% injection meter, and chrom_log_str(end) corresponds to the last Power
% flow meter.  Thus, the power injection meters are arranged first followed
% by the power flow meters in numerical order.  This way a value 0 or 1
% (true or false) can be assigned to each of the num_genes meters in the
% system corresponding to whether the meter is present or not.  Thus each
% "meter arrangement" can be uniquely represented by a value of
% chrom_log_str.

% Set the option 'PopulationType' to 'bitString' to indicate that the
% population is bit string.  Choose the mutation function as uniform.  Set
% options for the Genetic algorithm.
options =  gaoptimset('PopulationType','bitString',...
 'MutationFcn',{@mutationuniform,0.1});
options =  gaoptimset(options,'StallGenLimit',50);
options =  gaoptimset(options,'StallTimeLimit',inf);
options =  gaoptimset(options,'Generations',1000);
options =  gaoptimset(options,'Display', 'iter');
options =  gaoptimset(options,'PopulationSize', 10);

% Plot functions
options =  gaoptimset(options,'PlotFcns',{@gaplotbestf1,@gaplotscores});
% options =  gaoptimset(options,'PlotFcns',{@gaplotbestf});

rand('state', 71); % These two commands are only included to
randn('state', 59); % make the results reproducible.

FitnessFcn = {@meter_cost_fcn, circ_top};
GenomeLength = num_genes;
[x, val, reason, output, finalpop, finalscores] =
ga(FitnessFcn,GenomeLength,options);
fprintf(1,'Best solution: %s Min Fitness Fcn val: %d\n',char(x + '0'),val);
meters_present = find(x);
disp(['Best solution has meters ' num2str(meters_present) ' present']);
disp(['Total number of meters present = ' num2str(length(meters_present))]);
fprintf(1,'Final Population and Corresponding Scores \n');
for k = 1:size(finalpop, 1)
    fprintf(1,'%s\t%d\n',char(finalpop(k,:) + '0'), finalscores(k));
end

figure(gcf); subplot(2,1,1); axis auto
profile viewer;
return;
```

C.2 6-Bus Test System

```
% 6-BUS 6 BRANCH TEST SYSTEM

clear all;
dbstop if error;

profile on;
% Circuit description.  circ_top is a structure that contains all the
% information pertaining to a specific circuit topology.  It contains
% information about the number of buses, the number of branches connecting
% buses, how the branches are connected between buses, and also all
% possible meter locations - both Power flow meters and Power injection
% meters - in the circuit.
circ_top.num_bus = 6;
circ_top.num_branch = 6; %8;
circ_top.branch(1).bus1 = 1;
circ_top.branch(1).bus2 = 2;
circ_top.branch(2).bus1 = 2;
circ_top.branch(2).bus2 = 3;
circ_top.branch(3).bus1 = 3;
circ_top.branch(3).bus2 = 4;
circ_top.branch(4).bus1 = 4;
circ_top.branch(4).bus2 = 5;
circ_top.branch(5).bus1 = 5;
circ_top.branch(5).bus2 = 6;
circ_top.branch(6).bus1 = 6;
circ_top.branch(6).bus2 = 1;

% All possible measurements.  There are 6 Power injection measurements and 6 %
power flow measurements possible for this topology.  This will vary from
% topology to topology.
circ_top.Power_inj(1).bus = 1;
circ_top.Power_inj(2).bus = 2;
circ_top.Power_inj(3).bus = 3;
circ_top.Power_inj(4).bus = 4;
circ_top.Power_inj(5).bus = 5;
circ_top.Power_inj(6).bus = 6;

circ_top.Power_flow(1).bus1 = 1;
circ_top.Power_flow(1).bus2 = 2;
circ_top.Power_flow(2).bus1 = 2;
circ_top.Power_flow(2).bus2 = 3;
circ_top.Power_flow(3).bus1 = 3;
circ_top.Power_flow(3).bus2 = 4;
circ_top.Power_flow(4).bus1 = 4;
circ_top.Power_flow(4).bus2 = 5;
circ_top.Power_flow(5).bus1 = 5;
circ_top.Power_flow(5).bus2 = 6;
circ_top.Power_flow(6).bus1 = 1;
circ_top.Power_flow(6).bus2 = 6;

% Inputs above.  All the code below is derived from the inputs above.
% Maximum number of power injection measurements and power flow
% measurements possible.
circ_top.max_pwr_inj_meas = numel(circ_top.Power_inj);
circ_top.max_pwr_flow_meas = numel(circ_top.Power_flow);

% Maximum measurements possible.  This dictates the number of genes in the
% chromosome (in the genetic algorithm).  Each gene in the chromosome
% corresponds to one of these 12 meters mentioned above.  So there will be
% a total of 2^12 chromosomes, since each of the genes can either be 0 or 1
% corresponding to whether the meter represented by that particular gene is
% absent or present.
max_meas_possible = circ_top.max_pwr_inj_meas + circ_top.max_pwr_flow_meas;
num_genes = max_meas_possible;
num_chromosomes = 2^(num_genes);

% The chromosomes are expressed as logical vectors of size equal to the
% number of genes.  chrom_log_str(1) corresponds to the first Power
% injection meter, and chrom_log_str(end) corresponds to the last Power
% flow meter.  Thus, the power injection meters are arranged first followed
% by the power flow meters in numerical order.  This way a value 0 or 1
% (true or false) can be assigned to each of the num_genes meters in he
% system corresponding to whether the meter is present or not.  Thus each
% "meter arrangement" can be uniquely represented by a value of
% chrom_log_str.

% Set the option 'PopulationType' to 'bitString' to indicate that the
% population is bit string.  Choose the mutation function as uniform.  Set
% options for the Genetic algorithm.
options =  gaoptimset('PopulationType','bitString',...
 'MutationFcn',{@mutationuniform,0.1});
options =  gaoptimset(options,'StallGenLimit',50);
options =  gaoptimset(options,'StallTimeLimit',inf);
options =  gaoptimset(options,'Generations',1000);
options =  gaoptimset(options,'Display', 'iter');
options =  gaoptimset(options,'PopulationSize', 10);

% Plot functions
options =  gaoptimset(options,'PlotFcns',{@gaplotbestf1,@gaplotscores});
% options =  gaoptimset(options,'PlotFcns',{@gaplotbestf});

rand('state', 71); % These two commands are only included to
randn('state', 59); % make the results reproducible.

FitnessFcn = {@meter_cost_fcn, circ_top};
GenomeLength = num_genes;
[x, val, reason, output, finalpop, finalscores] =
ga(FitnessFcn,GenomeLength,options);
fprintf(1,'Best solution: %s Min Fitness Fcn val: %d\n',char(x + '0'),val);
meters_present = find(x);
disp(['Best solution has meters ' num2str(meters_present) ' present']);
disp(['Total number of meters present = ' num2str(length(meters_present))]);
fprintf(1,'Final Population and Corresponding Scores \n');
for k = 1:size(finalpop, 1)
    fprintf(1,'%s\t%d\n',char(finalpop(k,:) + '0'), finalscores(k));
end

figure(gcf); subplot(2,1,1); axis auto
profile viewer;
return;
```

C.3 18-Bus Test System

```
%  18-BUS, 17-BRANCH TEST SYSTEM

clear all;
dbstop if error;

profile on;
% Circuit description.  circ_top is a structure that contains all the
% information pertaining to a specific circuit topology.  It contains
% information about the number of buses, the number of branches connecting
% buses, how the branches are connected between buses, and also all
% possible meter locations - both Power flow meters and Power injection
% meters - in the circuit.
circ_top.num_bus = 18;
circ_top.num_branch = 17;
circ_top.branch(1).bus1 = 1;
circ_top.branch(1).bus2 = 5;
circ_top.branch(2).bus1 = 2;
circ_top.branch(2).bus2 = 5;
circ_top.branch(3).bus1 = 3;
circ_top.branch(3).bus2 = 6;
circ_top.branch(4).bus1 = 4;
circ_top.branch(4).bus2 = 6;
circ_top.branch(5).bus1 = 5;
circ_top.branch(5).bus2 = 6;
circ_top.branch(6).bus1 = 5;
circ_top.branch(6).bus2 = 7;
circ_top.branch(7).bus1 = 7;
circ_top.branch(7).bus2 = 8;
circ_top.branch(8).bus1 = 8;
circ_top.branch(8).bus2 = 9;
circ_top.branch(9).bus1 = 5;
circ_top.branch(9).bus2 = 10;
circ_top.branch(10).bus1 = 10;
circ_top.branch(10).bus2 = 11;
circ_top.branch(11).bus1 = 11;
circ_top.branch(11).bus2 = 12;
circ_top.branch(12).bus1 = 6;
circ_top.branch(12).bus2 = 13;
circ_top.branch(13).bus1 = 13;
circ_top.branch(13).bus2 = 14;
circ_top.branch(14).bus1 = 14;
circ_top.branch(14).bus2 = 15;
circ_top.branch(15).bus1 = 6;
circ_top.branch(15).bus2 = 16;
circ_top.branch(16).bus1 = 16;
circ_top.branch(16).bus2 = 17;
circ_top.branch(17).bus1 = 17;
circ_top.branch(17).bus2 = 18;

% All possible measurements.  There are 18 Power injection measurements and 17
power
% flow measurements possible for this topology.  This will vary from
% topology to topology.
circ_top.Power_inj(1).bus = 1;
circ_top.Power_inj(2).bus = 2;
circ_top.Power_inj(3).bus = 3;
circ_top.Power_inj(4).bus = 4;
circ_top.Power_inj(5).bus = 5;
circ_top.Power_inj(6).bus = 6;
circ_top.Power_inj(7).bus = 7;
circ_top.Power_inj(8).bus = 8;
circ_top.Power_inj(9).bus = 9;
circ_top.Power_inj(10).bus = 10;
circ_top.Power_inj(11).bus = 11;
circ_top.Power_inj(12).bus = 12;
circ_top.Power_inj(13).bus = 13;
circ_top.Power_inj(14).bus = 14;
circ_top.Power_inj(15).bus = 15;
circ_top.Power_inj(16).bus = 16;
circ_top.Power_inj(17).bus = 17;
circ_top.Power_inj(18).bus = 18;

circ_top.Power_flow(1).bus1 = 1;
circ_top.Power_flow(1).bus2 = 5;
circ_top.Power_flow(2).bus1 = 2;
circ_top.Power_flow(2).bus2 = 5;
circ_top.Power_flow(3).bus1 = 3;
circ_top.Power_flow(3).bus2 = 6;
circ_top.Power_flow(4).bus1 = 4;
circ_top.Power_flow(4).bus2 = 6;
circ_top.Power_flow(5).bus1 = 5;
circ_top.Power_flow(5).bus2 = 6;
circ_top.Power_flow(6).bus1 = 5;
circ_top.Power_flow(6).bus2 = 7;
circ_top.Power_flow(7).bus1 = 7;
circ_top.Power_flow(7).bus2 = 8;
circ_top.Power_flow(8).bus1 = 8;
circ_top.Power_flow(8).bus2 = 9;
circ_top.Power_flow(9).bus1 = 5;
circ_top.Power_flow(9).bus2 = 10;
circ_top.Power_flow(10).bus1 = 10;
circ_top.Power_flow(10).bus2 = 11;
circ_top.Power_flow(11).bus1 = 11;
circ_top.Power_flow(11).bus2 = 12;
circ_top.Power_flow(12).bus1 = 6;
circ_top.Power_flow(12).bus2 = 13;
circ_top.Power_flow(13).bus1 = 13;
circ_top.Power_flow(13).bus2 = 14;
circ_top.Power_flow(14).bus1 = 14;
circ_top.Power_flow(14).bus2 = 15;
circ_top.Power_flow(15).bus1 = 6;
circ_top.Power_flow(15).bus2 = 16;
circ_top.Power_flow(16).bus1 = 16;
circ_top.Power_flow(16).bus2 = 17;
circ_top.Power_flow(17).bus1 = 17;
circ_top.Power_flow(17).bus2 = 18;

% Inputs above.  All the code below is derived from the inputs above.
% Maximum number of power injection measurements and power flow
% measurements possible.
circ_top.max_pwr_inj_meas = numel(circ_top.Power_inj);
circ_top.max_pwr_flow_meas = numel(circ_top.Power_flow);

% Maximum measurements possible.  This dictates the number of genes in the
% chromosome (in the genetic algorithm).  Each gene in the chromosome
% corresponds to one of these 35 meters mentioned above.  So there will be
% a total of 2^35 chromosomes, since each of the genes can either be 0 or 1
% corresponding to whether the meter represented by that particular gene is
% absent or present.
max_meas_possible = circ_top.max_pwr_inj_meas + circ_top.max_pwr_flow_meas;
num_genes = max_meas_possible;
num_chromosomes = 2^(num_genes);

% The chromosomes are expressed as logical vectors of size equal to the
% number of genes.  chrom_log_str(1) corresponds to the first Power
% injection meter, and chrom_log_str(end) corresponds to the last Power
% flow meter.  Thus, the power injection meters are arranged first followed
% by the power flow meters in numerical order.  This way a value 0 or 1
% (true or false) can be assigned to each of the num_genes meters in the
% system corresponding to whether the meter is present or not.  Thus each
% "meter arrangement" can be uniquely represented by a value of
% chrom_log_str.

% Set the option 'PopulationType' to 'bitString' to indicate that the
% population is bit string.  Choose the mutation function as uniform.  Set
% options for the Genetic algorithm.
options =  gaoptimset('PopulationType','bitString',...
 'MutationFcn',{@mutationuniform,0.1});
```

```
options =  gaoptimset(options,'StallGenLimit',50);
options =  gaoptimset(options,'StallTimeLimit',inf);
options =  gaoptimset(options,'Generations',1000);
options =  gaoptimset(options,'Display', 'iter');
options =  gaoptimset(options,'PopulationSize', 10);

% Add some plot functions
options =  gaoptimset(options,'PlotFcns',{@gaplotbestf1,@gaplotscores});
% options =  gaoptimset(options,'PlotFcns',{@gaplotbestf});

rand('state', 71); % These two commands are only included to
randn('state', 59); % make the results reproducible.


FitnessFcn = {@meter_cost_fcn, circ_top};
GenomeLength = num_genes;
[x, val, reason, output, finalpop, finalscores] =
ga(FitnessFcn,GenomeLength,options);
fprintf(1,'Best solution: %s Min Fitness Fcn val: %d\n',char(x + '0'),val);
meters_present = find(x);
disp(['Best solution has meters ' num2str(meters_present) ' present']);
disp(['Total number of meters present = ' num2str(length(meters_present))]);
fprintf(1,'Final Population and Corresponding Scores \n');
for k = 1:size(finalpop, 1)
    fprintf(1,'%s\t%d\n',char(finalpop(k,:) + '0'), finalscores(k));
end

figure(gcf); subplot(2,1,1); axis auto
profile viewer;
return;
```

APPENDIX D

CODE FOR FITNESS FUNCTION CALCULATION

```
function [meter_cost] = meter_cost_fcn(pop, circ_top)
% This is the fitness function finding routine for the genetic algorithm.
% It takes in the circuit topology, and the population (which is a
% chromosome) and finds the meter cost for that particular meter arrangement.

% Rename pop as chrom_log_str, because the population is just a logical
% vector of 0's and 1's representing whether a meter is absent or present
% respectively at a particular location in the circuit.
chrom_log_str = pop;

% Actual number of measurements... Corresponds to the number of meters
% placed.  Pmeas is a struct that will scan through the whole list of
% power measurements - Power flow and Power injection measurements - and
% depending on which gene in the chromosome is 1 or 'true', pick that
% particular measure as an actual measurement.  This is done for both Pinj
% and Pflow.
Pmeas = struct();
k = 1;
for l = 1:circ_top.max_pwr_inj_meas
    if chrom_log_str(l) == true
        Pmeas.Pinj(k) = circ_top.Power_inj(l);
        k = k+1;
    end
end

k = 1;
for l = 1:circ_top.max_pwr_flow_meas
      if chrom_log_str(l+circ_top.max_pwr_inj_meas) == true
        Pmeas.Pflow(k) = circ_top.Power_flow(l);
        k = k+1;
    end
end

% If there are no Power flow measures, just assign an empty struct to
% Pmeas.Pflow
if isfield(Pmeas,'Pflow') == 0
    Pmeas.Pflow = struct([]);
end
% If there are no Power injection measures, just assign an empty struct to
% Pmeas.inj
if isfield(Pmeas, 'Pinj') == 0
    Pmeas.Pinj = struct([]);
end

% Initialize the weights to be used here.
wts.pen_flow_meter = 1; %1.0; % Penalty for number of flow meters
wts.pen_inj_meter = 2; %4.0; % Penalty for number of injection meters
wts.pen_crit = 10; %1; %0.5; %0.1; % Penalty for critical measurements

pen_first = 0; %3; %3; % Generic penalty factor for FO contingency
pen_second = 0; %124; % Generic penalty factor for FO contingency
prob_fail_flow = 0.5; %0.5; % Fail prob for 1 flow meter. Inv. proportional to
pen_flow_meter
prob_fail_inj = 0.4;% 0.2; % Fail prob for 1 inj meter. Inv. prop to
pen_inj_meter
wts.pen_first_flow = pen_first*prob_fail_flow;
wts.pen_first_inj = pen_first*prob_fail_inj;
wts.pen_second_flow = pen_second*prob_fail_flow*prob_fail_flow;
wts.pen_second_inj = pen_second*prob_fail_inj*prob_fail_inj;
wts.pen_second_mixed = pen_second*prob_fail_inj*prob_fail_flow;
```

```
% Call the function that determines if this system of measurements is
% observable and if it is then it finds the critical measurements.  The
% outputs are obs_flag and Pcrit.  If obs_flag == 0, then the system is not
% observable and Pcrit is an empty struct.  If obs_flag == 1, then the
% system is observable and Pcrit contains the list of critical
% measurements, if any.
[obs_flag, fo_cont_nonobsbranch, so_cont_nonobsbranch, Pcrit] = ...
    FindCriticalMeasurements(circ_top.num_bus,...
                             circ_top.num_branch,...
                             circ_top.branch,...
                             Pmeas,...
                             wts);

% Cost function:  This is where the cost is calculated for a
% particular population/chromosome/meter arrangement (different names
% meaning the same thing).  The eventual output should be in meter_cost,
% because that's what has to be minimized.

if obs_flag == 0
    meter_cost = inf;
else
% keyboard;
    if isfield(Pcrit,'Pflow') == 0
        Pcrit.Pflow = struct([]);
    end
    if isfield(Pcrit, 'Pinj') == 0
        Pcrit.Pinj = struct([]);
    end

    fo_cost_inj = 0;
    if (wts.pen_first_inj && numel(Pmeas.Pinj))
        for l = 1:numel(Pmeas.Pinj)
            fo_cost_inj = fo_cost_inj +
numel(fo_cont_nonobsbranch.meter(l).non_obs_br);
        end
        fo_cost_inj = fo_cost_inj/numel(Pmeas.Pinj);
    end

    fo_cost_flow = 0;
    if (wts.pen_first_flow && numel(Pmeas.Pflow))
        for l = 1:numel(Pmeas.Pflow)
            fo_cost_flow = fo_cost_flow + ...

numel(fo_cont_nonobsbranch.meter(l+numel(Pmeas.Pinj)).non_obs_br);
        end
        fo_cost_flow = fo_cost_flow/numel(Pmeas.Pflow);
    end

    so_cost_inj = 0;
    if (wts.pen_second_inj && numel(Pmeas.Pinj))
        for l=1:numel(Pmeas.Pinj)
            for k=1:numel(Pmeas.Pinj)
                so_cost_inj = so_cost_inj +
numel(so_cont_nonobsbranch.meter(l,k).non_obs_br);
            end
        end
        so_cost_inj = so_cost_inj/(numel(Pmeas.Pinj)*numel(Pmeas.Pinj));
    end

    so_cost_mixed = 0;
    if (wts.pen_second_mixed && numel(Pmeas.Pinj) && numel(Pmeas.Pflow))
        for l=numel(Pmeas.Pinj)+1:(numel(Pmeas.Pinj)+numel(Pmeas.Pflow))
            for k=1:numel(Pmeas.Pinj)
                so_cost_mixed = so_cost_mixed + ...
                    numel(so_cont_nonobsbranch.meter(l,k).non_obs_br) + ...
                    numel(so_cont_nonobsbranch.meter(k,l).non_obs_br);
            end
        end
        so_cost_mixed = so_cost_mixed/(numel(Pmeas.Pinj)*numel(Pmeas.Pflow));
    end

    so_cost_flow = 0;
    if (wts.pen_second_flow && numel(Pmeas.Pflow))
        for l=numel(Pmeas.Pinj)+1:(numel(Pmeas.Pinj)+numel(Pmeas.Pflow))
            for k=numel(Pmeas.Pinj)+1:(numel(Pmeas.Pinj)+numel(Pmeas.Pflow))
                so_cost_flow = so_cost_flow + 
numel(so_cont_nonobsbranch.meter(l,k).non_obs_br);
            end
        end
        so_cost_flow = so_cost_flow/(numel(Pmeas.Pflow)*numel(Pmeas.Pflow));
    end


   meter_cost = wts.pen_inj_meter*numel(Pmeas.Pinj)
         wts.pen_flow_meter*numel(Pmeas.Pflow)...
        + wts.pen_crit*numel(Pcrit.Pflow) + wts.pen_crit*numel(Pcrit.Pinj)...
        + wts.pen_first_inj*fo_cost_inj + wts.pen_first_flow*fo_cost_flow...
        + wts.pen_second_inj*so_cost_inj
        + wts.pen_second_mixed*so_cost_mixed...
        + wts.pen_second_flow*so_cost_flow;
end
return;
```

```
function [is_obs_flag, focont_nonobsbranches, socont_nonobsbranches, Pcrit] =
...
    FindCriticalMeasurements(num_bus, num_branch, b, Pmeas, weights)

display_verbose = 0;
warning('off', 'all');
% This is just a function form of the same routine in test2.m

% Assuming reference bus or the slack bus to be always bus 1.

% Bus incidence matrix A calculation.  The start_bus location for a branch
% has an entry of 1, and the end bus has an entry of -1.  All the other
% entries in the row are zeros.
A = zeros(num_branch, num_bus);
for l = 1:num_branch
    A(l, b(l).bus1) = 1;
    A(l, b(l).bus2) = -1;
end

% Build the Jacobian matrix.  Start out with an all zero matrix with number
% of rows equal to the total meters present, and the number of columns
% equal to the number of buses.
Haa = zeros((numel(Pmeas.Pinj) + numel(Pmeas.Pflow)), num_bus);
% Start first with Power injection meters.
for k=1:(numel(Pmeas.Pinj))
```

```
    for l=1:numel(b)
        if (b(l).bus1 == Pmeas.Pinj(k).bus)
            Haa(k, Pmeas.Pinj(k).bus) = Haa(k, Pmeas.Pinj(k).bus) + 1;
            Haa(k, b(l).bus2) = Haa(k, b(l).bus2) - 1;
        end
        if (b(l).bus2 == Pmeas.Pinj(k).bus)
            Haa(k, Pmeas.Pinj(k).bus) = Haa(k, Pmeas.Pinj(k).bus) + 1;
            Haa(k, b(l).bus1) = Haa(k, b(l).bus1) - 1;
        end
    end
end
% Then come the Power flow measurements.
for k=1:(numel(Pmeas.Pflow))
    Haa(k+numel(Pmeas.Pinj), Pmeas.Pflow(k).bus1) = 1;
    Haa(k+numel(Pmeas.Pinj), Pmeas.Pflow(k).bus2) = -1;
end

% Build states of the system.  theta_t is the set of solutions for the
% equation Haa*x = 0.  It forms the null space of Haa.  The idea is that
% for a system to be observable, if Haa*theta_t = 0, then A*theta_t = 0.
% Else the system is not observable.
theta_t = null(Haa);

% branch flow vector
Pb = A*theta_t;

% Because the calculations have limited precision, the values will never be
% exactly 0.  So, assume that anything less than 10^-6 is 0.
Pb(find(abs(Pb)<1e-6)) = 0;

% For system to be observable Pb =0
if isempty(find(Pb, 1))
    if display_verbose
        disp('System is observable... Finding critical measurements');
    end
    is_obs_flag = 1;
else
    if display_verbose
        disp('System is non-observable...Skipping critical measurements');
    end
    is_obs_flag = 0;
end

% To find critical measurements, the system has to be observable first.
if (is_obs_flag == 1)

    if (weights.pen_first_inj || weights.pen_first_flow)
        [focont_cost, focont_nonobsbranches] = FirstOrderContAnal(A, Haa);
    else
        focont_nonobsbranches = struct([]); % Null struct
    end

    if (weights.pen_second_inj || weights.pen_second_mixed ||
weights.pen_second_flow)
        [socont_cost, socont_nonobsbranches] = SecondOrderContAnal(A, Haa);
    else
        socont_nonobsbranches = struct([]); % Null struct
    end

    if (weights.pen_crit)
```

```
        % Critical measurements calculation
        % Removing first column corresponding to slack bus. Haa_red has
        % (num_bus-1) columns and rows equal to the number of meters placed.
        Haa_red = Haa(:, 2:end);

        % Lower triangular matrix calculation after LU decomposition.  The top
        % square matrix of L becomes L1, and the bottom rows form M2.  Thus, L1
        % is a (num_bus-1) x (num_bus-1), while M2 is (num_meters-num_bus+1) x
        % (num_bus-1)
        [L,U] = lu(Haa_red);
        L1 = L(1:(num_bus-1),:);
        M2 = L(num_bus:end, :);

        T = M2 * inv(L1);
        T(find(abs(T) < 1e-6)) = 0;
        null_col_T = find(sum(T,1) == 0);

        l_pflow = 1;
        l_pinj = 1;
        for k = 1:length(null_col_T)
            H_row_indx = null_col_T(k);
            if display_verbose
                disp(['Row ' num2str(H_row_indx) ' of Haa_red is critical']);
            end
            if H_row_indx <= numel(Pmeas.Pinj)
                if display_verbose
                    disp(['P_inj at bus ' num2str(Pmeas.Pinj(H_row_indx).bus) '
is critical']);
                end
                Pcrit.Pinj(l_pinj) = Pmeas.Pinj(H_row_indx);
                l_pinj = l_pinj+1;
            else
                if display_verbose
                    disp(['P_flow between bus ' num2str(Pmeas.Pflow(H_row_indx-
numel(Pmeas.Pinj)).bus1)...
                            ' and bus ' num2str(Pmeas.Pflow(H_row_indx-
numel(Pmeas.Pinj)).bus2)...
                            ' is critical']);
                end
                Pcrit.Pflow(l_pflow) = Pmeas.Pflow(H_row_indx-
numel(Pmeas.Pinj));
                l_pflow = l_pflow+1;
            end
        end
        if isempty(null_col_T)
            if display_verbose
                disp('No critical measurements found');
            end
            Pcrit.Pinj = struct([]); % Null struct
            Pcrit.Pflow = struct([]); % Null struct
        end
    else
        Pcrit.Pinj = struct([]); % Null struct
        Pcrit.Pflow = struct([]); % Null struct
    end

else % if non-observable
    Pcrit = struct([]); % Null struct
    focont_nonobsbranches = struct([]); % Null struct
    socont_nonobsbranches = struct([]); % Null struct
```

```
end % if (is_obs_flag == 1)


% function [CritMeas] = CriticalMeas(

% This routine does the first order Contingency Analysis.  For a given
% observable system it finds out which branches become unobservable as we
% remove one meter at a time.
function [cost, ContAnal] = FirstOrderContAnal(A, Haa)
cost = 0;

for l = 1:size(Haa, 1)
    % Remove l'th measurement.  G will give the Jacobian for a system with
    % the l'th meter missing
    G = Haa;
    G(l,:) =[];
    % Find the solution set for G*x = 0.
    theta_t = null(G);

    % branch flow vector for the given solution.  It has to be 0 for the
    % system to be observable
    Pb = A*theta_t;
    Pb(find(abs(Pb)<1e-6)) = 0;

    clear non_obs_branches; % important, otherwise gives buggy results.
    for k = 1:size(Pb,2)
        non_obs_branches(k) = numel(find(Pb(:,k)));
    end
    % Find the solution from the solution set which results in maximum
    % number of unobservable branches.
    [max_non_obs_branches, max_indx] = max(non_obs_branches);
    % The unobservable branches when the l'th meter is removed from the
    % measurement system are stored in ContAnal.meter(l).non_obs_br vector.
    % It's a column vector and is passed all the way to the meter_cost
    % function where it can be used appropriately.
    ContAnal.meter(l).non_obs_br = find(Pb(:,max_indx));

    % For checking purposes calculate the total number of unobservable
    % branches for each missing measurement.
    cost = cost + max_non_obs_branches;
end

return;

% This routine does the Second order Contingency Analysis.  For a given
% observable system it finds out which branches become unobservable as we
% remove two meters at a time.
function [cost, ContAnal] = SecondOrderContAnal(A, Haa)
cost = 0;
% keyboard;
for k = 1:size(Haa, 1)
    for l = 1:size(Haa, 1)
        if l == k
            ContAnal.meter(k,l).non_obs_br = [];
        else
            % Remove l'th measurement.  G will give the Jacobian for a system
with
            % the l'th meter missing
            G = Haa;
            G([k l],:) = [];
            % Find the solution set for G*x = 0.
            theta_t = null(G);

            % branch flow vector for the given solution.  It has to be 0 for
            %the system to be observable
            Pb = A*theta_t;
            Pb(find(abs(Pb)<1e-6)) = 0;

            clear non_obs_branches; % important, otherwise gives buggy results.
            for m = 1:size(Pb,2)
                non_obs_branches(m) = numel(find(Pb(:,m)));
            end
            % Find the solution from the solution set which results in maximum
            % number of unobservable branches.
            [max_non_obs_branches, max_indx] = max(non_obs_branches);
            % The unobservable branches when the l'th meter is removed from the
            % measurement system are stored in ContAnal.meter(l).non_obs_br
vector. It's a column vector and is passed all the way to the meter_cost
            % function where it can be used appropriately.
            ContAnal.meter(k,l).non_obs_br = find(Pb(:,max_indx));

            cost = cost + max_non_obs_branches;
        end
    end
end
return;
```