FAULT IDENTIFICATION AND LOCATION IN POWER TRANSMISSION LINES USING MULTI-RESOLUTION ANALYSIS (MRA) AND PATTERN RECOGNITION

BY

ANEKE JUDE IFUNANYA
2015237001F

DEPARTMENT OF ELECTRICAL ENGINEERING

FACULTY OF ENGINEERING
NNAMDI AZIKIWE UNIVERSITY, AWKA
ANAMBRA STATE, NIGERIA.

SEPTEMBER, 2019

FAULT IDENTIFICATION AND LOCATION IN POWER TRANSMISSION LINES USING MULTI-RESOLUTION ANALYSIS (MRA) AND PATTERN RECOGNITION

BY

ANEKE JUDE IFUNANYA
2015237001F

A

DISSERTATION SUBMITTED TO THE SCHOOL OF POST GRADUATE STUDIES IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF DOCTOR OF PHYLOSOPHY (Ph.D.) IN ELECTRICAL ENGINEERING

NNAMDI AZIKIWE UNIVERSITY, AWKA
ANAMBRA STATE

CERTIFICATION

I hereby certify that, Aneke Jude Ifunanya withRegistration No: 2015237001F is the original author of this dissertation **(Fault Identification and Location in Power Transmission Lines Using Multi-Resolution Analysis (MRA) and Pattern Recognition)** and that this work has not been presented anywhere in part or in full for the award of diploma, degree or certificate. All sources of information are acknowledged.

_____

ANEKE JUDE IFUNANYA

APPROVAL PAGE

This is to certify that this dissertation has been approved and acceptedhaving met the requirements in partial fulfilment for the award of Doctor of Philosophy (Ph.D.) in the Department of Electrical Engineering, Nnamdi Azikiwe University, Awka.

APPROVED BY

Engr. Dr. O. A. Ezechukwu, FNSE, FIEEE Supervisor	_____ Signature	_____ Date
Engr. Prof. S. A. Ike External Examiner	_____ Signature	_____ Date
Engr. Dr. J. C. Onuegbu H.O.D., Electrical Engineering	_____ Signature	_____ Date
Engr. Prof. H. C. Godwin Dean, Faculty of Engineering	_____ Signature	_____ Date
Engr. Prof. P. K. Igbokwe Dean, School of Postgraduate Studies	_____ Signature	_____ Date

DEDICATION

This dissertation is dedicated to all my family members.

ACKNOWLEDGEMENT

All praises, thanks and sincere gratitude and appreciation are to God Almighty, the Lord of the world, for helping me to accomplish this work.

First, I would like to express my profound gratitude to my supervisor and the PG coordinator, Engr. Dr. O. A. Ezechukwu. His invaluabletechnical advice, suggestions, discussions, and kind support were the mainsources for the successful completion of the dissertation. He gave me the opportunity to work on a key issue and a highly interesting area in power system.

Secondly, I must not fail to recognize the assistance of the amiable Head of the Department, Engr. Dr. J. C. Onuegbu, who has been piloting the affairs of the department to a greater height. His professional advice and contributions to this work are most gratefully appreciated.

I especially want to thank Engr. Prof. F. O. Enemuoh, for his kind support and contributions to the work throughout my study period. And also to Engr. M. N. Eleanya, who actually initiated this research. I really appreciate his encouragement and motivation even before and during the research period. I am also exceedingly grateful to Engr. Dr. V. N. Agu, for his wholesome initiativesand suggestions towards the actualization of this work.

My undiluted thanks go to Prof. S. Sengupta, Dept. of Electronics & Electrical Communication Engineering, I. I. T., Kharagpur, India for his numerous online lectures on Artificial Intelligence (especially Artificial Neural Networks, ANN).

A bouquet of roses to all my lecturers and friends namely Engr. Dr. A. E. Anazia, Engr. T. F. Uketui, Engr. Prof. H. C. Inyiama,Engr. Prof. V. E. Idigo, Engr. Dr. C. O. Ohaneme, Engr. Dr. Ken Apkado, Engr. S. N. Onyedikachi, Engr. K. C. Obute, Engr. O. S. Ezennaya, Engr. C. C. Okoro etc and to all the staff of the Department for their invaluable assistance and providing pleasant atmosphere and inspiring research environment.

Finally, I wish I could put all the stars in the Universe in a vault to express with each one of them my love for my wonderful wife (Mrs V. C. Aneke) and also my parents (Mr. & Mrs. R. C. Aneke). Thank you for the best gift of my life and for making my dream come true. Nothing would have been possible without your support and love.

ABSTRACT

Fault Identification, classification and location on Ikeja West – Benin 330kV electric power transmission lines have been achieved using wavelet transform and artificial neural networks. This work proposed an improved solution based on wavelet transform and neural network back-propagation algorithm. The simulated three-phase fault current and voltage waveforms in the power transmission-line were first pre-processed and then decomposed using wavelet multi-resolution analysis to obtain the high frequency details and low frequency approximations. The data parameters of the patterns formed with the high frequency signal components were arranged as inputs of the neural network, whose task was to indicate the occurrence of a fault on the lines. The data parameters of the patterns formed using low frequency approximations were arranged as inputs of the second neural network, whose task was to indicate the exact fault type. The new method used both low and high frequency information of the fault signal to achieve an exact location of the fault**.**The neural networks were trained to recognize patterns and classify data. Feed forward networks have been employed along with back propagation algorithm for each of the three phases in the fault location process. Neural network (NN) designs with varying number of hidden layers and neurons per hidden layer have been provided to validate the choice of the neural networks in each step and simulation environment for real time fault diagnosis and detection was also presented. An analysis of the learning and generalization characteristics of elements in power system was presented using Neural Network toolbox in MATLAB/SIMULINK environment. Cross-Entropies of 8.1803e-3, 7.3899e-3 and 4.2980e-3 were achieved by the successfully trained NNs for the fault identification, classification and single line-to-ground fault location purposes respectively. Also the simulation results obtained from training the NNs for the fault location purposes indicated an average percentage error in the expected fault points of 0.6500%, 0.5811%, 0.7305% and 0.6447% for single line-to-ground, double line-to-ground, double line and three phase fault locations respectively with fault resistance of 15Ω and also an average percentage error of the expected results of 0.7284%, 0.7611%, 0.6837% and 0.6721% for single line-to-ground, double line-to-ground, double line and three phase fault locations respectively with fault resistance of 70Ω. These results demonstrated that wavelet multi-resolution analysis and neural network pattern recognition approachis efficient in identifying and locating faults on transmission lines as satisfactory performance was achieved

especially when compared to the conventional methods such as impedance and travelling wave methods.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF NOMENCLATURE

ACRONYMS:

AD	: Deterministic Annealing
ANN	: Artificial Neural Network
CFOOGW	: Composite Fiber Optic Overhead Ground Wire
DLG	: Double Line-to-Ground fault
DWT	: Discrete wavelet Transform
EMTDC	: Electromagnetic Transients including DC
FFT	: Fast Fourier Transform
FIA	: Fault Inception Angle
FL	: Fuzzy logic
GD	: Ground Detector
GI	: Ground Index
GS	: Generating Station
LL	: Line-to-Line
LLL	: Three Phase Fault
MLP	: Multilayer Perceptron
MSE	: Mean Square Error
NN	: Neural Network
PNN	: Probabilistic Neural Network
PSCAD	: Power System Computer Aided Design
RBF	: Radial basis functions
SLG	: Single Line-to-ground
SOFM	: Self Organizing Feature Map
STFT	: Short Time Fourier Transform
WER	: Wavelet Energy Ratio

CHAPTERONE

INTRODUCTION

1.1 Background of the Study

Several decades ago, there has been a rapid growth in the power grid all over the world which eventually led to the installation of a huge number of new transmission and distribution lines. Moreover, the introduction of new marketing concepts such as deregulation has increased the need for reliable and uninterrupted supply of electric power to the end users who are very sensitive to power outages (Das & Novosel, 2000).

Occurrence of a fault in a power system is one of the most important factors that hinder the continuous supply of electricity and power(IEEE, 2005). Any abnormal flow of current in a power system's components is called a fault in the power system. These faults cannot be completely avoided since a portion of these faults also occur due to natural reasons which are beyond the control of mankind. Hence, it is very important to have a well-coordinated protection system that detects any kind of abnormal flow of current in the power system, identifies the type of fault and then accurately locates the position of the fault in the power system. The faults are usually taken care of by devices that detect the occurrence of a fault and eventually isolate the faulted section from the rest of the power system.

As a result, some of the important challenges for the incessant supply of power are detection, classification and location of faults (Saha,Das, Verho& Novosel, 2006). Most of the research done in the field of protective relaying of power systems concentrates on transmission line fault protection due to the fact that transmission lines are relatively very long and can run

through various geographical terrain and hence it can take from a few minutes to several hours to physically check the line for faults (Eriksson, Saha& Rockefeller, 2015).

Hence, many utilities are implementing fault locating devices in their power quality monitoring systems that are equipped with Global Information Systems for easy location of these faults. Fault location techniques can be broadly classified into the following categories (Saha, Izykowski, & Rosolowski, 2010):

- Impedance measurement-based methods
- Travelling-wave phenomenon-based methods
- High-frequency components of currents and voltages generated by faults based methods
- Artificial Intelligence based method.

An overhead transmission line is one of the main components in every electric power system. The transmission line is exposed to the environment and the possibility of experiencing faults on the transmission line is generally higher than that on other main components. Line faults are the most common faults, they may be triggered by lightning strokes, trees may fall across lines, fog and salt spray on dirty insulators may cause the insulator strings to flash over, and ice and snow loadings may cause insulator strings to fail mechanically (Anamika, Thoke and Patel, 2008). Fault classification, faulted phase selection and location play a critical role in the protection for a transmission line. Accurate and fast fault detection, classification and location under a variety of fault conditions are important requirements from the point of service restoration and reliability. The process of fault classification, faulted phase selection and location involves:

1. Identifying the type of fault, e.g., single-phase to ground fault, phase-to-phase fault, etc. Therefore, the relay can select different algorithm elements to deal with different fault situations.
2. Identifying the faulted-phase to satisfy single-pole tripping and auto-reclosing requirements for operation.
3. Correct location of the fault distance, the maintenance crew can find and fix the problem to restore the service as quickly as possible. Rapid restoration of the service reduces outage time and loss of revenue. The speed and accuracy of protective relay can be improved by accurate and fast detection and classification (Haque and Kashtiban, 2005).

1.2 Problem Statement

Some of the important causes of the incessant and epileptic supply of electric power are inaccurate and slow pace of fault detection, classification and location of faults. Power system protection is the art of applying and setting up relays or fuses or both, to provide maximum sensitivity to faults and abnormal conditions and also to avoid false alarms during normal operating conditions (Jain, Thoke, Patel, 2008). So it is desirable that a correct decision be made by the protective device as to whether the trouble is an abnormal condition or just a transient which the system can absorb and return to normal working condition. The protective relays are more of a preventive device which operates after a fault has occurred and it helps in minimizing the duration of trouble and limiting the damage, outage time and related problems. For the system to operate properly, it is necessary to isolate the trouble area quickly with a minimum number of system disturbances. Both failure to operate and incorrect operation can result in major system upsets involving increased equipment damage, increased personnel hazards and possibly long interruption of service. Wavelet Transform and Artificial

Neural Network modelhave been proposed in this dissertation for fast and accurate detection, classification and location of the fault position with utmost accuracy.

1.3 Aim and Objectives of the Dissertation

The aim of this dissertation is to model, develop and implementa fault identification and location in power transmission lines using multi-resolution analysis (MRA) and pattern recognition.The objectives of this dissertation are:

a) To model the three phase 280km long Ikeja-West to Benin 330kVpower transmission line.

b) To perform simulation at various intervals (km) on the model for different fault conditions.

c) To perform feature extraction by decomposing the faulty current and voltage waveforms of different fault scenarios into coarse approximations and detail coefficients using multi-resolution tool of wavelet transform.

d) To perform summation analysis of extracted detail and approximation coefficients.

e) To model a neural network based fault detector that can with utmost precision, detect faults in power transmission lines.

f) To model a neural network based fault classifier and locator that can classify and locate faults in power transmission lines using Feed-forward networks along with back propagation algorithm.

1.4 Motivation for the Research

Electrical faults are the most damaging among the disturbances that could possibly happen in the power system. Although some faults are transient in nature that occur in just a few cycles, they subject the generator to mechanical and temperature stresses beyond its operating limits (Khorashadi-Zadeh, 2004). The more frequent the occurrences of these events in the power network, the faster will be the rate of deterioration or wear of the network.

The prime motive behind this dissertation is the significant impact a very accurate fault locator could make if employed in a power transmission and distribution system, in terms of the amount of money and time that can be saved. The main goal of fault location is to locate a fault in the power system with the highest practically achievable accuracy. When the physical dimensions and the size of the transmission lines are considered, the accuracy with which the designed fault locator locates faults in the power system becomes very important.

One of the important aspects that this dissertation concentrates on is the analysis of the transmission line's phase voltages and currents during various fault conditions and how they can be effectively utilized in the design of an efficient fault locator. This dissertation drew its initial motivation from (Reddy, and Mohanta, 2008) which demonstrates a method that could be used for location of faults in transmission lines using neural fuzzy logic. However, when extensively studied, it can be noted that a fault locator with satisfactorily high accuracy can be more easily achieved with the help of wavelet transform and artificial neural networks by the use of a large amount of data set for training and the learning process. This eliminates the need for high proficiency in power systems which is a necessity when working with expert fuzzy systems. Hence, this dissertation focuses on the design of a fault locator that can be even used by people who aren't experts in the field of power systems.

1.5 Scope of Work

The scope of this dissertation is to successfully model, develop, test and implement a complete strategy for the fault identification and location in power transmission lines using a wavelet transform multi-resolution analysis and pattern recognition neural network based fault locator that can, with very high accuracy detect, classify and locate faults in power transmission lines. A Feed-forward network along with Levenberg-Marquardt and Scaled Conjugate Gradient back-propagation algorithm will be employed for each of the three phases in the Fault location process.

1.6 Justification of the Dissertation

Transmission lines constitute the major part of electric power system. Transmission and distribution lines are vital links between the generating unit and consumers to achieve the continuity of electric supply. To economically transfer large blocks of power between systems and from remote generating sites, High Voltage (HV) and Extra high voltage (EHV) overhead transmission systems are being used. Transmission lines also form a link in interconnected system operation for bi-directional flow of power. Transmission lines run over hundreds of kilometres to supply electrical power to the consumers (Aggarwal, et al, 1994) and (Bo, et al, 2000). They are exposed to atmosphere and environmental hazards, hence chances of occurrence of fault in transmission line is very high which when they eventually occur has to be immediately taken care of in order to minimize damage caused by it.

Having an effective automated means of identifying and determining the location of the fault even right from the control-room will significantly improve continuity of power supply.It will also facilitate quicker repair, improve system availability and performance, reduce operating

cost and save time and effort of maintenance crew searching in, sometimes in harsh environmental conditions (Jain, Kale and Thoke, 2006). It has always been an interest for engineers to detect and locate the faults in the power system as early as possible. Fast clearing and restoration is very essential as it not only provides reliability but sometimes also stops propagation of disturbances which may lead to blackouts.

CHAPTERTWO

LITERATURE REVIEW

2.1 The Nature and Causes of Power System Faults

The nature of a fault is simply defined as any abnormal condition, which causes a reduction in the basic insulation strength between phase conductors, or between phase conductors and earth or any earthed screens surrounding the conductors. In practice, a reduction is not regarded as a fault until is it is detectable, that is until it results either in an excess current or in a reduction of the impedance between conductors, or between conductors and earth, to a value below that of the lowest load impedance normal to the circuit (Xia, Zhuang and Huang, 2010). Thus, a higher degree of pollution on an insulator string, although it reduces the insulation strength of the affected phase, does not become a fault until it causes a flashover across the string, which in turn produces excess current or other detectable abnormality, for example abnormal current in an arc-suppression coil (Tang, et al, 2000). Following are some of the main causes:

2.1.1 Lightning

More than half of the electrical faults occurring on overhead power transmission lines are caused by lightning (see Figure 2.1) (Furukawa, Usada, Isozaki, and Irie, 1989). The main conventional approaches for reduction of the lightning flashover faults on power lines are lowering of the footing resistance and employing of multiple shielding wires, and differential insulation.

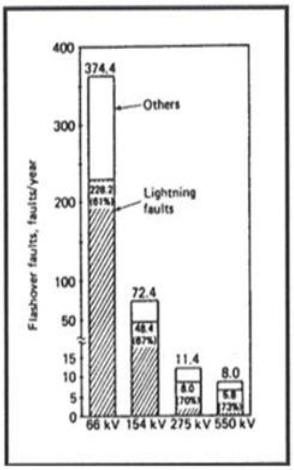


Figure 2.1: Flashover Faults on Transmission Lines

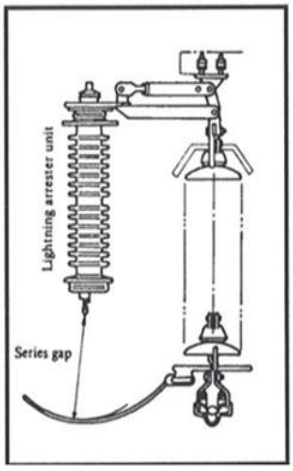


Figure 2.2: Configuration of Lightning Arrester

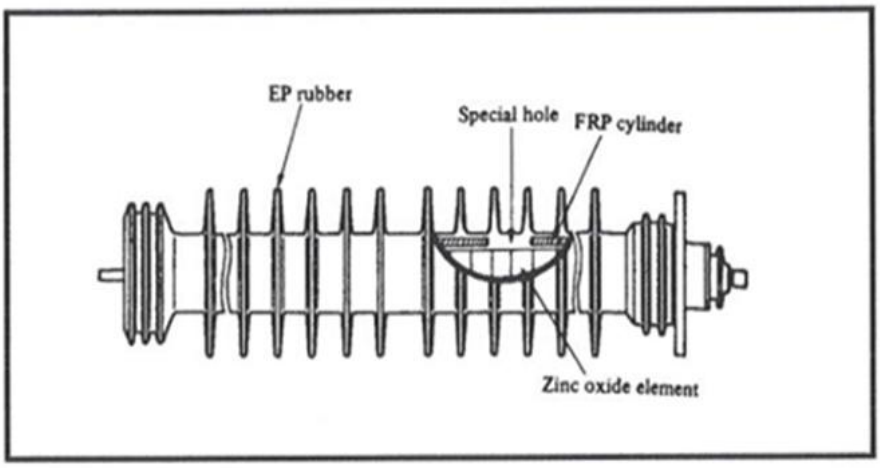


Figure 2.3: Lightning Arrester unit (Zinc Oxide elements in the FRP cylinder aremoulded with Ep rubber) (Solanki and Song, 2003)

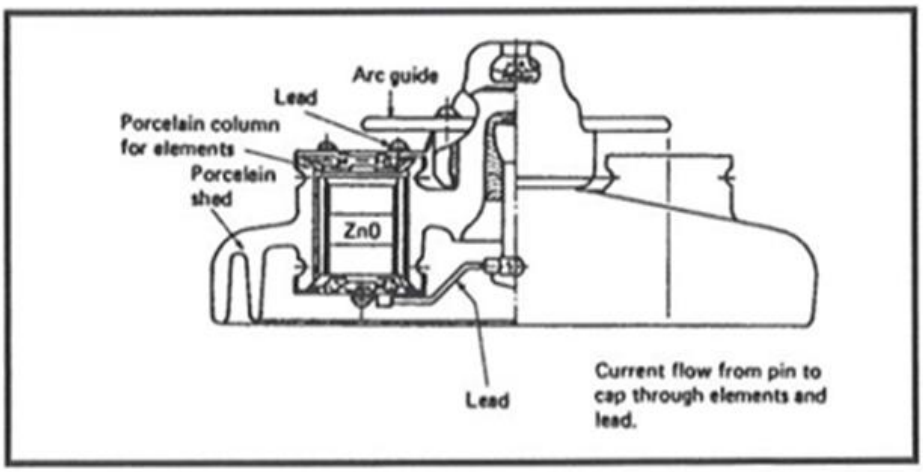


Figure 2.4: Lightning Arrester Unit ZnO (Solanki and Song, 2003)

However, these methods have not been sufficient to prevent flashover faults. In the meantime, application of arresters to lines has been a better solution in recent years. This alternate approach is to install an arrester to prevent the flashover of insulator assemblies. It is important that the arrester should be strong enough in order to withstand excessive lightning strikes. The suspension-type line arrester was developed by incorporating ZnO elements into the shed of a conventional suspension insulator (Figures 2.2, 2.3 and 2.4). It has an arrester function along with the normal electrical and mechanical functions of a line insulator. It is a gapless type that has the advantage of reliable surge absorption with no delay in discharge. The new arrester holds promise not only for the prevention of lightning faults, but also as means of overall transmission systems. (Gonen and Turan, 1987).

2.1.2 Pollution

Pollution is commonly caused by deposited soot or cement dust in industrial areas, and by salt deposited by wind-borne sea-spray in coastal areas. A high degree of pollution on an insulator string, although it reduces the insulation strength of the affected phase, does not become a fault until it causes a flashover across the string, which in turn reduces excess current or other detectable abnormality, for example abnormal current in an arc-suppression coil.

2.1.3 Fire

The occurrence of fire under transmission lines is responsible for a great number of line outages in many countries. Faults are mainly due to conductor to ground short circuit or phase-to-phase short circuit depending on line configuration and voltage level. To reduce these outages to minimum, the clearance of existing line right-of-way must be increased in forests. Clearing of vegetation on the line right of way in such areas is also a consideration.

Another problem arising from burning is the contamination of the insulators due to the accumulation of particles (soot, dust) on its surfaces. In this case, the line insulation requirements should be determined in such a way that the outages under fire could be reduced to a minimum. (Fonseca, et al, 1990).

Other causes of faults on overhead lines are trees, birds, aircraft, fog, ice, snow loading, punctured or broken insulators, material failure, damages, open-circuit conductors and abnormal loading.

2.2 Categorization of Transmission Line Faults

Power system faults may be categorized as shunt faults and series faults. The most occurring type of shunt faults is Single Line-to-ground faults (SLG), which is one of the four types of shunt faults, which occur along the power lines. This type of fault occurs when one conductor falls to ground or contacts the neutral wire. It could also be as a result of falling trees in a rainy storm. This type can be represented as in Figure 2.5.

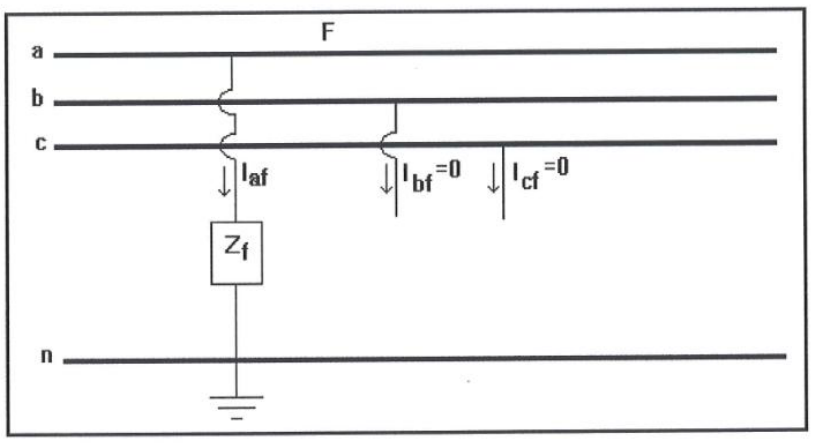


Figure 2.5: Single Line-to-Ground Fault

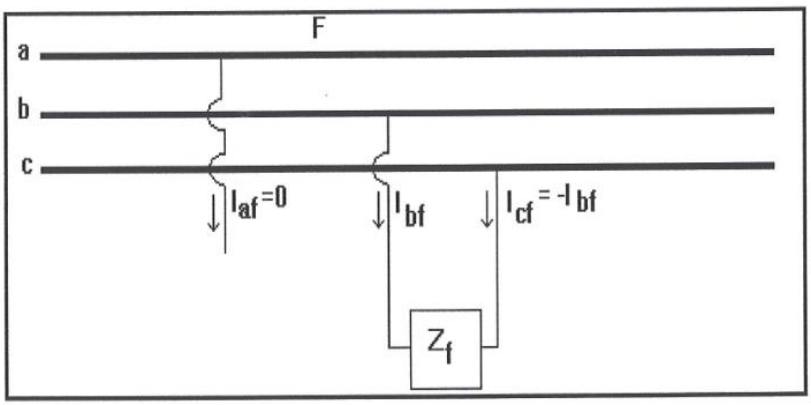


Figure 2.6: Single Line-to-Line Fault

The second most occurring type of shunt faults is the Line-to-Line fault (LL). It is the result of two conductors being short-circuited. As in the case of a large bird standing on one transmission line and touching the other, or if a tree branch falls on top of the two of the power lines. This type could be represented as in Figure 2.6.

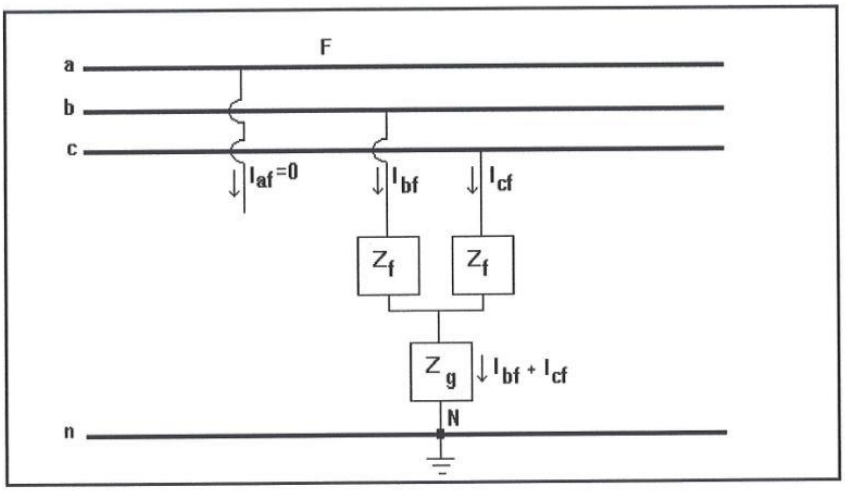


Figure 2.7: Double Line-to-Ground Fault

The third type of fault is the Double Line-to-Ground fault (DLG), Figure 2.7. This can be a result of a tree falling on two of the power lines, (Fonseca, et al, 1990) or other causes. The fourth and least occurring type of fault is the balanced three phases (Figure 2.8), which can occur by a contact bridging the three power lines in many different forms.

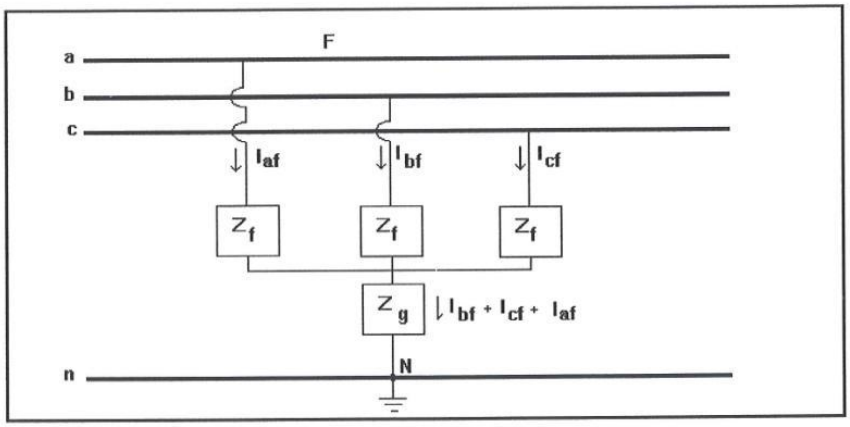


Figure 2.8: Three Phase Fault

The fourth type of fault which is the balanced three-phase fault is defined as the simultaneous short circuit across the three phases. It occurs infrequently, but it is most severe type of fault encountered. Because the network is balanced, it is solved on a per-phase basis. The other two phases carry identical currents except for phase shift. The reactance of the synchronous generator under short circuit conditions is a time-varying quantity, and for network analysis three reactances were defined. The sub-transient reactance of the first few cycles has short circuit current, transient reactance in the next 30 cycles and the synchronous reactance thereafter. Since the duration of the short circuit current depends on the time of operation of the protective system, it is not always easy to decide which reactance to use. Generally, the sub-transient reactance is used for determining the interrupting capacity of the circuit breakers.

Series faults can occur along the power lines as the "result of an unbalanced series impedance condition of the lines in the case of one or two broken lines for example. In practice, a series fault is encountered, for example, when lines (or circuits) that are controlled by circuit breakers (or fuses) or any device does not open all the three phases; one or two phases of the line (or the circuit) may be open while the other phases or phase is closed." (Gonen, Turan, 1987)

2.3 Power Protection Systems

One of the most important components of a power protection system is the relay which is a device that trips the circuit breakers when the input voltage and current signals correspond to the fault conditions designed for the relay operation. Relays in general can be classified into the following categories (Wright, Christopoulos, 1993):

- Type of constructions
- Type of characteristics
- Input quantity or stimulus
- Function they perform

Among the various relays that are used for the protection of power lines distance relays are the most relevant to fault locators. Usually a pair of these distance relays is used for the protection of a two-terminal transmission line (Ziegler, 2006).

2.4 Transmission Line Fault Location Techniques

The transmission line fault location process, as mentioned before, has been researched for a while and several innovative and efficient techniques have been proposed andanalyzed by several authors (Djuric, Radojevic, Terzija, 1998). These techniques can be broadly classified as impedance-based methods, travelling wave based methods and Artificial Intelligence based methods. Each of these methods is discussed briefly in the following sub-sections.

2.4.1 Impedance Measurement Based Methods

In the case of Impedance based methods, the operation of the distance relay greatly relies on the fault resistance and is not successful in cases with very high fault resistance (Eriksson, Saha, Rockefeller, 2015). Impedance based methods can be classified into single-ended methods and two-ended methods depending upon the number of terminals at which the voltage and current data are collected.

The basic logic behind a single-ended impedance-based fault locator is to calculate the location of the fault from the apparent impedance seen looking into the line from one end. The various impedance-based methods available in literature are discussed in the upcoming subsections.

2.4.1.1 Simple Reactance Method

The measured voltage and current values at the terminal are used to calculate the impedance of the line to the fault position as shown in equation 2.1. Once the line impedance per unit length has been determined, the fault distance can be calculated accordingly as illustrated by equations 2.2 and 2.3(Karl, David, 2009).

$$V_A = x.Z_L.I_A + V_f \qquad (2.1)$$

Where;V_A is the voltage at terminal A,

$x$is the distance to the fault from the terminal A,

$I_A$is the current flowing out of the terminal A, V_f is the fault voltage and

$Z_L$is the line impedance.

$$V_A = x.Z_L.I_A + R_f I_f \tag{2.2}$$

Where I_f is the fault current and R_f is the fault resistance as shown in Fig 2.9.

$$x = \frac{(V_A/I_A)}{Z_L} - \frac{R_f}{Z_L\left(\frac{I_A}{I_f}\right)} \tag{2.3}$$

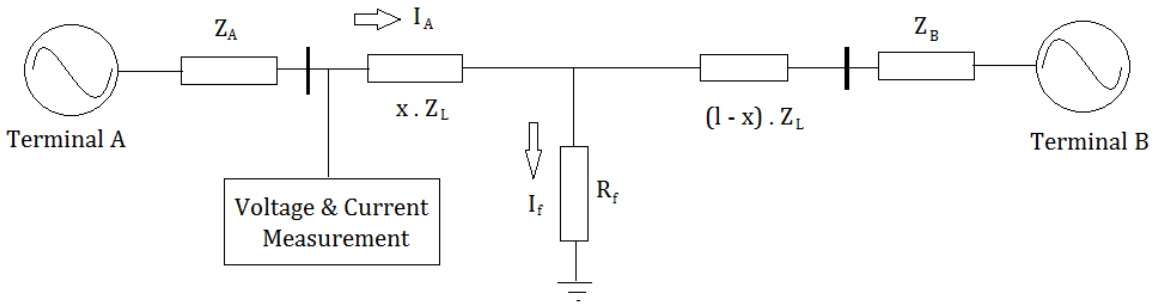


Figure 2.9: Faulty Transmission Line illustrating simple-reactance method.

2.4.1.2 Takagi Method

The Takagi method (Edmund, Schweitzer, 1988) is a very simple yet innovative single-ended impedance based Fault location technique and is illustrated by Fig 2.10. It requires both the

pre-fault and fault data and enhances the simple reactance method by minimizing the effect of fault resistance and reducing the effect of load flow.

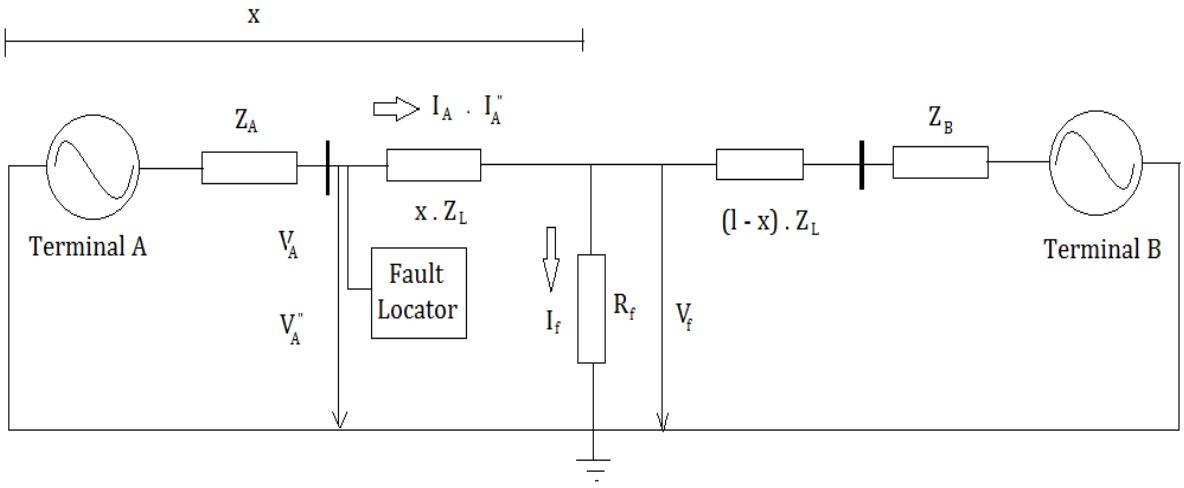


Figure 2.10: A single-phase circuit illustrating Takagi method.

The Fault Resistance is given by

$$R_f = \frac{V_A - Z_C I_A \tanh \gamma x}{(\frac{V_A''}{Z_C} \tanh \gamma x - I_A'')\psi \varepsilon^{j\theta}} \quad (2.4)$$

Where;V_A is voltage measured at terminal A, I_A is the current flowing out of terminal A, γ is the propagation constant, Z_C is the characteristic impedance, Z_L is the line impedance, I_A'' is the superposition current which is the difference between the fault current and the pre-fault current.

And $x = \frac{Im(V_A . I_A''^*)}{Im(Z_L I_A . I_A''^*)}$ (2.5)

is the distance to the fault from terminal A. Where;

$$Z_L = \gamma Z_C \quad (2.6)$$

2.4.1.3 Modified Takagi Method

The modified Takagi method also called the Zero Sequence current method does not require pre-fault data because it uses zero-sequence current instead of the superposition current for ground faults (Aurangzeb, Crossley, Gale, 2001). The location of the fault in this method is given by x inequation 2.7.

$$x = \frac{Im(V_A.I_R^*.e^{-j\beta})}{Im(Z_{1L}.I_A.I_R^*.e^{-j\beta})} \quad (2.7)$$

Where I_R^* is the conjugate of zero-sequence current and β is the zero-sequence current angle. The position of the fault 'x' is given by equation 2.7; V_A is voltage measured at terminal A, I_A is the current flowing out of terminal A and Z_{1L} is the positive sequence line impedance.

2.4.2 Travelling Wave Based Methods

Travelling wave based methods have been widely used (Bo, Weller, Redfern, 1999) for the purpose of fault location and are usually based on the correlation between the forward and backward waves travelling along the transmission line as shown in Figure 2.7. The basic idea is to successively identify the fault initiated by high-frequency travelling waves at the fault locator.

The time taken by the high frequency components for propagation is used for the location of fault (Silva, Oleskovicz, Coury, 2004). In Figure 2.11, a single phase lossless transmission line of length 'l' is considered with a travelling wave velocity of υ, capacitance and

inductance per unit length 'C' and 'L' and a characteristic impedance of Z_C. Assuming the occurrence of a fault at a distance of 'x' from the terminal A, the voltage and current values are given by (2.8) and (2.9).

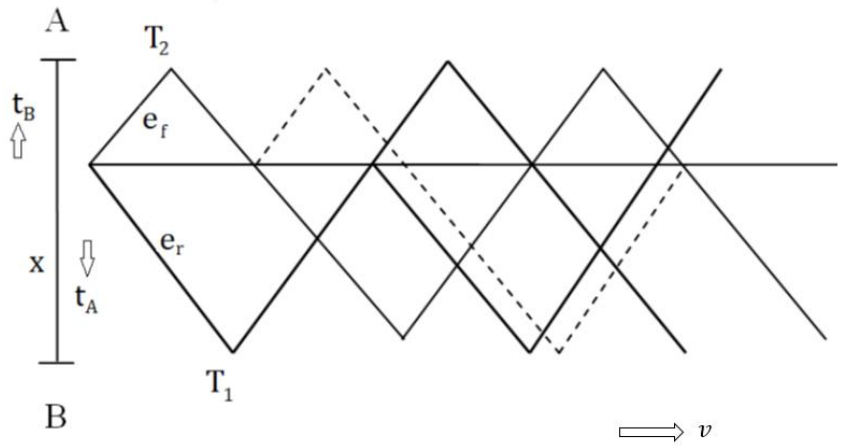


Figure 2.11: Illustration of Travelling wave based Fault Location.

$$\frac{\partial e}{\partial x} = -L' \frac{\partial i}{\partial t} \tag{2.8}$$

$$\frac{\partial i}{\partial x} = -C' \frac{\partial e}{\partial t} \tag{2.9}$$

Whose solutions are given by (2.10) and (2.11).

$$e(x,t) = e_f(x - vt) + e_r(x + vt) \tag{2.10}$$

$$i(x,t) = \frac{1}{Z_C} e_f(x - vt) - \frac{1}{Z_C} e_r(x + vt) \tag{2.11}$$

The times τ_A and $\tau_B$taken for the waves to travel from the fault to the discontinuity are to be determined using GPS technology. Once this is done, the fault location (x) can be readily determined by the followingequation 2.12.

$$x = \frac{l - c(\tau_A - \tau_B)}{2} \quad (2.12)$$

Where: c is the wave propagation speed of 299.79 m/sec.

2.4.3 Artificial Intelligence (AI) Methods

AI is a subfield of computer science that investigates how the action of human beings can be mimicked by machines (Warwick, Ekwue and Aggarwal, 2015). Both the numeric, non-numeric and symbolic computations are included in the area of AI. The mimicking of intelligence includes not only the ability to make rational decisions, but also to deal with missing data, adapt to existing situations and improve itself in the long time horizon based on the accumulated experience.

From quite a few years, intelligent based methods are being used in the process of fault detection and location. Three major families of artificial intelligence based techniques that have been widely used in modern power system are (Saha and Kasztenny, 2014):

- Expert System Techniques (XPS),
- Fuzzy Logic Systems (FLS),
- Artificial Neural Networks (ANN).

2.4.3.1 Expert Systems

The first systems included a few heuristic rules based on the expert's experience. In such systems, the knowledge takes the form of the so called production rules written using the *If ... then ...* syntax (knowledge base). The system includes also the facts which generally describe the domain and the state of the problem to be solved (data base). A generic inference engine uses the facts and the rules to deduce new facts which allow the firing of other rules. The knowledge base is a collection of domain-specific knowledge and the inference system is the

logic component for processing the knowledge base to solve the problem. This process continues until the base of facts is saturated and a conclusion has been reached as shown in Figure 2.12. To guide the reasoning and to be more efficient, these systems may incorporate some strategies known as meta-knowledge. Rule based systems represent still the majority of the existing expert systems.

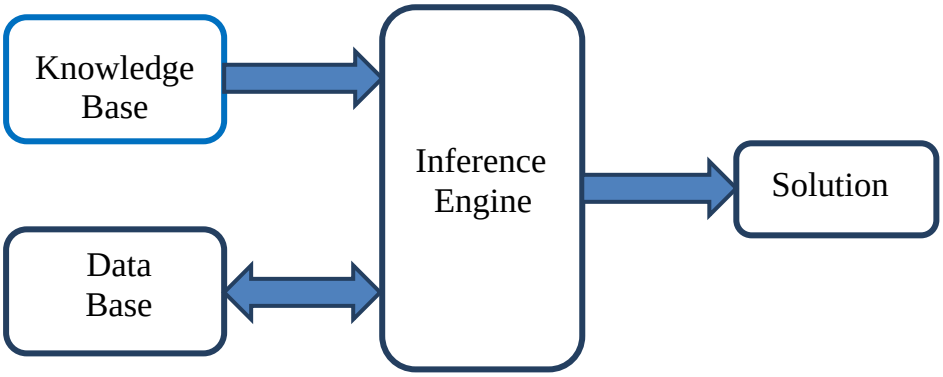


Figure 2.12: Simplified block diagram of an XPS

There are few applications of XPS to power system protection reported, but all of them solve the off-line tasks such as settings coordination, post-fault analysis and fault diagnosis (Saha and Kasztenny, 2014). As yet there is no application reported of the XPS technique employed as a decision making tool in an on-line operating protective relay. The basic reason for this is that there is no extensive rule base that describes the reasoning process applicable to protective relaying. Instead, only a few rules or criteria are collected (Wiszniewski and Kasztenny, 2013).

2.4.3.2 Fuzzy Logic Systems

Fuzzy logic (FL) can be defined as a problem-solving control system methodology that lends itself to implementation in systems ranging from simple, small, embedded micro-controllers to large, networked, multi-channel personal computer (PC) or workstation-based data acquisition and control systems. Fuzzy based classification technique employs a simple, rule-based *IF X AND Y THEN Z* approach to a solving control problem rather than attempting to model a system mathematically.

With reference to Figure 2.13, the fuzzy approach to protective relaying assumes that (Kasztenny, Rosolowski, Saha, Hillstrom, 2015):

- The criteria signals are fuzzified in order to account for dynamic errors of the measuring algorithms. Thus, instead of real numbers, the signals are represented by fuzzy numbers. Since the fuzzification process provides a special kind of flexible filtering, faster measuring algorithms that speed up the relays may be used.
- The thresholds for the criteria signals are also represented by fuzzy numbers to account for the lack of precision in dividing the space of the criteria signals between the tripping and blocking regions.
- The fuzzy signals are compared with the fuzzy settings. The comparison result is a fuzzy logic variable between the Boolean absolute levels of truth and false.
- Several relaying criteria are used in parallel. The criteria are aggregated by means of formal multi-criteria decision-making algorithms that allow the criteria to be weighted according to their reasoning ability.
- The tripping decision depends on multi-criteria evaluation of the status of a protected element. Additional decision factors may include the amount of available information, or the expected costs of the relay mal-operation.

Figure 2.13: Simplified block diagram of the fuzzy logic approach

The Fuzzy BasedFault Classification is based on angular differences among the sequence components of the fundamental during fault current as well as on their relative magnitudes. The phasor diagram of a phase "a" to ground fault is shown in the Figure 2.14.

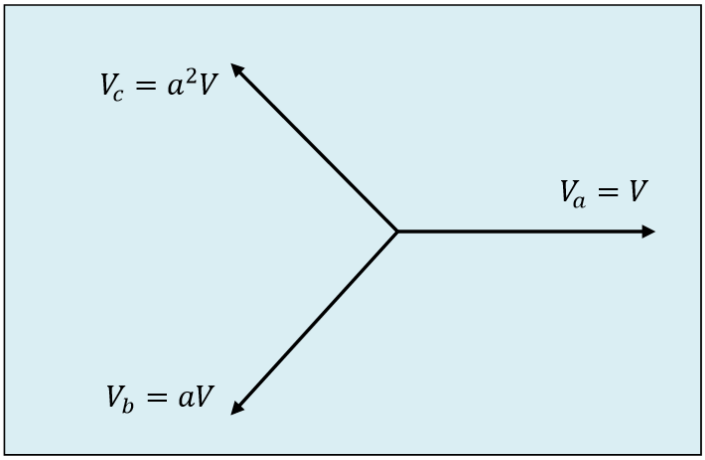


Figure 2.14: Phasor diagram for a-g fault

The zero, positive and negative sequence components of the post fault currents relative to phase "a" are denoted as I_{aof}, I_{a1f} and I_{a2f} respectively. The angles between the positive and negative sequence components of phase a, b and c are given as

$$\begin{aligned} arg\_A &= \left|Arg(I_{a1f}) - Arg(I_{a2f})\right| = 0^0 \\ arg\_B &= \left|Arg(I_{b1f}) - Arg(I_{b2f})\right| = 120^0 \\ arg\_C &= \left|Arg(I_{c1f}) - Arg(I_{c2f})\right| = 120^0 \end{aligned} \quad (2.13)$$

The magnitudes of I_{aof}, I_{a1f} and I_{a2f} are related by

$$R_{of} = |I_{aof} / I_{a1f}| = 1 \text{ and } R_{2f} = |I_{a2f} / I_{a1f}| = 1 \quad (2.14)$$

Similarly, the magnitudes and angle between the positive and negative sequence components are obtained for other types of asymmetric faults.

For every type of fault, there exists a unique set of these five parameters. So it is possible to formulate simple logic base for determining the fault type from the values of the five inputs. The different inputs are represented by a corresponding fuzzy variable. Now a fuzzy rule will be developed using these five variables to detect the type of fault. For example:

If arg_A is "approximately 30^0" and arg_B is "approximately 150^0" and arg_C is "approximately 150^0" and R_{of} is "high" and R_{sf} is "high" then fault type is "a-g"

In this method, only 3 parameters are sufficient and it identifies 10 types of short-circuit faults accurately. But the main disadvantage with this method is that it is applicable to only asymmetric faults and it is not very effective if the aim is to classify not just by the type of fault.

2.4.3.3 Artificial Neural Networks (ANNs)

The ANNs are very different from expert systems since they do not need a knowledge base to work. Instead, they have to be trained with numerous actual cases. An ANN is a set of elementary neurons which are connected together in different architectures organized in layers of what is biologically inspired. An elementary neuron can be seen like a processor which

makes a simple non-linear operation of its inputs producing its single output. The ANN techniques are attractive because they do not require tedious knowledge acquisition, representation and writing stages and, therefore, can be successfully applied for tasks not fully described in advance.

The ANNs are not programmed or supported by knowledge base as are Expert systems. Instead, they learn a response based on a given inputs and required output by adjusting the node weights and biases accordingly. The speed of processing, allowing real time applications, is also advantage. Since ANNs can provide excellent pattern recognition, they are proposed by many researchers to perform different tasks in power system relaying for signal processing and decision making (Lukowicz, Rosolowski, 2013).

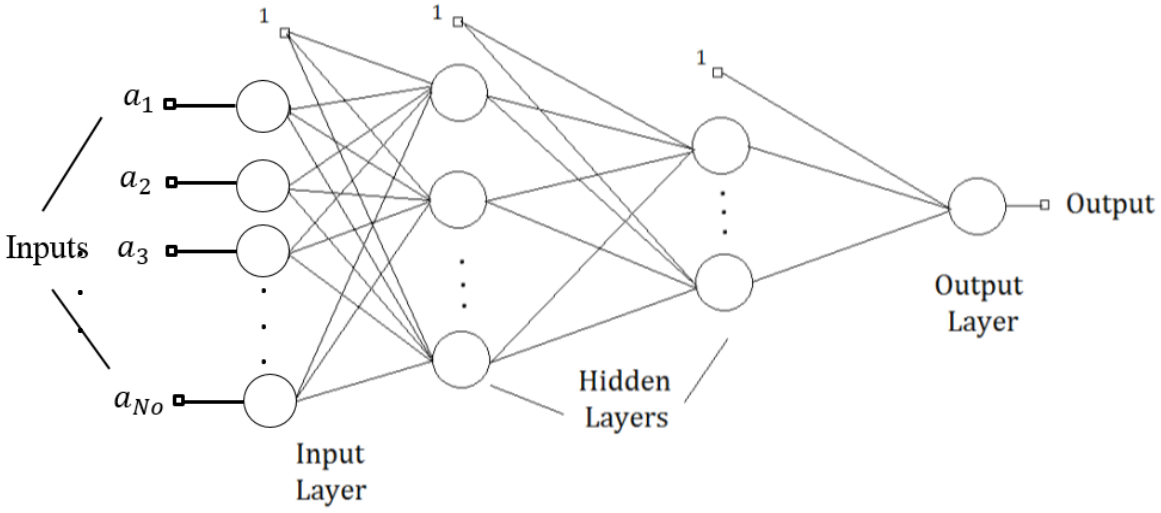


Figure 2.15: A basic three-layer architecture of a feed-forward ANN.

An Artificial Neural Network (ANN) can be described as a set of elementary neurons that are usually connected in biologically inspired architectures and organized in several layers (Cichoki, Unbehauen, 1993). The structure of a feed-forward ANN, also known as the perceptron is shown in Fig 2.15. There are N_i numbers of neurons in each ith layer and the

inputs to these neurons are connected to the previous layer neurons. The input layer is fed with the excitation signals. Simply put, an elementary neuron is like a processor that produces an output by performing a simple non-linear operation on its inputs (Haykin, 1994). A weight is attached to each and every neuron and training an ANN is the process of adjusting different weights tailored to the training set. An Artificial Neural Network learns to produce a response based on the inputs given by adjusting the node weights. Hence, we need a set of data referred to as the training data set, which is used to train the neural network.

In Fig. 2.15, a_1, a_2 … a_{N0} is the set of inputs to the ANN. Due to their outstanding pattern recognition abilities ANNs are used for several purposes in a wide variety of fields including signal processing, computers and decision making. Some important notes on artificial neural networks are (Kezunovic, 1997):

- Either signal features extracted using certain measuring algorithms or even unprocessed samples of the input signals are fed into the ANN.
- The most recent along with a few older samples of the signals are fed into the ANN.
- The output provided by the neural network corresponds to the concerned decision which might be the type of fault, existence of a fault or the location of a fault.
- The most important factor that affects the functionality of the ANN is the training pattern that is employed.
- Pre-processing and post-processing techniques may be employed as well to enhance the learning process and reduce the training time of the ANN.

One of the biggest drawbacks of applications that make use of artificial neural networks is that no well-defined guide exists to help choose the ideal number of hidden layers to be used and the number of neurons per each hidden layer. From a different perspective, it is advantageous considering the wonderful ability to generalize. A vital feature of ANN is its dedication to parallel computing. Hence it can produce a correct output corresponding to any input even if the concerned input was not fed into the ANN during the training process. Another challenge in the ANN based application development was to synthesize the algorithm for the adaptive learning process (Gail, Carpenter, 1997). The back error-propagation algorithm is the basic algorithm in which the neuron weights are adjusted in consecutive steps to minimize the error between the actual and the desired outputs. This process is known as supervised learning.

Neural networks have seen an explosion of interest over the last few years and are being successfully applied across an extraordinary range of problem domains, in areas as diverse as finance, medicine, engineering, geology, physics and biology. The excitement stems from the fact that these networks are attempts to model the capabilities of the human brain (Vasilic and Kezunovic, 2002). From a statistical perspective neural networks are interesting because of their potential use in prediction and classification problems.

Artificial neural networks (ANNs) are non-linear data driven self-adaptive approach as opposed to the traditional model based methods. They are powerful tools for modelling, especially when the underlying data relationship is unknown. ANNs can identify and learn correlated patterns between input data sets and corresponding target values (Aggarwal and Song, 1997). After training, ANNs can be used to predict the outcome of new independent input data. ANNs imitate the learning process of the human brain and can process problems involving non-linear and complex data even if the data are imprecise and noisy. Thus they are

ideally suited for the modelling of agricultural data which are known to be complex and often non-linear (Pao, 2012).

A very important feature of these networks is their adaptive nature, where "learning by example" replaces "programming" in solving problems. This feature makes such computational models very appealing in application domains where one has little or incomplete understanding of the problem to be solved but where training data is readily available (Kezunovic, Rikalo and Sobajic, 1995).

These networks are "neural" in the sense that they may have been inspired by neuroscience but not necessarily because they are faithful models of biological neural or cognitive phenomena (Kezunovic, Rikalo, 1996). In fact, majority of the network are more closely related to traditional mathematical and/or statistical models such as non-parametric patternclassifiers, clustering algorithms, nonlinear filters, and statistical regression models than they are to neurobiology models.

Neural networks (NNs) have been used for a wide variety of applications where statistical methods are traditionally employed. They have been used in classification problems, such as identifying underwater sonar currents, recognizing speech, and predicting the secondary structure of globular proteins (Sidhu, Singh, and Sachdev, 1995). In time-series applications, NNs have been used in predicting stock market performance. As statisticians or users of statistics, problems are normally solved through classical statistical methods, such as discriminant analysis, logistic regression, Bayes analysis, multiple regression, and ARIMA time-series models. It is, therefore, time to recognize neural networks as a powerful tool for data analysis (Kezunovic, Rikalo and Sobajic, 1996).

In conclusion, the XPS, FLS and ANN approaches have their own advantages and limitations. XPS and FLS methods require a knowledge base, that is, an expertise body of coded information of any particular system under consideration before they could be applied. This makes them ill-disposed to generalized application. ANN on the other hand does not require a knowledge base hence it is well suited to generalized and rapid deployment. This is the reason for the choice of ANN in this dissertation for fault identification and location in electric power transmission lines.

2.5 Power System Fault Identification Techniques

2.5.1 Fault Identification Using Short Fourier Transform

The Discrete Fourier Transform of the signal gives the frequency components that exist in the signal. But since power system fault waveforms are non-stationary by nature and since engineers are also interested in determining the place where the fault has occurred, Short Time Fourier Transform (STFT) was used for detecting the power system transmission line faults. In STFT, the signal is divided into small enough segments, where these segments (portions) of the signal can be assumed to be stationary (Robi, 2014). For this purpose, a window function "w" is chosen. The width of this window must be equal to the segment of the signal where its stationarity is valid.

This window function is first located to the very beginning of the signal. That is, the window function is located at $t = 0$. Let's suppose that the width of the window is "T" seconds. At this time instant ($t = 0$), the window function will overlap with the first $T/2$ seconds. The window function and the signal are then multiplied. By doing this, only the first $T/2$ seconds of the signal is being chosen, with the appropriate weighting of the window. Then this product

is assumed to be just another signal, whose Fourier transform (FT) is to be taken. In other words, FT of this product is taken, just as taking the FT of any signal.

The result of this transformation is the FT of the first $T/2$ seconds of the signal. If this portion of the signal is stationary, as it is assumed, then there will be no problem and the obtained result will be a true frequency representation of the first $T/2$ seconds of the signal. The next step would be shifting this window to a new location, multiplying with the signal, and taking the FT of the product. This procedure is followed; until the end of the signal is reached by shifting the window with $t^{'}$ seconds intervals.

Thus, Short Time Fourier Transform of the signal can be defined as

$$S\,T\,F\,T\,x\,(t,f) = \int [x(t)^{*} w^{*}(t - t^{'})]^{*} e^{-i2\pi f t}\, dt \qquad (2.15)$$

Where;*x(t)* is the signal, *w(t)* is the window function and *w** is the complex conjugate. Thus, this gives a time frequency representation of the signal. So, it gives the frequency response as well the occurrence of that frequency in time. It is computationally fast and detects faulted waveforms effectively. But this technique depends on the selection of a good window. This is called the Resolution problem, in other words choosing the appropriate size of the window plays a vital part in the fault detection process. Narrower windows have good time resolution but poor frequency resolution, but on the other hand broader windows have good frequency resolution but poor time resolution (Robi, 2014). But using the wavelet transform overcomes the resolution problem.

2.5.2 Fault Identification Using Composite Fiber-Optic

In electric power supply services, power transmission lines are very important and very indispensable. For that, power transmission lines are equipped with various protection

systems that are checked periodically because of the unexpected faults that may destroy the lines. For the purpose of protecting these lines, a system was invented to discover the fault location using Composite Fiber Optic Overhead Ground Wire (COOGW). This system deals mainly with most causes of fault situations such as lightning, dew, snow, fog, or gales. This fault location system was developed to find out where electrical fault occurred on overhead power transmission lines by detecting the current induced in the ground wire (Anderson, 2003) and (Cook, 1996). Any fault situation needs the fastest processing in fixing the fault. For that, the fault location system helps engineers to detect the point or the section where an electrical fault happened in very logic time.

Since the fault information is uncertain, the fault location deals with the fault information as a current distribution pattern throughout the power line, and applies Fuzzy Theory to realize the human-like manner of fault used by power engineers. Mainly, the fault location method measures the current induced in the COOGW at many points along the line, these points are various sensors mounted on the tower and transmits the information to the central monitoring station through the optical fiber within the COOGW (Razi, Hagh and Abrabian, 2007). So the fault information system is mainly given by sensing and data transmission. Electrical faults occurring on power transmission lines can be classified into two types: grounding and short circuit fault.

The transmission that gets to the central monitor station deals with current characteristic features. In order to locate the fault, engineers must use the features of currents that deal with the phase angle and the amplitude and relate these features to Fuzzy Theory (Gaudrat, Giusiano, and Huiart, 2004). The idea arose of using Fuzzy Theory as a fault theory algorithm similar to this kind of human thinking.

There were some sets for Fuzzy theory:

S_{dl} (large amplitude change),

$Sd\emptyset$ (large phase angle change),

S_{dlp} (large angle change),

Sis/2 (amplitude approximately half of saturation current Is) and finally,

S_{in} (amplitude larger than normal).

For that the possibility of grounding faults occurring can be expressed by:

$$S_g = (S_{dl} \cup S_{d\emptyset} \cup S_{dlp}) \cap S_{in} \quad (2.16)$$

Maximum Fault grounding is expressed by

$$F_g = [m(Sdl) + m(Sd\emptyset) + m(Sdlp)] * m(Sln) \quad (2.17)$$

The possibility of short circuit fault is expressed

$$S_s = \left(Sdl \cup Sd\emptyset \cup \frac{Sis}{2}\right) \cap Sln \quad (2.18)$$

2.6 Application of NN to Power System Fault Classification

Different techniques have been used for power system fault classification in the past. In the next few sections, a few of these approaches are discussed.

2.6.1 Back Propagation Neural Network

Various applications of Neural Networks have been used in the past to improve the protection scheme in the transmission lines (Gaudrat, Giusiano, and Huiart, 2004). They have been used in fault classification, fault section estimation, adaptive relaying and fault diagnosis. Many of

these methods are based on back propagation, Radial basis function and Finite Impulse response neural networks. A few of these approaches are discussed here.

A typical Back Propagation Neural Network is a non-linear regression technique which attempts to minimize the global error. Its training includes both forward and backward propagation, with the desired output used to generate the error values for back propagation to iteratively improve the output. The back propagation neural network must have at least one input layer and one output layer. The hidden layers are optional. A Typical back propagation neural network is shown in the Figure 2.12 (Anderson, 2003):

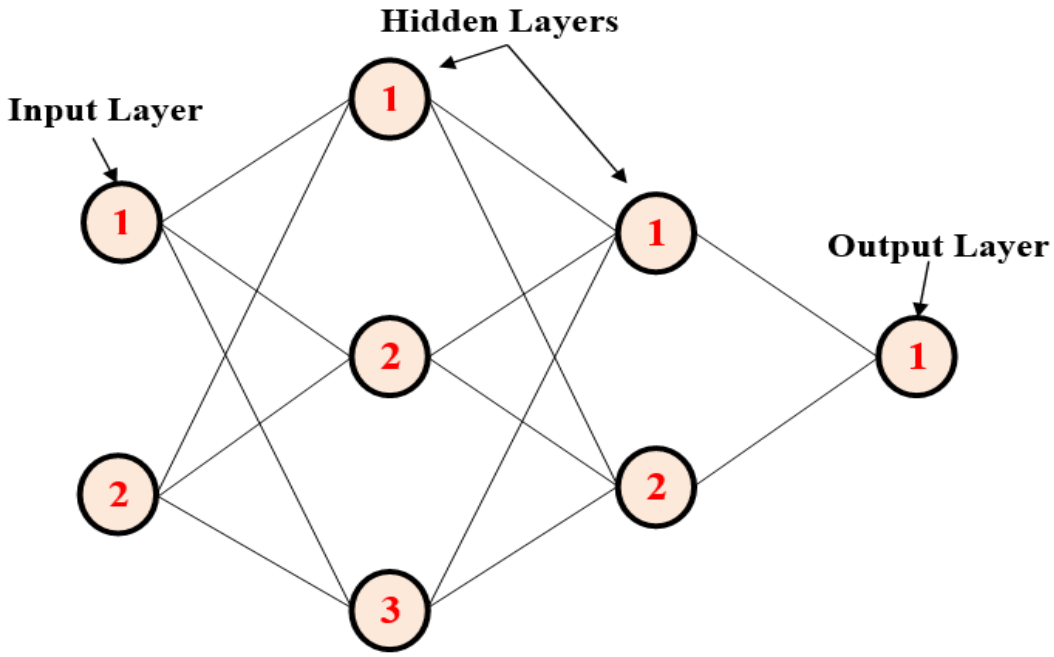


Figure 2.16: Back propagation neural network

The Back propagation neural network in Figure 2.16consists of four layers: an input layer with two neurons, hidden layers with three and two neurons respectively and an outputlayer with one neuron. In Figure 2.16, the output of a neuron in a layer goes to all neurons in the following layer and each neuron has its own weights. Initially the weights of the input layer

are assumed to be 1 for each input. The output of the back propagation neural network is reached by applying input values to the input layer, passing the output of each neuron to the following layer as input.

The number of neurons in the input layer depends on the number of possible inputs that are available, while the number of neurons in the output layer depends on the number of desired outputs. The number of hidden layers and how many neurons in each hidden layer cannot be well defined in advance, and could change per network configuration and type of data. In general, the addition of a hidden layer could allow the network to learn more complex patterns, but at the same time decreases its performance (Anderson, 2003). Ideally, a network configuration could be started using a single hidden layer, and more hidden layers would be added if it is noticed that the network is not learning as well as it should.

The Back-propagation training algorithm (Anderson, 2003) could be summarized as follows: The Input data sample is first presented to the network and then the network's output taken from the output layer is compared with the desired output and the error is calculated in each output neuron. And now for each neuron, a scaling factor called the local error is calculated which indicates how much higher or lower the output must be adjusted to match the desired output. The weights are modified to lower this local error. This process gets repeated until the error falls within the acceptable value (pre-defined threshold) which would indicate that the neural network has been trained successfully. On the other side, if the maximum number of iterations is reached, then it indicates that the training was not successful.

2.7 Wavelet Transform

Transient voltages and currents during fault carry high frequency component or harmonics which carry important information regarding type and location of fault. Wavelets can be very effectively used in analyzing transient phenomenon of these fault signals. Multi-resolution analysis is one of the tools of wavelet transform, which decomposes the original signal to low frequency signal called approximations (A) and high frequency signals called details (D) (Kim, and Aggarwal, 2000, 2001). Wavelet analysis is a relatively new signal processing tool and is applied recently by many researchers in power systems due to its strong capability of time and frequency domain analysis. The two areas with most applications are power quality analysis and power system protection (Santoso et al, 2013) and (Osman and Malik, 2004).

2.8 Characteristics of Neural Networks (NNs)

- The NNs exhibit mapping capabilities, that is, they can map input patterns to their associated output patterns.
- The NNs learn by examples. Thus, NN architectures can be 'trained' with known examples of a problem before they are tested for their 'inference' capability on unknown instances of the problem. They can, therefore, identify new objects previously untrained.
- The NNs possess the capability to generalize. Thus, they can predict new outcomes from past trends.
- The NNs are robust systems and are fault tolerant. They can, therefore, recall full patterns from incomplete, partial or noisy patterns.
- The NNs can process information in parallel, at high speed, and in a distributed manner (Coury, and Jorge, 1998) & (Philippe, and Daniel, 2003).

2.9 Basics of Artificial Neural Networks

The terminology of artificial neural networks has developed from a biological model of the brain (Kumar, Raghuwanshi, Singh, Wallender and Pruitt, 2002). A neural network consists of a set of connected cells; the neurons. The neurons receive impulses from either input cells or other neurons and perform some kind of transformation of the input and transmit the outcome to other neurons or to output cells. The neural networks are built from layers of neurons connected so that one layer receives input from the preceding layer of neurons and passes the output on to the subsequent layer.

A neuron is a real function of the input vector $(y_1, \dots, y_k)$. The output is obtained as;

$$f(x_j) = f\left(\alpha_j + \sum_{i=1}^{k} w_{ij}\, y_j\right) \tag{2.19}$$

Where;f is a function, typically the sigmoid (logistic or tangent hyperbolic) function. A graphical presentation of neuron is given in Figure 2.17. Mathematically, a Multi-Layer Perceptron network is a function consisting of compositions of weighted sums of the functions corresponding to the neurons.

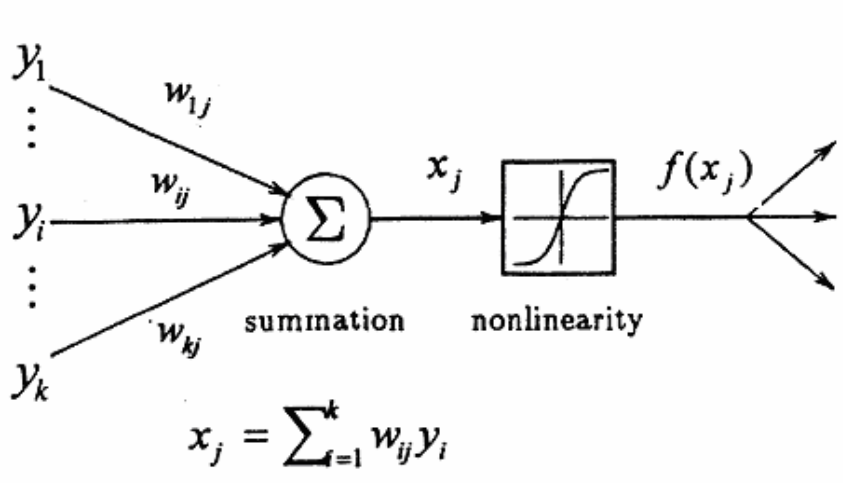


Figure 2.17: Single Neuron

2.9.1 Neural Networks Architectures

An ANN is defined as a data processing system consisting of a large number of simple highly inter connected processing elements (artificial neurons) in an architecture inspired by the structure of the cerebral cortex of the brain. There are several types of architecture of NNs. However, the two most widely used NNs are the Feed forward networks and the Recurrent networks.

In a feed forward network, information flows in one direction along connecting pathways, from the input layer via the hidden layers to the final output layer (Kohzadi, Boyd, Kermanshashi, and Kaastra, 1996). There is no feedback (loops) i.e., the output of any layer does not affect that same or preceding layer.

Recurrent networks differ from feed forward network architectures in the sense that there is at least one feedback loop. Thus, in these networks, for example, there could exist one layer with feedback connections. There could also be neurons with self-feedback links, i.e. the output of a neuron is fed back into itself as input.

2.9.2 Learning/Training Strategies

The basic concept behind the successful application of neural networks in any field is to determine the weights to achieve the desired target and this process is called learning or training. The three different learning mechanisms usually employed are supervised, unsupervised and reinforced learning.

2.9.2.1 Supervised Learning

In the case of supervised learning the network weights are modified with the prime objective of minimization of the error between a given set of inputs and their corresponding target values (Tarafdar, Haque and Kashtiban, 2005). Hence, the training data-set is known which is a set of inputs and the corresponding targets the neural network should output ideally. This is called supervised learning because both the inputs and the expected target values are known prior to the training of ANN.

2.9.2.2 Unsupervised Learning

On the other hand, in the case of unsupervised learning, the relationship between the inputs and the target values are not known. The neural network is trained with a training data set in which only the input values are known. Hence, it is very important to choose the right set of examples for efficient training. These examples are usually chosen using some sort of a similarity principle (Tarafdar, Haque and Kashtiban, 2005). The most commonly used unsupervised learning algorithms are the Self-Organizing Map (SOM) and the Adaptive Resonance Theory (ART).

2.9.2.3 Reinforced learning

In this method, a teacher though available, does not present the expected answer but only indicates if the computed output is correct or incorrect. The information provided helps the network in its learning process. A reward is given for a correct answer computed and a penalty for a wrong answer. But, reinforced learning is not one of the popular forms of learning.

The learning strategy employed depends on the structure of the neural network. Feed-forward networks are trained using the supervised learning strategy. The supervised learning strategy for a feed-forward neural network has been shown in the Fig. 2.18.

The set of input-output pairs (shown in Fig. 2.18) that are used to train the neural network are obtained prior to the training process either by using physical measurements or by performing some kind of simulations. Fig. 2.18 shows that the teacher teaches the neural network to modify its weights according to the error 'e' between the outputs and the targets. The weights of the neural network are then modified iteratively according to (2.20). The general idea behind supervised learning and the mathematics involved has been adopted from (Tarafdar, et al, 2005).

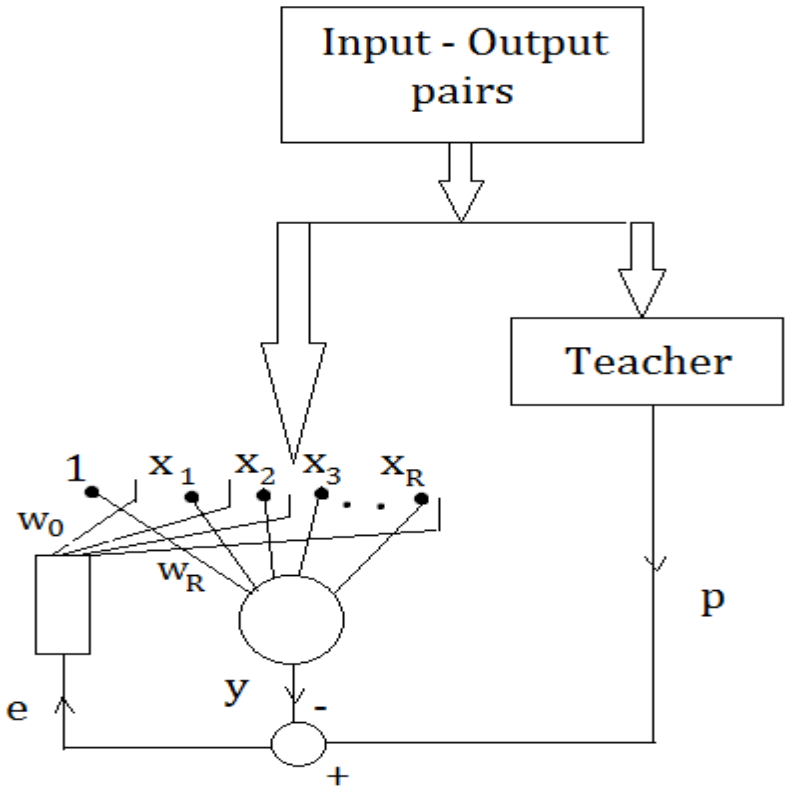


Figure 2.18: Scheme of supervised learning.

$$w_{ji}(n+1) = w_{ji}(n) + \Delta w_{ji}(n) \qquad (2.20)$$

Where: $w_{ji}(n)$ and $w_{ji}(n+1)$ are the previous and the modified weights connected between the ith and the jth adjoining layers respectively. $\Delta w_{ji}(n)$stands for the correction or modification factor and n stands for the number of the iteration. If the jth neuron in a single layer neural networkis considered, the training efficiency is enhanced by minimizing the error between the actual output of the jth neuron and the output that has been dictated by the teacher. Let $y_j(n)$ and $p_j(n)$be the actual and the teacher-requested outputs for the jth neuron in the nth iteration. Then the error value of that iteration is given by equation 2.21.

$$e_j(n) = p_j(n) - y_j(n) \tag{2.21}$$

The vector e(n) that stores the values of all the errors is also a function of the weights w(n) for the corresponding layers' inputs. The value by which the weighing coefficients change (also called the correction factor) is given by the followingequation 2.22.

$$\Delta w_{ji}(n) = \eta e_j(n) x_i(n) \tag{2.22}$$

Where: x_i is the ith input signal and η is the rate at which the learning process takes place. As mentioned earlier, learning process aims at the minimization of the error function. The same criterion can also be achieved by the usage of a Least Squares Method (LSM). Hence, if there are L neurons in a particular network, the cost function to be ultimately minimized is given by equation 2.23.

$$S_2(w) = \frac{1}{2}\sum_{j=1}^{L}(p_j - y_j)^2 \tag{2.23}$$

If the number of learning pairs with an input vector x(n) and an output vector d(n) of the form (x(n), d(n)) are P in the training set, then during the nth iteration of the learning process, we have:

$$S_2\big(w(n)\big) = \frac{1}{2}\sum_{n=1}^{p}\sum_{j=1}^{L}(p_j(n) - y_j(n))^2 \qquad (2.24)$$

Since the activation functions that are employed are often non-linear, minimization of the aboveequation 2.24 is a non-linear problem. Several numerical methods that can handle non-linear functions effectively are available and are based on the steepest-decent method. The steepest-decent method is an extension to the Laplace's method of integral approximation where the contour integral in a complex plane is deformed to approach a stationary point in the direction of the steepest decent (Vasilic, and Kezunovic, 2004). The back-error-propagation learning technique is based on the steepest-decent method and is usually widely applied in a version known as the Levenberg-Marquardt algorithm (Vasilic, and Kezunovic, 2004).

The back-error-propagation algorithm chooses random weights for the neural network nodes, feeds in an input pair and obtains the result. Then, the error for each node is calculated starting from the last stage and by propagating the error backwards (Lahiri, Pradhan and Mukhopadhyaya, 2005). Once this is done, the weights are updated and the process is repeated with the entire set of input output pairs available in the training data set. This process is continued till the network converges with respect to the desired targets. The back-error-propagation technique is widely used for several purposes including its application to error functions (other than the sum of squared errors) and for the evaluation of Jacobian and Hessian matrices. The correction values are calculated as functions of errors estimated from

the minimization ofequation 2.24. This process is carried out layer by layer throughout the network in the backward direction. This algorithm is pictorially depicted in Fig 2.19.

The corresponding weighing vectors are shown in blocks $\mathbf{A^{(M)}}, \mathbf{A^{(M-1)}}, \ldots, \mathbf{A^{(1)}}$ and the errors that are propagated to the lower layers are calculated and stored in the blocks $\mathbf{B^{(M-1)}}, \mathbf{B^{(M-2)}}, \ldots, \mathbf{B^{(2)}}$. The back-error-propagation algorithm has been implemented in many ways but the basic idea remains the same. The only thing that changes in each of these implementations is the method used for the calculation of the weights that are iteratively upgraded when passed backward from layer to layer in the neural network.

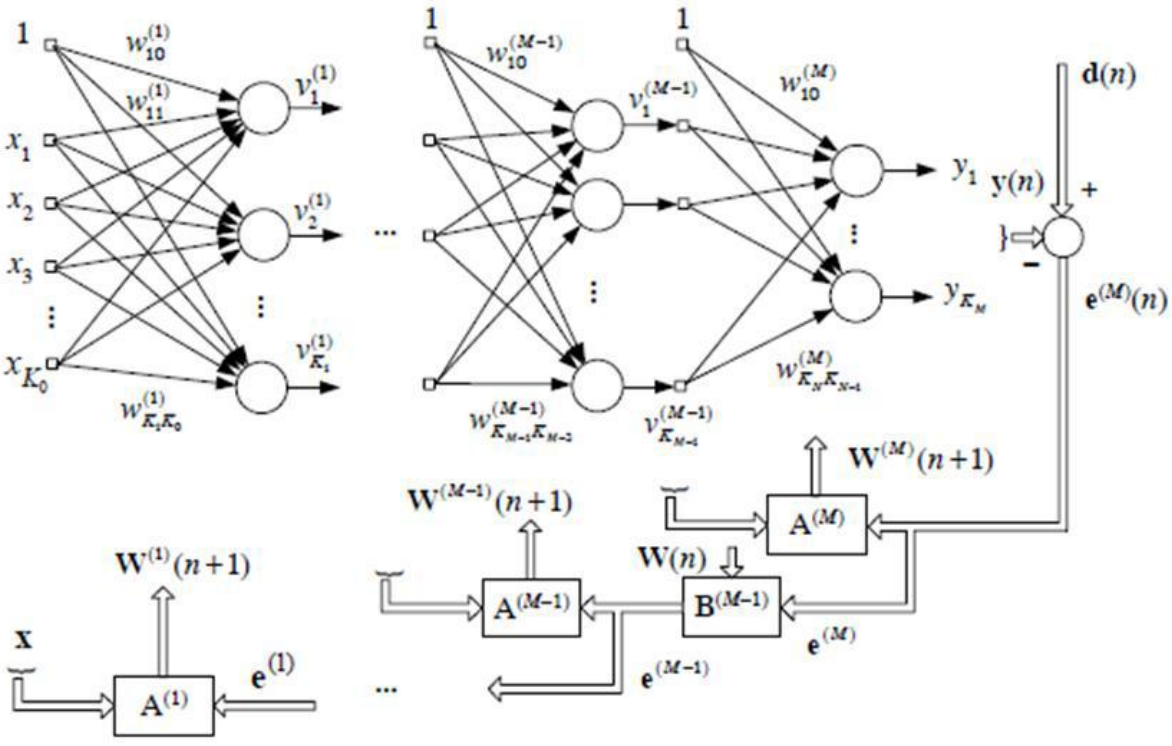


Figure 2.19 Structure of back-error-propagation algorithm [adopted from Vasilic, and Kezunovic, (2004)]

The modifications involved are also used in the training process of recurrent networks (Vasilic and Kezunovic, 2004). The rate at which the learning process takes place can be estimated by keeping a check on the correction values in successive stages. The total number of iterations required to achieve satisfactory convergence rate depends on the following factors:

- Size of the neural network
- Structure of the network
- The problem being investigated
- The learning strategy employed
- Size of the training/learning set

The efficiency of a chosen ANN and the learning strategy employed can be estimated by using the trained network on some test cases with known output values. This test set is also a part of the learning set. Hence, the entire set of data consists of the training data set along with the testing data set. The former is used to train the neural network and the latter is used to evaluate the performance of the trained artificial neural network.

2.10 Types of Neural Networks

The most important class of neural networks for real world problems solving includes

- Multilayer Perceptron
- Radial Basis Function Networks
- Kohonen Self Organizing Feature Maps

2.10.1 Multilayer Perceptrons

The most popular form of neural network architecture is the multilayer perceptron (MLP). A multilayer perceptron:

- has any number of inputs.
- has one or more hidden layers with any number of units.
- uses linear combination functions in the input layers.
- uses generally sigmoid activation functions in the hidden layers.
- has any number of outputs with any activation function.
- has connections between the input layer and the first hidden layer, between the hidden layers, and between the last hidden layer and the output layer.

Given enough data, enough hidden units, and enough training time, an MLP with just one hidden layer can learn to approximate virtually any function to any degree of accuracy. (A statistical analogy is approximating a function with *n*th order polynomials.) For this reason, MLPs are known as universal approximators and can be used when there is little prior knowledge of the relationship between inputs and targets. Although one hidden layer is always sufficient provided there are enough data, there are situations where a network with two or more hidden layers may require fewer hidden units and weights than a network with one hidden layer, so using extra hidden layers sometimes can improve generalization (Dalstein and Kulicke, 1995) and (Magnago and Abur, 2014).

2.10.2 Radial Basis Function Networks

Radial basis functions (RBF) networks are also feedforward, but have only one hidden layer. An RBF network:

- has any number of inputs.

- typically has only one hidden layer with any number of units.
- uses radial combination functions in the hidden layer, based on the squared Euclidean distance between the input vector and the weight vector.
- typically uses exponential or softmax activation functions in the hidden layer, in which case the network is a Gaussian RBF network.
- has any number of outputs with any activation function.
- has connections between the input layer and the hidden layer, and between the hidden layer and the output layer.

MLPs are said to be distributed-processing networks because the effect of a hidden unit can be distributed over the entire input space. On the other hand, Gaussian RBF networks are said to be local-processing networks because the effect of a hidden unit is usually concentrated in a local area centered at the weight vector.

2.10.3 Kohonen Neural Network

Self-Organizing Feature Map (SOFM, or Kohonen) networks are used quite differently to the other networks. Whereas all the other networks are designed for supervised learning tasks, SOFM networks are designed primarily for unsupervised learning. At first glance this may seem strange. Without outputs, what can the network learn? The answer is that the SOFM network attempts to learn the structure of the data (Lipmann, 1989) and (Bouthiba, 2004). One possible use is therefore in exploratory data analysis. A second possible use is in novelty detection.

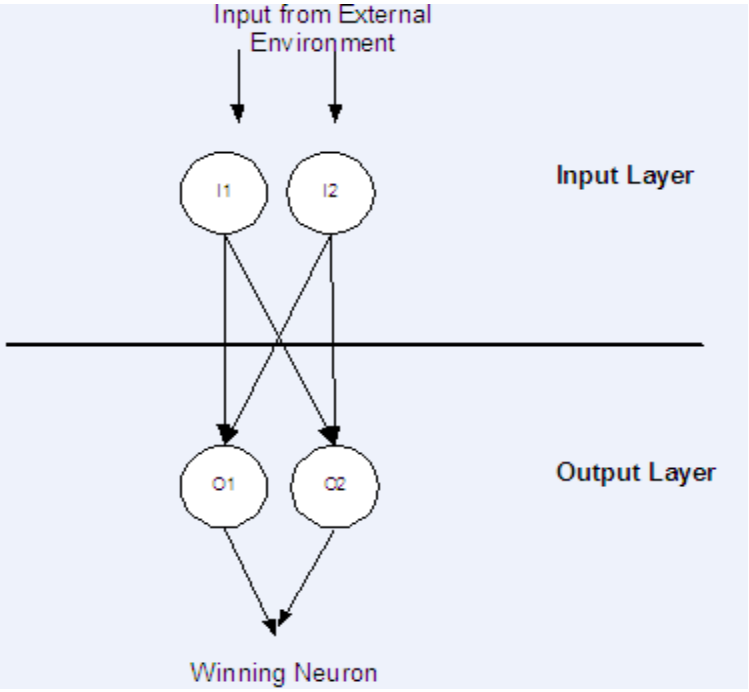


Figure 2.20: A Kohonen Neural Network Applications

SOFM networks can learn to recognize clusters in the training data, and respond to it. If new data, unlike previous cases, is encountered, the network fails to recognize it and this indicates novelty. A SOFM network has only two layers: the input layer, and an output layer of radial units (also known as the topological map layer).

Neural networks have proven to be effective mapping tools for a wide variety of problems and consequently they have been used extensively by practitioners in almost every application domain ranging from agriculture to zoology. Since neural networks are best at identifying patterns or trends in data, they are well suited for prediction or forecasting applications. A very small sample of applications has been indicated in the next paragraph.

Anderson, 2003 developed neural network model for forecasting financial and economic time series. Kohzadi, Boyd, Kermanshashi, and Kaastra, 1996 used a feed-forward neural network to compare ARIMA and neural network price forecasting performance. Lipmann,

(1989)demonstrated the applicability of neural network technology for plant diseases forecasting. Tarafdar, Haque and Kashtiban, 2005 provided the general summary of the work in ANN forecasting, providing the guidelines for neural network modelling, general paradigm of the ANNs especially those used for forecasting, modelling issue of ANNs in forecasting and relative performance of ANN over traditional statistical methods. Tarafdar, Haque and Kashtiban, 2005also developed the models for predicting milk production from farm inputs using standard feed forward ANN. Kumar, et al, (2002) studied utility of neural networks for estimation of daily grass reference crop evapotranspiration and compared the performance of ANNs with the conventional method used to estimate evapo-transpiration.

Omid, Omidvar, Elliott,(1997) developed MLP based forecasting model for maximum and minimum temperatures for ground level at Dum Dum station, Kolkata on the basis of daily data on several variables, such as mean sea level pressure, vapour pressure, relative humidity, rainfall, and radiation for the period 1989-95. Gaudrat, Giusiano and Huiart, (2004) compared the performance of MLP and that of linear regression for epidemiological data with regard to quality of prediction and robustness to deviation from underlying assumptions of normality, homoscedasticity and independence of errors.

The large number of parameters that must be selected to develop a neural network model for any application indicates that the design process still involves much trial and error. The next section provides a practical introductory guide for designing a neural network model.

2.11 Development of an ANN model

The various steps in developing a neural network model are:

A. Variable selection

The input variables important for modelling variable(s) under study are selected by suitable variable selection procedures.

B. Formation of training, testing and validation sets

The data set is divided into three distinct sets called training, testing and validation sets. The training set is the largest set and is used by neural network to learn patterns present in the data. The testing set is used to evaluate the generalization ability of a supposedly trained network (Pao and Sobajic, 1988). A final check on the performance of the trained network is made using validation set.

C. Neural network architecture

Neural network architecture defines its structure including number of hidden layers, number of hidden nodes and number of output nodes etc (Hagan, Demuth, Beale, 1996).

• **Number of hidden layers:** The hidden layer(s) provide the network with its ability to generalize. In theory, a neural network with one hidden layer with a sufficient number of hidden neurons is capable of approximating any continuous function. In practice, neural network with one and occasionally two hidden layers are widely used and have to perform very well.

• **Number of hidden nodes:** There is no magic formula for selecting the optimum number of hidden neurons. However, some thumb rules are available for calculating number of hidden neurons. A rough approximation can be obtained by the geometric pyramid rule proposed by (Master, 1993). For a three-layer network with n input and m output neurons, the hidden layer would have *sqrt(n*m)* neurons.

• **Number of output nodes:** Neural networks with multiple outputs, especially if these outputs are widely spaced, will produce inferior results as compared to a network with a single output.

• **Activation function:** Activation functions are mathematical formulae that determine the output of a processing node. Each unit takes its net input and applies an activation function to it. Non-linear functions have been used as activation functions such as logistic, tanh etc. The purpose of the transfer function is to prevent output from reaching very large value which can 'paralyze' neural networks and thereby inhibit training. Transfer functions such as sigmoid are commonly used because they are nonlinear and continuously differentiable which are desirable for network learning (Eisa and Tayeb, (2013) and (Zhihong and Jean-Claud, 2000).

D. Evaluation Criteria

The most common error function minimized in neural networks is the sum of squared errors and this is the one used in this work. Other error functions offered by different software include least absolute deviations, least fourth powers, asymmetric least squares and percentage differences.

E. Neural Network Training

Training a neural network to learn patterns in the data involves iteratively presenting it with examples of the correct known answers. The objective of training is to find the set of weights between the neurons that determine the global minimum of error function. This involves decision regarding the number of iteration i.e., when to stop training a neural network and the selection of learning rate (a constant of proportionality which determines the size of the weight adjustments made at each iteration) and momentum values (how past weight changes

affect current weight changes) (Stacchini et al, 2001) and (Venkatesan and Balamurugan, 2001).

2.11.1 Software

Several softwares on neural networks have been developed, to cite a few;

Commercial Software: Statistica Neural Network, TNs2Server, DataEngine, Know Man Basic Suite, Partek, Saxon, ECANSE - Environment for Computer Aided Neural Software Engineering, Neuroshell, Neurogen, Matlab:Neural Network Toolbar.

Freeware Software: Net II, Spider Nets Neural Network Library, NeuDC, Binary Hopfeild Net with free Java source, Neural shell, PlaNet, Valentino Computational Neuroscience Work bench, Neural Simulation language version-NSL, Brain neural network Simulator.

The computing world has a lot to gain from neural networks. Their ability to learn by example makes them very flexible and powerful. A large number of claims have been made about the modelling capabilities of neural networks, some exaggerated and some justified. Hence, to best utilize ANNs for different problems, it is essential to understand the potential as well as limitations of neural networks. For some tasks, neural networks will never replace conventional methods, but for a growing list of applications, the neural architecture will provide either an alternative or a complement to these existing techniques (Zadeh, 2006).

2.12 Review of Related Literature

In the work of Mamta and Patel, (2012), they presented the application of wavelet multi resolution analysis in combination with artificial neural network for classification and location

of single line to ground fault only. The method uses energy of spectrum of detail coefficients at levels 1 and 5 (D1 and D5) for two consecutive data windows for classification and location of faults as shown in Figure 2.21.

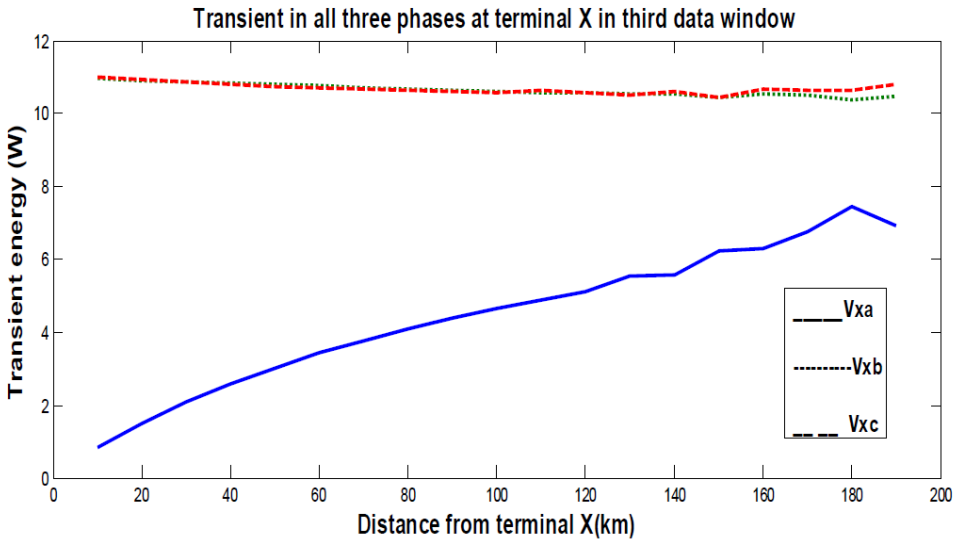


Fig. 2.21: Decaying transient energy of faulted and healthy lines

Wavelet transform was used to get details D1 and D5 of the voltage signals. Capabilities of neural network in pattern classification were utilized to classify the faults. After successful classification, details of fault signals were used to locate the fault. Simulation studies were performed for fault conditions with faults at different phases, at different locations and at different fault inception angles and performance of the proposed scheme was investigated.

The classification of faults was close to exact and the location of the faults was identified with above 90% accuracy. But the limitations of this work are that results were provided for only one type of fault which is single line to ground faults. One of the targets of our proposed algorithm and scheme will be to extend it to other faults also with improved effectiveness.

Hasabe and Vaidya, (2014)in their work performed fault detection and fault classification on 220kV transmission line. Their technique depends upon the current signals. The features were extracted from the current signals by using wavelet transform. A moving data window of one cycle (400 samples) was taken and decomposition was done and energy of the details coefficients at level 5 was obtained for each data window. As the fault signals contain the high amount of harmonic components, the energy of the signal increases at the occurrence of fault as shown in Fig. 2.22.

The feature vector was then given as input to the neural network. Simulation studies were performed and the performance of the scheme with different system parameters and conditions was investigated. The test result showed that the accuracy obtained with the "tansig-logsig" transfer function for hidden layers I and II was satisfactory. But, the paper dealt with fault detection and classification only and could not generate enough information as per the location of the faults. Hence, this dissertation has tried to extend the research to other power system protection problems, such as finding fault location as well.

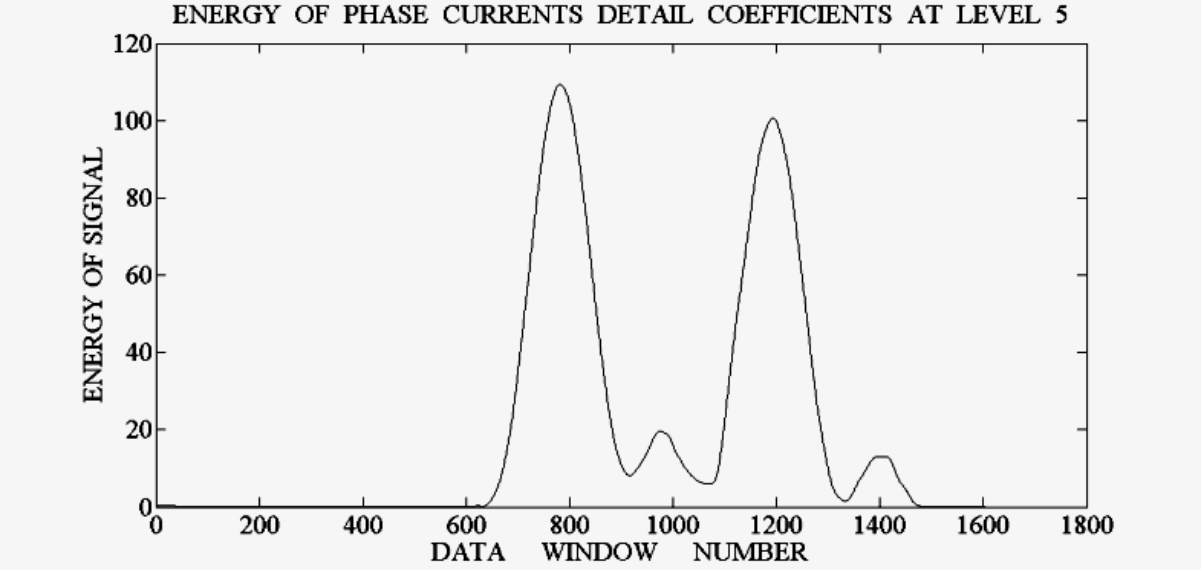


Fig. 2.22: Energy of the detail level 5 vs. window number.

Fault classification and location on a series compensated transmission line using artificial neural network of different faults namely SLG, LL, LLG, and LLL at 50km, 100km, 150km, 200km, 250km, 300km has been performed by Girish and Nitin, (2015). A feed forward neural network was created to accurately classify and locate fault as close as possible to its actual location. The simulation was done in Matlab 2014 using neural network tool box on a 300km series compensated line. The result showed that the fault was classified accurately in shortest of time & the fault was located within the marginal limits. The average error was calculated, & found that it was below 5%. But, aside the challenge of not pre-processing the input data vector of the neural network, the intervals of the fault locations simulated were too high or far apart and as a result interpolating for a value (location) between any two consecutive locations did not yield expected result/response.

The challenge of pre-processing the input data-set was taken care by Kale, Bhide, Bedekar and Mohan, (2008) as they proposed an accurate technique of automation of identification of faults on parallel transmission lines. The method employed depended on the current signals extracted from the local relay location. Wavelet Transform was used to extract distinctive features in the input signals. This feature vector then acts as input to the neural network improving its speed and accuracy. Capabilities of neural network in pattern classification were utilized. Simulation studies were performed and the performance of the scheme with different system parameters and conditions was investigated. The proposed algorithm was found to be immune to the effect of mutual coupling, fault resistance, remote end infeed, fault location and fault inception angle.But there were no adequate information/data about the fault location. Though the paper deals with fault classification only but can be extended to the other power system protection problems such as finding fault location.

Atul and Navita, (2015) presented a neural network approach for fault detection and classification in double circuit transmission line. The neural network was trained by various sets of data available from simulation of the model for different faults conditions. Their proposed method is as shown in Figure 2.23. In Figure 2.23, the current signals from the power system were sampled at 12.5 KHz frequency. For fault detection and classification seven samples of current was taken and given as inputs to ANN which gives output either 1 or 0 indicating fault state or no-fault state.

The obtained simulation result showed that the proposed scheme was able to classify the faulty phase and faulty transmission line correctly. But the entire process of detecting and classifying fault by ANN was cumbersome and sluggish and a lot is involved. The reason for this is that the raw input data sets were fed into the neural network without pre-processing them. Having identified this as a challenge, a concise effort has been made in this work to address that, hence a processing tool called wavelet transform has been used to take care of the challenge.

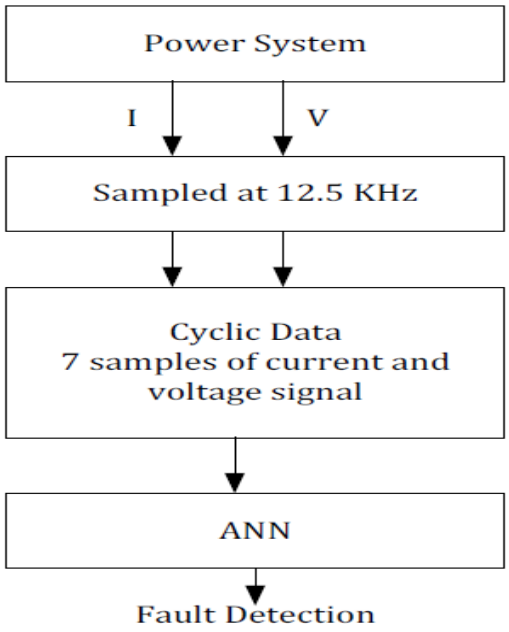


Figure 2.23: ANN Scheme for fault detection and classification used by Atul and Navita, (2015)

Hiroyuki, Hikaru, Toshiyuki and Shoichi, (2002) understood the importance of pre-processing the extracted signals before feeding them into the neural network, which was why they employed the processing tool of Fast Fourier transform (FFT). In their paper, a new hybrid method was proposed to handle a fault detection problem that estimates the location and type of fault. The proposed method applied Fast Fourier transform (FFT) to the measured current waveforms in three phases to extract features of fault. The results in frequency domain were given to Deterministic Annealing (DA) clustering as input variables. FFT itself also works as a pre-conditional technique for Multi-Layer Perceptron (MLP). The proposed method made use of DA clustering and MLP. DA clustering was more effective in a sense of global clustering that was not affected by the initial solutions. DA clustering played an important role to classify input data of MLP into clusters. MLP was constructed at each cluster. That allowed MLP to learn efficiently because of data similarity. The proposed method was applied to a sample system. A comparison was made between the proposed and conventional methods. The simulation results have shown that the proposed method was much better than the conventional ones. Compared with the conventional method, the proposed method had reduced the average error of 30% and the maximum error of 17.1% in terms of the fault location. Also, it had contributed to improving the recognition rate of 7.2% in terms of the fault type (Hiroyuki, et al, 2002). But one major issue with this approach is the choice of data pre-processing tool used; Fast Fourier transform which is associated with resolution problem.

And ordinary Fourier transform of a signal gives information about all the frequencies present in the signal but does not give any information about the time at which these frequencies were present. Wavelet transform is a tool which helps the signal to be analyzed in time as well as

frequency domain effectively. It provides non-uniform division of frequency domain i.e. it uses short window at high frequencies and long window at low frequencies. Using multi-resolution analysis, a particular band of frequencies present in the fault signal can be analyzed. Fourier transform and wavelet transform are the two major tools which are a great help in frequency domain analysis of any signal (Abdollahi and Seyedtabali, 2010).

In their work, Nanand Mladen, (2014) proposed an improved solution based on wavelet transform and self-organized neural network aimed at solving the problem of differentiating the internal faults from external ones using local-end data and providing the exact fault type at the same time. The paper applied advanced signal processing and artificial intelligence techniques to achieve that objective. A comprehensive protection scheme wasdesigned as shown in Figure 2.24.

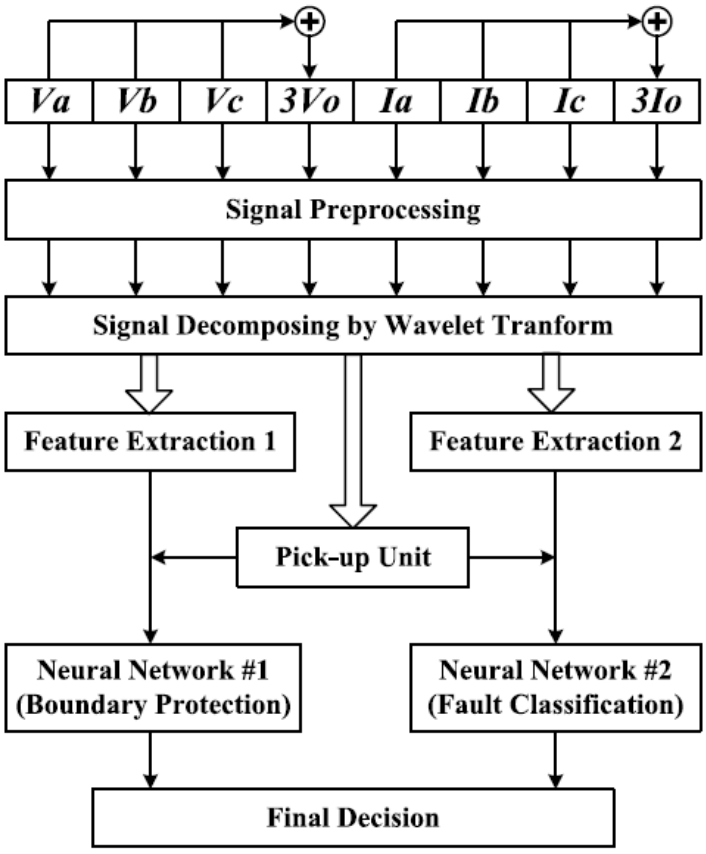


Figure 2.24: Overview of proposed protection scheme by Nanand Mladen, (2014)

Figure 2.24 shows the three-phase secondary voltage and current signals ($V_a, V_b, V_c, and\ I_a, I_b, I_c$) which were obtained at the sampling rate of 200 kHz. The zero-sequence voltage and current $3V_0\ and\ 3I_0$ were obtained by adding up the values. Theproperties of the proposed scheme were as explained. The signal pre-processing stage eliminated most of the influences from pre-fault loads, system conditions and power swings while the wavelet transform provided an efficient way to extract signal components at different frequency bands. The neural network provided an intelligent method and a "soft" criterion for feature comparison (Nan at el, 2014). Both high frequency details and low frequency approximations were used in the proposed method and that can avoid confusing fault conditions with other kinds of non-fault disturbances. The protection tasks were distributed into two neural networks so that each neural network had different task as both neural networks took half cycle data window, therefore the protection speed was satisfied. Unfortunately, as a serious weakness, the proposed approach has the generalization for different fault and system operating conditions, as well as different system parameters. It needs to be further studied and adjusted to cope with different system structures such as parallel lines and multi-terminal lines.

Mohammad and Rahul, 2014 worked on finding different types of fault in transmission lines with the help of two different materials. The scheme used neural network and wavelet transform together, to choose a proper way for solving the problem. Wavelet transform has strong mathematical, very fast and accurate tools for transient signal processing in the transmission lines. Artificial neural network that can make a difference between measured signals and associated signal that has different pattern was used (Mohammad at el, 2014). The

algorithm used time-frequency analysis of faulted transient line with the help of wavelet transform, and then this allowed the artificial neural network to identify what phase was inflicted with fault. Matlab software was used for simulation of fault signals and verifying the correctness of the algorithm. There were different types of fault which was given to the software and the result showed where and what phase that was faulty. Notwithstanding, the choice of the size of the data set used for the simulation was very small and would not allow for conclusive decision to be made.

Karthikeyan, (2007) aimed at detecting and classifying the power system transmission line faults. To deal with the problem of an extremely large data-set with different fault situations, a three step optimized Neural Network approach was proposed. The approach utilized Discrete Wavelet Transform for detection and two different types of self-organized, unsupervised Adaptive Resonance Theory Neural Networks for classification. The fault scenarios were simulated using Alternate Transients Program and the performance of this highly improved scheme was compared with the existing techniques. The simulation results proved that the proposed technique handled large data more efficiently and time of operation was considerably less when compared to the existing methods. Being highly interested in the proposed network handling large data, Karthikeyan, (2007) was unable to suggest a model to test and validate the results obtained from the neural networks; by implication the accuracy of the results cannot be guaranteed.

Mollanezhad and Akbari, (2013) presented a Probabilistic Neural Network (PNN) and new feature selection technique for fault classification in transmission lines. Initially, wavelet transform was used for feature extraction from half cycle of post-fault three phase currents at one end of line. In the proposed method, three classifiers corresponding to three phases were

fed by normalized particular features as Wavelet Energy Ratio (WER) and Ground Index (GI).

Figure 2.25 shows the structure of the proposed fault classification system by Mollanezhad and Akbari, (2013). The algorithm consists of three stages, including feature extraction, feature selection and fault classification. The system takes half cycle of the post-fault from the three phase currents at the current recorder (relay location). The ground current was calculated from the samples of the three line currents $I_g = I_a + I_b + I_c$. In this algorithm, three PNNs and one ground detector (GD) had been used for fault classification method. Each of the three PNNs (PNNa, PNNb, PNNc) is used to identify the faulted phase(s) and the ground detector (GD) was used to determine the involvement of the ground in the fault. At the output of each PNN, the value '1' and '0' denotes the presence or absence of the fault, respectively**.**

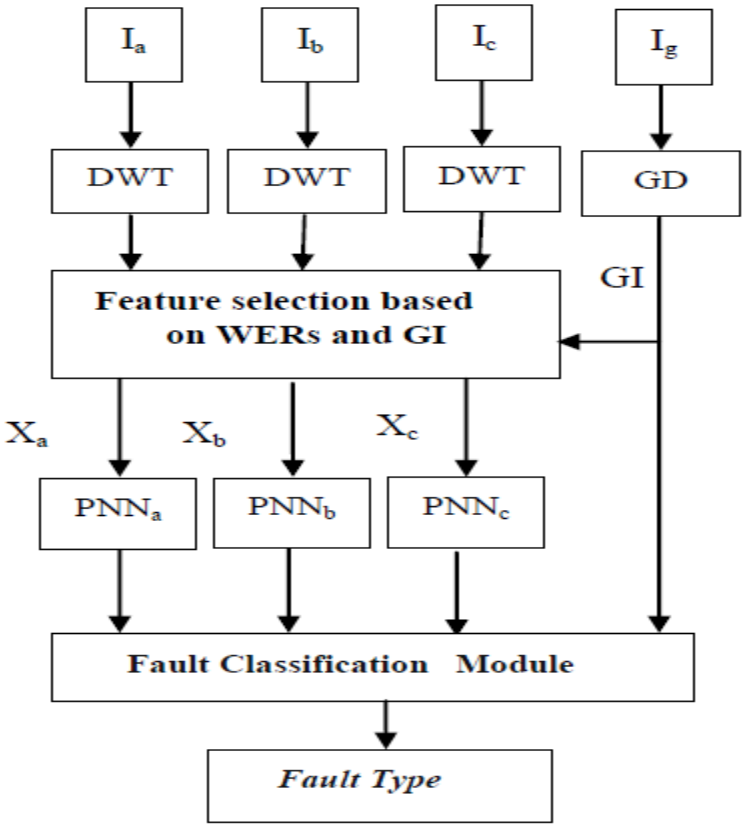


Figure 2.25: Structure of the fault classifier by Mollanezhad and Akbari, 2013.

The PNNs were trained to provide faulted phase selection in different ten fault types. Finally, logic outputs of classifiers and GI identify the fault type. The feasibility of the proposed algorithm was tested on transmission line using PSCAD/EMTDC software. Variation of operating conditions in train cases was limited, but it was wide for test cases. Also, quantity of the test data-sets was larger than the train data-sets. The results indicated that the proposed technique was of high speed, accurate and robust for a wide variation in operating conditions and noisy environments. But the accuracy of the results was not verified more so when only half cycle post fault currents was used for the classification instead of the entire cycle.

The advantages of DWT over FFT and STFT in feature extraction were demonstrated by Amit and Abhijit, (2016) in their proposed method for locating fault point. The advantages are

due to good time and frequency localization characteristics. Analysis results presented, clearly showed that particular wavelet components can be used as the features to locate the fault. Then an accurate fault location technique based on ANN was developed. As an ANN was trained to classify the fault type, another separate ANNs were designed to accurately locate the actual fault position on a practical system. In this respect, three-layer feed-forward ANNs and the Levenberg-Marquardt (LM) algorithm was used to adjust the weights and biases to achieve the desired non-linear mapping from inputs to outputs. Through a series of tests and modifications, it was shown that the ANNs can very accurately classify the type of fault under different system and fault conditions. Thus, it can be concluded that the proposed approach based on combined WT and ANN was robust to different case studies; this is a significant advantage and can be directly attributed to the fact that WT technique effectively extracts the very crucial time-frequency features from transient signals and ANN approach is able to give a very high accuracy in the fault classification and fault location. But these results were based on the probabilistic predictions as a result they are subject to failure.

In the work of Srinivasa and Baddu, (2013), the proposed algorithm presented a fault discrimination method based on the three-phase current and voltage waveforms measured when fault events occurred in the power transmission-line network. The algorithm for fault classification employed wavelet multi-resolution analysis (MRA) to overcome the difficulties associated with conventional voltage and current based measurements due to effect of factors such as fault inception angle, fault impedance and fault distance. The proposed algorithm for fault location was different from conventional algorithms that were based on deterministic computations on a well-defined model to be protected. The wavelet transform captured the dynamic characteristics of the non-stationary transient fault signals using wavelet MRA coefficients. Using wavelet MRA technique, the summation of detail coefficients for sixth

level were extracted from the current signal. From the magnitude of only the detail coefficient summations, the presence of fault in a particular phase was detected. A generalized algorithm based on wavelets was verified for the classification of transmission line faults. The advantage of this algorithm is that it is independent of fault location, impedance and inception angle but it lacks the capability to provide enough information that will help to crystallize the location of the faults.

Xinzhou, Wei and Tao, (2009) in their work titled:"Fault classification and faulted-phase selection based on the initial current travelling wave", proposed an algorithm of fault classification and faulted phase selection for a single circuit transmission line based on the initial current travelling wave. Identification of simultaneous faults on transmission system using wavelet transform was proposed. However, authors have reported that further improvement in their proposed algorithm was needed to achieve the desired accuracy.

Das, Singh and Sinha, (2006) showed an application of artificial neural network approach to fault classification for double circuit transmission lines using superimposed sequence components of current signals. Comparison of the Fourier Transform method with Wavelet Transform method for detection and classification of faults on transmission lines was done. But the authors have reported that wavelet transform based approach gave better results only when more than one phase was involved in the fault. It may be noted that majority of the faults are ground faults that involve only one of the phase conductors and ground.

Bo, 1998 used a specially designed multi-channel filter to extract the transient current signals for two signal outputs $If1, If2$ with centre frequency at 80kHz and 1kHz respectively. Then the ratio of the energy spectrum for $If1, If2$ was calculated and compared to a threshold to

find out whether the fault is internal or external. The advantage of this method was justified by the result from a performance study. Still some issues are yet to be addressed by this method:

a) The direction of the external faults cannot be distinguished since only the current signal was used. The method also had no phase selection function available;
b) The theoretical basis for selection of the centre frequency of the extracted features and selection of the thresholds was not apparent;
c) The reliability of the method was unknown since only high frequency signal was used. It may be affected by the disturbance from noise, switching, lightning, etc;
d) There were no extensive studies provided for the performance evaluation under various fault conditions. As mentioned earlier, the boundary conditions are highly dependent on fault type, fault resistance, fault angle, etc.

Vasilic and Kezunovic, (2005) provided a new boundary protection scheme aimed at solving those issues that could not be addressed by Bo, (1998). First of all, the voltage and current signal will both be used; this can provide more information about the direction of the fault point. The new scheme used wavelet transform as the feature extraction tool thus there was no need to design extra filters. Wavelet transform has a strong capability of extracting the signal component under different frequency bands while retaining the time domain information. Secondly, the extracted features would be handled using a self-organized neural network algorithm (Vasilic, and Kezunovic, 2005). With its strong capability of generalization and training mechanism, it could be used as an alternative solution when theoretical basis for

dealing with the fault generated high frequency signal components was not well defined. The neural network based algorithm is also capable of implementing a superior fault classification scheme. Finally, the new scheme used both the low and high frequency components of the fault signal to eliminate impact from non-fault disturbances. The reliability and robustness of the method will be verified by an extensive study for various kinds of faults.

2.13 Summary of Literature Review/Knowledge Gap

This work has successfully reviewed a lot of works done by different researchers in application of MRA and/or NN pattern recognition for fault identification and location in transmission lines.

However, none of the works in the available literatures, has yet explored pattern recognition based fault identification and location in an overhead transmission line using the summation of the decomposed detail coefficients of the extracted faulty voltage and current waveforms for all the three phases for ten different faults and also for non-fault case.

Additionally, the case study network (the Nigerian 330kV Ikeja-West to Benin Power transmission line) is yet to have simulation procedures and results peculiar and specific to its parameters.

CHAPTER THREE

MATERIALS ANDMETHODS

The materials used in this work are as follows:

3.1 Laptop Computer System

Shortened to just laptop is a small, portable personal computer (PC) with a ‘‘clamshell’’ form factor, typically having a thin LCD or LED computer screen mounted on the inside of the upper lid of the clamshell and an alphanumeric keyboard on the inside of the lower lid. The clamshell is opened up to use the computer. For this work, given the volume of data and multiple continuous tasks of neural networks, there is minimum specifications of laptop that can be used. More so when the MATLAB software 2016 edition used for this work could not be run on a computer with 32-bit operating system. Hence the basic specification of the HP 250 laptop computer used for this work is as follows:

Processor:	Intel(R) Celeron(R) CPU N2820 @ 2.13GHz
Installed memory (RAM):	4.00 GB (3.90 GB usable)
System type:	64-bit Operating System, x64-based processor
Storage:	500GB
Screen Size:	15.6 inches
Windows edition:	Windows 8.1 Pro @ 2013 Microsoft Corporation.

3.2 Ikeja West – Benin 330kVTransmission Line

The Nigerian Ikeja West – Benin Electric Power Transmission Line is the second longest 330kV transmission line of about 280kM long, located at the southern part of Nigerian grid system. At the Ikeja West end it has the Egbin Generating Station of 637MW feeding into it and at the Benin end it has two generators supplying it. The generators are Sapele Generating Station(GS) with 70MW capacity and Delta IV GS with capacity 498MW. The three main factors on which the choice of this network is made are basically because of its long distance, availability of records and proximity to the research center.

3.3 Power World Simulator for power Flow Direction Determination

For the simulation of the network and determination of power flow direction the PowerWorld^{TM} Simulator Version 12.0 licensed by Power World Corporation (*for University Educational Use*) is utilized. This simulator has the ability to perform numerous power system analyses like transient, contingency analysis, stability analysis and the determination of many other important parameters that are relevant for reliable operation of the system.

The rich interactive user interface of the simulator can be assessed at the run and edit modes. It is at the edit mode that the one-line diagram is constructed with symbolic representation of the network components of loads, generators, transformers and circuit breakers at their relative position to each other. Thereafter load flow analysis can be executed in the run mode using any iterative method of choice on its 2D interface.

The simulator can analyse multiple power sources on the network system, predicting the network bus voltages, transmission line losses and determine bus active and reactive powers. The one–line diagram of the network when solved, displays results with animated capacity.

The simulator by default indicates dynamically the direction of power flow by the movement of arrows on the transmission lines.

The run mode of power world simulator enables the simulation of the Nigeria 330kVIkeja West – Benin Transmission Line. The load flow for this model was implemented using Newton-Raphson interactive method. Bus voltages, phase angle, line losses, real and reactive power and line flows are obtained by inserting line impedance data, load and generation schedules into the dialogue box of power world simulator operated in the edit mode.

The one line diagram of the network is shown in the run mode with the resultant power flow indicated by the direction of dynamic arrows (*in run mode*) in Figure 3.2. From Figure 3.2 it can be seen that the resultant power flow is in the direction of bus B (Benin) to bus A (Ikeja West)

3.4 Matrix Laboratory Software

Matrix Laboratory Software, in short form known as Matlab, is a multi-paradigm numerical computing environment and proprietary programming language developed by MathWorks. Matlab allows matrix manipulations, plotting functions and data, implementations of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, C~~#~~, Java, Fortran and Python.

In this work, Matlab is used to perform fault analysis, wavelet decompositions, and train and test the neural networks. The results of the fault simulations in Matlab Simulink environment were sent to Matlab workspace from where wavelet transformation was performed on them using the wavelet graphic user interface (GUI) and the new results were still stored in Matlab

workspace. The results were now fed to the neural network GUI from the workspace for training and testing the neural network for pattern recognition.

3.5 AutoCAD

AutoCAD is a commercial computer-aided design (CAD) and drafting software application developed and marketed by Autodesk. It has a few vertical programs for disciple enhancements such as:

AutoCAD Advance steel,

AutoCAD Architecture,

AutoCAD Civil 3D,

AutoCAD Electrical,

AutoCAD escad,

AutoCAD Map 3D,

AutoCAD Mech, etc.

In this work, AutoCAD2007 was used to draw the sketch of the chosen neural networks as shown in chapter four.

3.6 Data

Data is a set of subjects with respect to qualitative or quantitative variables. Data and information or knowledge are often used interchangeably; however, data becomes information when it is viewed in context or in post-analysis.

All the data used in this work was extracted from the model of Figure 3.3 after performing simulations. The data was captured by the Matlab ‘‘To Workspace block’’ in discrete form.

For each type of fault simulated there is data associated with it as captured and this is known as raw data. The data was pre-processed by decomposing the data into detail coefficients and coarse approximations using wavelet transform analysis and these data set of data is known as the pre-processed data. These data were summed up as illustrated in Appendix B and subdivided into the training data, testing data and validation data. A sample of the pre-processed and summed-up data for single line to ground fault is shown in Appendix H.

These training data, testing data and validation data were now fed into the chosen neural networks for purpose of training and testing the neural networks such that they can master the patterns. For the purpose of training 60% of the entire data was used, for testing 20% and for validation the remaining 20% was used.

3.7 Outline of the Proposed Scheme

Each of the steps taken in this dissertation has been depicted in the flowchart shown in Figure 3.1. Firstly, the entire data is extracted and collected from the model of Figure 3.3 after simulation with the help of "To Workspace block". While the two scopes, labelled Vabc and Iabc display the results of the fault analysis graphically as shown in Figure 3.6 – 3.15. The "To Workspace block" was used to captured the results in discrete forms and sent to Matlab workspace for storage and further actions. From the Matlab workspace the data is decomposed and filtered into low frequency bands & high frequency bands using multi-resolution analysis (MRA) tool of wavelet transform. The results are graphically presented in Figure 3.17 – 3.26 and were still sent back to MATLAB workspace in discrete form for further action to be taken on the data.

From there, using the Matlab codes as shown in Appendix A and B, these results (decomposed signals) are summed up as W_a, W_b, and W_c which are the summation of the 5th level detail coefficients of both the decomposed voltage and current waveforms for the three phases or the summation of the 5th level approximations for both the decomposed current and voltage

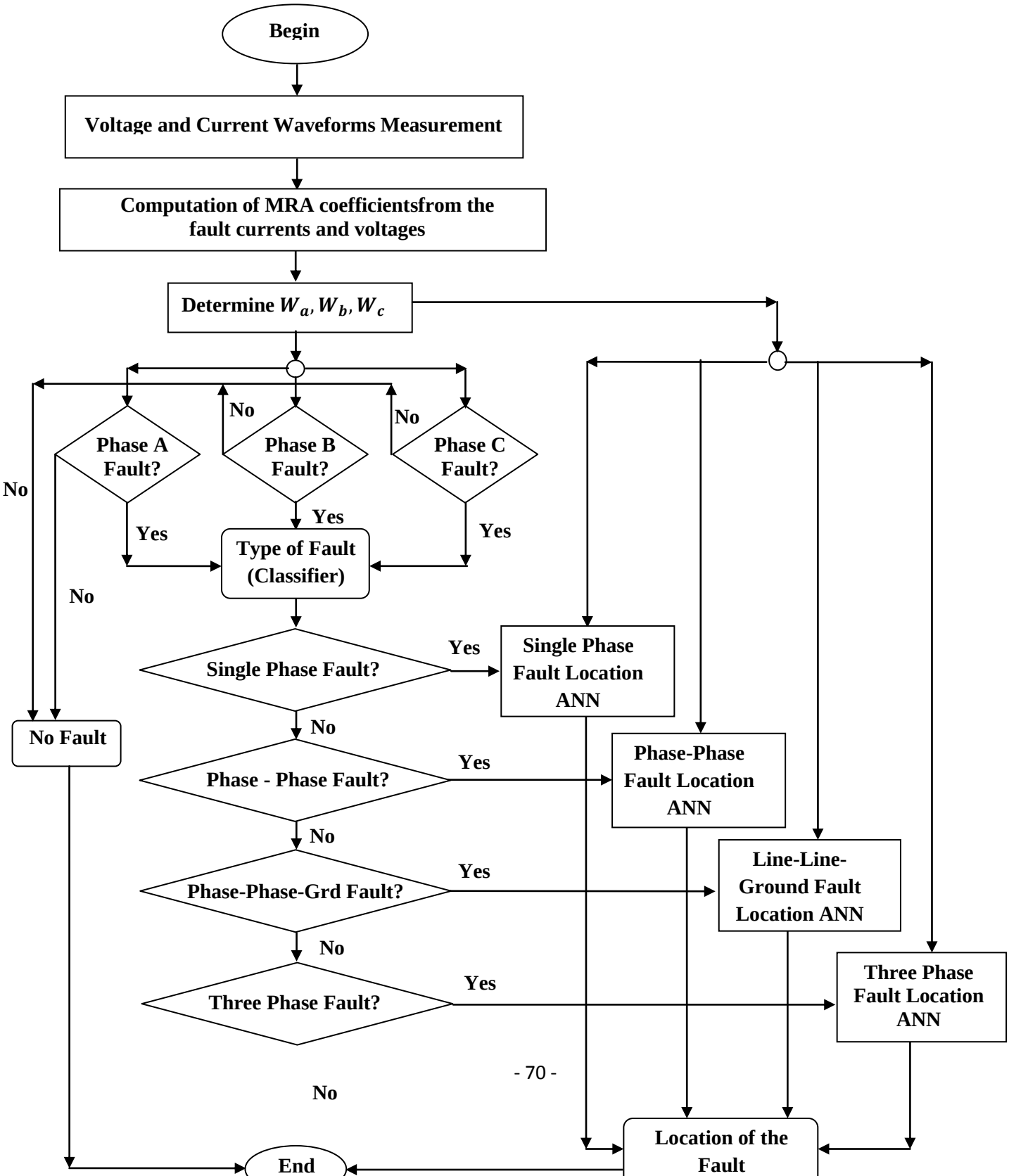


Figure 3.1: Flowchart outline of the proposed scheme.

waveforms for the three phases or the summation of both the 5th level details and approximations of both the decomposed current and voltage waveforms for the three phases for detection, classification and location of faults respectively. These W_a, W_b, and W_c (sample shown in Appendix H) are now inputted into the respective neural networks for training and testing the networks for mastering in other to recognize the patterns in the data for identification, classification and location purposes. These results were subdivided into three sets namely the training (60%), the testing (20%) and validation (20%) data sets before importing them into Matlab neural network graphic user interface (GUI) for the purpose of training and testing different chosen neural networks as discussed in subsequent sections.

Then, the excellent pattern recognition and classification abilities of neural networks have been cleverly utilized in this dissertation to address the issue of transmission line fault finding on the adopted Nigerian Transmission line.

The second step in the process is fault detection using neural networks. Once it is known that a fault has occurred on the transmission line, the next step is to classify the fault into the different categories based on the phases that are faulted. Then, the final step is to pin-point the position of the fault on the transmission line.

The goal of this dissertation as stated before is to propose an integrated method to perform each of these tasks using wavelet multi-resolution analysis tool and pattern recognition capability of artificial neural networks. A back-propagation based neural network has been used for the purpose of fault detection and another similar one for the purpose of fault classification. For each of the different kinds of faults, separate neural networks have been employed for the purpose of fault location.

3.8 Modelling the Power Transmission Line System

Figure 3.2: One-line diagram of the Nigerian 330kV Ikeja West - Benin transmission line system (the studied system).

The Nigerian 330kV Ikeja West - Benin transmission line system has been adopted and used to develop and implement the proposed strategy using ANNs. Figure 3.2shows a PowerWorld Simulator one-line diagram(in run mode) of the system that has been used throughout the research. PowerWorld was used to determine the direction of flow of energy in the model. The system consists of two buses of 330kV each located on each ends of the transmission line. The transmission line has been modelled using distributed parameters so that it more

accurately describes a very long transmission line. The fault record of the transmission line is as shown in Table 3.1 for information sake. These fault records actually have no impact or effect in the behaviour of the transmission line, therefore they played no role in modelling the transmission line. The line parameters (as obtained from Appendix G) determine the behaviour of the transmission line and have been used to characterize the transmission line.

Table 3.1: Number of Occurrence of Over-Current (O/C) Faults on Ikeja West – Benin Transmission Line (From TCN, 2019)

Fault Type		**Fault Nature Notation by TCN**	**Year**		
			2016	**2017**	**2018**
Phase to Ground	A-G	O/C RØ + E/F	12	8	14
Phase to Ground	B-G	O/C YØ + E/F	9	17	11
Phase to Ground	C-G	O/C BØ + E/F	13	10	9
Double Phase	AB	O/C R, YØ	16	12	10
Double Phase	BC	O/C Y, BØ	13	26	7
Double Phase	CA	O/C B, RØ	6	11	12
Double Phase to Ground	AB-G	O/C R, YØ + E/F	18	24	19
Double Phase to Ground	BC-G	O/C Y, BØ + E/F	28	15	17
Double Phase to Ground	CA-G	O/C B, RØ + E/F	14	21	16
Three Phase	ABC	O/C R, Y, BØ	7	4	6
Total No of Occurrence			**136**	**148**	**121**

Where; O/C stands for overcurrent; RØ, YØ and BØ stands for red, yellow and blue phases respectively; E/F stands for earth fault. Therefore, O/C R, YØ+E/F implies overcurrent on red and yellow phases plus earth fault.

This power system was remodelled and simulated using the SimPowerSystems toolbox in Simulink with the help of Matlab (R2016a). A snapshot of the model used for obtaining the trainingand testing data sets is shown in Figure 3.3. In Figure 3.3, Z1- Z0 are the source impedances of the generators on either side. A ‘‘three-phase fault block’’ was used to simulate faults at various positions on the transmission line.The three phase V-I measurement block is used to measure the voltage and current samples at the terminal A. The transmission line (1 and 2 together) is 280 km long and the three-phase fault block is used to simulate various types of faults at varying locations along the transmission line with different fault resistances.

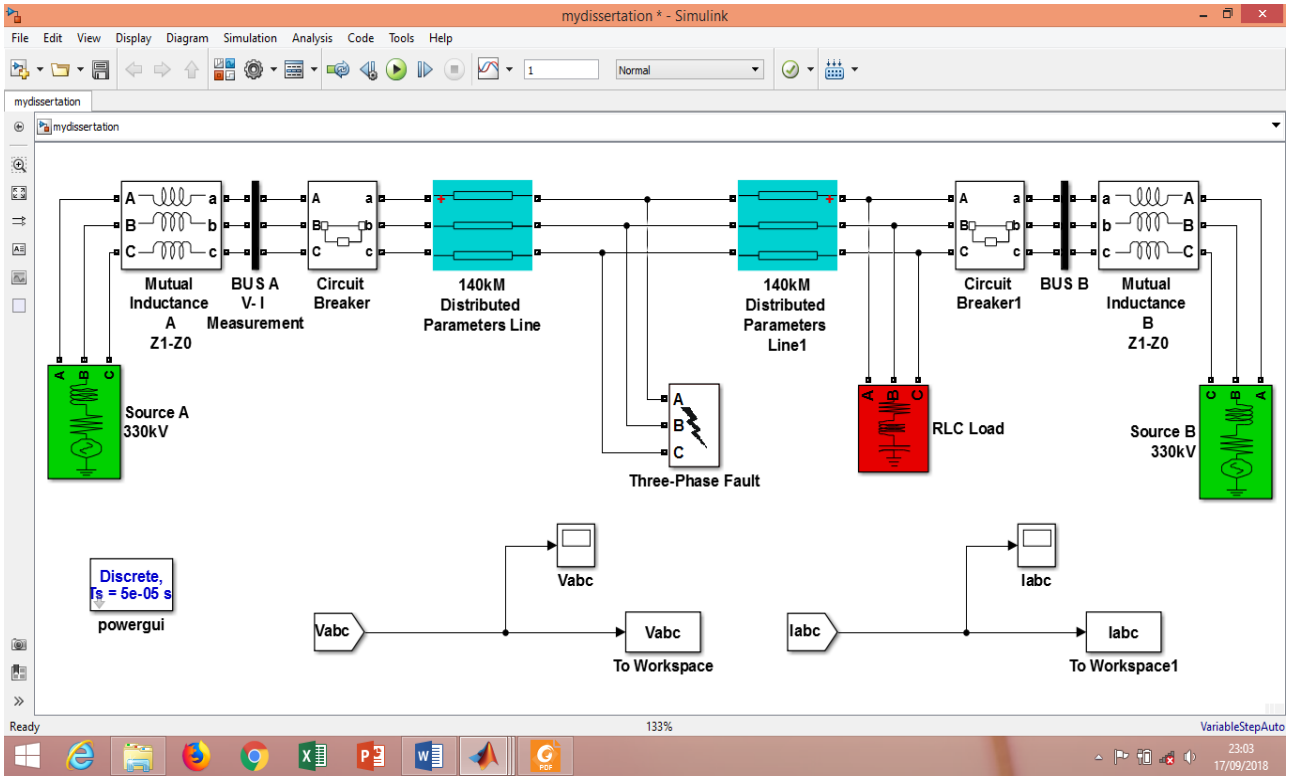


Figure 3.3: Snapshot of the studied model in SimPowerSystems.

The complete specifications of the components in theIkeja-west to Benin transmission line model in Figure 3.3 are as listed in sub-sections 3.7.1 - 3.7.6 and the values of the line parameters are lifted from Appendix G:

3.8.1 Three Phase Source

As can be seen, there are two three phase sources which are connected to the transmission line at both sides. Both power sources from left to right have generation capacity of 637MW and 568MW respectively.

Table 3.2: Parameters of the two Power Sources

S/N	Parameters	Egbin Power Plant(Source A at the Left)	Delta IV + Sapele power plants (Source B at the Right)
1	Installed Capacity (MW)	637	568
2	Phase to phase rms voltage (V)	330e3	330e3
3	Phase angle of phase A (degree)	0	-30
4	Frequency (Hz)	50	50
5	Internal connection	Yg	Yg
6	Three phase short circuit level at base voltage (VA)	250e6	1915e6
7	Base voltage (Vrms ph-ph)	330e3	330e3
8	X/R ratio	12.37/2.46	12.37/2.46

3.8.2 Circuit Breaker

One of the components in Figure 3.3 is the circuit breaker. In both circuit breakers, all the parameters are the same and they are listed below. These blocks are connected in series with the three-phase transmission lines that are to be opened when fault occurs. The breaker timing is defined directly from the dialog box or applied as an external logical signal. From the

'External control' box, the external control input will be adjusted. Parameters for both circuit breakers:

Transition times(s)	= [1/60]
Breaker resistance Ron (ohms)	= 0.001
Snubbers resistance Rp (ohms)	= $1e6$
Snubbers capacitance Cp (Farad)	= Positive infinity
Initial status of breakers	= closed

3.8.3 Three Phase Series RLC Load

The parameters of the Three phase series RLC load are as underlisted:

Configuration	= Y grounded
Nominal phase to phase voltage Vn (Vrms)	= $330e3$
Nominal frequency Fn (Hz)	= 50
Active power P (W)	= $1205e6$
Inductive reactive power QL (Positive Var)	= 240
Capacitive reactive power QC (0)	= 100

3.8.4 Distributed Parameters Line

One of the major components is the distributed parameters line, and the parameters of two of them are the same except in length, (it was broken into two parts of length in km), and the reason is that the line is about 280km long, during the modelling different places of fault occurrence are needed to be chosen, so that the total length of these two blocks should be 280km at each point in time. For example, if it is needed to show that the fault is happening at 90km from source A, then the length of the first line (block) will be set at 90km whilethe

length of the second line will be set at 190km. Rest of the length in different part will be same during the modelling, but if there is need to change the output of the signal, each part can be changed.

This block implements an N-phases distributed parameter line model. The RLC parameters are specified by [N×N] matrices. In modelling the three-phase symmetrical line, the complete [N×N] matrices were specified by simply entering the sequence parameters vectors: the positive and zero sequence parameters for the three-phase transposed line. This block has these parameters amounts:

Table 3.3: Parameters of the Distributed Parameter Lines (Appendix G)

S/N	Parameters	Amount
1	Number of phases [N]	3
2	Frequency used for RLC specification (Hz)	50
3	Resistance per unit length (ohms/kM) [N×N matrix]	[0.01010.0799]
4	Inductance per unit length (H/kM) [N×N matrix]	[1.637e−3 12.626e−3]
5	Capacitance per unit length (F/kM) [N×N matrix]	[1.162e−9 7.751e−9]
6	Line length (kM)	It is selective and the total length of the both sides should be equal to 280km because the transmission line is supposed to be 280km
7	Measurements	Phase to ground voltage is being measured

3.8.5 Three Phase V-I Measurement

Another component in of Figure 3.3 is the three-phase V-I measurement. From the Figure, they are two, one at right and one at left. These blocks were shrunk to form a bus-bar like structure. Their parameter is almost same, the difference is only in their output signal label of voltage and current. At the left, the label is I_{abc} for current and for voltage is V_{abc} and at the right, the label for current is Iabc1 and for voltage is Vabc1. The block can output the voltages and currents in per unit values or in volts and amperes.

Figure 3.4, shows the ‘‘V-I measurement’’ component in isolation. There are two ports which are labelled Vabc (three phase voltage) and Iabc (three phase current) and they are to be connected to the ‘‘scope blocks’’and ‘‘To Workspace blocks’’ for their output to be displayed graphically and be sent to Matlab environment in discrete form respectively.

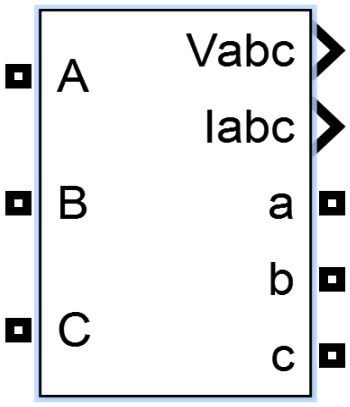


Figure 3.4: Three Phase V-I Measurement

3.8.6 Three Phase Fault:

Another component in of Figure 3.3 is the three phase fault. With this block, different types of faults and also resistance for ground line can be chosen. The block has different phases (phase A, phase B, phase C) and also ground fault. Combination of any of them can be

chosentogether with or without ground. In other words, this block was used to program a fault (short-circuit) between any phase and the ground. The fault timing was defined directly from the dialog box (though an external logical signal can be applied). If the 'External control' box was checked, the external control input would appear. Parameters:

Fault resistance Ron (ohms)	=	15
Transition status [1, 0, 1 ...)	=	[1 0]
Transition times (s)	=	[1/60 5/60]
Snubbers resistance Rp (ohms)	=	$1e6$
Sunbbers capacitance Cp (Farad)	=	Positive infinity
Measurement	=	none

Figure 3.5, shows the connection of the ‘‘scope blocks’’ and‘‘To Workspace blocks’’ components. These two components are used for results collections in pictorial and discrete forms respectively.

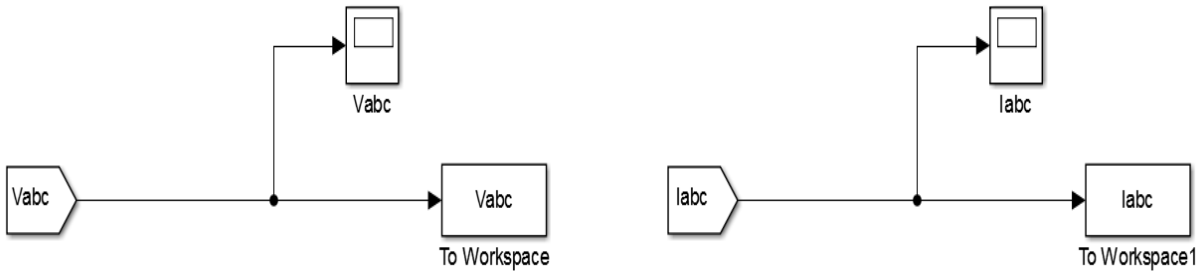


Figure 3.5: Voltage and current blocks to scope blocks and ‘‘To workspace blocks’’

3.9 Acquisition of Data and Pre-Processing

A reduction in the size of the neural network improves the performance of the same and this can be achieved by performing feature extraction using wavelet transformation of the input signal. By doing this, all of the important and relevant information present in the waveforms of the voltage and current signals can be used effectively. The fault currents and voltages have been generated from the model of Figure 3.3 after simulation using Matlab as shown in Figures 3.6 – 3.15for fault condition of L-G, L-L, L-L-G and L-L-L. Moreover, the sampling time taken for the analysis is 100us, which relates to a sampling frequency of 10 kHz.

Figure 3.6 is the waveform of the current of the single line to ground fault on phase A measured at 140km away from source A during occurrence of fault on the line. The red, blue and green curves represent fault conditions of phases A, B and C respectively. From the graph it can be seen that the fault occurred at 50ms after which the magnitude of fault current on phase A rose reasonably while the other two phases almost remained stable which is clear from the Figure 3.6.

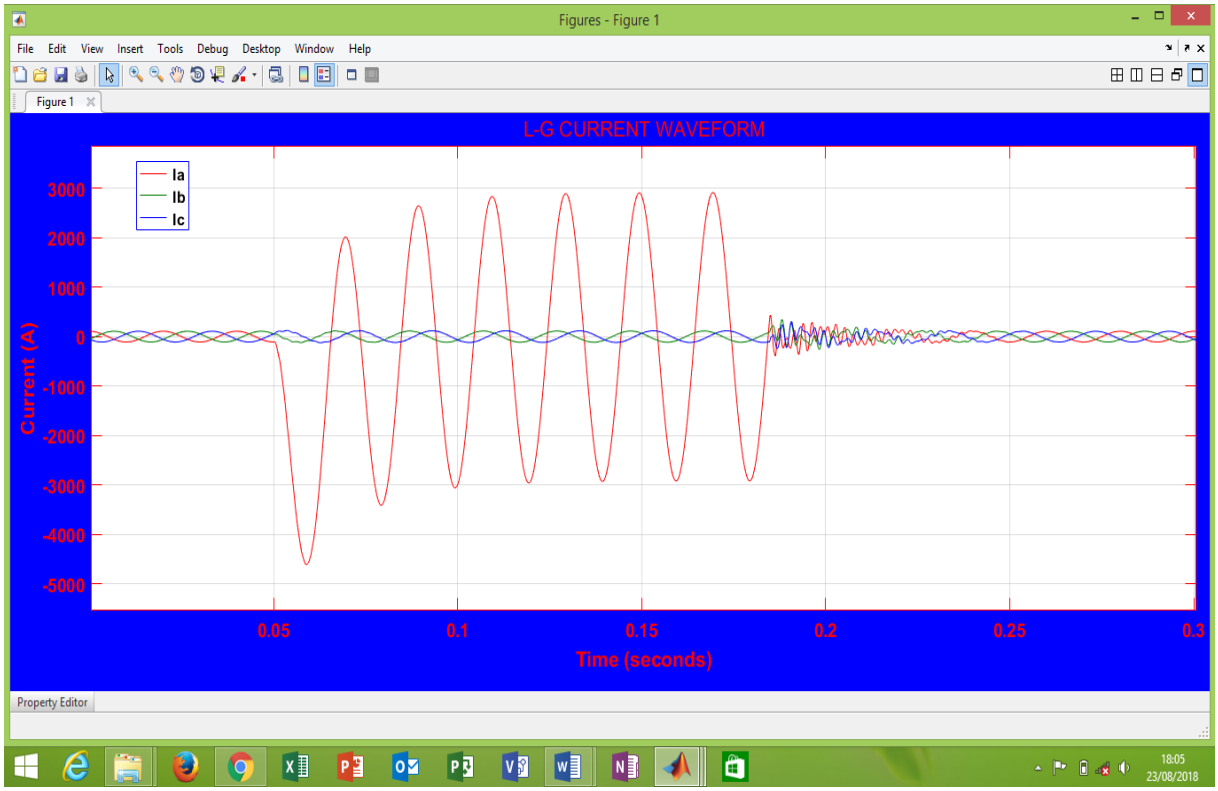


Fig. 3.6: Current waveform of LG Fault on phase A at a distance of 140kM from the source

Figure 3.7 is the corresponding waveform of the voltage of the single line to ground fault on phase A measured at 140km away from source A during occurrence of a transient fault on the line. As aforementioned, the red, blue and green curves represent fault conditions of phases A, B and C respectively. The fault has caused a voltage drop on phase A when the fault occurred after 50ms. The other two phases almost remained stable.

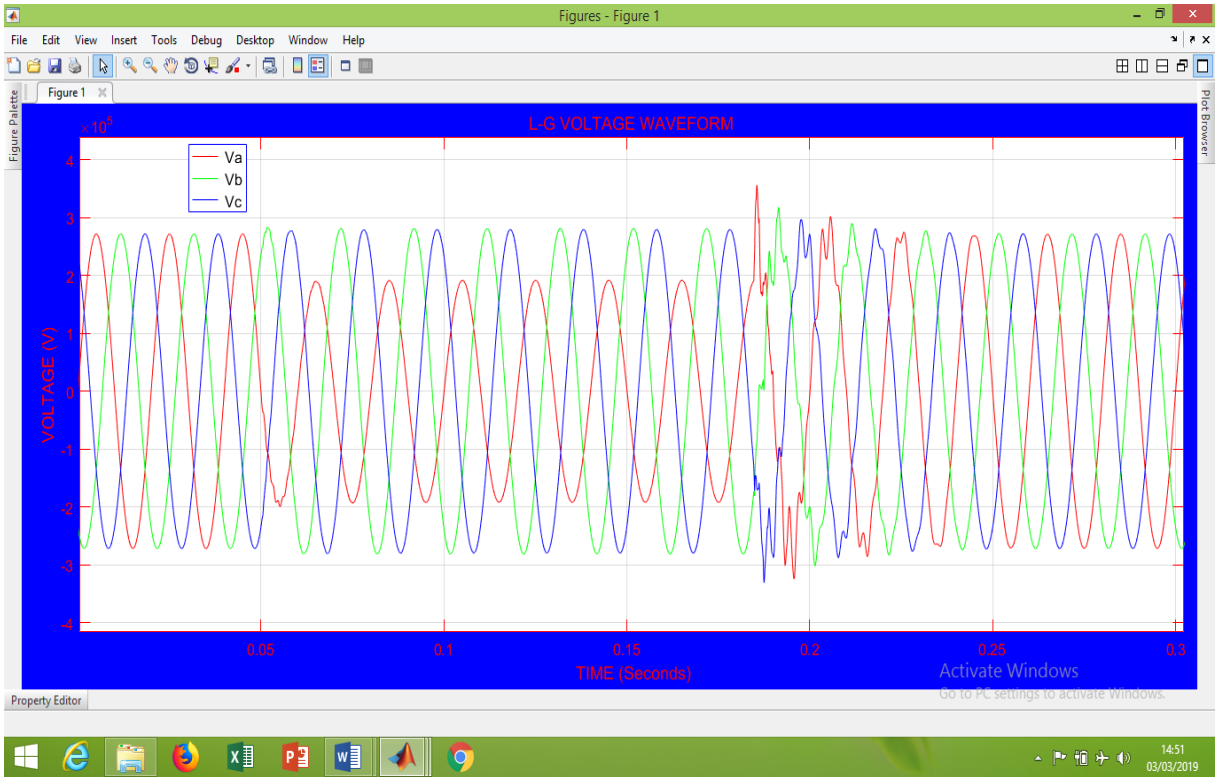


Fig. 3.7: Voltage waveform of LG Fault on phase A at a distance of 140kM from the source

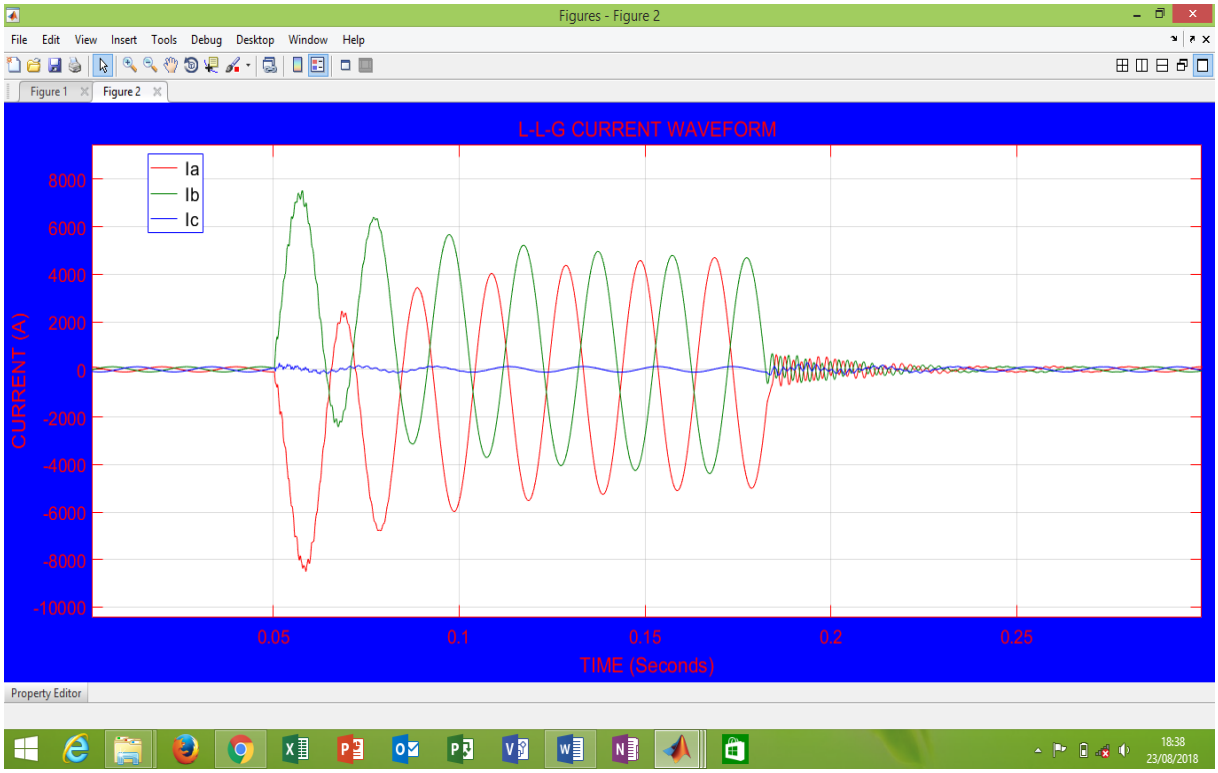


Fig. 3.8: Current waveform of LL-G Fault on phase A & B at a distance of 140km from the source

In Figure 3.8, the red, blue and green curves represent fault conditions of phases A, B and C respectively. Figure 3.8 is the waveform of the current of the double line to ground fault on phase A and B measured at 140km away from source A during occurrence of a transient fault on the line. From the graph it can be seen that the fault occurred at 50ms after which the magnitude of fault current on phases A and B rose reasonably high (enough to cause flow disruption) while the other phase almost remained stable which is clear from the Figure 3.8.

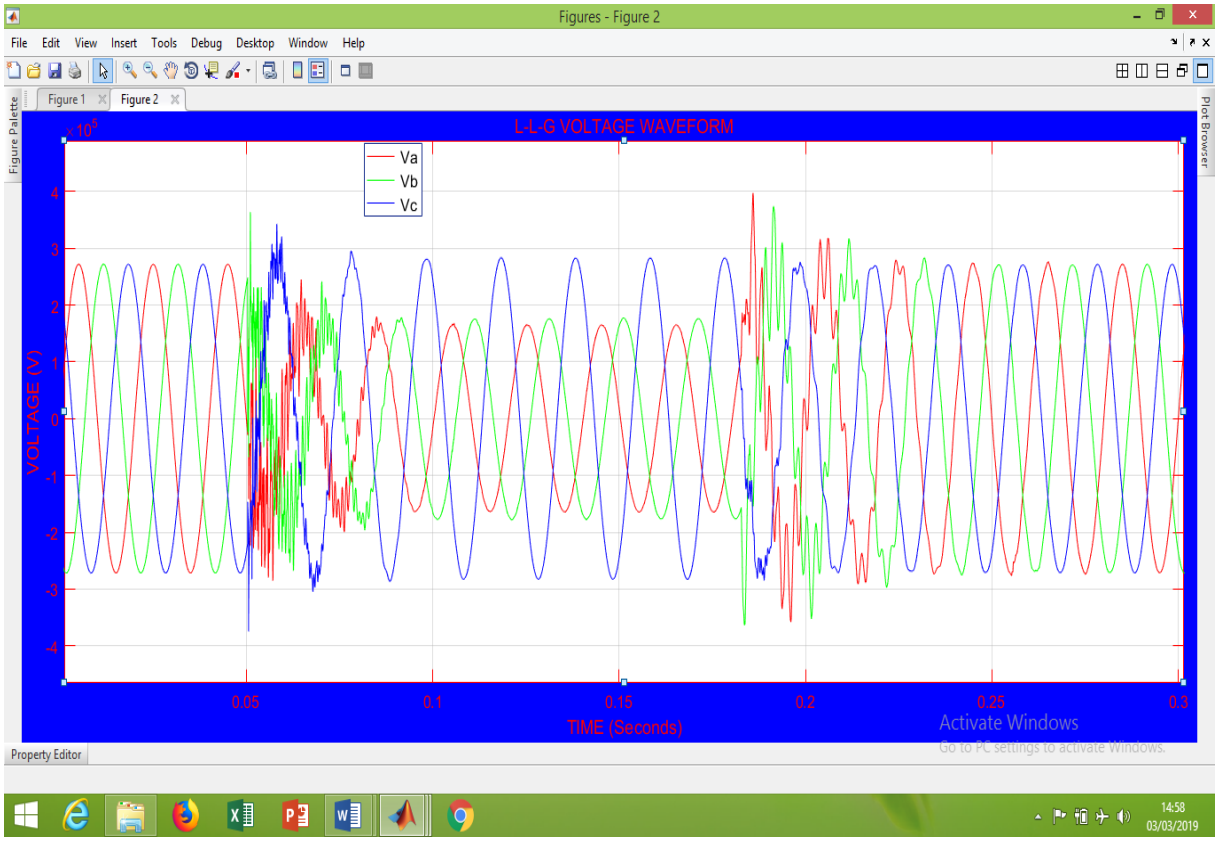


Fig. 3.9: Voltage waveform of LL-G Fault on phase A & B at a distance of 140km from the source

Figure 3.9 is the corresponding waveform of the voltage of the double line to ground fault on phase A measured at 140km away from source A during occurrence of a transient fault on the line. The red, blue and green curves represent fault conditions of phases A, B and C respectively. The fault has caused voltage drops on phases A and B after the fault occurred at 50ms. Phase C almost remained stable.

Figure 3.10 shows the current waveform of the double line (without ground) fault on phase A and B measured at 140km away from source A during occurrence of a transient fault on the line. The red, blue and green curves represent fault conditions of phases A, B and C respectively. From the graph it can be seen that the fault occurred at 50ms after which the

magnitude of fault current on phases A and B increased leading to instability in the system. The phase C almost remained stable as shown in the Figure 3.10.

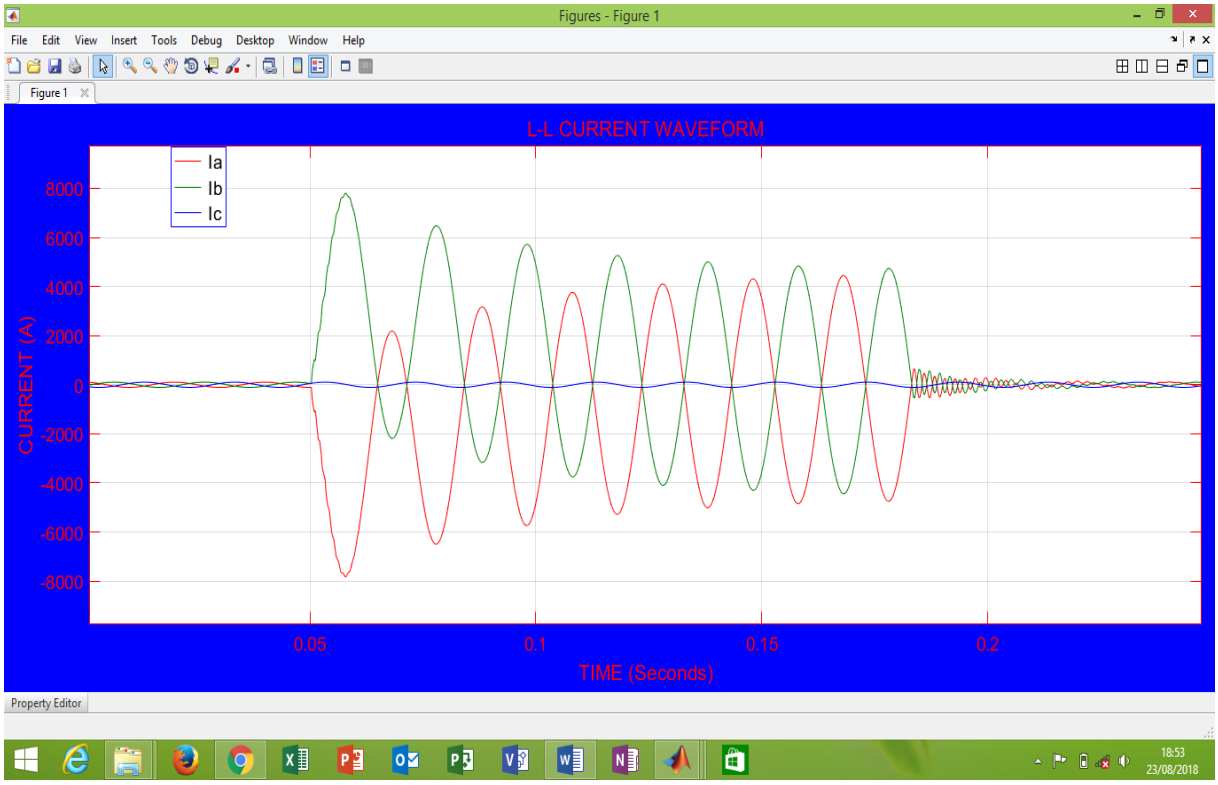


Fig. 3.10: Current waveform of LL Fault on phases A & B at a distance of 140km from the source

Figure 3.11 shows the corresponding voltage waveform of the double line fault on phase A and B measured at 140km away from source A during occurrence of a transient fault on the line. As aforementioned, the red, blue and green curves represent fault conditions of phases A, B and C respectively. The fault has caused significant voltage drops on phases A and Bwhen the fault occurred at 50ms. Phase C almost remained stable.

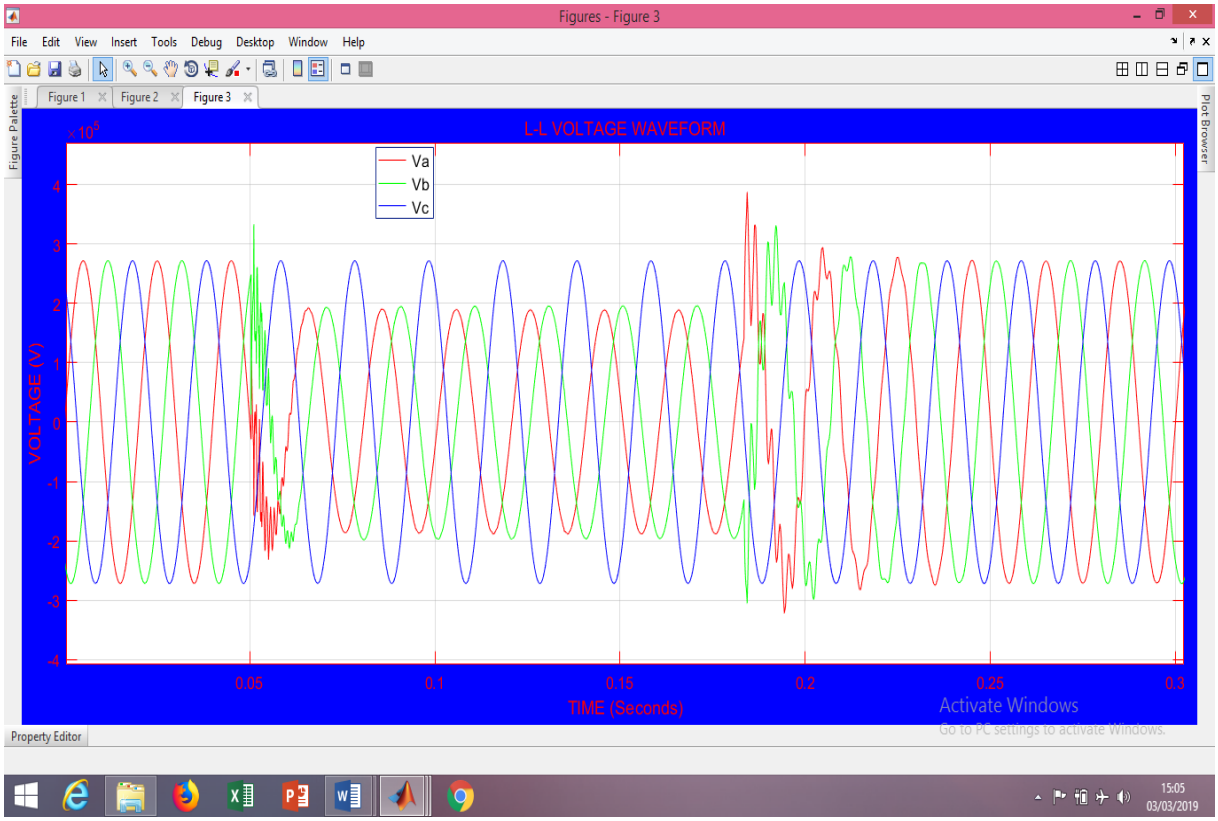


Fig. 3.11: Voltage waveform of LL Fault on phase A & B at a distance of 140km from the source

In Figure 3.12, the red, blue and green curves represent fault conditions of phases A, B and C respectively. Figure 3.12 depicts the current waveforms of three phase fault (with or without ground) on phase A, B and C measured at 140km away from source A during occurrence of a fault on the line. From the graph it can be seen that the fault occurred at 50ms after which the magnitude of fault current on the three phases rose reasonably high, enough to cause flow disruption.

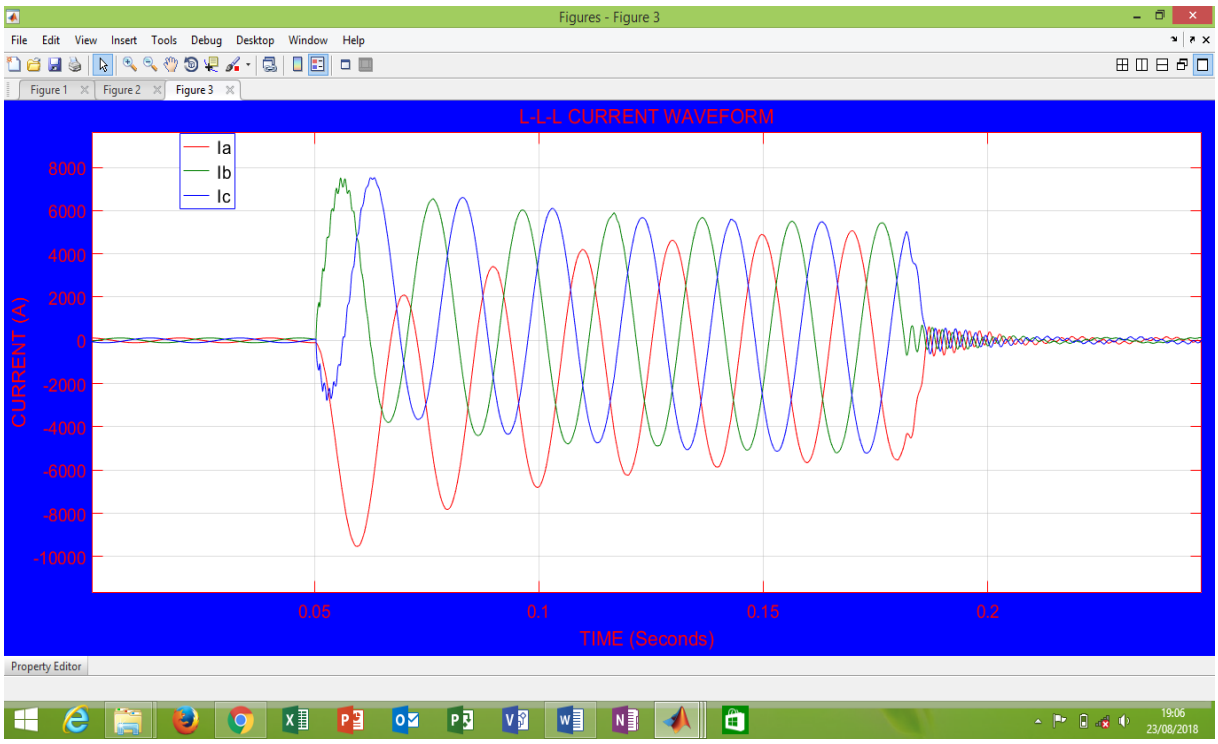


Fig. 3.12: Current waveform of LLL Fault on all the phases at a distance of 140km from the source

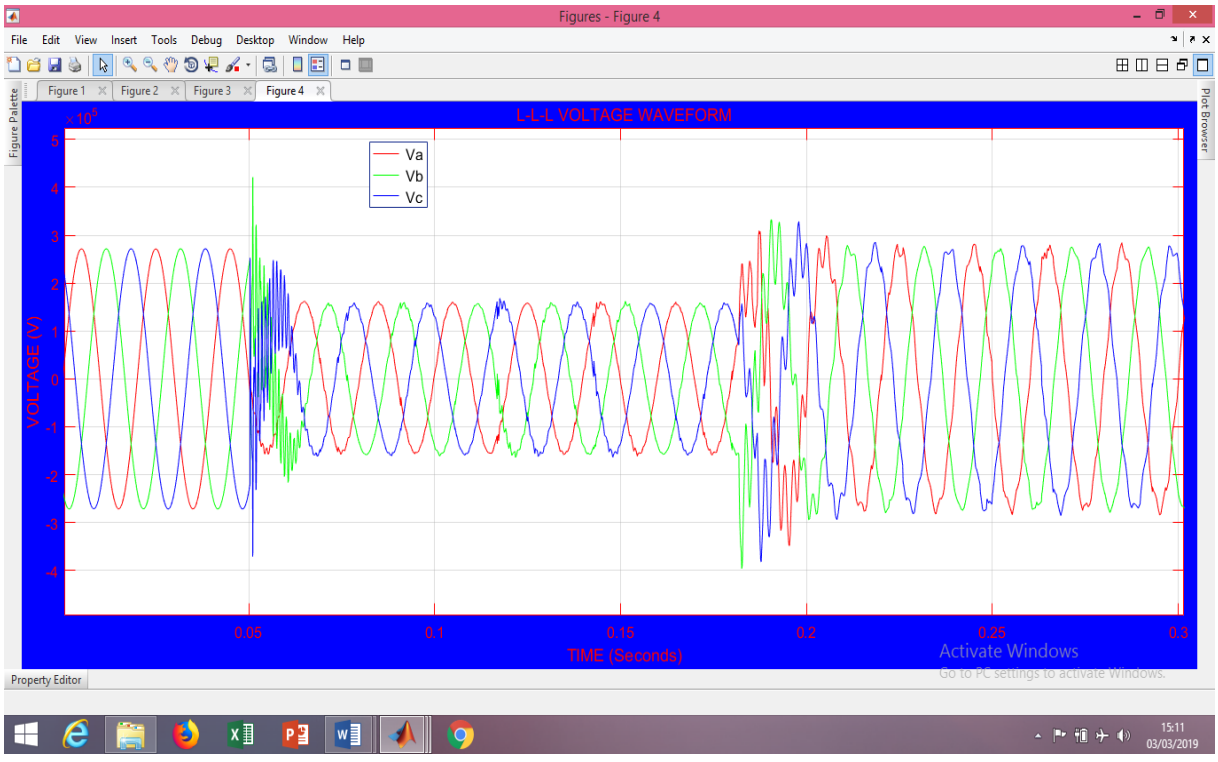


Fig. 3.13: Voltage waveform of LLL Fault on all the phases at a distance of 140km from the source

Figure 3.13 depicts the corresponding voltage waveforms of three phase fault (with or without ground) measured at 140km away from source A during occurrence of a transient fault on the line. As said above, the red, blue and green curves represent fault conditions of phases A, B and C respectively. The fault has caused simultaneousvoltage drops on phases A, B and C after the fault occurred at 50ms.

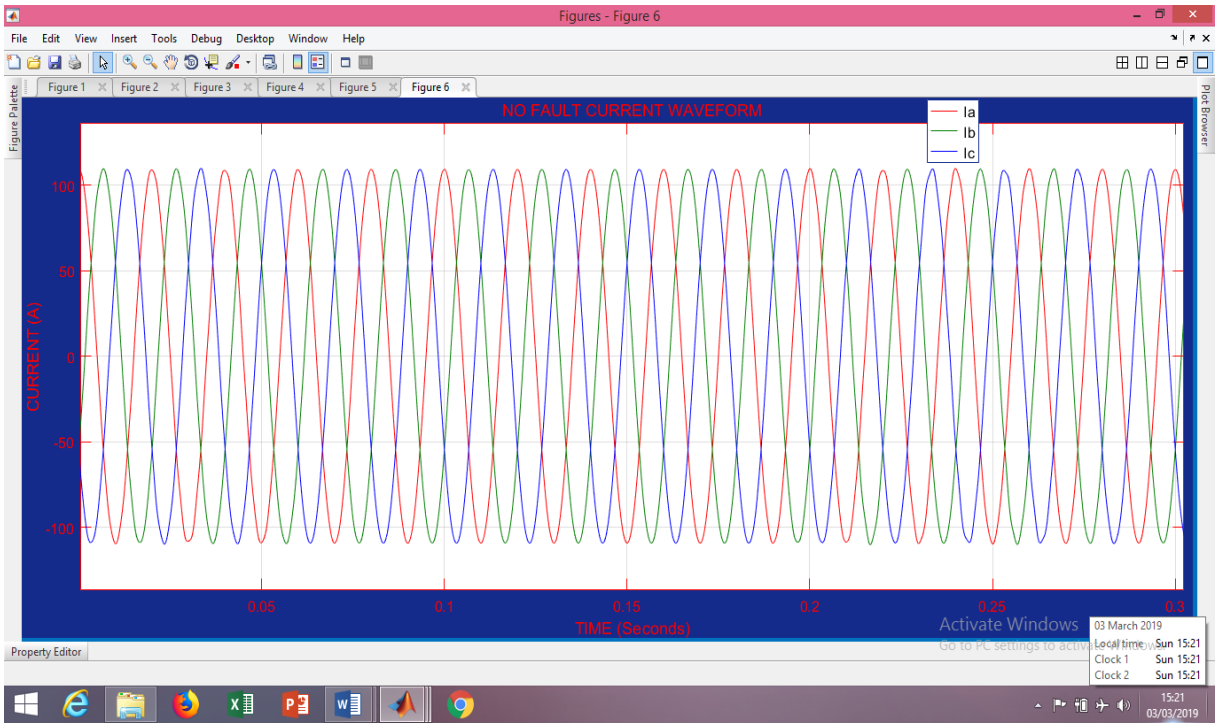


Fig. 3.14: Current waveform of No Fault condition on all the phases at a distance of 140km from the source

In Figure 3.14, the red, green and blue curves represent fault conditions of phases A, B and C respectively. Figure 3.14 depicts the current waveforms of No fault on phase A, B and C captured at 140km away from source A. From the graph it can be seen that the magnitude of currents on the three phases remained unchanged.

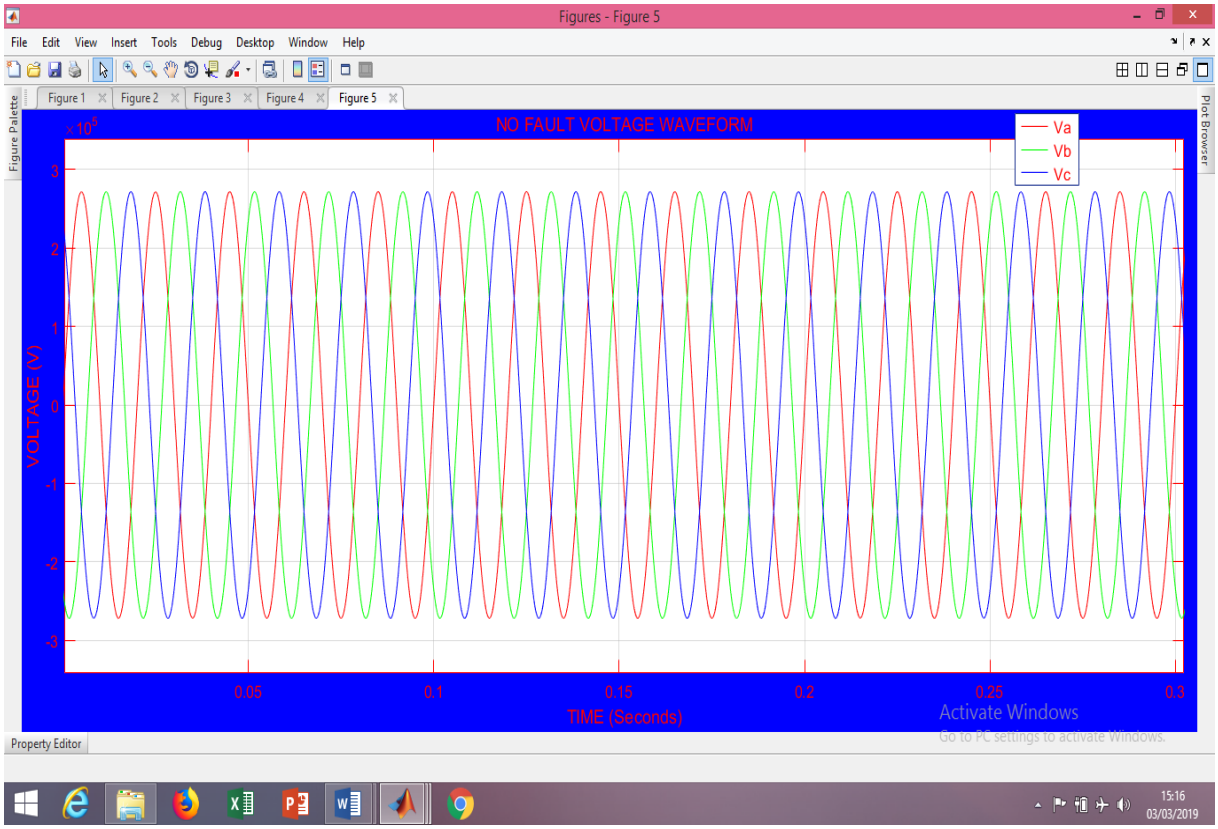


Fig. 3.15: Voltage waveform of No Fault condition on all the phases at a distance of 140km from the source

Figure 3.15 depicts the corresponding voltage waveforms of No fault condition measured at 140km away from source A at no fault condition. As aforementioned, the red, green and blue curves represent fault conditions of phases A, B and C respectively. The voltage of the three phases (phases A, B and C) has remained unchanged.

These results for each simulation are captured and stored in Matlab with the help of the ‘‘To Workspace blocks’’ tool as shown in Figure 3.5. Then to perform wavelet decomposition, these captured results or data are imported into the Matlab wavelet toolbox where they are decomposed into detail coefficients and coarse approximations using the multi-resolution analysis tool as presented.

3.10 Wavelet Transform Analysis of Fault Signals

From the point of linear algebra, a signal (fault signal) can be decomposed into linear combination of the basis if the signal is in the space spanned by the basis. Given a one-dimensionalfault signal$f(x)$, it can be expressed as,

$$f(x) = \sum_k \alpha_k \emptyset_k\,(x) \tag{3.1}$$

Where$k$ is an integer index of the finite or infinite sum, the α_k are expansion coefficients, x is the continuous variable in space and the $\emptyset_k(x)$are expansion functions, or the basis. To realize the set of function $\{\emptyset_k\}$, a scaled and shifted function $\emptyset_{r,s}(x)$ of some basic function (k) are used as stated below:

$$\emptyset_{r,s}(x) = 2^{r}/_{2}\emptyset(2^r x - s) \tag{3.2}$$

$$r, s \in Z$$

Where $r\ and\ s$ are the scaling and the shift parameters respectively. And thus, the class of functions$\{\emptyset_{r,s}(x)\}$ are called the Scaling functions. If r is fixed to some value as $r = r_0$, which implies that the only variable is s, then the $\emptyset_{r_0\,,s}(x)$ can be used to cover only a limited space of the entire square integrable space, $L^2(R)$. The class of functions which are used to cover the difference sub-space are called the Wavelet functions and are mathematically represented as

$$\varphi_{r,s}(x) = 2^{r}/_{2}\varphi(2^r x - s) \tag{3.3}$$

The relationship between the scaling and the wavelet functions is give below

$$\varphi(x) = \sum_k h_\varphi(n)\sqrt{2}\emptyset(2x - n) \quad (3.4)$$

$$n \in Z$$

$h_\varphi(n)$is the coefficient associated with the shift parameter (n)

Therefore, given a set of scaling $\{\emptyset_{r_0,s}(x)\}$ and wavelet $\varphi_{r,s}(x)$ functions, the continuous function $f(x)$ can be approximated as below:

$$f(x) = \sum_S \alpha_{r_0,s}\emptyset_{r_0,s}(x) + \sum_{r=r_0}^{\infty}\sum_s b_{r,s}\varphi_{r,s}(x) \quad (3.5)$$

$$r \geq r_0$$

Where the coefficients $\alpha_{r_0,s}$ and $b_{r,s}$ can be realized as:

$$\alpha_{r_0,s} = \int f(x)\emptyset_{r_0,s}x\,dx$$

$$b_{r,s} = \int f(x)\varphi_{r,s}x\,dx$$

The definition of continuous wavelet transform (CWT) for a given signal x(t) with respect to a mother wavelet (t) is:

$$CWT(a,b) = \frac{1}{\sqrt{2}}\int_\infty^\infty x(t)\varphi\left(\frac{t-b}{a}\right)dt \quad (3.6)$$

where a is the scale factor and b is the translation factor.

For CWT, t, and b are all continuous. Unlike Fourier transform, the wavelet transform requires selection of a mother wavelet for different applications. One of the most popular mother wavelets found for power system transient analysis in the literature is Daubechies's wavelet family. In this work, the db5 wavelet is selected as the mother wavelet for detecting

the short duration, fast decaying fault generated transient signals. Db5 is suitable for power analysis especially fault analysis unlike other mother wavelets like Haar, Morlet, Simlet etc which are good for analysing images and pictures (Hasabe and Vaidya, 2014).

3.10.1 Discrete Wavelet Transform of the Fault Signals

The application of wavelet transform in engineering areas usually requires discrete wavelet transform (DWT), which implies the discrete form of t, a, b in section (3.10).Here, the discrete form of the values r, k, a, x, and $f(x)$ will now be j, s, w, n and $s(x)$ respectively. The representation of DWT can be written as:

$$W_{\emptyset}(j_0, k) = \frac{1}{\sqrt{M}} \sum_n s(n) \emptyset_{j_0,k}(n) \quad (3.7)$$

$$j \geq j_0$$

$$W_{\varphi}(j, k) = \frac{1}{\sqrt{M}} \sum_n s(n) \varphi_{j,k}(n) \quad (3.8)$$

Where, $s(n)$ is the discrete fault signal to be decomposed, $1/\sqrt{M}$ is a normalizing factor. Equations 3.7 and 3.8 are the scaling function and the wavelet transform respectively. Normally, if j_0 is fixed at zero (0), M is selected to be 2^J; $M = 2^J$ and summation is performed up to j= 0, 1, 2, …, J-1, then the discrete form of the wavelet function $\varphi_{r,s}$ is now;

$$\varphi_{j,k}(n) = 2^{\frac{j}{2}} \varphi(2^j n - k) \quad (3.9)$$

Putting $\varphi_{j,k}(n)$ into equation 3.8 gives

$$W_{\varphi}(j, k) = \frac{1}{\sqrt{M}} \sum_n s(n) * 2^{\frac{j}{2}} \varphi(2^j n - k) \quad (3.10)$$

But the relationship between the scaling function and the discrete wavelet function is given by,

$$\varphi(n) = \sum_p h_\varphi(p)\sqrt{2} * \emptyset(2n - p) \quad (3.11)$$

Where, p is the new shift parameter, n is the scaling parameter and h_φ is a high-pass filter.

If equation 3.11 is scaled up by a factor to the power j and shifted by k unit i.e. to get $\varphi(2^j n - k)$,equation 3.11 can be written as

$$\varphi\left(2^j n - k\right) = \sum_p h_\varphi(p)\sqrt{2} * \emptyset(2(2^j n - k) - p) \quad (3.12)$$

Let $p = m - 2k$, then $m = p + 2k$

Then,

$$\varphi\left(2^j n - k\right) = \sum_p h_\varphi(m - 2k)\sqrt{2} * \emptyset(2^{j+1} n - m) \quad (3.13)$$

Putting(3.13) into(3.10) gives;

$$W_\varphi(j,k) = \frac{1}{\sqrt{M}} \sum_n s(n) * 2^{\frac{j}{2}} * \sum_p h_\varphi(m - 2k)\sqrt{2} * \emptyset(2^{j+1} n - m)$$

Interchanging the summation order gives;

$$W_\varphi(j,k) = \sum_m h_\varphi(m - 2k)\left[\frac{1}{\sqrt{M}} \sum_n s(n) * 2^{\frac{j+1}{2}} * \emptyset(2^{j+1} n - m)\right] \quad (3.14)$$

Relatingequation 3.10 with equation 3.14;

$$W_\emptyset(j+1,m) = \frac{1}{\sqrt{M}} \sum_n s(n) * 2^{\frac{j+1}{2}} * \emptyset(2^{j+1} n - m)$$

Therefore,

$$W_{\varphi}(j,k) = \sum_m h_{\varphi}(m-2k)W_{\emptyset}(j+1,m) \quad (3.15)$$

In very similar way, can be obtained;

$$W_{\emptyset}(j,k) = \sum_m h_{\emptyset}(m-2k)W_{\emptyset}(j+1,m) \quad (3.16)$$

The implication ofequations3.15 and 3.16 is that a scaling function $W_{\emptyset}(j+1,m)$ is being convolved with a highpass analysis filter $h_{\varphi}(m-2k)$ to yield the detail information and low pass analysis filter $h_{\emptyset}(m-2k)$ to yield the coarse approximations of the fault signals respectively. This decomposition of the fault signal by successive high pass and low pass filtering of the time domain signal is called multi-resolution analysis (MRA). In a block diagram, up to fifth level decomposition is represented as shown in Figure 3.16.

The discrete wavelet transform (DWT) of the fault signal represents a 1-D signal $s(n)$ in terms of shifted versions of a low-pass scaling function and shifted and dilated versions of a prototype band pass wavelet function.

Having converted the continuous fault signals into discrete signals, the sampled signals $W_{\emptyset}(j+1,m)$ are passed through a high pass filter $h_{\varphi}(m-2k)$and a low pass filter $h_{\emptyset}(m-2k)$. A process called wavelet decomposition using multi-resolution analysis. Then the outputs from both filters are decimated by 2 to obtain the detail coefficients and the approximation coefficients at level 1 (D1 and A1). The approximation coefficients are then sent to the second stage to repeat the procedure. Finally, the fault signal is decomposed at the expected level.

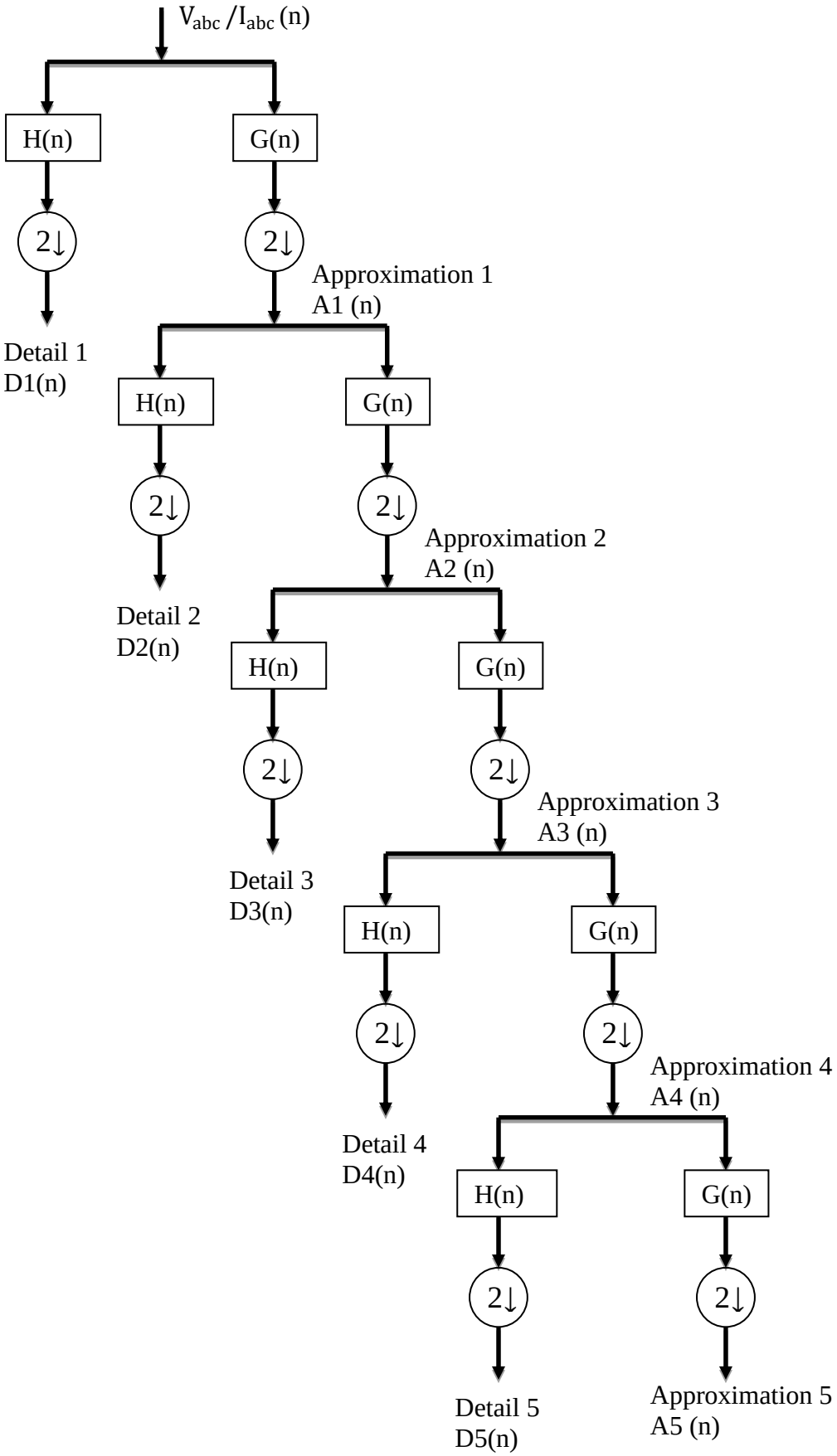

V_{abc} /I_{abc} (n)
H(n)
G(n)
2↓
2↓
Approximation 1
A1 (n)
Detail 1
D1(n)
H(n)
G(n)
2↓
2↓
Approximation 2
A2 (n)
Detail 2
D2(n)
H(n)
G(n)
2↓
2↓
Approximation 3
A3 (n)
Detail 3
D3(n)
H(n)
G(n)
2↓
2↓
Approximation 4
A4 (n)
Detail 4
D4(n)
H(n)
G(n)
2↓
2↓
Detail 5
D5(n)
Approximation 5
A5 (n)

Fig. 3.16: FifthLevel Multi-Resolution Analysis (MRA) representation of I_{abc} and V_{abc}

3.11 Decomposition of the Signals (I_{abc} and V_{abc}) using Multi-resolution Tool of Wavelet Transform in Matlab

In order to reduce the computational burden, the sampling frequency should not be too high but it should be high enough to capture all the information concerning the faults. The generated current and voltage signal (see Figure 3.6 to 3.15) for each case is analyzed using wavelet transform. A sampling frequency of 10 kHz is selected. Daubechies wavelet Db5 is used as mother wavelet.This is becauseDb5 has good performance results for power system fault analysis(Hasabe and Vaidya, 2014). Detail coefficients of fault current and voltage signals in 5th level (d5), give the frequency components corresponding to second and third harmonics. On this basis, summation of 5th level detail coefficients of the three phase currents I_a, I_b and $I_c$and voltage V_a, V_b and $V_c$are being used for the purpose of detection of faults in the transmission line.

The total number of wavelet levels considered is 5. Hence a fifth level wavelet output corresponds to a frequency band of 5-10 kHz. Down sampling by two was done at succeeding levels leading to a third level output corresponding to a frequency band of 2.5-5 kHz, i.e. it consists of 2nd and 3rd harmonic components and are predominant in the situation of transmission line faults. The wavelet toolbox in Matlab has been used for DWT operation. Different decomposition levels such as A5 (Approximation at level five); detail levels one, two, three, represented as D1, D2, D3, D4, D5 respectively were extracted using wavelet toolbox. The Appendix B gives summations of wavelet coefficients of fifth values for voltage and current in phases A, B and C respectively for single line to ground (L-G), double line to ground (L-LG), double line (L-L), and three phase symmetrical (L-L-L) faults for different fault inception angles and fault locations.

Figures 3.17 - 3.26 show the wavelet decomposition of the Simulink extracted fault waveforms of Figures 3.6- 3.15 respectively via multi-resolution analysis. The sampling frequency is 10kHz, the signal information captured by D1 is between 2.5kHz and 5kHz of the frequency band. D2 captures the information between 0.125 kHz and 0.25 kHz. D3 captures the information between 0.0625kHz and 0.125kHz, and A3 retains the rest of the information of original signal between 0 and 0.0625kHz.By such means, useful information from the original signal can easily be extracted into different frequency bands and at the same time the information is matched to the related time period.

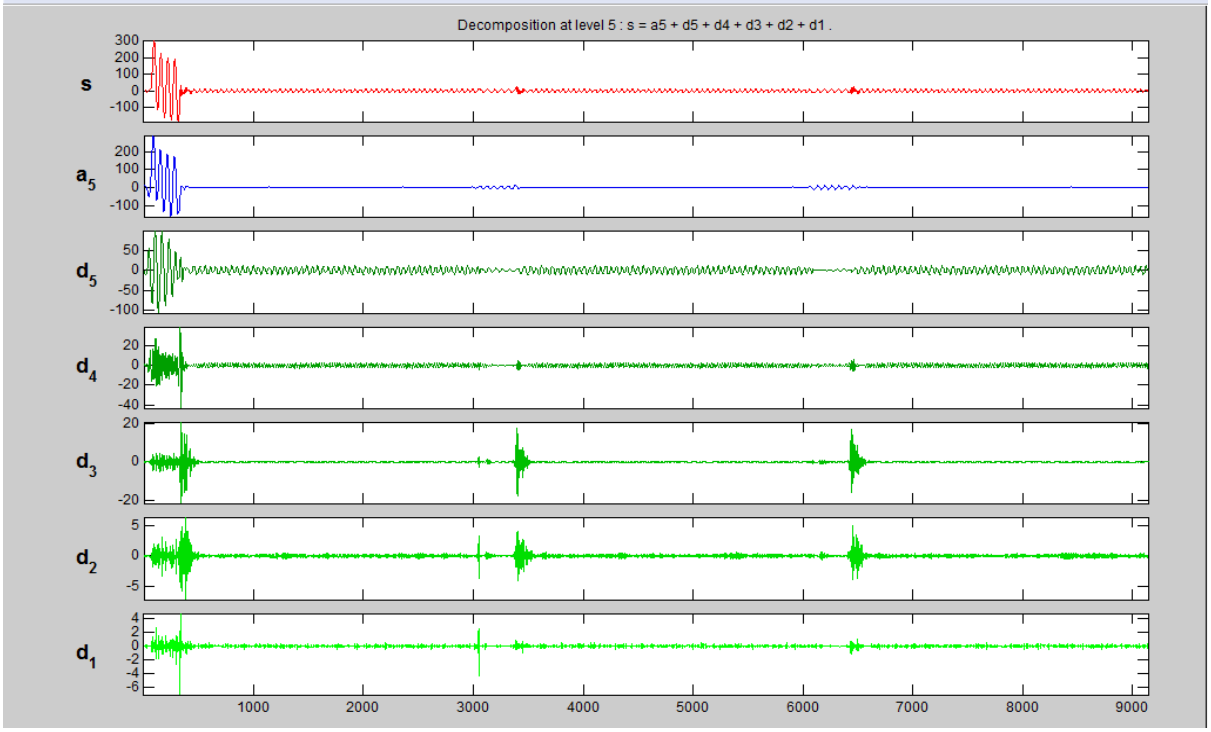


Figure 3.17: Decomposed signal of Current waveform of LG Fault on phase A at a distance of 140km from the source

Figure 3.17 depicts the snapshot of the fifth level decomposition of the sampled current waveforms of LG fault on phase A (of Figure 3.6). Db5 wavelet has been used to make a 5-

level decomposition. d_1, d_2, d_3, d_4, and d_5 represent the detail coefficients for level 1, 2, 3, 4, and 5 respectively while a_5 is the fifth level approximation coefficient. As clearly shown, the three phases are concatenated together of matric data of 9141x3 in all. The first set of data of 3047x1 in a column is for phase A, the second set of data of 3047x1 is for phase B and the third set of data of 3047x1 is for phase C. The information of original signal is clearly represented at each frequency band.

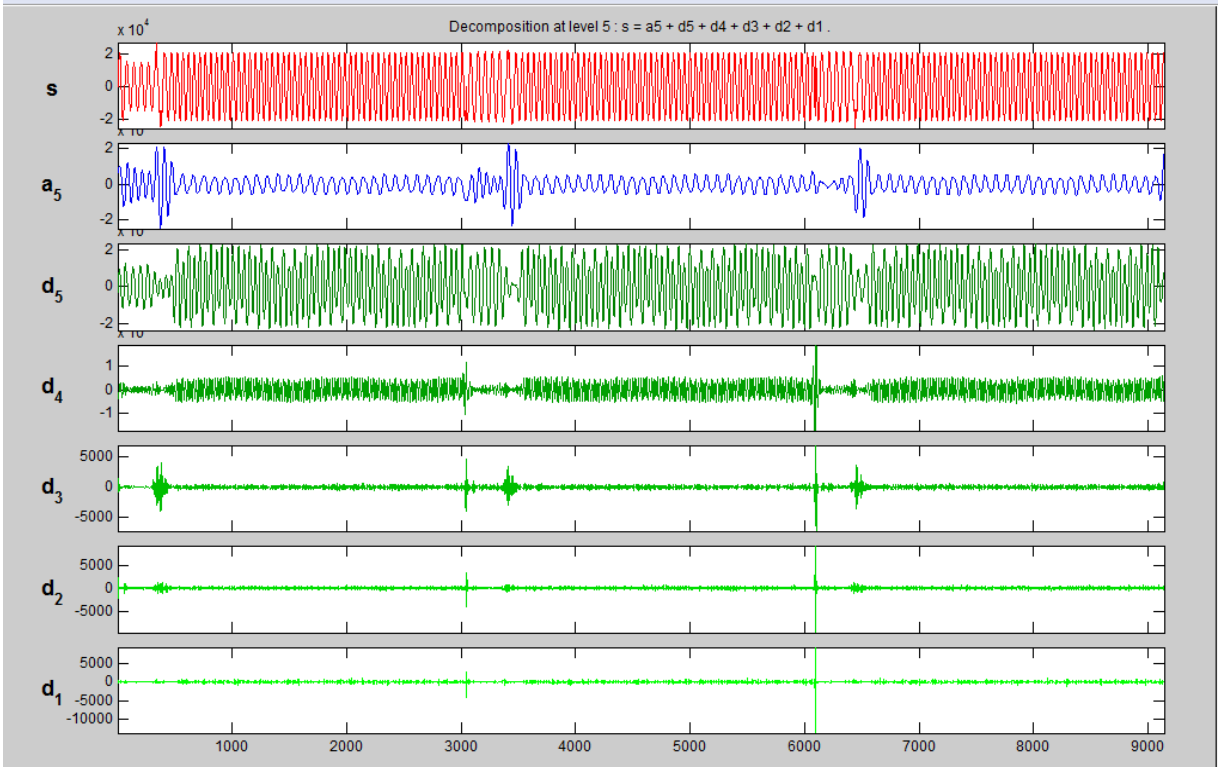


Figure 3.18: Decomposed signal of voltage waveform of LG Fault on phase A at a distance of 140km from the source

The snapshot of the fifth level decomposition of the sampled voltage waveforms of LG fault on phase A (of Figure 3.7) is depicted in Figure 3.18. d_1, d_2, d_3, d_4, and d_5 represent the detail coefficients for levels 1, 2, 3, 4, and 5 respectively while a_5 is the fifth level approximation coefficient. As clearly shown, the three phases are concatenated together of

matric data of 9141x3 in all. The first set of data of 3047x1 in a column (from 1 to 3047 in x axis) is for phase A, the second set of data of 3047x1 (from 3048 to 6095 in x axis) is for phase B and the third set of data of 3047x1 (from 6096 to 9141 in x axis) is for phase C.

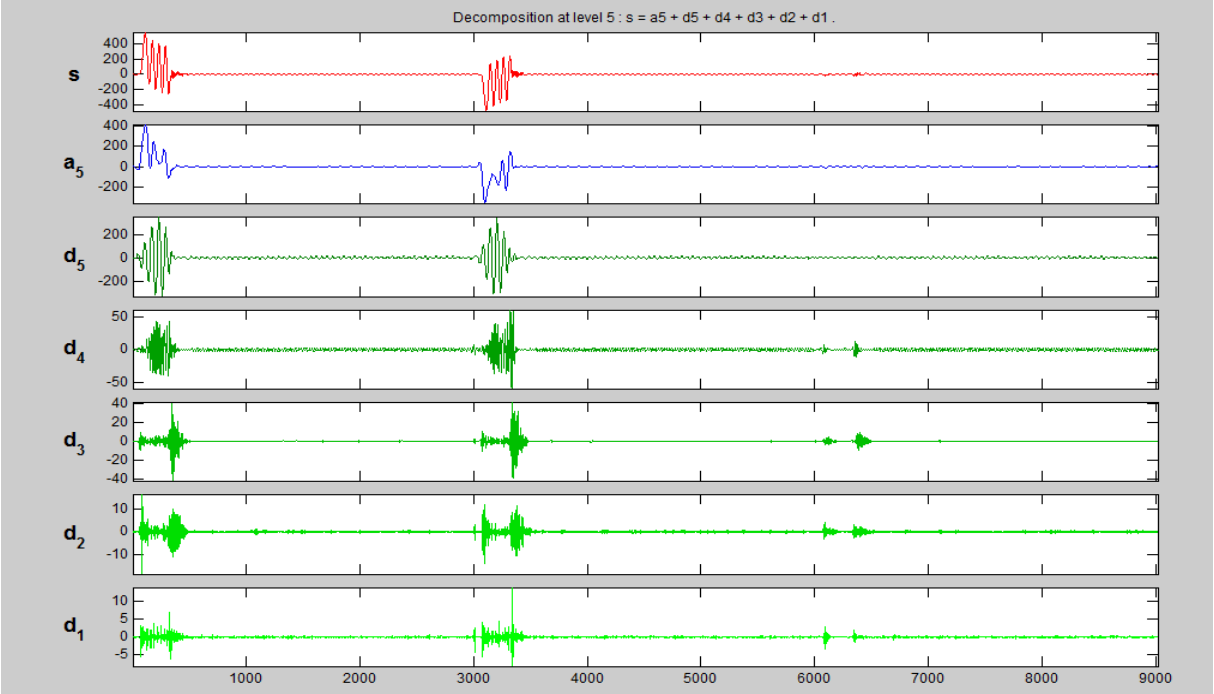


Figure 3.19: Decomposed signal of Current waveform of LLG Fault on phase A at a distance of 140km from the source

Figure 3.19 shows the snapshot of the fifth level decomposition of the sampled current waveforms of LLG fault on phase A and B (of Figure 3.8). d_1, d_2, d_3, d_4, and d_5 represent the detail coefficients for levels 1, 2, 3, 4, and 5 respectively while a_5 is the fifth level approximation coefficient. As clearly shown, the three phases are concatenated together of matric data of 9021x3 in all. The first set of data of 3007x1 in a column is for phase A, the second set of data of 3007x1 is for phase B and the third set of data of 3007x1 is for phase C. The information of original signal is clearly represented at each frequency band.

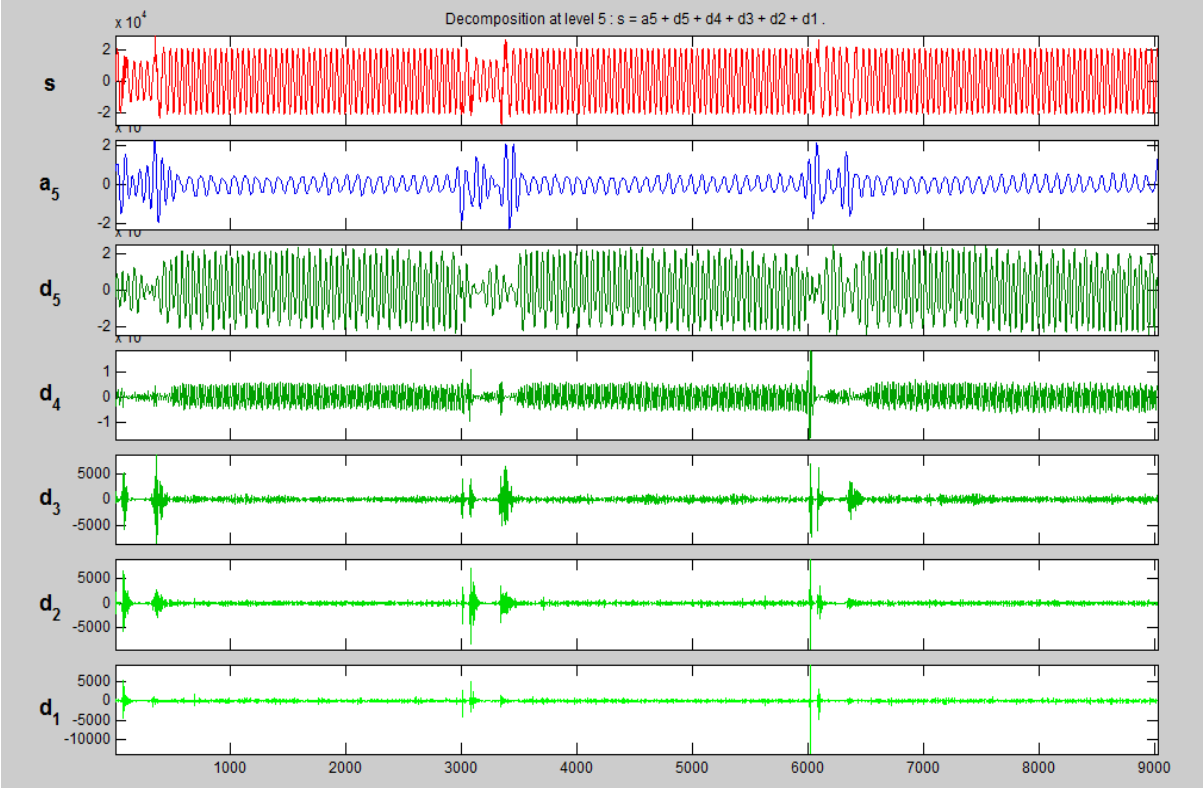


Figure 3.20: Decomposed signal of voltage waveform of LLG Fault on phase A at a distance of 140km from the source

The snapshot of the fifth level decomposition of the sampled voltage waveforms of LLG fault on phase A and B (of Figure 3.9) is shown in Figure 3.20. d_1, d_2, d_3, d_4, and d_5 represent the detail coefficients for levels 1, 2, 3, 4, and 5 respectively while a_5 is the fifth level approximation coefficient. As clearly shown, the three phases are concatenated together of matric data of 9021x3 in all. The first set of data of 3007x1 in a column is for phase A, the second set of data of 3007x1 is for phase B and the third set of data of 3007x1 is for phase C.

Figure 3.21 depicts the snapshot of the fifth level decomposition of the sampled current waveforms of LL fault on phase A and B (of Figure 3.10). d_1, d_2, d_3, d_4, and d_5 represent the detail coefficients for levels 1, 2, 3, 4, and 5 respectively while a_5 is the fifth level

approximation coefficient. As clearly shown, the three phases are concatenated together of matric data of 9420x3 in all. The first set of data of 3140x1 in a column is for phase A, the second set of data of 3140x1 is for phase B and the third set of data of 3140x1 is for phase C. The information of original signal is clearly represented at each frequency band.

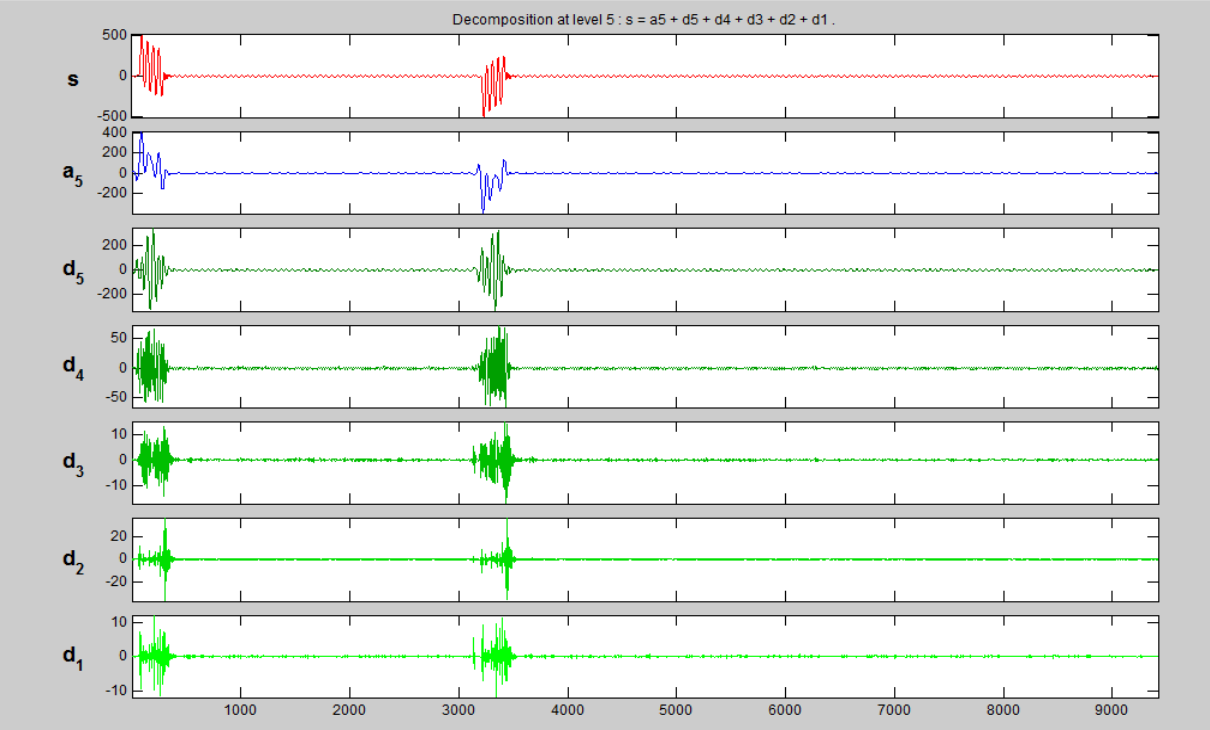


Figure 3.21: Decomposed signal of Current waveform of LL Fault on phase A at a distance of 140km from the source

The snapshot of the fifth level decomposition of the sampled voltage waveforms of LL fault on phase A and B (of Figure 3.11) is depicted in Figure 3.22. d_1, d_2, d_3, d_4, and d_5 represent the detail coefficients for levels 1, 2, 3, 4, and 5 respectively while a_5 is the fifth level approximation coefficient. As clearly shown, the three phases are concatenated together of matric data of 9420x3 in all. The first set of data of 3140x1 in a column is for phase A, the

second set of data of 3140x1 is for phase B and the third set of data of 3140x1 is for phase C. The information of original signal is clearly represented at each frequency band.

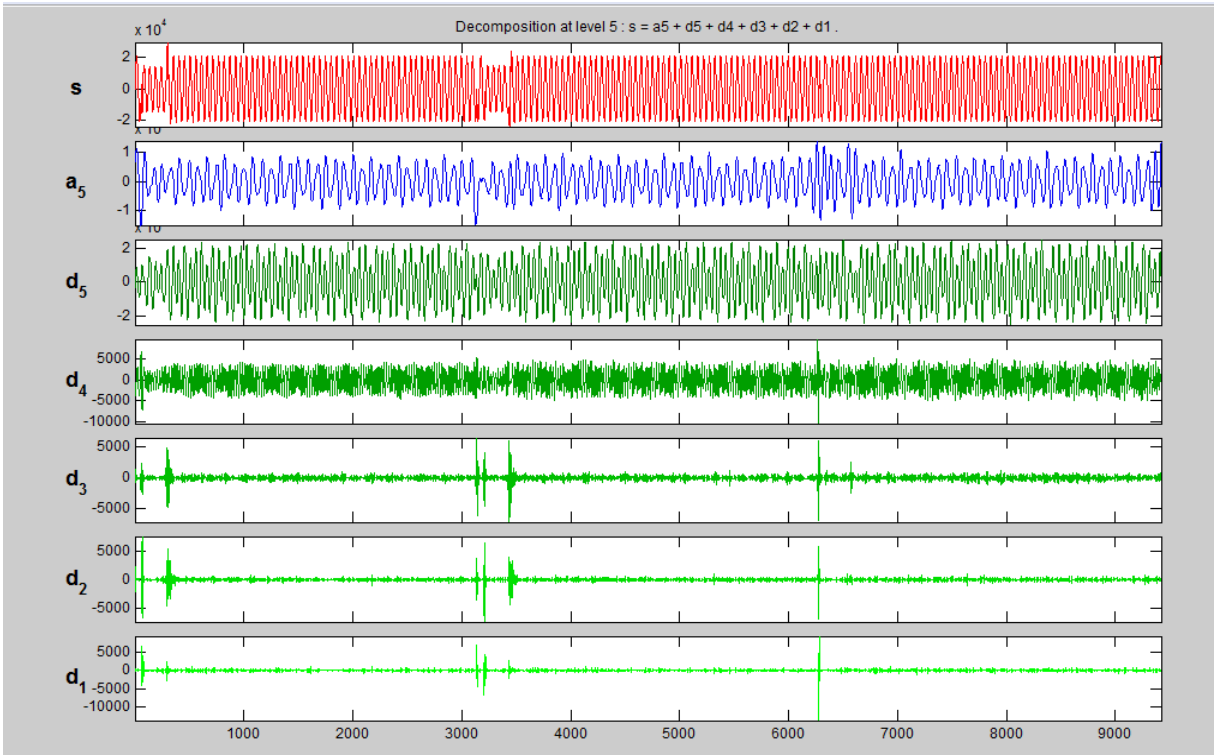


Figure 3.22: Decomposed waveform of voltage waveform of LL Fault on phase A at a distance of 140km from the source

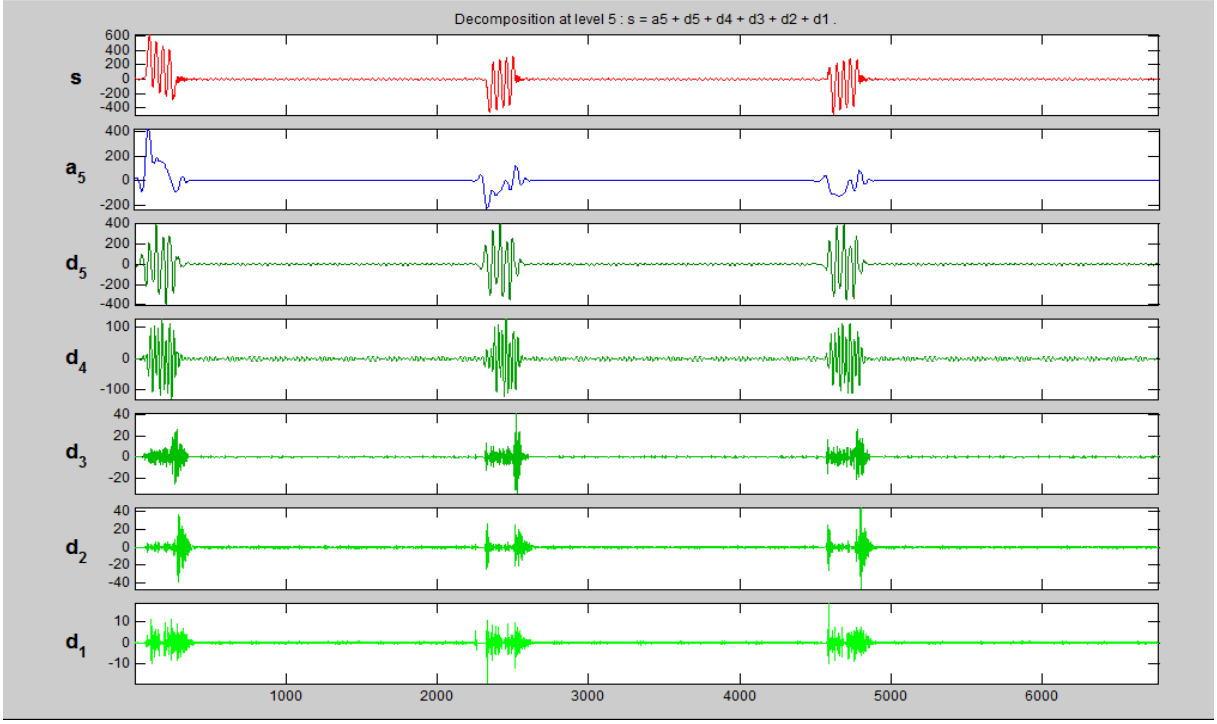


Figure 3.23: Decomposed waveform of Current waveform of LLL Fault on the three phases at a distance of 140km from the source

Figure 3.23 shows the snapshot of the fifth level decomposition of the sampled current waveforms of LLL fault on the three phases (of Figure 3.12). d_1, d_2, d_3, d_4, and d_5 represent the detail coefficients for levels 1, 2, 3, 4, and 5 respectively while a_5 is the fifth level approximation coefficient. As clearly shown, the three phases are concatenated together of matric data of 6768x3 in all. The first set of data of 2256x1 in a column is for phase A, the second set of data of 2256x1 is for phase B and the third set of data of 2256x1 is for phase C.

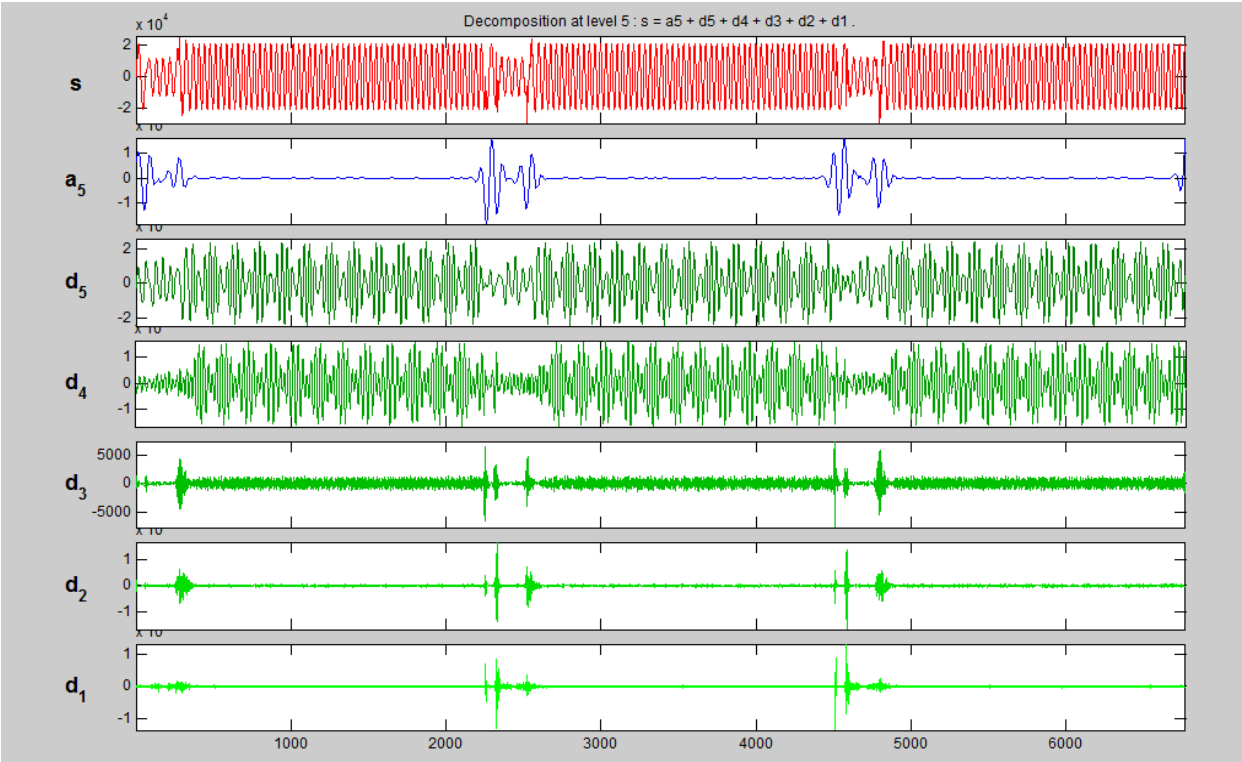


Figure 3.24: Decomposed signal of voltage waveform of LLL Fault on the three phases at a distance of 140km from the source

The snapshot of the fifth level decomposition of the sampled voltage waveforms of LLL fault on the three phases (of Figure 3.13) is shown in Figure 3.24. d_1, d_2, d_3, d_4, and d_5 represent the detail coefficients for levels 1, 2, 3, 4, and 5 respectively while a_5 is the fifth level approximation coefficient. As clearly shown, the three phases are concatenated together of matric data of 6768x3 in all. The first set of data of 2256x1 in a column is for phase A, the second set of data of 2256x1 is for phase B and the third set of data of 2256x1 is for phase C. The information of original signal is clearly represented at each frequency band.

Figure 3.25 shows the snapshot of the fifth level decomposition of the sampled current waveforms of No fault condition on the three phases (of Figure 3.14). d_1, d_2, d_3, d_4, and d_5 represent the detail coefficients for levels 1, 2, 3, 4, and 5 respectively while a_5 is the fifth

level approximation coefficient. As clearly shown, the three phases are concatenated together along the row with matric data of 18897x3 in all. The first set of data of 6299x1 in a column is for phase A, the second set of data of 6299x1 is for phase B and the third set of data of 6299x1 is for phase C.

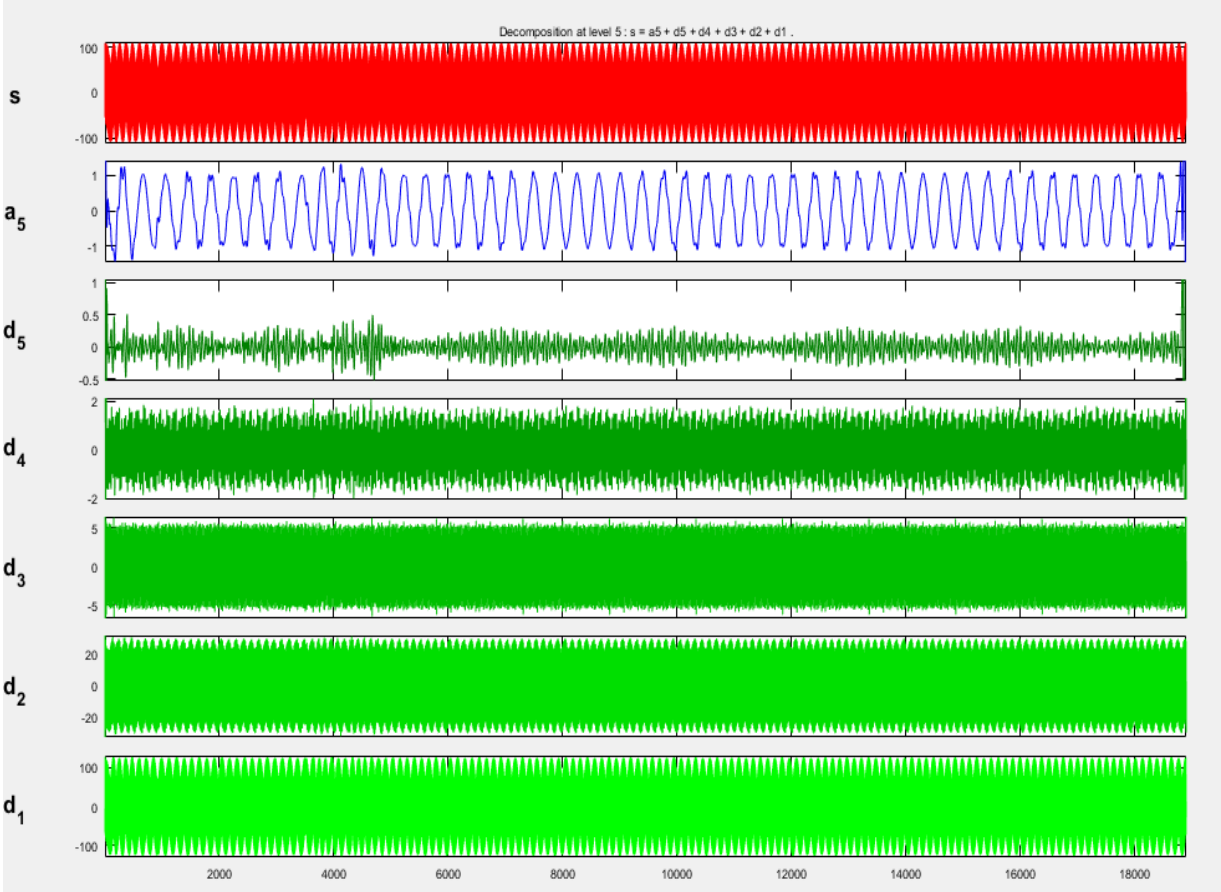


Figure 3.25: Decomposed waveform of Current waveform of No Fault condition on the three phases at a distance of 140km from the source

The snapshot of the fifth level decomposition of the sampled voltage waveforms of No fault condition on the three phases (of Figure 3.15) is shown in Figure 3.26. d_1, d_2, d_3, d_4, and d_5 represent the detail coefficients for levels 1, 2, 3, 4, and 5 respectively while a_5 is the fifth level approximation coefficient. As clearly shown, the three phases are concatenated together of matric data of 18897x3 in all. The first set of data of 6299x1 in a column is for phase A, the

second set of data of 6299x1 is for phase B and the third set of data of 6299x1 is for phase C. The information of original signal is clearly represented at each frequency band.

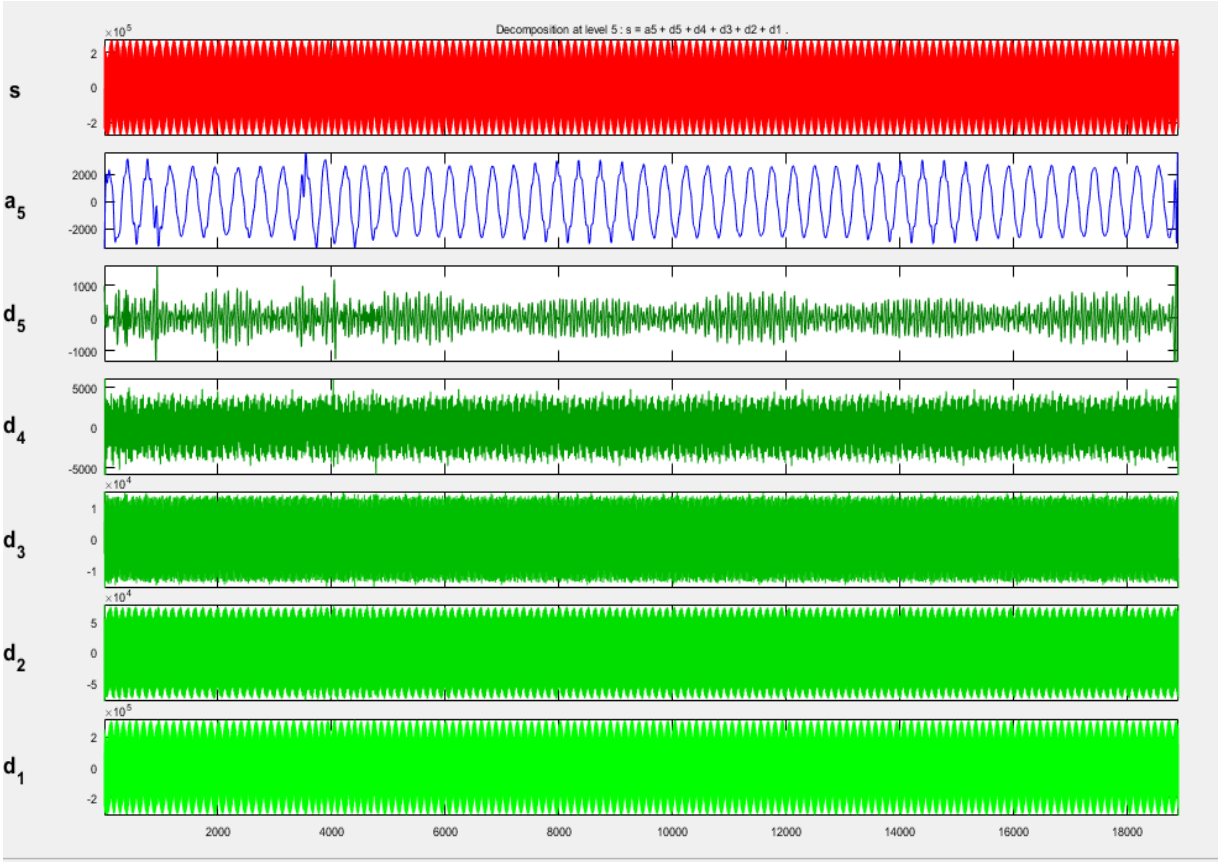


Figure 3.26: Decomposed signal of voltage waveforms of No Fault condition on the three phases at a distance of 140km from the source

The wavelet toolbox in Matlab was used for the above signal decompositions as it provides a lot of useful techniques for wavelet analysis.Based on the sampling rate the signal is divided into 12 decomposition levels. Among different levels only 5th level is considered for analysis because the frequency corresponding to this level is covering 2nd and 3rd harmonics which are dominant in the fault conditions.

As stated before in section 3.7, the new results or data (decomposed signals) are sent back to Matlab workspace in readiness for further usage. From there, using the Matlab codes as shown in Appendix A and B, these results (decomposed signals) are summed up as W_a, W_b, and W_c which are the summation of the 5th level detail coefficients of both the decomposed voltage and current waveforms for the three phases or the summation of the 5th level approximations for both the decomposed current and voltage waveforms for the three phases or the summation of both the 5th level details and approximations of both the decomposed current and voltage waveforms for the three phases fault detection, classification and location respectively.TheseW_a, W_b, and W_c(sample shown in Appendix H) are now used as input to the respective neural networks for training and testing them for mastering in other to recognize the patterns for identification, classification and location purposes.

Finally, these data are imported in the Matlab neural network graphic user interface (GUI) for the purpose of training and testing different chosen neural networks as discussed below.

3.12 Model of a Neuron

A basic neuron model as shown in Fig 3.27 can be described by a function that calculates the output as a function of a number of inputs to it. The basic idea behind the entire neuron model, including the activation functions illustrated below, has been adopted from (Anderson, 2003).

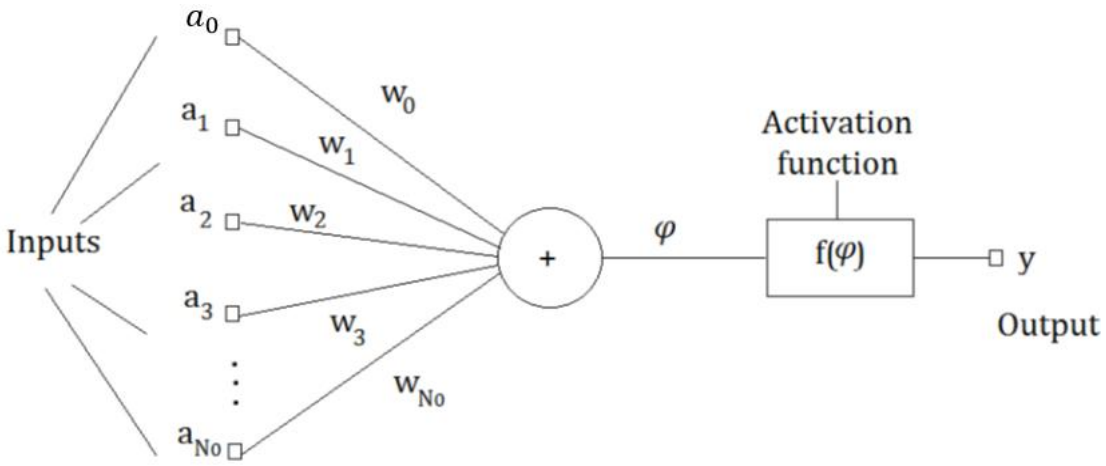


Figure 3.27: Mathematical Model of a Neuron.

The output of the neuron is given by

$$f(\varphi) = f(\sum_{i=0}^{N_0} w_i a_i) \tag{3.17}$$

Where: $w_0 a_0$ is the threshold value (polarization),f(φ) is the neuron activation function, φ is the summation output signal and y is the neuron output.

$$\varphi = \boldsymbol{W^T A} \tag{3.18}$$

Where: $$\boldsymbol{W} = [w_0 w_1 \dots w_{N_0}],\ \boldsymbol{A} = [a_0 a_1 \dots a_{N_0}]^{\boldsymbol{T}} \tag{3.19}$$

The activation function acts as a squashing function, such that the output of a neuron in a neural network is between certain values (usually 0 and 1, or -1 and 1). An activation function decides how powerful the output from the neuron should be, based on the sum of its inputs. Depending upon the application's requirements, the most appropriate activation function is

chosen. The activation function f (φ) can be in different forms.A few of which are illustrated below:

- **Threshold (or Step) Function**

This takes on a value of 0 if the summed input is less than a certain threshold value (φ), and the value 1 if the summed input is greater than or equal to the threshold value.

$$f(\varphi) = \begin{cases} 1, & \text{if } \varphi \geq 0 \\ 0, & \text{if } \varphi < 0 \end{cases}$$

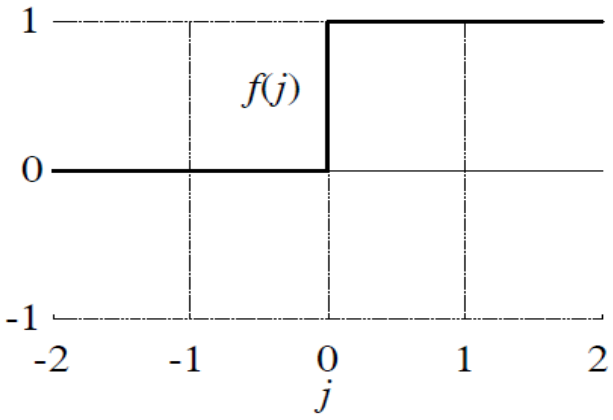


Figure 3.28: Step activation function

- **Piece wise linear function**

This function again can take on the values of 0 or -1, but can also take on values between that depending on the amplification factor in a certain region of linear operation.

$$f(\varphi) = \begin{cases} 1, & \text{for } \varphi > 1 \\ -1, & \text{for } \varphi < -1 \\ \varphi, & \text{for } -1 < \varphi < 1 \end{cases}$$

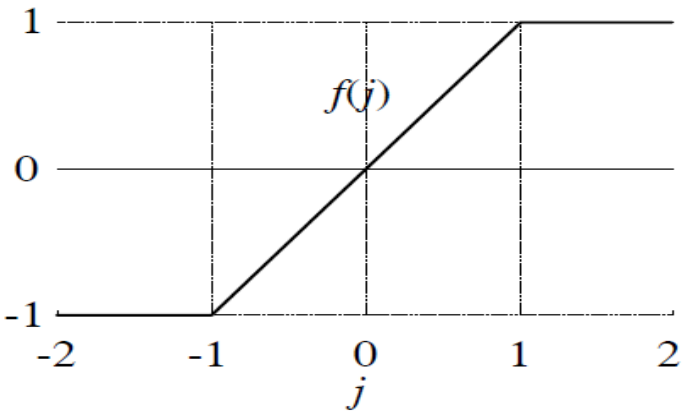


Figure 3.29: Piece wise linear activation function.

- **Sigmoid bipolar function**

This function can range between 0 and 1, but it is also sometimes useful to use the -1 to 1 range. An example of the sigmoid function is the hyperbolic tangent function

$$f(\varphi) = \tanh(\beta\varphi) = \frac{1 - e^{-2\beta\varphi}}{1 + e^{-2\beta\varphi}}$$

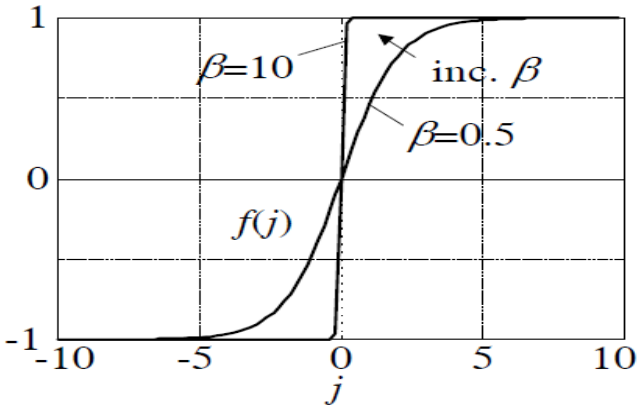


Figure 3.30: Bipolar activation function.

- **Sigmoid unipolar function**

This function can range between 0 and 1 and in this work the Sigmoid activation function has been used since the emphasis is that the neural network output should be 0 or 1 (no or yes).

$$f(\varphi) = \frac{1}{1+e^{-\beta\varphi}}$$

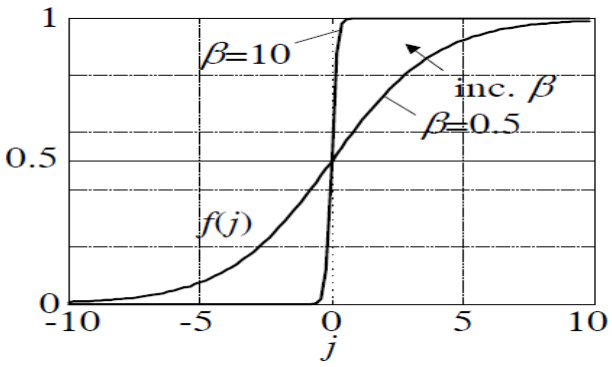


Figure3.31: Sigmoid unipolar activation function.

Based on the way the neurons are interconnected in a model, neural networks can be broadly classified into two types namely feed-forward and feed-back networks. As the name suggests, feedback networks unlike feed-forward networks have a feedback connection fed back into the network along with the inputs. Due to their simplicity and the existence of a well-defined learning algorithm, only feed-forward networks have been used in this dissertation for the simulation and hence the application is presented in the upcoming sections.

3.13 The Feed-forward Networks

Feed-forward networks are the simplest neural networks where there is no feedback connection involved in the network and hence the information travel is unidirectional (El-

Sharkawi, Niebur, 1996). A feed-forward network with $a_{\mathrm{N}0}$ input and y_{NR} outputsignals is shown in Fig 3.32. The computation process in the ith layer can be described by the following (3.20)

$$\boldsymbol{p}^{(i)} = f^{(i)}\left(\boldsymbol{w}^{(i)}\boldsymbol{g}^{(i-1)}\right) \tag{3.20}$$

Where $\boldsymbol{p}^{(i)} = \left[p_1^{(i)} p_2^{(i)} p_3^{(i)} \ \ldots \ p_{N_i}^{(i)}\right]^{\boldsymbol{T}}$ is the fault signal vector at the output of the *i*th layer.

And $\boldsymbol{W}^{(i)} = \begin{pmatrix} w_{10}^{(i)} & w_{11}^{(i)} & \cdots & w_{1N_{i-1}}^{(i)} \\ w_{20}^{(i)} & w_{21}^{(i)} & \cdots & w_{2N_{i-1}}^{(i)} \\ \vdots & \vdots & \ddots & \vdots \\ w_{N_i 0}^{(i)} & w_{N_i 1}^{(i)} & \cdots & w_{N_i N_{i-1}}^{(i)} \end{pmatrix}$ is the weighing matrix between the (*i*-1)th and the *i*th layer.

$$\boldsymbol{g}^{(i-1)} = \begin{cases} A & for\ i = 1 \\ \begin{bmatrix} 1 \\ \boldsymbol{p}^{(i-1)} \end{bmatrix} & for\ i = 2,3,\ldots,R \end{cases} \tag{3.21}$$

A is the vector containing the input signals,$\boldsymbol{f}^{(i)}$ is the activation function of the neurons in the *i*th layer and *p*is the number of processing layers. All the neurons in a particular layer are assumed to be similar in all aspects and the number of hidden layers can be more than one and is usually determined by the purpose of the neural network. The output of the processed neural network is represented by the output vector:

$$\boldsymbol{y} = \boldsymbol{p}^{(\boldsymbol{R})} = [\mathrm{y}_1 \mathrm{y}_2 \mathrm{y}_3 \ \ldots \mathrm{y}_{\mathrm{N_R}}]^T \tag{3.22}$$

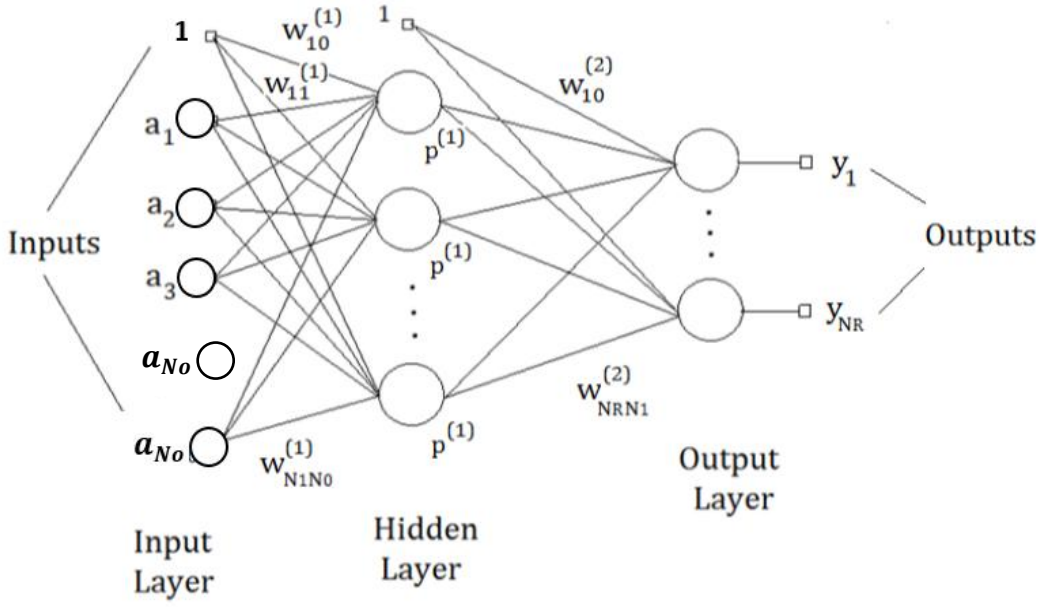


Figure 3.32: Structure of a two-layered feed-forward network.

3.14 The Training or Learning Process

Two important steps in the application of neural networks for any purpose are training and testing. The first of the two steps namely training of the chosen neural networks is dealt with in this section. Training is the process by which the neural network learns from the inputs and updates its weights accordingly. In order to train the neural network, a set of data called the training data-set is needed which is a set of input output pairs fed into the neural network. Thereby, the neural network is taught what the output should be, when that particular input is fed into it. The ANN slowly learns the training set and slowly develops an ability to generalize upon this data and will eventually be able to produce an output when a new data is provided to it. During the training process, the neural network's weights are updated with the prime goal of minimizing the performance function. This performance function can be user defined, but usually feed-forward networks employ Mean Square Error as the performance function and the same is adopted throughout this work.

As already mentioned, all the voltages and currents fed into the neural network are the summation of the wavelet decomposed fault voltage and current values of all the three phases. The outputs, depending upon the purpose of the neural network might be the fault condition, the type of fault or the location of the fault on the transmission line.

For the task of training the neural networks for different stages, sequential feeding of input and output pair has been adopted. In order to obtain a large training set for efficient performance, each of the ten kinds of faults has been simulated at different locations along the considered transmission line.

Apart from the type of fault, the phases that are faulted and the distance of the fault along the transmission line, thefault distance has been varied at an incremental factor of every 15kmon a 280km transmission line. That is, at every 15km, the 10 kinds of faults were simulated.

3.14.1 Training the Fault Identification Neural Network

Various topologies of Multi-Layer Perceptron have been studied, for the purpose of fault identification. The various factors that played a role in deciding the ideal topology are the network size, the learning strategy employed and the training data set size.

The back-propagation algorithm has been adopted as the ideal topology, after an exhaustive study. Even though the basic back-propagation algorithm is relatively slow due to the small learning rates employed, few techniques can significantly enhance the performance of the algorithm. One of such strategy is to use the Levenberg-Marquardt or Scaled Conjugate Gradient optimization technique. The selection of the apt network size is very vital because this not only reduces the training time but also greatly enhance the ability of the neural

network to represent the problem in hand. Unfortunately, there is no thumb rule that can dictate the number of hidden layers and the number of neurons per hidden layer in a given problem.

In this work, in the first stage which is the fault identification phase, the network takes in three inputs (W_a, W_b, and W_c) at a time, which are the summation of the detail coefficients of both current and voltage for all three phases. The entire input data set (3047x3 vector matrix as generated in Appendix B) are subdivided into three; 60%, 20%, 20% for the training set, validation set and testing set respectively giving a set of three inputs and one output in each input-output pair. Large and enough percentage of the input data obtained has to be used as the training data-set so as to give the NN enough data samples to learn or master. Hence, the choice of the 60% for training the NN in this work.The output of the neural network is just a yes or a no (1 or 0) depending on whether or not a fault has been detected.

3.14.2 Training the Fault Classifier Neural Network

Fault classifiers based on neural networks made use of multilayer perceptron neural network and employed the back-propagation learning strategy. Although back-propagation learning strategy as aforementioned is inherently slow in learning and poses difficulty in choosing the optimal size of the network, it is undoubtedly the ideal strategy to be employed when there is a large training set available because back-propagation algorithm can provide a very compact distributed representation of complex data sets.

The same process that was employed in the previous section (section 3.14.1) is also followed in this section in terms of the design and development of the classifier neural network. The designed network takes in sets of three inputs (W_a, W_b, and W_c) at a time, which are the summation of the coarse approximations of both current and voltage for all three phases. The

entire input data set (3007x3 vector matrix as generated in Appendix B) are subdivided into three; 60%, 20%, 20% for the training set, validation set and testing set respectively giving a set of three inputs and one output in each input-output pair. The neural network has four outputs, each of them corresponding to the fault condition of each of the three phases and one output for the ground line. Hence, the outputs are either a 0 or 1 denoting the absence or presence of a fault on the corresponding line (A, B, C or G). Where A, B and C denote the three phases of the transmission line and G denotes the ground. Hence the various possible permutations can represent each of the various faults accordingly. The proposed neural network was able to accurately distinguish between the ten possible categories of faults. The truth table representing the faults and the ideal output for each of the faults is illustrated in Table 3.4.

Table 3.4: Fault classifier ANN outputs for various faults

Type of Fault		**Network Outputs**			
		A	**B**	**C**	**G**
L-G	A-G Fault	1	0	0	1
	B-G Fault	0	1	0	1
	C-G Fault	0	0	1	1
L-L	A-B Fault	1	1	0	0
	B-C Fault	0	1	1	0
	C-A Fault	1	0	1	0
L-L-G	A-B-G Fault	1	1	0	1
	B-C-G Fault	0	1	1	1
	C-A-G Fault	1	0	1	1

3 -Phase	A-B-C Fault	1	1	1	0

3.14.3a Training the Neural Network for Single Line – Ground Fault Location

Feed forward back – propagation neural networks have been surveyed for the purpose of single line – ground fault location, mainly because of the availability of sufficient relevant data for training. In order to train the neural network, several single phase faults have been simulated on the transmission line model. For each of the three phases, faults have been simulated at every 15km on the 280km long transmission line. Here, the summed-up detail coefficients and approximations (W_a, W_b, and W_c) of the decomposed fault currents and voltage of the three phases, are given as inputs to the neural network. The entire input data set (3047x3 vector matrix) are subdivided into three; 60%, 20%, 20% for the training set, validation set and testing set respectively giving a set of three inputs and one output in each input-output pair. The output of the neural network is the distance of the fault from terminal A. Efficiency of each of the trained networks was analyzed based on their regression performance and their performance in the testing phase. The test performance plots are obtained by simulating various faults on different phases at varying locations and calculating the error in the output produced by the Neural Network as shown in chapter four.

3.14.3b Training the Neural Network for Line – Line Fault Location

Because of the availability of sufficient data to train the network, feed forward back – propagation neural networks have been surveyed for the purpose of line – line fault location and secondly training automatically stops when generalization stops improving, as indicated

by a decrease in the mean square error of the validation curves (see chapter four). In order to train the neural network, several line – line faults have been simulated on the transmission line model. For each pair formed by the three phases, faults have been simulated at every 15km on a 280km long transmission line. In this case, the summed-up detail and approximation coefficients (W_a, W_b, and W_c) of the fault currents and voltages of the three phases, are given as inputs to the neural network. The entire input data set (which is a 3140x3 vector matrix) are subdivided into three; 60%, 20%, 20% for the training set, validation set and testing set respectively giving a set of three inputs and one output in each input-output pair. The output of the neural network is the distance to the fault from terminal A. Hence, each input output pair consists of three inputs and one output. An exhaustive survey on various neural networks has been performed by varying the number of hidden layers and the number of neurons per hidden layer.

3.14.3c Training the Neural Network for Double Line – Ground Fault Location

Feed forward back – propagation algorithm was once again used for the purpose of double line – ground fault location on transmission lines. The reason for doing so, as already mentioned is that these networks perform very efficiently when there is availability of a sufficiently large training data set. For the purpose of training the neural network, several double line – ground faults have been simulated on the modelled transmission line (Figure 3.3) on each of the three phases. The various factors that were varied were the fault distance (incremented by 15km each time) and the phases that were faulted. In this case, the summed-up detail and approximation coefficients (W_a, W_b, and W_c) of the decomposed fault currents and voltages of the three phases, were given as inputs to the neural network. The entire input data set (which is a 3007x3 vector matrix) is subdivided into three; 60%, 20%, 20% for the

training set, validation set and testing set respectively giving a set of three inputs and one output in each input-output pair. The neural network's output is the distance to the fault from terminal A. Thus, each input output pair fed into the neural network has a set of three inputs and one output.

3.14.3d Training the Neural Network for Three Phase Fault Location

Feed forward back – propagation algorithm was once again used for the purpose of three phase fault location on transmission lines. The reason for doing so, as already mentioned is that these networks perform very efficiently when there is availability of a sufficiently large training data set. For the purpose of training the neural network, several three phase faults have been simulated on the modelled transmission line. The various factors that were varied were the fault distance (incremented by 15km each time) and the fault resistance (one of the chosen ten different fault resistances).

Here, the summed-up detail and approximation coefficients (W_a, W_b, and W_c) of the decomposed fault currents and voltages of the three phases, are given as inputs to the neural network. The entire input data set (which is a 2256 x3 vector matrix) is subdivided into three; 60%, 20%, 20% for the training set, validation set and testing set respectively giving a set of three inputs and one output in each input-output pair. The neural network's output is the distance to the fault from terminal A. Thus, each input output pair fed into the neural network has a set of three inputs and one output.

3.15 Training Process of the Neural Networks using Back-Propagation Algorithm

As mention in section 3.14, the basic concept behind the successful application of neural networks in any field is to determine the weights to achieve the desired target and this process

is called learning or training. The network weights are modified with the prime objective of minimization of the error between a given set of inputs and their corresponding target values. However, in this work, the back-propagation algorithm is employed in training all the neural networks. The steps below are mathematical calculation and illustration of network back-propagation algorithm. For this purpose, a NN architecture of (3.3.4.1) has been adopted as shown in Figure 3.33.

From Figure 3.33,$\delta h1$, $\delta h2$, $\delta h3$, $\delta h4$, $\delta h5$, $\delta h6$, $\delta h7$ and $\delta o1$ are the local gradients of all the eight nodes (neurons) n1, n2, n3, n4, n5, n6, n7 and n8 respectively. Where v1, v2, v3, v4, v5, v6, v7 and v8 are the algebraic sum of all the inputs to the nodes (neurons) n1, n2, n3, n4, n5, n6, n7 and n8 respectively and w is the synaptic weight, b is the bias weight, y is the actual neuron output, and d8 is the desired (target) output.

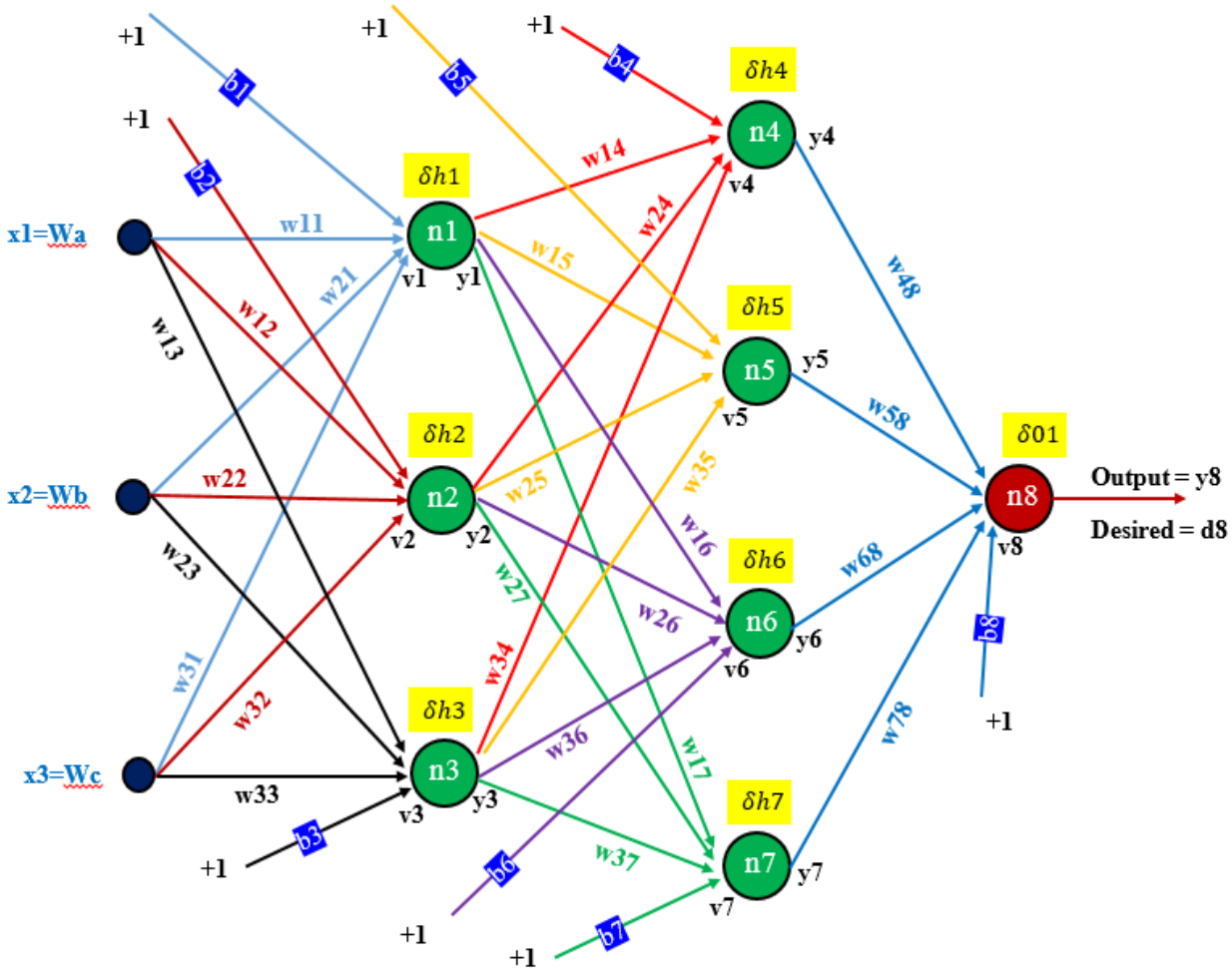


Figure 3.33: Mathematical Implementation of Back-Propagation Algorithm

Step 1: Performing the First Forward Pass (Feed-forward)

The aim here is to determine the actual output (y_8) of the network (Figure 3.33) and from there the error (e) can be calculated. The algebraic sums (V) of the inputs to the neurons and the actual outputs (y) of individual neurons in the network are determined as follows:

$$v1 = 1 * b1 + x1 * w11 + x2 * w21 + x3 * w31$$

$$y1 = f(v1) \qquad 3.23$$

$$v2 = 1 * b2 + x1 * w1 + x2 * w22 + x3 * w32$$

$$y2 = f(v2) \quad 3.24$$

$$v3 = 1 * b3 + x1 * w13 + x2 * w23 + x3 * w33$$

$$y3 = f(v3) \quad 3.25$$

$$v4 = 1 * b4 + y1 * w14 + y2 * w24 + y3 * w34$$

$$y4 = f(v4) \quad 3.26$$

$$v5 = 1 * b5 + y1 * w15 + y2 * w25 + y3 * w35$$

$$y5 = f(v5) \quad 3.27$$

$$v6 = 1 * b6 + y1 * w16 + y2 * w26 + y3 * w36$$

$$y6 = f(v6) \quad 3.28$$

$$v7 = 1 * b7 + y1 * w17 + y2 * w27 + y3 * w37$$

$$y7 = f(v7) \quad 3.29$$

$$v8 = 1 * b8 + y4 * w48 + y5 * w58 + y6 * w68 + y7 * w78$$

$$y8 = f(v8) \quad 3.30$$

Now,

the error (e) = desired output $(d8)$ − actual output $(y8)$

$$e = d8 - y8$$

Step 2: Performing the First Backward Pass (Back-propagation)

The aim here is to use the error obtained to adjust the synaptic weights in other to minimize the error. The neuron activation function used is sigmoid activation function as written in (3.31)

$$f(v) = \frac{1}{1+\exp(-v)} \tag{3.31}$$

And its derivative is

$$f'(v) = f(v)[1 - f(v)] \tag{3.32}$$

First the local gradient ($\delta o1$) of the output neuron (n8) in the network of Figure 3.33 is determined as:

$$\delta o1 = f'(v8) * e \ = f'(v8) * (d8 - y8)$$

$$= f'(1 * b8 + y4 * w48 * y5 * w58 + y6 * w68 + y7 * w78) * (d8 - y8) \tag{3.33}$$

Next the local gradients of all the seven hidden neurons starting from those in the last (2nd) hidden layer to those in the first hidden layer are determined as:

For the 2nd Hidden Layer

$$\delta h4 \ = f'(v4) * (\delta o1 * w48)$$

$$= f'(1 * b4 + y1 * w14 * y2 * w24 + y3 * w34) * (\delta o1 * w48) \tag{3.34}$$

$$\delta h5 \ = f'(v5) * (\delta o1 * w58)$$

$$= f'(1 * b5 + y1 * w15 * y2 * w25 + y3 * w35) * (\delta o1 * w58) \tag{3.35}$$

$$\delta h6 = f'(v6) * (\delta o1 * w68)$$

$$= f'(1 * b6 + y1 * w16 * y2 * w26 + y3 * w36) * (\delta o1 * w68) \quad (3.36)$$

$$\delta h7 = f'(v7) * (\delta o1 * w78)$$

$$= f'(1 * b7 + y1 * w17 * y2 * w27 + y3 * w37) * (\delta o1 * w78) \quad (3.37)$$

For the First Hidden Layer

$$\delta h1 = f'(v1) * [(\delta h4 * w14) * (\delta h5 * w15) * (\delta h6 * w16) * (\delta h7 * w17)] \quad (3.38)$$

$$\delta h2 = f'(v2) * [(\delta h4 * w24) * (\delta h5 * w25) * (\delta h6 * w26) * (\delta h7 * w27)] \quad (3.39)$$

$$\delta h3 = f'(v3) * [(\delta h4 * w34) * (\delta h5 * w35) * (\delta h6 * w36) * (\delta h7 * w37)] \quad (3.40)$$

Step 3:

Adjusting the weights of the network with the local gradients as obtained using the back-propagation general learning rule:

$$\boldsymbol{w(n+1) = w(n) + \alpha * w(n-1) + \eta * \delta(n) * y} \quad \textbf{(3.41)}$$

Where, $w(n+1)$ is the new weight, $w(n)$ is the current weight, $w(n-1)$ is the previous weight, α is the mobility factor, η is the learning rate or the training parameter, $\delta(n)$ is the current local gradient.

Applying (3.41) to Figure 3.33, the new weights read:

$$w48(n+1) = w48(n) + \alpha * w48(n-1) + \eta * \delta o1(n) * y4 \quad (3.42)$$

$$w58(n+1) = w58(n) + \alpha * w58(n-1) + \eta * \delta o1(n) * y5 \quad (3.43)$$

$$w68(n+1) = w68(n) + \alpha * w68(n-1) + \eta * \delta o1(n) * y6 \quad (3.44)$$

$$w78(n+1) = w78(n) + \alpha * w78(n-1) + \eta * \delta o1(n) * y7 \quad (3.45)$$

$$w14(n+1) = w14(n) + \alpha * w14(n-1) + \eta * \delta h4(n) * y1 \quad (3.46)$$

$$w15(n+1) = w15(n) + \alpha * w15(n-1) + \eta * \delta h5(n) * y1 \quad (3.47)$$

$$w16(n+1) = w16(n) + \alpha * w16(n-1) + \eta * \delta h6(n) * y1 \quad (3.48)$$

$$w17(n+1) = w17(n) + \alpha * w17(n-1) + \eta * \delta h7(n) * y1 \quad (3.49)$$

$$w24(n+1) = w24(n) + \alpha * w24(n-1) + \eta * \delta h4(n) * y2 \quad (3.50)$$

$$w25(n+1) = w25(n) + \alpha * w25(n-1) + \eta * \delta h5(n) * y2 \quad (3.51)$$

$$w26(n+1) = w26(n) + \alpha * w26(n-1) + \eta * \delta h6(n) * y2 \quad (3.52)$$

$$w27(n+1) = w27(n) + \alpha * w27(n-1) + \eta * \delta h7(n) * y2 \quad (3.53)$$

$$w34(n+1) = w34(n) + \alpha * w34(n-1) + \eta * \delta h4(n) * y3 \quad (3.54)$$

$$w35(n+1) = w35(n) + \alpha * w35(n-1) + \eta * \delta h5(n) * y3 \quad (3.55)$$

$$w36(n+1) = w36(n) + \alpha * w36(n-1) + \eta * \delta h6(n) * y3 \quad (3.56)$$

$$w37(n+1) = w37(n) + \alpha * w37(n-1) + \eta * \delta h7(n) * y3 \quad (3.57)$$

$$w11(n+1) = w11(n) + \alpha * w11(n-1) + \eta * \delta h1(n) * x1 \quad (3.58)$$

$$w12(n+1) = w12(n) + \alpha * w12(n-1) + \eta * \delta h2(n) * x1 \quad (3.59)$$

$$w13(n+1) = w13(n) + \alpha * w13(n-1) + \eta * \delta h3(n) * x1 \quad (3.60)$$

$$w21(n+1) = w21(n) + \alpha * w21(n-1) + \eta * \delta h1(n) * x2 \quad (3.61)$$

$$w22(n+1) = w22(n) + \alpha * w22(n-1) + \eta * \delta h2(n) * x2 \quad (3.62)$$

$$w23(n+1) = w23(n) + \alpha * w23(n-1) + \eta * \delta h3(n) * x2 \quad (3.63)$$

$$w31(n+1) = w31(n) + \alpha * w31(n-1) + \eta * \delta h1(n) * x3 \quad (3.64)$$

$$w32(n+1) = w32(n) + \alpha * w32(n-1) + \eta * \delta h2(n) * x3 \quad (3.65)$$

$$w33(n+1) = w33(n) + \alpha * w33(n-1) + \eta * \delta h3(n) * x3 \quad (3.66)$$

$$b8(n+1) = b8(n) + \alpha * b8(n-1) + \eta * \delta o1(n) * 1 \quad (3.67)$$

$$b7(n+1) = b7(n) + \alpha * b7(n-1) + \eta * \delta h7(n) * 1 \quad (3.68)$$

$$b6(n+1) = b6(n) + \alpha * b6(n-1) + \eta * \delta h6(n) * 1 \quad (3.69)$$

$$b5(n+1) = b5(n) + \alpha * b5(n-1) + \eta * \delta h5(n) * 1 \quad (3.70)$$

$$b4(n+1) = b4(n) + \alpha * b4(n-1) + \eta * \delta h4(n) * 1 \quad (3.71)$$

$$b3(n+1) = b3(n) + \alpha * b3(n-1) + \eta * \delta h3(n) * 1 \quad (3.72)$$

$$b2(n+1) = b2(n) + \alpha * b2(n-1) + \eta * \delta h2(n) * 1 \quad (3.73)$$

$$b1(n+1) = b1(n) + \alpha * b1(n-1) + \eta * \delta h1(n) * 1 \quad (3.74)$$

3.15.1 Applying Specific Values to Step 1, 2 and 3

For further illustration, a complete forward and backward sweep of the feedforward network (3.3.4.1 architecture) is performed as shown below using the back-propagation algorithm discussed in section 3.15 with specific values. In this case the neural network architecture of (3.3.4.1) is used as shown in Figure 3.34with the parameter values attached. The values of the

initial weights are chosen arbitrary and the inputs (x1, x2 and x3) are the representatives of the summation of the decomposed signals (Wa, Wb and Wc respectively).

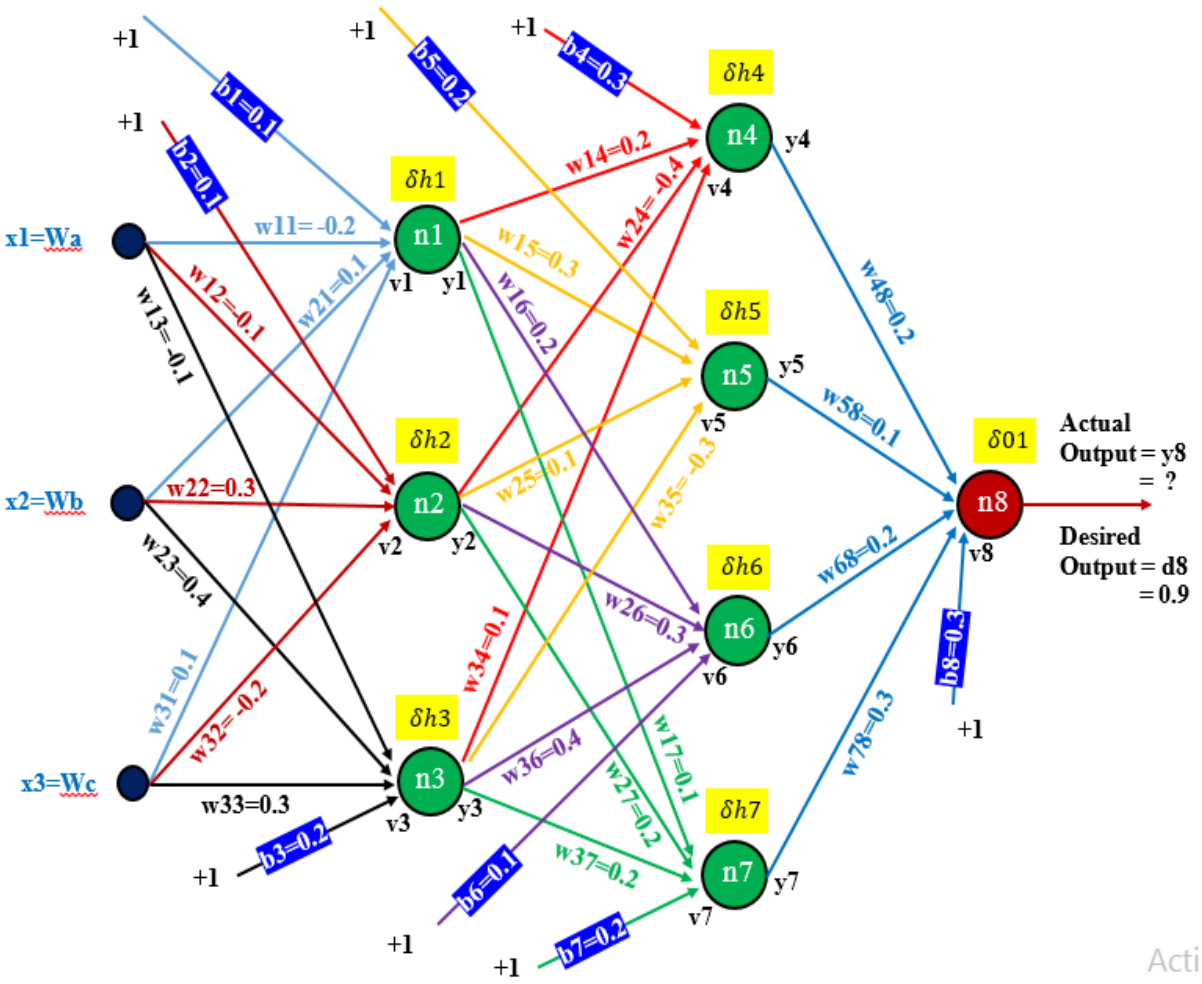


Figure 3.34: A (3.3.4.1) NN Architecture with the corresponding weights shown

Assumptions:

Let the desired (target) output (d) = 0.9 and actual output (y) is unknown,

Learning rate, $\eta = 0.25$,

Mobility Factor, $\alpha = 0.0001$

The Forward Pass

$$v1 = 1 * b1 + x1 * w11 + x2 * w21 + x3 * w31$$

$$= 1 * 0.1 + 0.1 * (-0.2) + 0.9 * 0.1 + 0.5 * 0.1 = 0.22$$

$$y1 = f(v1) = f(0.22) = \frac{1}{1+\exp(-0.22)} = 0.555$$

$$v2 = 1 * b2 + x1 * w12 + x2 * w22 + x3 * w32$$

$$= 1 * 0.1 + 0.1 * (-0.1) + 0.9 * 0.3 + 0.5 * (-0.2) = 0.26$$

$$y2 = f(v2) = f(0.26) = \frac{1}{1+\exp(-0.26)} = 0.565$$

$$v3 = 1 * b3 + x1 * w13 + x2 * w23 + x3 * w33$$

$$= 1 * 0.2 + 0.1 * (-0.1) + 0.9 * 0.4 + 0.5 * 0.3 = 0.7$$

$$y3 = f(v3) = f(0.7) = \frac{1}{1+\exp(-0.7)} = 0.66819$$

$$v4 = 1 * b4 + y1 * w14 + y2 * w24 + y3 * w34$$

$$= 1 * 0.3 + 0.555 * 0.2 + 0.565 * (-0.4) + 0.66819 * 0.1 = 0.251819$$

$$y4 = f(v4) = f(0.251819) = \frac{1}{1+\exp(-0.251819)} = 0.56262$$

$$v5 = 1 * b5 + y1 * w15 + y2 * w25 + y3 * w35$$

$$= 1 * 0.2 + 0.555 * 0.3 + 0.565 * 0.1 + 0.66819 * (-0.3) = 0.22254$$

$$y5 = f(v5) = f(0.22254) = \frac{1}{1 + \exp(-0.22254)} = 0.55541$$

$$v6 = 1 * b6 + y1 * w16 + y2 * w26 + y3 * w36$$

$$= 1 * 0.1 + 0.555 * 0.2 + 0.565 * 0.3 + 0.66819 * 0.4 = 0.64778$$

$$y6 = f(v6) = f(0.64778) = \frac{1}{1 + \exp(-0.64778)} = 0.65651$$

$$v7 = 1 * b7 + y1 * w17 + y2 * w27 + y3 * w37$$

$$= 1 * 0.2 + 0.555 * 0.1 + 0.565 * 0.2 + 0.66819 * 0.2 = 0.50088$$

$$y7 = f(v7) = f(0.50088) = \frac{1}{1 + \exp(-0.50088)} = 0.62267$$

$$v8 = 1 * b8 + y4 * w48 + y5 * w58 + y6 * w68 + y7 * w78$$

$$= 1 * 0.3 + 0.56262 * 0.2 + 0.55541 * 0.1 + 0.65651 * 0.2 + 062267 * 0.3 = 0.78617$$

$$y8 = f(v8) = f(0.78617) = \frac{1}{1+\exp(-0.78617)} = 0.68701$$

Please note the activation function, $f(v)$ used in the forward pass (feed-forward) and not its derivative $f^{'}(v)$ which is used during the backward pass (back-propagation).

Then,

the error (e) = target output $(d8)$ − actual output $(y8)$

$$= 0.9 - 0.68701 = 0.21299$$

Therefore, after the forward pass, there is an error of 0.21299 which means the back-propagation is required to adjust the weights in other to get the weights that will reduce the error to the global minimal value. The idea is that the actual output should be equal to the target output.

The Backward Pass

Here, the target is to go backward to find out the new weights of the network. This is achieved by applying back-propagation algorithm. First the local gradients of the neurons (nodes) are calculated as was done before but in this case there are specific values. The local gradients of the eight neurons are as follows starting with that of the output neuron.

$$\delta o1 = f^{'}(v8) * e$$

$$= f(v8)[1 - f(v8)] * d8 - y8$$

$$= 0.78617[1 - 0.78617] * (0.9 - 0.68701) = 0.03581$$

$$\delta h4 = f^{'}(v4) * (\delta o1 * w48)$$

$$= f(v4)[1 - f(v4)] * (\delta o1 * w48)$$

$$= 0.251819[1 - 0.251819] * (0.03581 * 0.2) = 0.00135$$

$$\delta h5 = f^{'}(v5) * (\delta o1 * w58)$$

$$= f(v5)[1 - f(v5)] * (\delta o1 * w58)$$

$$= 0.22254[1 - 0.22254] * (0.03581 * 0.1) = 0.00062$$

$$\delta h6 = f^{'}(v6) * (\delta o1 * w68)$$

$$= f(v6)[1 - f(v6)] * (\delta o1 * w68)$$

$$= 0.64778[1 - 0.64778] * (0.03581 * 0.2) = 0.00163$$

$$\delta h7 = f^{'}(v7) * (\delta o1 * w78)$$

$$= f(v7)[1 - f(v7)] * (\delta o1 * w78)$$

$$= 0.50088[1 - 0.50088] * (0.03581 * 0.3) = 0.00269$$

$$\delta h1 = f'(v1) * [(\delta h4 * w14) + (\delta h5 * w15) + (\delta h6 * w16) + (\delta h7 * w17)]$$

$$= 0.22[1 - 0.22] * \begin{bmatrix} (0.00135 * 0.2) + (0.00062 * 0.3) \\ +(0.00163 * 0.2) + (0.00269 * 0.1) \end{bmatrix} = 0.00018$$

$$\delta h2 = f'(v2) * [(\delta h4 * w24) + (\delta h5 * w25) + (\delta h6 * w26) + (\delta h7 * w27)]$$

$$= 0.26[1 - 0.26] * \begin{bmatrix} (0.00135 * -0.4) + (0.00062 * 0.1) \\ +(0.00163 * 0.3) + (0.00269 * 0.2) \end{bmatrix} = 0.00011$$

$$\delta h3 = f'(v3) * [(\delta h4 * w34) + (\delta h5 * w35) + (\delta h6 * w36) + (\delta h7 * w37)]$$

$$= 0.7[1 - 0.7] * \begin{bmatrix} (0.00135 * 0.1) + (0.00062 * -0.3) \\ +(0.00163 * 0.4) + (0.00269 * 0.2) \end{bmatrix} = 0.00024$$

Next is to adjust the weights of the network using the general learning rule expression of (3.41):

$$\boldsymbol{w(n+1) = w(n) + \alpha * w(n-1) + \eta * \delta(n) * y}$$

$$w48(n+1) = w48(n) + \alpha * w48(n-1) + \eta * \delta o1(n) * y4$$

$$= 0.2 + 0.0001 * 0.2 + 0.25 * 0.03581 * 0.56262 = 0.20506$$

$$w58(n+1) = w58(n) + \alpha * w58(n-1) + \eta * \delta o1(n) * y5$$

$$= 0.1 + 0.0001 * 0.1 + 0.25 * 0.03581 * 0.55541 = 0.10498$$

$$w68(n+1) = w68(n) + \alpha * w68(n-1) + \eta * \delta o1(n) * y6$$

$$= 0.2 + 0.0001 * 0.2 + 0.25 * 0.03581 * 0.65651 = 0.20590$$

$$w78(n+1) = w78(n) + \alpha * w78(n-1) + \eta * \delta o1(n) * y7$$

$$= 0.3 + 0.0001 * 0.3 + 0.25 * 0.03581 * 0.62267 = 0.30560$$

$$w14(n+1) = w14(n) + \alpha * w14(n-1) + \eta * \delta h4(n) * y1$$

$$= 0.2 + 0.0001 * 0.2 + 0.25 * 0.00135 * 0.555 = 0.20021$$

$$w15(n+1) = w15(n) + \alpha * w15(n-1) + \eta * \delta h5(n) * y1$$

$$= 0.3 + 0.0001 * 0.3 + 0.25 * 0.00062 * 0.555 = 0.30012$$

$$w16(n+1) = w16(n) + \alpha * w16(n-1) + \eta * \delta h6(n) * y1$$

$$= 0.2 + 0.0001 * 0.2 + 0.25 * 0.00163 * 0.555 = 0.20025$$

$$w17(n+1) = w17(n) + \alpha * w17(n-1) + \eta * \delta h7(n) * y1$$

$$= 0.1 + 0.0001 * 0.1 + 0.25 * 0.00269 * 0.555 = 0.10038$$

$$w24(n+1) = w24(n) + \alpha * w24(n-1) + \eta * \delta h4(n) * y2$$

$$= -0.4 + 0.0001 * -0.4 + 0.25 * 0.00135 * 0.565 = -0.39985$$

$$w25(n+1) = w25(n) + \alpha * w25(n-1) + \eta * \delta h5(n) * y2$$

$$= 0.1 + 0.0001 * 0.1 + 0.25 * 0.00062 * 0.565 = 0.10089$$

$$w26(n+1) = w26(n) + \alpha * w26(n-1) + \eta * \delta h6(n) * y2$$

$$= 0.3 + 0.0001 * 0.3 + 0.25 * 0.00163 * 0.565 = 0.30026$$

$$w27(n+1) = w27(n) + \alpha * w27(n-1) + \eta * \delta h7(n) * y2$$

$$= 0.2 + 0.0001 * 0.2 + 0.25 * 0.00269 * 0.565 = 0.20040$$

$$w34(n+1) = w34(n) + \alpha * w34(n-1) + \eta * \delta h4(n) * y3$$

$$= 0.1 + 0.0001 * 0.1 + 0.25 * 0.00135 * 0.0.66819 = 0.10024$$

$$w35(n+1) = w35(n) + \alpha * w35(n-1) + \eta * \delta h5(n) * y3$$

$$= -0.3 + 0.0001 * -0.3 + 0.25 * 0.00062 * 0.66819 = -0.29993$$

$$w36(n+1) = w36(n) + \alpha * w36(n-1) + \eta * \delta h6(n) * y3$$

$$= 0.4 + 0.0001 * 0.4 + 0.25 * 0.00163 * 0.66819 = 0.40031$$

$$w37(n+1) = w37(n) + \alpha * w37(n-1) + \eta * \delta h7(n) * y3$$

$$= 0.2 + 0.0001 * 0.2 + 0.25 * 0.00269 * 0.66819 = 0.20047$$

$$w11(n+1) = w11(n) + \alpha * w11(n-1) + \eta * \delta h1(n) * x1$$

$$= (-0.2) + 0.0001 * (-0.2) + 0.25 * 0.00018 * 0.1 = -0.20002$$

$$w12(n+1) = w12(n) + \alpha * w12(n-1) + \eta * \delta h2(n) * x1$$

$$= (-0.1) + 0.0001 * (-0.1) + 0.25 * 0.0011 * 0.1 = -0.10001$$

$$w13(n+1) = w13(n) + \alpha * w13(n-1) + \eta * \delta h3(n) * x1$$

$$= (-0.1) + 0.0001 * (-0.1) + 0.25 * 0.00024 * 0.1 = -0.10000$$

$$w21(n+1) = w21(n) + \alpha * w21(n-1) + \eta * \delta h1(n) * x2$$

$$= (0.1) + 0.0001 * (0.1) + 0.25 * 0.00018 * 0.9 = 0.10004$$

$$w22(n+1) = w22(n) + \alpha * w22(n-1) + \eta * \delta h2(n) * x2$$

$$= 0.3 + 0.0001 * 0.3 + 0.25 * 0.00011 * 0.9 = 0.30005$$

$$w23(n+1) = w23(n) + \alpha * w23(n-1) + \eta * \delta h3(n) * x2$$

$$= (0.4) + 0.0001 * (0.4) + 0.25 * 0.00024 * 0.9 = 0.40009$$

$$w31(n+1) = w31(n) + \alpha * w31(n-1) + \eta * \delta h1(n) * x3$$

$$= 0.1 + 0.0001 * 0.1 + 0.25 * 0.00018 * 0.5 = 0.10003$$

$$w32(n+1) = w32(n) + \alpha * w32(n-1) + \eta * \delta h2(n) * x3$$

$$= -0.2 + 0.0001 * -0.2 + 0.25 * 0.00011 * 0.5 = -0.20001$$

$$w33(n+1) = w33(n) + \alpha * w33(n-1) + \eta * \delta h3(n) * x3$$

$$= 0.3 + 0.0001 * -0.3 + 0.25 * 0.00024 * 0.5 = 0.30006$$

$$b8(n+1) = b8(n) + \alpha * b8(n-1) + \eta * \delta o1(n) * 1$$

$$= 0.3 + 0.0001 * 0.3 + 0.25 * 0.03581 * 1 = 0.30898$$

$$b7(n+1) = b7(n) + \alpha * b7(n-1) + \eta * \delta h7(n) * 1$$

$$= 0.2 + 0.0001 * 0.2 + 0.25 * 0.00269 * 1 = 0.20069$$

$$b6(n+1) = b6(n) + \alpha * b6(n-1) + \eta * \delta h6(n) * 1$$

$$= 0.1 + 0.0001 * 0.1 + 0.25 * 0.00163 * 1 = 0.10042$$

$$b5(n+1) = b5(n) + \alpha * b5(n-1) + \eta * \delta h5(n) * 1$$

$$= 0.2 + 0.0001 * 0.2 + 0.25 * 0.00062 * 1 = 0.20018$$

$$b4(n+1) = b4(n) + \alpha * b4(n-1) + \eta * \delta h4(n) * 1$$

$$= 0.3 + 0.0001 * 0.3 + 0.25 * 0.00135 * 1 = 0.30037$$

$$b3(n+1) = b3(n) + \alpha * b3(n-1) + \eta * \delta h3(n) * 1$$

$$= 0.2 + 0.0001 * 0.2 + 0.25 * 0.00024 * 1 = 0.20008$$

$$b2(n+1) = b2(n) + \alpha * b2(n-1) + \eta * \delta h2(n) * 1$$

$$= 0.1 + 0.0001 * 0.1 + 0.25 * 0.00011 * 1 = 0.10004$$

$$b1(n+1) = b1(n) + \alpha * b1(n-1) + \eta * \delta h1(n) * 1$$

$$= 0.1 + 0.0001 * 0.1 + 0.25 * 0.00018 * 1 = 0.10004$$

Now these new weights as calculated above are used to perform another (second) forward pass.

Second Forward Pass (First Iteration)

$$\boldsymbol{v1(n+1)} = 1 * b1(n+1) + x1 * w11(n+1) + x2 * w21(n+1) + x3 * w31(n+1)$$

$$= 1 * 0.10004 + 0.1 * (-0.20002) + 0.9 * 0.10004 + 0.5 * 0.10003 = 0.220089$$

$$\boldsymbol{y1(n+1)} = f(v1(n+1)) = f(0.220089) = \frac{1}{1+\exp(-0.220089)} = 0.55480$$

$$\boldsymbol{v2(n+1)} = 1 * b2(n+1) + x1 * w12(n+1) + x2 * w22(n+1) + x3 * w32(n+1)$$

$$= 1 * 0.10004 + 0.1 * (-0.10001) + 0.9 * 0.30005 + 0.5 * (-0.20001) = 0.260079$$

$$\boldsymbol{y2(n+1)} = f(v2(n+1)) = f(0.260079) = \frac{1}{1+\exp(-0.260079)} = 0.56466$$

$$\boldsymbol{v3(n+1)} = 1 * b3(n+1) + x1 * w13(n+1) + x2 * w23(n+1) + x3 * w33(n+1)$$

$$= 1 * 0.20008 + 0.1 * (-0.10000) + 0.9 * 0.40009 + 0.5 * 0.30006 = 0.700191$$

$$\boldsymbol{y3(n+1)} = f(v3(n+1)) = f(0.700191) = \frac{1}{1+\exp(-0.700191)} = 0.66823$$

$$\boldsymbol{v4}(\boldsymbol{n}+\boldsymbol{1}) = 1 * b4(n+1) + y1(n+1) * w14(n+1) + y2(n+1) * w24(n+1)$$
$$+y3(n+1) * w34(n+1)$$

$$= 1 * 0.30037 + 0.55480 * 0.20021 + 0.56466 * (-0.39985) + 0.66823 * 0.10024$$

$$= 0.25265$$

$$\boldsymbol{y4}(\boldsymbol{n}+\boldsymbol{1}) = f(v4(n+1)) = f(0.25265) = \frac{1}{1+\exp(-0.25265)} = 0.56283$$

$$\boldsymbol{v5}(\boldsymbol{n}+\boldsymbol{1}) = 1 * b5(n+1) + y1(n+1) * w15(n+1) + y2(n+1) * w25(n+1)$$
$$+y3(n+1) * w35(n+1)$$

$$= 1 * 0.20018 + 0.55480 * 0.30012 + 0.56466 * 0.10089 + 0.66823 * (-0.29993)$$

$$= 0.22323$$

$$\boldsymbol{y5}(\boldsymbol{n}+\boldsymbol{1}) = f(v5(n+1)) = f(0.22323) = \frac{1}{1+\exp(-0.22323)} = 0.55558$$

$$\boldsymbol{v6}(\boldsymbol{n}+\boldsymbol{1}) = 1 * b6(n+1) + y1(n+1) * w16(n+1) + y2(n+1) * w26(n+1)$$
$$+y3(n+1) * w36(n+1)$$

$$= 1 * 0.10042 + 0.55480 * 0.20025 + 0.56466 * 0.30026 + 0.66823 * 0.40031$$

$$= 0.64856$$

$$\boldsymbol{y6}(\boldsymbol{n}+\boldsymbol{1}) = f(v6(n+1)) = f(0.64856) = \frac{1}{1+\exp(-0.64856)} = 0.65669$$

$$\boldsymbol{v7}(\boldsymbol{n}+\boldsymbol{1}) = 1 * b7(n+1) + y1(n+1) * w17(n+1) + y2(n+1) * w27(n+1)$$
$$+y3(n+1) * w37(n+1)$$

$$= 1 * 0.20069 + 0.55480 * 0.10038 + 0.56466 * 0.20040 + 0.66823 * 0.20047$$

$$= 0.50350$$

$$\boldsymbol{y7(n+1)} = f(v7(n+1)) = f(0.50350) = \frac{1}{1+\exp(-0.50350)} = 0.62328$$

$$\boldsymbol{v8(n+1)} = 1 * b8(n+1) + y4(n+1) * w48(n+1) + y5(n+1) * w58(n+1)$$

$$+y6(n+1) * w68(n+1) + y7(n+1) * w78(n+1)$$

$$= 1 * 0.30898 + 0.56283 * 0.20506 + 0.55558 * 0.10498 + 0.65669 * 0.20590$$

$$+062328 * 0.30560 = 0.80841$$

$$\boldsymbol{y8(n+1)} = f(v8(n+1)) = f(0.80841) = \frac{1}{1+\exp(-0.80841)} = 0.69177$$

Please note the activation function, $f(v)$ used in the forward pass (feed-forward) and not its derivative $f'(v)$ which is used during the backward pass (back-propagation).

Therefore,

the error $(e(n+1))$ = target output $(d8)$ − actual output $(y8(n+1))$

$$= 0.9 - 0.69177 = 0.20823$$

Then after one complete forward and backward pass new weights and output have been obtained. The results are compared with the old inputs and output as shown in Table 3.5.

From the Table 3.5 the error reduced after the first forward and backward pass from $0.21299\ to\ 0.20823$. The forward and backward passes continue until the error becomes almost zero. The results obtained after a few more complete forward and backward passes are as shown in Table 3.5.

Table 3.5: Comparison of the results obtained after one complete forward and backward pass

S/N	Component	Old	New (After One Iteration)
1	$v1$	0.22000	0.22009
2	$y1$	0.55500	0.55480
3	$v2$	0.26000	0.26008
4	$y2$	0.56500	0.56466
5	$v3$	0.70000	0.70019
6	$y3$	0.66819	0.66823
7	$v4$	0.25182	0.25265
8	$y4$	0.56262	0.56283
9	$v5$	0.22254	0.22323
10	$y5$	0.55541	0.55558
11	$v6$	0.64778	0.64856
12	$y6$	0.65651	0.65669
13	$v7$	0.50088	0.50350
14	$y7$	0.62267	0.62328
15	$v8$	0.78617	0.80841
16	$y8$	0.68701	0.69177
	$e = d8 - y8$	$0.9 - 0.68701 = 0.21299$	$0.9 - 0.69177 = 0.20823$

After the second pass (iteration) $e = 0.20003$

After the third pass (iteration) $e = 0.16253$

After the fourth pass (iteration) $e = 0.14505$

After 50 passes (iteration) $e = 0.01053$

After 100 passes (iteration) $e = 0.00319$

After 300 passes (iteration) $e = 0.00038$

Error is getting reduced after each pass until it converges (target and actual output equal each other). So, this is how the neural networks were trained using back-propagation algorithm.

3.16 The Testing Process

After training as already mentioned in the section 3.15, the next important step to be performed before the application of neural networks is to test the trained neural network. Testing the artificial neural network is very important in order to make sure the trained network can generalize well and produce desired outputs when new data is presented to it.

There are three techniques that were used to test the performance of the trained networks, they are discussed in this section. One such technique was to plot the best linear regression fit between the actual neural network's outputs and the desired targets. Analyzing the slope of this line gives us an idea on the training process. Ideally the slope should be 1. Also, the correlation coefficient (r), of the outputs and the targets measures how well the ANN's outputs track the desired targets. The closer the value of 'r' is, to 1, the better the performance of the neural network. Another technique employed to test the neural network was to plot the confusion matrix and look at the actual number of cases that have been classified positively by the neural network. Ideally, if this percentage is a 100 it means there has been no confusion in the classification process. Hence, if the confusion matrix indicates very low positive classification rates, it indicates that the neural network might not perform well. The last and a very obvious means of testing the neural network was to present it with a whole new set of data with known inputs and targets and calculate the percentage error in the neural networks

output. If the average percentage error in the ANN's output is acceptable, the neural network has passed the test and can be readily applied for future use.

The Neural Network toolbox in Simulink by The MathWorks divides the entire set of data provided to it into three different sets namely the training set, validation set and the testing set. The training data set as indicated above is used to train the network by computing the gradient and updating the network weights. The validation set is provided to the network during the training process (just the inputs without the outputs) and the error in validation data set is monitored throughout the training process. When the network starts over fitting the data, the validation errors increase and when the number of validation fails increase beyond a particular value, the training process stops to avoid further over fitting the data and the network is returned at the minimum number of validation errors. The test set was not used during the training process but was used to test the performance of the trained network. If the test set reaches the minimum value of MSE at a significantly different iteration than the validation set, then the neural network will not be able to provide satisfactory performance.

CHAPTER FOUR

RESULTS AND DISCUSSION

4.1 Fault Identification

4.1.1 Simulation Results of Training the Fault Identification Neural Network

For illustration purposes, several neural networks (with varying number of hidden layers and neurons per hidden layer) that achieved satisfactory performance are shown and the best neural network has been described further in detail. Figures 4.1 – 4.2 show the error performance plots of neural networks with 2 and 1 hidden layers respectively. After extensive simulations, the desired network has five hidden layers with 8 neurons in the first hidden layer, 10 neurons in the second hidden layer, 20 neurons in the third hidden layer, 15 neurons in the fourth hidden layer and 6 neurons in the fifth hidden layer. The plot of the mean square error performance result of the desired or chosen network has been depicted in Fig 4.3 and the plot of various error performance results have been shown in Figures 4.4 – 4.7. Appendix C shows the Matlab codes for the various plots.

Fig 4.1 shows the plot of the training performance result of the neural network 3.15.10.1 (3neurons in the input layer, two hidden layers with fifteen and ten neurons in them respectively and one neuron in the output layer). The network did not achieve the desired Cross-Entropy goal of 1e-2 by the end of the training process, so it has been considered to not have been trained correctly.

Figure 4.1: Mean-square error performance of the network (3.15.10.1).

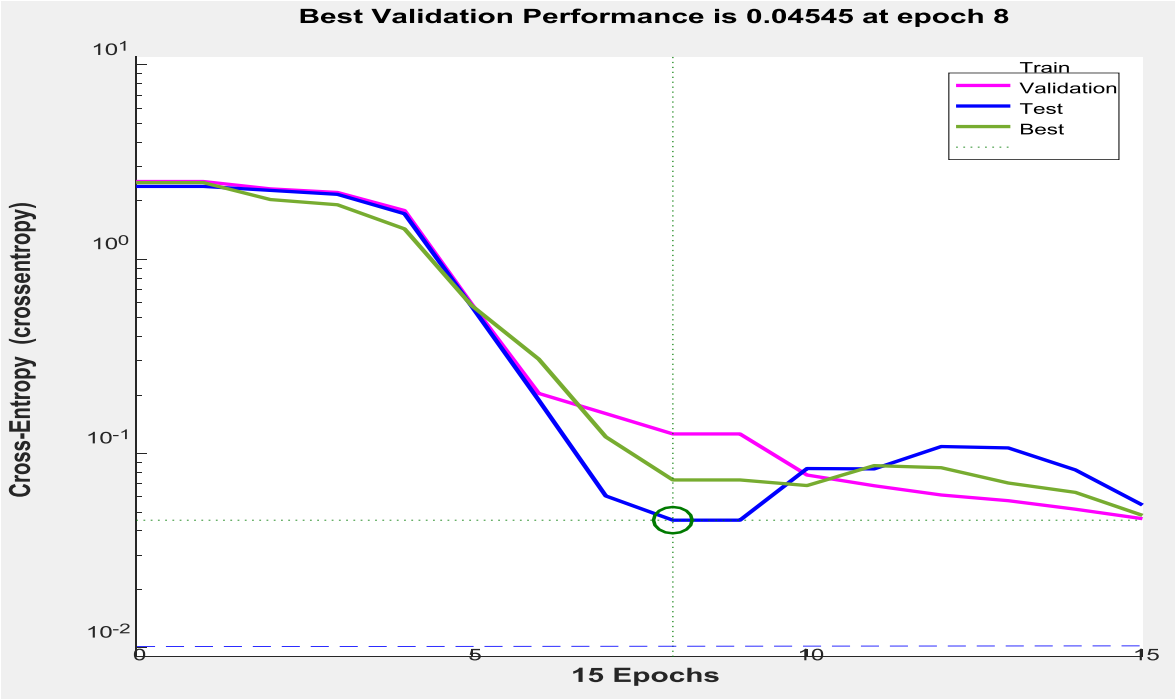


Figure 4.2: Mean-square error performance of the network (3.25.1).

Figure 4.2 shows the plot of training performance result of the neural network with 3.25.1 configuration (3 neurons in the input layer, one hidden layer with 25 neurons and one neuron in the output layer). It is to be noted that the neural network could not achieve the Cross-Entropy goal of 1e-2 by the end of the training process. Therefore, it did not train satisfactorily.

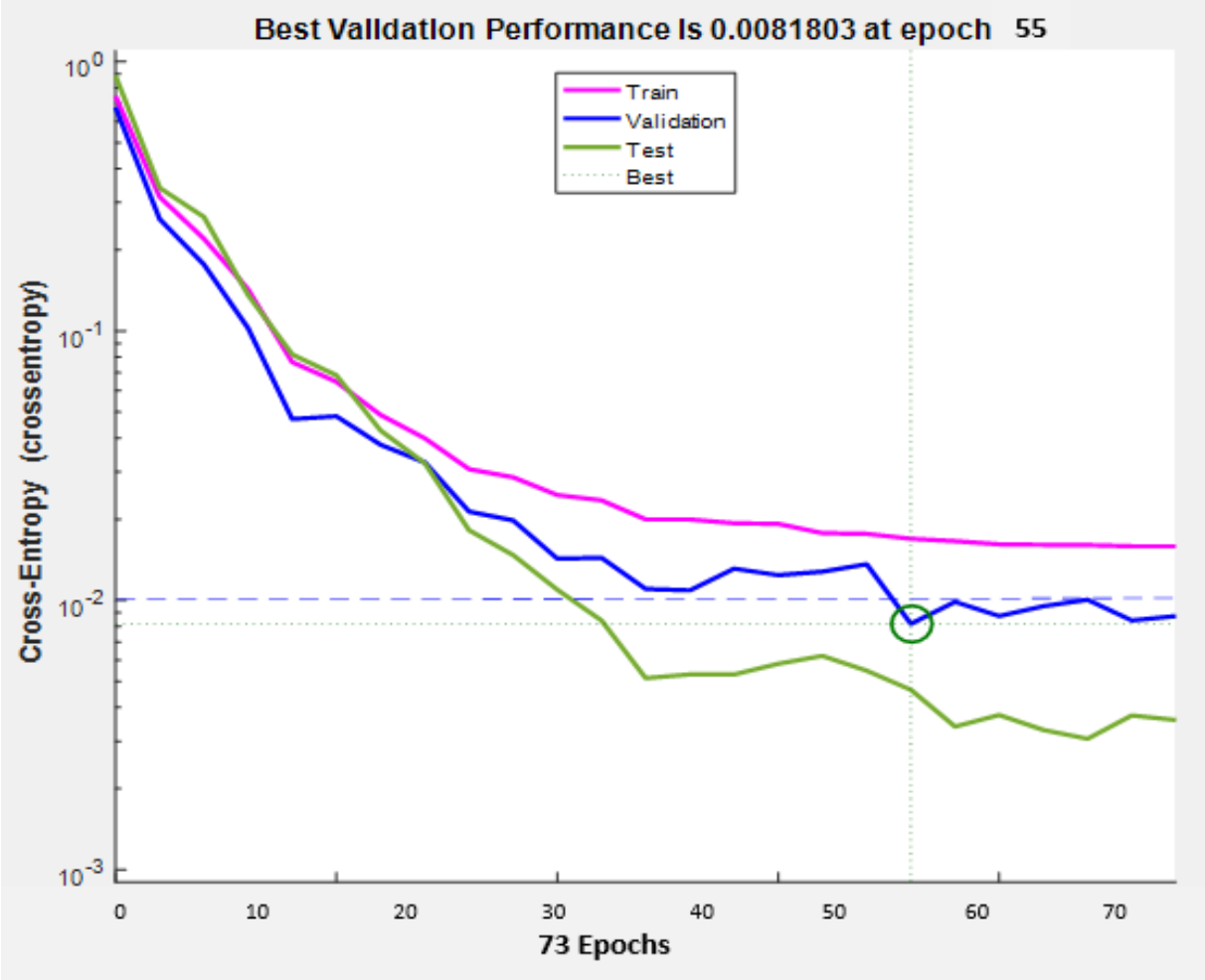


Figure 4.3: Mean-square error performance of the network (3.8.10.20.15.6.1).

Figure 4.3 shows the plot of training performance result of the neural network with (3.8.10.20.15.6.1) configuration (3 neurons in the input layer, 5 hidden layers with 8, 10, 20,

15 and 6 neurons in them respectively and one neuron in the output layer). From the training performance plots of Figure 4.3, it is to be noted that very satisfactory training performance has been achieved by the neural network with the (3.8.10.20.15.6.1) configuration (3 neurons in the input layer, 5 hidden layers with 8, 10, 20, 15 and 6 neurons in them respectively and one neuron in the output layer). The overall Cross-Entropy of the trained neural network is way below the value of 1e-2 and is actually8.18036e-3 by the end of the training process. Hence this has been chosen as the ideal ANN for the purpose of fault detection.

4.1.2 Simulation Results of Testing the Fault Identification Neural Network

Once the neural network has been trained, its performance has to be tested by three different factors. The first of these is by plotting the best linear regression that relates the targets to the outputs as shown in Fig 4.4.

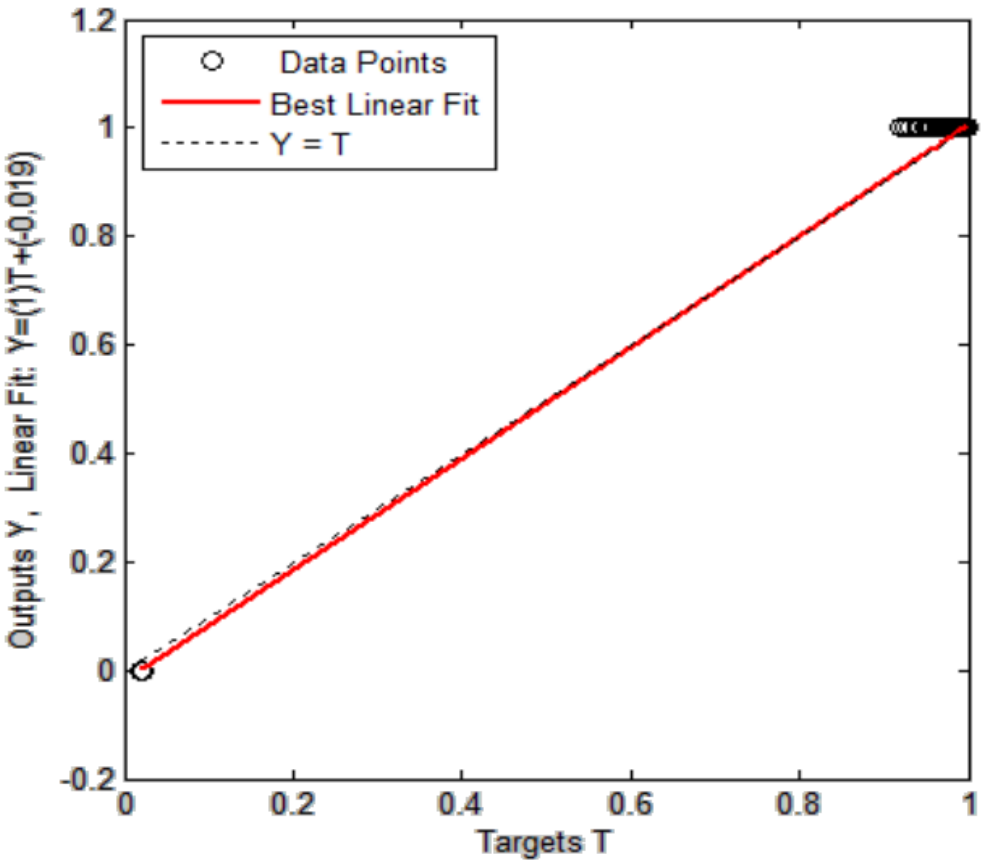


Figure 4.4: Regression fit of the outputs vs. targets for the network (3.8.10.20.15.6.1).

The correlation coefficient (r) is a measure of how well the neural network's targets can track the variations in the outputs (0 being no correlation at all and 1 being complete correlation). The correlation coefficient in this case has been found to be0.99967 which indicates excellent correlation.

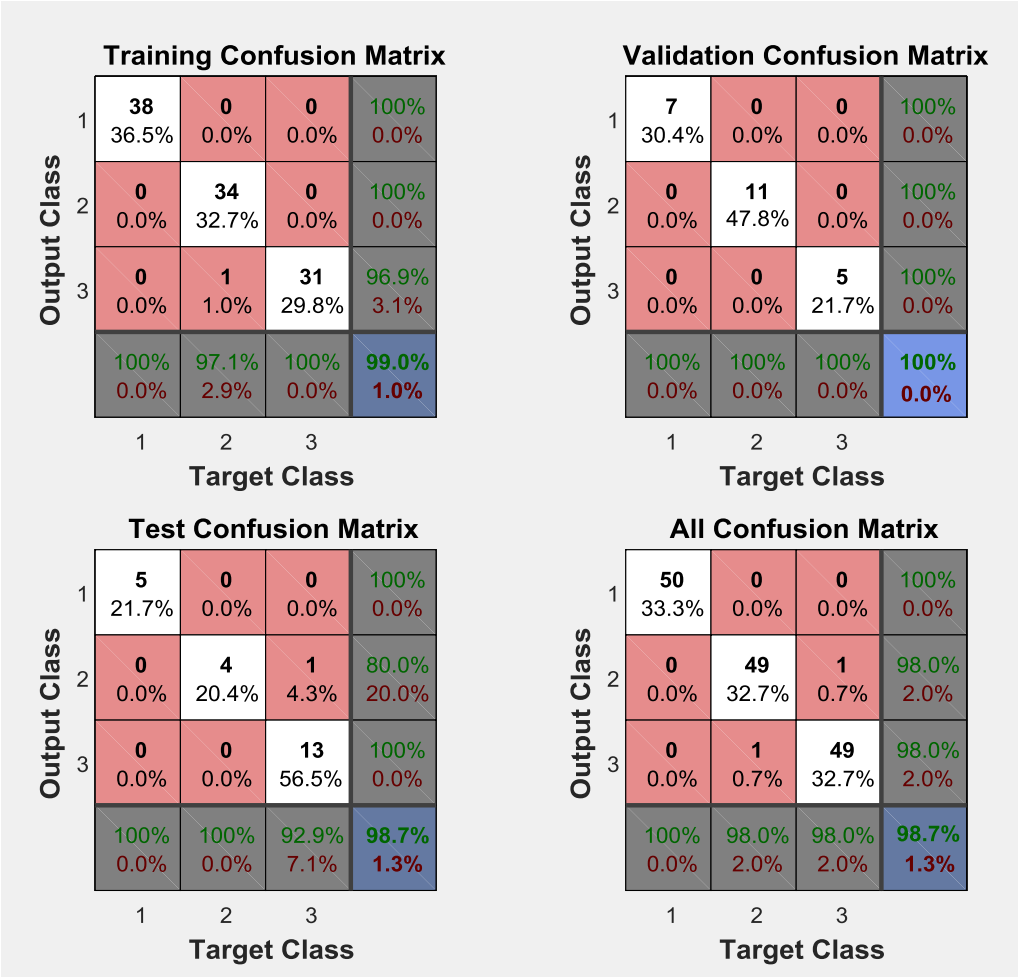


Figure 4.5: Confusion matrices for Training, Testing and Validation Phases.

Another means of testing the performance of the neural network is to plot the confusion matrices for the various types of errors that occurred for the trained neural network. Fig 4.5 plots the confusion matrix for the three phases of training, testing and validation. The

diagonal cells in white colour indicate the number of cases that have been classified correctly by the neural network and the off-diagonal cells which are in pink indicate the number of cases that have been wrongly classified by the ANN. The last cell in blue in each of the matrices indicates the total percentage of cases that have been classified correctly in green and the vice-versa in red. It can be seen that the chosen neural network has 98.7% accuracy in fault identification.

After the test set has been fed into the neural network and the results obtained, it was noted that the efficiency of the neural network in terms of its ability to detect the occurrence of a fault is a 98.7%. Hence the neural network can, with utmost accuracy, differentiate a normal situation from a fault condition on a transmission line.

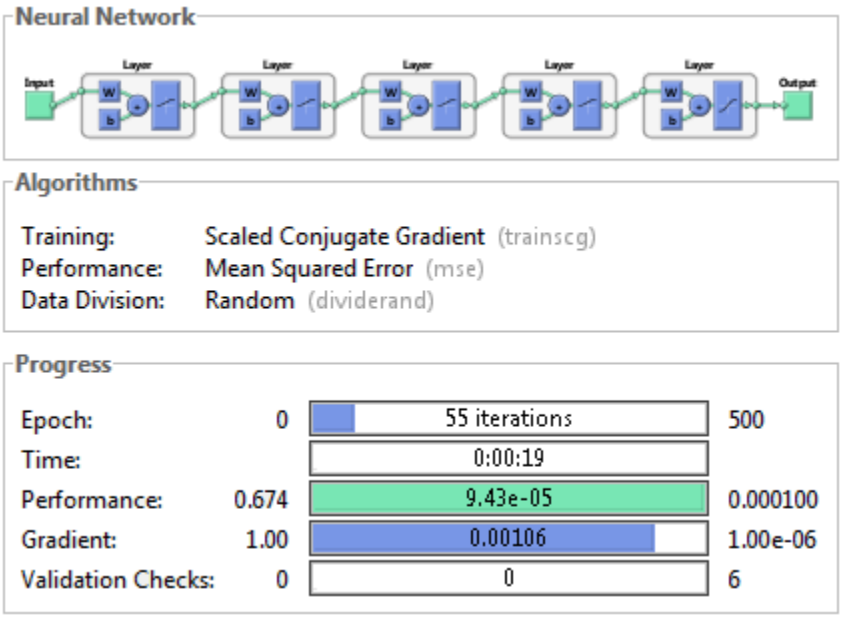


Figure 4.6: Overview of the ANN (3.8.10.20.15.6.1) chosen for fault detection.

Figure 4.6 presents a snapshot of the trained ANN with the (3.8.10.20.15.6.1) configuration and it is to be noted that the number of iterations required for the training process were 55. It can be seen that the mean square error in fault detection achieved by the end of the training process was 9.43e-5 and that the number of validation check fails were zero by the end of the training process.

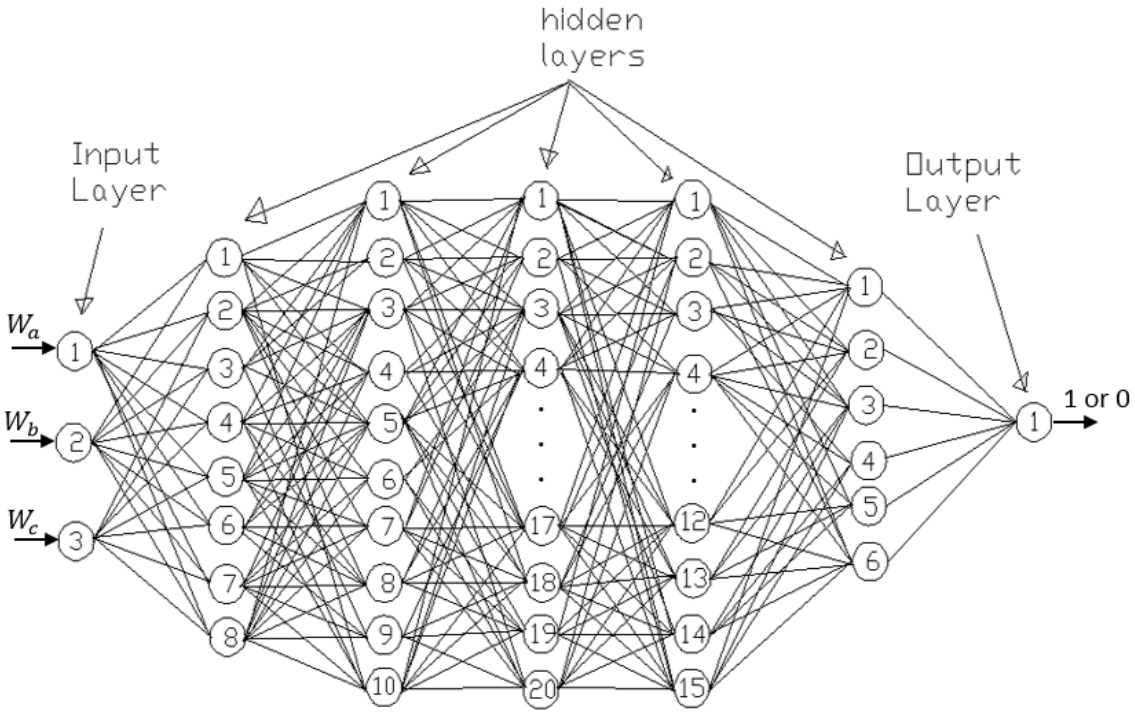


Figure 4.7: Chosen ANN for Fault Detection(3.8.10.20.15.6.1)

The structure of the chosen neural network for fault detection is shown in Fig 4.7 with the input layer, hidden layers and the output layer labelled. It is to be noted that there are 3 neurons in the input layer, 5 hidden layers with 8, 10, 20, 15 and 6 neurons in them respectively and one neuron in the output layer. This is a pictorial representation of how the neurons in the respective layers are connected together through the synaptic weights. It shows the interconnections between the input layer and the hidden layers, and also between the

hidden layers and the output layer. Any given neuron Figure 4.7 is connected to all the neurons in the layer in front.

4.2 Fault Classification

Once a fault has been detected on the power line, the next step is to identify the type of fault. This section presents the results of the fault classification phase using neural networks. The results of the different neural networks that were trained and tested are provided which is followed by that of the chosen network.

4.2.1 Simulation Results of Training the Fault Classifier Neural Network

Back-propagation networks with a variety of combinations of hidden layers and the number of neurons per hidden layer have been analyzed. Of these, the ones that achieved satisfactory performance are shown followed by the best neural network which has been described further in detail. Figures 4.8 – 4.12 show the error performance plots of neural networks with 3, 2, 2, 1 and 2 hidden layers respectively. After extensive simulations, the desired network has three hidden layers with 12 neurons in the first hidden layer, 35 neurons in the second hidden layer, 24 neurons in the third hidden layer (3.12.35.24.4). The chosen network has been depicted in Fig 4.13 and the various error performance plots have been shown in Figures 4.14 – 4.17. Appendix D shows the Matlab codes for the various plots.

Fig 4.8 shows the training performance plot of the neural network 3.10.5.25.4 (3 neurons in the input layer, 3 hidden layers with 10, 5 and 25 neurons in them respectively and four neurons in the output layer). It can be seen that the best validation performance in terms of the

Cross-Entropy by the end of the training process is 2.4416e-2 which is greater than the desired Cross-Entropy of 1e-2. So the network did not train correctly.

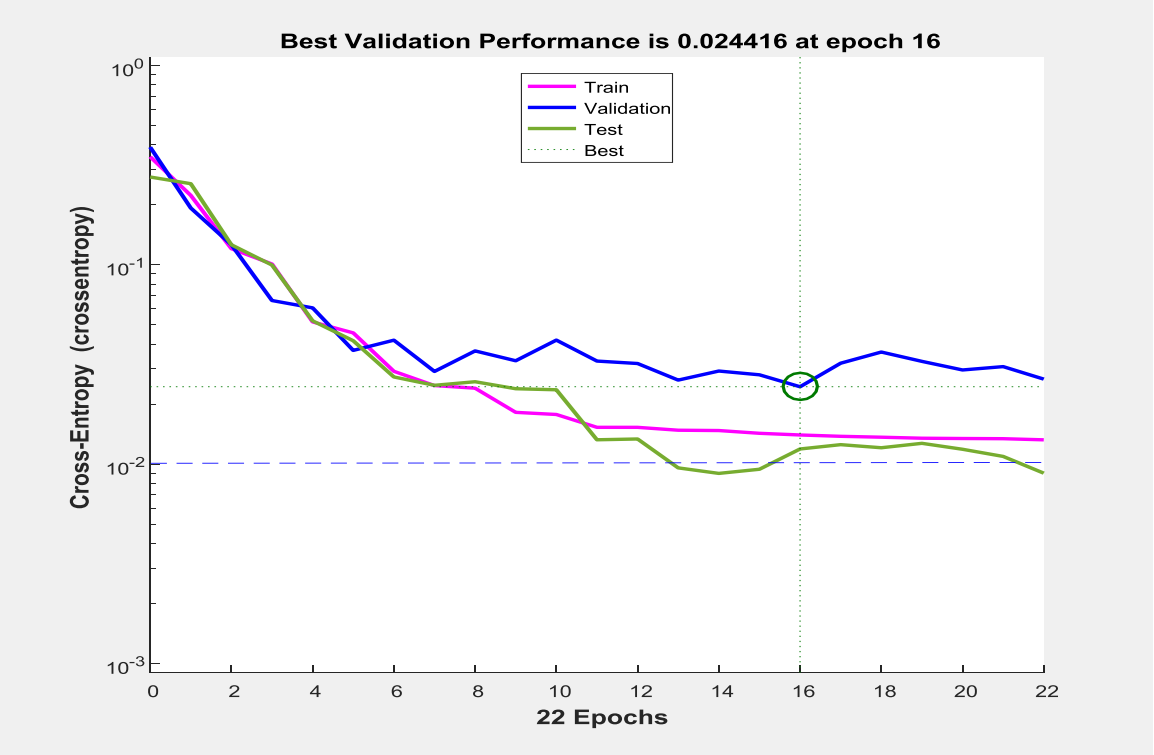


Figure 4.8: Mean-square error performance of the network with configuration (3.10.5.25.4).

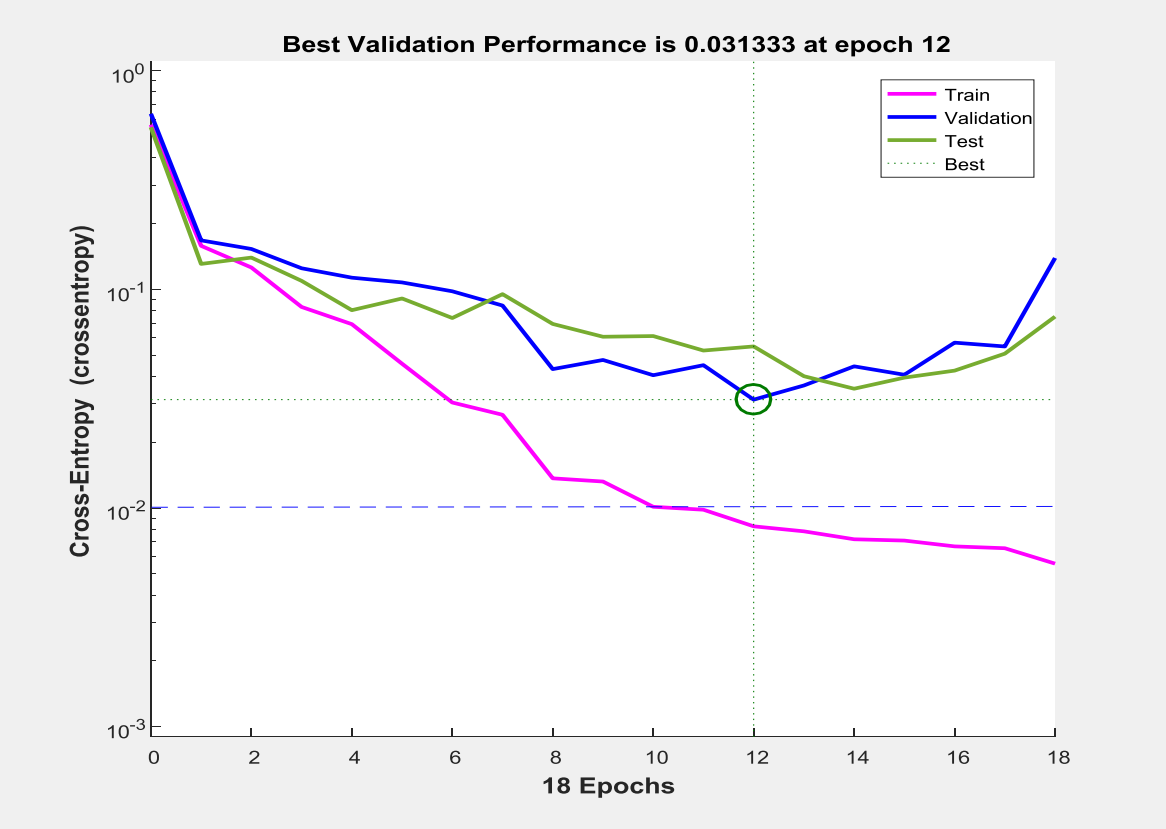


Figure 4.9: Mean-square error performance of the network with configuration (3.20.10.4).

Figure 4.9 shows the training performance plot of the neural network 3.20.10.4 (3 neurons in the input layer, 2 hidden layers with 20 and 10 neurons in them respectively and four neurons in the output layer). It can be seen that the best validation performance in terms of the Cross-Entropy by the end of the training process is 3.1333e-2. Hence, training was not correctly done.

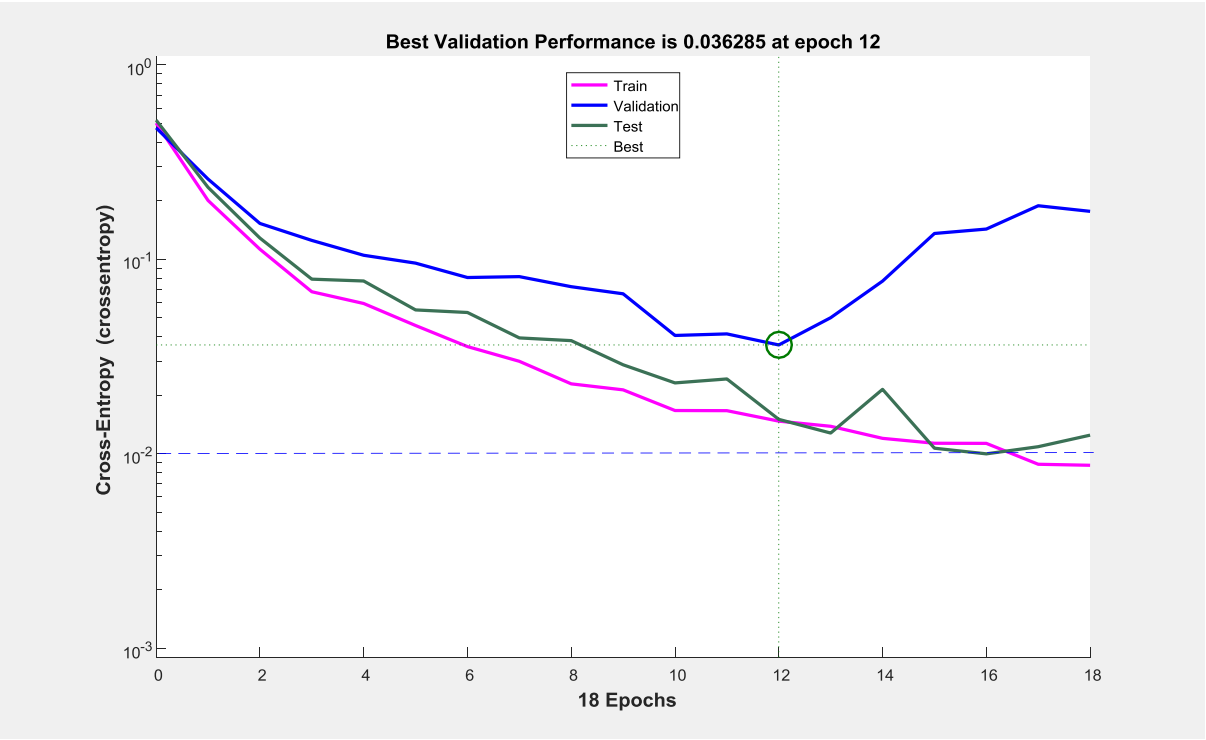


Figure 4.10: Mean-square error performance of the network with configuration (3.10.5.4).

Figure 4.10 shows the training performance plot of the neural network 3.10.5.4 (3 neurons in the input layer, 2 hidden layers with 10 and 5 neurons in them respectively and four neurons in the output layer). It can be seen that the best validation performance in terms of the Cross-Entropy by the end of the training process in this case is 3.6285e-2. Training was not successful.

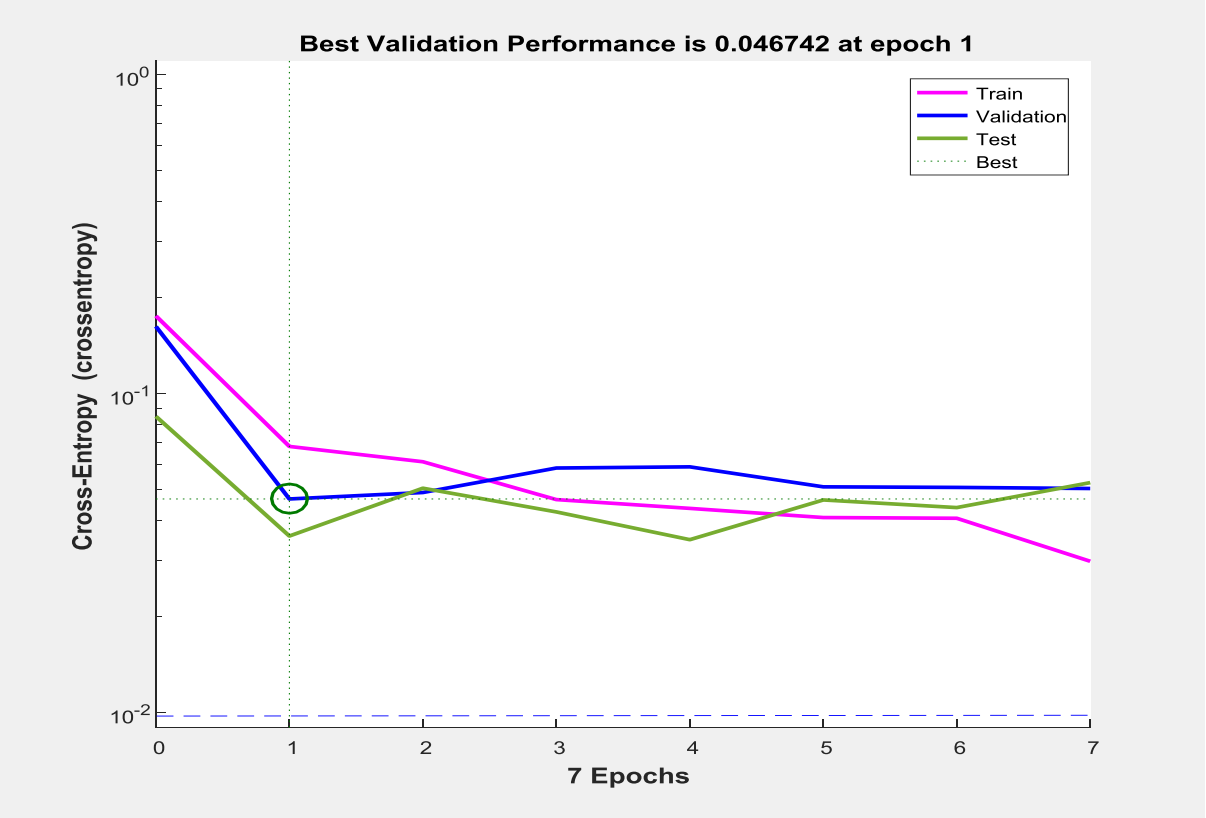


Figure 4.11: Mean-square error performance of the network with configuration (3.20.4).

Figure 4.11 shows the training performance plot of the neural network 3.20.4 (3 neurons in the input layer, 1 hidden layer with 20 neurons in it and four neurons in the output layer). It can be seen that the best validation performance in terms of the Cross-Entropy by the end of the training process in this case is 4.6742e-2. The network failed to train correctly.

Figure 4.12 shows the training performance plot of the neural network 3.10.5.4 (3neurons in the input layer, two hidden layers with 10 and 5 neurons in them respectively and four neurons in the output layer). It can be seen that the best validation performance in terms of the Cross-Entropy by the end of the training process in this case almost hit the Cross-Entropy goal of 1e-2 and is 1.119e-2. But then it did not train correctly.

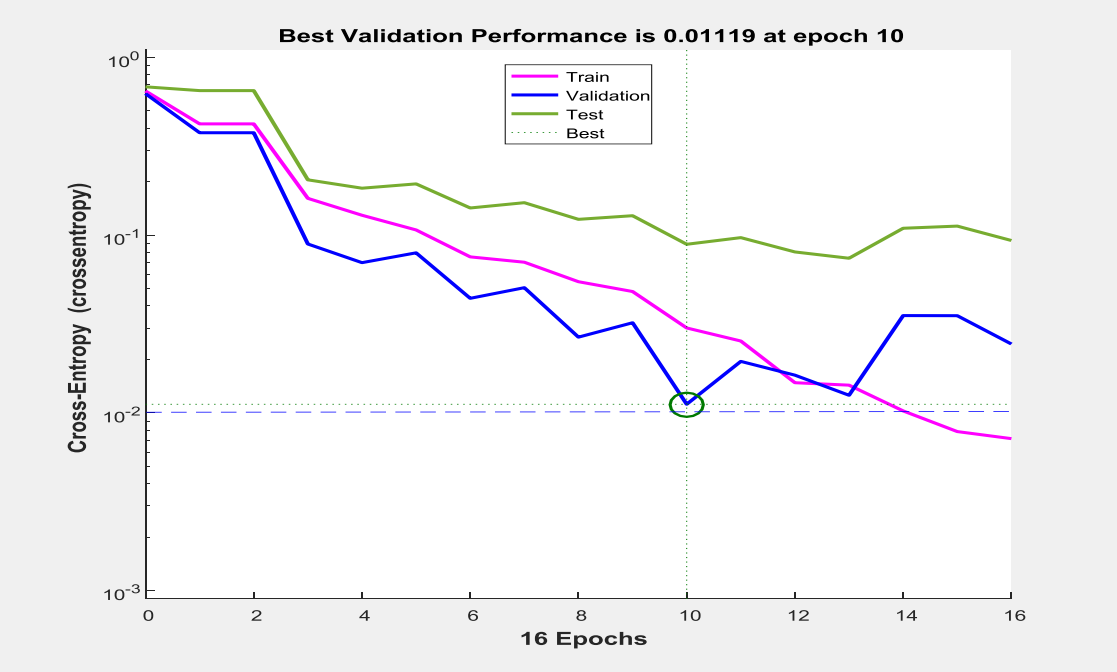


Figure 4.12: Mean-square error performance of the network with configuration (3.10.5.4).

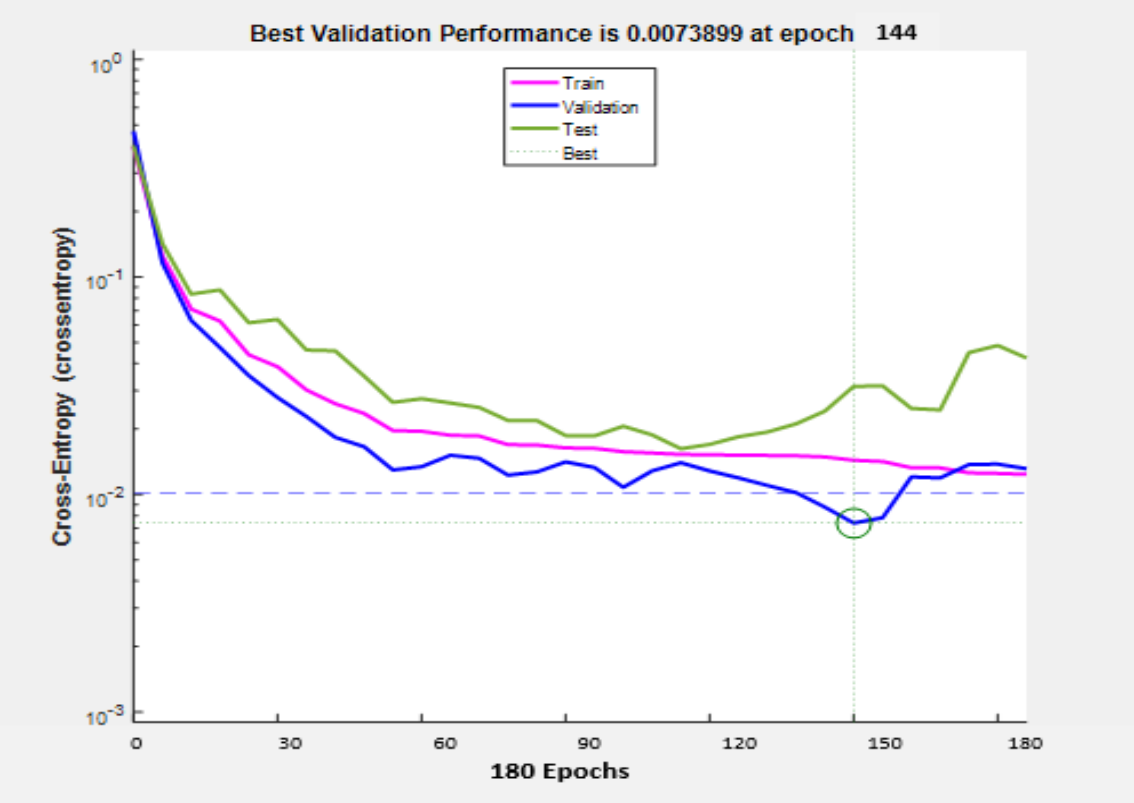


Figure 4.13: Mean-square error performance of the network with configuration (3.12.35.24.4).

Figure 4.13 shows the training performance plot of the neural network 3.12.35.24.4 (3 neurons in the input layer, 3 hidden layers with 12, 35 and 24 neurons in it respectively and four neurons in the output layer). It can be seen that the best validation performance in terms of the Cross-Entropy by the end of the training process in this case is 7.3899e-3 which is below the Cross-Entropy goal of 1e-2. Hence, the training was successful.

From the above training performance plots, it is to be noted that satisfactory training performance has been achieved by the neural network with the 3.12.35.24.4 configuration (3 neurons in the input layer, 12, 35 and 24 neurons in the hidden layers respectively and four neurons in the output layer). The overall Cross-Entropy of the trained neural network is 7.3899e-3 and it can be seen from Figure4.13 that the testing and the validation curves have similar characteristics which is an indication of efficient training. Hence this has been chosen as the ideal ANN for the purpose of fault classification.

4.2.2 Simulation Results of Testing the Fault Classifier Neural Network

Once the neural network has been trained, its performance has been tested by taking three different factors into consideration. The first of these is by plotting the best linear regression that relates the targets to the outputs as shown in Figure 4.14. The correlation coefficient in this case was found to be 0.98108 which indicates satisfactory correlation between the targets and the outputs. The dotted line in the figure indicates the ideal regression fit and the red solid line indicates the actual fit of the neural network. It can be seen that both lines track each other very closely which is an indication of very good performance by the neural network.

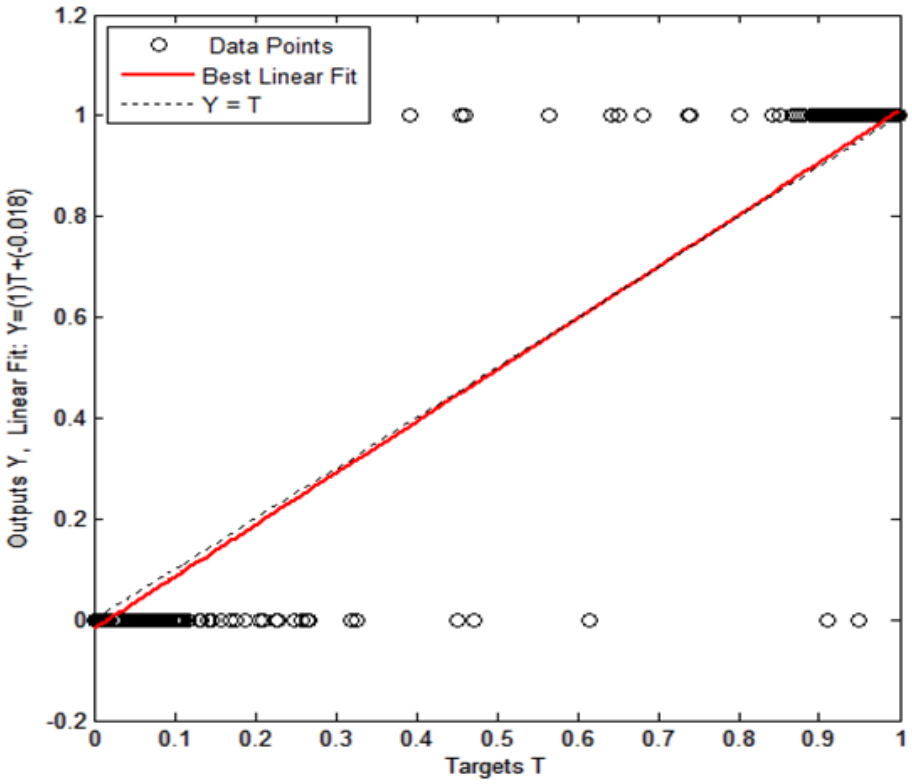


Figure 4.14: Regression fit of the Outputs vs. Targets of ANN with configuration (3.12.35.24.4).

The second approach of testing the performance of the neural network is to plot the confusion matrices for the various types of errors that occurred for the trained neural network. Figure 4.15 plots the confusion matrix for the three phases of training, testing and validation. The diagonal cells in white colour indicate the number of cases that have been classified correctly by the neural network and the off-diagonal cells which are in pink indicate the number of cases that have been wrongly classified by the ANN. The last cell in blue in each of the matrices indicates the total percentage of cases that have been classified correctly in green and the vice-versa in red. It can be seen that the chosen neural network has 98.0% accuracy in fault classification.

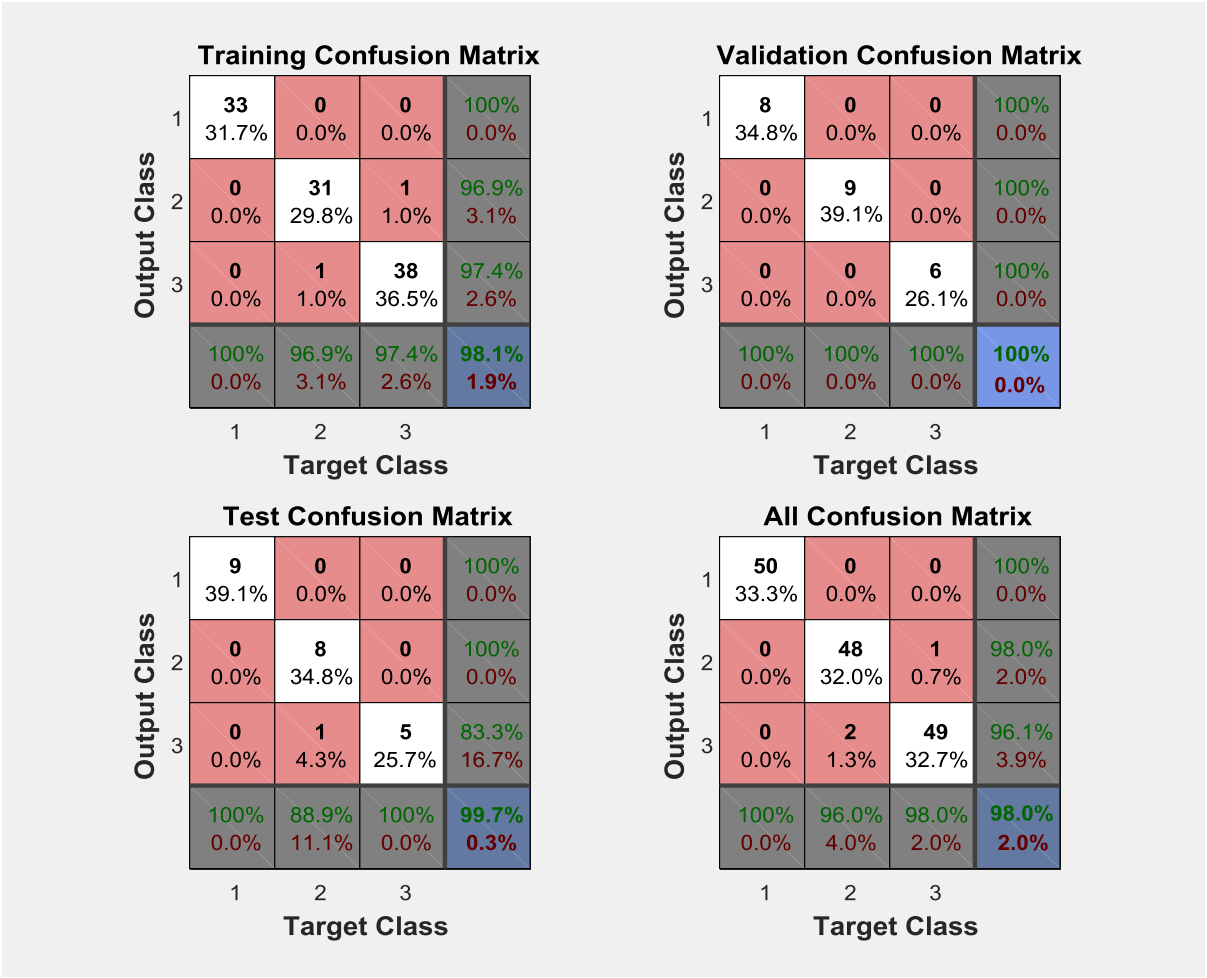


Figure 4.15: Confusion matrices for Training, Testing and Validation Phases of the ANN with configuration (3.12.35.24.4).

The third step in the testing process is to create a separate set of data called the test set to analyze the performance of the trained neural network. A total of 300 different test cases have been simulated with 200 cases corresponding to different types of faults (about 20 cases for each of the ten faults where the fault resistance and the fault location have been varied in each case). The rest of the 100 cases correspond to the no-fault situation.

After the test set has been fed into the neural network and the results obtained, it was noted that the efficiency of the neural network in terms of its ability to identify the type of the fault is a 99 percent. Hence the neural network can, with utmost accuracy, differentiate between the ten possible types of faults on a transmission line.

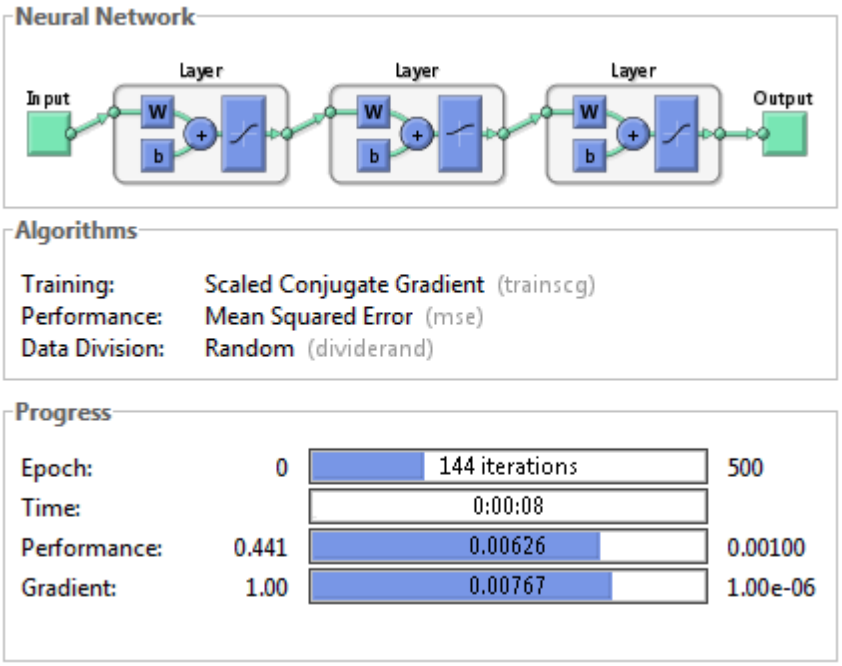


Figure 4.16: Overview of the ANN with configuration (3.12.35.24.4), chosen as fault classifier.

Figure 4.16 provides an overview on the neural network and is a screen shot of the training window simulated using the Artificial Neural Network Toolbox in Simulink. Important things to be noted are that the training process converged in about 144 iterations and that the performance in terms of mean square error achieved by the end of the training process was 6.26e-3.

Figure 4.17 shows the structure of the chosen ANN for the purpose of fault classification and the neural network has 3 neurons in the input layer, 12, 35 and 24 neurons in the hidden layers respectively and four neurons in the output layer as shown. Each of the neurons in the output layer would indicate the fault condition on each of the three phases (A, B and C) and the fourth neuron is to identify if the fault is a ground fault. An output of 0 corresponds to no fault while an output of 1 indicates that the phase is faulted and the combinations give the fault type.

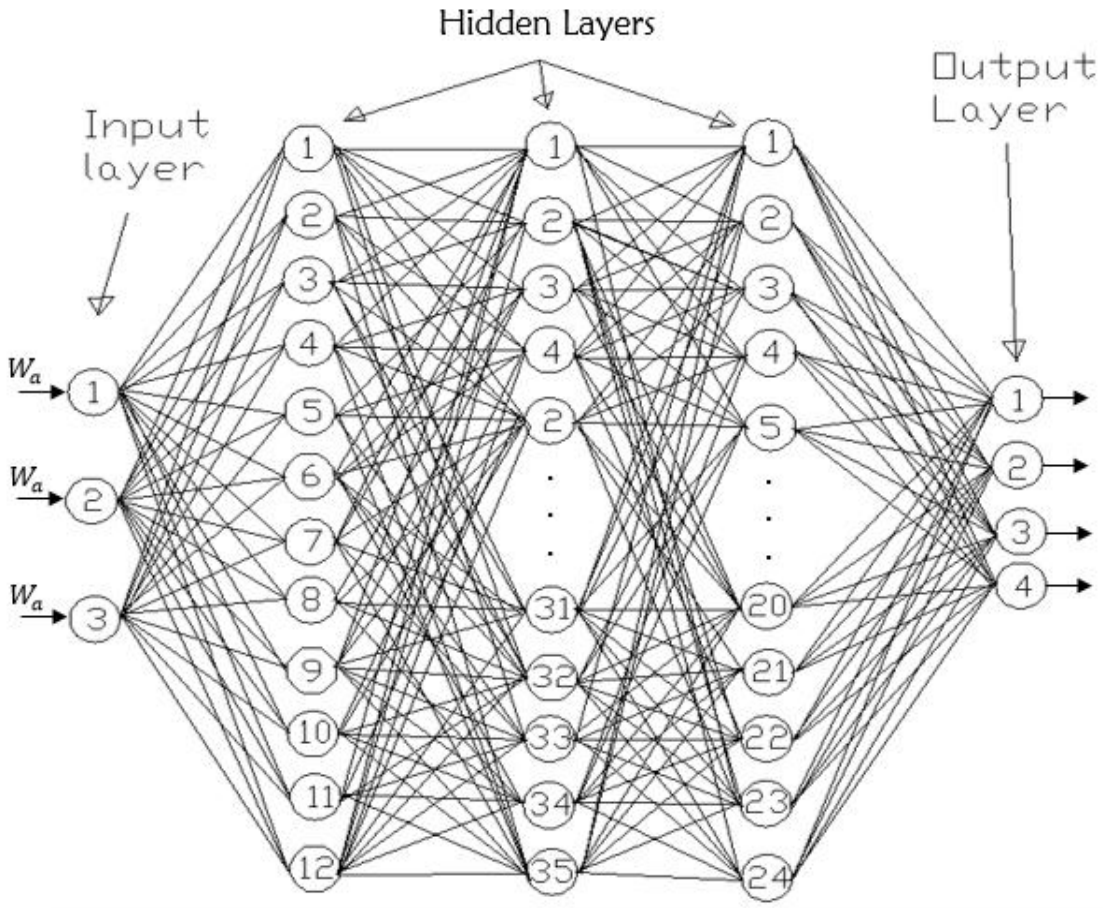


Figure 4.17: Chosen ANN for Fault Classification (3.12.35.24.4).

4.3 Fault Location

This section shows the results of the design, development and the implementation of the neural network based fault locators for each of the various types of faults. This forms the third step in the entire process of fault location after the inception of the fault. The following subsections deal with the various kinds of faults and their error performances individually.

4.3.1 Single Line – Ground Faults

Now that the occurrence of a fault on a transmission line can be detected and also classified into various fault categories, the next step is to pin-point the location of the fault from either ends of the transmission line. Three possible single line – ground faults exist (A-G, B-G, C-G), corresponding to each of the three phases (A, B or C) being faulted.

4.3.1a Simulation Results of Training the Neural Network for Single Line – Ground Fault Location

Firstly, a few of the various neural networks (with varying combination of hidden layers and number of neurons per hidden layer) that performed reasonably well are presented along with their respective error performances and then the chosen neural network is shown with all its characteristics depicted in detail. Efficiency of each of the trained networks is analyzed based on their regression performance and their performance in the testing phase. The test performance plots are obtained by simulating various faults on different phases at varying locations and calculating the error in the output produced by the Neural Network. Figures 4.18 – 4.23 show the error performance and regression plots of neural networks with 2, 2 and 2 hidden layers. The chosen neural network, after extensive simulations has three hidden layers with 8 neurons in the first hidden layer, 10 neurons in the second hidden layer, 20 neurons in the third hidden layer (3.8.10.20.1).The chosen network has been depicted in Figure 4.24 and its various error performance plots have been shown in Figures 4.25 – 4.30. Appendix E shows the Matlab codes for the various plots.

Figure 4.18 plots the best linear regression fit between the outputs and the targets of the neural network with 3 neurons in the input layer, 2 hidden layers with 25 and 15 neurons in them respectively and 1 neuron in the output layer (3.25.15.1). The correlation coefficient (r) as mentioned earlier is a measure of how well the neural network relates the outputs and the

targets. The closer the value of r is to 1, the better the performance of the neural network. The value of r in this case is found to be 0.89799. In order to test the performance of this network, 12 different single phase faults have been simulated on different phases with the fault distance being incremented by 25km in each case and the percentage error in calculated output has been presented.

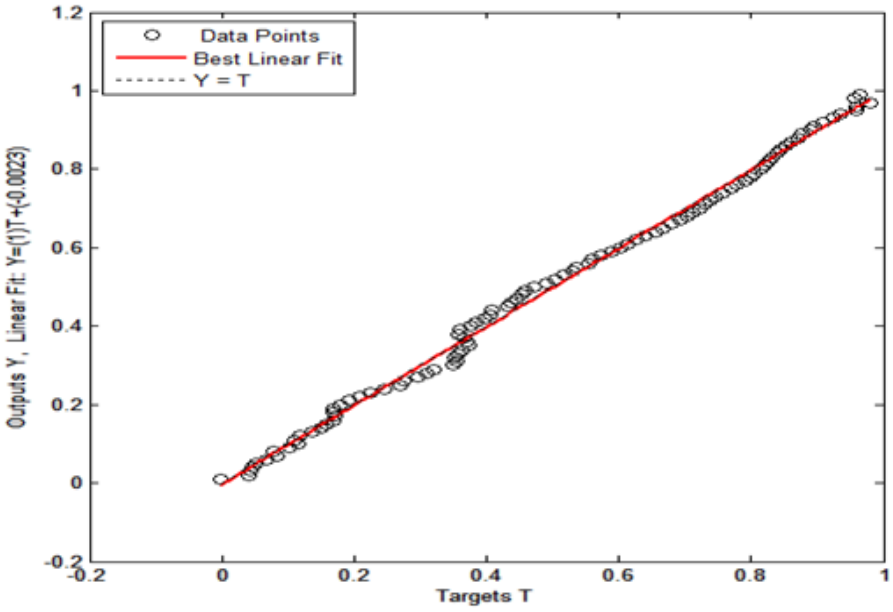


Figure 4.18: Regression fit of the Outputs vs. Targets with configuration (3.25.15.1).

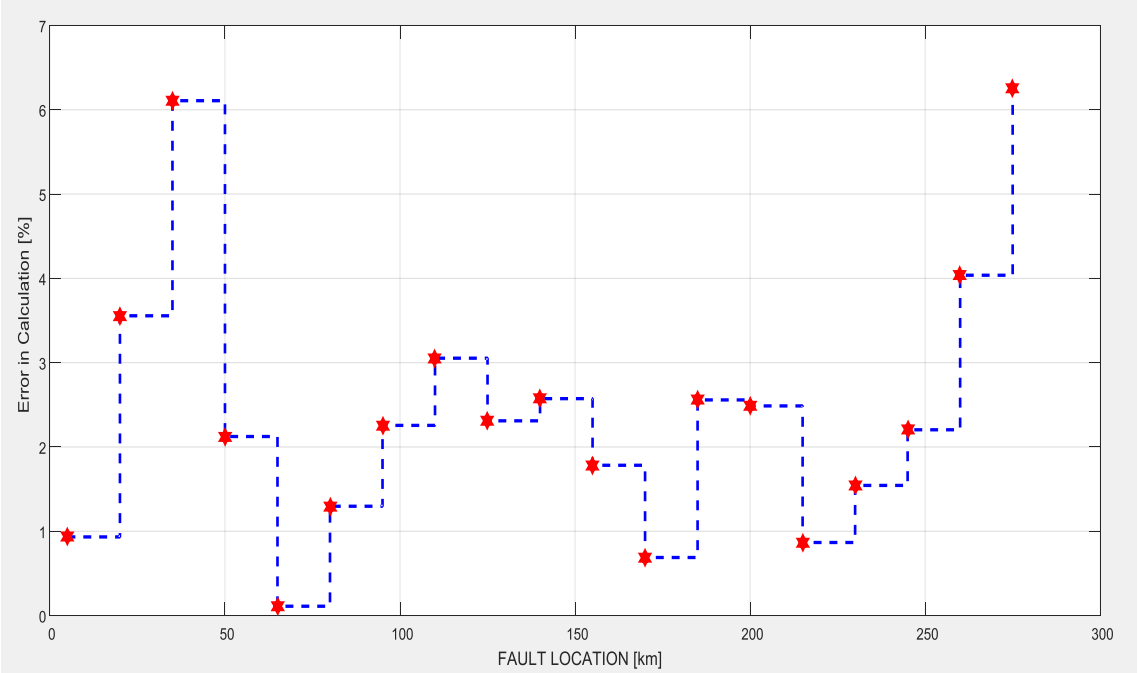


Figure 4.19: Test Phase performance of the Neural Network with configuration (3.25.15.1).

Figure 4.19 shows the results of this test conducted on the neural network (3.25.15.1). The maximum error is almost 6.28% which is not acceptable.

Figure 4.20 plots the best linear regression fit between the outputs and the targets of the neural network with 3 neurons in the input layer, 5 and 20 neurons in the hidden layers respectively and 1 neuron in the output layer (3.5.20.1). The value of the correlation coefficient r in this case is found to be 0.9959. In order to test the performance of this network, 19 different single phase faults have been simulated on different phases with the fault distance being incremented by 15km in each case and the percentage error in calculated output has been presented. Fig 4.21 shows the plot of results of this test conducted on the neural network (3.5.20.1). The maximum error is around 5.58% which is not very satisfactory.

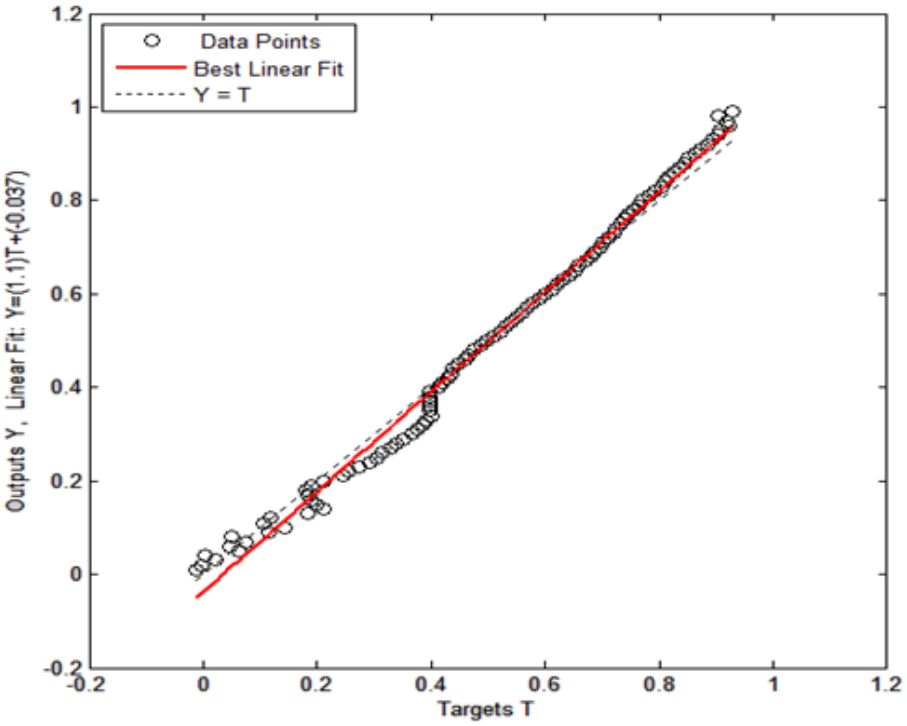


Figure 4.20: Regression fit of the outputs versus targets with configuration (3.5.20.1).

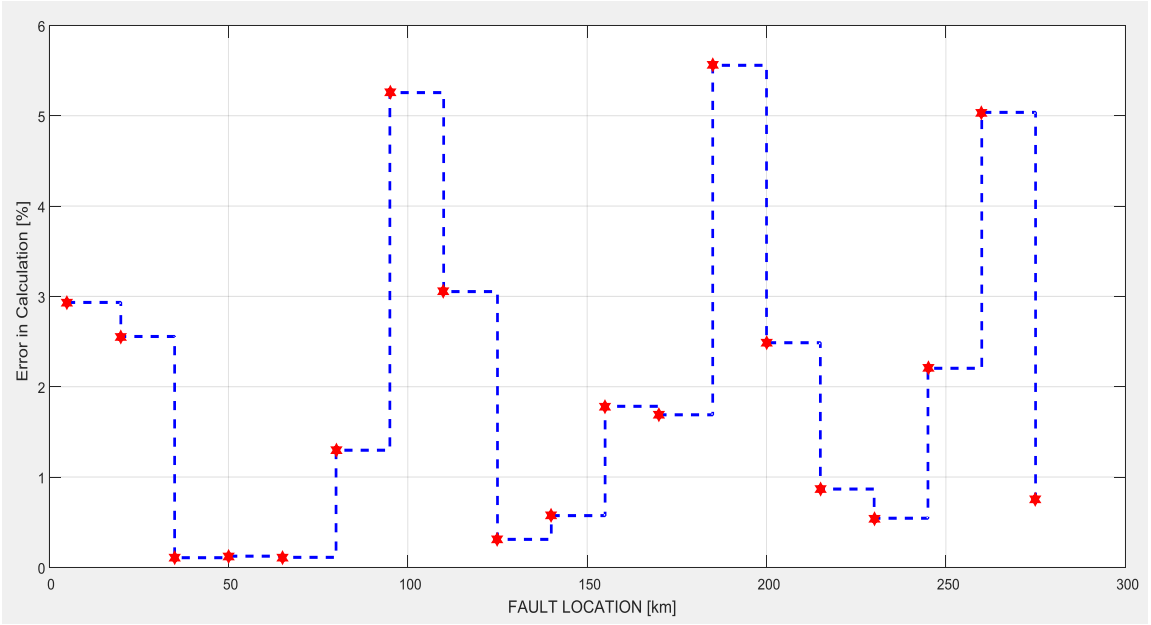


Figure 4.21: Test phase performance of the ANN with configuration (3.5.20.1)

Figure 4.22 plots the best linear regression fit between the outputs and the targets of the neural network with 6 neurons in the input layer, 12 and 25 neurons in the hidden layers respectively and 1 neuron in the output layer (3.12.25.1). The value of the correlation coefficient r in this case is found to be 0.89906.

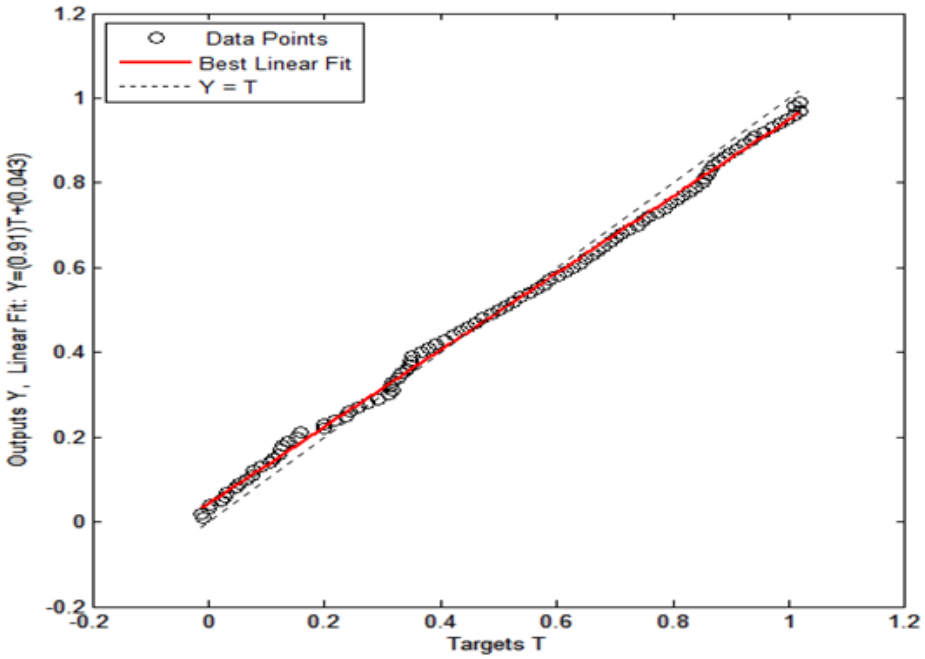


Figure 4.22: Regression fit of the outputs versus targets with configuration (3.12.25.1).

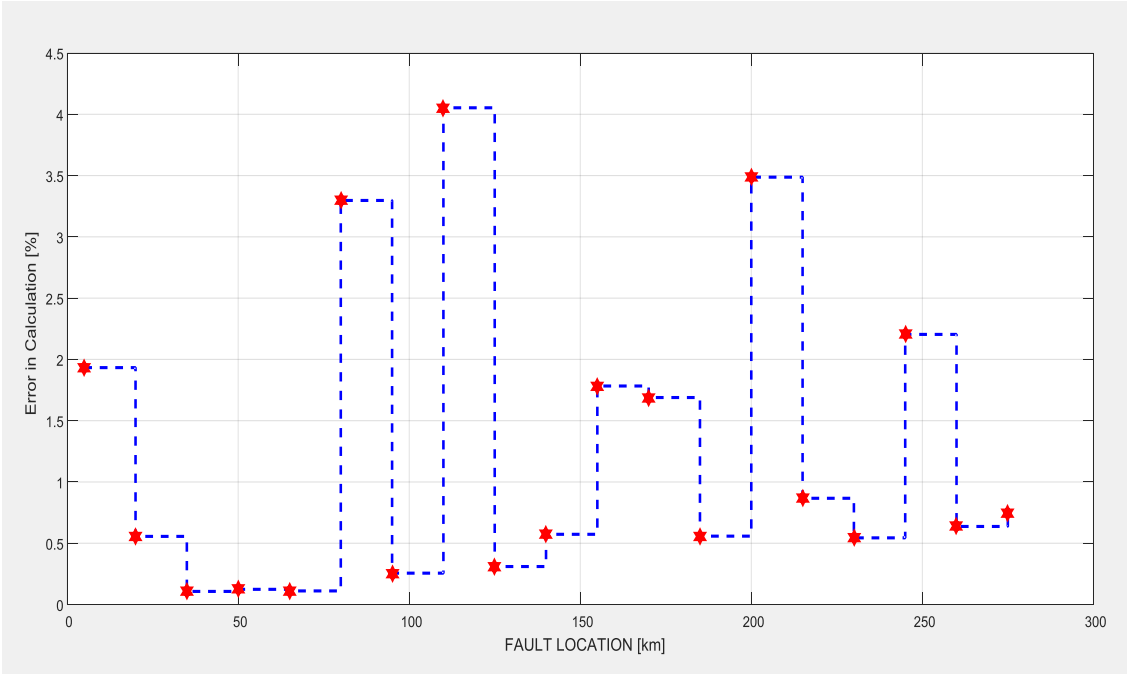


Figure 4.23: Test phase performance of the neural network with configuration (3.12.25.1).

In order to test the performance of this network, 19 different single phase faults have been simulated on different phases with the fault distance being incremented by 15km in each case and the percentage error in calculated output has been presented. Fig 4.23 shows the results of this test conducted on the neural network (3.12.25.1). The maximum error is around 4.07% which is still not satisfactory.

Figure 4.24 plots the best linear regression fit between the outputs and the targets of the neural network with 3 neurons in the input layer, 8, 10 and 20 neurons in the 3 hidden layers respectively and 1 neuron in the output layer (3.8.10.20.1). The value of the correlation coefficient r in this case is found to be 0.99924 which is by far the best and the closest to one and the network has been chosen as the best trained network.

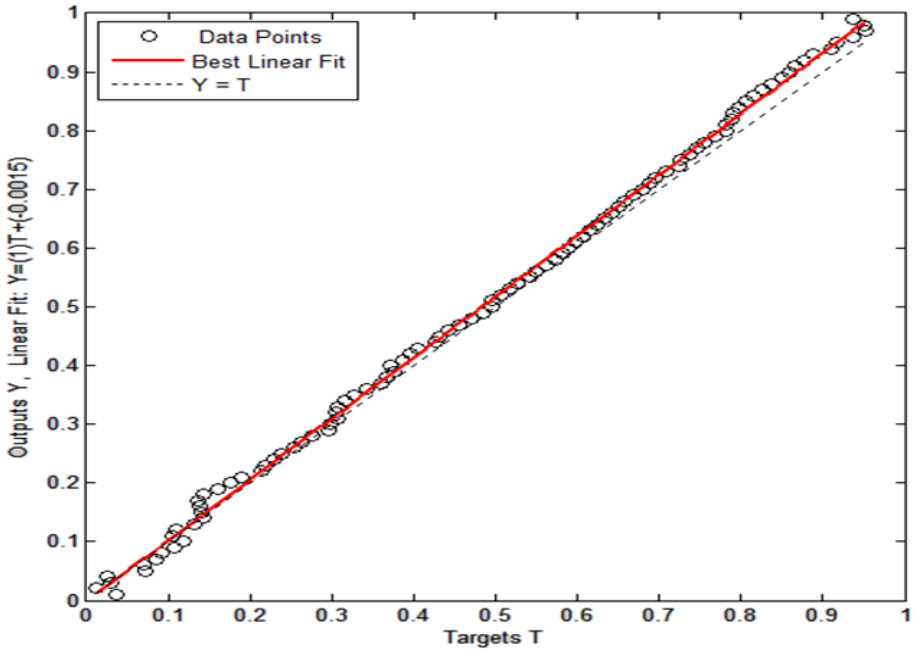


Figure 4.24: Regression fit of the outputs versus targets with configuration (3.8.10.20.1).

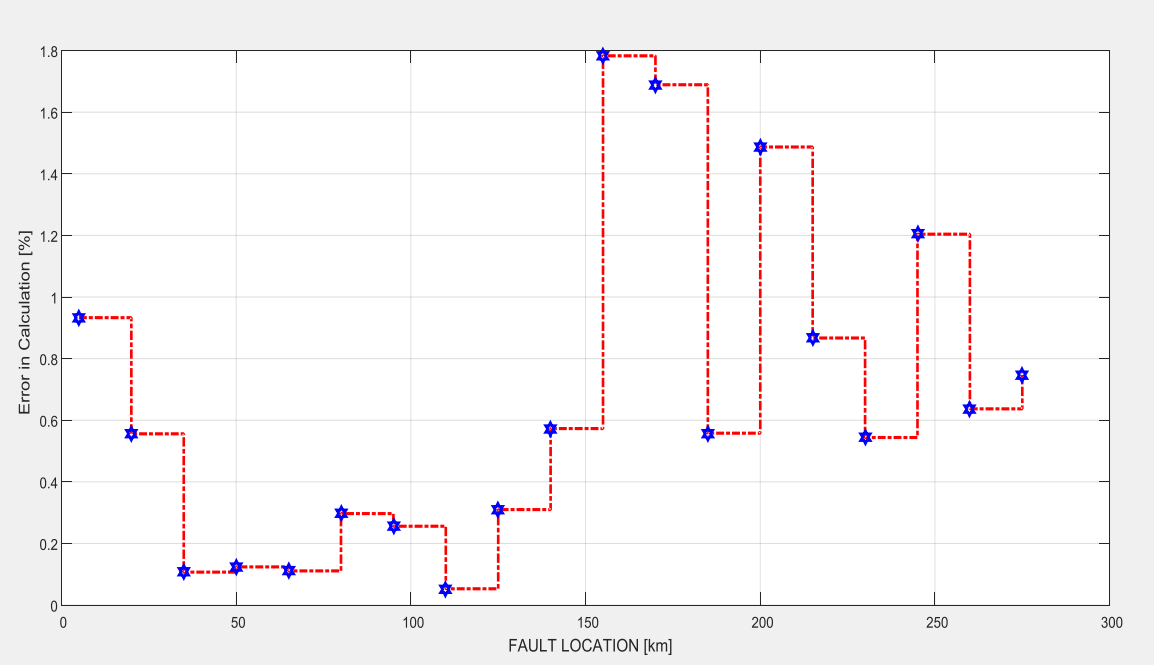


Figure 4.25: Test phase performance of the ANN with configuration (3.8.10.20.1).

In order to test the performance of this network, 19 different single phase faults have been simulated on different phases with the fault distance being incremented by 15km in each case and the percentage error in calculated output has been calculated. Figure 4.25 shows the results of this test conducted on the neural network (3.8.10.20.1). It can be seen that the maximum error is around 1.78% which is very satisfactory. It is to be noted that the average error in fault location is just 0.89%.

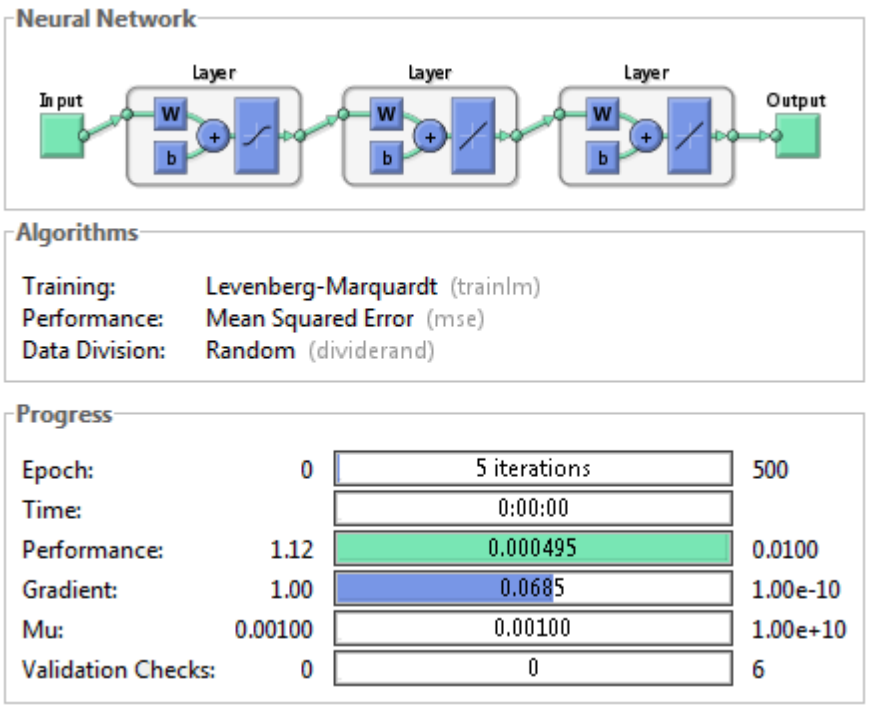


Figure 4.26: Overview of the chosen ANN with configuration (3.8.10.20.1).

Figure 4.26 shows an overview of the chosen ANN and it can be seen that the training algorithm used is Levenberg - Marquardt algorithm. The performance function chosen for the training process is mean square error.

4.3.1b Simulation Results of Testing the Neural Network for Single Line – Ground Fault Location

Several factors have been considered while testing the performance of the neural networks. One prime factor that evaluates the efficiency of the ANN is the test phase performance already illustrated in Fig 4.27. As already mentioned, the average and the maximum error percentages are in tolerable ranges and hence the network's performance is considered satisfactory.

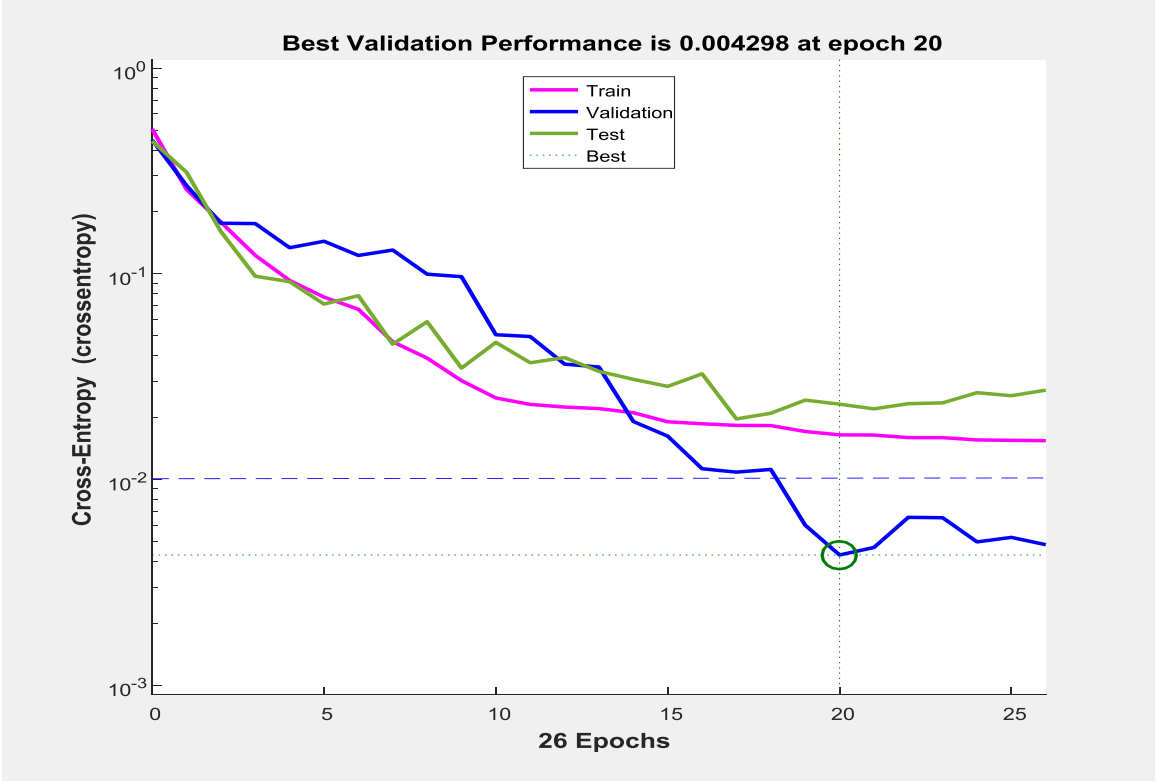


Figure 4.27: Mean-square error performance of the network with configuration (3.8.10.20.1).

Figure 4.27 plots the mean-square error as a function of time during the learning process and it can be seen that the achieved Cross-Entropy is about 4.298e-3 which is way below the Cross-Entropy goal of 1e-2.

The result of another form of analysis is provided by Figure4.28, which is the gradient and validation performance plot. It can be seen that there is a steady decrease in the gradient and also that the number of validation fails are 0 during the entire process which indicates smooth and efficient training.

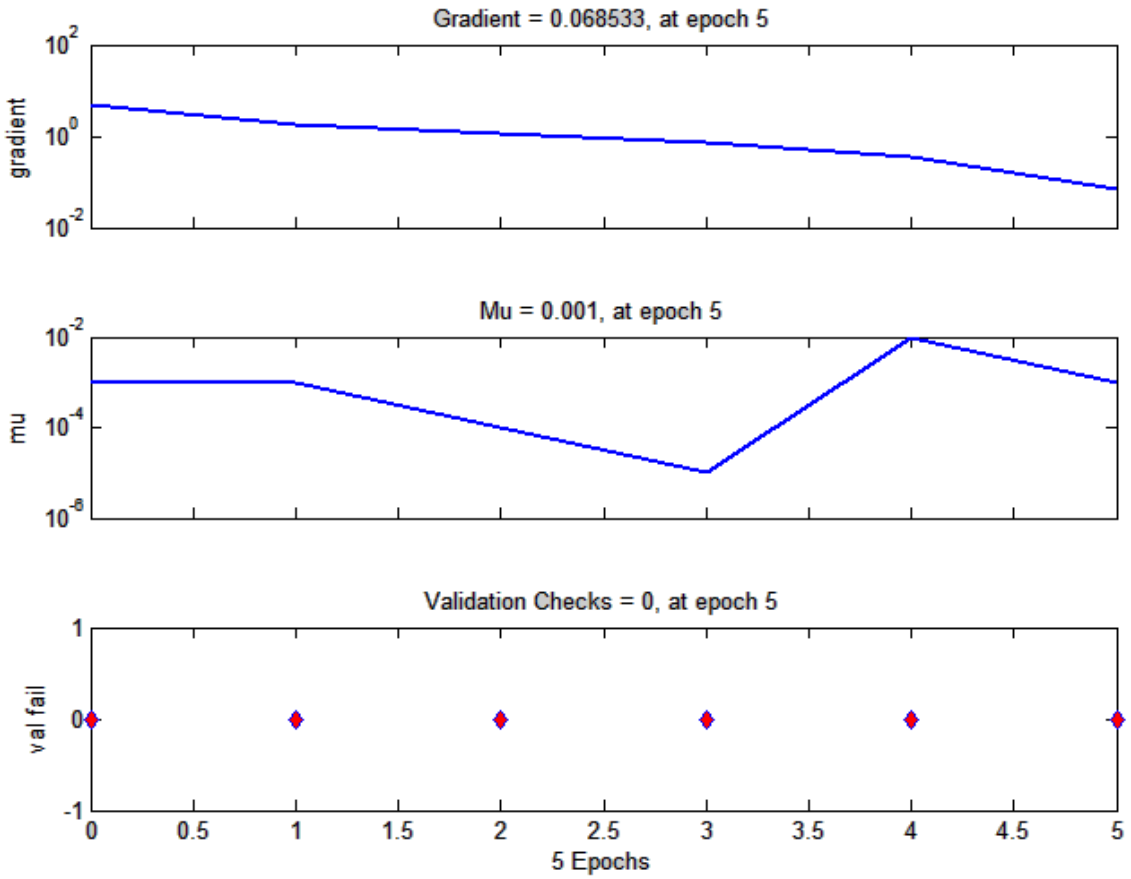


Figure 4.28: Gradient and validation performance of the network with configuration (3.8.10.20.1).

The third factor that is considered while evaluating the performance of the network is the correlation coefficient of each of the various phases of training, validation and testing. Figure 4.29 shows the regression plots of the various phases such as training, testing and validation. It can be seen that the best linear fit very closely matches the ideal case with an overall correlation coefficient of 0.99924.

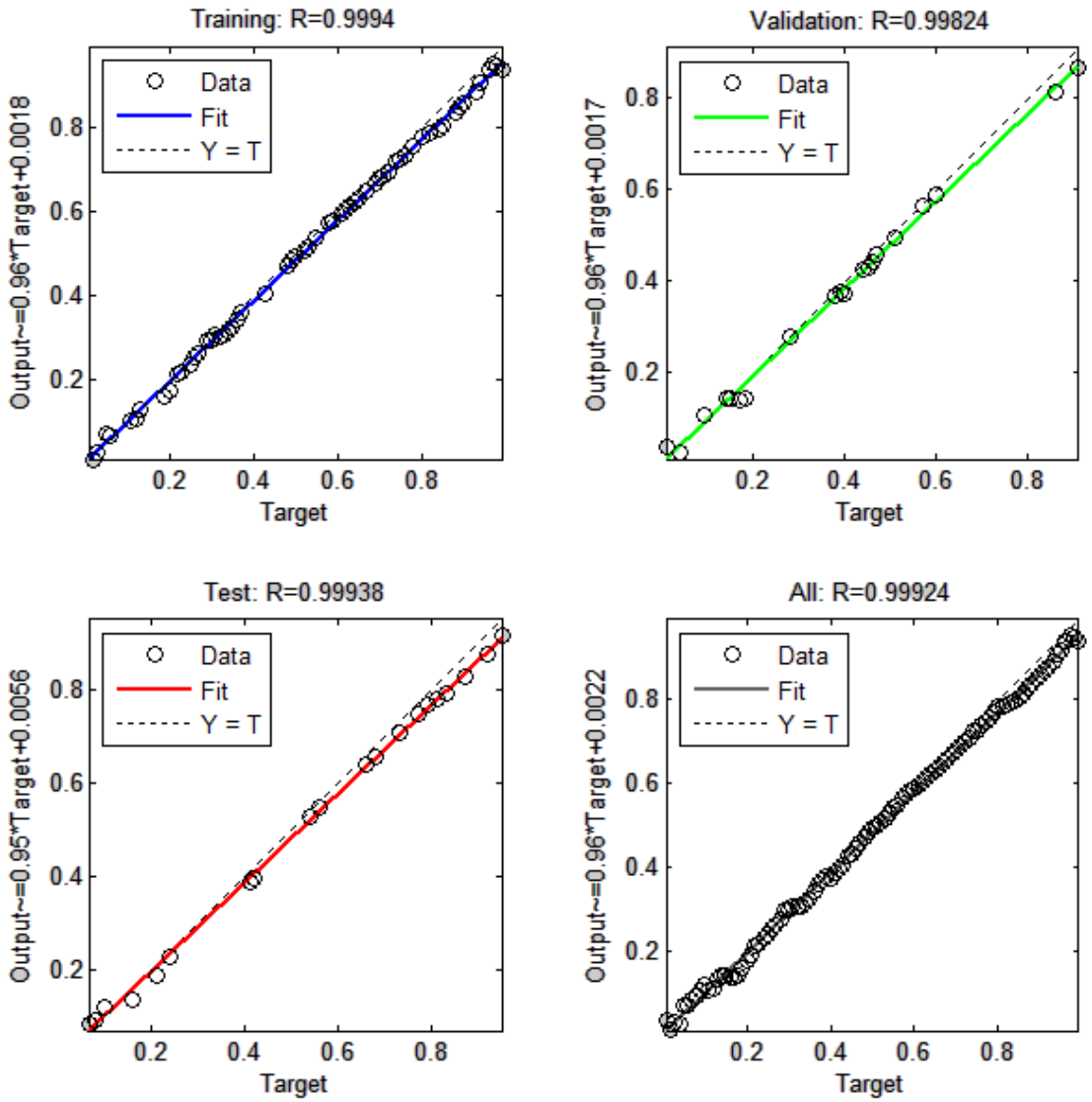


Figure 4.29: Regression plots of various phases of learning of the ANN with configuration (3.8.10.20.1).

Fig 4.30 shows the structure of the chosen ANN for single line – ground faults with 3 neurons in the input layer, 8, 10, and 20 neurons in the 3 hidden layers respectively and 1 neuron in the output layer (3.8.10.20.1).This is a pictorial representation of how the neurons in the respective layers are connected together through the synaptic weights. It shows the interconnections between the input layer and the hidden layers, and also between the hidden

layers and the output layer. It can be seen that any given neuron is connected to all the neurons in the layer in front.

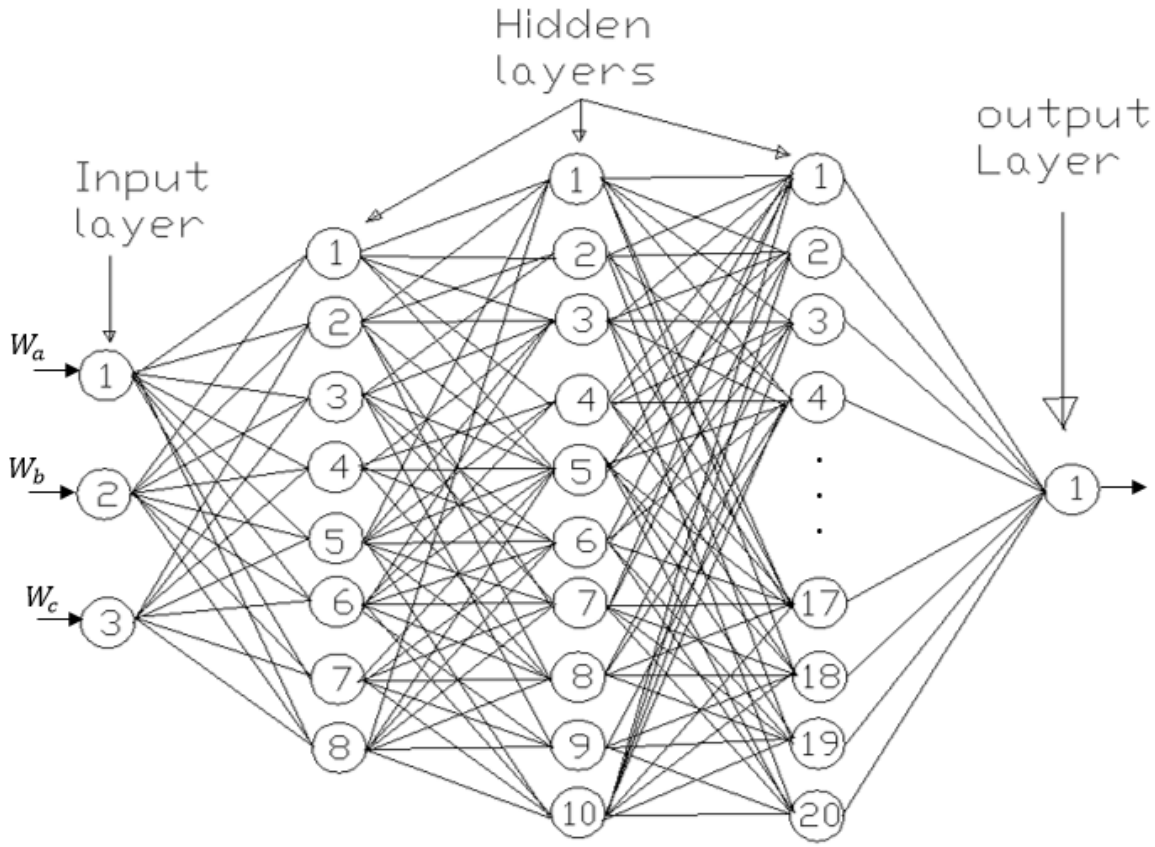


Figure 4.30: Structure of the chosen ANN with configuration(3.8.10.20.1).

Table 4.1 illustrates the percentage errors in Fault location as a function of Fault Distance and Fault Resistance. Two different cases have been considered (shown in adjacent columns), one with a fault resistance of 15 Ohms and another with a fault resistance of 70 Ohms. It is to be noted that the resistance of 15 Ohms was used as a part of training data set and hence the average percentage error in fault location in this case is 0.65%. The second case illustrates the same with a different fault resistance of 70 Ohms which is relatively very high and is not a part of the training set. Hence, the performance of the neural network in this case illustrates its ability to generalize and react upon new data. It is to be noted that the average error in this

case is 0.7284% which is very satisfactory. Thus, the neural networks performance is considered satisfactory and can be used for the purpose of single line – ground fault location.

Table 4.1: Percentage errors as a function of fault distance and fault resistance for the ANN chosen for single line - ground fault location.

Serial No:	% Error vs. Fault Distance (Fault Resistance = 15 Ω)			% Error vs. Fault Distance (Fault Resistance = 70 Ω)		
	Fault Distance (KM)	Estimated Fault Location	Percentage Error	Fault Distance (KM)	Estimated Fault Location	Percentage Error
1	5	4.95	0.01	10	10.06	0.6
2	20	20.25	1.25	25	25.20	0.80
3	35	35.28	0.80	40	40.52	1.30
4	50	50.13	0.26	55	54.87	0.24
5	65	65.75	1.15	70	69.51	0.70
6	80	80.49	0.61	85	85.76	0.88
7	95	95.58	0.61	100	101.12	1.12
8	110	111.12	1.02	115	115.49	0.43
9	125	125.79	0.63	130	131.02	0.78
10	140	141.53	1.09	145	146.06	0. 73
11	155	153.86	0.74	160	161.07	0.67
12	170	171.05	0.62	175	174.50	0.29
13	185	185.88	0.48	190	191.84	0.97
14	200	201.13	0.57	205	207.01	0.98
15	215	214.56	0.2	220	222.05	0.93
16	230	229.49	0.22	235	236.02	0.43
17	245	246.62	0.66	250	251. 89	0.76
18	260	258.15	0.71	265	266.37	0.52
19	275	273.01	0.72	280	282.00	0.71

4.3.2 Line – Line Faults

The results of the design, development and performance of neural networks for the purpose of Line–Line fault location are discussed in this section. Now that the occurrence of a single line – ground fault on a transmission line can be determined,the next task is to pin-point the location of the line – line fault from either ends of the transmission line when it occurs. Three possible line – line faults exist (A-B, B-C, C-A), corresponding to each of the three phases (A, B or C) being faulted.

4.3.2a Simulation Results of Training the Neural Network for Line – Line Fault Location

An exhaustive survey on various neural networks has been performed by varying the number of hidden layers and the number of neurons per hidden layer. Certain neural networks that achieved satisfactory performance are presented first along with their error performance plots. Of these ANNs, the most appropriate ANN is chosen based on its Cross Entropy performance and the Regression coefficient of the Outputs versus Targets. Figures 4.31 – 4.32 show the Cross-Entropy and the Test phase performance plots of the neural networks 3.10.20.5.1 with 3 hidden layers. Figures 4.33 – 4.34 show the Cross-Entropy and the Test phase performance plots of the neural network 3.10.1 with 1 hidden layer.

Figure 4.31 shows the performance of the neural network (in terms of training, testing and validation) with 3 neurons in the input layer, 3 hidden layers with 24, 18 and 5 neurons in them respectively and 1 neuron in the output layer (3.24.18.5.1). It can be seen that the best Cross-Entropy performance of this neural network is 2.8469e-2which is above the Cross-Entropy goal of 1e-2. It was found that the correlation coefficient between the outputs and the targets was 0.80269 in this case which is not very satisfactory.

The chosen neural network has four hidden layers with 12 neurons in the first hidden layer, 5 neurons in the second hidden layer, 15 neurons in the third hidden layer and 30 neurons in the fourth hidden layer (3.12.5.15.30.1).

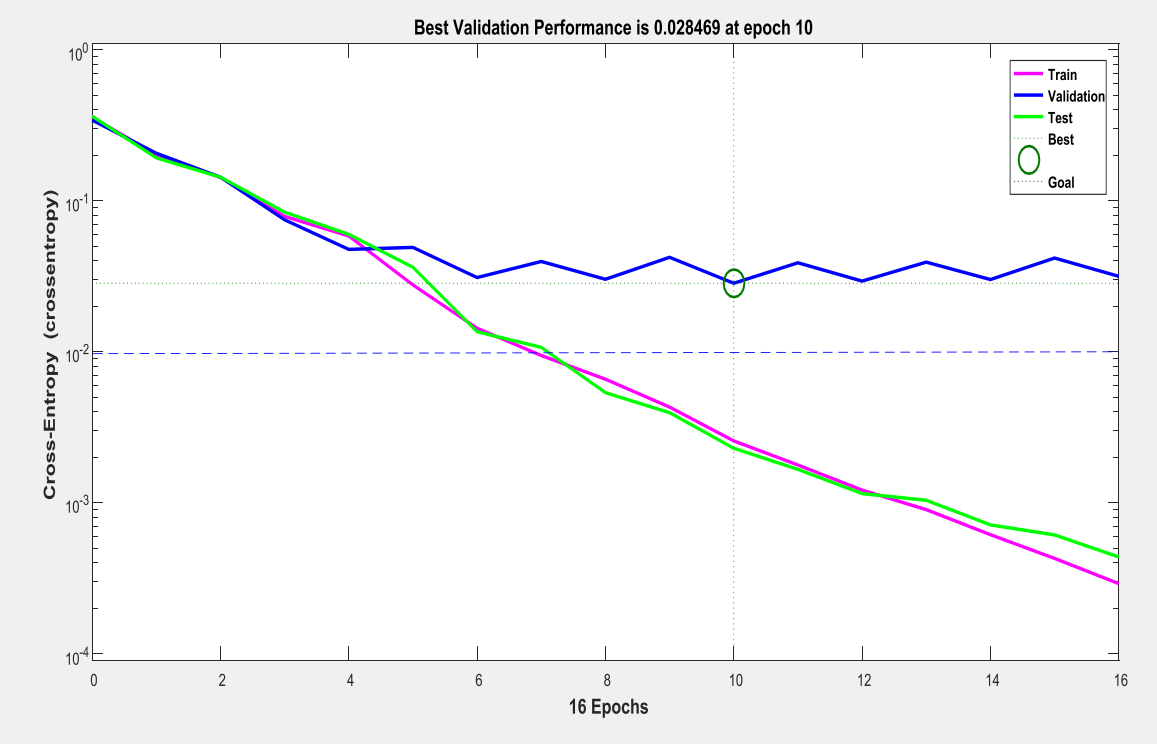


Figure 4.31: Mean Square Error performance plot with configuration (3.24.18.5.1).

In order to test the performance of this network, 19 different line – line faults have been simulated on different phases with the fault distance being incremented by 15km in each case and the percentage error in calculated output has been calculated. Figure 4.32 shows the results of this test conducted on the neural network (3.24.18.5.4). The maximum error is around 7.35% which again is not satisfactory.

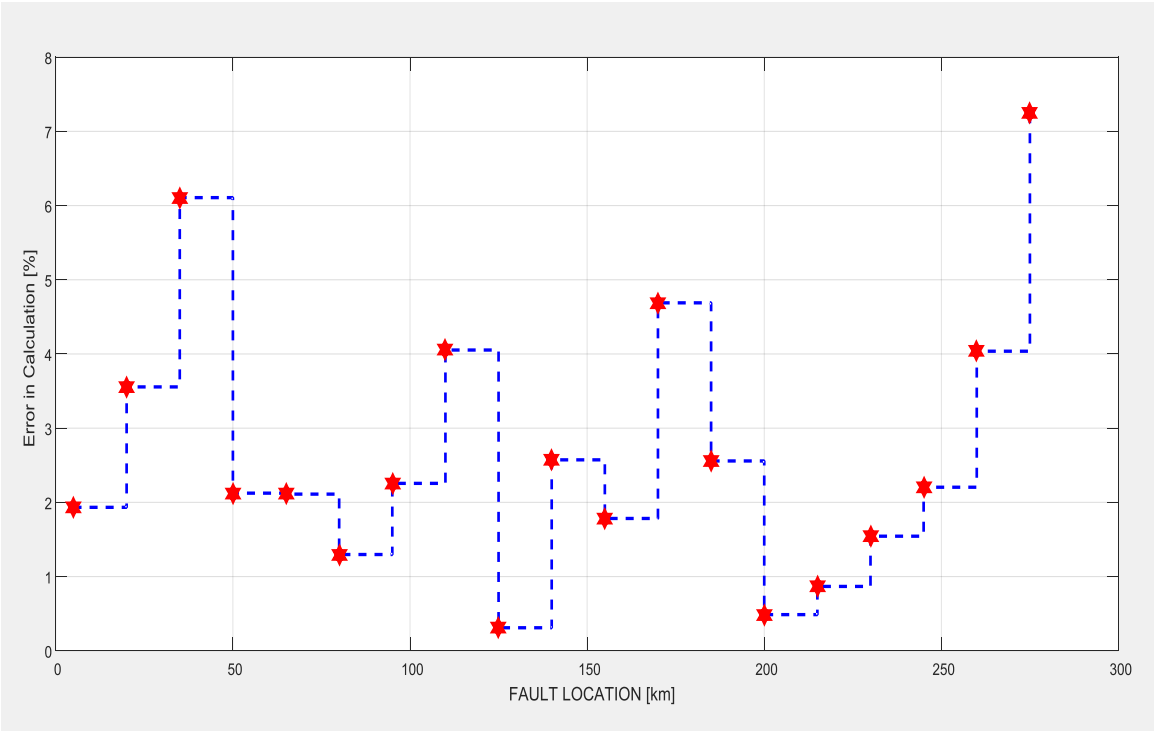


Figure 4.32: Test Phase performance of the ANN with configuration (3.24.18.5.1).

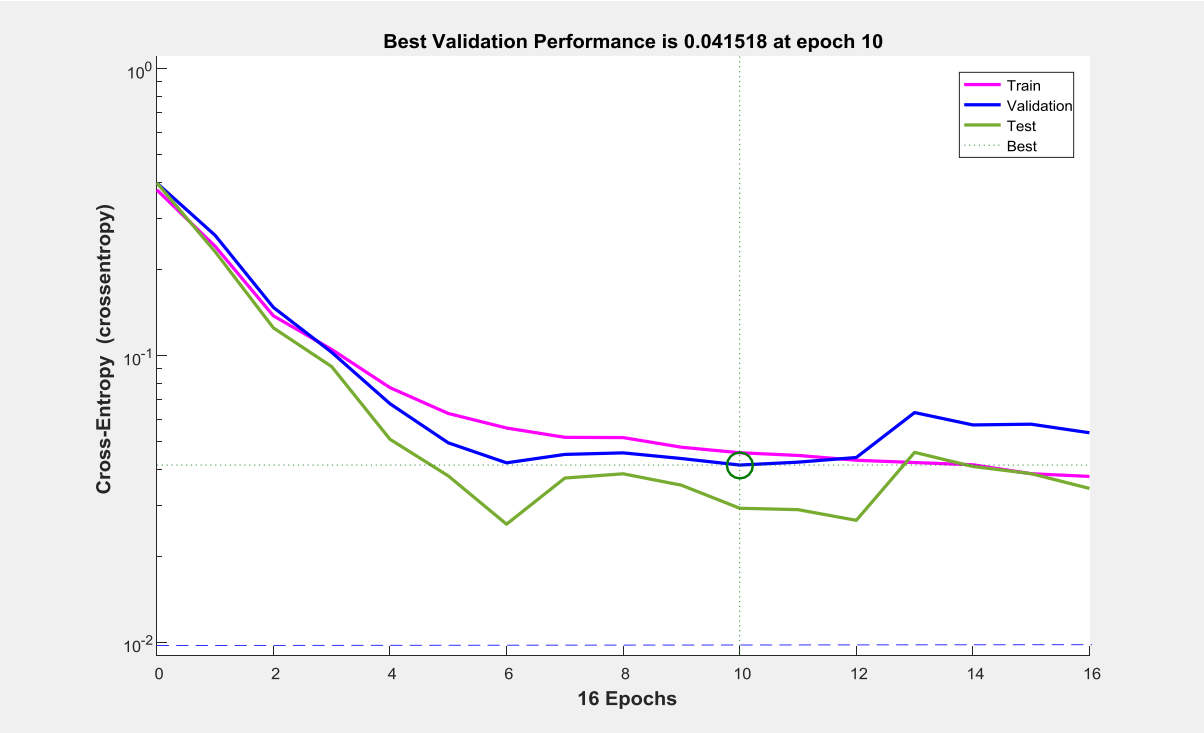


Figure 4.33: Mean Square Error performance plot with configuration (3.10.5.1).

Figure 4.33 shows the performance of the neural network (in terms of training, testing and validation) with 3 neurons in the input layer, 10 neurons in the hidden layer and 1 neuron in the output layer (3.10.1). It can be seen that the best Cross-Entropy performance of this neural network is 4.1518e-2 which is above the Cross-Entropy goal of 1e-2. It was found that the correlation coefficient between the outputs and the targets was 0.7025 for this neural network and this is not acceptable.

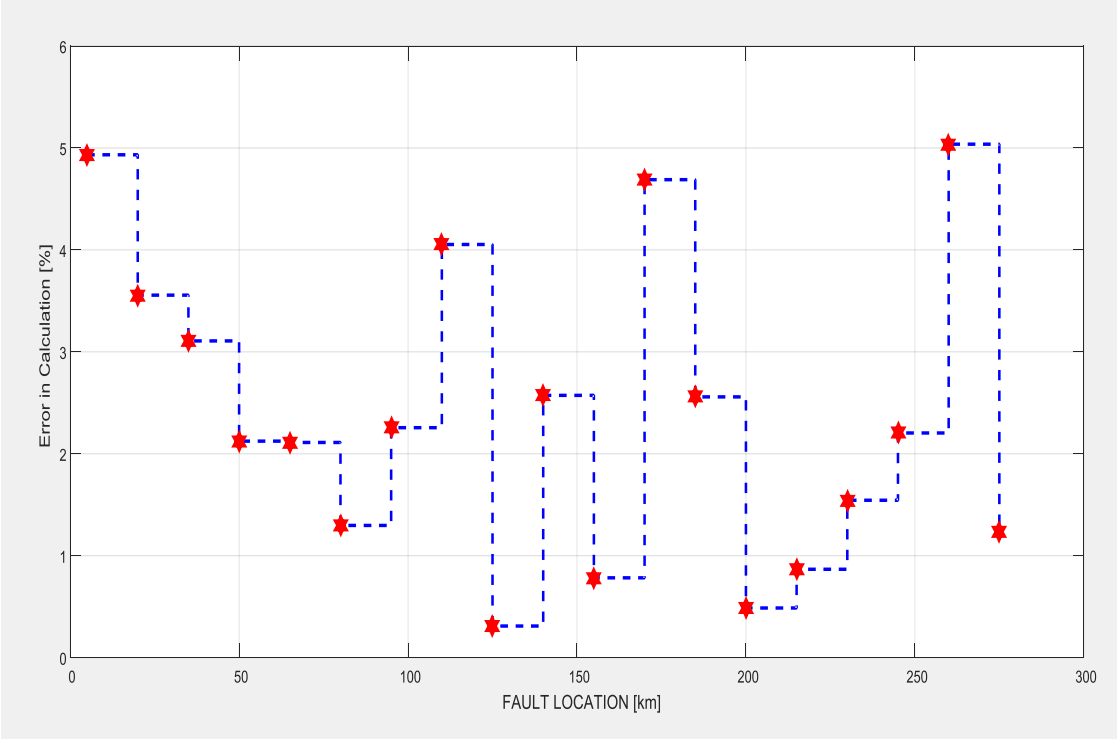


Figure 4.34: Test Phase performance of the ANN with configuration (3.10.1).

In order to test the performance of this network, 19 different line – line faults have been simulated on different phases with the fault distance being incremented by 15 km in each case and the percentage error in calculated output has been calculated. Fig 4.34 shows the results of this test conducted on the neural network (3.10.1). The maximum error is around 5.06% which is unacceptable.

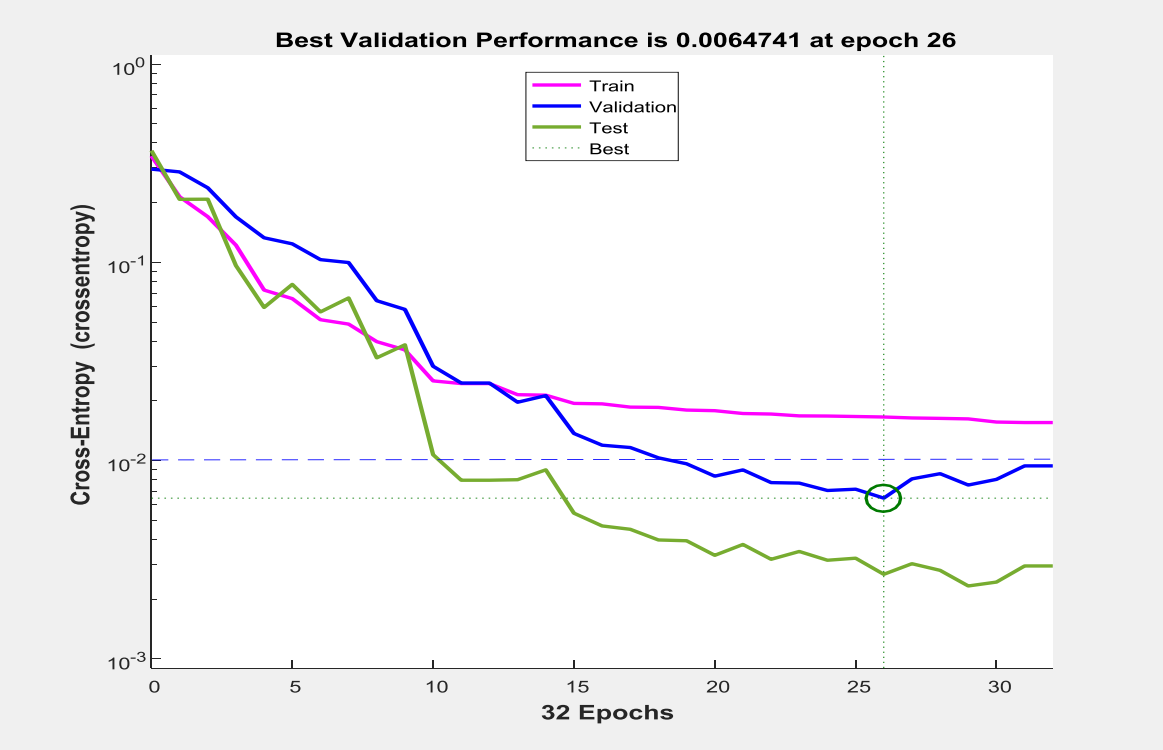


Figure 4.35: Mean Square Error performance of the ANN with configuration (3.12.5.15.30.1).

Figure 4.35 shows the performance of the neural network (in terms of training, testing and validation) with 3 neurons in the input layer, 4 hidden layers with 12, 5, 15 and 30 neurons in them respectively and 1 neuron in the output layer (3.12.5 .15.30.1). It can be seen that the best Cross-Entropy performance of this neural network is 6.4741e-3 which is below the Cross-Entropy goal of 1e-2. It was found that the correlation coefficient between the outputs and the targets was 0.99648 for this neural network. Therefore, this network has been trained correctly.

In order to test the performance of this network, 19 different phase to phase faults have been simulated on different phases with the fault distance being incremented by 15km in each case and the percentage error in calculated output has been presented.

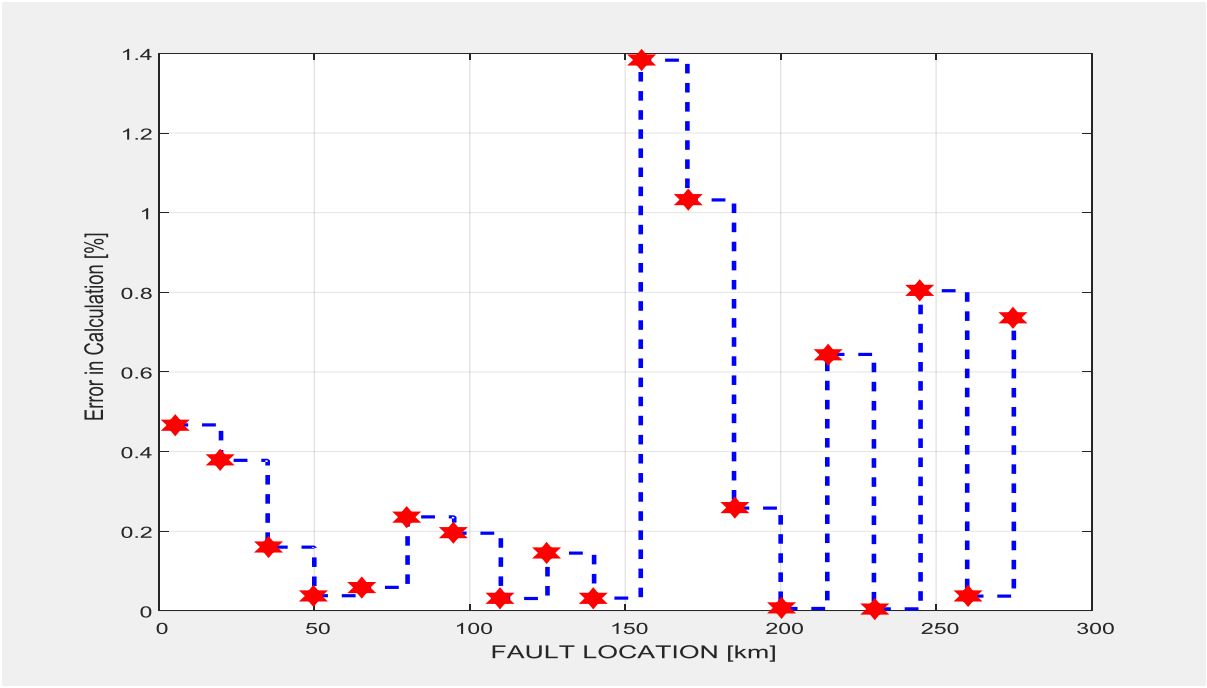


Figure 4.36: Test phase performance of the neural network with configuration (3.12.5.15.30.1).

Figure 4.36 shows the results of this test conducted on the neural network (3.12.5.15.30.1). It can be seen that the maximum error is around 1.37% which is very satisfactory. It is to be noted that the average error in fault location is 0.97 percent. Hence, this neural network has been chosen as the ideal network for the purpose of line – line fault location on transmission lines.

Figure 4.37 shows an overview of the chosen ANN and it can be seen that the training algorithm used is Levenberg - Marquardt algorithm. The performance function chosen for the training process is mean square error. Fig 4.38shows the plots the best linear regression fit between the outputs and the targets and the correlation coefficient for the same has been found to be 0.99648 which is a decently good regression fit.

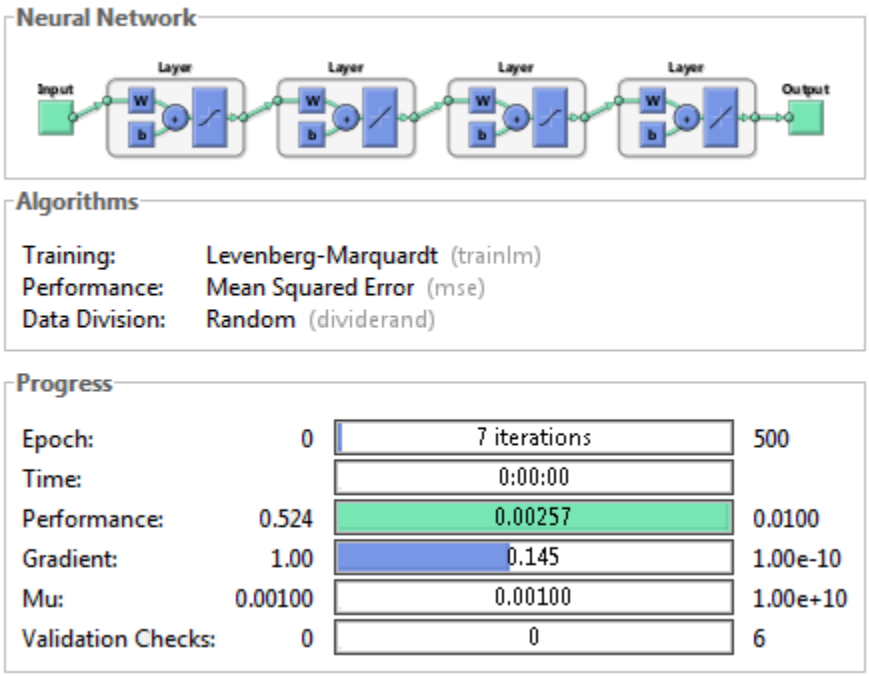


Figure 4.37: Overview of the chosen ANN for Line-Line Faults (3.12.5.15.30.1).

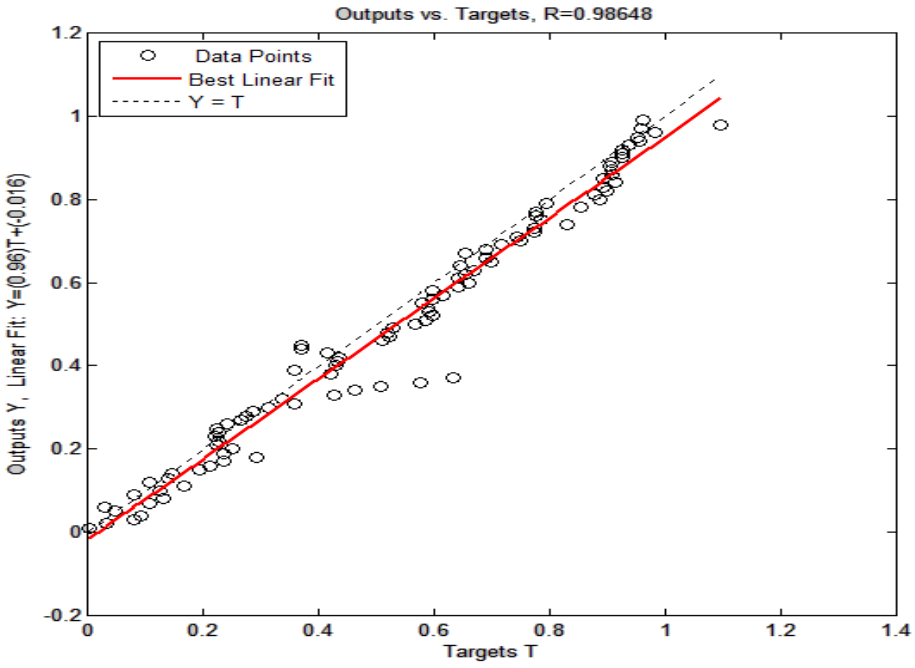


Figure 4.38: Regression fit of the outputs versus targets with configuration (3.12.5.15.30.1)

4.3.2b Simulation Results of Testing the Neural Network for Line – Line Fault Location

Several factors have been considered while testing the performance of the chosen neural network. One prime factor that evaluates the efficiency of the ANN is the test phase performance plot which is already illustrated in Figure 4.36. As already mentioned, the average and the maximum error percentages are in tolerable ranges and hence the network's performance is considered satisfactory. Another means of evaluating the ANN is provided by Fig 4.39, which is the gradient and validation performance plot. It can be seen that there is a steady decrease in the gradient and also that the number of validation fails did not exceed 1 during the entire process which indicates smooth and efficient training because the validation and the test phases reached the Cross-Entropy goal at the same time approximately.

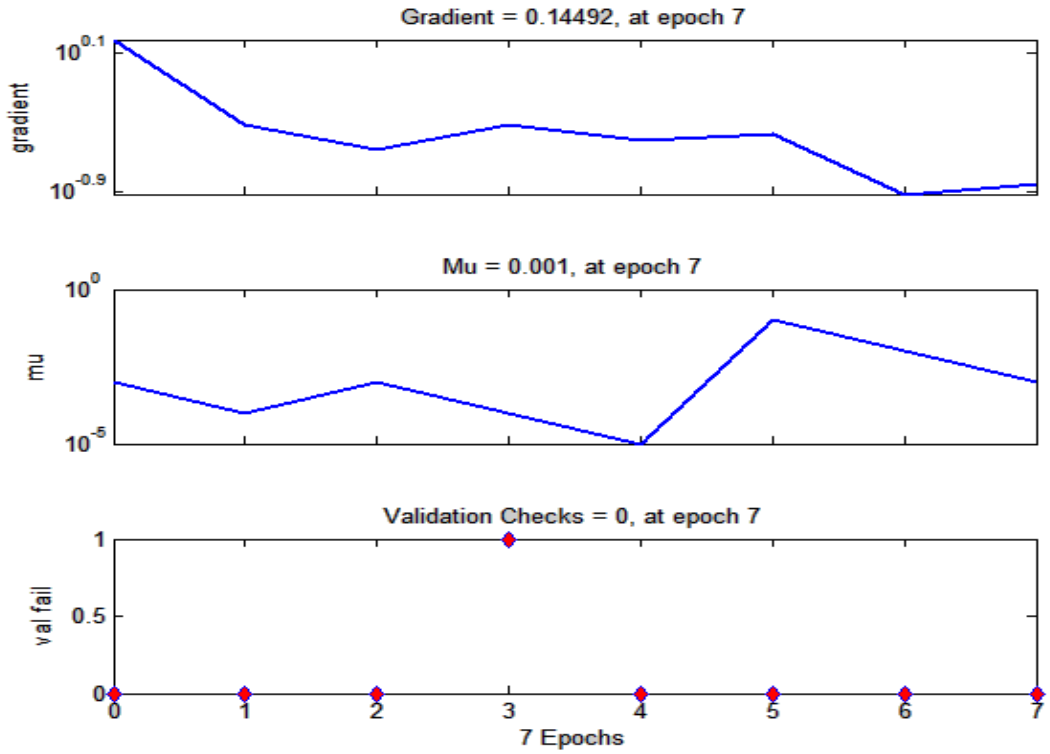


Figure 4.39: Gradient and validation performance plot of the ANN (3.12.5.15.30.1).

The third factor that is considered while evaluating the performance of the network is the correlation coefficient of each of the various phases of training, validation and testing. Figure 4.40 shows the regression plots of the various phases such as training, testing and validation. It can be seen that the best linear fit very closely matches the ideal case with an overall correlation coefficient of 0.98648.

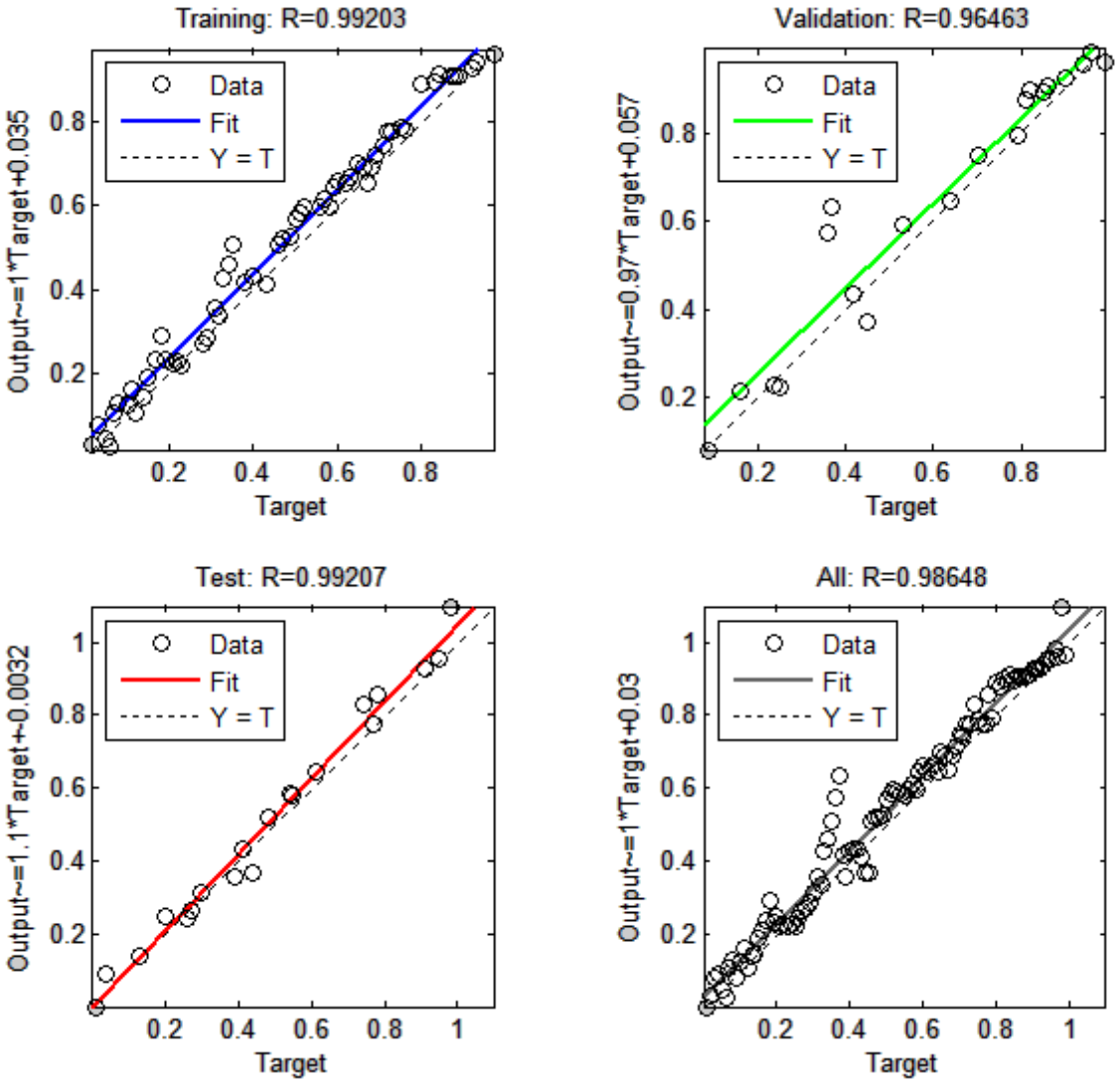


Figure 4.40: Regression plots of the various phases of learning of the chosen ANN (3.12.5.15.30.1).

Figure 4.41 shows the structure of the chosen ANN for line – line faults with 3neurons in the input layer, 4 hidden layers with 12, 5, 15 and 30 neurons in them respectively and 1 neuron in the output layer (3.12.5.15.30.1). This is a pictorial representation of how the neurons in the respective layers are connected together through the synaptic weights. It shows the interconnections between the input layer and the hidden layers, and also between the hidden layers and the output layer. It can be seen that any given neuron is connected to all the neurons in the layer in front.

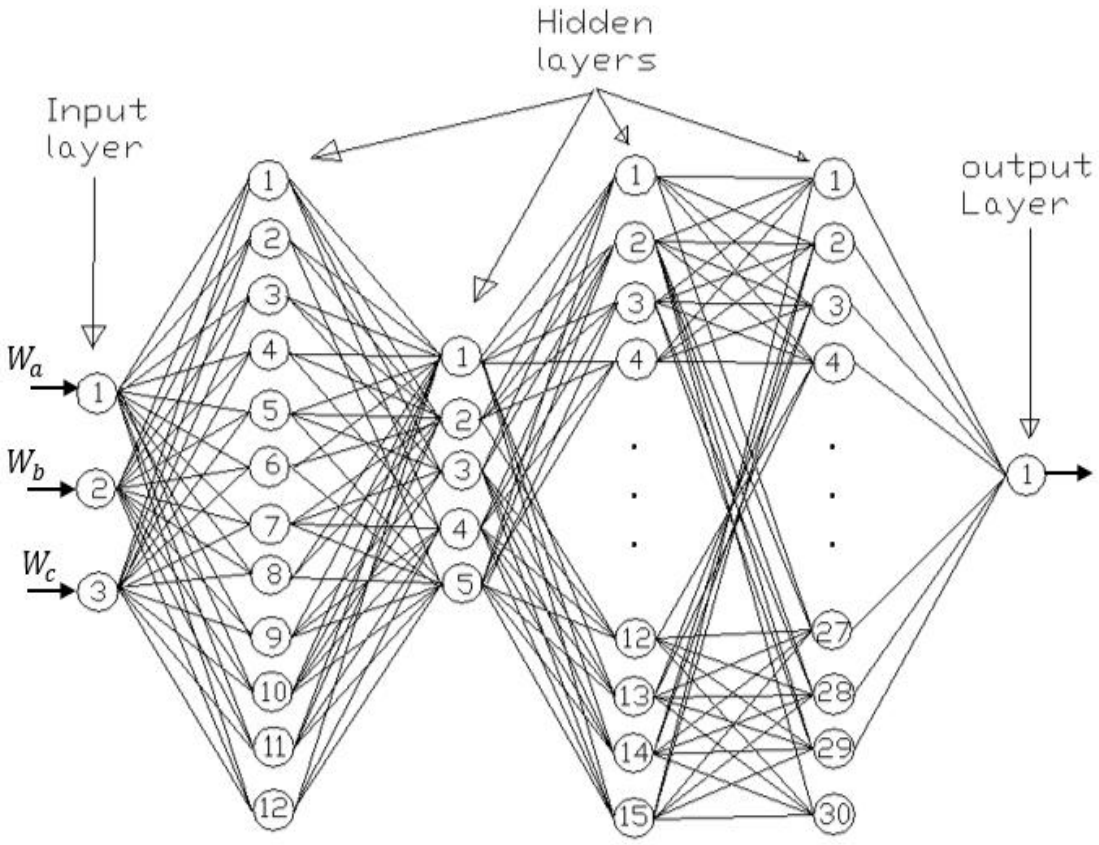


Figure 4.41: Structure of the chosen Neural Network (3.12.5.15.30.1).

Table 4.2 illustrates the percentage errors in Fault location as a function of Fault Distance and Fault Resistance. Two different cases have been considered (shown in adjacent columns), one with a fault resistance of 15 Ohms and another with a fault resistance of 70 ohms. It is to

benoted that the resistance of 15 Ohms was used as a part of training data set and hence the average percentage error in fault location in this case is 0.5811 %.

Table 4.2: Percentage errors as a function of fault distance and fault resistance for the ANN chosen for line - line fault location.

Serial No:	% Error vs. Fault Distance (Fault Resistance = 15 Ω)			% Error vs. Fault Distance (Fault Resistance = 70 Ω)		
	Fault Distance (Km)	Estimated Fault Location	Percentage Error	Fault Distance (Km)	Estimated Fault Location	Percentage Error
1	5	5.07	1.40	10	10.08	0.80
2	20	20.17	0.85	25	25.13	0.52
3	35	35.12	0.34	40	41.28	0.07
4	50	50.04	0.08	55	55.51	0.93
5	65	65.18	0.28	70	70.47	0.67
6	80	80.39	0.49	85	86.16	1.36
7	95	95.38	0.04	100	100.82	0.82
8	110	110.17	0.15	115	115.89	0.77
9	125	125.37	0.30	130	130.88	0.68
10	140	140.95	0.68	145	146.16	0.80
11	155	154.84	0.10	160	161.00	0.63
12	170	170.98	0.56	175	176.96	1.12
13	185	186.03	0.56	190	191.92	1.01
14	200	201.32	0.66	205	206.81	0.88
15	215	213.56	0.67	220	221.19	0.54
16	230	228.56	0.63	235	236.73	0.74
17	245	247.43	0.99	250	251.00	0.40
18	260	262.76	1.06	265	266.82	0.69
19	275	278.30	1.2	280	282.88	1.03

The second case illustrates the same with a different fault resistance of 70 ohms which is relatively very high and is not a part of the training set. Hence, the performance of the neural

network in this case illustrates its ability to generalize and react upon new data. It is to be noted that the average error in this case is just 0.7611 % which is still very satisfactory. Thus, the neural networks performance is considered satisfactory and can be used for the purpose of line – line fault location.

4.3.3 Double Line – Ground Faults

The results of the design, development and performance of neural networks for the purpose ofDouble Line – Ground fault location are shown and discussed in this section. The third category of faults is the double line – ground faults. Three possible double line – ground faults exist which are denoted as ABG, BCG and ACG (based on which two of the three phases A, B and C are faulted).

4.3.3a Simulation Results of Training the Neural Network for Double Line – Ground Fault Location

A few neural networks that achieved satisfactory performance are presented first along with their error performance plots. Of these ANNs, the most appropriate ANN is chosen based on its Cross Entropy performance and the Regression coefficient of the Outputs vs. Targets. Figures 4.42 – 4.47 show the Cross-Entropy and the Test phase performance plots of the neural networks (3.20.1),(3.10.15.1)and (3.20.15.1) with 1 and 2 hidden layers respectively. Figures 4.48 – 4.49 show the MSE and the Test phase performance plots of the neural network 3.10.5.1 and 3.21.35.15.7.1 which has shown satisfactory performance. After exhaustive survey, the chosen neural network has four hidden layers with 21 neurons in the first hidden layer, 35 neurons in the second hidden layer, 15 neurons in the third hidden layer and 7 neurons in the fourth hidden layer (3.21.35.15.7.1).

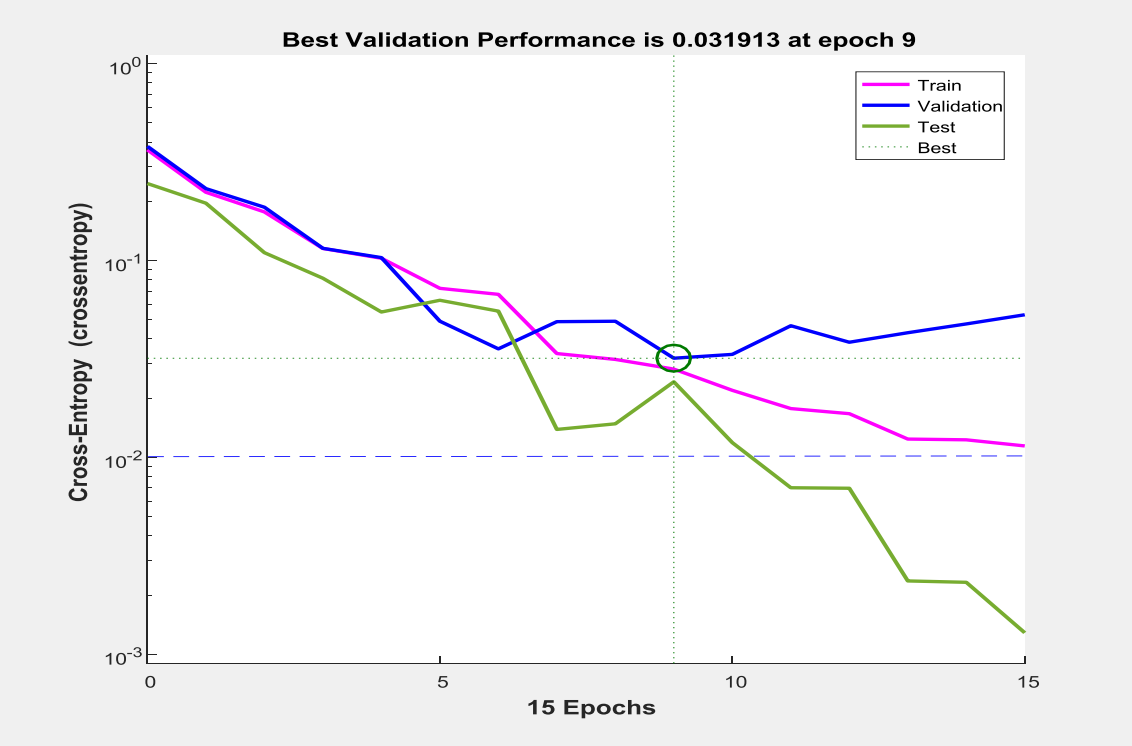


Figure 4.42: Mean Square Error performance of the ANN with configuration (3.20.1).

Figure 4.42 shows the performance of the neural network (in terms of training, testing and validation) with 3 neurons in the input layer, 20 neurons in the hidden layer and 1 neuron in the output layer (3.20.1). It can be seen that the best Cross-Entropy performance of this neural network is 3.1913e-2 which is above the Cross-Entropy goal of 1e-2 (denoted by the blue dotted line). It was found that the correlation coefficient between the outputs and the targets was 0.91393 in this case. Therefore, the network failed to train correctly.

In order to test the performance of this network, 19 different double line – ground faults have been simulated on different phases with the fault distance being incremented by 15km in each case and the percentage error in ANN's output has been calculated. Figure4.43 shows the results of this test conducted on the neural network (3.20.1). It can be seen that the maximum error is higher than 5.9% which is exorbitantly high.

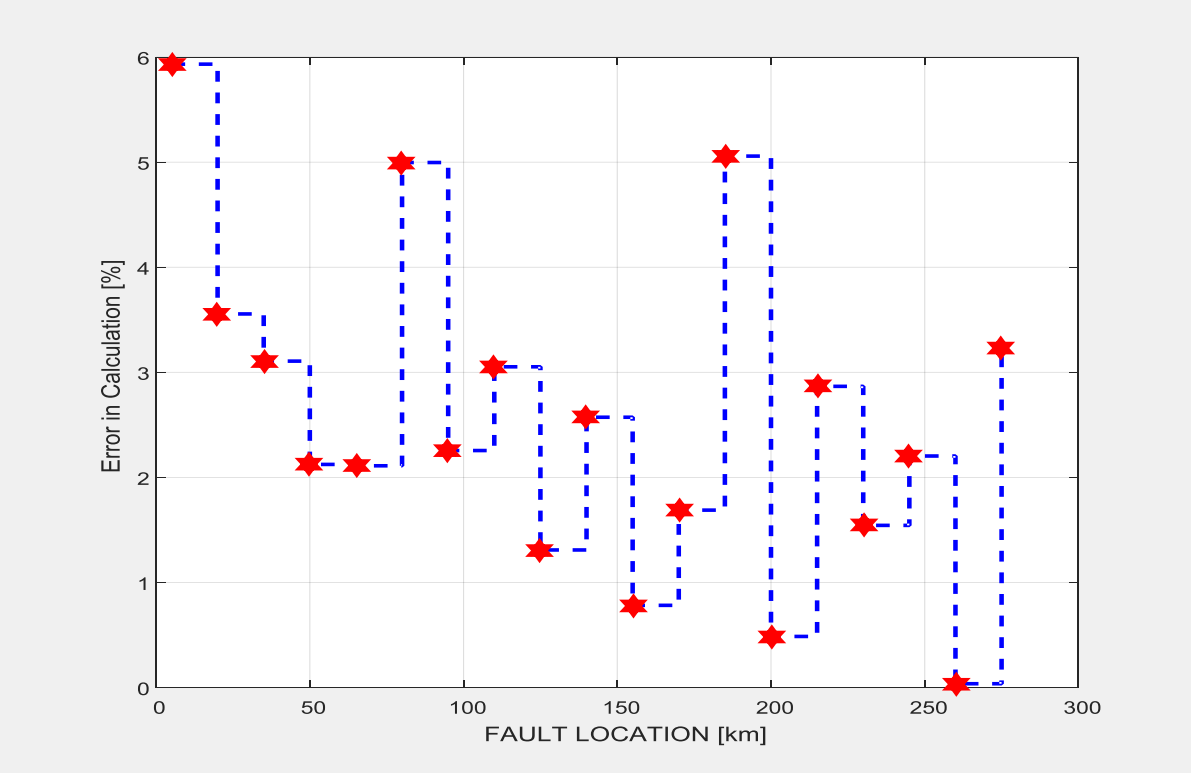


Figure 4.43: Test Phase performance of the ANN with configuration (3.20.1).

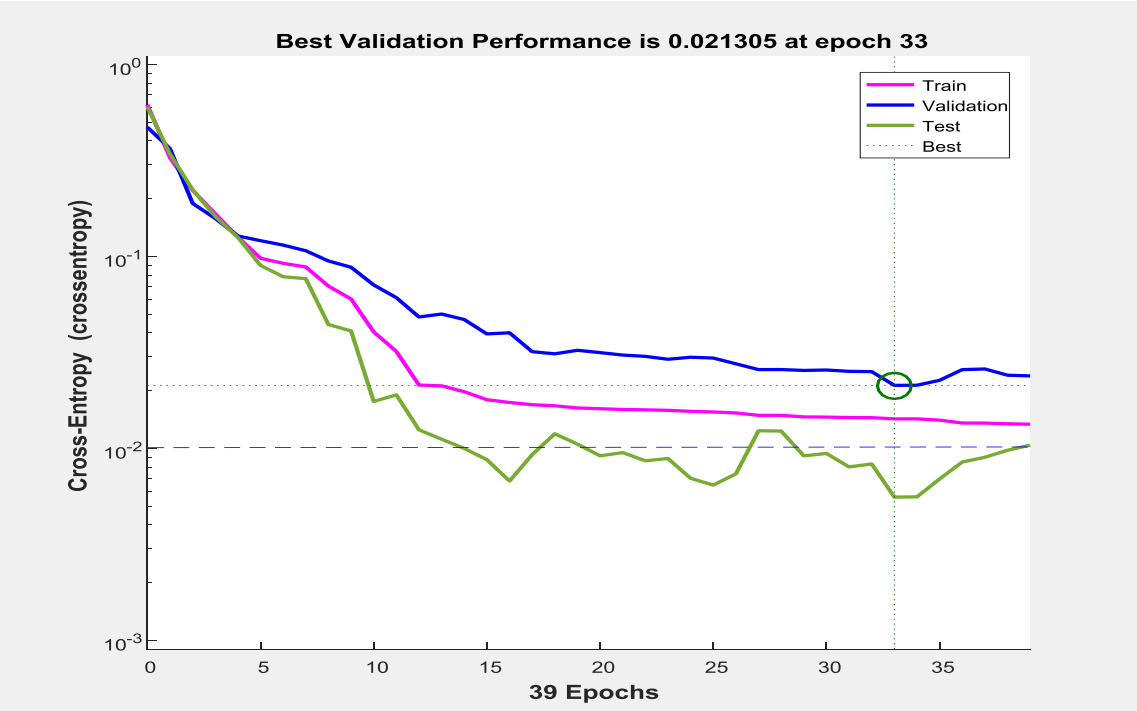


Figure 4.44: Mean Square Error performance of the ANN with configuration (3.10.15.1).

Figure 4.44 shows the performance of the neural network (in terms of training, testing and validation) with 3 neurons in the input layer, 10 and 15 neurons in the hidden layers respectively and 1 neuron in the output layer (3.10.15.1). It can be seen that the best Cross-Entropy performance of this neural network is 2.1305e-2 which is below the Cross-Entropy goal of 1e-2 (denoted by the blue dotted line in the figure). It was found that the correlation coefficient between the outputs and the targets was 0.96804 for this neural network which is not satisfactory.

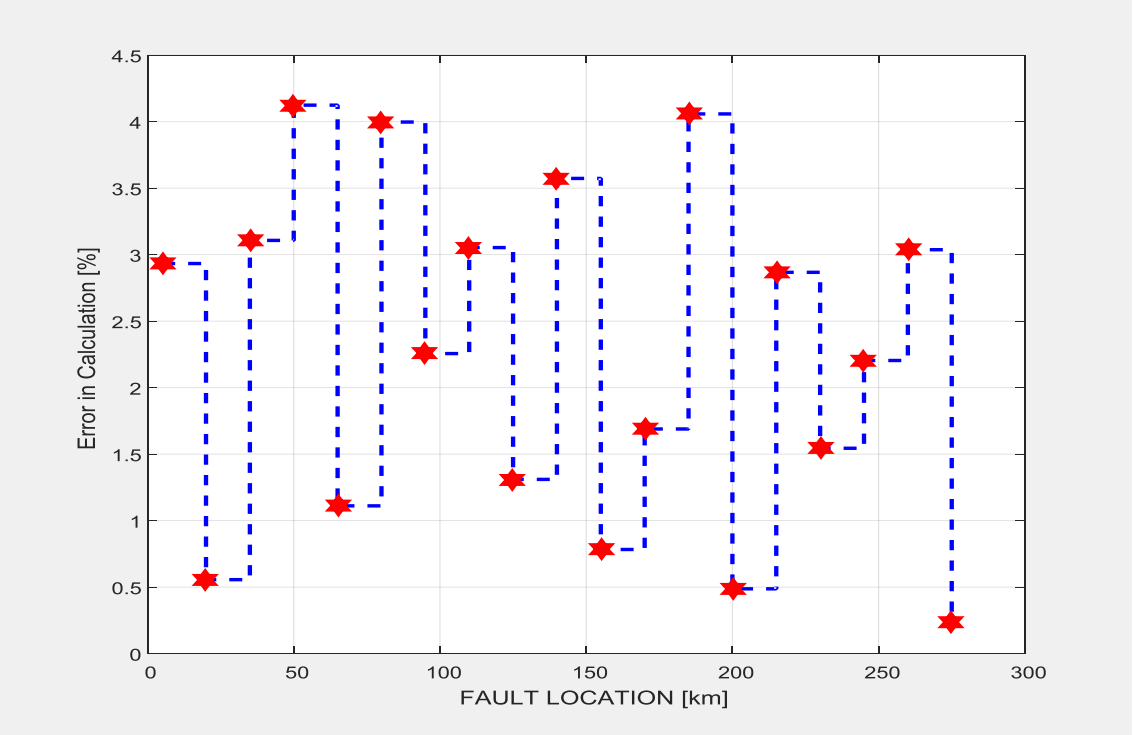


Figure 4.45: Test Phase performance of the ANN with configuration (3.10.15.1).

In order to test the performance of this network the same method adopted for the earlier case is followed. 19 different double line – ground faults have been simulated on different phases with the fault distance being incremented by 15km in each case and the percentage error in ANN's output has been calculated. Figure 4.45 shows the results of this test conducted on the

neural network (3.10.15.1). It is to be noted that the maximum error is higher than 4.27% which is too high for this purpose.

Figure 4.46 shows the performance of the neural network (in terms of training, testing and validation) with 3 neurons in the input layer, 2 hidden layers with 20 and 15 neurons in them respectively and 1 neuron in the output layer (3.20.15.1). It can be seen that the best Cross-Entropy performance of this neural network is 2.7218e-2 which is above the Cross-Entropy goal of 1e-2 (denoted by the blue dotted line in the figure). It was found that the correlation coefficient between the outputs and the targets was 0.99813 for this neural network.

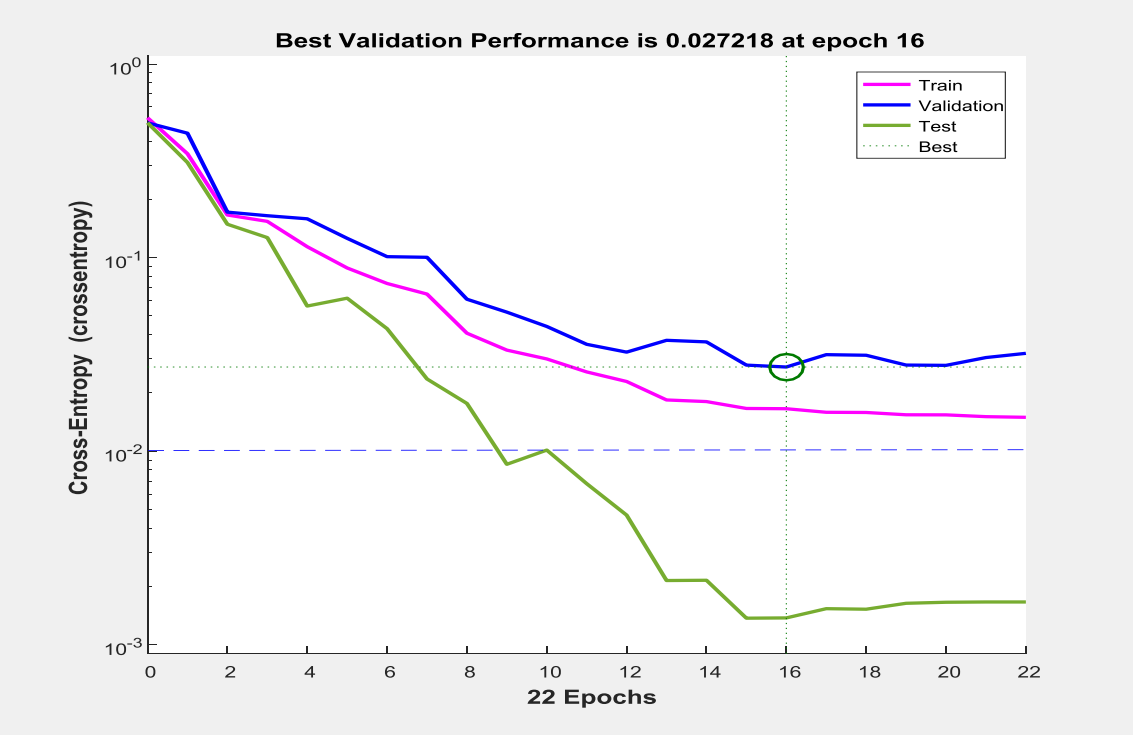


Figure 4.46: Mean Square Error performance of the neural network with configuration (3.20.15.1).

In order to test the performance of this network the same method adopted for the earlier case is followed. 19 different double line – ground faults have been simulated on different phases with the fault distance being incremented by 15 km in each case and the percentage error in

ANN's output has been calculated. Fig 4.47 shows the results of this test conducted on the neural network (3.20.15.1). It is to be noted that the maximum error is higher than 4.25% which is still not satisfactory for this purpose.

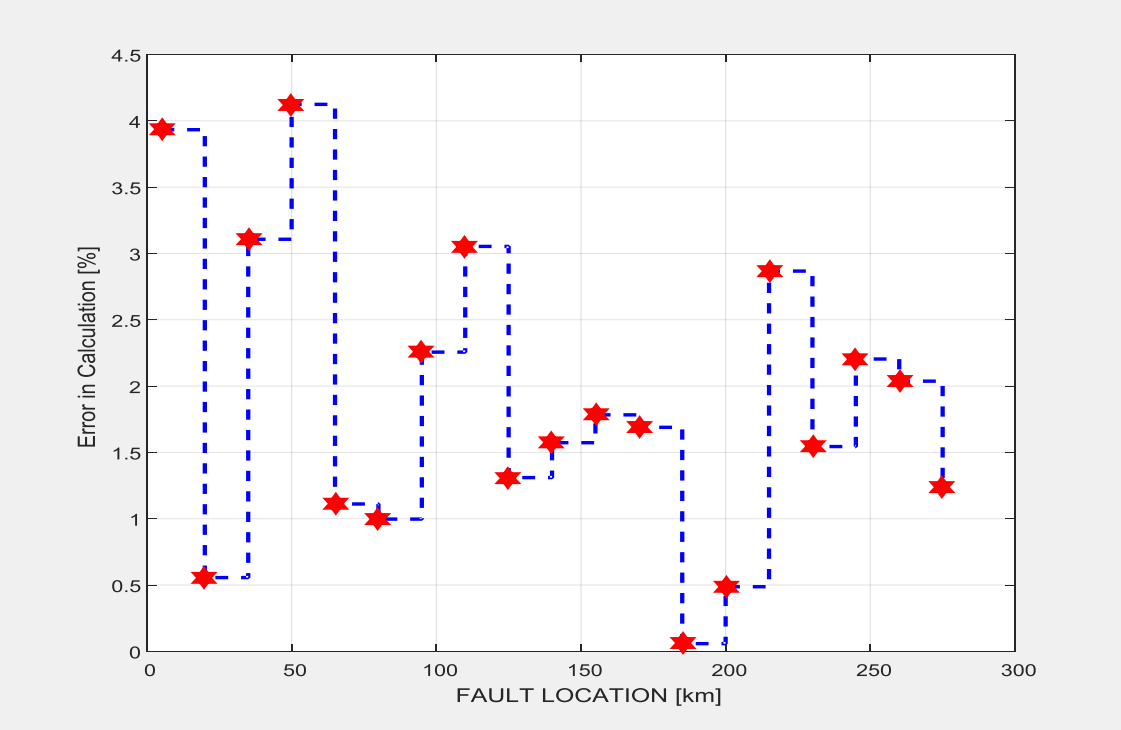


Figure 4.47: Test Phase performance of the ANN (3.20.15.1).

Fig 4.48 shows the performance of the neural network (in terms of training, testing and validation) with 3 neurons in the input layer, 4 hidden layers with 21, 35, 15 and 7 neurons in them respectively and 1 neuron in the output layer (3.21.35.15.7.1). It can be seen that the best Cross-Entropy performance of this neural network is 2.1645e-3at epoch 7which is below the Cross-Entropy goal of 1e-2 (denoted by the blue dotted line in the figure). It was found that the correlation coefficient between the outputs and the targets was 0.99329 for this neural network which indicates very good regression fit.

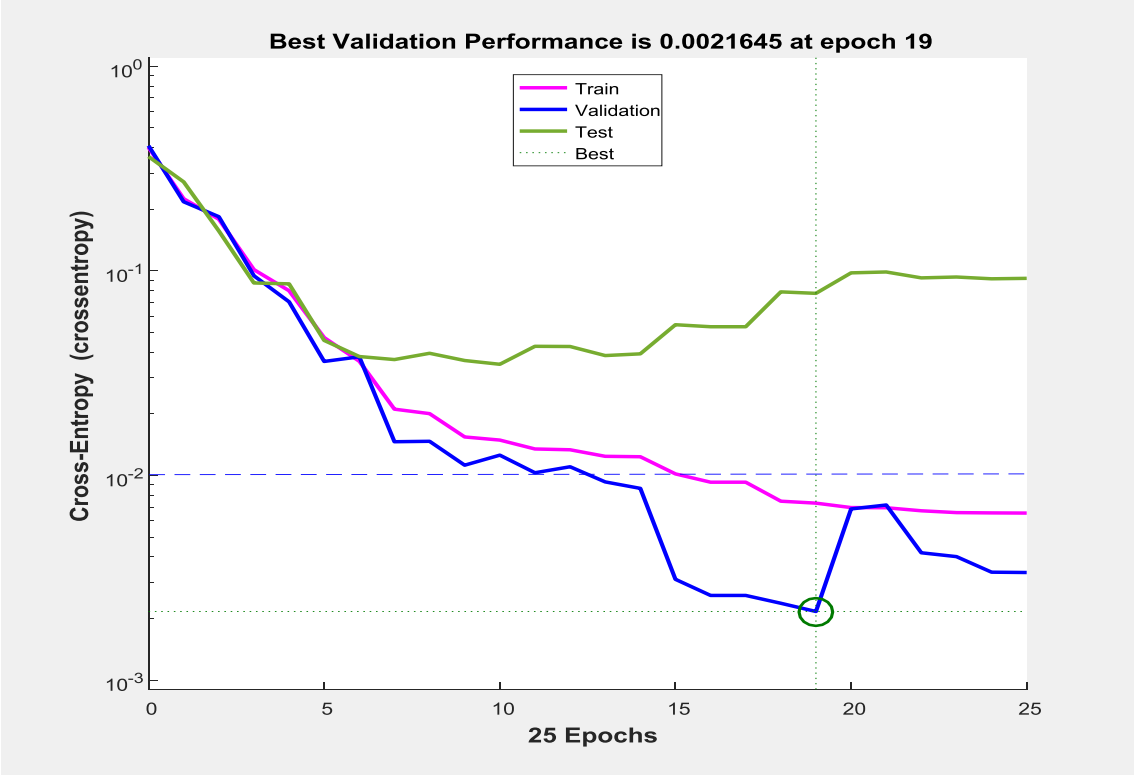


Figure 4.48: Mean Square Error performance of the neural network with configuration (3.21.35.15.7.1).

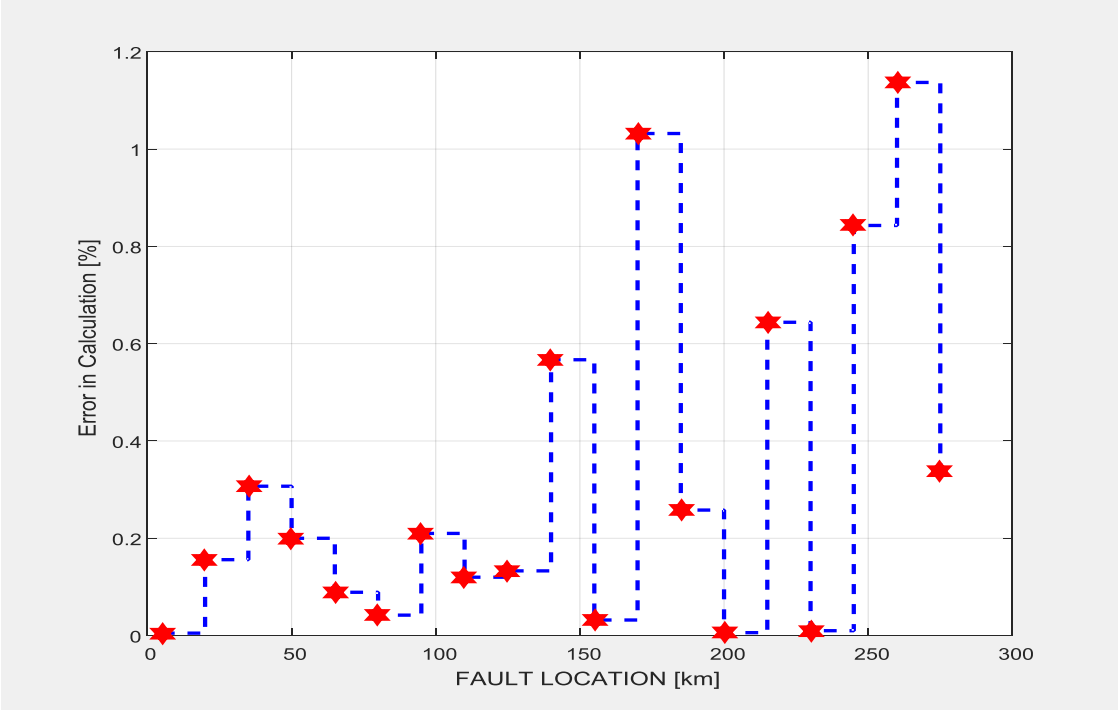


Figure 4.49: Test phase performance of the ANN (3.21.35.15.7.1).

In order to test the performance of this network, 19 different double line – ground faults have been simulated on different phases with the fault distance being incremented by 15km in each case and the percentage error in calculated output has been calculated. Figure 4.49 shows the results of this test conducted on the neural network (3.21.35.15.7.1). It can be seen that the maximum error is around 1.69% which is very satisfactory. It is to be noted that the average error in fault location is just 0.863 percent. Hence, this neural network has been chosen as the ideal network for the purpose of double line – ground fault location on transmission lines.

Figure 4.50 shows an overview of the chosen ANN and it can be seen that the training algorithm used is Levenberg - Marquardt algorithm. The performance function chosen for the training process is mean square error. Figure 4.51 plots the best linear regression fit between the outputs and the targets. As already mentioned, the correlation coefficient in this case is found to be 0.99329 which is very good.

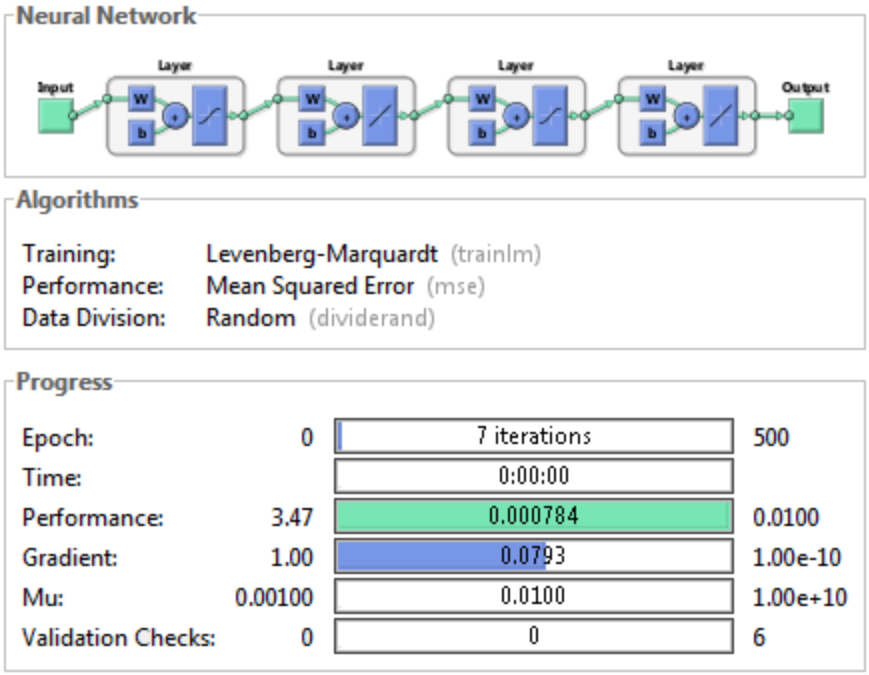


Figure 4.50: Overview of the chosen ANN (3.21.35.15.7.1) for Double Line-Ground Faults.

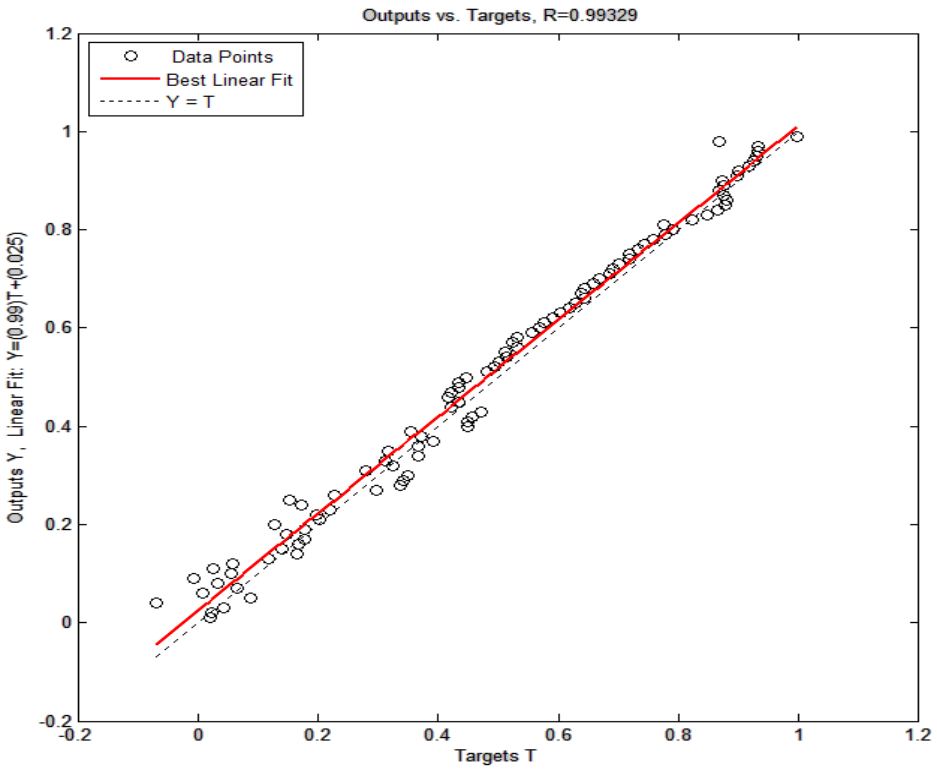


Figure 4.51: Regression fit of the outputs versus targets with configuration(3.21.35.15.7.1).

4.3.3b Simulation Results of Testing the Neural Network for Double Line – Ground Fault Location

Now that the neural network has been trained, the next important step is to analyze the performance of this network which is called testing. The methods and means by which this neural network has been tested are discussed here under. One important factor that helps test the network is the test phase performance plot as shown in Fig 4.49. It is to be noted that both the average as well as the maximum error percentages are in acceptable levels and hence the networks performance is satisfactory. Another means of determining the efficiency of a trained neural network is to check the gradient and validation performance plot as shown in Fig 4.52. It can be seen that there is a steady decrease in the gradient and also that the maximum number of validation fails is 3 during the training process. This indicates efficient training because the validation phase follows the test phase closely if the number of validation

fails is low. This further implies that the neural network can generalize new data fed into it more effectively.

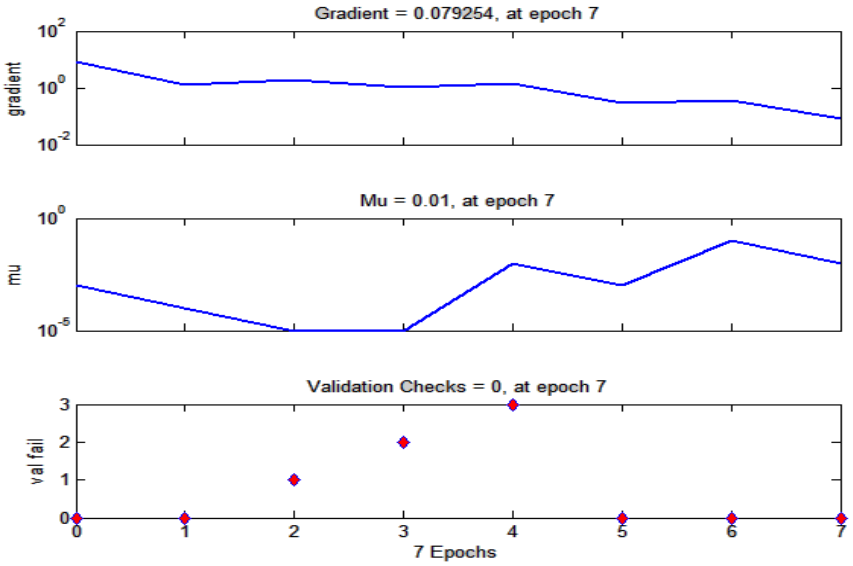


Figure 4.52: Gradient and validation performance plot of ANN with configuration (3.21.35.15.7.1).

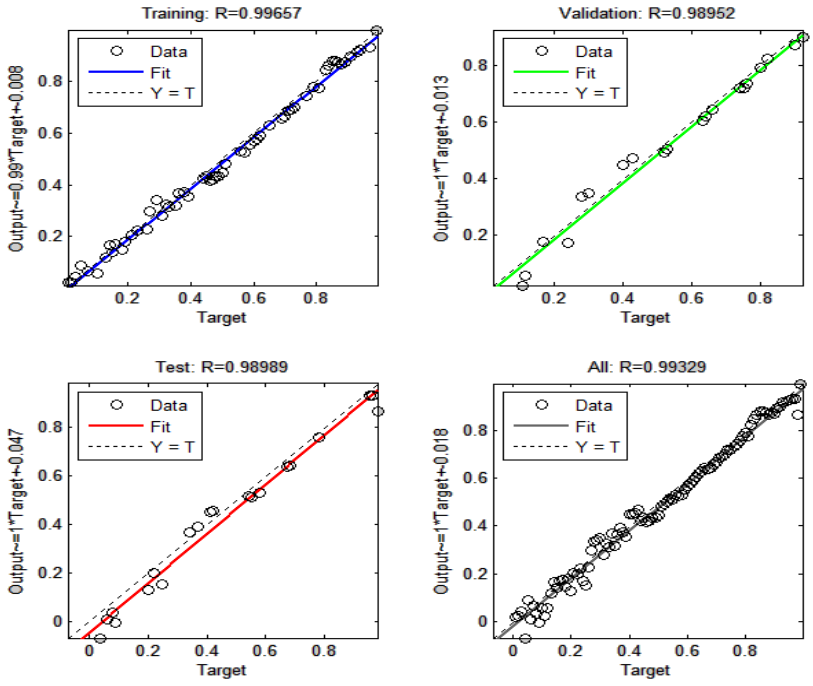


Figure 4.53: Regression plots of the various stages of learning of ANN (3.21.35.15.7.1).

The third factor that is considered while evaluating the performance of the network is the correlation coefficient of each of the various phases of training, validation and testing. Figure 4.53 shows the regression plots of the various phases such as training, testing and validation. It can be seen that the best linear fit very closely matches the ideal case with an overall correlation coefficient of 0.99329.

Figure 4.54 shows the structure of the chosen ANN for double line - ground fault location with 3 neurons in the input layer, 4 hidden layers with 21, 35, 15 and 7 neurons in them respectively and 1 neuron in the output layer (3.21.35.15.7.1). This is a pictorial representation of how the neurons in the respective layers are connected together through the synaptic weights. It shows the interconnections between the input layer and the hidden layers, and also between the hidden layers and the output layer. It can be seen that any given neuron is connected to all the neurons in the layer in front.

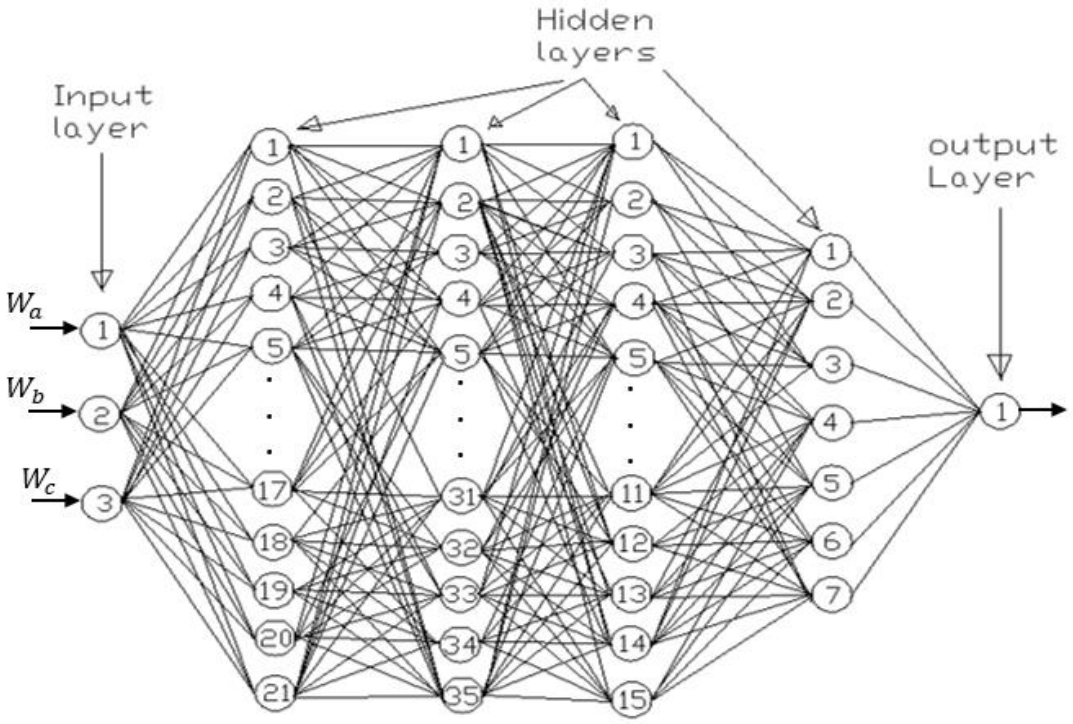


Figure 4.54: Structure of the chosen ANN (3.21.35.15.7.1).

Table 4.3 illustrates the percentage errors in fault location as a function of Fault Distance and Fault Resistance. Two different cases have been considered (shown in adjacent columns), one with a fault resistance of 15 Ohms and another with a fault resistance of 70 Ohms.

Table 4.3: Percentage errors as a function of fault distance and fault resistance for the ANN chosen for double line - ground fault location.

Serial No:	% Error vs. Fault Distance (Fault Resistance = 15 Ω)			% Error vs. Fault Distance (Fault Resistance = 70 Ω)		
	Fault Distance (km)	Estimated Fault Location	Percentage Error	Fault Distance (km)	Estimated Fault Location	Percentage Error
1	5	4.92	1.60	10	10.08	0.08
2	20	20.07	0.35	25	25.11	0.44
3	35	35.23	0.66	40	40.03	0.08
4	50	50.21	0.42	55	55.41	0.75
5	65	65.12	0.18	70	70.15	0.21
6	80	80.67	0.84	85	85.18	0.21
7	95	95.41	0.43	100	100.77	0.77
8	110	110.27	0.25	115	116.67	1.45
9	125	126.34	1.07	130	128.30	1.31
10	140	141.55	1.11	145	146.17	0.81
11	155	153.87	0.73	160	161.02	0.64
12	170	171.32	0.78	175	174.21	0.45
13	185	186.23	0.66	190	191.55	0.82
14	200	201.83	0.92	205	207.22	1.08
15	215	216.76	0.82	220	221.68	0.76
16	230	232.00	0.87	235	237.02	0.86
17	245	243.00	0.82	250	252.01	0.80
18	260	261.68	0.65	265	267.02	0.76
19	275	273.01	0.72	280	282.00	0.71

It is to be noted that the resistance of 15 Ohms was used as a part of training data set and hence the average percentage error in fault location in this case is 0.7305%. The second case illustrates the same with a different fault resistance of 70 Ohms which is relatively very high and is not a part of the training set. Hence, the performance of the neural network in this case illustrates its ability to generalize and react upon new data. It is to be noted that the average error in this case is 0.6837% which is still acceptable. Thus, the neural networks performance is considered satisfactory and can be used for the purpose of double line – ground fault location.

4.3.4 Three Phase Faults

The results of the design, development and performance of neural networks for the purpose of three-phase fault location are discussed in this section. The fourth and the final category of faults are the three phase faults. There exists only one kind of three phase faults which is denoted as ABC fault where in all the three phases A, B and C are faulted.

4.3.4a Simulation Results of Training the Neural Network for Three Phase Fault Location

A few neural networks that achieved satisfactory performance are presented first along with their error performance plots. Of these ANNs, the most appropriate ANN is chosen based on its Cross-Entropy and the Regression coefficient of the Outputs vs. Targets. Figures 4.55 – 4.57show the Cross-Entropy and the Test phase performance plots of the neural network 3.16.10.1 with 2 hidden layers. Figures 4.58 –4.60 show the Cross-Entropy and the Test phase performance plots of the neural network 3.24.1 with 1 hidden layer which gives unsatisfactory performance. After exhaustive survey, the chosen neural network has five hidden layers with 6 neurons in the first hidden layer, 21 neurons in the second hidden layer, 16 neurons in the

third hidden layer, 10 neurons in the fourth hidden layer and 5 neurons in the fifth hidden layer (3.6.21.16.10.5.1).

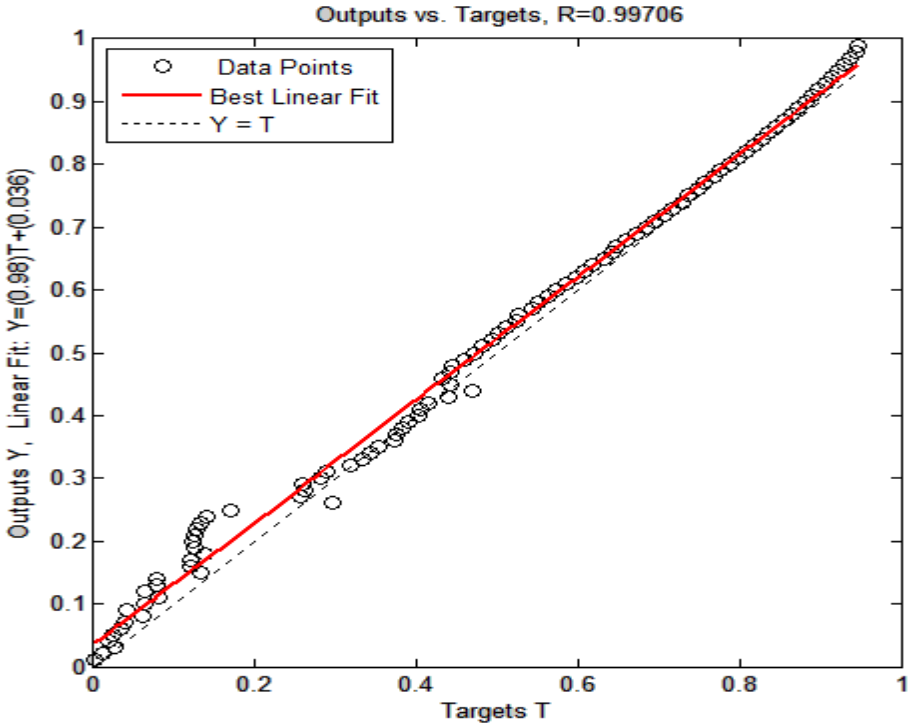


Figure 4.55: Regression fit of the outputs versus targets of ANN with configuration (3.16.10.1).

Figure 4.55 plots the best linear regression fit between the outputs and the targets of the neural network with 3 neurons in the input layer, 2 hidden layers with 16 and 10 neurons in them respectively and 1 neuron in the output layer (3.16.10.1). The correlation coefficient (r) as mentioned earlier is a measure of how well the neural network relates the outputs and the targets. The closer the value of r is, to 1, the better the performance of the neural network. The value of r in this case is found to be 0.87906. Therefore, the training is considered unsuccessful.

Figure 4.56 shows the performance of the neural network (in terms of training, testing and validation) with 3 neurons in the input layer, 2 hidden layers with 17 and 5 neurons in them

respectively and 1 neuron in the output layer (3.17.5.1). It can be seen that the best Cross-Entropy performance of this neural network is 3.0847e-2 which is below the Cross-Entropy goal of 1e-2 (denoted by the blue line).

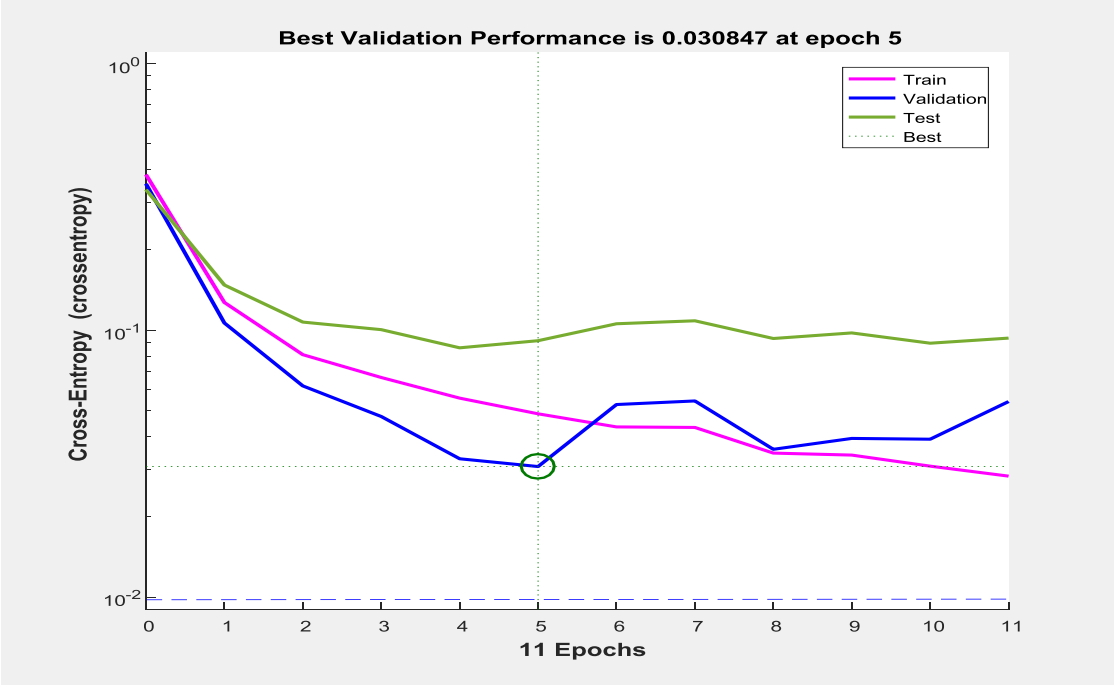


Figure 4.56: MSE performance of the neural network with configuration (3.17.5.1).

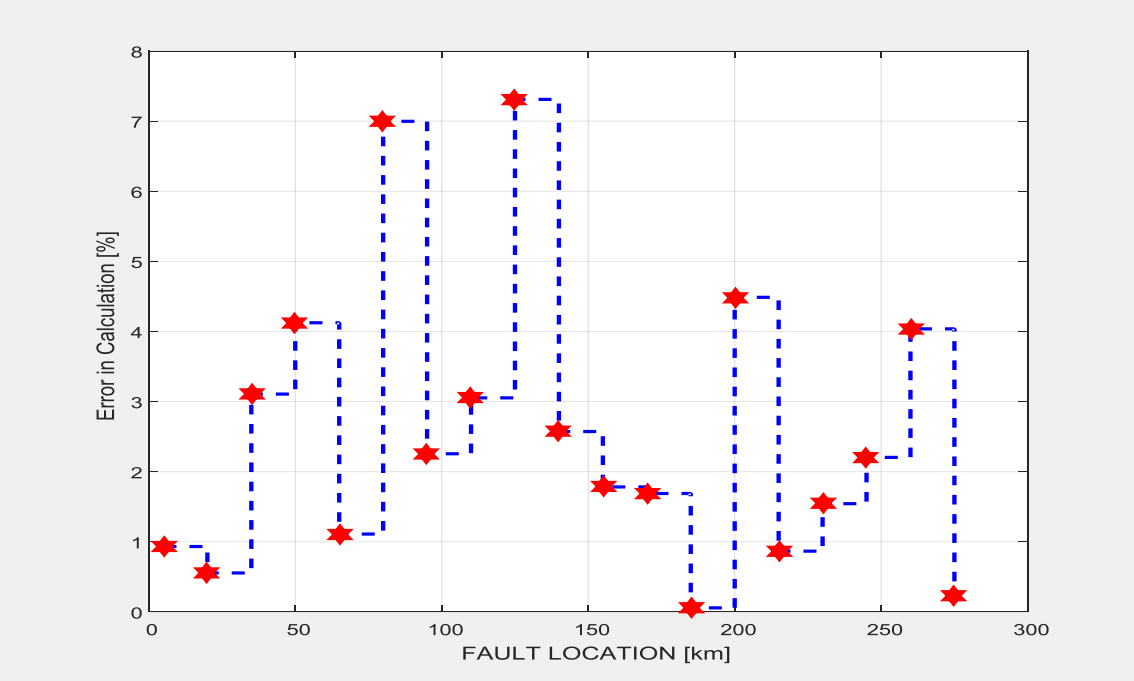


Figure 4.57: Test Phase performance of the ANN with configuration (3.17.5.1).

In order to test the performance of this network, 19 different three phase faults have been simulated on the transmission line with the fault distance being incremented by15km in each case and the percentage error in ANN's output has been calculated. Figure4.57 shows the results of this test conducted on the neural network (3.17.5.1). It can be seen that the maximum error is higher than 4.38% which is fairly satisfactory. However neural networks that can perform better are more desirable.

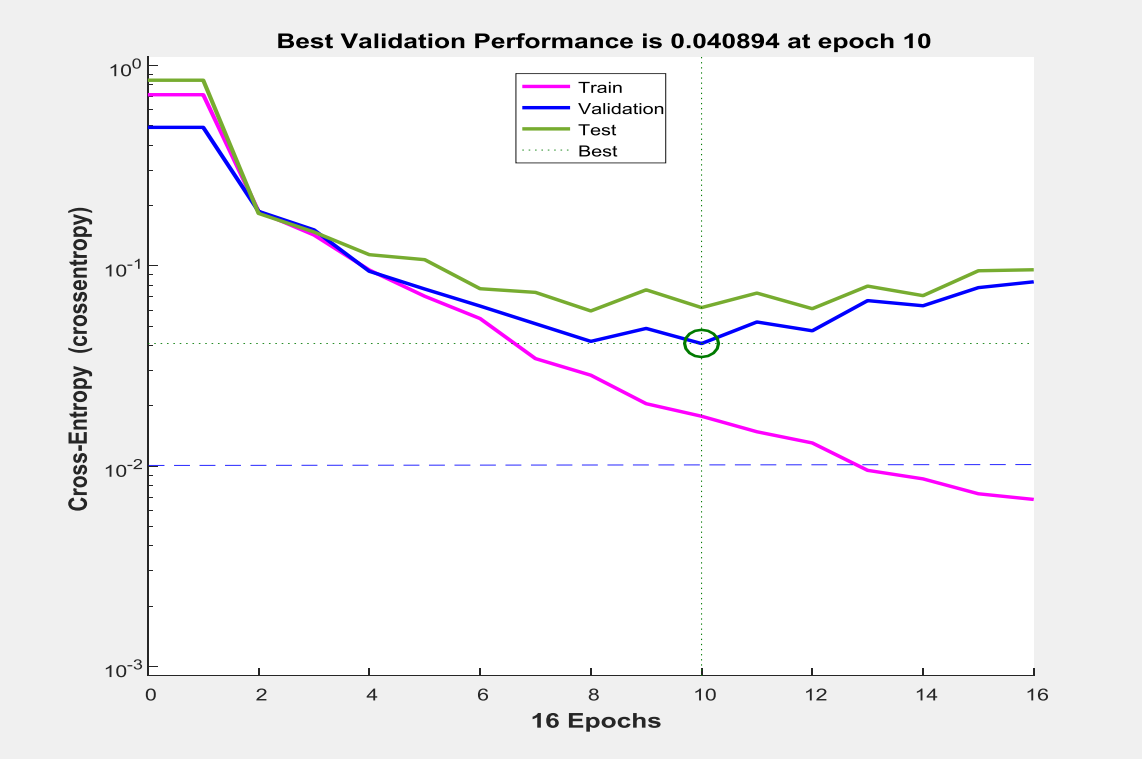


Figure 4.58: MSE performance of the neural network with configuration (3.24.1).

Figure 4.58 shows the performance of the neural network (in terms of training, testing and validation) with 3 neurons in the input layer, 1 hidden layer with 24 neurons in it and 1 neuron in the output layer (3.24.1). It can be seen that the best Cross-Entropy performance of this

neural network is 4.0894e-2 which is below the Cross-Entropy goal of 1e-2 (denoted by the blue dotted line).

Figure 4.59 plots the best linear regression fit between the outputs and the targets of the neural network with 3 neurons in the input layer, 1 hidden layer with 24 neurons in it and 1 neuron in the output layer (3.24.1). The correlation coefficient (r) as mentioned earlier is a measure of how well the neural network relates the outputs and the targets. The closer the value of r is, to 1, the better the performance of the neural network. The value of r in this case is found to be 0.99804 which is an improvement from the previous case (3.17.5.1) but still not satisfactory.

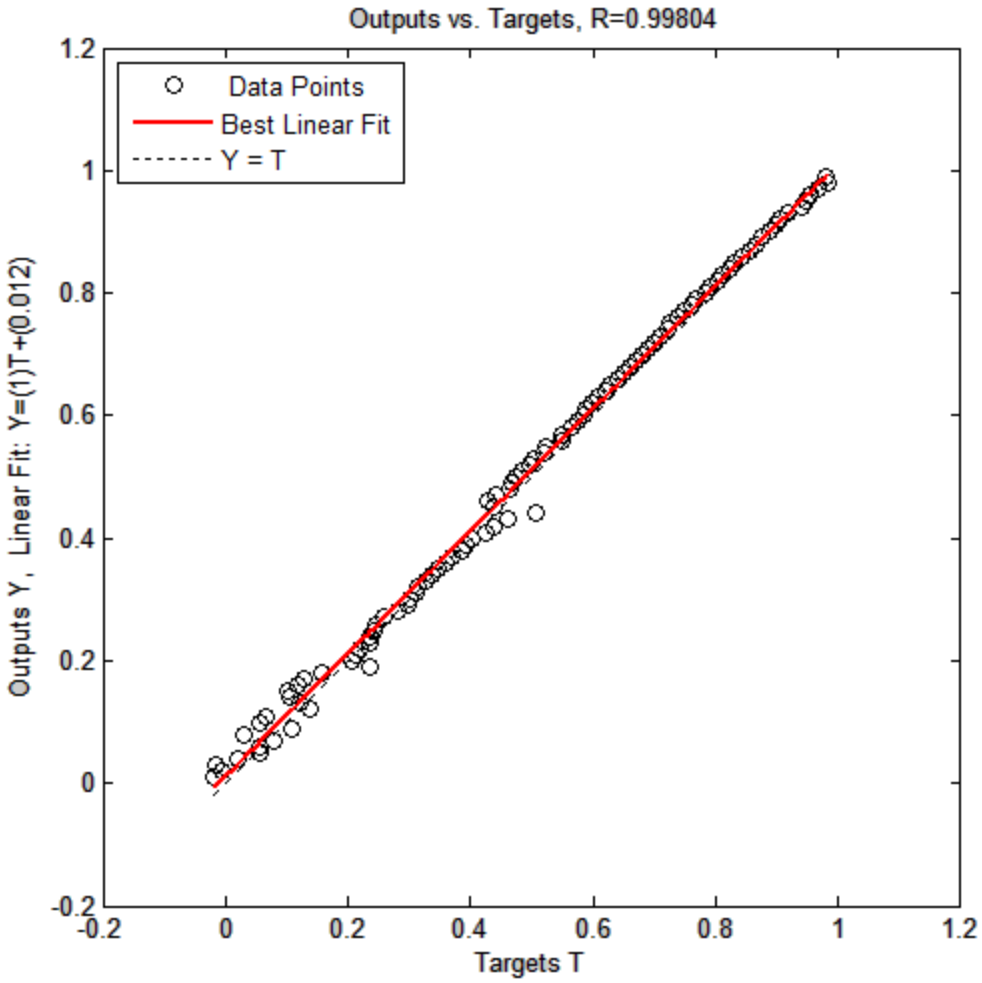


Figure 4.59: Regression fit for the outputs versus targets of ANN with configuration (3.24.1).

In order to test the performance of this network, 19 different three phase faults have been simulated on the transmission line with the fault distance being incremented by15km in each case and the percentage error in ANN's output has been calculated. Figure 4.60 shows the results of this test conducted on the neural network (3.24.1). It can be seen that the maximum error is just lower than 3.1% which is a significant improvement from the previous case but still need improvement.

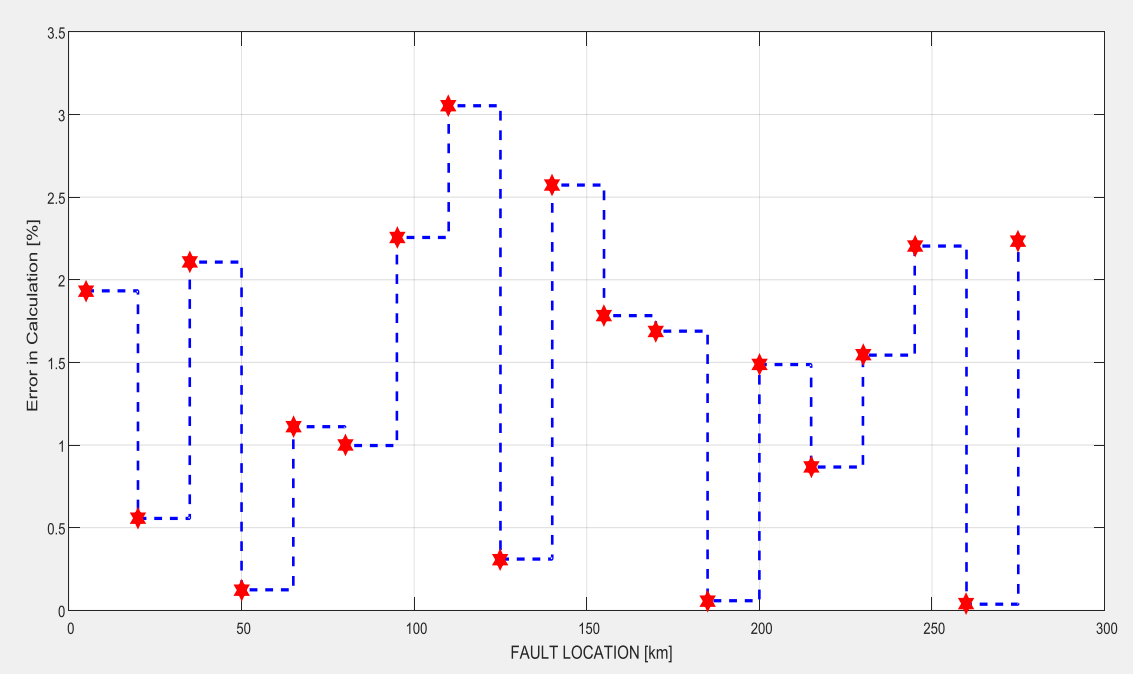


Figure 4.60: Test Phase performance of the ANN with configuration (3.24.1).

Figure 4.61 plots the best linear regression fit between the outputs and the targets of the neural network with 3 neurons in the input layer, 3 hidden layers with 6, 21, 16, 10 and 5 neurons in them respectively and 1 neuron in the output layer (3.6.21.16.10.5.1). The correlation

coefficient (r) as mentioned earlier is a measure of how well the neural network relates the outputs and the targets. The closer the value of r is, to 1, the better the performance of the neural network. The value of r in this case is found to be 0.99897 which is very close to 1.

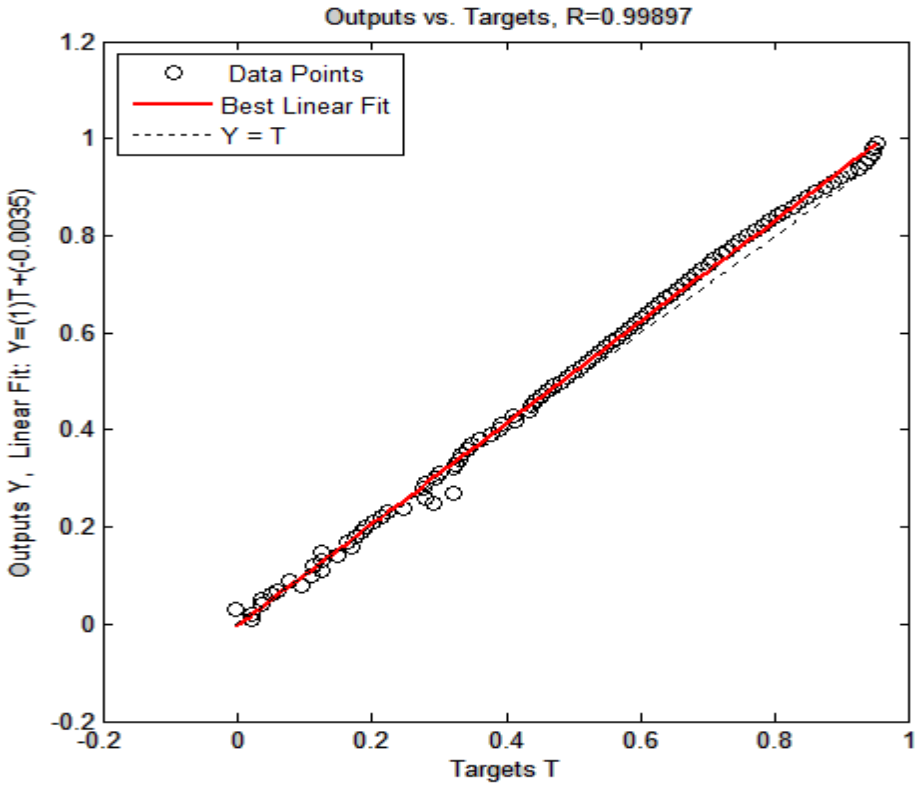


Figure 4.61: Regression fit of the outputs versus targets of ANN (3.6.21.16.10.5.1).

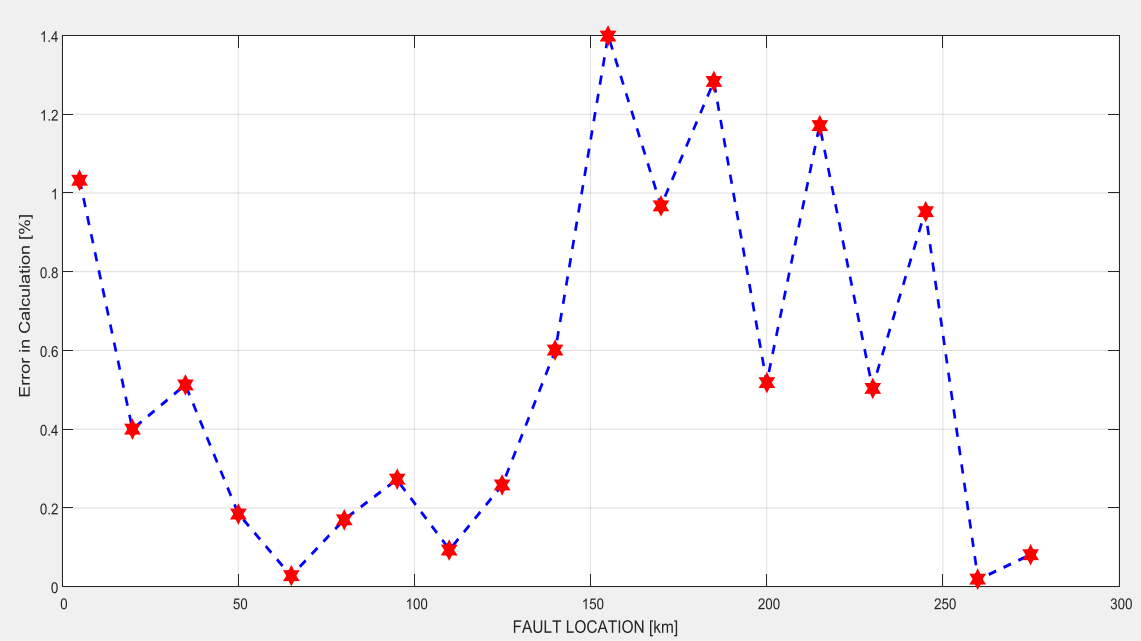


Figure 4.62: Test Phase performance of the ANN (3.6.21.16.10.5.1).

In order to test the performance of this network, 19 different three phase faults have been simulated on the transmission line with the fault distance being incremented by 15 km in each case and the percentage error in ANN's output has been calculated. Figure 4.62 shows the results of this test conducted on the neural network (3.6.21.16.10.5.1). It can be seen that the maximum error is around 1.4% which is very satisfactory. It is to be noted that the average error in fault location is 0.677%. Hence, this neural network has been chosen as the ideal network for the purpose of three phase fault location on transmission lines.

Figure4.63shows an overview of the chosen ANN and it can be seen that the training algorithm used is Levenberg - Marquardt algorithm. The performance function chosen for the training process is mean square error.

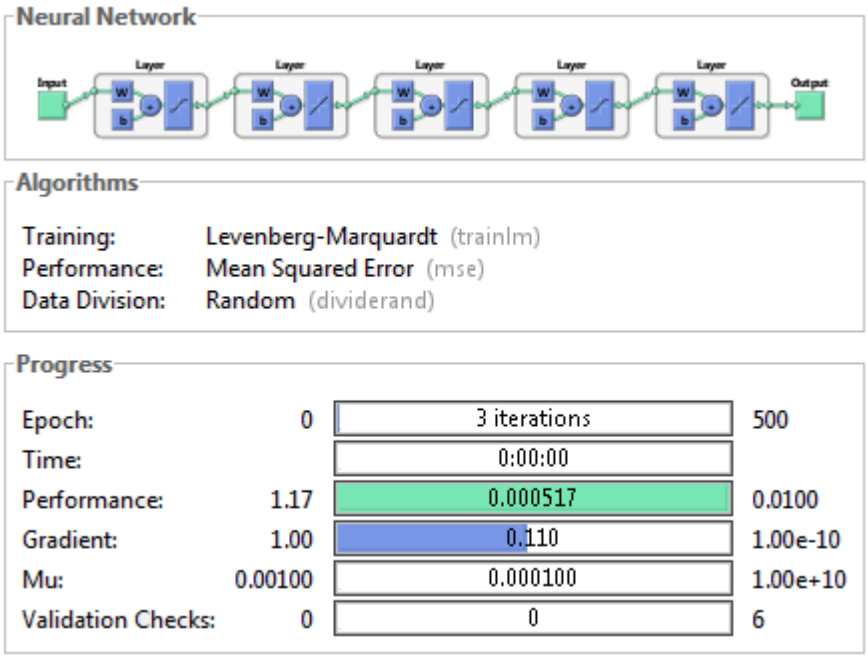


Figure 4.63: Overview of the chosen neural network for three phase fault location.

Figure4.64shows the performance of the neural network (in terms of training, testing and validation) with 3 neurons in the input layer, 4 hidden layers with 6, 21, 16, 10 and 5 neurons in them respectively and 1 neuron in the output layer (3.6.21.16.10.5.1). It can be seen that the best Cross-Entropy performance of this neural network is 1.9196e-3 (denoted by the dotted green line) which is below the Cross-Entropy goal of 1e-2 (denoted by the blue dotted line).

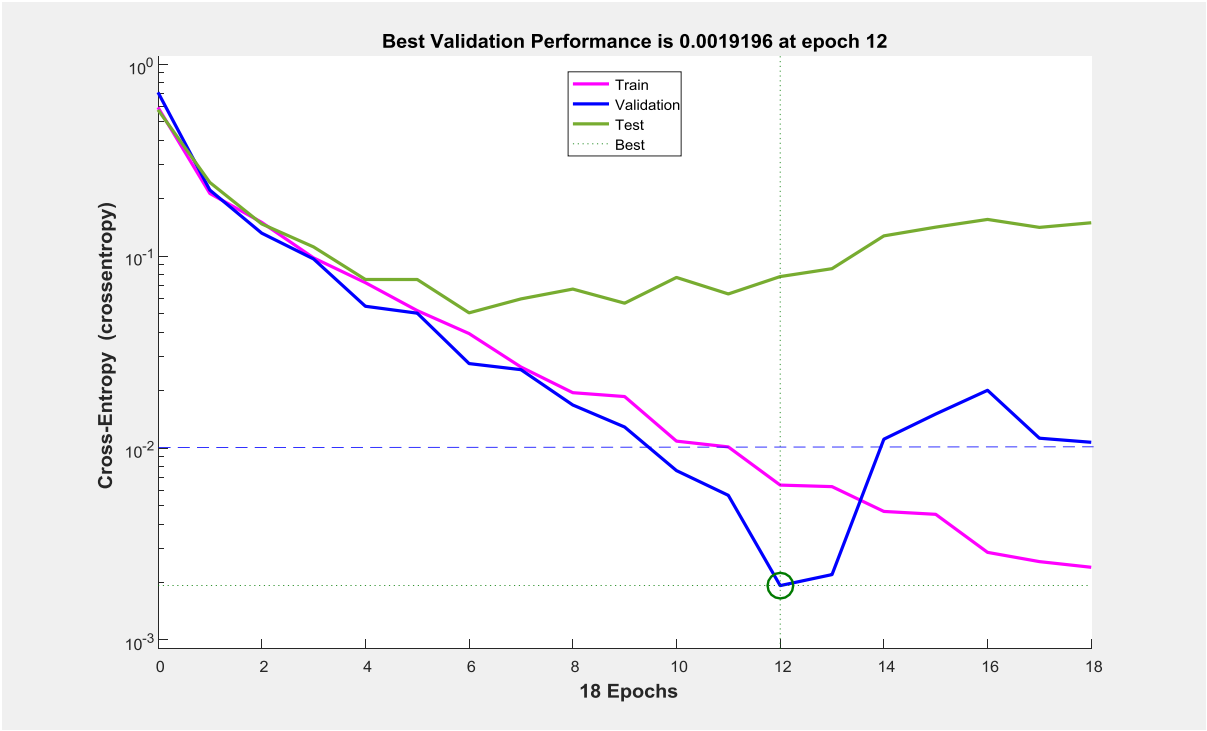


Figure 4.64: Mean Square Error performance of the neural network (3.6.21.16.10.5.1).

4.3.4b Simulation Results of Testing the Neural Network for Three Phase Fault Location

Now that the neural network has been trained, the next step is to analyze the performance of this network which is called testing. The results of the methods and means by which this neural network has been tested are discussed here in this section. One important factor that helps test the network is the test phase performance plot as shown in Fig 4.64. It is to be noted that both the average as well as the maximum error percentages in accurately determining the location of the fault are in acceptable levels and hence the network's performance is satisfactory.

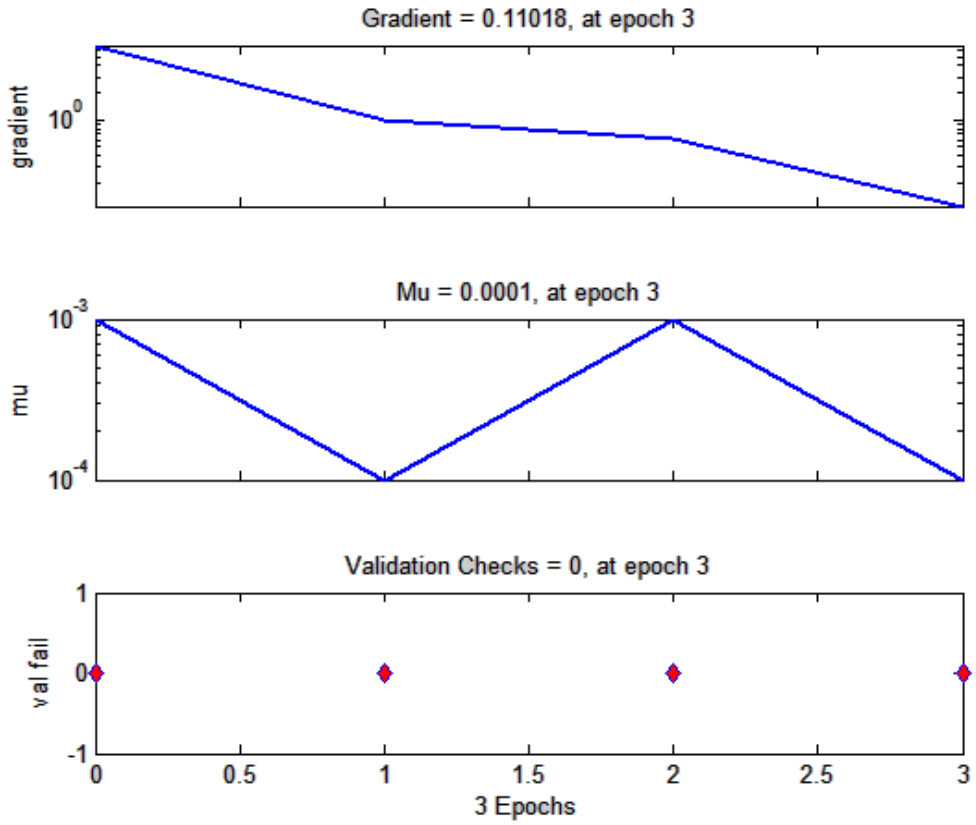


Figure 4.65: Gradient and validation performance plots of the ANN (3.6.21.16.10.5.1).

Another important means of determining the efficiency of a trained neural network is to check the gradient and validation performance plot as shown in Figure4.65. It can be seen that there is a steady and smooth decrease in the gradient and also that the maximum number of validation fails is 0 during the training process. This indicates efficient training because the validation phase follows the test phase closely if the number of validation fails is low. This is further indicated by the test and validation curves on Figure 4.66. This further implies that the neural network can generalize new data fed into it more effectively.

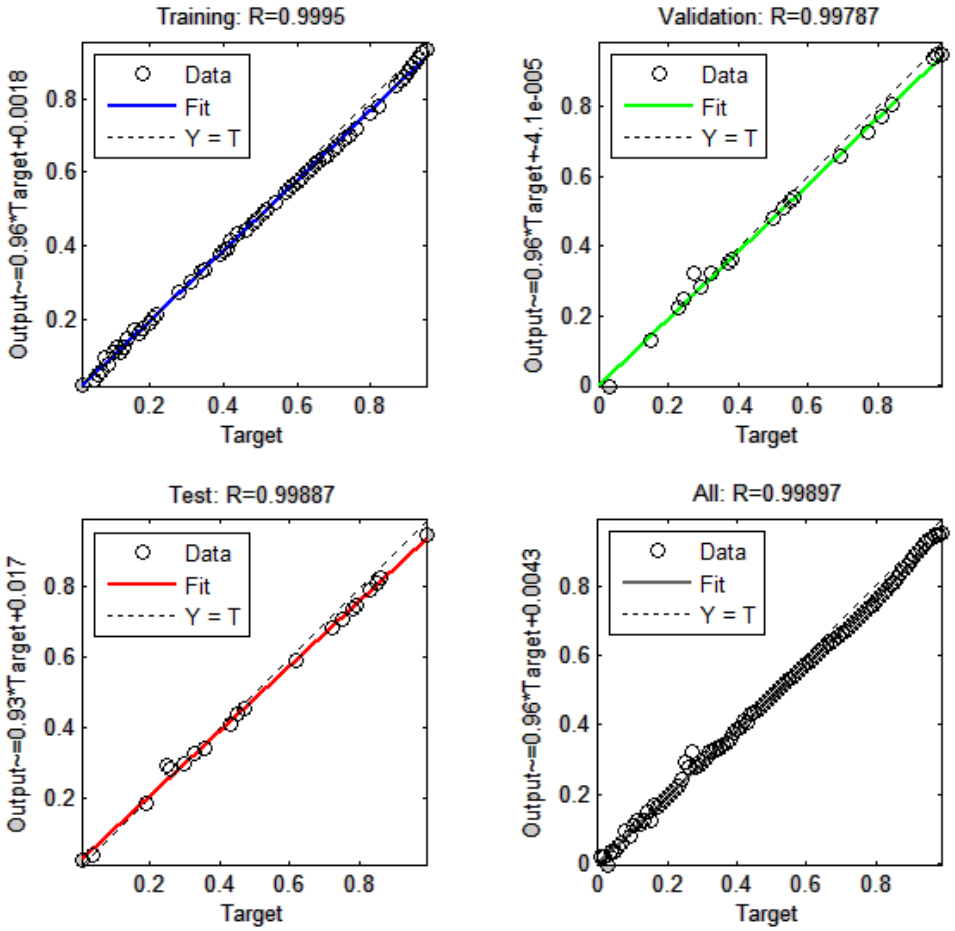


Figure 4.66: Regression plots of the various phases of learning of the ANN (3.6.21.16.10.5.1).

The third factor that is considered while evaluating the performance of the network is the correlation coefficient of each of the various phases of training, validation and testing. Figure4.66shows the regression plots of the various phases such as training, testing and validation. It can be seen that the best linear fit very closely matches the ideal case with an overall correlation coefficient of 0.99897.

Figure4.67shows the structure of the chosen ANN for three-phase faults with 3neurons in the input layer, 5 hidden layers with 6, 21, 16, 10 and 5 neurons in them respectively and 1 neuron in the output layer (3.6.21.16.10.5.1).

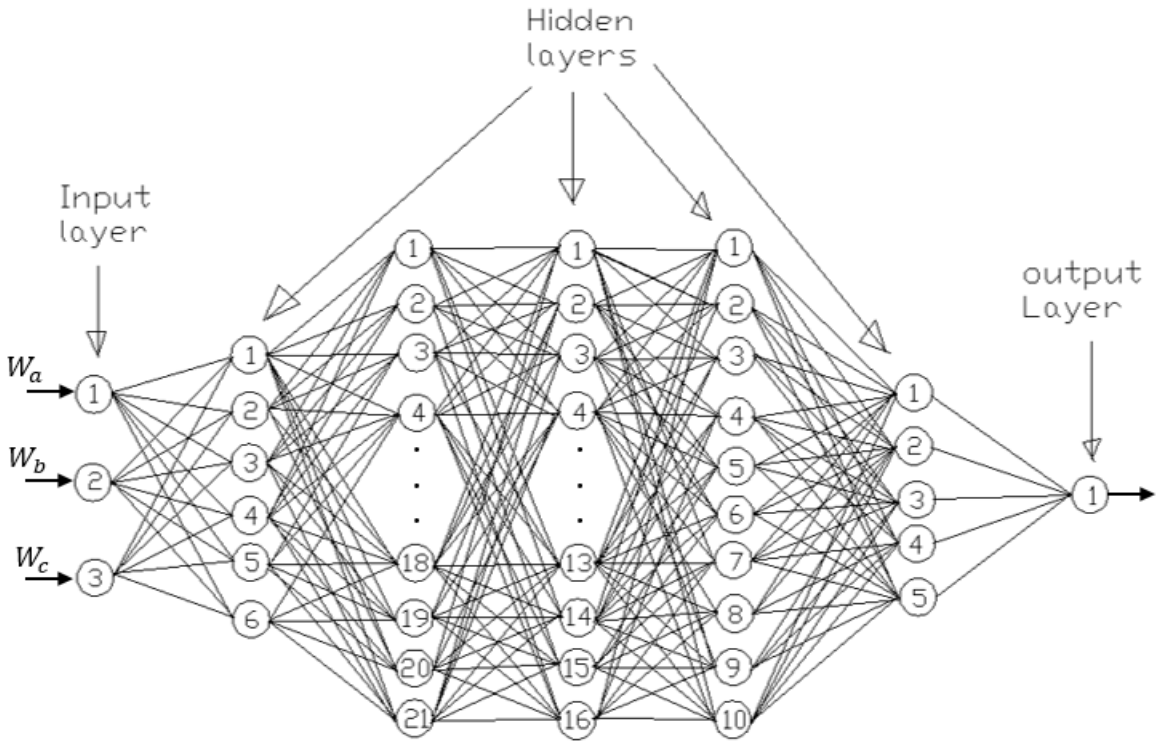


Figure 4.67: Structure of the chosen ANN (3.6.21.16.10.5.1).

Table 4.4 illustrates the percentage errors in Fault location as a function of Fault Distance and Fault Resistance. Two different cases have been considered (shown in adjacent columns), one with a fault resistance of 15 Ohms and another with a fault resistance of 70 Ohms. It is to be noted that the resistance of 15 Ohms was used as a part of training data set and hence the average percentage error in fault location in this case is 0.6447%. The second case illustrates the same with a different fault resistance of 70 Ohms which is relatively very high and is not a part of the training set. Hence, the performance of the neural network in this case illustrates its ability to generalize and react upon new data. It is to be noted that the average error in this case is 0.6721% which is still acceptable. Thus, the neural networks performance is considered satisfactory and can be used for the purpose of three phase fault location.

Table 4.4: Percentage errors as a function of fault distance and fault resistance for the ANN chosen for three phase fault location.

Serial No:	% Error vs. Fault Distance (Fault Resistance = 15 Ω)			% Error vs. Fault Distance (Fault Resistance = 70 Ω)		
	Fault Distance (km)	Estimated Fault Location	Percentage Error	Fault Distance (km)	Estimated Fault Location	Percentage Error
1	5	5.03	0.60	10	10.12	1.20
2	20	20.18	0.90	25	25.20	0.80
3	35	35.16	0.46	40	39.96	0.10
4	50	50.09	0.18	55	55.46	0.84
5	65	64.11	1.37	70	70.23	0.33
6	80	80.28	0.35	85	84.11	1.05
7	95	95.53	0.56	100	99.46	0.54
8	110	110.10	0.09	115	114.79	0.18
9	125	125.66	0.53	130	131.72	1.32
10	140	140.83	0.59	145	146.22	0.84
11	155	154.03	0.63	160	161.21	0.76
12	170	171.22	0.72	175	174.00	0.57
13	185	185.73	0.39	190	191.32	0.69
14	200	202.10	1.05	205	206.71	0.83
15	215	216.37	0.64	220	221.14	0.52
16	230	231.78	0.77	235	236.12	0.48
17	245	246.33	0.54	250	251. 32	0.53
18	260	258.00	0.77	265	265.88	0.33
19	275	278.00	1.09	280	277.60	0.86

4.4 Validation Check

As a way of validatory check on the results obtained in this work, effort has been made to compare the results to those obtained by authors whose works were among those reviewed in section 2.13.

Table 4.5: Percentage errors as a function of fault distance for single line to ground fault location. Copied from Mamta and Patel,(2012)

Type of Fault	Location of Fault (kM)	Wavelet and ANN based Fault Locator Output (kM)	Percentage Error
LG-A	55	53.9622	-1.88
	78	77.2255	-0.99
	92	92.0286	0.09
	134	139.2605	3.9
	167	176.5844	5.74
LG-B	28	27.4472	-1.97
	87	90.3275	3.82
	103	103.2484	0.241
	145	148.303	2.27
	176	189.379	7.6
LG-C	15	13.6723	-8.85
	38	36.9657	-2.72
	97	95.8447	1.19
	116	113.323	-2.3
	182	179.2931	-1.48

For the results of single line to ground fault location obtained in this work, when compared to the work done by Mamta and Patel,(2012) which is among the works reviewed in the related literatures, the percentage error margins were significantly reduced. The average percentage errors recorded by Mamta and Patel,(2012) for line – ground faults on phase A, B and C were

2.52%, 3.18% and 3.31% respectively as calculated from Table 4.5 whereas in this work an average percentage error of 0.65% and 0.7284% were achieved for 15 Ohms and 70 Ohms fault resistances respectively as aforementioned, thereby confirming the potency of the method proposed in this work.

For the results of line to line fault location, when compared to the work done by Girish and Nitin,(2015) which is among the works reviewed in the related literatures, the percentage error margins were significantly reduced. The average percentage error recorded by Girish and Nitin,(2015) was 3.35% as also inferred from Table 4.6 whereas in this work an average percentage error of 0.5811% and 0.7611 % were achieved for 15 Ohms and 70 Ohms fault resistances respectively as aforementioned, thereby confirming the potency of the method proposed in this work.

Table 4.6: Percentage errors as a function of fault distance for line to line fault location. Copied from Girish and Nitin,(2015)

S/No	Fault Distance (kM)	Measured Fault Distance (kM)	Percentage Error
1	50	41.61	2.79
2	100	89.97	3.34
3	150	138.69	3.77
4	200	190	3.34
5	250	237.875	4.04
6	300	291.6	2.8

For the results of double line to ground fault location, (Ayyagari, 2011) in their work reported an average percentage error of 1.122% as confirmed by Table 4.7. But in this work an average percentage error of 0.7305% and 0.6837% were achieved for 15 Ohms and 70 Ohms fault

resistances respectively. It can also be seen that the percentage error margins were significantly reduced., thereby confirming the potency of the method proposed in this work.

Table 4.7: Percentage errors as a function of fault distance for double line to ground fault location. Copied from (Ayyagari, 2011)

S/No	Fault Distance (kM)	Measured Fault Distance (kM)	Percentage Error
1	50	51.17	0.39
2	100	102.52	0.84
3	150	153.63	1.21
4	200	201.98	0.66
5	250	255.19	1.73

For the results of three phase fault location, (Ayyagari, 2011) also in their work reported an average percentage error of 0.836% as confirmed by Table 4.8. But in this work an average percentage error of 0.6437% and 0.6721 % were achieved for 15 Ohms and 70 Ohms fault resistances respectively. It can also be seen that the percentage error margins were significantly reduced, which again confirms the potency of the method proposed in this work.

Table 4.8: Percentage errors as a function of fault distance for three phase fault location. Copied from (Ayyagari, 2011)

S/No	Fault Distance (kM)	Measured Fault Distance (kM)	Percentage Error
1	50	51.41	0.47
2	100	103.03	1.01
3	150	152.37	0.79
4	200	201.99	0.63
5	250	253.84	1.28

CHAPTER FIVE

CONCLUSION, FINDINDS AND RECOMMENDATION

5.1 Conclusions

This dissertation has implemented the usage of neural networks as an alternative and effective method for the detection, classification and location of faults on transmission lines. The methods employed, make use of the summation of the wavelet decomposed fault voltage and current values of all the three phases as inputs to the neural networks. Various possible kinds of faults namely single line-ground, line-line, double line-ground and three phase faults have been taken into consideration in this work and separate ANNs have been implemented for each of these faults.

All the neural networks used in this dissertation belong to the back-propagation neural network architecture with either Levenberg-Marquardt or Scaled Conjugate Gradient algorithm. A fault location scheme for the transmission line system, right from the detection of faults on the line to the fault location stage has been devised successfully by using artificial neural networks.

The simulation results obtained proved that satisfactory performance has been achieved by all of the proposed neural networks in general. As further illustrated, depending on the application of the neural network and the size of the training data set, the size of the ANN (the number of hidden layers and number of neurons per hidden layer) keeps varying. The importance of choosing an appropriate ANN configuration, in order to get the best performance from the network, has been stressed upon in this work. The sampling frequency adopted for sampling the voltage and current waveforms in this work is just 10 kHz which is

very low compared to what has been used in the literatures (a major portion of the works in literature utilized 12 kHz – 25 kHz).

This is of significant importance because, the lower the sampling frequency, the lesser the computational burden on the industrial computer that uses the neural networks. This means a lot of energy savings because a continuous online detection scheme of this kind consumes a large amount of energy, a major portion of which is due to the continuous sampling of waveforms. The above mentioned are some significant improvements that this work offers over existing neural network based techniques for transmission line fault location.

Matlab R2016a along with the SimPowerSystems toolbox in Simulink was used to simulate the entire power transmission line model and to obtain the training data set. In order to train and analyze the performance of the neural networks, the Artificial Neural Networks Toolbox was used extensively.

5.2 Findings

Some important findings that can be drawn from this dissertation are:

- Neural Networks are indeed a reliable and attractive scheme for an ideal transmission line fault location scheme especially in view of the increasing complexity of the modern power transmission systems.
- It was very essential to investigate and analyze the advantages of a particular neural network structure and learning algorithm before choosing it for an application because there should be a trade-off between the training characteristics and the performance factors of any neural network.

- The results obtained proved that satisfactory performance was achieved by all the modelled NN based fault detectors. Back-Propagation NNs were very efficient when sufficiently large training data set was available. Also, the low average percentage errors obtained showed that the NNs had a high degree of precision and accuracy in fault finding when compared to the other existing methods.

5.3 Recommendation

The following are the recommendations suggested. Firstly, ANN should be introduced in protectionof the transmission lines in Nigerian grid system as this will go a long way in faster response to repairs and restoration of the faulted transmission lines within the network.

Secondly, as there are still many areas of ANNapart from Multilayer perceptron to be explored like self-organizing feature map (SOFM or Kohonen) networks and radial basis neural (RBF) network as hinted in sub-section 2.10.2 and 2.10.3, power system protection researchers should channel their resources and time to this area since the results of this work showed that multi-layer perceptron ANN has high efficiency when it comes to fault detections and locations. As it would be quite useful to analyze all these possible neural network types and to provide a comparative analysis on each of the architectures and their performance characteristics.

Finally, Levenberg-Marquardt (LM) back-propagation algorithm is good when there is sufficient amount of training data-set as it provides very fast convergence when compared to other learning algorithms but when the NN size is enormously big, LM algorithm becomes slower as there would high increase in the Jacobian matrix to be computed. Effort, therefore,

should be made by researchers to improve further the convergence rate of LM algorithm under this condition.

5.4 Contribution to Knowledge

The contributions to knowledge are itemized as follows:

- ► The dissertation has successfully developed a multi-resolution analysis (MRA) and pattern recognition based fault identification and location in an overhead transmission line using the summation of the decomposed detail coefficients and approximations of the extracted faulty voltage and current waveforms for all the three phases for ten different faults and also non-fault case.

- ► The case study network (the Nigerian 330kV Ikeja West – Benin Power Transmission line system) has received simulation procedure and results specific to its parameters which enabled this work to explore the peculiarity of the Nigerian power system.

REFERENCE

Abdollahi, A., Seyedtabali, S. (2010). Transmission line fault location estimation by Fourier & Wavelet Transforms using ANN. The 4th International Power Engineering and Optimization Conf. (PEOCO2010), Shah Alam, Selanger, Malasia.

Aggarwal, R. K., Johns, A. T., Song, Y. H., Dunn, R. W., and Fitton, D. S. (March, 1994). Neural-network based adaptive single-pole auto-reclosure technique for EHV transmission systems. IEE Proc. on Generation, Transmission and Distribution, vol. 14, no. 2, pp. 155–160.

Aggarwal, R., Song, Y. (June, 1997).*Artificial neural network in power systems, I.General introduction to neural computing.* Power Engineering Journal, Vol 11, No.23 pp. 129-134.

Amit, M. P., and Abhijit, S. P. (2016).Fault Location on Transmission line using Wavelet Transform and Artificial Neural Network. IOSR Journal of Electricaland Electronics Engineering (IOSR-JEEE) e-ISSN: 2278-1676, p- ISSN: 2320-3331, Volume 11, Issue 4 Ver. II, PP 59-62. www.iosrjournals.org

Anamika, J., Thoke, A. S., and Patel, R. N. (2008). Fault Classification of Double Circuit Transmission Line Using Artificial Neural Network. International Journal of Electrical Systems Science and Engineering 1;4 © www.waset.org.

Anderson, J. A. (2003). *An Introduction to Neural Networks.*Prentice Hall.

Atul, A. K., and Navita, G. P. (2015). Fault Detection and Fault Classification of DoubleCircuit Transmission Line Using Artificial Neural Network. International Research Journal of Engineering and Technology (IRJET) e- ISSN: 2395 -0056 Volume: 02 Issue: 08 | www.irjet.net p-ISSN: 2395-0072.

Aurangzeb, M., Crossley, P. A., Gale, P. (2001).*Fault location using high frequency travelling waves measured at a single location on transmission line*. Proceedings of 7th International conference on Developments in Power System Protection – DPSP, IEE CP479, pp. 403-406.

Ayyagari, S. B. (2011). Artificial neural network based fault location for transmission lines. *University of Kentucky Master's Theses.* Paper 657. http://uknowledge.uky.edu/gradschool_theses/657

Bo, Z. Q. (1998). A new non-communication protection technique for transmission lines. IEEE Trans. Power Delivery, vol. 13, no. 4, pp. 1073–1078.

Bo, Z. Q., Jiang, F., Chen, Z., Dong, X., Weller, G., and Redfern, M. A. (2000). Transient based protection for power transmission systems. In Proc. IEEE PES Winter Meeting, Singapore, vol. 3, pp. 1832–1837.

Bo, Z. Q., Weller, G., Redfern, M. A. (1999). *Accurate fault location technique for distribution system using fault-generated high frequency transient voltage signals,*IEEE Proceedings of Generation, Transmission and Distribution 146(1), pp. 73-79.

Bouthiba T. (2004). Fault location in EHV transmission lines using artificial neural networks. Int. J. Appl. Math. Comput. Sci., Vol. 14, No. 1, pp. 69-78. 35. Sanaye-Pasand M, Kharashadi

Cichoki, A., Unbehauen, R. (1993). *Neural networks for optimization and signal processing*. John Wiley & Sons, Inc., New York.

Cook, V. (1996). *Fundamental aspects of fault location algorithms used in distance protection*. Proceedings of IEE Conference 133(6), pp. 359-368.

Coury, D.V., and Jorge. D.C. (1998). *Artificial neural network approach to distance protection of transmission lines.* IEEE Trans. on Power Delivery, vol. 13, No. pp 102-108.

Dalstein, T., Kulicke, B. (1995).*Neural network approach to fault classification for highspeed protective relaying.* IEEE Transactions on Power Delivery, vol. 4, pp. 1002 – 1009.

Das, R., Novosel, D. (2000). *Review of fault location techniques for transmission and sub – transmission lines*. Proceedings of 54th Annual Georgia Tech Protective Relaying Conference.

Das, D., Singh, N.K., Sinha, A.K. (2006). A Comparison Of Fourier Transform And Wavelet Transform Methods For Detection And Classification Of Faults On Transmission Lines. In Proc. of IEEE Power India Conf.

Djuric, M. B., Radojevic, Z. M., Terzija, V. V. (1998). *Distance Protection and fault location utilizing only phase current phasors.* IEEE Transactions of Power Delivery 13(4), pp. 1020-1026.

Edmund, O., Schweitzer, III. (1988).*A Review of Impedance-Based Fault Locating experience.* Proceedings of the 15th Annual Western Protective Relay Conference, Spokane, WA, 24-27.

Eisa Bashier and M. Tayeb, (2013). Faults Detection in Power Systems Using Artificial Neural Network. *American Journal of Engineering Research (AJER) 2013, Volume-02, Issue-06, pp-69-75*

El-Sharkawi, M., Niebur, D. (1996).*A tutorial course on artificial neural networks with applications to Power systems.*IEEE Publ. No. 96TP 112-0.

Eriksson, L., Saha, M. M, Rockefeller, G. D. (2015).*An accurate fault locator with compensation for apparent reactance in the fault resistance resulting from remote-end feed.* IEEE Trans on PAS 104(2), pp. 424-436.

Fonseca, J. R., Tan, A. L., Monassi, V., Janquira, W.S., Silva, R.P., Assuncano, L.A.R., and Melo, M.O.C. (1990).*Effects of Agricultural Fires on the Performanceof Overhead Transmission Lines*. IEEE Transactions on PowerDelivery, 5(2), 687-694.

Furukawa, S., Usada, O., Isozaki, T., and Irie, T. (1989). *Development and Application of Lightning Arresters for Transmission Lines.*IEEE Transactions onPower Delivery, 4(4), 2121-2127.

Gaudrat, J., Giusiano B., and Huiart.(2004). *Comparison of the performance of multi-layer perceptron and linear regression for epidemiological data*. ComputerStatist.& Data Anal., 44, 547-70.

Gail, A., Carpenter. (1997). *Distributed Learning, Recognition, and Prediction by ART and ARTMAP Neural Networks*.Neural Networks, Vol.10, No.8, pp.1473-1494.

Girish P. A., and Nitin U. G. (2015).Fault classification & location of series compensated transmission line using artificial neural network. International journal of advances in electronics and computer science, ISSN: 2393-2835 volume-2, issue-8.

Gonen, Turan, (1987). *Modern Power System Analysis.*John Wiley and Sons Inc.

Hagan, M. T., Demuth, H. B., Beale, M. H. (1996). *Neural network design.* PWS Publishing.

Haque, M. T., and Kashtiban, A. M. (2005). Application of Neural Networks in Power Systems; A Review. Proceedings of world academy of science, engineeringand technology Vol. 6 ISSN 1307-6884.

Hasabe, R. P., and Vaidya A. P. (2014). Detection and classification of faults on 220 KV transmission line using wavelet transform and neural network. International Journal of Smart Grid and Clean Energy, vol. 3, no. 3.

Haykin, S. (1994).*Neural Networks: A comprehensive foundation*. Macmillan Collage Publishing Company, Inc., New York.

Hiroyuki, M., Hikaru A., Toshiyuki, Y., and Shoichi, U. (2012). A Hybrid Intelligent System for Fault Detection in Power Systems. IEEE

IEEE. (2005). Guide for determining fault location on AC transmission and distribution lines. IEEE Power Engineering Society Publ., New York, IEEE Std C37.114.

Jain, A., Kale, V. S., Thoke, A.S. (2006). Application of artificial neural network to transmission line faulty phase selection and fault distance location. Proc. of IASTED International Energy and Power System Conf., 2006, pp. 262-267.

Jain, A., Thoke, A.S., Patel, R. N. (2008). Fault classification of double circuit transmission line using artificial neural network. International Journal of Electrical Systems Science and Engineering, vol.1, No. 4, pp.230-235.

Kale, V. S., Bhide, S. R., Bedekar, P. P., and Mohan, G.V.K. (2008). Detection and Classification of Faults on Parallel Transmission Lines using Wavelet Transform and Neural Network. World Academy of Science, Engineering and Technology 22.

Karl Zimmerman, David Costello. (2009). *Impedance-based fault location experience.* Schweitzer Engineering Laboratories, Inc. Pullman, WA USA.

karthikeyan kasinathan. (2007). Power system fault detection and classification by wavelet transforms and adaptive resonance theory neural networks. *University of Kentucky Master's Theses. Paper 452. http://uknowledge.uky.edu/gradschool_theses/452*

Kasztenny B., Rosolowski E., Saha M. and Hillstrom B. (2015). A self-organizing fuzzy logic based protective relay – an application to power transformer protection. IEEE Transactions on Power Delivery, Vol. 12, No. 3, pp. 1119-27.

Kezunovic, M. (1997).*A survey of neural net applications to protective relaying and fault analysis.*International Journal of Engineering Intelligent Systems for Electronics, Engineering and Communications 5(4), pp. 185-192.

Kezunoic, M. (1997).A survey of neural net application to fault analysis," Eng. Int. Sys., vol.5, no. 4, pp. 185-192.

Kezunovic, M., Rikalo, I. (1996).*Detect and classify faults using neural nets*. IEEE Computer Applications in Power, pp 42-47.

Kezunovic, M., Rikalo, M. I., and Sobajic, D. (1995).*High-speed fault detection and classification with neural nets*.Electric Power Systems Research, vol. 34, pp.109-116.

Kezunovic, M., Rikalo, M. I., and Sobajic, D. (1996). *Real-time and off-line transmission line fault classification using neural networks*. Engineering Intelligent Systems, vol. 10 No. pp 57- 63.

Khorashadi-Zadeh, H. (2004). Artificial neural network approach to fault classification for double circuit transmission lines.In Proc. of IEEE Transmission and Distribution Conf., pp. 859-862.

Kim, C. H., and Aggarwal, R.(2000). Wavelet transforms in power systems Part 1: General introduction to the wavelet transforms. Power Engineering Journal, vol. 14, no. 2, pp. 81–88.

Kim, C. H., and Aggarwal, R. (2001). Wavelet transforms in power systems Part 2: Examples of application to actual power system transients. Power Engineering Journal, vol. 15, no. 4, pp. 193–202.

Kohzadi, N., Boyd, S. M., Kermanshashi, B. and Kaastra.I. (1996).*A comparison of artifial neural network and time series models for forecasting commodity prices.*Neuro-computting, 10, 169-181.

Kumar, M., Raghuwanshi, N. S., Singh, R., Wallender W. W., and Pruitt, W. O.(2002). Estimating Evapotranspiration using Artificial Neural Network. Journal of irrigation and Drainage Enginneering, 128, 224-233.

Lahiri U, Pradhan A.K, Mukhopadhyaya S. (2005). Modular neural-network based directional relay for transmission line protection. IEEE Trans. on Power Delivery, vol. 20, no. 4, pp. 2154-2155.

Lipmann, R. P. (1989). *Pattern Classification using Neural Networks*. IEEE Communication Mag. Pp 27-62.

Lukowicz M., Rosolowski E. (2013). Artificial neural network based dynamic compensation of current transformer errors. Proceedings of the 8th International Symposium on Short-Circuit Currents in Power Systems, Brussels, pp. 19-24.

Magnago, F. H., Abur, A. (2014). *Advanced techniques for transmission and distribution system fault location.* Proceedings of CIGRE – Study committee 34 Colloquium and Meeting, Florence, paper 215.

Mamta Patel, and Patel, R. N. (2012). Fault Detection and Classification on a Transmission Line using Wavelet Multi Resolution Analysis and Neural Network. *International Journal of Computer Applications (0975 – 8887) Volume 47– No.22.*

Master T. (1993). *Practical neural network recipes in C++*.Academic press, New York.

Mohammad, A. A., and Rahul, S. D. (2014). Using Wavelet for Finding Fault Place and Neural Network for Types ofFault in Transmission Lines. International Journal of Engineering Research and General Science Volume 2, Issue 4, ISSN 2091-2730, www.ijergs.org.

Mollanezhad, M. H., and Akbari, F. A. (2013). Accurate Fault Classification of Transmission Line UsingWavelet Transform and Probabilistic Neural Network. Iranian Journal of Electrical & Electronic Engineering, Vol. 9, No. 3.

Nan Zhangand Mladen Kezunovic. (2014). Transmission Line Boundary Protection Using Wavelet Transform and Neural Network. IEEE.

Omid, M., Omidvar, D., Elliott L. (1997). *Neural systems for Control.*Elsevier Science & Technology Books.

Osman, A. H., and Malik, O. P. (2004). Transmission line distance protection based on wavelet transform. IEEE Trans. Power Delivery, vol. 19, no. 2, pp.515–523

Pao, Y. H. (2012). *Adaptive Pattern Recognition and Neural Networks*. Reading: Wesley. p.309.

Pao, Y. H., Sobajic, D. J. (1988). *Autonomous Feature Discovery of Clearing time assessment.* Symposium of Expert System Applications to Power Systems, Stockholm – Helsinki, pp. 5.22-5.27.

Philippe, C., and Daniel, F. (2003).*Back-Propagation Neural Network Tutorial.*Internet tutorial"http://ieee.uow.edu.au/~daniel/software/libneural/BPN_tutorial/BPN_English/BPN_English/BPN_English.html", April 2003.

Razi, K., Hagh, N.T., Abrabian, G. (2007). High accurate fault classification of power transmission line using fuzzy logic.Proc. of IEEE International Power Engg.Conf., pp. 42-46.

Reddy, M. J, Mohanta, D. K. (2008). *Adaptive-neuro-fuzzy inference system approach for transmission line fault classification and location incorporating effects of power swings.*Proceedings of IET Generation, Transmission and Distribution, pp.

Robi Polikar, (2014). *The Engineer's ultimate guide to Wavelet analysis: The WaveletTutorial.*http://users.rowan.edu/polikar/WAVELETS/WTtutorial.html

Saha. M. M., Das R., Verho, P., Novosel, D. (2006). *Review of fault location techniques for distribution systems.*Proceedings of Power Systems and Communications Infrastructure for the Future Conference, Beijing, 6p.

Saha, M. M., Izykowski, J., Rosolowski, E. (2010). *Fault Location on Power Networks*. Springer publications.

Saha M. M. and Kasztenny B. (2014). The special issue on AI applications to power system protection. *International Journal of Engineering Intelligent System.* Vol. 5, No 4, pp.185-93.

Santoso, E. J., Powers, W. M., Grady, and Hofmann, P. (2013). Power quality assessment via wavelet transform analysis. IEEE Trans. Power Delivery, vol.11, no. 2, pp. 924–930.

Sidhu, T.S., Singh, H. and Sachdev, M.S. (1995). *Design, implementation and testing of an artificial neural network based fault direction discriminator for protecting transmission lines.* IEEE Trans. On Power Delivery, vol. 10, no., pp 697-706.

Silva, M., Oleskovicz, M., Coury, D. V. (2004).*A fault locator for transmission lines using travelling waves and wavelet transform theory.*Proceedings of 8th International conference on Developments in Power System Protection – DPSP, IEE CP500, pp. 212-215.

Solanki, M., Song.Y.H. (2003). Transient protection of EHV transmission line using discrete wavelet analysis, Power Engineering Society General Meating, IEEE.

Srinivasa, R. P. and Baddu, N. B. (2013).Pattern Recognition Approach for Fault Identification in Power Transmission Lines. *Int. Journal of Engineering Research and Applications,* www.ijera.com, *ISSN: 2248-9622, Vol. 3, Issue 5, pp.1051-1056*

Stacchini J. C., Rodrigues M. A. P., Schilling M. T. and Filho M. B. (2001). Fault location in electrical power systems using intelligent systems techniques. *IEEE Trans. On Power Delivery, vol. 16, No. 1, January 2001, pp 59-67.*

Tang, Y., Wang HF, Aggarwal RK et al. (2000). *Fault indicators in transmission and distribution systems.*Proceedings of International conference on Electric Utility Deregulation and Restructuring and Power Technologies – DRPT, pp. 238-243.

Tarafdar, M., Haque and Kashtiban, A. M. (2005).*Applications of Neural Networks in Power Systems; A Review.* Transactions on Engineering, Computing and Technology V6 ISSN1305-5313.

Vasilic, S., and Kezunovic, M. (2002).*An Improved neural network algorithm for classifying the transmission line faults.*in Proc. IEEE PES Power Winter Meeting, New York, NY, Jan 2002, vo.2, pp. 918-923.

Vasilic, S., and Kezunovic, M. (2004).Fuzzy *ART Neural Network Algorithm for Classifying the Power System Fault*s. IEEE Transactions on Power Delivery, pp 1-9.

Vasilic, S., and Kezunovic, M. (2005). Fuzzy ART neural network algorithm for classifying the power system faults. IEEE Trans. Power Delivery, vol. 20, no. 2, pp. 1306–1314.

Venkatesan R. and Balamurugan B. (2001). A real time hardware fault detector using an artificial neural network for distance protection," *IEEE Trans. On Power Delivery, vol. 16, No. 1, January 2001, pp 75-82.*

Warwick K., Ekwue A. and Aggarwal R. (ed). (2015). Artificial intelligence techniques in power systems. The Institution of Electrical Engineers, London.

Wiszniewski A. and Kasztenny B. (2013). Primary protective relays with elements of expert systems. Proceedings of 2013 CIGRE Session, Paris, France, Paper 34,2, CN.

Wright, A., and Christopoulos, C. (1993).*Electrical Power System Protection*.Chapman &Hall publications, London.

Xia, M.C., Zhuang, Y., Huang. (2010). Wavelet analysis in transient based protection for power system high voltage transmission line.Proceedings of the 2010 International Conference on Wavelet Analysis and Pattern Recognition, Quingdao.

Xinzhou, D., Wei, K., & Tao, C. (2009). Fault classification and faulted-phase selection based on the initial current travelling wave. IEEE Trans. Power Delivery, vol. 24, No.2.

Zadeh H. (2006). An extended ANN-based high speed accurate distance protection algorithm. Electric Power and Energy Systems, vol. 28, no. 6, pp. 387 -395.

Zhihong Chen and Jean-Claud Maun, (Feb. 2000). Artificial neural network approach to single-ended fault locator for transmission lines. IEEE transactions on power systems, vol. 15. No. I.

Ziegler, G. (2006). *Numerical Distance Protection, Principles and Applications*. Siemens AG, Publicis MCD Verlag, Erlangen.

APPENDIX

A. MATLAB DECOMPOSITION OF THE EXTRACTED FAULT WAVEFORMS

```
>> % From Workspace (Iabc and Vabc are sent workspace from
        % simulink environment after simulation)
>> Va = Vabc(:,1); % 1x3047 matrix vector
>> Vb = Vabc(:,2);
>> Vc = Vabc(:,3);
>> Ia = Iabc(:,1);
>> Ib = Iabc(:,2);
>> Ic = Iabc(:,3);
>> % After Wavelet Decomposition Va, Vb, Vc, Ia, Ib, Ic becomes
>> % dVa, dVb, dVc, dIa, dIb, dIc respectively
        % dVa_sum, etc are 1x3047 matrix vectors
```

B. MATLAB SUMMATION OF THE DETAIL AND APPROXIMATION COEFFICIENTS OF THE DECOMPOSED WAVEFORMS

```
>> dVa_sum = (dVa(1,:)+dVa(2,:)+dVa(3,:)+dVa(4,:)+dVa(5,:));
>> dVb_sum = (dVb(1,:)+dVb(2,:)+dVb(3,:)+dVb(4,:)+dVb(5,:));
>> dVc_sum = (dVc(1,:)+dVc(2,:)+dVc(3,:)+dVc(4,:)+dVc(5,:));
>> dIa_sum = (dIa(1,:)+dIa(2,:)+dIa(3,:)+dIa(4,:)+dIa(5,:));
```

```
>> dIb_sum = (dIb(1,:)+dIb(2,:)+dIb(3,:)+dIb(4,:)+dIb(5,:));

>> dIc_sum = (dIc(1,:)+dIc(2,:)+dIc(3,:)+dIc(4,:)+dIc(5,:));

>> Wa = dVa_sum + dIa_sum; % 1x3047 matrix vector

>> Wb = dVb_sum + dIb_sum;

>> Wc = dVc_sum + dIc_sum;

>> Wa = Wa';

>> Wb = Wb';

>> Wc = Wc';

>> W = [Wa, Wb, Wc];% Input to the neural network, Wcan be a

                % 3047x3, 3007x3, 3140x3 or 2256x3 matrix vector
```

C. RESULTS FOR FAULT IDENTIFICATION

i) Mean Square ErrorPerformance Plot

```
function createfigure(X1, YMatrix1, X2, Y1, X3, Y2, X4, Y3)
%CREATEFIGURE(X1, YMATRIX1, X2, Y1, X3, Y2, X4, Y3)
%  X1:  vector of x data
%  YMATRIX1:  matrix of y data
%  X2:  vector of x data
%  Y1:  vector of y data
%  X3:  vector of x data
%  Y2:  vector of y data
%  X4:  vector of x data
%  Y3:  vector of y data

%  Auto-generated by MATLAB on 11-Feb-2017 11:17:59

% Create figure
figure1 = figure('Tag','TRAINING_PLOTPERFORM','NumberTitle','off',...
   'Name','Neural Network Training Performance (plotperform), Epoch 87, Validation stop.');

% Create axes
axes1 = axes('Parent',figure1,'YScale','log','YMinorTick','on');
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes1,[0 87]);
%% Uncomment the following line to preserve the Y-limits of the axes
```

```
% ylim(axes1,[0.09 1.1]);
hold(axes1,'all');

% Create multiple lines using matrix input to semilogy
semilogy1 = semilogy(X1,YMatrix1,'Parent',axes1,'LineWidth',2);
set(semilogy1(1),'Color',[0 0 1],'DisplayName','Train');
set(semilogy1(2),'Color',[0 0.8 0],'DisplayName','Validation');
set(semilogy1(3),'Color',[1 0 0],'DisplayName','Test');

% Create semilogy
semilogy(X2,Y1,'Parent',axes1,'LineStyle',':','Color',[0 0.48 0],...
    'DisplayName','Best');

% Create semilogy
semilogy(X3,Y2,'Parent',axes1,'MarkerSize',16,'Marker','o','LineWidth',1.5,...
    'LineStyle','none',...
    'Color',[0 0.48 0]);

% Create semilogy
semilogy(X4,Y3,'Parent',axes1,'LineStyle',':','Color',[0 0 0]);

% Create title
title('Best Validation Performance is 0.26986 at epoch 81',...
    'FontWeight','bold',...
    'FontSize',12);

% Create ylabel
ylabel('Mean Squared Error  (mse)','FontWeight','bold','FontSize',12);

% Create xlabel
xlabel('87 Epochs','FontWeight','bold','FontSize',12);

% uicontrol currently does not support code generation, enter 'doc uicontrol' for correct input
syntax
% In order to generate code for uicontrol, you may use GUIDE. Enter 'doc guide' for more
information

% uicontrol(...);

% Create legend
legend(axes1,'show');
```

ii) Regression Fit

```
function createfigure(X1, YMatrix1, X2, Y1, YMatrix2, X3, YMatrix3, X4, YMatrix4, X5)
%CREATEFIGURE(X1, YMATRIX1, X2, Y1, YMATRIX2, X3, YMATRIX3, X4,
YMATRIX4, X5)
%  X1:  vector of x data
%  YMATRIX1:  matrix of y data
%  X2:  vector of x data
%  Y1:  vector of y data
%  YMATRIX2:  matrix of y data
%  X3:  vector of x data
%  YMATRIX3:  matrix of y data
%  X4:  vector of x data
%  YMATRIX4:  matrix of y data
%  X5:  vector of x data

%  Auto-generated by MATLAB on 11-Feb-2017 11:30:47

% Create figure
figure1 = figure('Tag','TRAINING_PLOTREGRESSION','NumberTitle','off',...
    'Name','Neural Network Training Regression (plotregression), Epoch 87, Validation stop.');

% uicontrol currently does not support code generation, enter 'doc uicontrol' for correct input
syntax
% In order to generate code for uicontrol, you may use GUIDE. Enter 'doc guide' for more
information

% uicontrol(...);

% Create axes
axes1 = axes('Parent',figure1,...
    'Position',[0.13 0.593876272242609 0.304276096520464 0.331123727757391],...
    'PlotBoxAspectRatio',[1 1 1]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes1,[0 1]);
%% Uncomment the following line to preserve the Y-limits of the axes
% ylim(axes1,[0 1]);
box(axes1,'on');
hold(axes1,'all');

% Create xlabel
xlabel('Target','FontWeight','bold','FontSize',12);

% Create ylabel
ylabel('Output ~= 0.32*Target + 0.33','FontWeight','bold','FontSize',12);
```

```
% Create title
title('Training: R=0.62376','FontWeight','bold','FontSize',12);

% Create multiple lines using matrix input to plot
plot1 = plot(X1,YMatrix1,'Parent',axes1);
set(plot1(1),'LineStyle',':','DisplayName','Y = T','Color',[0 0 0]);
set(plot1(2),'LineWidth',2,'DisplayName','Fit');

% Create plot
plot(X2,Y1,'Parent',axes1,'Marker','o','LineStyle','none',...
    'DisplayName','Data',...
    'Color',[0 0 0]);

% Create legend
legend1 = legend(axes1,'show');
set(legend1,'Location','NorthWest');

% Create axes
axes2 = axes('Parent',figure1,...
    'Position',[0.587518430936152 0.593876272242609 0.317481569063848
0.331123727757391],...
    'PlotBoxAspectRatio',[1 1 1]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes2,[0 1]);
%% Uncomment the following line to preserve the Y-limits of the axes
% ylim(axes2,[0 1]);
box(axes2,'on');
hold(axes2,'all');

% Create xlabel
xlabel('Target','FontWeight','bold','FontSize',12);

% Create ylabel
ylabel('Output ~= 0.075*Target + 0.6','FontWeight','bold','FontSize',12);

% Create title
title('Validation: R=0.19465','FontWeight','bold','FontSize',12);

% Create multiple lines using matrix input to plot
plot2 = plot(X1,YMatrix2,'Parent',axes2);
set(plot2(1),'LineStyle',':','DisplayName','Y = T','Color',[0 0 0]);
set(plot2(2),'LineWidth',2,'Color',[0 1 0],'DisplayName','Fit');

% Create plot
```

```
plot(X3,Y1,'Parent',axes2,'Marker','o','LineStyle','none',...
    'DisplayName','Data',...
    'Color',[0 0 0]);

% Create legend
legend2 = legend(axes2,'show');
set(legend2,'Location','NorthWest');

% Create axes
axes3 = axes('Parent',figure1,...
    'Position',[0.13 0.11 0.304276096520464 0.329981370316705],...
    'PlotBoxAspectRatio',[1 1 1]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes3,[0 1]);
%% Uncomment the following line to preserve the Y-limits of the axes
% ylim(axes3,[0 1]);
box(axes3,'on');
hold(axes3,'all');
% Create xlabel
xlabel('Target','FontWeight','bold','FontSize',12);

% Create ylabel
ylabel('Output ~= 0.16*Target + 0.44','FontWeight','bold','FontSize',12);

% Create title
title('Test: R=0.29941','FontWeight','bold','FontSize',12);

% Create multiple lines using matrix input to plot
plot3 = plot(X1,YMatrix3,'Parent',axes3);
set(plot3(1),'LineStyle',':','DisplayName','Y = T','Color',[0 0 0]);
set(plot3(2),'LineWidth',2,'Color',[1 0 0],'DisplayName','Fit');

% Create plot
plot(X4,Y1,'Parent',axes3,'Marker','o','LineStyle','none',...
    'DisplayName','Data',...
    'Color',[0 0 0]);

% Create legend
legend3 = legend(axes3,'show');
set(legend3,'Location','NorthWest');

% Create axes
axes4 = axes('Parent',figure1,...
    'Position',[0.587518430936152 0.11 0.317481569063848 0.329981370316705],...
```

```
   'PlotBoxAspectRatio',[1 1 1]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes4,[0 1]);
%% Uncomment the following line to preserve the Y-limits of the axes
% ylim(axes4,[0 1]);
box(axes4,'on');
hold(axes4,'all');

% Create xlabel
xlabel('Target','FontWeight','bold','FontSize',12);

% Create ylabel
ylabel('Output ~= 0.25*Target + 0.4','FontWeight','bold','FontSize',12);

% Create title
title('All: R=0.48534','FontWeight','bold','FontSize',12);

% Create multiple lines using matrix input to plot
plot4 = plot(X1,YMatrix4,'Parent',axes4);
set(plot4(1),'LineStyle',':','DisplayName','Y = T','Color',[0 0 0]);
set(plot4(2),'LineWidth',2,'Color',[0.4 0.4 0.4],'DisplayName','Fit');

% Create plot
plot(X5,Y1,'Parent',axes4,'Marker','o','LineStyle','none',...
   'DisplayName','Data',...
   'Color',[0 0 0]);

% Create legend
legend4 = legend(axes4,'show');
set(legend4,'Location','NorthWest');
```

iii) Confusion Plot

```
function createfigure(X1, Y1, Y2, Y3)
%CREATEFIGURE(X1, Y1, Y2, Y3)
%  X1:  vector of x data
%  Y1:  vector of y data
%  Y2:  vector of y data
%  Y3:  vector of y data

%  Auto-generated by MATLAB on 11-Feb-2017 11:38:30

% Create figure
```

```
figure1 = figure('Tag','TRAINING_PLOTTRAINSTATE','NumberTitle','off',...
   'Name','Neural Network Training Training State (plottrainstate), Epoch 87, Validation
stop.');

% uicontrol currently does not support code generation, enter 'doc uicontrol' for correct input
syntax
% In order to generate code for uicontrol, you may use GUIDE. Enter 'doc guide' for more
information
% uicontrol(...);

% Create axes
axes1 = axes('Parent',figure1,'YScale','log','YMinorTick','on',...
   'XTickLabel','',...
   'Position',[0.13 0.709264705882353 0.775 0.19093662464986]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes1,[0 87]);
box(axes1,'on');
hold(axes1,'all');

% Create semilogy
semilogy(X1,Y1,'Parent',axes1,'MarkerFaceColor',[1 0 0],'LineWidth',2);

% Create ylabel
ylabel('gradient');

% Create title
title('Gradient = 0.12202, at epoch 87');

% Create axes
axes2 = axes('Parent',figure1,'XTickLabel','',...
   'Position',[0.13 0.409632352941176 0.775 0.19093662464986]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes2,[0 87]);
box(axes2,'on');
hold(axes2,'all');

% Create plot
plot(X1,Y2,'Parent',axes2,'MarkerFaceColor',[1 0 0],'Marker','diamond',...
   'LineWidth',1,...
   'LineStyle','none');

% Create ylabel
ylabel('val fail');
```

```
% Create title
title('Validation Checks = 6, at epoch 87');

% Create axes
axes3 = axes('Parent',figure1,'Position',[0.13 0.11 0.775 0.19093662464986]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes3,[0 87]);
box(axes3,'on');
hold(axes3,'all');

% Create plot
plot(X1,Y3,'Parent',axes3,'MarkerFaceColor',[1 0 0],'LineWidth',2);

% Create ylabel
ylabel('lr');

% Create xlabel
xlabel('87 Epochs');

% Create title
title('Learning Rate = 0.69738, at epoch 87');
```

D. RESULTS FOR FAULT CLASSIFICATION

i) Mean Square ErrorPerformance Plot

```
function createfigure(X1, YMatrix1, X2, Y1, X3, Y2, X4, Y3)
%CREATEFIGURE(X1, YMATRIX1, X2, Y1, X3, Y2, X4, Y3)
%  X1:  vector of x data
%  YMATRIX1:  matrix of y data
%  X2:  vector of x data
%  Y1:  vector of y data
%  X3:  vector of x data
%  Y2:  vector of y data
%  X4:  vector of x data
%  Y3:  vector of y data

%  Auto-generated by MATLAB on 11-Feb-2017 11:22:42

% Create figure
figure1 = figure('Tag','TRAINING_PLOTPERFORM','NumberTitle','off',...
    'Name','Neural Network Training Performance (plotperform), Epoch 115, Validation
stop.');
```

```
% Create axes
axes1 = axes('Parent',figure1,'YScale','log','YMinorTick','on');
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes1,[0 115]);
%% Uncomment the following line to preserve the Y-limits of the axes
% ylim(axes1,[9e-05 1.1]);
hold(axes1,'all');

% Create multiple lines using matrix input to semilogy
semilogy1 = semilogy(X1,YMatrix1,'Parent',axes1,'LineWidth',2);
set(semilogy1(1),'Color',[0 0 1],'DisplayName','Train');
set(semilogy1(2),'Color',[0 0.8 0],'DisplayName','Validation');
set(semilogy1(3),'Color',[1 0 0],'DisplayName','Test');

% Create semilogy
semilogy(X2,Y1,'Parent',axes1,'LineStyle',':','Color',[0 0.48 0],...
    'DisplayName','Best');

% Create semilogy
semilogy(X3,Y2,'Parent',axes1,'MarkerSize',16,'Marker','o','LineWidth',1.5,...
    'LineStyle','none',...
    'Color',[0 0.48 0]);

% Create semilogy
semilogy(X4,Y3,'Parent',axes1,'LineStyle',':','Color',[0 0 0]);

% Create title
title('Best Validation Performance is 0.0009082 at epoch 109',...
    'FontWeight','bold',...
    'FontSize',12);

% Create ylabel
ylabel('Mean Squared Error  (mse)','FontWeight','bold','FontSize',12);

% Create xlabel
xlabel('115 Epochs','FontWeight','bold','FontSize',12);

% uicontrol currently does not support code generation, enter 'doc uicontrol' for correct input
syntax
% In order to generate code for uicontrol, you may use GUIDE. Enter 'doc guide' for more
information

% uicontrol(...);
```

```
% Create legend
legend(axes1,'show');
```

ii) Regression Fit

```
function createfigure(X1, YMatrix1, X2, Y1, YMatrix2, X3, YMatrix3, X4, YMatrix4, X5)
%CREATEFIGURE(X1, YMATRIX1, X2, Y1, YMATRIX2, X3, YMATRIX3, X4,
YMATRIX4, X5)
%  X1:  vector of x data
%  YMATRIX1:  matrix of y data
%  X2:  vector of x data
%  Y1:  vector of y data
%  YMATRIX2:  matrix of y data
%  X3:  vector of x data
%  YMATRIX3:  matrix of y data
%  X4:  vector of x data
%  YMATRIX4:  matrix of y data
%  X5:  vector of x data

%  Auto-generated by MATLAB on 11-Feb-2017 11:35:21

% Create figure
figure1 = figure('Tag','TRAINING_PLOTREGRESSION','NumberTitle','off',...
   'Name','Neural Network Training Regression (plotregression), Epoch 261, Minimum
gradient reached.');

% uicontrol currently does not support code generation, enter 'doc uicontrol' for correct input
syntax
% In order to generate code for uicontrol, you may use GUIDE. Enter 'doc guide' for more
information

% uicontrol(...);

% Create axes
axes1 = axes('Parent',figure1,...
   'Position',[0.13 0.593876272242609 0.304276096520464 0.331123727757391],...
   'PlotBoxAspectRatio',[1 1 1]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes1,[0 1]);
%% Uncomment the following line to preserve the Y-limits of the axes
% ylim(axes1,[0 1]);
box(axes1,'on');
```

```
hold(axes1,'all');

% Create xlabel
xlabel('Target','FontWeight','bold','FontSize',12);

% Create ylabel
ylabel('Output ~= 1*Target + 0.0023','FontWeight','bold','FontSize',12);

% Create title
title('Training: R=1','FontWeight','bold','FontSize',12);

% Create multiple lines using matrix input to plot
plot1 = plot(X1,YMatrix1,'Parent',axes1);
set(plot1(1),'LineStyle',':','DisplayName','Y = T','Color',[0 0 0]);
set(plot1(2),'LineWidth',2,'DisplayName','Fit');

% Create plot
plot(X2,Y1,'Parent',axes1,'Marker','o','LineStyle','none',...
    'DisplayName','Data',...
    'Color',[0 0 0]);

% Create legend
legend1 = legend(axes1,'show');
set(legend1,'Location','NorthWest');

% Create axes
axes2 = axes('Parent',figure1,...
    'Position',[0.587518430936152 0.593876272242609 0.317481569063848
0.331123727757391],...
    'PlotBoxAspectRatio',[1 1 1]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes2,[0 1]);
%% Uncomment the following line to preserve the Y-limits of the axes
% ylim(axes2,[0 1]);
box(axes2,'on');
hold(axes2,'all');

% Create xlabel
xlabel('Target','FontWeight','bold','FontSize',12);

% Create ylabel
ylabel('Output ~= 1*Target + 7e-06','FontWeight','bold','FontSize',12);

% Create title
```

```
title('Validation: R=1','FontWeight','bold','FontSize',12);

% Create multiple lines using matrix input to plot
plot2 = plot(X1,YMatrix2,'Parent',axes2);
set(plot2(1),'LineStyle',':','DisplayName','Y = T','Color',[0 0 0]);
set(plot2(2),'LineWidth',2,'Color',[0 1 0],'DisplayName','Fit');

% Create plot
plot(X3,Y1,'Parent',axes2,'Marker','o','LineStyle','none',...
    'DisplayName','Data',...
    'Color',[0 0 0]);

% Create legend
legend2 = legend(axes2,'show');
set(legend2,'Location','NorthWest');

% Create axes
axes3 = axes('Parent',figure1,...
    'Position',[0.13 0.11 0.304276096520464 0.329981370316705],...
    'PlotBoxAspectRatio',[1 1 1]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes3,[0 1]);
%% Uncomment the following line to preserve the Y-limits of the axes
% ylim(axes3,[0 1]);
box(axes3,'on');
hold(axes3,'all');

% Create xlabel
xlabel('Target','FontWeight','bold','FontSize',12);

% Create ylabel
ylabel('Output ~= 0.98*Target + 0.0077','FontWeight','bold','FontSize',12);

% Create title
title('Test: R=0.99998','FontWeight','bold','FontSize',12);

% Create multiple lines using matrix input to plot
plot3 = plot(X1,YMatrix3,'Parent',axes3);
set(plot3(1),'LineStyle',':','DisplayName','Y = T','Color',[0 0 0]);
set(plot3(2),'LineWidth',2,'Color',[1 0 0],'DisplayName','Fit');

% Create plot
plot(X4,Y1,'Parent',axes3,'Marker','o','LineStyle','none',...
    'DisplayName','Data',...
```

```
   'Color',[0 0 0]);

% Create legend
legend3 = legend(axes3,'show');
set(legend3,'Location','NorthWest');

% Create axes
axes4 = axes('Parent',figure1,...
   'Position',[0.587518430936152 0.11 0.317481569063848 0.329981370316705],...
   'PlotBoxAspectRatio',[1 1 1]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes4,[0 1]);
%% Uncomment the following line to preserve the Y-limits of the axes
% ylim(axes4,[0 1]);
box(axes4,'on');
hold(axes4,'all');

% Create xlabel
xlabel('Target','FontWeight','bold','FontSize',12);

% Create ylabel
ylabel('Output ~= 0.99*Target + 0.0029','FontWeight','bold','FontSize',12);

% Create title
title('All: R=0.99998','FontWeight','bold','FontSize',12);

% Create multiple lines using matrix input to plot
plot4 = plot(X1,YMatrix4,'Parent',axes4);
set(plot4(1),'LineStyle',':','DisplayName','Y = T','Color',[0 0 0]);
set(plot4(2),'LineWidth',2,'Color',[0.4 0.4 0.4],'DisplayName','Fit');

% Create plot
plot(X5,Y1,'Parent',axes4,'Marker','o','LineStyle','none',...
   'DisplayName','Data',...
   'Color',[0 0 0]);

% Create legend
legend4 = legend(axes4,'show');
set(legend4,'Location','NorthWest');
```

iii) Confusion Plot

```
function createfigure(X1, Y1, Y2, Y3)
```

```
%CREATEFIGURE(X1, Y1, Y2, Y3)
%  X1:  vector of x data
%  Y1:  vector of y data
%  Y2:  vector of y data
%  Y3:  vector of y data

%  Auto-generated by MATLAB on 11-Feb-2017 11:39:47

% Create figure
figure1 = figure('Tag','TRAINING_PLOTTRAINSTATE','NumberTitle','off',...
   'Name','Neural Network Training Training State (plottrainstate), Epoch 115, Validation
stop.');

% uicontrol currently does not support code generation, enter 'doc uicontrol' for correct input
syntax
% In order to generate code for uicontrol, you may use GUIDE. Enter 'doc guide' for more
information

% uicontrol(...);
% Create axes
axes1 = axes('Parent',figure1,'YScale','log','YMinorTick','on',...
   'XTickLabel','',...
   'Position',[0.13 0.709264705882353 0.775 0.19093662464986]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes1,[0 115]);
box(axes1,'on');
hold(axes1,'all');

% Create semilogy
semilogy(X1,Y1,'Parent',axes1,'MarkerFaceColor',[1 0 0],'LineWidth',2);

% Create ylabel
ylabel('gradient');

% Create title
title('Gradient = 0.007177, at epoch 115');

% Create axes
axes2 = axes('Parent',figure1,'XTickLabel','',...
   'Position',[0.13 0.409632352941176 0.775 0.19093662464986]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes2,[0 115]);
box(axes2,'on');
hold(axes2,'all');
```

```
% Create plot
plot(X1,Y2,'Parent',axes2,'MarkerFaceColor',[1 0 0],'Marker','diamond',...
    'LineWidth',1,...
    'LineStyle','none');

% Create ylabel
ylabel('val fail');
% Create title
title('Validation Checks = 6, at epoch 115');

% Create axes
axes3 = axes('Parent',figure1,'Position',[0.13 0.11 0.775 0.19093662464986]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes3,[0 115]);
box(axes3,'on');
hold(axes3,'all');

% Create plot
plot(X1,Y3,'Parent',axes3,'MarkerFaceColor',[1 0 0],'LineWidth',2);

% Create ylabel
ylabel('lr');

% Create xlabel
xlabel('115 Epochs');

% Create title
title('Learning Rate = 2.7338, at epoch 115');
```

E. RESULTS FOR FAULT LOCATION

i) Test PhasePerformance Plot

```
function createfigure(X1, YMatrix1, X2, Y1, X3, Y2, X4, Y3)
%CREATEFIGURE(X1, YMATRIX1, X2, Y1, X3, Y2, X4, Y3)
%  X1:  vector of x data
%  YMATRIX1:  matrix of y data
%  X2:  vector of x data
%  Y1:  vector of y data
%  X3:  vector of x data
%  Y2:  vector of y data
%  X4:  vector of x data
```

```
%  Y3:  vector of y data

%  Auto-generated by MATLAB on 11-Feb-2017 11:26:09

% Create figure
figure1 = figure('Tag','TRAINING_PLOTPERFORM','NumberTitle','off',...
   'Name','Neural Network Training Performance (plotperform), Epoch 261, Minimum
gradient reached.');

% Create axes
axes1 = axes('Parent',figure1,'YScale','log','YMinorTick','on');
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes1,[0 261]);
%% Uncomment the following line to preserve the Y-limits of the axes
% ylim(axes1,[9e-11 1.1]);
hold(axes1,'all');

% Create multiple lines using matrix input to semilogy
semilogy1 = semilogy(X1,YMatrix1,'Parent',axes1,'LineWidth',2);
set(semilogy1(1),'Color',[0 0 1],'DisplayName','Train');
set(semilogy1(2),'Color',[0 0.8 0],'DisplayName','Validation');
set(semilogy1(3),'Color',[1 0 0],'DisplayName','Test');

% Create semilogy
semilogy(X2,Y1,'Parent',axes1,'LineStyle',':','Color',[0 0.48 0],...
   'DisplayName','Best');

% Create semilogy
semilogy(X3,Y2,'Parent',axes1,'MarkerSize',16,'Marker','o','LineWidth',1.5,...
   'LineStyle','none',...
   'Color',[0 0.48 0]);

% Create semilogy
semilogy(X4,Y3,'Parent',axes1,'LineStyle',':','Color',[0 0 0]);

% Create title
title('Best Validation Performance is 2.4041e-10 at epoch 261',...
   'FontWeight','bold',...
   'FontSize',12);

% Create ylabel
ylabel('Mean Squared Error  (mse)','FontWeight','bold','FontSize',12);

% Create xlabel
```

```
xlabel('261 Epochs','FontWeight','bold','FontSize',12);

% uicontrol currently does not support code generation, enter 'doc uicontrol' for correct input
syntax
% In order to generate code for uicontrol, you may use GUIDE. Enter 'doc guide' for more
information

% uicontrol(...);
% Create legend
legend(axes1,'show');

% Create line
annotation(figure1,'line',[0.132142857142857 0.905357142857143],...
   [0.596619047619052 0.595238095238095],'LineStyle','-.');
```

ii) Confusion Plot

```
function createfigure(X1, YMatrix1, X2, Y1, X3, Y2, X4, Y3)
%CREATEFIGURE(X1, YMATRIX1, X2, Y1, X3, Y2, X4, Y3)
%  X1:  vector of x data
%  YMATRIX1:  matrix of y data
%  X2:  vector of x data
%  Y1:  vector of y data
%  X3:  vector of x data
%  Y2:  vector of y data
%  X4:  vector of x data
%  Y3:  vector of y data

%  Auto-generated by MATLAB on 11-Feb-2017 11:27:55

% Create figure
figure1 = figure('Tag','TRAINING_PLOTPERFORM','NumberTitle','off',...
   'Name','Neural Network Training Performance (plotperform), Epoch 258, Minimum
gradient reached.');

% Create axes
axes1 = axes('Parent',figure1,'YScale','log','YMinorTick','on');
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes1,[0 258]);
%% Uncomment the following line to preserve the Y-limits of the axes
% ylim(axes1,[9e-10 0.011]);
hold(axes1,'all');
```

```
% Create multiple lines using matrix input to semilogy
semilogy1 = semilogy(X1,YMatrix1,'Parent',axes1,'LineWidth',2);
set(semilogy1(1),'Color',[0 0 1],'DisplayName','Train');
set(semilogy1(2),'Color',[0 0.8 0],'DisplayName','Validation');
set(semilogy1(3),'Color',[1 0 0],'DisplayName','Test');

% Create semilogy
semilogy(X2,Y1,'Parent',axes1,'LineStyle',':','Color',[0 0.48 0],...
    'DisplayName','Best');

% Create semilogy
semilogy(X3,Y2,'Parent',axes1,'MarkerSize',16,'Marker','o','LineWidth',1.5,...
    'LineStyle','none',...
    'Color',[0 0.48 0]);

% Create semilogy
semilogy(X4,Y3,'Parent',axes1,'LineStyle',':','Color',[0 0 0]);

% Create title
title('Best Validation Performance is 1.5877e-06 at epoch 258',...
    'FontWeight','bold',...
    'FontSize',12);

% Create ylabel
ylabel('Mean Squared Error  (mse)','FontWeight','bold','FontSize',12);

% Create xlabel
xlabel('258 Epochs','FontWeight','bold','FontSize',12);

% uicontrol currently does not support code generation, enter 'doc uicontrol' for correct input
syntax
% In order to generate code for uicontrol, you may use GUIDE. Enter 'doc guide' for more
information

% uicontrol(...);

% Create legend
legend(axes1,'show');
```

iii) Mean Square ErrorPerformance Plot

```
function createfigure(X1, Y1, Y2, Y3)
%CREATEFIGURE(X1, Y1, Y2, Y3)
```

```
%  X1:  vector of x data
%  Y1:  vector of y data
%  Y2:  vector of y data
%  Y3:  vector of y data

%  Auto-generated by MATLAB on 11-Feb-2017 11:41:01

% Create figure
figure1 = figure('Tag','TRAINING_PLOTTRAINSTATE','NumberTitle','off',...
   'Name','Neural Network Training Training State (plottrainstate), Epoch 261, Minimum
gradient reached.');

% uicontrol currently does not support code generation, enter 'doc uicontrol' for correct input
syntax
% In order to generate code for uicontrol, you may use GUIDE. Enter 'doc guide' for more
information

% uicontrol(...);

% Create axes
axes1 = axes('Parent',figure1,'YScale','log','YMinorTick','on',...
   'XTickLabel','',...
   'Position',[0.13 0.709264705882353 0.775 0.19093662464986]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes1,[0 261]);
box(axes1,'on');
hold(axes1,'all');

% Create semilogy
semilogy(X1,Y1,'Parent',axes1,'MarkerFaceColor',[1 0 0],'LineWidth',2);

% Create ylabel
ylabel('gradient');

% Create title
title('Gradient = 9.871e-06, at epoch 261');

% Create axes
axes2 = axes('Parent',figure1,'XTickLabel','',...
   'Position',[0.13 0.409632352941176 0.775 0.19093662464986]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes2,[0 261]);
box(axes2,'on');
hold(axes2,'all');
```

```
% Create plot
plot(X1,Y2,'Parent',axes2,'MarkerFaceColor',[1 0 0],'Marker','diamond',...
    'LineWidth',1,...
    'LineStyle','none');

% Create ylabel
ylabel('val fail');
% Create title
title('Validation Checks = 0, at epoch 261');

% Create axes
axes3 = axes('Parent',figure1,'Position',[0.13 0.11 0.775 0.19093662464986]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes3,[0 261]);
box(axes3,'on');
hold(axes3,'all');

% Create plot
plot(X1,Y3,'Parent',axes3,'MarkerFaceColor',[1 0 0],'LineWidth',2);

% Create ylabel
ylabel('lr');

% Create xlabel
xlabel('261 Epochs');

% Create title
title('Learning Rate = 3391.619, at epoch 261');
```

iv) Regression Fit

```
function createfigure(X1, YMatrix1, X2, Y1, YMatrix2, X3, YMatrix3, X4, YMatrix4, X5)
%CREATEFIGURE(X1, YMATRIX1, X2, Y1, YMATRIX2, X3, YMATRIX3, X4,
YMATRIX4, X5)
%  X1:  vector of x data
%  YMATRIX1:  matrix of y data
%  X2:  vector of x data
%  Y1:  vector of y data
%  YMATRIX2:  matrix of y data
%  X3:  vector of x data
%  YMATRIX3:  matrix of y data
```

```
%  X4:  vector of x data
%  YMATRIX4:  matrix of y data
%  X5:  vector of x data

%  Auto-generated by MATLAB on 11-Feb-2017 11:36:17

% Create figure
figure1 = figure('Tag','TRAINING_PLOTREGRESSION','NumberTitle','off',...
   'Name','Neural Network Training Regression (plotregression), Epoch 258, Minimum
gradient reached.');

% uicontrol currently does not support code generation, enter 'doc uicontrol' for correct input
syntax
% In order to generate code for uicontrol, you may use GUIDE. Enter 'doc guide' for more
information

% uicontrol(...);

% Create axes
axes1 = axes('Parent',figure1,...
   'Position',[0.13 0.593876272242609 0.304276096520464 0.331123727757391],...
   'PlotBoxAspectRatio',[1 1 1]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes1,[0 1]);
%% Uncomment the following line to preserve the Y-limits of the axes
% ylim(axes1,[0 1]);
box(axes1,'on');
hold(axes1,'all');

% Create xlabel
xlabel('Target','FontWeight','bold','FontSize',12);

% Create ylabel
ylabel('Output ~= 1*Target + 0.0022','FontWeight','bold','FontSize',12);

% Create title
title('Training: R=1','FontWeight','bold','FontSize',12);

% Create multiple lines using matrix input to plot
plot1 = plot(X1,YMatrix1,'Parent',axes1);
set(plot1(1),'LineStyle',':','DisplayName','Y = T','Color',[0 0 0]);
set(plot1(2),'LineWidth',2,'DisplayName','Fit');

% Create plot
```

```
plot(X2,Y1,'Parent',axes1,'Marker','o','LineStyle','none',...
    'DisplayName','Data',...
    'Color',[0 0 0]);

% Create legend
legend1 = legend(axes1,'show');
set(legend1,'Location','NorthWest');

% Create axes
axes2 = axes('Parent',figure1,...
    'Position',[0.587518430936152 0.593876272242609 0.317481569063848
0.331123727757391],...
    'PlotBoxAspectRatio',[1 1 1]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes2,[0 1]);
%% Uncomment the following line to preserve the Y-limits of the axes
% ylim(axes2,[0 1]);
box(axes2,'on');
hold(axes2,'all');

% Create xlabel
xlabel('Target','FontWeight','bold','FontSize',12);

% Create ylabel
ylabel('Output ~= 1*Target + 0.0013','FontWeight','bold','FontSize',12);

% Create title
title('Validation: R=1','FontWeight','bold','FontSize',12);

% Create multiple lines using matrix input to plot
plot2 = plot(X1,YMatrix2,'Parent',axes2);
set(plot2(1),'LineStyle',':','DisplayName','Y = T','Color',[0 0 0]);
set(plot2(2),'LineWidth',2,'Color',[0 1 0],'DisplayName','Fit');

% Create plot
plot(X3,Y1,'Parent',axes2,'Marker','o','LineStyle','none',...
    'DisplayName','Data',...
    'Color',[0 0 0]);

% Create legend
legend2 = legend(axes2,'show');
set(legend2,'Location','NorthWest');

% Create axes
```

```
axes3 = axes('Parent',figure1,...
   'Position',[0.13 0.11 0.304276096520464 0.329981370316705],...
   'PlotBoxAspectRatio',[1 1 1]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes3,[0 1]);
%% Uncomment the following line to preserve the Y-limits of the axes
% ylim(axes3,[0 1]);
box(axes3,'on');
hold(axes3,'all');

% Create xlabel
xlabel('Target','FontWeight','bold','FontSize',12);

% Create ylabel
ylabel('Output ~= 1*Target + 5.6e-05','FontWeight','bold','FontSize',12);
% Create title
title('Test: R=1','FontWeight','bold','FontSize',12);

% Create multiple lines using matrix input to plot
plot3 = plot(X1,YMatrix3,'Parent',axes3);
set(plot3(1),'LineStyle',':','DisplayName','Y = T','Color',[0 0 0]);
set(plot3(2),'LineWidth',2,'Color',[1 0 0],'DisplayName','Fit');

% Create plot
plot(X4,Y1,'Parent',axes3,'Marker','o','LineStyle','none',...
   'DisplayName','Data',...
   'Color',[0 0 0]);

% Create legend
legend3 = legend(axes3,'show');
set(legend3,'Location','NorthWest');

% Create axes
axes4 = axes('Parent',figure1,...
   'Position',[0.587518430936152 0.11 0.317481569063848 0.329981370316705],...
   'PlotBoxAspectRatio',[1 1 1]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes4,[0 1]);
%% Uncomment the following line to preserve the Y-limits of the axes
% ylim(axes4,[0 1]);
box(axes4,'on');
hold(axes4,'all');

% Create xlabel
```

```
xlabel('Target','FontWeight','bold','FontSize',12);

% Create ylabel
ylabel('Output ~= 1*Target + 0.0017','FontWeight','bold','FontSize',12);

% Create title
title('All: R=1','FontWeight','bold','FontSize',12);

% Create multiple lines using matrix input to plot
plot4 = plot(X1,YMatrix4,'Parent',axes4);
set(plot4(1),'LineStyle',':','DisplayName','Y = T','Color',[0 0 0]);
set(plot4(2),'LineWidth',2,'Color',[0.4 0.4 0.4],'DisplayName','Fit');

% Create plot
plot(X5,Y1,'Parent',axes4,'Marker','o','LineStyle','none',...
    'DisplayName','Data',...
    'Color',[0 0 0]);

% Create legend
legend4 = legend(axes4,'show');
set(legend4,'Location','NorthWest');
```

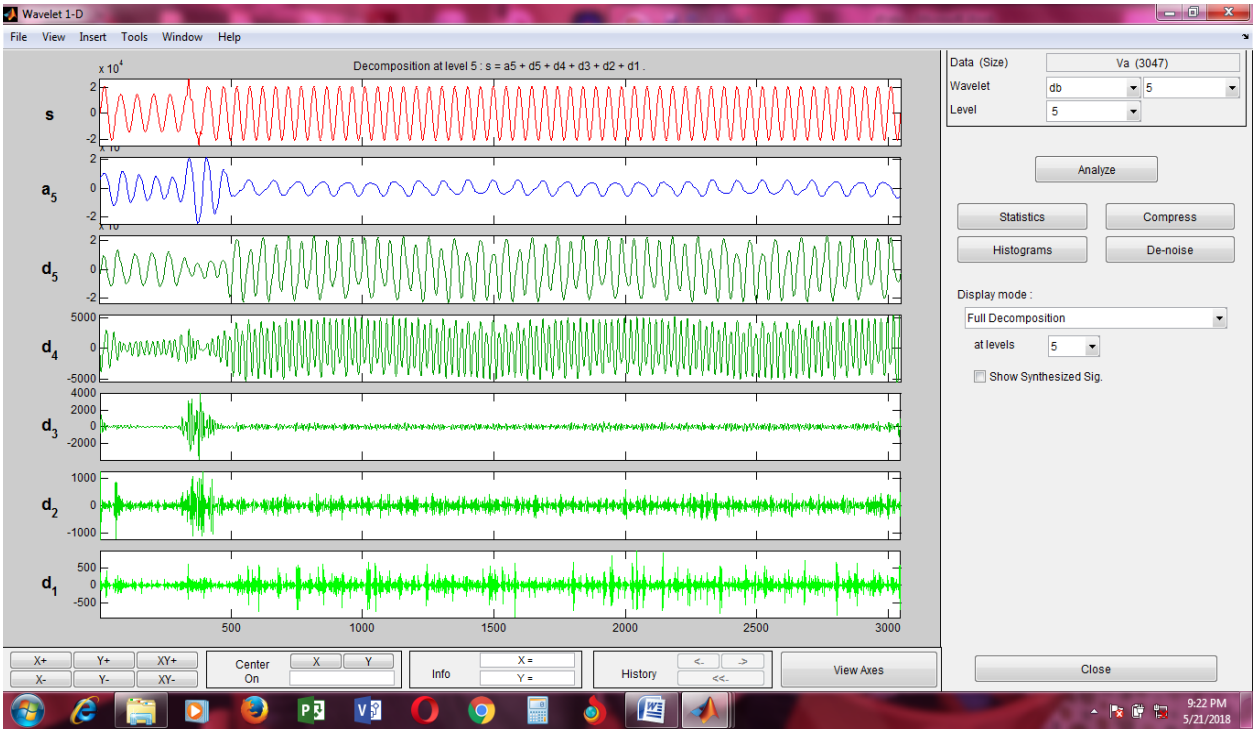
v) Gradient and validation performance plots

```
function createfigure(X1, Y1, Y2, Y3)
%CREATEFIGURE(X1, Y1, Y2, Y3)
%  X1:  vector of x data
%  Y1:  vector of y data
%  Y2:  vector of y data
%  Y3:  vector of y data

%  Auto-generated by MATLAB on 11-Feb-2017 11:42:05

% Create figure
figure1 = figure('Tag','TRAINING_PLOTTRAINSTATE','NumberTitle','off',...
    'Name','Neural Network Training Training State (plottrainstate), Epoch 258, Minimum
gradient reached.');

% uicontrol currently does not support code generation, enter 'doc uicontrol' for correct input
syntax
% In order to generate code for uicontrol, you may use GUIDE. Enter 'doc guide' for more
information
```

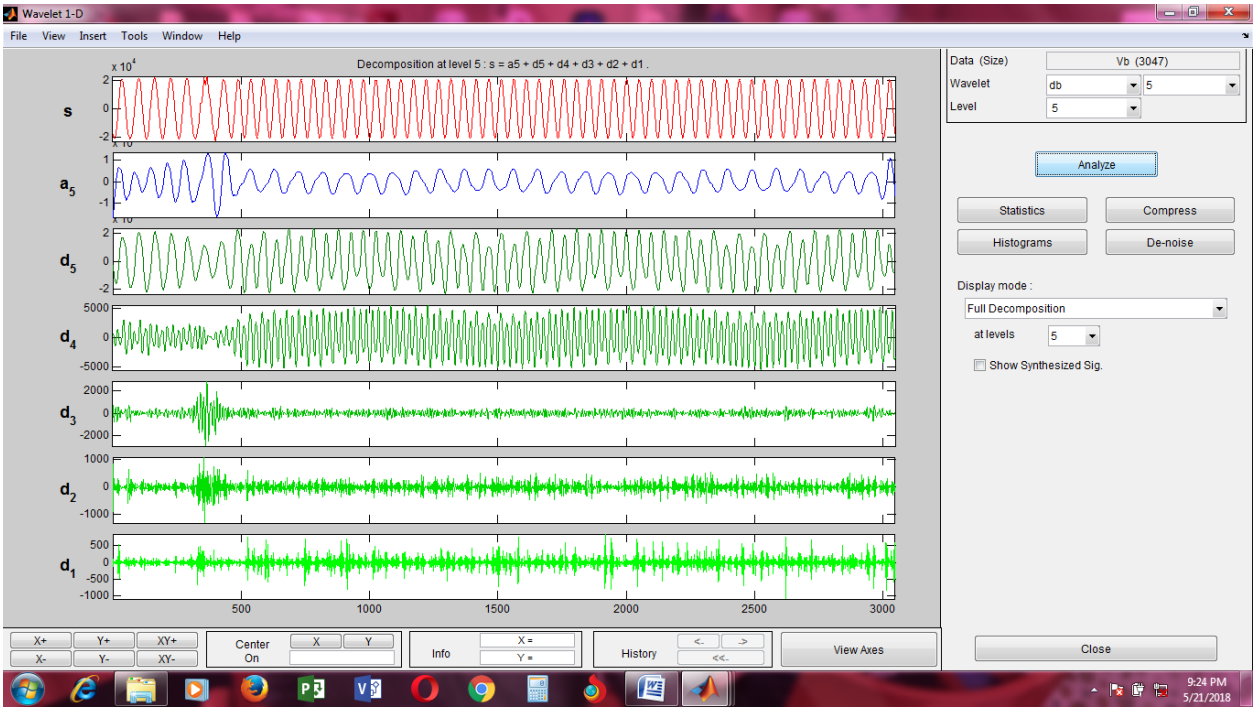
```
% uicontrol(...);

% Create axes
axes1 = axes('Parent',figure1,'YScale','log','YMinorTick','on',...
   'XTickLabel','',...
   'Position',[0.13 0.709264705882353 0.775 0.19093662464986]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes1,[0 258]);
box(axes1,'on');
hold(axes1,'all');

% Create semilogy
semilogy(X1,Y1,'Parent',axes1,'MarkerFaceColor',[1 0 0],'LineWidth',2);

% Create ylabel
ylabel('gradient');

% Create title
title('Gradient = 9.7126e-06, at epoch 258');

% Create axes
axes2 = axes('Parent',figure1,'XTickLabel','',...
   'Position',[0.13 0.409632352941176 0.775 0.19093662464986]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes2,[0 258]);
box(axes2,'on');
hold(axes2,'all');

% Create plot
plot(X1,Y2,'Parent',axes2,'MarkerFaceColor',[1 0 0],'Marker','diamond',...
   'LineWidth',1,...
   'LineStyle','none');

% Create ylabel
ylabel('val fail');

% Create title
title('Validation Checks = 0, at epoch 258');

% Create axes
axes3 = axes('Parent',figure1,'Position',[0.13 0.11 0.775 0.19093662464986]);
%% Uncomment the following line to preserve the X-limits of the axes
% xlim(axes3,[0 258]);
```

```
box(axes3,'on');
hold(axes3,'all');

% Create plot
plot(X1,Y3,'Parent',axes3,'MarkerFaceColor',[1 0 0],'LineWidth',2);

% Create ylabel
ylabel('lr');
% Create xlabel
xlabel('258 Epochs');

% Create title
title('Learning Rate = 2929.808, at epoch 258');
```

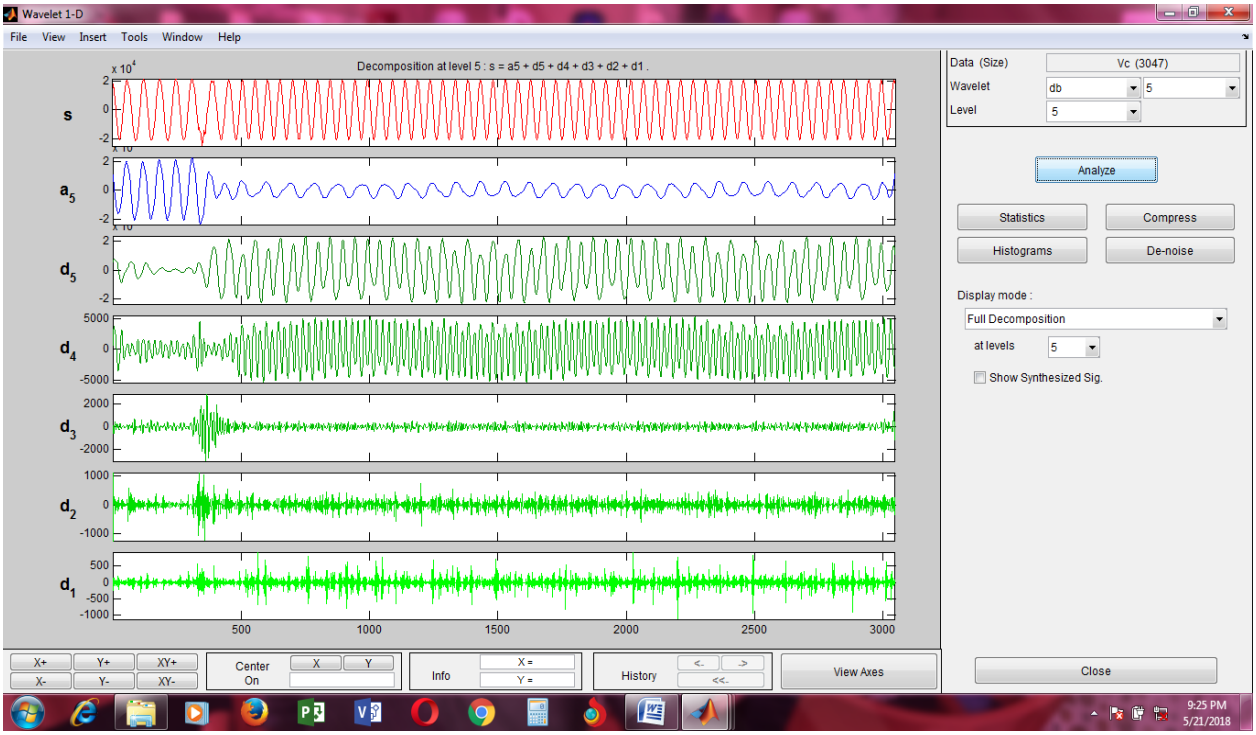
F. SIGNAL DECOMPOSITION OF CURRENT AND VOLTAGE WAVEFORMS OF ALL THE FOUR TYPES OF FAULTS



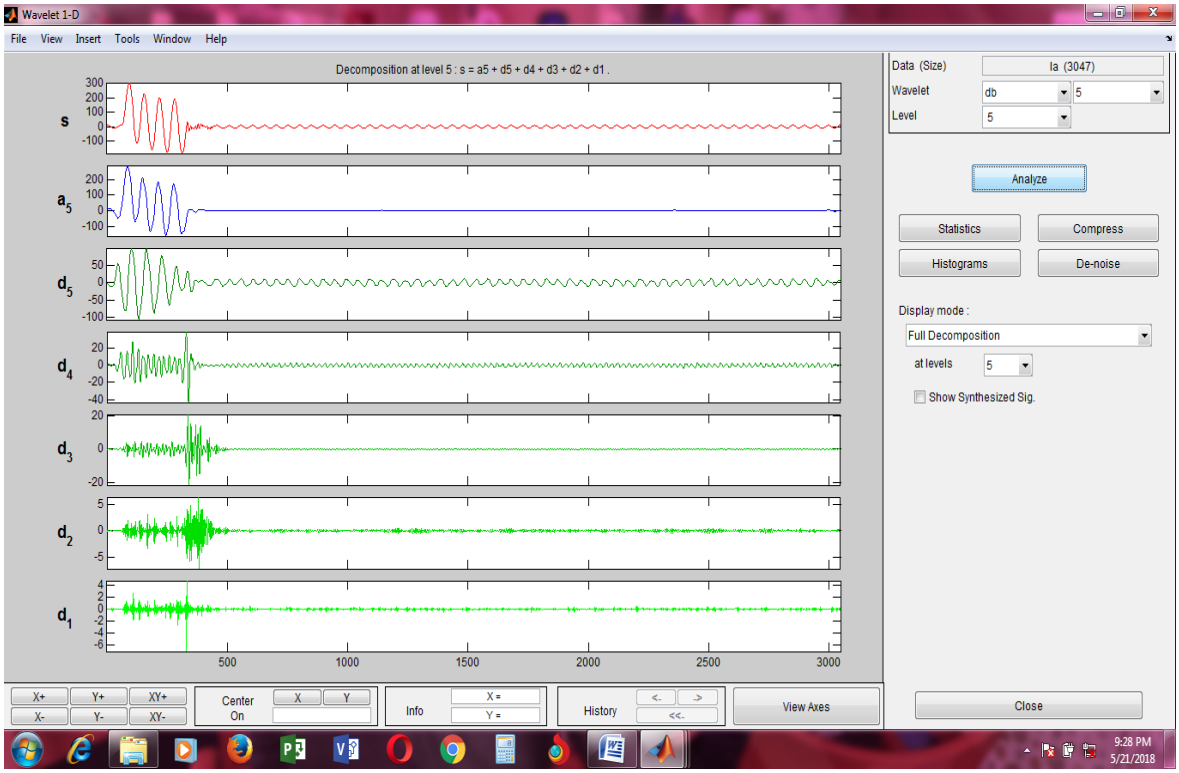
P1: Decomposed signal of voltage waveform of LG Fault of phase A at a distance of 140km from the source



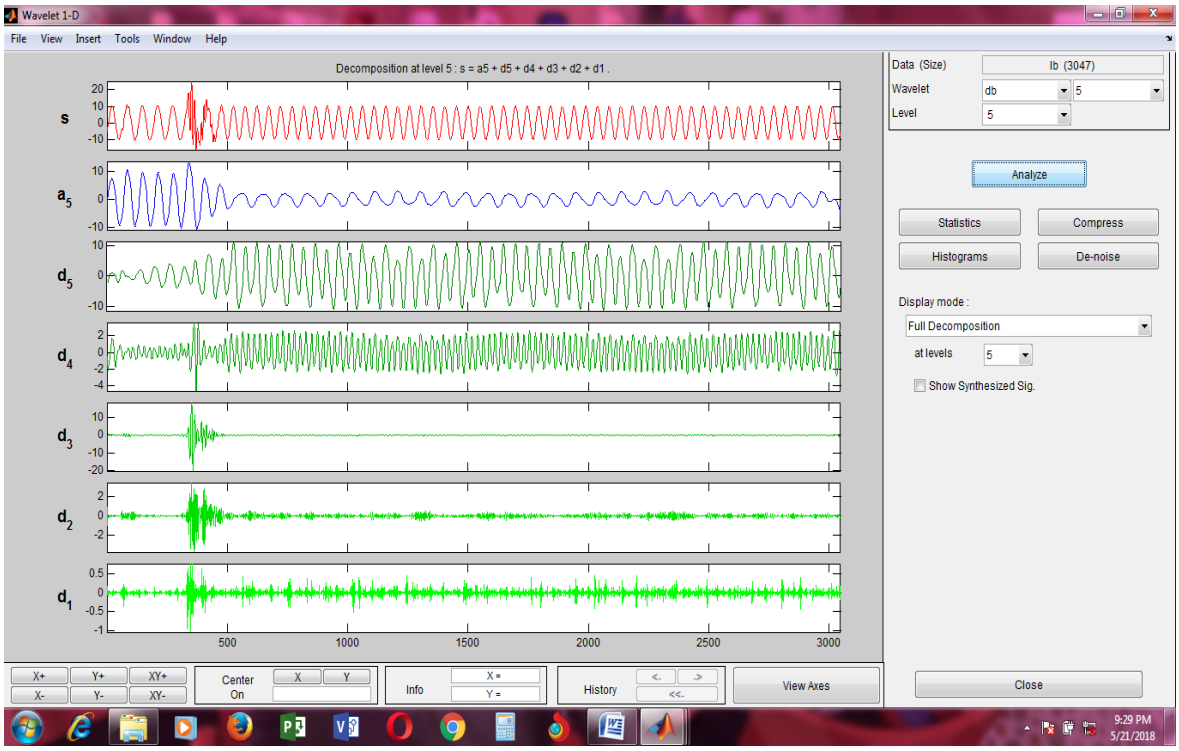
P2: Decomposed signal of voltage waveform of LG Fault of phase B at a distance of 140km from the source



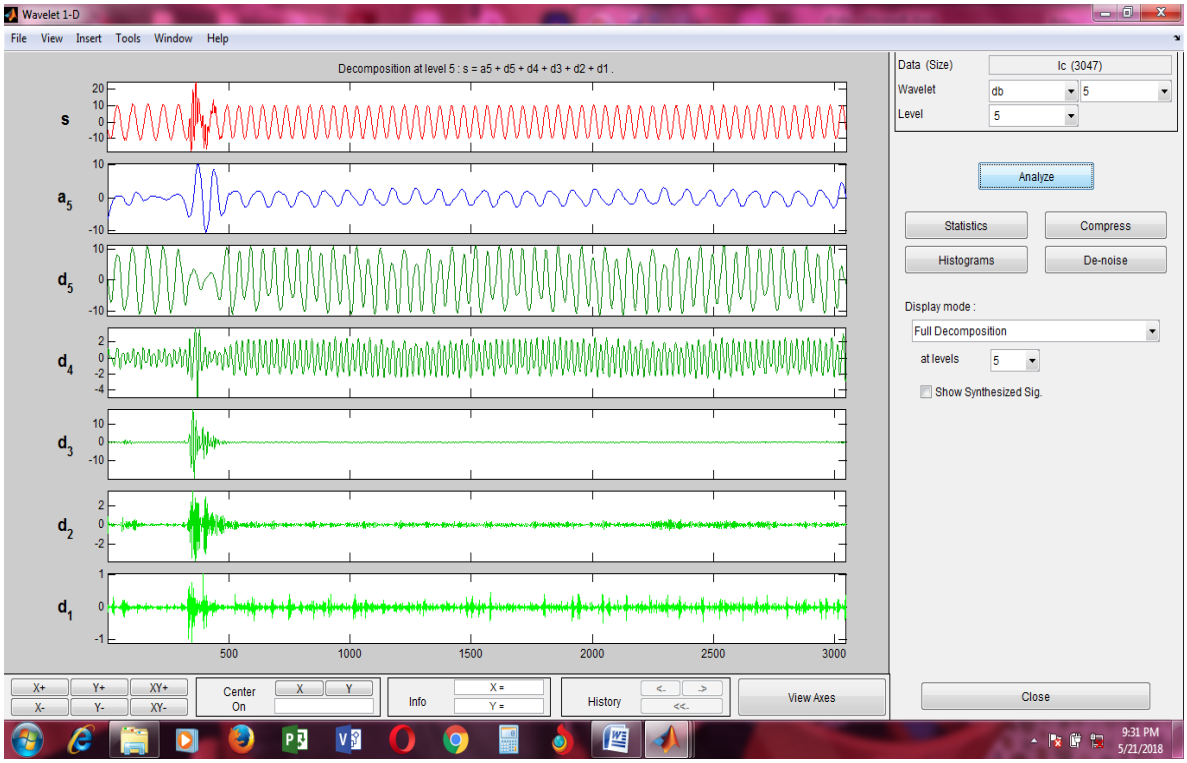
P3: Decomposed signal of voltage waveform of LG Fault of phase C at a distance of 140km from the source



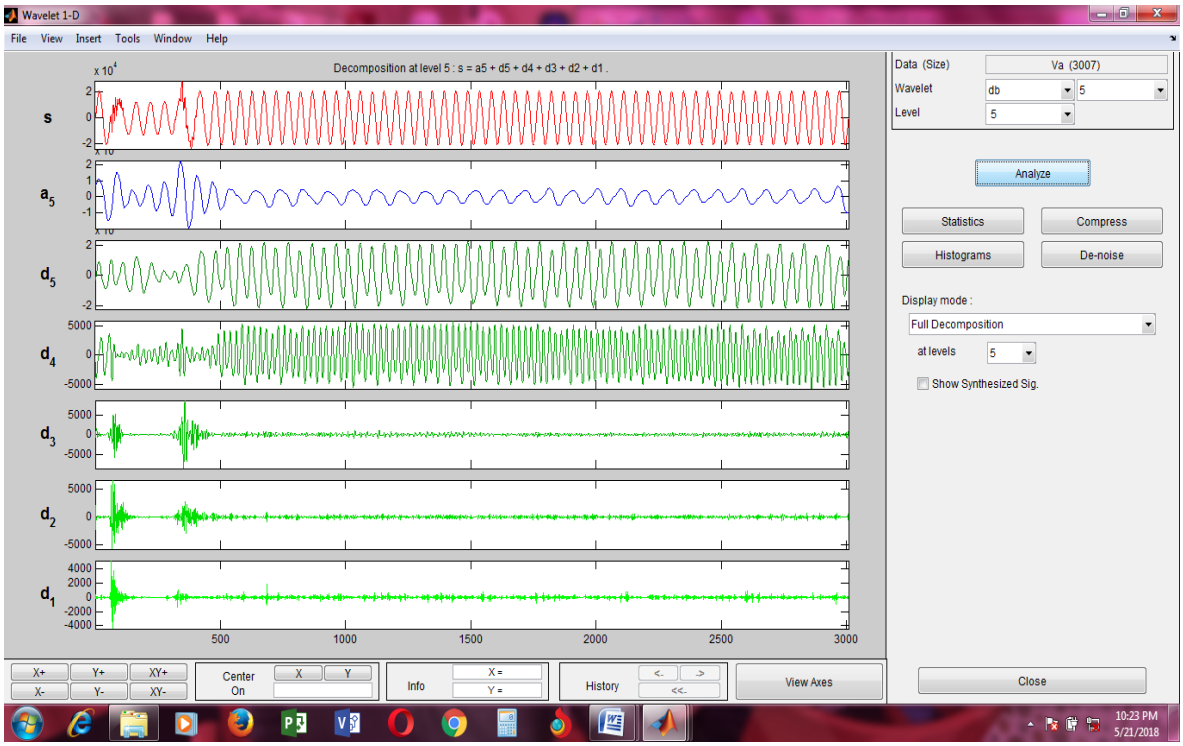
P4: Decomposed signal of current waveform of LG Fault of phase A at a distance of 140km from the source



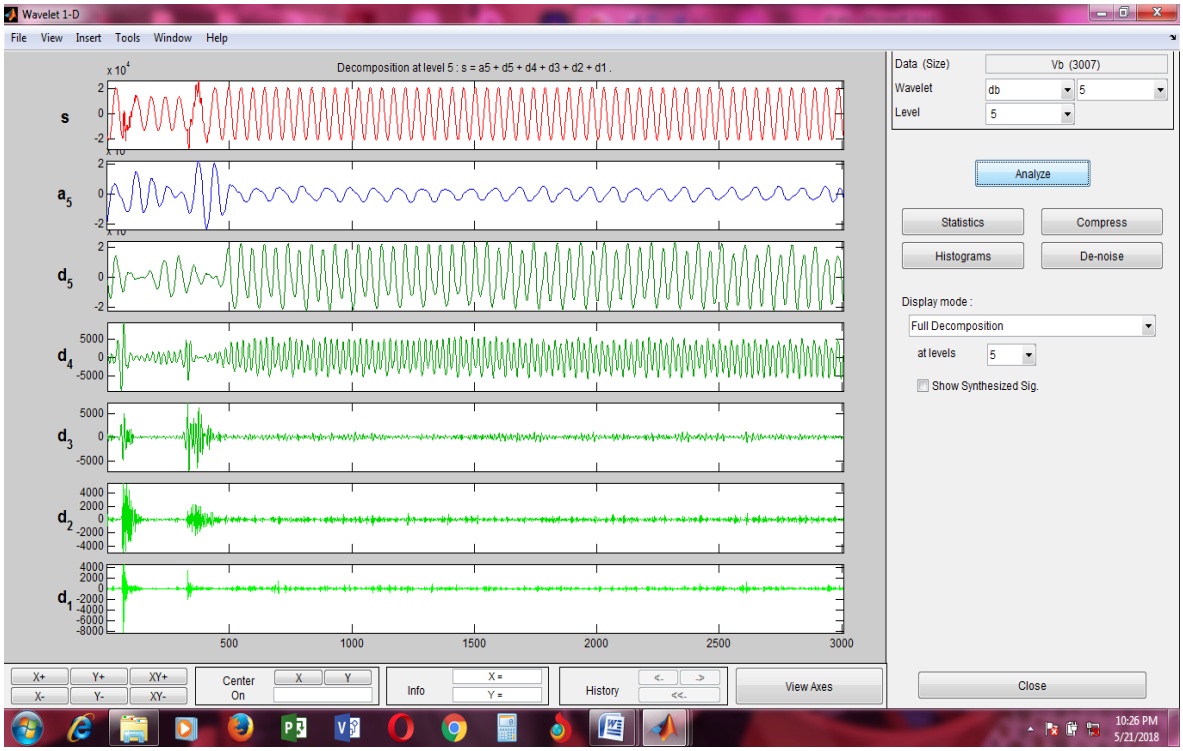
P5: Decomposed signal of current waveform of LG Fault of phase B at a distance of 140km from the source



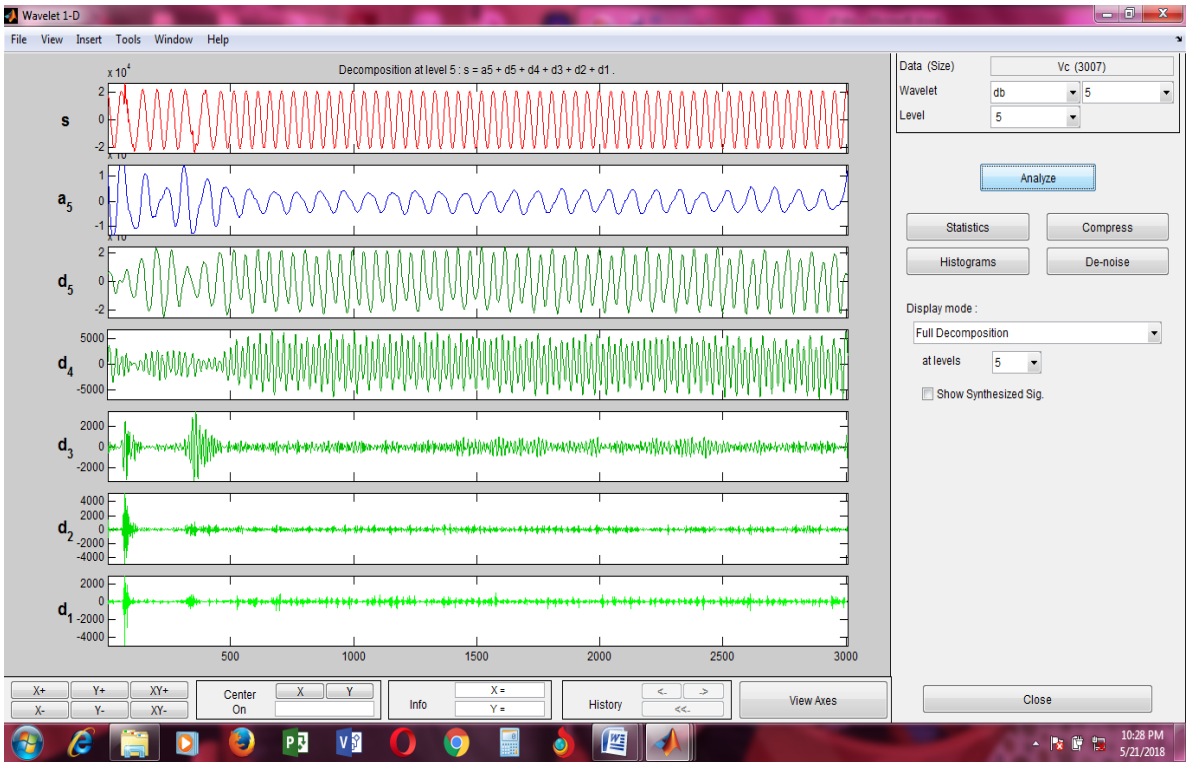
P6: Decomposed signal of current waveform of LG Fault of phase C at a distance of 140km from the source



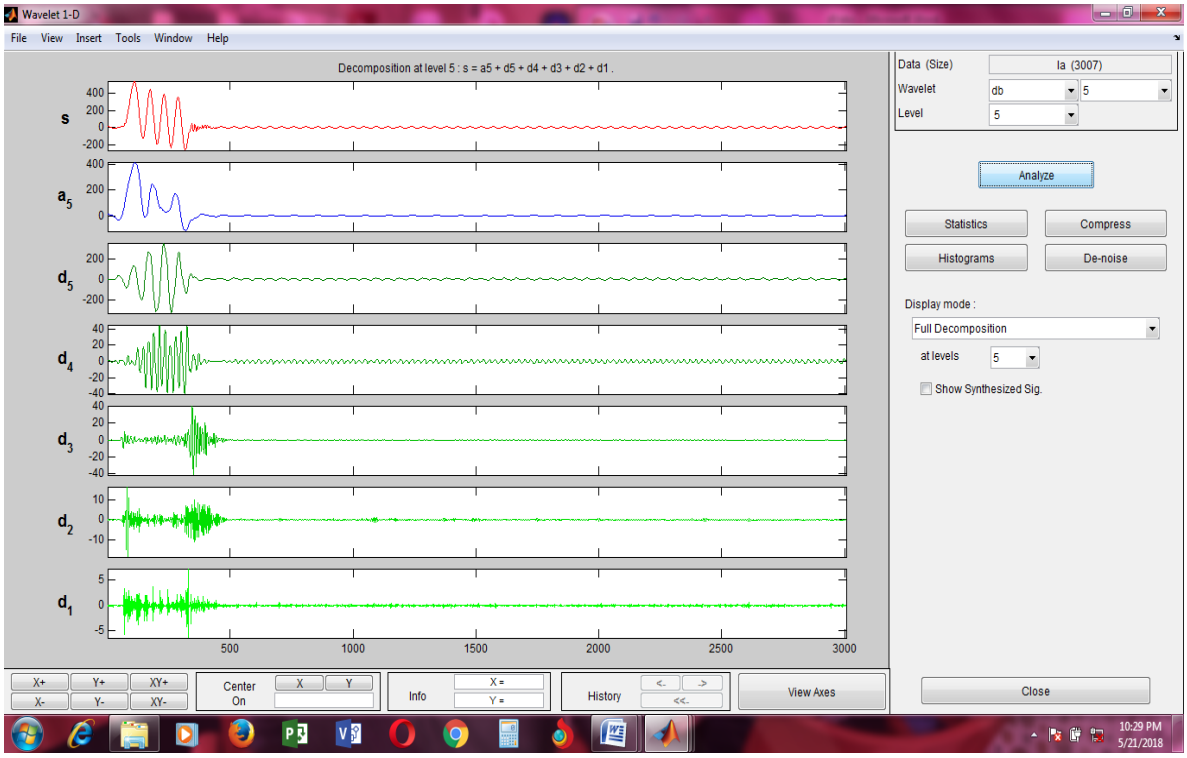
P7: Decomposed signal of voltage waveform of LLG Fault of phase A at a distance of 140km from the source



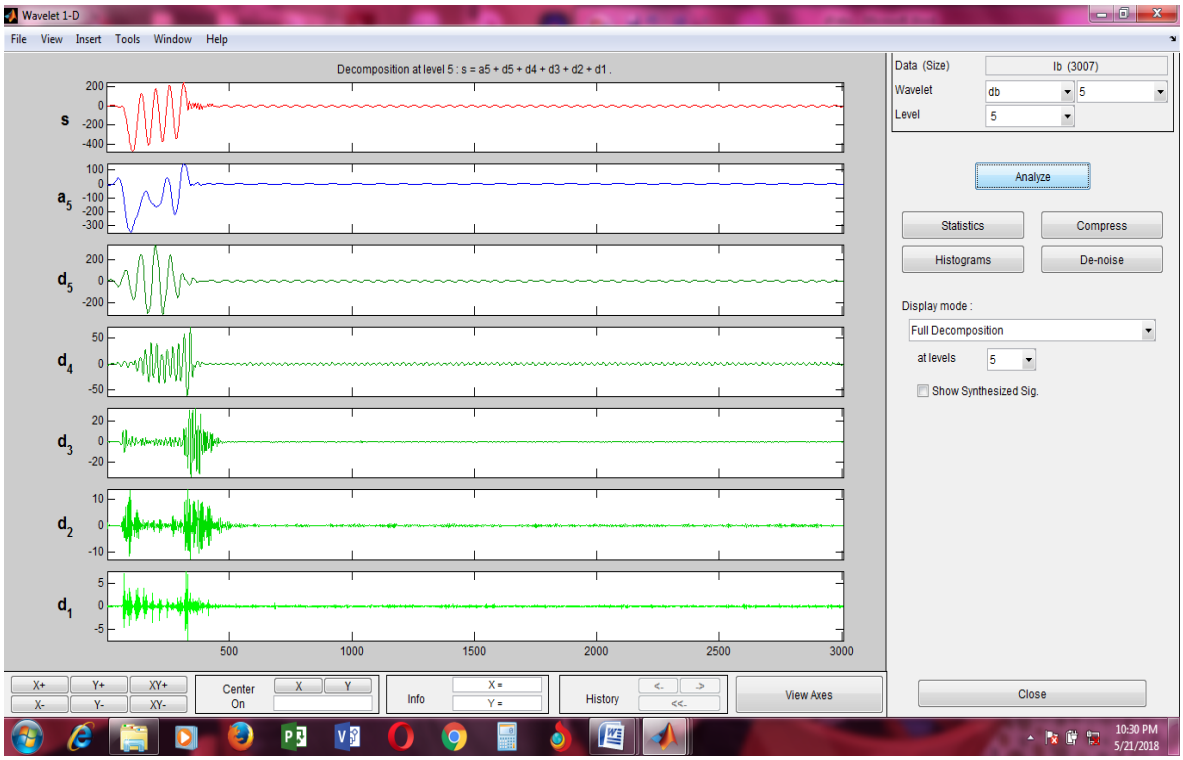
P8: Decomposed signal of voltage waveform of LLG Fault of phase B at a distance of 140km from the source



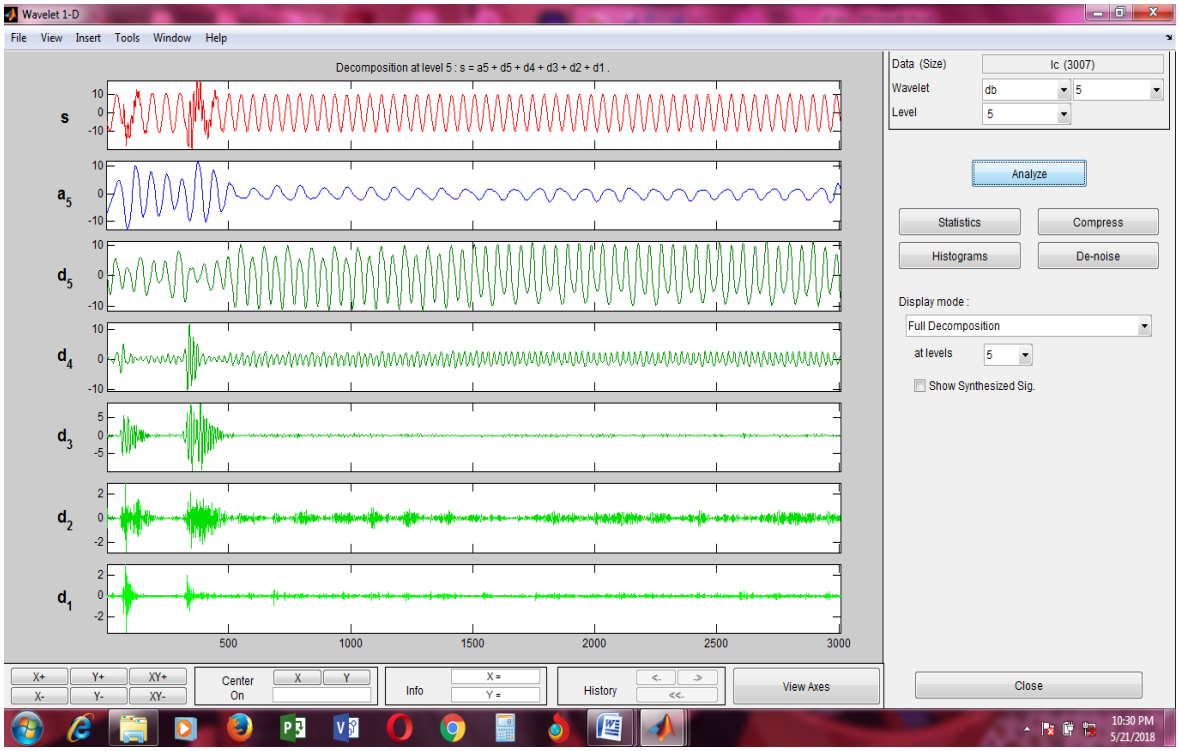
P9: Decomposed signal of voltage waveform of LLG Fault of phase C at a distance of 140km from the source



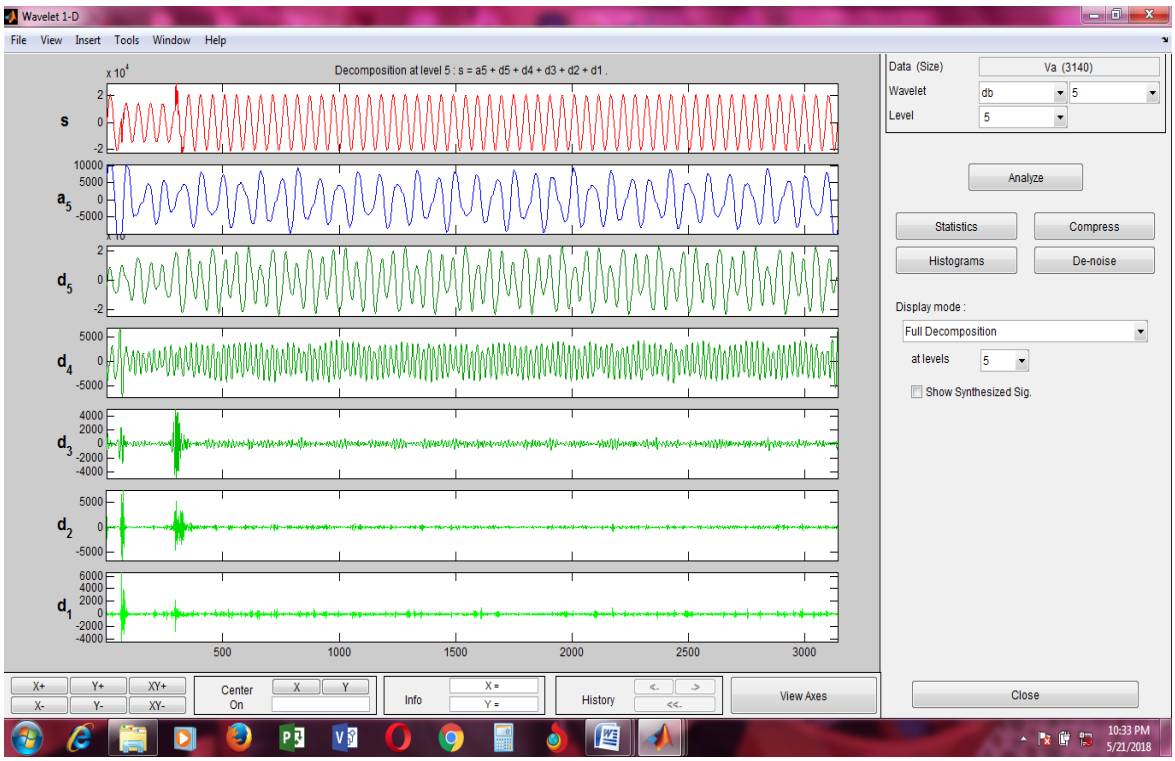
P10: Decomposed signal of current waveform of LLG Fault of phase A at a distance of 140km from the source



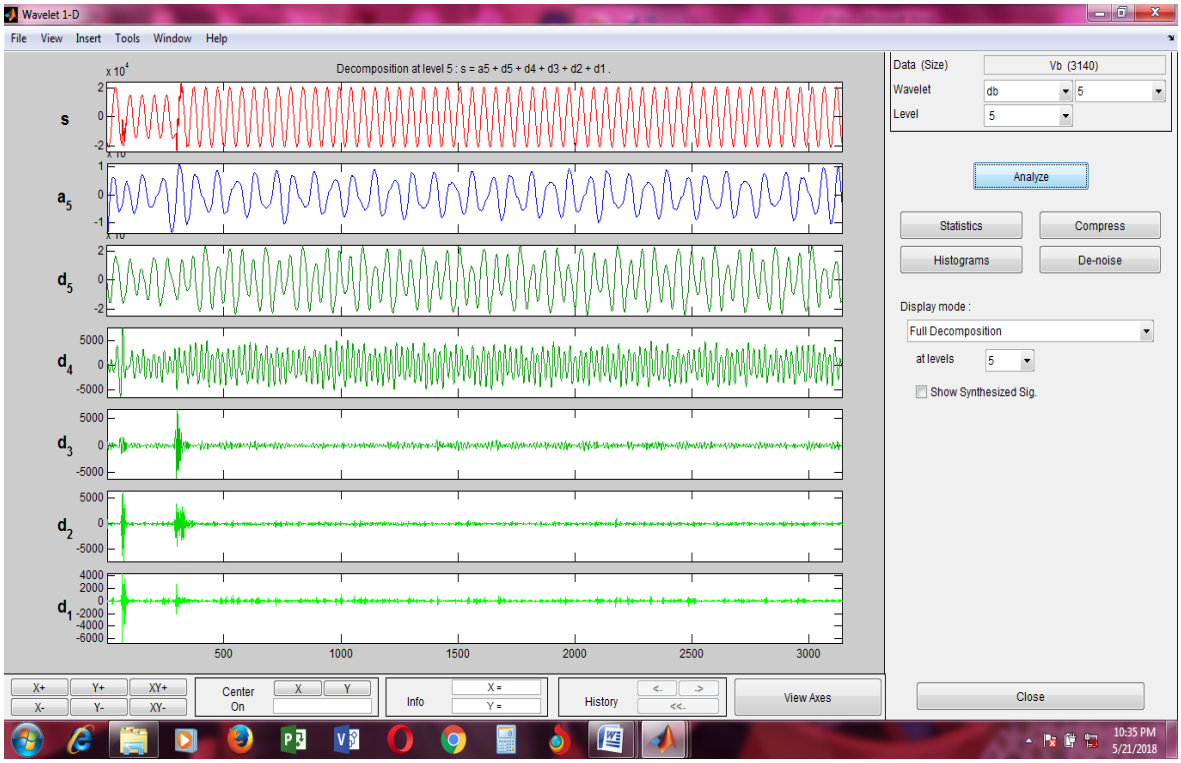
P11: Decomposed signal of current waveform of LLG Fault of phase B at a distance of 140km from the source



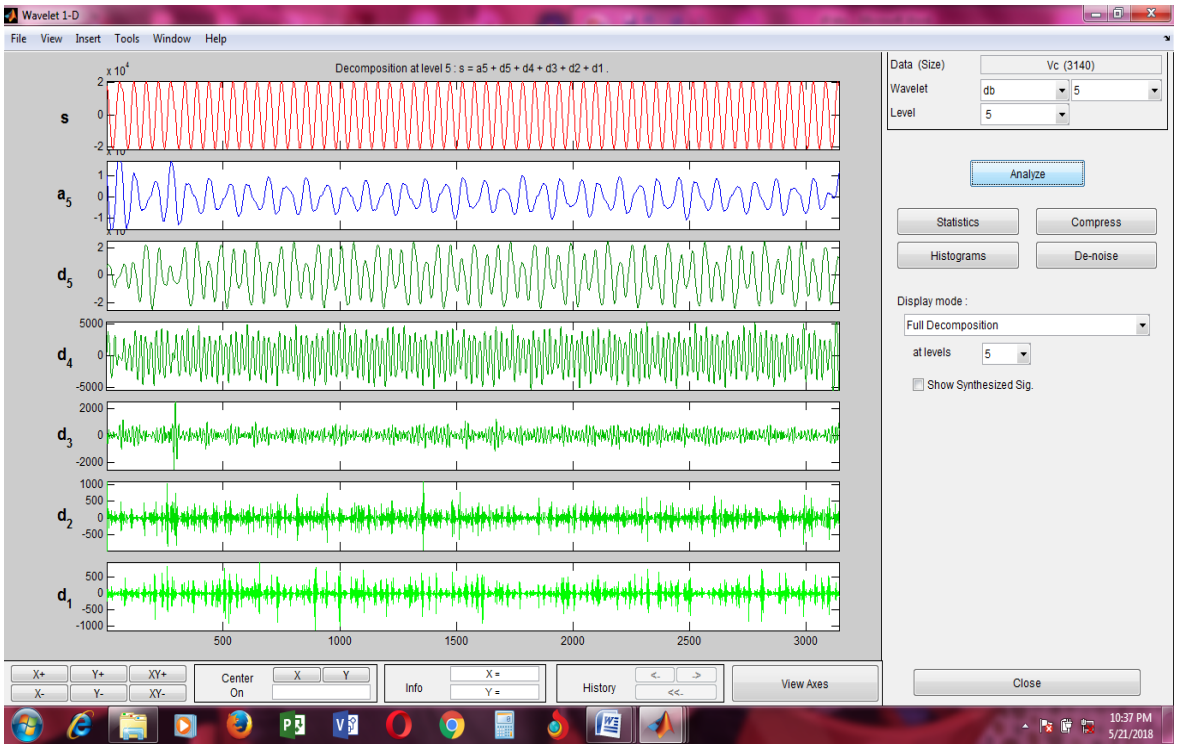
P12: Decomposed signal of current waveform of LLG Fault of phase C at a distance of 140km from the source



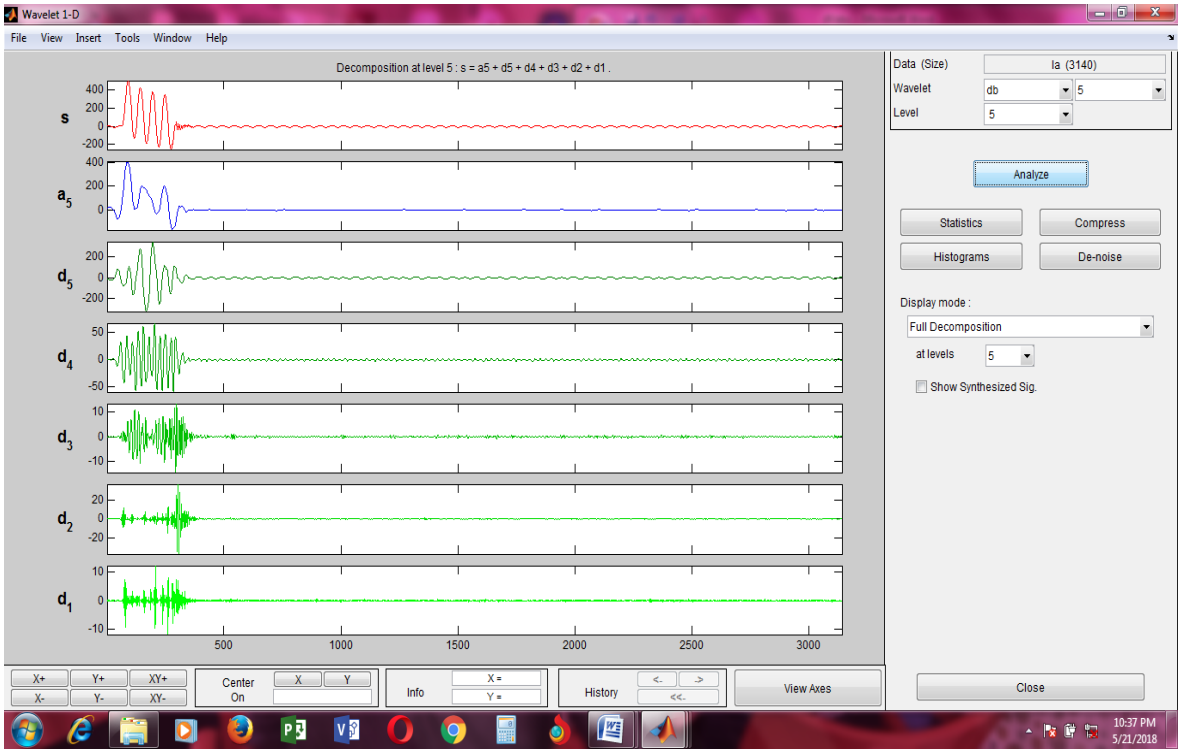
P13: Decomposed signal of voltage waveform of LL Fault of phase A at a distance of 140km from the source



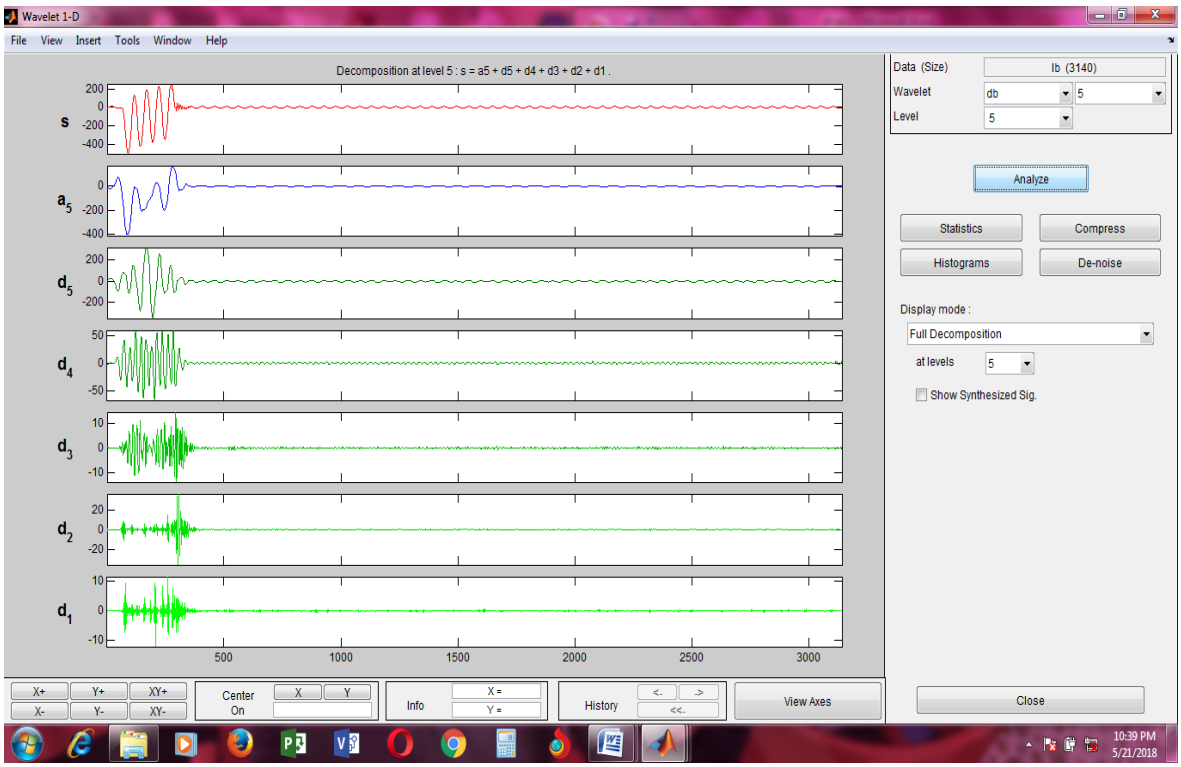
P14: Decomposed signal of voltage waveform of LL Fault of phase B at a distance of 140km from the source



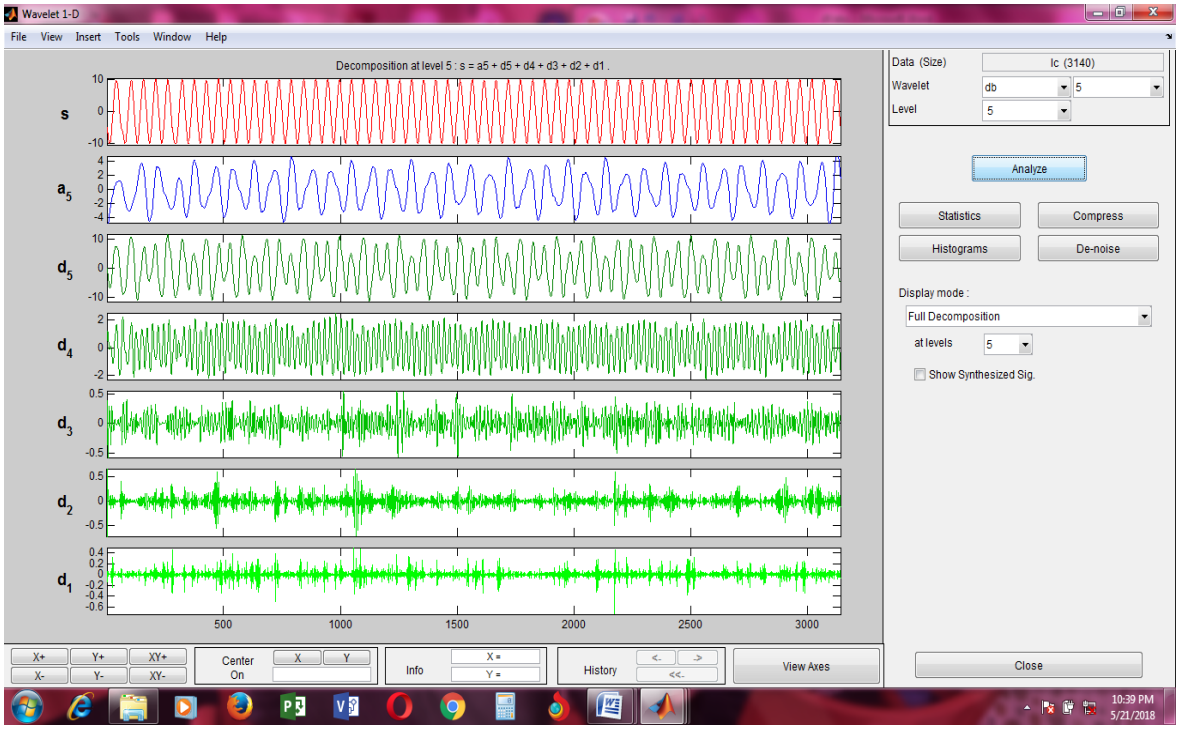
P15: Decomposed signal of voltage waveform of LL Fault of phase C at a distance of 140km from the source



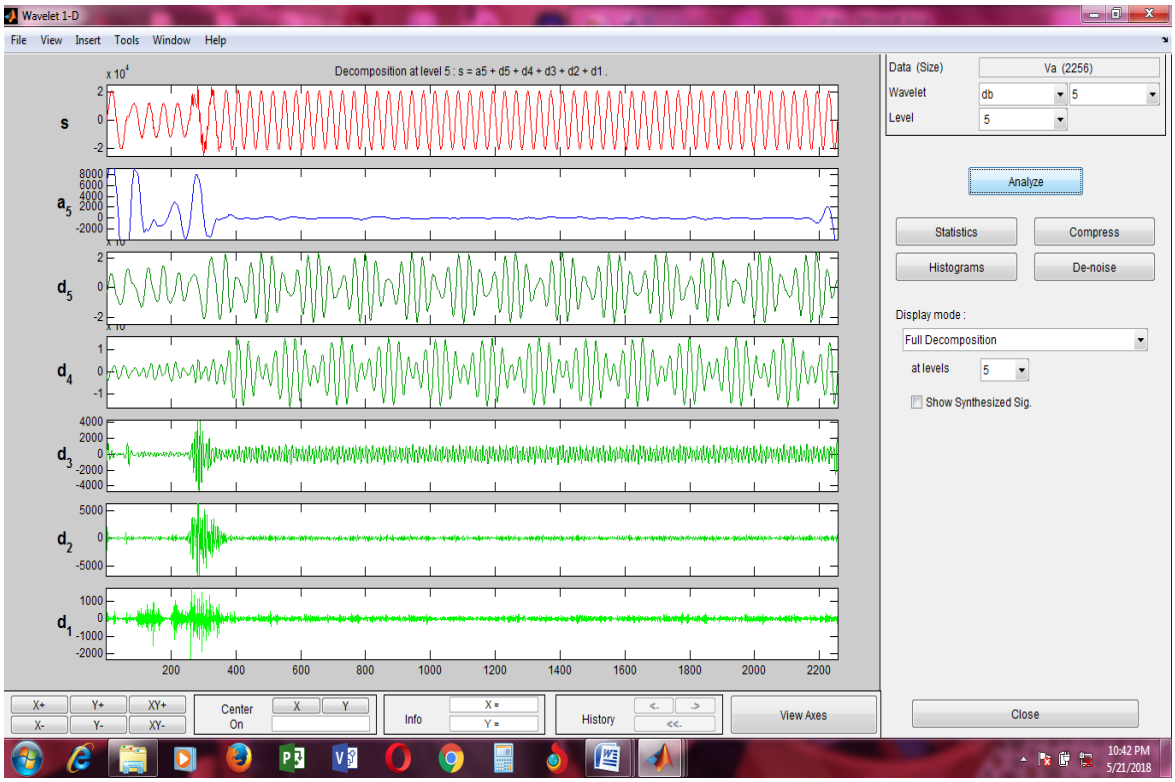
P16: Decomposed signal of current waveform of LL Fault of phase A at a distance of 140km from the source



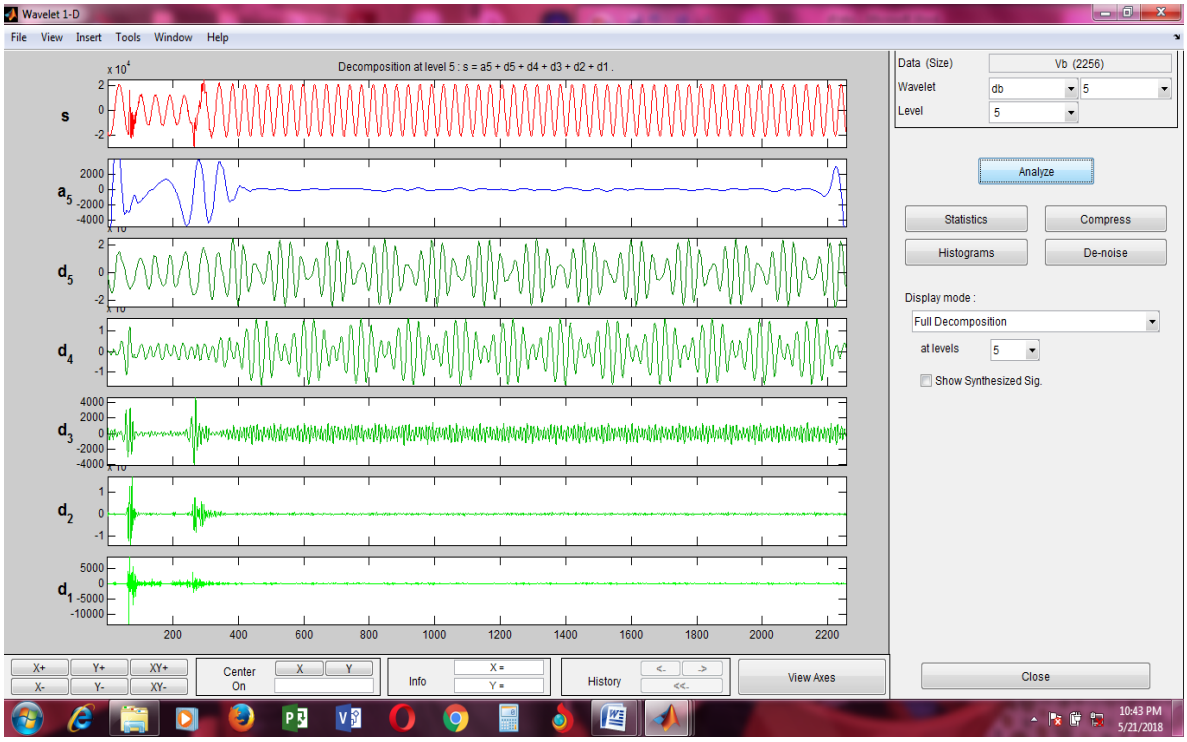
P17: Decomposed signal of current waveform of LL Fault of phase B at a distance of 140km from the source



P18: Decomposed signal of current waveform of LL Fault of phase C at a distance of 140km from the source

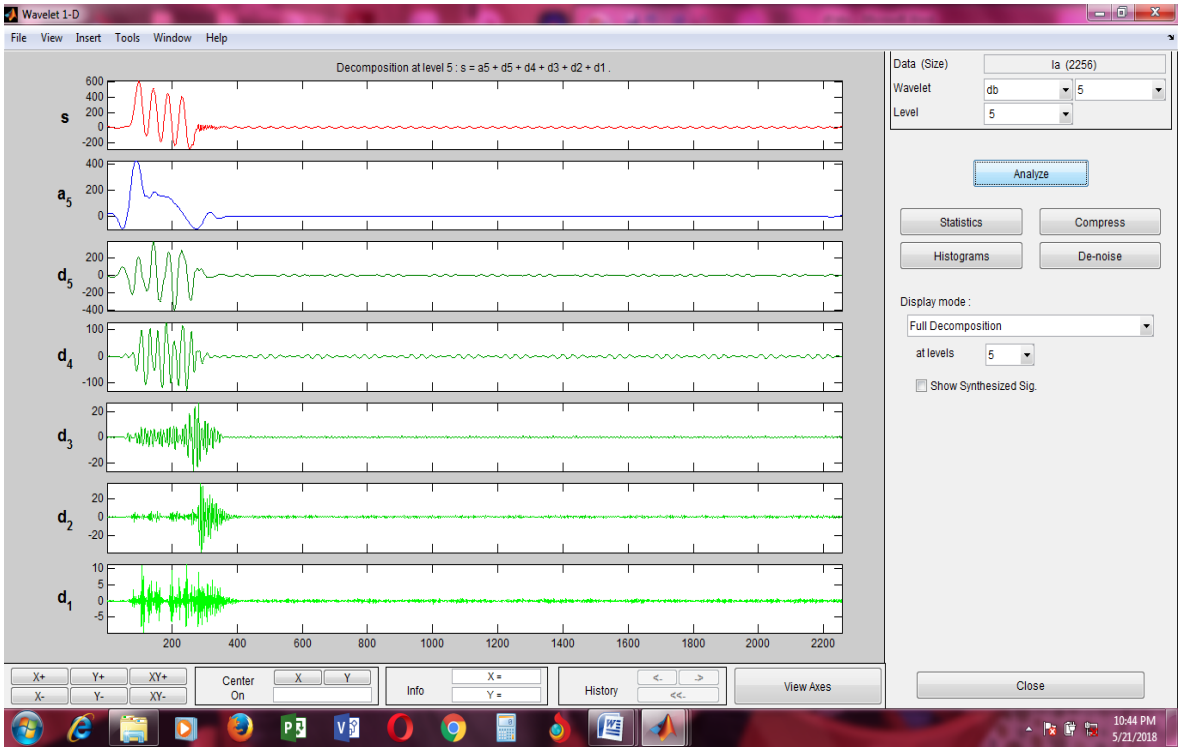


P19: Decomposed signal of voltage waveform of LLL Fault of phase A at a distance of 140km from the source

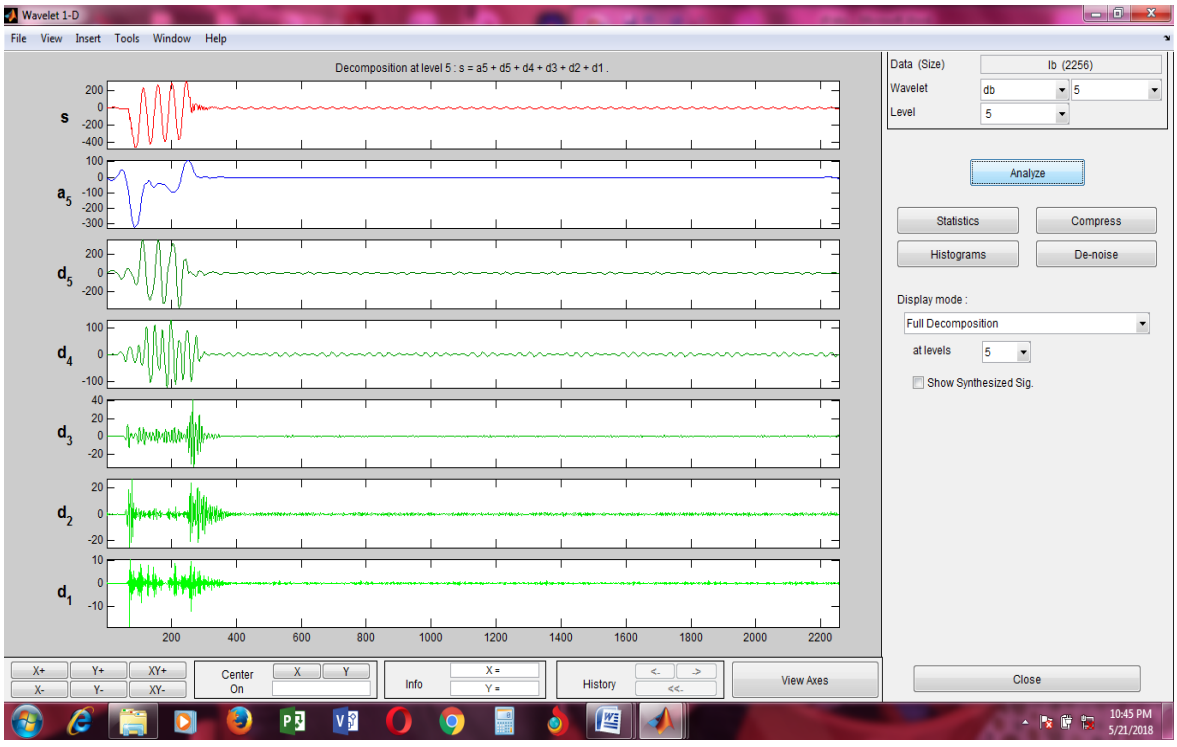


P20: Decomposed signal of voltage waveform of LLL Fault of phase B at a distance of 140km from the source

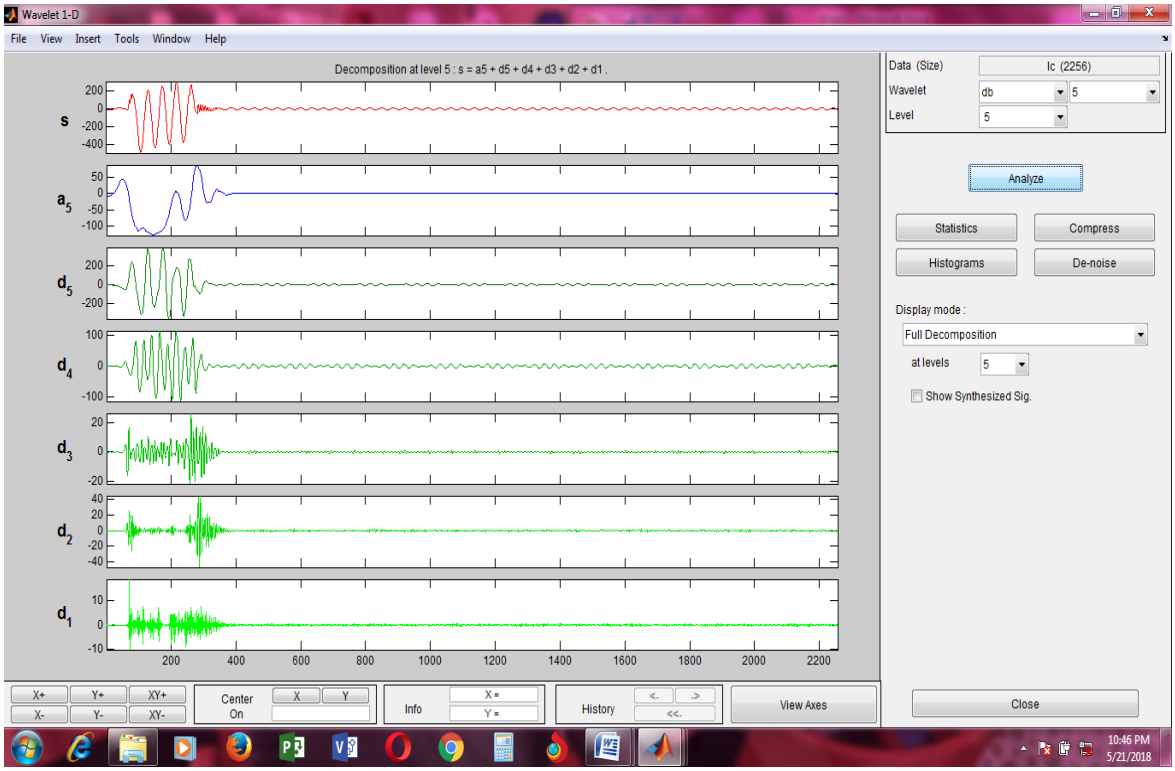
P21: Decomposed signal of voltage waveform of LLL Fault of phase C at a distance of 140km from the source



P22: Decomposed signal of current waveform of LLL Fault of phase A at a distance of 140km from the source



P23: Decomposed signal of current waveform of LLL Fault of phase B at a distance of 140km from the source



P24: Decomposed signal of current waveform of LLL Fault of phase C at a distance of 140km from the source

G NIGERIAN 330KV TRANSMISSION LINE PARAMETERS (Ogbuefi and Madueme, 2015)

S/N	CIRCUIT (BUSES)		LENGHT	IMPEDANCE	ADMITTANCE	SHUNT
	FROM	TO	(KM)	Z.(p.u.)	Y (pu)	$\frac{y}{2}(p.u)$
1.	Akamgbe	Ik-West	17	0.0006+j0.0051	22.75-j19.32	0.065
2.	Ayede	Oshogbo	115	0.0041+jo.o349	3.333-j38.37	0.437
3.	Ik-West	Egbin	62	0.0022+j0.0172	7.308-j57.14	0.257
4.	Ik-West	Benin	280	0.0101+j0.0799	1.637-j12.626	1.162
5.	Oshogbo	Jebba	249	0.0056+j0.477	0.0246-j3.092	0.597
6.	Oshogbo	Benin	251	0.0089+j0.0763	1.508-j12.932	0.954
7.	JebbaTs	JebbaGs	8	0.003+j0.0022	3.174-j1.594	0.033
8.	Jebba TS	Shiroro	244	0.0087+j0.0742	1.559-j13.297	0.927
9.	Jebba TS	Kainji	81	0.0022+j0.0246	3.607-j40.328	0.308
10.	Kainji	B.Kebbi	310	0.0111+j0.942	1.235-j10.478	1.178
11.	Shiroro	Kaduna	96	0.0034+j0.0292	3.935-j3.379	0.364
12.	Kaduna	Kano	320	0.0082+j0.0899	1.657-j14.17	0.874
13.	Jos	Gombe	265	0.0095+j0.081	1.923-j15.456	1.01
14	Benin	Ajaokuta	195	0.007+j0.056	1.429-j12.180	0.745
15.	Benin	Sapele	50	0.0018+j0.0139	3.194-j17.555	0.208
16.	Benin	Onitsha	137	0.0049+j0.0416	2.8-j33.771	0.521
17.	Onitsa	N.Heaven	96	0.0034+j0.0292	3.935-j3.379	0.0355
18.	Onitsha	Alaoji	138	0.0049+j0.0419	2.754-j33.553	0.524
19.	Alaoji	Afam	25	0.009+j0.007	59.230-j53.846	0.104
20.	Sapele	Aladja	63	0.0023+j0.019	5.284-j51.913	0.239
21.	Delta	Aladja	30	0.0011+j0.0088	13.995-j1.119	0.171
22.	*Kainji GS	Jebba TS	81	0.0022+j0.0246	3.607-j40.328	0.308
23.	Ayede	Ik West	137	0.0049+j0.0416	2.8-j33.771	0.521
24.	Egbin TS	Aja	27.5	0.0022+j0.0172	7.316-j57.2036	0.257
25	Egbin TS	Aja	27.5	0.0022+j0.0172	7.316 –j57.2036	0.257
26.	Kaduna	Jos	197	0.007+j0.0599	1.924 –j16.469	0.748
27.	Jos	Makurdi	275	0.0029+j0.0246	4.726-j40.093	0
28.	Oshogbo	Ik West	252	0.0049+j0.0341	4.128-j28.732	0.521
29.	Benin	Delta	107	0.0022+j0.019	6.013-j51.935	0.239
30	Onitsha	Okpai	80	0.009+j0.007	59.230-j53.846	0.104
31.	Geregu	Ajokuta	5	0.0022+j0.0172	7.316-j57.203	0.257
32.	Shiroro TS	Kaduna	96	0.0034+j0.0292	3.935-j3.379	0.364

H PART OF RESULTS OF APPENDIX B

```
>> w

w =

  1.0e+04 *

 Columns 1 through 7

   -0.6537   -0.4414   -0.2693   -0.2993   -0.3309   -0.1639   -
0.0056
   -0.0918   -0.2976   -0.4773   -0.5936   -0.7209   -0.9091   -
1.0840
    0.7409    0.7350    0.7432    0.8903    1.0499    1.0720
1.0894

 Columns 8 through 14

    0.1093    0.2839    0.4454    0.5838    0.6953    0.7768
0.8587
   -1.2469   -1.3933   -1.5038   -1.5808   -1.6280   -1.6614   -
1.6615
    1.1384    1.1111    1.0611    1.0006    0.9373    0.8901
0.8091

 Columns 15 through 21

    0.9272    0.9793    1.0100    1.0244    1.0178    0.9828
0.8991
   -1.6280   -1.5650   -1.4940   -1.3977   -1.2471   -1.0776   -
0.8151
    0.7080    0.5936    0.4927    0.3827    0.2394    0.1056   -
0.0726

 Columns 22 through 28

    0.8374    0.7694    0.5953    0.5054    0.4180    0.2149
0.0853
   -0.6663   -0.5054   -0.1689   -0.0038    0.1571    0.4321
0.5991
   -0.1591   -0.2516   -0.4135   -0.4883   -0.5614   -0.6329   -
0.6699

 Columns 29 through 35

   -0.0477   -0.2175   -0.3911   -0.5487   -0.7022   -0.8302   -
0.9477
    0.7594    0.9383    1.0962    1.2236    1.3398    1.4444
1.5424
   -0.6967   -0.7054   -0.6891   -0.6585   -0.6207   -0.5968   -
0.5770

 Columns 36 through 42

   -1.1034   -1.1941   -1.2522   -1.2917   -1.3199   -1.3308   -
1.3363
    1.6439    1.7219    1.7841    1.8407    1.8843    1.9084
1.8992
   -0.5225   -0.5095   -0.5132   -0.5301   -0.5453   -0.5581   -
0.5425

 Columns 43 through 49

   -1.3199   -1.2931   -1.2495   -1.1881   -1.1146   -1.0279   -
0.9271
    1.8707    1.8104    1.7338    1.6404    1.5095    1.3657
1.2141
   -0.5291   -0.4938   -0.4585   -0.4238   -0.3631   -0.3021   -
0.2469

 Columns 50 through 56

   -0.8122   -0.6679   -0.5148   -0.3265   -0.1620   -0.0039
0.1504
    1.0537    0.8472    0.6568    0.4149    0.2052    0.0011   -
0.2115
   -0.1964   -0.1288   -0.0856   -0.0256    0.0269    0.0810
0.1481

 Columns 57 through 63

    0.3106    0.4769    0.6989    0.8742    1.0180    0.9689
0.9177
   -0.4211   -0.6140   -0.8309   -0.9872   -1.1087   -1.1338   -
1.1540
    0.2068    0.2433    0.2484    0.2401    0.2290    0.3150
0.3985

 Columns 64 through 70

    1.0707    1.1296    1.0666    1.0409    1.0534    1.1159
1.1617
   -1.2614   -1.3129   -1.4367   -1.5414   -1.6347   -1.7001   -
1.7931
    0.3654    0.3712    0.3467    0.3679    0.4163    0.4746
0.4785

 Columns 71 through 77

    1.1034    1.0153    0.9597    0.9667    0.9027    0.8136
0.7158
   -1.8669   -1.9083   -1.9460   -1.9067   -1.8819   -1.8491   -
1.8115
    0.4697    0.5110    0.5769    0.6369    0.6937    0.6692
0.6801

 Columns 78 through 84

    0.6614    0.6385    0.5727    0.4602    0.3572    0.3040
0.2350
   -1.7256   -1.6138   -1.4521   -1.2924   -1.1201   -0.9779   -
0.8396
    0.6440    0.5757    0.4611    0.3887    0.3194    0.3286
0.3200

 Columns 85 through 91

    0.1407    0.0448   -0.0540   -0.1042   -0.1578   -0.2231   -
0.3244
   -0.6425   -0.4474   -0.2045    0.0292    0.2574    0.4525
0.6644
    0.2615    0.1735    0.0539   -0.0783   -0.2002   -0.2944   -
0.3782

 Columns 92 through 98
```

-0.3992 -0.4679 -0.5506 -0.6412 -0.7465 -0.8488 -0.9298
0.8417 1.0201 1.2082 1.3680 1.5059 1.6338 1.7528
-0.4300 -0.4614 -0.4825 -0.4874 -0.4704 -0.4519 -0.4424

Columns 99 through 105

-0.9827 -1.0299 -1.0741 -1.0660 -1.0373 -1.0192 -0.9885
1.8557 1.9524 2.0363 2.1002 2.1616 2.2045 2.2257
-0.4499 -0.4732 -0.4881 -0.5329 -0.5880 -0.6136 -0.6506

Columns 106 through 112

-0.9576 -0.9246 -0.8879 -0.8356 -0.7845 -0.7422 -0.6804
2.2226 2.1797 2.1064 2.0274 1.9250 1.7407 1.5349
-0.6751 -0.6673 -0.6351 -0.6276 -0.6105 -0.4843 -0.3506

Columns 113 through 119

-0.6048 -0.5132 -0.4098 -0.2969 -0.2034 -0.1318 -0.0612
1.3358 1.0808 0.8436 0.6481 0.4364 0.2469 0.0378
-0.2528 -0.1415 -0.0752 -0.0360 -0.0027 0.0489 0.1302

Columns 120 through 126

0.0130 0.0938 0.1776 0.2571 0.3926 0.4746 0.5543
-0.1985 -0.4284 -0.6411 -0.8438 -1.0831 -1.2335 -1.3879
0.2378 0.3411 0.4188 0.4725 0.4822 0.4980 0.5033

Columns 127 through 133

0.6636 0.7773 0.8741 0.9536 1.0406 1.1037 1.1223
-1.5482 -1.6761 -1.7902 -1.8868 -1.9770 -2.0554 -2.1075
0.4715 0.4218 0.3751 0.3520 0.3151 0.2901 0.3133

Columns 134 through 140

1.1187 1.1414 1.1293 1.0861 1.0376 1.0087 0.9753
-2.1554 -2.1773 -2.1833 -2.1863 -2.1737 -2.1002 -1.9970
0.3482 0.3251 0.3417 0.4004 0.4511 0.4214 0.3774

Columns 141 through 147

0.9295 0.8636 0.7858 0.6994 0.6029 0.4956 0.3924
-1.8631 -1.7006 -1.5152 -1.3107 -1.1071 -0.9073 -0.7254
0.3270 0.2728 0.2101 0.1422 0.0900 0.0490 0.0301

Columns 148 through 154

0.2925 0.1978 0.1094 0.0107 -0.0758 -0.1421 -0.2094
-0.5433 -0.3551 -0.1604 0.0664 0.2888 0.4837 0.6661
0.0224 0.0017 -0.0407 -0.1030 -0.1816 -0.2548 -0.3079

Columns 155 through 161

-0.2841 -0.3961 -0.5032 -0.6322 -0.6983 -0.7784 -0.9036
0.8370 1.0213 1.1854 1.3580 1.4765 1.5849 1.7010
-0.3521 -0.3640 -0.3603 -0.3302 -0.3142 -0.2808 -0.2056

Columns 162 through 168

-0.9944 -1.0575 -1.1040 -1.1241 -1.1410 -1.1465 -1.1529
1.7919 1.8750 1.9470 2.0035 2.0459 2.0728 2.0734
-0.1666 -0.1448 -0.1326 -0.1499 -0.1675 -0.1973 -0.1947

Columns 169 through 175

-1.1304 -1.0854 -1.0407 -1.0096 -0.9629 -0.8843 -0.7981
2.0585 2.0371 1.9979 1.8856 1.7196 1.5816 1.4306
-0.2023 -0.2478 -0.2785 -0.2271 -0.1464 -0.1349 -0.1127

Columns 176 through 182

-0.7171 -0.6294 -0.5295 -0.4155 -0.3209 -0.2322 -0.1338
1.2867 1.1336 0.9468 0.7450 0.5882 0.4213 0.2129
-0.1119 -0.1056 -0.0786 -0.0506 -0.0526 -0.0408 0.0032

Columns 183 through 189

-0.0299 0.0661 0.1382 0.2156 0.3049 0.4197 0.5328
-0.0084 -0.2331 -0.4289 -0.6171 -0.7999 -0.9804 -1.1443
0.0606 0.1313 0.1962 0.2411 0.2740 0.2791 0.2690

Columns 190 through 196

0.6157 0.7007 0.8184 0.9243 1.0192 1.0889 1.1205
-1.2830 -1.4101 -1.5319 -1.6348 -1.7313 -1.8165 -1.8945
0.2654 0.2434 0.1776 0.1194 0.0709 0.0365 0.0513

Columns 197 through 203

1.1473 1.2227 1.2218 1.2059 1.1749 1.1391 1.1054
-1.9617 -1.9916 -2.0206 -2.0377 -2.0375 -2.0218 -1.9621
0.0712 0.0011 0.0257 0.0710 0.1013 0.1481 0.1533

Columns 204 through 210

1.0644 1.0072 0.9346 0.8529 0.7631 0.6623 0.5580
-1.8769 -1.7511 -1.6193 -1.4448 -1.2713 -1.1137 -0.9328
0.1370 0.0987 0.0865 0.0506 0.0306 0.0212 0.0127

Columns 211 through 217

0.4453 0.3470 0.2498 0.1492 0.0366 -0.0596 -0.1406
-0.7368 -0.5792 -0.4101 -0.2201 0.0001 0.2198 0.4200
0.0018 0.0128 0.0118 -0.0170 -0.0636 -0.1227 -0.1780

Columns 218 through 224

-0.2315 -0.3303 -0.4526 -0.5942 -0.6657 -0.7418 -0.8631
0.6139 0.7955 0.9739 1.1534 1.2745 1.3909 1.5025
-0.2121 -0.2334 -0.2248 -0.1830 -0.1769 -0.1533 -0.0742

Columns 225 through 231

-0.9710 -1.0499 -1.1253 -1.1812 -1.2069 -1.2286 -1.2373
1.5943 1.6819 1.7619 1.8331 1.8944 1.9380 1.9664
-0.0071 0.0343 0.0806 0.1048 0.0890 0.0747 0.0618

Columns 232 through 238

-1.2229 -1.1966 -1.1664 -1.1264 -1.0823 -1.0216 -0.9441
1.9859 1.9864 1.9583 1.9063 1.8056 1.6864 1.5526
0.0262 -0.0111 -0.0323 -0.0450 -0.0153 0.0068 0.0166

Columns 239 through 245

-0.8576 -0.7654 -0.6674 -0.5680 -0.4614 -0.3550 -0.2539
1.3965 1.2515 1.0898 0.9273 0.7551 0.5892 0.4244
0.0287 0.0286 0.0286 0.0237 0.0209 0.0095 0.0078

Columns 246 through 252

-0.0910 -0.0049 0.0834 0.1614 0.2824 0.3767 0.4631
0.1305 -0.0518 -0.2397 -0.4372 -0.6408 -0.8119 -0.9599
0.0388 0.0653 0.1192 0.1648 0.1865 0.1986 0.1958

Columns 253 through 259

0.5467 0.6573 0.7630 0.8766 0.9587 1.0357 1.1188
-1.0936 -1.2393 -1.3584 -1.4638 -1.5486 -1.6340 -1.7102
0.1885 0.1551 0.1087 0.0275 -0.0285 -0.0770 -0.1342

Columns 260 through 266

1.1763 1.2107 1.2193 1.2413 1.2308 1.2023 1.1544
-1.7766 -1.8321 -1.8755 -1.8841 -1.8887 -1.8783 -1.8630
-0.1521 -0.1573 -0.1455 -0.1779 -0.1601 -0.1219 -0.0792

Columns 267 through 273

1.1075 1.0564 0.9887 0.8961 0.7946 0.6868 0.5882
-1.7836 -1.6689 -1.5321 -1.3712 -1.2199 -1.0643 -0.9164
-0.0961 -0.1223 -0.1458 -0.1596 -0.1523 -0.1484 -0.1231

Columns 274 through 280

0.4887 0.3777 0.2428 0.1434 0.0541 -0.0419 -0.1433
-0.7771 -0.6156 -0.4220 -0.2752 -0.1130 0.0763 0.2848
-0.0951 -0.0783 -0.0642 -0.0450 -0.0488 -0.0737 -0.1151

Columns 281 through 287

-0.2345 -0.3146 -0.3877 -0.4978 -0.5982 -0.6995 -0.7870
0.4786 0.6444 0.7946 0.9559 1.0907 1.2182 1.3221
-0.1526 -0.1754 -0.1923 -0.1697 -0.1408 -0.1026 -0.0505

Columns 288 through 294

-0.8642 -0.9717 -1.0488 -1.1016 -1.1181 -1.1170 -1.1734
1.4011 1.4781 1.5399 1.5943 1.6493 1.6957 1.6765
0.0098 0.1097 0.1733 0.2126 0.2139 0.1894 0.3023

Columns 295 through 301

-1.1464 -1.1020 -1.0479 -0.9833 -0.9145 -0.8372 -0.7394
1.6777 1.6714 1.6297 1.5789 1.4685 1.3522 1.2199
0.2727 0.2195 0.2135 0.1712 0.1775 0.1757 0.1688

Columns 302 through 308

-0.6321 -0.5238 -0.4203 -0.3185 -0.2177 -0.0841 0.0426
1.0641 0.9447 0.8035 0.6750 0.5511 0.3590 0.1874
0.1535 0.1142 0.0789 0.0376 -0.0020 -0.0228 -0.0621

Columns 309 through 315

0.1340 0.2085 0.3074 0.3848 0.4649 0.5557 0.6171
0.0627 -0.0777 -0.2597 -0.4271 -0.5978 -0.7611 -0.8819
-0.0950 -0.1067 -0.0996 -0.0714 -0.0523 -0.0638 -0.0745

Columns 316 through 322

0.5484 0.4763 0.5301 0.5853 0.6619 0.7039 0.7269
-0.8859 -0.8893 -0.9984 -1.0834 -1.1521 -1.2010 -1.2426
0.0081 0.0939 0.0741 0.0386 -0.0384 -0.0986 -0.1397

Columns 323 through 329

0.7254 0.7533 0.7222 0.6662 0.5737 0.4759 0.3776
-1.2743 -1.2935 -1.2927 -1.2690 -1.2524 -1.2072 -1.1238
-0.1588 -0.2234 -0.2321 -0.2085 -0.1588 -0.1084 -0.0870

Columns 330 through 336

0.2682 0.1365 -0.0004 -0.1031 -0.0180 0.1586 0.4430
-1.0333 -0.9719 -1.0209 -0.8947 -0.9479 -0.9147 -0.6438
-0.0587 0.0292 0.2421 0.2351 0.2287 0.2879 0.5729

Columns 337 through 343

0.5285 0.6025 0.5792 0.3602 0.1223 -0.0758 -0.3035
-0.5122 -0.4327 -0.3166 -0.1371 0.0913 0.3112 0.3300
0.7054 0.7308 0.7341 0.8121 0.8880 0.9213 0.7982

Columns 344 through 350

-0.5704 -0.6950 -0.7358 -0.6727 -0.5643 -0.4561 -0.3315
0.2896 0.2802 0.2720 0.2793 0.2831 0.2718 0.4356
0.5787 0.3815 0.1392 -0.0646 -0.2740 -0.4967 -0.5004

Columns 351 through 357

-0.2256 -0.2183 -0.2318 -0.3508 -0.4970 -0.4735 -0.4824
0.6610 0.8690 1.0371 1.2240 1.4461 1.5301 1.4234
-0.3973 -0.2770 -0.1847 -0.0982 0.0006 -0.0398 -0.2640

Columns 358 through 364

-0.4403 -0.3115 -0.1602 -0.0391 0.0043 0.0041 0.0207
1.2327 1.0492 0.8953 0.7832 0.6972 0.6997 0.7880
-0.5416 -0.8144 -1.0586 -1.2429 -1.3853 -1.4046 -1.3077

Columns 365 through 371

0.0708 0.1424 0.2409 0.3727 0.5534 0.7063 0.7651
0.9074 0.9887 1.0169 0.9966 0.9340 0.7777 0.5668
-1.1599 -1.0411 -0.9596 -0.9043 -0.8588 -0.8593 -0.9154

Columns 372 through 378

0.7368 0.6351 0.4520 0.2086 -0.0296 -0.1388 -0.0857
0.3418 0.1563 0.0123 -0.1078 -0.2076 -0.2798 -0.3204
-0.9676 -0.9540 -0.8666 -0.7139 -0.4907 -0.1938 0.0530

Columns 379 through 385

0.0259 0.1951 0.3594 0.4253 0.3779 0.2525 0.1214
-0.3790 -0.4768 -0.5775 -0.6804 -0.8238 -0.8862 -0.9286
0.2705 0.4505 0.6189 0.7694 0.8860 0.9226 0.9740

Columns 386 through 392

-0.0770 -0.2648 -0.4145 -0.4833 -0.4364 -0.3373 -0.2843
-1.0215 -1.0999 -1.2054 -1.2899 -1.3268 -1.3313 -1.2673
1.0267 1.0868 1.1386 1.1907 1.2683 1.3642 1.4962

Columns 393 through 399

-0.3111 -0.4021 -0.5192 -0.6533 -0.6818 -0.7328 -0.7889
-1.1431 -0.9999 -0.9026 -0.8487 -0.9302 -0.9272 -0.9464
1.6509 1.7981 1.8795 1.8814 1.6891 1.5878 1.4620

Columns 400 through 406

-0.7355 -0.6764 -0.6045 -0.5708 -0.5311 -0.4754 -0.3843
-0.9437 -0.8964 -0.7864 -0.6089 -0.4133 -0.2211 -0.0788

1.3302 1.2117 1.0947 1.0674 1.0396 0.9659
0.8354

Columns 407 through 413

-0.2473 -0.0508 0.0730 0.1479 0.2141 0.2305
0.2247
0.0175 0.0841 0.0791 0.0861 0.1332 0.2302
0.3739
0.6286 0.2931 0.0033 -0.2703 -0.5102 -0.7386 -
0.9067

Columns 414 through 420

0.2381 0.2855 0.3597 0.4424 0.5275 0.5546
0.5141
0.5683 0.7898 0.9685 1.1470 1.2492 1.3612
1.4206
-1.0538 -1.1572 -1.2332 -1.3441 -1.4644 -1.6216 -
1.7945

Columns 421 through 427

0.4721 0.4460 0.3912 0.4168 0.5364 0.6658
0.7773
1.4470 1.4244 1.4542 1.4635 1.4269 1.4048
1.3860
-1.9095 -2.0103 -2.0867 -2.1264 -2.1466 -2.1344 -
2.0963

Columns 428 through 434

0.8043 0.7759 0.6487 0.5129 0.4268 0.3596
0.2738
1.3970 1.3773 1.3469 1.2585 1.1508 1.0244
0.8941
-2.0106 -1.9218 -1.7963 -1.7025 -1.6186 -1.5414 -
1.3861

Columns 435 through 441

0.2206 0.1848 0.1333 0.0558 -0.0517 -0.2200 -
0.3587
0.7760 0.6694 0.5937 0.5318 0.4655 0.3407
0.1907
-1.1761 -0.9743 -0.7287 -0.4894 -0.2383 0.0467
0.2745

Columns 442 through 448

-0.4508 -0.5420 -0.6016 -0.6340 -0.6326 -0.5965 -
0.5267
0.0136 -0.2012 -0.3828 -0.5489 -0.7347 -0.8971 -
1.0646
0.4784 0.6627 0.8210 1.0131 1.2623 1.4778
1.6354

Columns 449 through 455

-0.4357 -0.3579 -0.2841 -0.2531 -0.2601 -0.2681 -
0.2344
-1.2115 -1.3743 -1.5645 -1.7329 -1.8260 -1.9232 -
2.0078
1.7755 1.8840 1.9636 2.0176 2.0654 2.0872
2.1041

Columns 456 through 462

-0.1952 -0.1317 -0.0800 -0.0114 0.1137 0.2125
0.2591
-2.0265 -2.0288 -1.9994 -1.9605 -1.9431 -1.9252 -
1.8690
2.1202 2.1098 2.0911 2.0533 1.9530 1.7960
1.6366

Columns 463 through 469

0.3095 0.3479 0.4055 0.5050 0.6190 0.6728
0.7152
-1.7963 -1.7073 -1.6020 -1.4600 -1.2732 -1.0915 -
0.8986
1.4506 1.2815 1.0945 0.8789 0.6517 0.4667
0.2646

Columns 470 through 476

0.7574 0.7807 0.8072 0.8249 0.8164 0.7821
0.7174
-0.6957 -0.4966 -0.3101 -0.1397 0.0597 0.2832
0.5291
0.0286 -0.2371 -0.5112 -0.7397 -0.9505 -1.1321 -
1.2739

Columns 477 through 483

0.6275 0.5249 0.3879 0.1928 -0.0138 -0.2089 -
0.3971
0.7840 1.0210 1.2612 1.5120 1.7319 1.9177
2.0830
-1.3903 -1.4893 -1.5768 -1.6566 -1.7148 -1.7436 -
1.7423

Columns 484 through 490

-0.5589 -0.7363 -0.8835 -1.0068 -1.1152 -1.2309 -
1.3613
2.2190 2.3579 2.4590 2.5173 2.5402 2.5297
2.4843
-1.7150 -1.6428 -1.5577 -1.4675 -1.3703 -1.2544 -
1.1155

Columns 491 through 497

-1.4693 -1.5765 -1.6563 -1.7434 -1.7282 -1.7054 -
1.6815
2.4146 2.3251 2.2095 2.0217 1.8662 1.6860
1.4832
-0.9700 -0.7921 -0.5966 -0.2924 -0.1187 0.0509
0.2365

Columns 498 through 504

-1.6502 -1.6004 -1.5091 -1.3926 -1.2270 -1.0471 -
0.8676
1.1962 0.9360 0.6640 0.3816 0.0664 -0.2223 -
0.4801
0.4792 0.6569 0.8183 0.9768 1.1375 1.2732
1.3693

Columns 505 through 511

-0.6826 -0.4521 -0.2143 0.0081 0.2420 0.4993
0.7885

-0.7226 -0.9824 -1.2181 -1.4055 -1.5771 -1.7362 -1.8784
1.4310 1.4501 1.4280 1.3795 1.3106 1.2162 1.0840

Columns 512 through 518

1.0529 1.3041 1.5671 1.7811 1.9619 2.1393 2.2878
-1.9840 -2.0621 -2.1181 -2.1385 -2.1257 -2.0695 -1.9776
0.9406 0.7760 0.5684 0.3639 0.1571 -0.0861 -0.3263

Columns 519 through 525

2.3783 2.4346 2.4874 2.4813 2.4456 2.3815 2.2773
-1.8693 -1.7526 -1.5413 -1.3829 -1.2114 -0.9998 -0.7682
-0.5166 -0.6804 -0.9357 -1.0859 -1.2240 -1.3832 -1.5180

Columns 526 through 532

2.0658 1.9138 1.7507 1.4986 1.2932 1.0923 0.8124
-0.3495 -0.1676 0.0091 0.3016 0.4768 0.6443 0.8532
-1.7221 -1.7494 -1.7625 -1.7883 -1.7612 -1.7306 -1.6636

Columns 533 through 539

0.5703 0.2764 0.0144 -0.2495 -0.5585 -0.7566 -0.9450
1.0054 1.1655 1.2887 1.3841 1.4689 1.4892 1.4918
-1.5807 -1.4500 -1.3092 -1.1371 -0.9067 -0.7264 -0.5403

Columns 540 through 546

-1.1576 -1.3703 -1.6090 -1.7416 -1.8581 -1.9683 -2.0480
1.4745 1.4243 1.3223 1.2304 1.1218 0.9762 0.7977
-0.3132 -0.0550 0.2821 0.5058 0.7310 0.9910 1.2528

Columns 547 through 553

-2.1046 -2.1101 -2.0897 -2.0394 -1.9116 -1.8238 -1.7175
0.5308 0.3521 0.1697 -0.0522 -0.3602 -0.5196 -0.6703
1.5773 1.7606 1.9213 2.0893 2.2687 2.3410 2.3858

Columns 554 through 560

-1.5835 -1.3828 -1.1802 -0.9425 -0.7009 -0.4535 -0.2257
-0.8349 -1.0387 -1.2175 -1.3856 -1.5254 -1.6303 -1.7023
2.4193 2.4240 2.3999 2.3290 2.2252 2.0817 1.9259

Columns 561 through 567

0.0566 0.2783 0.4906 0.7972 0.9278 1.0472 1.2793
-1.7648 -1.7815 -1.7669 -1.7143 -1.6356 -1.5478 -1.3532
1.7073 1.5037 1.2779 0.9186 0.7083 0.5009 0.0727

Columns 568 through 574

1.3437 1.4057 1.4899 1.5048 1.5011 1.4770 1.4156
-1.2223 -1.0984 -0.8263 -0.6318 -0.4184 -0.1949 0.0762
-0.1225 -0.3084 -0.6626 -0.8718 -1.0815 -1.2809 -1.4913

Columns 575 through 581

1.3135 1.2011 1.0704 0.8862 0.7144 0.5315 0.2423
0.3796 0.6350 0.8821 1.1723 1.4043 1.6181 1.8894
-1.6936 -1.8370 -1.9533 -2.0586 -2.1181 -2.1487 -2.1308

Columns 582 through 588

0.0845 -0.0688 -0.3034 -0.4811 -0.6542 -0.8798 -1.0860
2.0166 2.1187 2.2497 2.3182 2.3711 2.4006 2.3898
-2.1009 -2.0499 -1.9470 -1.8378 -1.7173 -1.5209 -1.3033

Columns 589 through 595

-1.2363 -1.3416 -1.4968 -1.5925 -1.6559 -1.7373 -1.7211
2.3294 2.2526 2.1175 1.9786 1.8194 1.5275 1.3400
-1.0925 -0.9105 -0.6206 -0.3865 -0.1641 0.2094 0.3810

Columns 596 through 602

-1.6866 -1.6211 -1.5364 -1.4539 -1.3187 -1.1524 -0.9677
1.1413 0.7662 0.5453 0.3451 0.0587 -0.2221 -0.4934
0.5454 0.8553 0.9913 1.1089 1.2597 1.3742 1.4609

Columns 603 through 609

-0.7565 -0.4400 -0.2566 -0.0707 0.3062 0.5195 0.7269
-0.7574 -1.1006 -1.2615 -1.4170 -1.6854 -1.8165 -1.9414
1.5137 1.5407 1.5183 1.4880 1.3793 1.2969 1.2144

Columns 610 through 616

0.9984 1.2554 1.5131 1.7153 1.8645 2.0575 2.1940
-2.0705 -2.1606 -2.2062 -2.2119 -2.1972 -2.1381 -2.0540
1.0719 0.9050 0.6931 0.4967 0.3329 0.0807 -0.1400

Columns 617 through 623

2.2747 2.3399 2.3737 2.3662 2.3261 2.2477 2.1294
-1.9609 -1.8154 -1.6238 -1.4154 -1.2313 -0.9847 -0.7146
-0.3139 -0.5246 -0.7500 -0.9509 -1.0947 -1.2629 -1.4146

Columns 624 through 630

1.9858 1.8294 1.5769 1.3840 1.1772 0.9696 0.7053
-0.4674 -0.2423 0.0879 0.2872 0.4725 0.6461 0.8466
-1.5184 -1.5871 -1.6648 -1.6713 -1.6498 -1.6156 -1.5518

Columns 631 through 637

0.4681 0.1359 -0.1737 -0.4290 -0.6432 -0.9074 -1.1348
1.0089 1.2052 1.3565 1.4413 1.4958 1.5355 1.5474
-1.4769 -1.3410 -1.1828 -1.0124 -0.8527 -0.6282 -0.4127

Columns 638 through 644

-1.3910 -1.6001 -1.7866 -1.9591 -2.0680 -2.1520 -2.2140
1.5252 1.4775 1.3920 1.2658 1.1459 1.0170 0.8594
-0.1342 0.1226 0.3946 0.6933 0.9221 1.1350 1.3546

Columns 645 through 651

-2.2448 -2.2405 -2.1931 -2.1176 -2.0071 -1.8354 -1.7005
0.6598 0.4457 0.2202 0.0131 -0.1964 -0.4546 -0.6046
1.5850 1.7948 1.9730 2.1045 2.2036 2.2900 2.3050

Columns 652 through 658

-1.5528 -1.3811 -1.1641 -0.9287 -0.6336 -0.4088 -0.1850
-0.7476 -0.8959 -1.0578 -1.2114 -1.3733 -1.4609 -1.5184
2.3003 2.2770 2.2218 2.1402 2.0070 1.8697 1.7035

Columns 659 through 665

0.0411 0.2988 0.5677 0.7517 0.9220 1.0892 1.2421
-1.5596 -1.5780 -1.5733 -1.5350 -1.4788 -1.3971 -1.2829
1.5185 1.2792 1.0056 0.7833 0.5568 0.3079 0.0408

Columns 666 through 672

1.3993 1.4822 1.5412 1.5830 1.5909 1.5677 1.5060
-1.1041 -0.9496 -0.7777 -0.5718 -0.3047 -0.0459 0.2287
-0.2952 -0.5325 -0.7636 -1.0112 -1.2862 -1.5219 -1.7347

Columns 673 through 679

1.4075 1.2993 1.1758 1.0043 0.8371 0.6565 0.3577
0.5221 0.7608 0.9867 1.2419 1.4538 1.6505 1.9094
-1.9296 -2.0600 -2.1625 -2.2462 -2.2909 -2.3071 -2.2671

Columns 680 through 686

0.2207 0.0839 -0.1910 -0.3437 -0.4894 -0.7321 -0.9075
1.9992 2.0704 2.1952 2.2268 2.2511 2.2660 2.2341
-2.2199 -2.1544 -2.0042 -1.8831 -1.7616 -1.5339 -1.3266

Columns 687 through 693

-1.0523 -1.2516 -1.3484 -1.4207 -1.5107 -1.5584 -1.5841
2.1685 2.0569 1.9422 1.8148 1.6308 1.4197 1.1879
-1.1162 -0.8053 -0.5937 -0.3941 -0.1201 0.1387 0.3962

Columns 694 through 700

-1.5755 -1.5360 -1.4616 -1.3556 -1.2196 -1.0616 -0.8405
0.9243 0.6521 0.3908 0.1095 -0.1651 -0.4382 -0.7605
0.6511 0.8838 1.0708 1.2461 1.3847 1.4998 1.6009

Columns 701 through 707

-0.6496 -0.4390 -0.1966 0.1018 0.3887 0.6166 0.8284
-1.0008 -1.2297 -1.4644 -1.7089 -1.9123 -2.0493 -2.1598
1.6503 1.6687 1.6611 1.6071 1.5236 1.4327 1.3314

Columns 708 through 714

1.0658 1.2988 1.5687 1.7063 1.8227 1.9519 2.0720
-2.2546 -2.3194 -2.3514 -2.3431 -2.3098 -2.2453 -2.1308
1.1888 1.0206 0.7827 0.6368 0.4872 0.2934 0.0588

Columns 715 through 721

-1.0086 -0.8758 -0.5650 -0.3760 -0.1806 0.0906
2.1705 2.2281 2.2339 2.2049 2.1485 2.0591 0.3668
1.9445 -0.5220 -0.7212 -1.1169 -1.3157 -1.5109 -1.7459 -
-1.9831 -1.7944 -1.6196 -1.4181 -1.1614 -0.9084 - 1.9484
0.6560
-0.1874 -0.4336 -0.6143 -0.7868 -0.9871 -1.1507 - Columns 771 through 777
1.2885

1.4776 1.3528 1.2100 1.0110 0.8568 0.6960
Columns 722 through 728 0.5128
0.6355 0.8789 1.1049 1.3646 1.5350 1.6855
1.7638 1.6091 1.4318 1.2283 0.9827 0.7039 1.8300
0.4791 -2.1131 -2.2317 -2.3150 -2.3755 -2.3918 -2.3814 -
-0.3263 -0.1172 0.0953 0.3252 0.5756 0.8302 2.3429
1.0022
-1.4374 -1.4919 -1.5272 -1.5535 -1.5582 -1.5341 - Columns 778 through 784
1.4813

0.2886 0.0639 -0.1654 -0.3822 -0.5999 -0.8199 -
Columns 729 through 735 1.0039
1.9712 2.0814 2.1605 2.2040 2.2121 2.1868
0.2503 0.0017 -0.2714 -0.5762 -0.8532 -1.1155 - 2.1244
1.3758 -2.2598 -2.1453 -1.9950 -1.8217 -1.6122 -1.3669 -
1.1554 1.3012 1.4366 1.5623 1.6443 1.6903 1.1205
1.6976
-1.4057 -1.3030 -1.1652 -0.9861 -0.7911 -0.5748 - Columns 785 through 791
0.3217

-1.1418 -1.2934 -1.4164 -1.4929 -1.5379 -1.5744 -
Columns 736 through 742 1.5574
2.0385 1.9090 1.7454 1.5624 1.3695 1.0656
-1.6054 -1.7945 -1.9928 -2.1657 -2.2676 -2.3257 - 0.8488
2.3697 -0.8967 -0.6156 -0.3289 -0.0695 0.1684 0.5087
1.6726 1.6279 1.5431 1.4121 1.2775 1.1484 0.7086
0.9431
-0.0672 0.1666 0.4498 0.7537 0.9901 1.1773 Columns 792 through 798
1.4267

-1.5170 -1.4547 -1.3349 -1.2085 -1.0447 -0.8085 -
Columns 743 through 749 0.6344
0.6237 0.3679 0.0446 -0.2271 -0.5036 -0.8447 -
-2.3745 -2.3502 -2.2821 -2.1864 -2.0562 -1.8111 - 1.0545
1.6609 0.8933 1.0868 1.2904 1.4356 1.5482 1.6532
0.7367 0.5625 0.3348 0.1326 -0.0759 -0.4181 - 1.6889
0.5652
1.6378 1.7877 1.9473 2.0538 2.1321 2.2292 Columns 799 through 805
2.2261

-0.4578 -0.1511 0.0459 0.2414 0.5044 0.7488
Columns 750 through 756 0.9789
-1.2535 -1.5522 -1.7227 -1.8780 -2.0526 -2.1835 -
-1.5051 -1.1625 -0.9858 -0.8072 -0.4474 -0.2404 - 2.2806
0.0400 1.7113 1.7033 1.6768 1.6366 1.5482 1.4347
-0.6985 -0.9921 -1.0879 -1.1844 -1.3707 -1.4361 - 1.3017
1.4801
2.2036 2.1546 2.0736 1.9916 1.8181 1.6765 Columns 806 through 812
1.5201

1.2434 1.4159 1.5673 1.7099 1.8568 1.9967
Columns 757 through 763 2.0682
-2.3550 -2.3798 -2.3758 -2.3480 -2.2783 -2.1657 -
0.3246 0.4944 0.6581 0.9680 1.1069 1.2361 2.0479
1.4541 1.1116 0.9639 0.8085 0.6381 0.4214 0.1689 -
-1.5515 -1.5405 -1.5280 -1.4761 -1.4121 -1.3329 - 0.0203
1.1398
1.2269 1.0461 0.8699 0.5082 0.3052 0.0968 - Columns 813 through 819
0.3142

2.1149 2.1407 2.1373 2.0959 2.0123 1.9082
Columns 764 through 770 1.7625
-1.9069 -1.7250 -1.4943 -1.1961 -0.9353 -0.7122 -
1.5306 1.5970 1.6819 1.6918 1.6915 1.6553 0.4385
1.5816 -0.2080 -0.4156 -0.6429 -0.8997 -1.0770 -1.1960 -
1.3240

Columns 820 through 826

1.5846 1.3641 1.1320 0.9164 0.6953 0.4368 0.1519
-0.1592 0.1409 0.4133 0.6275 0.8254 1.0327 1.2312
-1.4254 -1.5050 -1.5453 -1.5439 -1.5208 -1.4695 -1.3831

Columns 827 through 833

-0.1820 -0.4172 -0.6515 -0.8817 -1.1390 -1.3787 -1.7266
1.4279 1.5385 1.6258 1.6964 1.7474 1.7766 1.7498
-1.2459 -1.1213 -0.9744 -0.8147 -0.6084 -0.3980 -0.0231

Columns 834 through 840

-1.8848 -2.0167 -2.1491 -2.2975 -2.3837 -2.4141 -2.4170
1.7160 1.6503 1.5706 1.3760 1.2372 1.0618 0.8739
0.1688 0.3664 0.5785 0.9215 1.1466 1.3523 1.5431

Columns 841 through 847

-2.3735 -2.3129 -2.1837 -2.0700 -1.9221 -1.7434 -1.4954
0.6859 0.4980 0.2135 0.0279 -0.1618 -0.3629 -0.6044
1.6876 1.8149 1.9702 2.0421 2.0838 2.1063 2.0998

Columns 848 through 854

-1.2711 -1.0284 -0.6581 -0.4611 -0.2698 0.0598 0.2710
-0.7915 -0.9635 -1.1966 -1.2712 -1.3297 -1.4322 -1.4630
2.0626 1.9918 1.8547 1.7324 1.5996 1.3723 1.1920

Columns 855 through 861

0.4761 0.7259 0.9295 1.1321 1.3481 1.4595 1.5583
-1.4895 -1.4927 -1.4647 -1.3898 -1.2654 -1.1570 -1.0453
1.0134 0.7668 0.5351 0.2577 -0.0827 -0.3025 -0.5130

Columns 862 through 868

1.6799 1.7396 1.7732 1.7696 1.7411 1.6913 1.6217
-0.8314 -0.6465 -0.4374 -0.1353 0.0768 0.2813 0.4909
-0.8484 -1.0930 -1.3357 -1.6343 -1.8179 -1.9726 -2.1127

Columns 869 through 875

1.5017 1.3563 1.1873 1.0347 0.8424 0.6199 0.4280
0.7411 0.9910 1.2173 1.3902 1.5649 1.7340 1.8446
-2.2428 -2.3474 -2.4046 -2.4250 -2.4073 -2.3539 -2.2726

Columns 876 through 882

0.2350 -0.0132 -0.2928 -0.5073 -0.6655 -0.8529 -1.0200
1.9364 2.0275 2.0929 2.1008 2.0740 2.0240 1.9438
-2.1714 -2.0142 -1.8001 -1.5934 -1.4085 -1.1711 -0.9238

Columns 883 through 889

-1.2399 -1.3236 -1.3905 -1.4690 -1.5240 -1.5424 -1.5143
1.7947 1.6675 1.5217 1.3334 1.0767 0.8333 0.5875
-0.5548 -0.3439 -0.1312 0.1356 0.4472 0.7092 0.9269

Columns 890 through 896

-1.4587 -1.3146 -1.2101 -1.0880 -0.8601 -0.6991 -0.5380
0.3362 -0.0751 -0.2948 -0.5161 -0.8746 -1.0827 -1.2815
1.1225 1.3897 1.5048 1.6041 1.7347 1.7818 1.8195

Columns 897 through 903

-0.2887 -0.0303 0.1997 0.4160 0.7050 0.8945 1.0734
-1.5443 -1.7766 -1.9573 -2.1008 -2.2539 -2.3323 -2.3867
1.8330 1.8069 1.7575 1.6847 1.5489 1.4378 1.3132

Columns 904 through 910

1.2479 1.4327 1.6198 1.7542 1.8660 2.0088 2.0345
-2.4183 -2.4201 -2.3853 -2.3179 -2.2201 -2.0270 -1.8899
1.1705 0.9874 0.7654 0.5637 0.3541 0.0182 -0.1446

Columns 911 through 917

2.0454 2.0710 2.0146 1.9511 1.8118 1.6830 1.5503
-1.7404 -1.3620 -1.1730 -0.9877 -0.5949 -0.3821 -0.1717
-0.3050 -0.7091 -0.8416 -0.9634 -1.2169 -1.3008 -1.3785

Columns 918 through 924

1.3551 1.1331 0.9158 0.6885 0.2662 0.0655 -0.1328
0.1157 0.3940 0.6302 0.8467 1.1926 1.3215 1.4373

-1.4708 -1.5271 -1.5460 -1.5352 -1.4588 -1.3870 -1.3044

Columns 925 through 931

-0.4295 -0.7221 -0.9730 -1.2707 -1.5088 -1.7253 -1.9729
1.5924 1.7066 1.7913 1.8456 1.8674 1.8468 1.7816
-1.1629 -0.9846 -0.8183 -0.5748 -0.3586 -0.1216 0.1913

Columns 932 through 938

-2.0948 -2.1966 -2.3246 -2.3891 -2.4297 -2.4343 -2.4017
1.7192 1.6536 1.5066 1.3747 1.2232 0.9832 0.8076
0.3755 0.5430 0.8180 1.0144 1.2066 1.4511 1.5941

Columns 939 through 945

-2.3353 -2.2470 -2.0906 -1.9242 -1.7192 -1.5113 -1.2571
0.6267 0.4258 0.1519 -0.0898 -0.3284 -0.5383 -0.7529
1.7086 1.8212 1.9387 2.0140 2.0476 2.0496 2.0100

Columns 946 through 952

-0.9997 -0.7281 -0.4650 -0.1538 0.0894 0.3326 0.6037
-0.9417 -1.1053 -1.2397 -1.3680 -1.4419 -1.4901 -1.5223
1.9414 1.8335 1.7047 1.5218 1.3525 1.1575 0.9186

Columns 953 through 959

0.8308 1.0367 1.3086 1.4350 1.5505 1.7044 1.7766
-1.5152 -1.4796 -1.3785 -1.2902 -1.1843 -0.9852 -0.8337
0.6844 0.4429 0.0699 -0.1448 -0.3662 -0.7193 -0.9429

Columns 960 through 966

1.8348 1.8743 1.8698 1.8391 1.7741 1.6691 1.5654
-0.6806 -0.4226 -0.1872 0.0272 0.2675 0.5335 0.7377
-1.1542 -1.4517 -1.6826 -1.8663 -2.0416 -2.2026 -2.3031

Columns 967 through 973

1.4432 1.2628 1.0993 0.9193 0.6741 0.4810 0.2852
0.9329 1.1625 1.3333 1.4889 1.6692 1.7765 1.8606
-2.3761 -2.4253 -2.4326 -2.4082 -2.3433 -2.2576 -2.1458

Columns 974 through 980

0.0626 -0.1944 -0.4662 -0.6659 -0.8158 -1.0008 -1.1592
1.9322 1.9856 2.0084 1.9813 1.9263 1.8344 1.7074
-1.9949 -1.7912 -1.5422 -1.3154 -1.1106 -0.8336 -0.5483

Columns 981 through 987

-1.2765 -1.3734 -1.4601 -1.4938 -1.5004 -1.4819 -1.4134
1.5623 1.3984 1.1725 0.9787 0.7669 0.5152 0.1884
-0.2858 -0.0250 0.2875 0.5151 0.7335 0.9667 1.2250

Columns 988 through 994

-1.3362 -1.2311 -1.1012 -0.9121 -0.7094 -0.4816 -0.2720
-0.0510 -0.2964 -0.5573 -0.8637 -1.1510 -1.4213 -1.6407
1.3872 1.5275 1.6584 1.7759 1.8604 1.9029 1.9127

Columns 995 through 1001

-0.0500 0.2247 0.4176 0.6018 0.7963 1.0001 1.2237
-1.8377 -2.0437 -2.1680 -2.2719 -2.3592 -2.4194 -2.4485
1.8877 1.8190 1.7504 1.6701 1.5629 1.4193 1.2248

Columns 1002 through 1008

1.3729 1.5089 1.6424 1.7696 1.8678 1.9326 1.9582
-2.4332 -2.3917 -2.3215 -2.2096 -2.0616 -1.8776 -1.6805
1.0602 0.8829 0.6790 0.4400 0.1939 -0.0550 -0.2777

Columns 1009 through 1015

1.9523 1.9133 1.8401 1.7400 1.6122 1.4262 1.2526
-1.4618 -1.1506 -0.9176 -0.6812 -0.4146 -0.0880 0.1714
-0.4905 -0.7627 -0.9226 -1.0588 -1.1977 -1.3382 -1.4240

Columns 1016 through 1022

1.0558 0.7406 0.5503 0.3613 -0.0573 -0.2665 -0.4731
0.4223 0.7771 0.9456 1.1038 1.4108 1.5312 1.6467
-1.4782 -1.5177 -1.4959 -1.4652 -1.3535 -1.2647 -1.1736

Columns 1023 through 1029

-0.8438 -1.0559 -1.2581 -1.6439 -1.7847 -1.9101 -2.1205

1.8056 1.8741 1.9236 1.9521 1.9380 1.9260
1.8400
-0.9619 -0.8182 -0.6655 -0.3082 -0.1533 -0.0158
0.2805

Columns 1030 through 1036

-2.2235 -2.3119 -2.3796 -2.4215 -2.4274 -2.3928 -
2.3260
1.7669 1.6797 1.5622 1.4076 1.1809 0.9954
0.7972
0.4566 0.6322 0.8174 1.0138 1.2465 1.3974
1.5287

Columns 1037 through 1043

-2.2016 -2.0740 -1.9269 -1.7374 -1.5068 -1.2314 -
0.9896
0.4990 0.2957 0.0903 -0.1440 -0.3891 -0.6399 -
0.8195
1.7026 1.7783 1.8367 1.8814 1.8959 1.8713
1.8091

Columns 1044 through 1050

-0.7736 -0.4735 -0.2075 0.0418 0.4111 0.6095
0.8015
-0.9539 -1.1288 -1.2569 -1.3544 -1.4656 -1.4919 -
1.4975
1.7275 1.6023 1.4644 1.3126 1.0546 0.8824
0.6960

Columns 1051 through 1057

1.1610 1.3054 1.4410 1.6553 1.7623 1.8609
1.9495
-1.4696 -1.4168 -1.3667 -1.2275 -1.1151 -0.9842 -
0.8074
0.3086 0.1114 -0.0743 -0.4278 -0.6472 -0.8767 -
1.1421

Columns 1058 through 1064

2.0076 2.0227 2.0000 1.9509 1.8513 1.7483
1.6257
-0.5885 -0.3447 -0.1274 0.0746 0.3479 0.5558
0.7536
-1.4191 -1.6779 -1.8725 -2.0255 -2.1993 -2.3042 -
2.3794

Columns 1065 through 1071

1.3919 1.2578 1.1123 0.8158 0.6467 0.4838
0.2535
1.0466 1.1689 1.2808 1.5036 1.5874 1.6615
1.7490
-2.4384 -2.4267 -2.3931 -2.3195 -2.2341 -2.1453 -
2.0025

Columns 1072 through 1078

0.0223 -0.2385 -0.4850 -0.6829 -0.8512 -1.0597 -
1.1843
1.8087 1.8434 1.8403 1.8001 1.7347 1.6240
1.5112
-1.8310 -1.6049 -1.3553 -1.1172 -0.8835 -0.5643 -
0.3269

Columns 1079 through 1085

-1.2940 -1.3920 -1.4758 -1.5220 -1.5260 -1.4948 -
1.4282
1.3777 1.2242 1.0167 0.7879 0.5334 0.2781
0.0023
-0.0837 0.1678 0.4590 0.7341 0.9926 1.2167
1.4259

Columns 1086 through 1092

-1.3172 -1.1671 -1.0200 -0.8732 -0.6405 -0.4288 -
0.2601
-0.3188 -0.6445 -0.9043 -1.1273 -1.4213 -1.6526 -
1.8168
1.6360 1.8117 1.9243 2.0005 2.0619 2.0814
2.0769

Columns 1093 through 1099

-0.0168 0.1982 0.3941 0.6648 0.8251 0.9752
1.1186
-2.0158 -2.1635 -2.2701 -2.3769 -2.4112 -2.4199 -
2.4089
2.0326 1.9652 1.8760 1.7121 1.5861 1.4447
1.2903

Columns 1100 through 1106

1.2778 1.4153 1.6328 1.6841 1.7168 1.7670
1.7709
-2.3675 -2.2992 -2.1293 -2.0066 -1.8609 -1.6514 -
1.4064
1.0896 0.8839 0.4965 0.3225 0.1440 -0.1156 -
0.3645

Columns 1107 through 1113

1.7525 1.6914 1.5975 1.4853 1.3653 1.2195
1.0403
-1.1978 -0.9309 -0.6260 -0.3810 -0.1456 0.1074
0.3746
-0.5547 -0.7606 -0.9715 -1.1043 -1.2197 -1.3269 -
1.4149

Columns 1114 through 1120

0.8146 0.6134 0.3922 0.1391 -0.1503 -0.4275 -
0.7064
0.6624 0.8811 1.0971 1.3175 1.5372 1.7174
1.8673
-1.4771 -1.4945 -1.4893 -1.4566 -1.3870 -1.2899 -
1.1608

Columns 1121 through 1127

-0.9968 -1.2678 -1.4870 -1.7001 -1.9218 -2.0747 -
2.1974
1.9900 2.0741 2.1220 2.1431 2.1265 2.0863
2.0171
-0.9932 -0.8063 -0.6350 -0.4430 -0.2047 -0.0116
0.1803

Columns 1128 through 1134

-2.3163 -2.3801 -2.4185 -2.4349 -2.4032 -2.3333 -2.1903
1.8990 1.7772 1.6362 1.3995 1.2086 0.9980 0.6424
0.4172 0.6029 0.7823 1.0354 1.1946 1.3353 1.5479

Columns 1135 through 1141

-2.0632 -1.9279 -1.7295 -1.4936 -1.2659 -1.0208 -0.6803
0.4511 0.2645 0.0036 -0.2561 -0.4719 -0.6681 -0.9185
1.6122 1.6634 1.7259 1.7497 1.7378 1.6890 1.5988

Columns 1142 through 1148

-0.4617 -0.2485 0.1071 0.3240 0.5400 0.7967 1.0448
-1.0427 -1.1540 -1.3200 -1.3905 -1.4469 -1.4908 -1.5016
1.5044 1.4025 1.2128 1.0665 0.9069 0.6941 0.4568

Columns 1149 through 1155

1.3344 1.4974 1.6446 1.8353 1.9494 2.0476 2.1280
-1.4683 -1.4200 -1.3532 -1.2216 -1.1054 -0.9853 -0.8171
0.1340 -0.0774 -0.2913 -0.6138 -0.8440 -1.0623 -1.3108

Columns 1156 through 1162

2.1730 2.1705 2.1331 2.0757 1.9815 1.8577 1.7013
-0.6276 -0.3628 -0.1429 0.0338 0.2463 0.4556 0.6678
-1.5454 -1.8077 -1.9902 -2.1095 -2.2278 -2.3133 -2.3691

Columns 1163 through 1169

1.5087 1.3065 1.1064 0.8570 0.5731 0.3366 0.1329
0.8808 1.0618 1.2071 1.3606 1.5052 1.5889 1.6361
-2.3895 -2.3683 -2.3136 -2.2176 -2.0784 -1.9255 -1.7691

Columns 1170 through 1176

-0.0709 -0.2721 -0.4937 -0.7086 -0.9307 -1.0764 -1.2024
1.6640 1.6723 1.6592 1.6191 1.5414 1.4491 1.3358
-1.5931 -1.4003 -1.1655 -0.9105 -0.6107 -0.3727 -0.1334

Columns 1177 through 1183

-1.3168 -1.4225 -1.5069 -1.5383 -1.5391 -1.4930 -1.4279
1.2038 1.0200 0.7927 0.5848 0.3607 0.0333 -0.2133
0.1130 0.4025 0.7142 0.9535 1.1784 1.4596 1.6411

Columns 1184 through 1190

-1.3421 -1.2175 -1.0359 -0.8708 -0.7277 -0.5012 -0.3085
-0.4626 -0.7484 -1.0662 -1.3067 -1.4833 -1.7191 -1.8926
1.8047 1.9659 2.1021 2.1775 2.2110 2.2204 2.2011

Columns 1191 through 1197

-0.1421 0.1216 0.2762 0.4352 0.5993 0.7985 0.9803
-2.0208 -2.1812 -2.2461 -2.2909 -2.3201 -2.3301 -2.3092
2.1629 2.0596 1.9699 1.8557 1.7208 1.5317 1.3289

Columns 1198 through 1204

1.1963 1.2934 1.3799 1.4770 1.5749 1.6471 1.6279
-2.2363 -2.1480 -2.0351 -1.8918 -1.6772 -1.4041 -1.2137
1.0400 0.8547 0.6551 0.4148 0.1023 -0.2430 -0.4142

Columns 1205 through 1211

1.5930 1.5301 1.4478 1.3680 1.2426 1.0901 0.8504
-1.0172 -0.6719 -0.4382 -0.2157 0.0554 0.3224 0.6606
-0.5758 -0.8583 -1.0095 -1.1523 -1.2981 -1.4126 -1.5111

Columns 1212 through 1218

0.6718 0.4788 0.2476 -0.0313 -0.2898 -0.5594 -0.8459
0.8762 1.0859 1.3178 1.5578 1.7551 1.9255 2.0767
-1.5480 -1.5646 -1.5655 -1.5265 -1.4654 -1.3661 -1.2307

Columns 1219 through 1225

-1.0744 -1.2821 -1.4957 -1.7022 -1.8816 -2.0321 -2.1603
2.1743 2.2443 2.2910 2.3051 2.2839 2.2267 2.1368
-1.0999 -0.9622 -0.7953 -0.6029 -0.4023 -0.1946 0.0235

Columns 1226 through 1232

-2.2608 -2.3254 -2.3486 -2.3364 -2.2837 -2.1944 -2.0790
2.0080 1.8507 1.6706 1.4642 1.2046 0.9358 0.6873
0.2528 0.4747 0.6780 0.8722 1.0791 1.2586 1.3917

Columns 1233 through 1239

-0.5197 -0.8526 -1.0548 -1.2509 -1.4442 -1.6663 -1.9303 -1.7332 -1.5460 -1.3415 -1.1122 -0.8423 -0.5653

0.4261 0.1364 -0.0858 -0.3001 -0.5217 -0.7578 -0.9710

1.5042 1.5968 1.6317 1.6416 1.6339 1.6000 1.5363

Columns 1240 through 1246

-0.2789 -0.0166 0.2450 0.6395 0.8440 1.0359 1.3957

-1.1573 -1.2990 -1.4122 -1.5422 -1.5813 -1.6033 -1.5945

1.4363 1.3156 1.1671 0.9027 0.7374 0.5674 0.1989

Columns 1247 through 1253

1.5436 1.6959 1.9263 2.0427 2.1313 2.2201 2.2637

-1.5587 -1.5250 -1.4026 -1.3121 -1.2100 -1.0414 -0.8425

0.0151 -0.1709 -0.5237 -0.7306 -0.9213 -1.1787 -1.4212

Columns 1254 through 1260

2.2832 2.2594 2.1892 2.1064 2.0065 1.8830 1.7228

-0.6680 -0.4597 -0.2030 -0.0373 0.1268 0.2988 0.4926

-1.6152 -1.7997 -1.9862 -2.0691 -2.1332 -2.1819 -2.2153

Columns 1261 through 1267

1.5343 1.2900 1.0848 0.8620 0.6281 0.3526 0.0514

0.6894 0.9034 1.0449 1.1734 1.2906 1.4033 1.4988

-2.2237 -2.1934 -2.1297 -2.0354 -1.9188 -1.7558 -1.5502

Columns 1268 through 1274

-0.1523 -0.3500 -0.5504 -0.7610 -0.9623 -1.2182 -1.3312

1.5235 1.5304 1.5236 1.4944 1.4438 1.3219 1.2183

-1.3712 -1.1804 -0.9732 -0.7334 -0.4815 -0.1037 0.1129

Columns 1275 through 1281

-1.4205 -1.5295 -1.5890 -1.6389 -1.6469 -1.6253 -1.5574

1.0903 0.9028 0.7124 0.5223 0.2041 -0.0185 -0.2622

0.3301 0.6267 0.8766 1.1166 1.4428 1.6437 1.8196

Columns 1282 through 1288

-1.4737 -1.3124 -1.2067 -1.0717 -0.9216 -0.7009 -0.5101

-1.9934 2.1650 2.2615 2.3226 2.3658 2.3672 2.3415

Columns 1289 through 1295

-0.3043 -0.0259 0.1314 0.2831 0.5010 0.7102 0.9205

-1.9730 -2.1275 -2.1771 -2.2077 -2.2408 -2.2361 -2.2076

2.2773 2.1534 2.0457 1.9247 1.7398 1.5258 1.2871

Columns 1296 through 1302

1.1224 1.2671 1.3581 1.4684 1.5397 1.5754 1.5931

-2.1294 -2.0201 -1.8862 -1.7011 -1.4903 -1.2991 -1.0196

1.0070 0.7530 0.5282 0.2327 -0.0494 -0.2764 -0.5735

Columns 1303 through 1309

1.5693 1.5070 1.3820 1.2708 1.1580 0.9782 0.7558

-0.7804 -0.5197 -0.1288 0.1000 0.3106 0.6027 0.9004

-0.7890 -0.9873 -1.2533 -1.3707 -1.4687 -1.5809 -1.6562

Columns 1310 through 1316

0.5456 0.3129 0.0021 -0.2031 -0.4006 -0.6745 -0.9156

1.1523 1.3900 1.6680 1.8239 1.9622 2.1194 2.2307

-1.6978 -1.7029 -1.6702 -1.6207 -1.5616 -1.4449 -1.3151

Columns 1317 through 1323

-1.1365 -1.4193 -1.5794 -1.7137 -1.8247 -1.9451 -2.0337

2.3124 2.3703 2.3771 2.3494 2.3089 2.2228 2.1292

-1.1759 -0.9510 -0.7978 -0.6357 -0.4842 -0.2776 -0.0955

Columns 1324 through 1330

-2.1714 -2.1930 -2.1694 -2.1393 -2.0634 -1.9793 -1.8555

1.8889 1.7371 1.5412 1.3586 1.0671 0.8330 0.5910

0.2824 0.4559 0.6281 0.7807 0.9963 1.1463 1.2645

Columns 1331 through 1337

-1.6769 -1.5038 -1.3229 -1.1198 -0.8872 -0.6345 -0.3774

0.2765 0.0501 -0.1692 -0.3978 -0.6326 -0.8588 -1.0536

1.4004 1.4538 1.4921 1.5176 1.5198 1.4934 1.4310

Columns 1338 through 1344

-0.1055 0.1971 0.5019 0.7666 1.0187 1.3566 1.5311
-1.2302 -1.3961 -1.5348 -1.6267 -1.6872 -1.7167 -1.7092
1.3357 1.1990 1.0330 0.8601 0.6685 0.3601 0.1782

Columns 1345 through 1351

1.6964 1.9585 2.0863 2.1923 2.3071 2.3703 2.3995
-1.6917 -1.6068 -1.5401 -1.4661 -1.3156 -1.1621 -1.0118
-0.0048 -0.3516 -0.5462 -0.7261 -0.9915 -1.2081 -1.3877

Columns 1352 through 1358

2.3960 2.3588 2.2846 2.1551 2.0319 1.8890 1.7171
-0.7917 -0.5975 -0.3967 -0.1327 0.0442 0.2235 0.4117
-1.6042 -1.7613 -1.8878 -2.0224 -2.0761 -2.1125 -2.1287

Columns 1359 through 1365

1.5092 1.2515 1.0257 0.7907 0.5177 0.1930 -0.0842
0.6115 0.8294 0.9821 1.1157 1.2516 1.3808 1.4577
-2.1207 -2.0809 -2.0079 -1.9064 -1.7693 -1.5738 -1.3734

Columns 1366 through 1372

-0.3051 -0.5391 -0.7623 -1.0161 -1.1915 -1.3453 -1.4938
1.4886 1.5026 1.4905 1.4427 1.3750 1.2815 1.1511
-1.1836 -0.9635 -0.7281 -0.4265 -0.1835 0.0638 0.3427

Columns 1373 through 1379

-1.6237 -1.7141 -1.7581 -1.7807 -1.7692 -1.7302 -1.6603
0.9642 0.7543 0.5619 0.3616 0.0899 -0.1510 -0.3939
0.6595 0.9597 1.1961 1.4192 1.6793 1.8811 2.0541

Columns 1380 through 1386

-1.5242 -1.4217 -1.3080 -1.1789 -1.0051 -0.8277 -0.5668
-0.7062 -0.8921 -1.0672 -1.2394 -1.4264 -1.5883 -1.7772
2.2303 2.3138 2.3751 2.4183 2.4315 2.4160 2.3440

Columns 1387 through 1393

-0.3935 -0.2088 -0.0317 0.1972 0.3938 0.6319 0.8119
-1.8711 -1.9448 -1.9996 -2.0437 -2.0564 -2.0394 -1.9888
2.2647 2.1536 2.0312 1.8465 1.6626 1.4075 1.1769

Columns 1394 through 1400

0.9755 1.1894 1.2647 1.3260 1.4525 1.4594 1.4661
-1.8988 -1.7471 -1.6157 -1.4825 -1.1784 -1.0045 -0.8278
0.9233 0.5577 0.3510 0.1565 -0.2740 -0.4549 -0.6384

Columns 1401 through 1407

1.4506 1.4016 1.3292 1.2054 1.0724 0.9161 0.6975
-0.5470 -0.3019 -0.0569 0.2501 0.5073 0.7658 1.0744
-0.9037 -1.0997 -1.2723 -1.4555 -1.5797 -1.6819 -1.7719

Columns 1408 through 1414

0.4984 0.2915 0.0181 -0.1837 -0.3838 -0.5863 -0.8146
1.3212 1.5465 1.8008 1.9634 2.1044 2.2271 2.3376
-1.8197 -1.8380 -1.8189 -1.7796 -1.7206 -1.6408 -1.5230

Columns 1415 through 1421

-1.0553 -1.2244 -1.3789 -1.5462 -1.7212 -1.8467 -1.9122
2.4200 2.4505 2.4501 2.4178 2.3375 2.2295 2.1086
-1.3647 -1.2260 -1.0712 -0.8716 -0.6163 -0.3828 -0.1964

Columns 1422 through 1428

-1.9620 -1.9898 -1.9942 -1.9606 -1.8889 -1.7821 -1.6081
1.9525 1.7611 1.5196 1.2556 1.0042 0.7393 0.3865
0.0095 0.2287 0.4746 0.7050 0.8847 1.0428 1.2216

Columns 1429 through 1435

-1.4597 -1.3031 -1.1200 -0.8940 -0.6215 -0.4086 -0.1882
0.1618 -0.0579 -0.2987 -0.5609 -0.8393 -1.0192 -1.1873
1.2979 1.3609 1.4188 1.4550 1.4608 1.4279 1.3755

Columns 1436 through 1442

0.0607 0.3305 0.6282 0.9007 1.1671 1.4551 1.6570
-1.3570 -1.5148 -1.6615 -1.7663 -1.8401 -1.8826 -1.8908

1.2963 1.1842 1.0334 0.8656 0.6730 0.4276
0.2338

Columns 1443 through 1449

1.8389 2.0031 2.1584 2.2971 2.3732 2.4218
2.4472
-1.8764 -1.8395 -1.7643 -1.6485 -1.5375 -1.4101 -
1.2309
0.0374 -0.1636 -0.3941 -0.6487 -0.8357 -1.0117 -
1.2162

Columns 1450 through 1456

2.4289 2.3680 2.2800 2.1602 1.9930 1.7940
1.5651
-0.9771 -0.7588 -0.5701 -0.3557 -0.1063 0.1492
0.3977
-1.4518 -1.6092 -1.7098 -1.8045 -1.8867 -1.9432 -
1.9628

Columns 1457 through 1463

1.3403 1.1131 0.7706 0.5522 0.3389 0.0209 -
0.2063
0.5985 0.7677 1.0028 1.1124 1.2080 1.3417
1.4054
-1.9388 -1.8808 -1.7734 -1.6646 -1.5469 -1.3626 -
1.1991

Columns 1464 through 1470

-0.4296 -0.6603 -0.8968 -1.1540 -1.3292 -1.4828 -
1.6269
1.4522 1.4783 1.4751 1.4336 1.3703 1.2825
1.1686
-1.0225 -0.8181 -0.5782 -0.2796 -0.0411 0.2002
0.4582

Columns 1471 through 1477

-1.7598 -1.8670 -1.9342 -1.9637 -1.9527 -1.8967 -
1.8316
1.0067 0.8181 0.6045 0.3942 0.1597 -0.1191 -
0.3198
0.7531 1.0489 1.3297 1.5695 1.7930 2.0158
2.1514

Columns 1478 through 1484

-1.7522 -1.6375 -1.4768 -1.3207 -1.1418 -0.8227 -
0.6694
-0.5140 -0.7297 -0.9633 -1.1461 -1.3151 -1.5739 -
1.6558
2.2662 2.3672 2.4402 2.4668 2.4569 2.3966
2.3252

Columns 1485 through 1491

-0.5173 -0.1482 0.0208 0.1917 0.4385 0.6696
0.8556
-1.7254 -1.8893 -1.9128 -1.9322 -1.9458 -1.9184 -
1.8557
2.2427 2.0375 1.8920 1.7405 1.5073 1.2488
1.0000

Columns 1492 through 1498

1.0118 1.2525 1.3267 1.3894 1.4994 1.5105
1.5185
-1.7628 -1.5770 -1.4481 -1.3145 -1.0303 -0.8565 -
0.6839
0.7510 0.3245 0.1215 -0.0749 -0.4690 -0.6539 -
0.8345

Columns 1499 through 1505

1.4935 1.4226 1.3245 1.2115 0.9684 0.8293
0.6843
-0.3929 -0.0840 0.1899 0.4409 0.8636 1.0695
1.2655
-1.1005 -1.3386 -1.5144 -1.6523 -1.8320 -1.8988 -
1.9498

Columns 1506 through 1512

0.3705 0.2057 0.0501 -0.1899 -0.4295 -0.6440 -
0.8430
1.6157 1.7718 1.9139 2.0918 2.2361 2.3362
2.3997
-1.9862 -1.9776 -1.9641 -1.9018 -1.8066 -1.6922 -
1.5566

Columns 1513 through 1519

-1.0888 -1.2315 -1.3619 -1.4855 -1.6111 -1.7127 -
1.8097
2.4390 2.4299 2.3969 2.3433 2.2505 2.1284
1.9427
-1.3501 -1.1984 -1.0350 -0.8577 -0.6394 -0.4157 -
0.1330

Columns 1520 through 1526

-1.8475 -1.8561 -1.8455 -1.7905 -1.7118 -1.6051 -
1.4805
1.7714 1.5754 1.3434 1.0144 0.7861 0.5478
0.2920
0.0761 0.2807 0.5020 0.7761 0.9257 1.0573
1.1885

Columns 1527 through 1533

-1.2998 -1.1119 -0.8915 -0.6712 -0.4225 -0.0757
0.1352
-0.0167 -0.2981 -0.5720 -0.8102 -1.0407 -1.3217 -
1.4662
1.3165 1.4100 1.4635 1.4814 1.4632 1.3974
1.3310

Columns 1534 through 1540

0.3389 0.6826 0.9057 1.1246 1.3646 1.5865
1.8554
-1.5949 -1.7748 -1.8677 -1.9466 -2.0078 -2.0384 -
2.0226
1.2560 1.0922 0.9620 0.8221 0.6432 0.4519
0.1672

Columns 1541 through 1547

1.9943 2.1120 2.2309 2.3368 2.4076 2.4284
2.4108

-1.9935 -1.9449 -1.8684 -1.7328 -1.5780 -1.3851 -1.2098
-0.0007 -0.1671 -0.3625 -0.6041 -0.8296 -1.0432 -1.2009

Columns 1548 through 1554

2.3550 2.2445 2.1301 2.0024 1.8531 1.6535 1.4411
-0.9977 -0.6834 -0.4893 -0.3003 -0.1007 0.1357 0.3567
-1.3573 -1.5610 -1.6408 -1.7020 -1.7523 -1.7893 -1.7978

Columns 1555 through 1561

1.1353 0.9051 0.6663 0.4319 0.1368 -0.1258 -0.4232
0.6372 0.8077 0.9557 1.0924 1.2384 1.3518 1.4444
-1.7725 -1.7128 -1.6219 -1.5243 -1.3752 -1.2260 -1.0212

Columns 1562 through 1568

-0.6980 -0.9613 -1.1767 -1.3819 -1.5913 -1.7472 -1.8796
1.5010 1.5106 1.4922 1.4398 1.3531 1.2489 1.1251
-0.8030 -0.5493 -0.3156 -0.0579 0.2382 0.4983 0.7545

Columns 1569 through 1575

-2.0170 -2.0759 -2.1100 -2.1256 -2.1041 -2.0536 -1.9596
0.9008 0.7490 0.5832 0.4194 0.2001 -0.0281 -0.2673
1.1162 1.3269 1.5268 1.7062 1.9040 2.0817 2.2269

Columns 1576 through 1582

-1.8553 -1.7268 -1.4705 -1.3253 -1.1776 -0.7990 -0.6515
-0.4634 -0.6501 -0.9595 -1.0869 -1.1956 -1.4704 -1.5162
2.3187 2.3768 2.4300 2.4122 2.3732 2.2695 2.1677

Columns 1583 through 1589

-0.4982 -0.1867 0.0246 0.2228 0.4135 0.6263 0.8095
-1.5768 -1.6936 -1.7347 -1.7421 -1.7324 -1.6908 -1.6329
2.0750 1.8803 1.7101 1.5193 1.3189 1.0645 0.8234

Columns 1590 through 1596

1.0309 1.1853 1.3067 1.4212 1.4687 1.5020 1.5150
-1.5273 -1.4143 -1.2637 -1.0644 -0.8891 -0.7109 -0.4582
0.4964 0.2290 -0.0430 -0.3569 -0.5796 -0.7911 -1.0568

Columns 1597 through 1603

1.4919 1.4332 1.3364 1.2020 1.0406 0.8788 0.7065
-0.1907 0.0835 0.3720 0.6736 0.9640 1.2119 1.4371
-1.3013 -1.5167 -1.7084 -1.8756 -2.0046 -2.0907 -2.1436

Columns 1604 through 1610

0.4685 0.3028 0.1320 -0.0416 -0.2406 -0.4377 -0.6725
1.6951 1.8516 1.9947 2.1200 2.2334 2.3177 2.3787
-2.1635 -2.1544 -2.1267 -2.0784 -1.9928 -1.8800 -1.7062

Columns 1611 through 1617

-0.8328 -0.9880 -1.1452 -1.3148 -1.4541 -1.5643 -1.6331
2.3886 2.3721 2.3327 2.2509 2.1385 1.9851 1.8140
-1.5558 -1.3841 -1.1875 -0.9361 -0.6844 -0.4208 -0.1809

Columns 1618 through 1624

-1.6730 -1.7013 -1.6694 -1.6202 -1.5490 -1.4320 -1.2769
1.6119 1.3101 1.1026 0.8929 0.6279 0.3195 -0.0129
0.0612 0.3912 0.5668 0.7273 0.9212 1.1125 1.2898

Columns 1625 through 1631

-1.1371 -0.9819 -0.7969 -0.5846 -0.3331 -0.0702 0.2057
-0.2416 -0.4672 -0.7069 -0.9490 -1.2042 -1.4363 -1.6465
1.3787 1.4491 1.5038 1.5336 1.5373 1.5064 1.4407

Columns 1632 through 1638

0.4876 0.7534 1.0166 1.3347 1.5034 1.6569 1.8630
-1.8282 -1.9726 -2.0887 -2.1850 -2.2196 -2.2346 -2.2220
1.3406 1.2192 1.0721 0.8503 0.7162 0.5778 0.3590

Columns 1639 through 1645

2.0337 2.1504 2.2463 2.3224 2.3371 2.3328 2.3037
-2.1685 -2.1037 -1.9947 -1.8356 -1.6899 -1.5311 -1.3215
0.1347 -0.0467 -0.2517 -0.4868 -0.6471 -0.8017 -0.9823

Columns 1646 through 1652

2.2344 2.1105 1.9879 1.8439 1.6564 1.4263 1.1562
-1.0608 -0.7448 -0.5470 -0.3450 -0.0956 0.1690 0.4478
-1.1736 -1.3657 -1.4410 -1.4989 -1.5608 -1.5954 -1.6041

Columns 1653 through 1659

0.9458 0.7294 0.4932 0.2347 -0.0339 -0.3937 -0.6131
0.6221 0.7865 0.9513 1.1110 1.2568 1.4095 1.4748
-1.5679 -1.5159 -1.4444 -1.3456 -1.2229 -1.0158 -0.8617

Columns 1660 through 1666

-0.8275 -1.0421 -1.3130 -1.5320 -1.7461 -1.9388 -2.0472
1.5127 1.5398 1.5339 1.5111 1.4416 1.3369 1.2349
-0.6852 -0.4977 -0.2209 0.0208 0.3045 0.6019 0.8123

Columns 1667 through 1673

-2.1388 -2.2196 -2.2706 -2.2907 -2.2721 -2.2151 -2.0957
1.1266 0.9693 0.8055 0.5656 0.3917 0.2073 -0.0651
1.0121 1.2503 1.4651 1.7251 1.8805 2.0078 2.1608

Columns 1674 through 1680

-1.9811 -1.8561 -1.6938 -1.4929 -1.2733 -1.0330 -0.7579
-0.2455 -0.4152 -0.6048 -0.8007 -0.9767 -1.1386 -1.2998
2.2266 2.2713 2.2986 2.2935 2.2499 2.1716 2.0577

Columns 1681 through 1687

-0.4703 -0.2300 -0.0070 0.2945 0.4921 0.6837 0.8735
-1.4335 -1.5103 -1.5530 -1.5905 -1.5786 -1.5547 -1.5119
1.9037 1.7403 1.5600 1.2960 1.0865 0.8709 0.6384

Columns 1688 through 1694

1.0619 1.2389 1.3659 1.4689 1.5728 1.6134 1.6277
-1.4392 -1.3272 -1.1939 -1.0373 -0.7882 -0.5962 -0.3923
0.3772 0.0884 -0.1719 -0.4315 -0.7846 -1.0172 -1.2355

Columns 1695 through 1701

1.6156 1.5481 1.4767 1.3803 1.2544 1.0668 0.9271
-0.1392 0.1878 0.4275 0.6626 0.9154 1.2009 1.3878
-1.4764 -1.7360 -1.9042 -2.0430 -2.1698 -2.2677 -2.3149

Columns 1702 through 1708

0.7714 0.6082 0.3948 0.1718 -0.0569 -0.2421 -0.4431
1.5621 1.7230 1.8915 2.0365 2.1446 2.2029 2.2387
-2.3335 -2.3312 -2.2863 -2.2083 -2.0877 -1.9608 -1.7957

Columns 1709 through 1715

-0.6744 -0.8436 -0.9984 -1.1772 -1.3004 -1.4016 -1.5179
2.2523 2.2218 2.1667 2.0738 1.9562 1.8099 1.5792
-1.5779 -1.3782 -1.1683 -0.8966 -0.6558 -0.4083 -0.0612

Columns 1716 through 1722

-1.5454 -1.5532 -1.5536 -1.5175 -1.4406 -1.3505 -1.2409
1.4002 1.2127 0.9799 0.6990 0.3617 0.1345 -0.0958
0.1452 0.3405 0.5737 0.8185 1.0789 1.2161 1.3367

Columns 1723 through 1729

-1.1035 -0.9268 -0.7133 -0.5038 -0.2775 -0.0048 0.2763
-0.3469 -0.6203 -0.9101 -1.1538 -1.3852 -1.6258 -1.8395
1.4504 1.5471 1.6234 1.6576 1.6627 1.6306 1.5632

Columns 1730 through 1736

0.5184 0.7446 1.0254 1.2167 1.3989 1.5796 1.7640
-1.9987 -2.1255 -2.2438 -2.3030 -2.3395 -2.3522 -2.3273
1.4803 1.3810 1.2185 1.0862 0.9405 0.7727 0.5633

Columns 1737 through 1743

1.9183 2.0398 2.1190 2.1720 2.2172 2.1953 2.1521
-2.2672 -2.1645 -2.0398 -1.8838 -1.6315 -1.4575 -1.2729
0.3489 0.1247 -0.0791 -0.2882 -0.5857 -0.7378 -0.8792

Columns 1744 through 1750

2.0841 1.9655 1.8005 1.6510 1.4894 1.2671 1.0345
-1.0297 -0.7330 -0.4068 -0.1968 0.0079 0.2743 0.5191
-1.0544 -1.2325 -1.3937 -1.4541 -1.4973 -1.5414 -1.5536

Columns 1751 through 1757

1.6737 1.7753 2.0035 2.0474 2.0862 2.1604
0.7928 0.4826 0.2620 0.0315 -0.2080 -0.5063 -
2.1515
0.7780 -2.3640 -2.3254 -2.1862 -2.0913 -1.9963 -1.7687 -
0.7480 1.0041 1.1555 1.2900 1.4160 1.5422 1.6067
1.6345
-1.5407 -1.4867 -1.4174 -1.3215 -1.2080 -1.0359 - Columns 1807 through 1813
0.8565
-0.6998 -0.9066 -1.0724 -1.2070 -1.3601 -1.4258 -
Columns 1758 through 1764 1.4779
2.1301 2.0829 2.0021 1.8920 1.7219 1.5672
-1.0563 -1.3374 -1.5712 -1.7485 -1.9651 -2.1129 -
1.3969
2.2156 -1.4303 -1.1763 -0.9298 -0.6850 -0.3617 -0.1414
1.6927 1.7173 1.6989 1.6666 1.5782 1.4859 0.0810
1.3805
-0.6365 -0.3799 -0.1277 0.0819 0.3868 0.6270 Columns 1814 through 1820
0.8352
-1.5168 -1.5376 -1.5264 -1.4657 -1.3713 -1.2641 -
Columns 1765 through 1771 1.1109
1.2117 0.9754 0.7128 0.3996 0.1195 -0.1214 -
-2.3447 -2.3675 -2.3759 -2.3577 -2.2877 -2.2066 - 0.4171
2.0942 0.3051 0.5621 0.8136 1.0661 1.2518 1.3855
1.1127 0.9863 0.8572 0.6300 0.3850 0.1978 1.5280
0.0141
1.2320 1.3812 1.5187 1.7276 1.9027 2.0088 Columns 1821 through 1827
2.0802
-0.9371 -0.7314 -0.4856 -0.2520 -0.0461 0.2344
Columns 1772 through 1778 0.5133
-0.7006 -0.9860 -1.2759 -1.5107 -1.6975 -1.9160 -
-1.9108 -1.7448 -1.5693 -1.3587 -1.1135 -0.8562 - 2.1018
0.5896 1.6377 1.7174 1.7615 1.7627 1.7436 1.6816
-0.2540 -0.4422 -0.6170 -0.8046 -0.9921 -1.1561 - 1.5885
1.2931
2.1648 2.1871 2.1863 2.1633 2.1057 2.0123 Columns 1828 through 1834
1.8827
0.7367 0.9407 1.2397 1.4025 1.5486 1.6890
Columns 1779 through 1785 1.8432
-2.2237 -2.3135 -2.3964 -2.4198 -2.4184 -2.3973 -
-0.3155 -0.0347 0.2428 0.5023 0.7190 0.9207 2.3296
1.1726 1.4870 1.3729 1.1567 1.0173 0.8699 0.7084
-1.4089 -1.4970 -1.5563 -1.5774 -1.5644 -1.5248 - 0.4864
1.4375
1.7244 1.5317 1.3135 1.0751 0.8454 0.6040 Columns 1835 through 1841
0.2649
1.9612 2.0569 2.1210 2.1436 2.1369 2.0958
Columns 1786 through 1792 2.0221
-2.2350 -2.0924 -1.9224 -1.7280 -1.5293 -1.2318 -
1.3151 1.4375 1.5488 1.6423 1.7118 1.7463 1.0072
1.7406 0.2738 0.0355 -0.1986 -0.4156 -0.6076 -0.8640 -
-1.3469 -1.2383 -1.1052 -0.9282 -0.7199 -0.4345 - 1.0149
0.1809
0.0319 -0.1992 -0.4436 -0.7141 -0.9919 -1.3118 - Columns 1842 through 1848
1.5596
1.9160 1.7765 1.5631 1.3549 1.1567 0.9510
Columns 1793 through 1799 0.7226
-0.7707 -0.4940 -0.1449 0.1391 0.3613 0.5714
1.7036 1.6327 1.5188 1.3871 1.2511 1.0899 0.7819
0.8272 -1.1453 -1.2825 -1.4182 -1.4940 -1.5180 -1.5225 -
0.0437 0.3069 0.5911 0.8482 1.0645 1.2789 1.5045
1.5579
-1.7473 -1.9396 -2.1100 -2.2353 -2.3156 -2.3687 - Columns 1849 through 1855
2.3851
0.4770 0.1647 -0.1044 -0.3805 -0.7203 -0.9369 -
Columns 1800 through 1806 1.1448
0.9850 1.2107 1.3747 1.5111 1.6407 1.6981
0.6902 0.5502 0.1828 0.0439 -0.0900 -0.3917 - 1.7398
0.5448 -1.4620 -1.3754 -1.2702 -1.1306 -0.9203 -0.7612 -
0.5950

Columns 1856 through 1862

-1.4575 -1.6384 -1.8024 -1.9902 -2.1364 -2.2595 -2.3621

1.7638 1.7563 1.7353 1.6720 1.5818 1.4605 1.2747

-0.3063 -0.1180 0.0671 0.3183 0.5546 0.7991 1.0874

Columns 1863 through 1869

-2.3983 -2.4045 -2.3902 -2.3269 -2.2262 -2.1132 -1.9755

1.1446 1.0070 0.8384 0.5950 0.3301 0.1360 -0.0581

1.2536 1.3975 1.5519 1.7319 1.8961 1.9772 2.0336

Columns 1870 through 1876

-1.7926 -1.5662 -1.3083 -1.0784 -0.8559 -0.5360 -0.3114

-0.2765 -0.5100 -0.7414 -0.9108 -1.0489 -1.2264 -1.3146

2.0691 2.0763 2.0497 1.9891 1.9048 1.7624 1.6261

Columns 1877 through 1883

-0.0877 0.1479 0.4067 0.6583 0.9218 1.1266 1.2852

-1.3853 -1.4489 -1.4935 -1.5119 -1.4891 -1.4333 -1.3559

1.4731 1.3010 1.0868 0.8536 0.5673 0.3067 0.0707

Columns 1884 through 1890

1.4745 1.6153 1.7022 1.7758 1.8197 1.8265 1.7962

-1.2224 -1.0657 -0.9191 -0.7263 -0.5059 -0.2186 0.0407

-0.2521 -0.5497 -0.7832 -1.0495 -1.3138 -1.6078 -1.8369

Columns 1891 through 1897

1.7403 1.6515 1.5086 1.3940 1.2675 1.1195 0.9318

0.2637 0.4997 0.7777 0.9636 1.1378 1.3102 1.4889

-2.0040 -2.1512 -2.2864 -2.3575 -2.4053 -2.4296 -2.4207

Columns 1898 through 1904

0.6879 0.5155 0.3380 0.1500 -0.0676 -0.2865 -0.5447

1.6792 1.7785 1.8609 1.9308 1.9868 2.0201 2.0271

-2.3671 -2.2940 -2.1989 -2.0808 -1.9192 -1.7336 -1.4824

Columns 1905 through 1911

-0.7210 -0.8831 -1.0386 -1.1979 -1.3210 -1.4129 -1.4736

1.9924 1.9254 1.8354 1.6964 1.5393 1.3521 1.1509

-1.2715 -1.0423 -0.7967 -0.4985 -0.2183 0.0608 0.3228

Columns 1912 through 1918

-1.5055 -1.5034 -1.4641 -1.3961 -1.2355 -1.1275 -1.0081

0.9065 0.6511 0.3939 0.1320 -0.2805 -0.4845 -0.6896

0.5990 0.8523 1.0702 1.2641 1.5160 1.6120 1.6976

Columns 1919 through 1925

-0.7300 -0.5693 -0.4218 -0.1162 0.0535 0.2288 0.4883

-1.0934 -1.2891 -1.4682 -1.7648 -1.9086 -2.0418 -2.2047

1.8234 1.8584 1.8900 1.8809 1.8551 1.8130 1.7163

Columns 1926 through 1932

0.7487 0.9316 1.1075 1.2945 1.4979 1.6627 1.7752

-2.3336 -2.4040 -2.4450 -2.4582 -2.4320 -2.3683 -2.2827

1.5849 1.4723 1.3375 1.1637 0.9341 0.7056 0.5074

Columns 1933 through 1939

1.8644 1.9415 1.9798 1.9942 1.9771 1.9224 1.8314

-2.1653 -2.0010 -1.8144 -1.6038 -1.3326 -1.0939 -0.8426

0.3009 0.0595 -0.1654 -0.3904 -0.6446 -0.8284 -0.9888

Columns 1940 through 1946

1.6580 1.5205 1.3828 1.0960 0.9258 0.7532 0.4155

-0.4392 -0.2299 -0.0273 0.3706 0.5556 0.7358 1.0445

-1.2188 -1.2906 -1.3555 -1.4666 -1.4815 -1.4890 -1.4600

Columns 1947 through 1953

0.2148 0.0109 -0.4058 -0.6079 -0.8036 -1.1787 -1.3684

1.1951 1.3370 1.5813 1.6722 1.7555 1.8643 1.8976

-1.4098 -1.3478 -1.1755 -1.0642 -0.9519 -0.6856 -0.5293

Columns 1954 through 1960

-1.5513 -1.7918 -1.9984 -2.1360 -2.2478 -2.3491 -2.4217

1.9228 1.9144 1.8638 1.7997 1.7166 1.5886 1.4140

-0.3714 -0.1226 0.1346 0.3362 0.5312 0.7604
1.0077

Columns 1961 through 1967

-2.4462 -2.4329 -2.3767 -2.2886 -2.1664 -1.9912 -
1.8297
1.2449 1.0729 0.8339 0.6128 0.3855 0.1105 -
0.0841
1.2013 1.3600 1.5428 1.6758 1.7809 1.8807
1.9138

Columns 1968 through 1974

-1.6498 -1.4388 -1.1894 -0.9130 -0.6251 -0.3691 -
0.1160
-0.2769 -0.4795 -0.6891 -0.8910 -1.0647 -1.1894 -
1.2912
1.9267 1.9183 1.8785 1.8040 1.6898 1.5585
1.4072

Columns 1975 through 1981

0.2195 0.4463 0.6630 0.8816 1.1100 1.3407
1.4911
-1.4017 -1.4473 -1.4709 -1.4726 -1.4396 -1.3662 -
1.2807
1.1822 1.0010 0.8080 0.5910 0.3296 0.0255 -
0.2104

Columns 1982 through 1988

1.6229 1.7414 1.8446 1.9253 1.9649 1.9676
1.9280
-1.1775 -1.0505 -0.8856 -0.6803 -0.4678 -0.2420
0.0078
-0.4454 -0.6909 -0.9590 -1.2450 -1.4972 -1.7256 -
1.9358

Columns 1989 through 1995

1.8539 1.7589 1.6435 1.4671 1.3316 1.1778
1.0087
0.2540 0.4778 0.6865 0.9357 1.0869 1.2299
1.3668
-2.1079 -2.2367 -2.3300 -2.4028 -2.4185 -2.4077 -
2.3755

Columns 1996 through 2002

0.7942 0.5701 0.2581 0.0792 -0.1035 -0.3274 -
0.5818
1.5143 1.6409 1.7787 1.8174 1.8375 1.8546
1.8393
-2.3084 -2.2111 -2.0368 -1.8966 -1.7341 -1.5273 -
1.2575

Columns 2003 through 2009

-0.8293 -0.9578 -1.0741 -1.2377 -1.3324 -1.4136 -
1.4799
1.7892 1.7066 1.6107 1.4581 1.3141 1.1635
0.9735
-0.9599 -0.7488 -0.5366 -0.2204 0.0183 0.2501
0.5064

Columns 2010 through 2016

-1.5159 -1.5097 -1.4678 -1.3881 -1.2709 -1.1334 -
0.9878
0.7352 0.4896 0.2231 -0.0755 -0.3840 -0.6654 -
0.9127
0.7807 1.0201 1.2447 1.4636 1.6549 1.7987
1.9005

Columns 2017 through 2023

-0.8033 -0.5784 -0.3403 -0.1415 0.0543 0.2924
0.5522
-1.1800 -1.4582 -1.7098 -1.8909 -2.0418 -2.1946 -
2.3219
1.9833 2.0366 2.0501 2.0323 1.9875 1.9021
1.7698

Columns 2024 through 2030

0.7513 0.9179 1.2168 1.3267 1.4268 1.6702
1.7145
-2.3918 -2.4256 -2.4420 -2.4057 -2.3518 -2.2126 -
2.0991
1.6406 1.5076 1.2252 1.0790 0.9250 0.5424
0.3846

Columns 2031 through 2037

1.7591 1.8498 1.8612 1.8502 1.8175 1.7483
1.6366
-1.9927 -1.7639 -1.5794 -1.3745 -1.1551 -0.8727 -
0.5468
0.2336 -0.0859 -0.2818 -0.4756 -0.6624 -0.8757 -
1.0897

Columns 2038 through 2044

1.5005 1.3655 1.1942 1.0038 0.6755 0.4868
0.2854
-0.2610 -0.0323 0.2299 0.4848 0.8623 1.0409
1.2079
-1.2395 -1.3332 -1.4241 -1.4886 -1.5378 -1.5277 -
1.4933

Columns 2045 through 2051

-0.1552 -0.3466 -0.5254 -0.8573 -1.1001 -1.3328 -
1.5758
1.5321 1.6346 1.7443 1.8991 1.9862 2.0411
2.0689
-1.3769 -1.2880 -1.2189 -1.0418 -0.8861 -0.7083 -
0.4931

Columns 2052 through 2058

-1.7824 -1.9621 -2.1836 -2.2707 -2.3344 -2.3944 -
2.4294
2.0602 2.0268 1.9217 1.8459 1.7478 1.6085
1.3663
-0.2777 -0.0647 0.2619 0.4248 0.5866 0.7858
1.0631

Columns 2059 through 2065

-2.4190 -2.3586 -2.2757 -2.1161 -1.9748 -1.8120 -
1.6185

1.1365 0.9258 0.7283 0.4112 0.1992 -0.0151 -0.2438
1.2824 1.4328 1.5474 1.7049 1.7756 1.8271 1.8622

Columns 2066 through 2072

-1.3756 -1.1420 -0.8881 -0.4800 -0.2733 -0.0793 0.3073
-0.4887 -0.6867 -0.8714 -1.1423 -1.2402 -1.3207 -1.4651
1.8642 1.8287 1.7596 1.6223 1.5135 1.3999 1.1578

Columns 2073 through 2079

0.4869 0.6723 0.9391 1.1898 1.3986 1.5659 1.7287
-1.4928 -1.5253 -1.5413 -1.5199 -1.4588 -1.3766 -1.2520
1.0058 0.8530 0.6022 0.3301 0.0602 -0.1893 -0.4767

Columns 2080 through 2086

1.8811 1.9839 2.0456 2.0761 2.0649 2.0143 1.9540
-1.0824 -0.9021 -0.7384 -0.5194 -0.2622 -0.0221 0.1730
-0.7988 -1.0818 -1.3073 -1.5567 -1.8027 -1.9922 -2.1270

Columns 2087 through 2093

1.8493 1.7118 1.5477 1.3873 1.1909 0.9407 0.7439
0.4055 0.6399 0.8568 1.0305 1.2090 1.4047 1.5217
-2.2548 -2.3516 -2.4045 -2.4178 -2.3999 -2.3454 -2.2656

Columns 2094 through 2100

0.5447 0.3248 0.0761 -0.2277 -0.4131 -0.5890 -0.7661
1.6165 1.7030 1.7766 1.8380 1.8336 1.8068 1.7583
-2.1612 -2.0278 -1.8527 -1.6103 -1.4205 -1.2178 -0.9923

Columns 2101 through 2107

-0.9468 -1.1159 -1.2743 -1.4004 -1.4753 -1.5171 -1.5389
1.6785 1.5752 1.4295 1.2550 1.0696 0.8823 0.5205
-0.7316 -0.4592 -0.1552 0.1454 0.4057 0.6347 1.0184

Columns 2108 through 2114

-1.5146 -1.4653 -1.4028 -1.2634 -1.1151 -0.9357 -0.7768
0.3055 0.0851 -0.1472 -0.4863 -0.7907 -1.0725 -1.2974
1.2091 1.3802 1.5500 1.7497 1.9058 2.0082 2.0742

Columns 2115 through 2121

-0.5589 -0.3540 -0.1323 0.1153 0.3059 0.4868 0.6960
-1.5470 -1.7558 -1.9439 -2.1269 -2.2393 -2.3239 -2.3920
2.1058 2.1098 2.0763 2.0116 1.9335 1.8372 1.6960

Columns 2122 through 2128

0.9145 1.1388 1.2997 1.4138 1.5567 1.6750 1.7444
-2.4278 -2.4261 -2.3831 -2.3166 -2.2007 -2.0495 -1.8873
1.5133 1.2873 1.0834 0.9028 0.6440 0.3746 0.1428

Columns 2129 through 2135

1.7857 1.8145 1.7824 1.7231 1.6505 1.5292 1.3748
-1.7080 -1.4128 -1.2021 -0.9790 -0.7402 -0.4260 -0.0985
-0.0777 -0.4017 -0.5803 -0.7441 -0.9103 -1.1032 -1.2763

Columns 2136 through 2142

1.2356 1.0840 0.8836 0.6582 0.2981 0.1010 -0.0998
0.1224 0.3392 0.5935 0.8439 1.1918 1.3511 1.4955
-1.3581 -1.4232 -1.4771 -1.5021 -1.4900 -1.4521 -1.3957

Columns 2143 through 2149

-0.5244 -0.7236 -0.9110 -1.2433 -1.4354 -1.6102 -1.7766
1.7537 1.8455 1.9396 2.0495 2.0937 2.1086 2.1075
-1.2293 -1.1219 -1.0287 -0.8062 -0.6583 -0.4983 -0.3310

Columns 2150 through 2156

-1.9650 -2.1109 -2.2476 -2.3386 -2.3828 -2.3874 -2.3646
2.0660 2.0111 1.9017 1.7701 1.5926 1.4315 1.2376
-0.1010 0.0998 0.3458 0.5686 0.7902 0.9559 1.1270

Columns 2157 through 2163

-2.3136 -2.2163 -2.1050 -1.9609 -1.7700 -1.5126 -1.2756
1.0063 0.7422 0.5325 0.3084 0.0427 -0.2662 -0.5016
1.3073 1.4741 1.5725 1.6525 1.7273 1.7788 1.7772

Columns 2164 through 2170

-1.0656 -0.8101 -0.5655 -0.2338 0.0168 0.2634 0.5380
-0.6714 -0.8617 -1.0219 -1.2158 -1.3381 -1.4319 -1.5110
1.7371 1.6718 1.5875 1.4496 1.3213 1.1685 0.9731

Columns 2171 through 2177

0.8416 1.0890 1.2795 1.4963 1.6811 1.8481 1.9855
-1.5546 -1.5576 -1.5321 -1.4771 -1.3938 -1.2772 -1.1283
0.7130 0.4687 0.2526 -0.0192 -0.2872 -0.5709 -0.8572

Columns 2178 through 2184

2.0840 2.1440 2.1685 2.1540 2.1175 2.0552 1.9508
-0.9530 -0.7690 -0.5773 -0.3161 -0.1269 0.0620 0.2895
-1.1311 -1.3750 -1.5911 -1.8379 -1.9905 -2.1172 -2.2403

Columns 2185 through 2191

1.7771 1.6476 1.4992 1.3385 1.1219 0.9183 0.6034
0.5693 0.7302 0.8820 1.0294 1.1985 1.3319 1.5101
-2.3464 -2.3777 -2.3812 -2.3680 -2.3204 -2.2501 -2.1135

Columns 2192 through 2198

0.3844 0.1593 -0.0678 -0.3429 -0.5605 -0.7630 -1.0125
1.6000 1.6620 1.7080 1.7306 1.7164 1.6668 1.5772
-1.9844 -1.8213 -1.6402 -1.3877 -1.1559 -0.9038 -0.5647

Columns 2199 through 2205

-1.1356 -1.2458 -1.4574 -1.4991 -1.5319 -1.5634 -1.5420
1.4835 1.3831 1.1082 0.9600 0.8102 0.5031 0.2981
-0.3479 -0.1373 0.3491 0.5391 0.7217 1.0603 1.2439

Columns 2206 through 2212

-1.5192 -1.4309 -1.3234 -1.1966 -1.0549 -0.8610 -0.6763
0.0965 -0.2320 -0.5188 -0.7691 -1.0147 -1.2881 -1.5151
1.4227 1.6629 1.8422 1.9657 2.0696 2.1490 2.1914

Columns 2213 through 2219

-0.4748 -0.2426 -0.0586 0.1206 0.3133 0.5286 0.7914
-1.7233 -1.9265 -2.0589 -2.1643 -2.2542 -2.3232 -2.3709
2.1981 2.1691 2.1175 2.0437 1.9409 1.7946 1.5795

Columns 2220 through 2226

0.9261 1.0554 1.2350 1.3846 1.4960 1.6034 1.6595
-2.3584 -2.3255 -2.2598 -2.1571 -2.0378 -1.8625 -1.6798
1.4322 1.2701 1.0247 0.7725 0.5418 0.2591 0.0204

Columns 2227 through 2233

1.6827 1.6881 1.6462 1.5924 1.5073 1.3769 1.2186
-1.4676 -1.1558 -0.9419 -0.7299 -0.4559 -0.1487 0.1516
-0.2151 -0.5323 -0.7043 -0.8624 -1.0514 -1.2282 -1.3701

Columns 2234 through 2240

1.0431 0.8610 0.5082 0.3205 0.1278 -0.1849 -0.4045
0.4174 0.6613 1.0590 1.2385 1.4024 1.6443 1.7864
-1.4606 -1.5223 -1.5672 -1.5589 -1.5302 -1.4594 -1.3820

Columns 2241 through 2247

-0.6160 -0.8669 -1.1056 -1.4603 -1.6060 -1.7400 -1.9331
1.9217 2.0497 2.1481 2.2235 2.2375 2.2359 2.2022
-1.3057 -1.1828 -1.0425 -0.7633 -0.6315 -0.4959 -0.2690

Columns 2248 through 2254

-2.0884 -2.1853 -2.2750 -2.3264 -2.3348 -2.3152 -2.2503
2.1242 2.0515 1.9050 1.7427 1.5438 1.3606 1.0791
-0.0358 0.1338 0.3700 0.5837 0.7910 0.9545 1.1712

Columns 2255 through 2261

-2.1695 -2.0581 -1.9215 -1.7277 -1.5069 -1.2491 -1.0236
0.8813 0.6658 0.4285 0.1417 -0.1404 -0.4128 -0.6112
1.2882 1.3924 1.4930 1.5859 1.6474 1.6619 1.6348

Columns 2262 through 2268

-0.8005 -0.5505 -0.2901 0.0844 0.3305 0.5718 0.8156
-0.7855 -0.9668 -1.1360 -1.3453 -1.4495 -1.5210 -1.5766
1.5859 1.5173 1.4261 1.2609 1.1190 0.9491 0.7610

Columns 2269 through 2275

-2.2308 -2.1698 -2.0880 -1.9714 -1.8263 -1.6017 -1.0949 1.3551 1.5794 1.7615 1.9449 2.0797 1.4250
2.1831 1.4726 1.2624 1.0424 0.7866 0.5206 0.1657 -1.6018 -1.6009 -1.5595 -1.5022 -1.3891 -1.2733 -0.0417
1.1306 0.5069 0.2457 -0.0199 -0.2593 -0.5558 -0.8065 -Columns 2325 through 2331
1.0524 1.4769 1.4781 1.4532 1.4050 1.3035 1.2046
Columns 2276 through 2282 1.0795
-1.2236 -1.0158 -0.7340 -0.4589 -0.1436 0.0757 2.2777 2.2988 2.3001 2.2699 2.1957 2.1143 0.3047
1.9788 -0.2533 -0.4623 -0.7192 -0.9461 -1.1600 -1.2802 -0.8780 -0.7372 -0.5995 -0.3785 -0.1519 0.0447 1.3842
0.2665 -1.3997 -1.5616 -1.7006 -1.8914 -2.0438 -2.1590 -Columns 2332 through 2338
2.2454 0.9253 0.7165 0.4554 0.2652 0.0652 -0.1454 -
Columns 2283 through 2289 0.3941
0.5621 0.8602 1.1873 1.3902 1.5766 1.7564 1.8042 1.6522 1.4919 1.2753 1.0580 0.6955 1.9369
0.4865 -1.4874 -1.5768 -1.6427 -1.6554 -1.6418 -1.6111 -0.5008 0.6467 0.7873 0.9580 1.1063 1.3292 1.5429
1.4119 -2.3050 -2.2989 -2.2792 -2.2332 -2.1643 -2.0246 -Columns 2339 through 2345
1.8984 -0.6179 -0.9631 -1.1423 -1.3095 -1.4507 -1.5967 -
Columns 2290 through 2296 1.7294
2.0875 2.2583 2.3333 2.3694 2.3937 2.3871 0.2708 0.0727 -0.1507 -0.3635 -0.6303 -0.8604 -2.3660
1.0381 -1.4695 -1.2952 -1.1910 -1.0599 -0.9430 -0.7904 -1.4677 1.5080 1.5310 1.5405 1.5275 1.4810 0.6366
1.3978 -1.7385 -1.5807 -1.3803 -1.1769 -0.8971 -0.6207 -Columns 2346 through 2352
0.3597 -1.9106 -2.0303 -2.0893 -2.1100 -2.1238 -2.1037 -
Columns 2297 through 2303 2.0534
2.2822 2.1723 2.0249 1.8862 1.6506 1.4483 -1.1771 -1.3552 -1.4549 -1.5263 -1.5826 -1.6013 -1.2238
1.5913 -0.3715 -0.1419 0.0644 0.2237 0.4731 0.6555 1.3048 1.1264 0.9797 0.8145 0.6162 0.3373 0.8296
0.1067 -0.1277 0.2288 0.4752 0.7118 0.9664 1.2639 Columns 2353 through 2359
1.4847 -1.9555 -1.8407 -1.7070 -1.5345 -1.3149 -1.0545 -
Columns 2304 through 2310 0.8743
0.9002 0.6897 0.4719 0.2074 -0.0904 -0.4064 -1.5486 -1.4542 -1.3656 -1.2664 -1.1563 -1.0032 -0.5794
0.8490 1.0553 1.1510 1.2350 1.3271 1.4053 1.4608 -0.1432 -0.4749 -0.6977 -0.9008 -1.0936 -1.3103 -1.4538
1.5072 1.6918 1.9291 2.0633 2.1672 2.2500 2.3135 Columns 2360 through 2366
2.3561 -0.6899 -0.2960 -0.0908 0.1062 0.4679 0.6792
Columns 2311 through 2317 0.8878
-0.7447 -1.0642 -1.1951 -1.3212 -1.5091 -1.5924 -0.5399 -0.3959 -0.2454 -0.0861 0.1321 0.3442 1.6531
0.5591 1.4346 1.3602 1.2859 1.2150 1.0412 0.9132 -1.8086 -1.9257 -2.0084 -2.0894 -2.1645 -2.2247 -0.7653
2.2441 2.3486 2.3217 2.2537 2.1754 2.0324 1.8805 Columns 2367 through 2373
1.6849 1.2512 1.4402 1.6172 1.8258 2.0020 2.1591
Columns 2318 through 2324 2.2785
-1.7203 -1.7292 -1.7415 -1.7195 -1.6682 -1.5673 -0.7582 0.9073 1.0456 1.1847 1.3057 1.4359 1.4352
1.4667 0.4691 0.2890 0.1242 -0.1063 -0.3338 -0.5918 -0.8433

Columns 2374 through 2380

2.3514 2.3959 2.4166 2.3969 2.3412 2.2325 2.1114
-1.3084 -1.1834 -0.9863 -0.8171 -0.6292 -0.3679 -0.1618
-1.0431 -1.2125 -1.4304 -1.5798 -1.7121 -1.8646 -1.9496

Columns 2381 through 2387

1.9656 1.7506 1.5699 1.3733 1.1615 0.9102 0.6444
0.0418 0.3045 0.4738 0.6381 0.7982 0.9673 1.1234
-2.0074 -2.0551 -2.0437 -2.0114 -1.9597 -1.8775 -1.7678

Columns 2388 through 2394

0.3402 0.0375 -0.2034 -0.4104 -0.6512 -0.8664 -1.1635
1.2688 1.3786 1.4308 1.4584 1.4755 1.4677 1.4082
-1.6090 -1.4160 -1.2274 -1.0480 -0.8243 -0.6013 -0.2447

Columns 2395 through 2401

-1.3174 -1.4508 -1.5670 -1.6907 -1.7758 -1.8375 -1.8650
1.3377 1.2342 1.1278 0.9462 0.7868 0.5546 0.2988
-0.0203 0.2166 0.4392 0.7445 0.9890 1.2830 1.5662

Columns 2402 through 2408

-1.8405 -1.8043 -1.7070 -1.5953 -1.4719 -1.3432 -1.1064
0.0566 -0.1438 -0.4313 -0.6839 -0.8965 -1.0801 -1.3463
1.7838 1.9480 2.1383 2.2792 2.3684 2.4233 2.4527

Columns 2409 through 2415

-0.9594 -0.7964 -0.6121 -0.3750 -0.1354 0.0934 0.2918
-1.4793 -1.6038 -1.7276 -1.8567 -1.9543 -2.0078 -2.0272
2.4387 2.4002 2.3397 2.2317 2.0897 1.9145 1.7354

Columns 2416 through 2422

0.5380 0.7901 0.9773 1.0991 1.2376 1.3365 1.4324
-2.0311 -1.9989 -1.9246 -1.8285 -1.6882 -1.5379 -1.3521
1.4931 1.2088 0.9473 0.7294 0.4505 0.2014 -0.0803

Columns 2423 through 2429

1.5122 1.5351 1.5259 1.4850 1.4090 1.3098 1.1364
-1.1135 -0.9128 -0.6955 -0.4176 -0.1424 0.1372 0.4997
-0.3987 -0.6223 -0.8304 -1.0674 -1.2667 -1.4470 -1.6361

Columns 2430 through 2436

1.0070 0.8546 0.6670 0.4040 0.1630 -0.0902 -0.3029
0.7198 0.9402 1.1905 1.4838 1.7271 1.9355 2.0863
-1.7268 -1.7948 -1.8574 -1.8878 -1.8902 -1.8453 -1.7834

Columns 2437 through 2443

-0.5321 -0.8026 -0.9900 -1.1551 -1.3279 -1.4950 -1.6608
2.2146 2.3385 2.4008 2.4376 2.4476 2.4205 2.3533
-1.6825 -1.5359 -1.4107 -1.2825 -1.1196 -0.9255 -0.6925

Columns 2444 through 2450

-1.8003 -1.8917 -1.9519 -2.0079 -2.0063 -1.9793 -1.9181
2.2476 2.1257 1.9769 1.7542 1.5547 1.3375 1.0379
-0.4474 -0.2339 -0.0250 0.2538 0.4516 0.6418 0.8802

Columns 2451 through 2457

-1.8271 -1.7060 -1.4941 -1.3418 -1.1823 -0.8299 -0.6504
0.7848 0.5277 0.1385 -0.0668 -0.2662 -0.6796 -0.8403
1.0423 1.1783 1.3556 1.4085 1.4485 1.5095 1.4907

Columns 2458 through 2464

-0.4771 -0.1487 0.0773 0.3114 0.5636 0.8370 1.1345
-0.9939 -1.2496 -1.3932 -1.5265 -1.6500 -1.7574 -1.8428
1.4710 1.3984 1.3159 1.2150 1.0864 0.9204 0.7082

Columns 2465 through 2471

1.3682 1.5816 1.8148 1.9858 2.1332 2.2876 2.3618
-1.8848 -1.8978 -1.8769 -1.8303 -1.7612 -1.6251 -1.5138
0.5166 0.3162 0.0620 -0.1554 -0.3720 -0.6625 -0.8480

Columns 2472 through 2478

2.4038 2.4276 2.4128 2.3648 2.2682 2.1445 1.9851
-1.3848 -1.2267 -0.9910 -0.7658 -0.5227 -0.2943 -0.0678

-1.0189 -1.2010 -1.4219 -1.5989 -1.7455 -1.8502 -1.9173

Columns 2479 through 2485

1.7853 1.5813 1.3652 1.0642 0.8495 0.6231 0.3858
0.1859 0.4035 0.6063 0.8554 0.9918 1.1148 1.2319
-1.9713 -1.9847 -1.9714 -1.9196 -1.8413 -1.7379 -1.6177

Columns 2486 through 2492

0.0936 -0.1762 -0.4522 -0.7173 -0.9447 -1.1398 -1.4455
1.3561 1.4472 1.5078 1.5335 1.5179 1.4800 1.3566
-1.4497 -1.2710 -1.0556 -0.8163 -0.5732 -0.3402 0.0889

Columns 2493 through 2499

-1.5691 -1.6698 -1.8387 -1.8770 -1.9296 -1.9527 -1.9405
1.2661 1.1565 0.8905 0.7442 0.6165 0.3187 0.1154
0.3030 0.5133 0.9482 1.1328 1.3130 1.6340 1.8251

Columns 2500 through 2506

-1.8837 -1.8018 -1.6869 -1.5665 -1.3492 -1.2221 -1.0649
-0.0989 -0.3398 -0.5756 -0.7914 -1.0798 -1.2131 -1.3438
1.9826 2.1416 2.2625 2.3579 2.4290 2.4352 2.4087

Columns 2507 through 2513

-0.8741 -0.6135 -0.3902 -0.1582 0.0916 0.2889 0.4775
-1.4895 -1.6542 -1.7689 -1.8502 -1.9101 -1.9216 -1.9130
2.3637 2.2678 2.1592 2.0084 1.8184 1.6327 1.4355

Columns 2514 through 2520

0.6718 0.8649 1.0603 1.1760 1.2796 1.3724 1.4556
-1.8835 -1.8229 -1.7191 -1.6018 -1.4657 -1.3094 -1.1129
1.2117 0.9580 0.6588 0.4258 0.1861 -0.0629 -0.3427

Columns 2521 through 2527

1.5158 1.5209 1.4914 1.4186 1.3246 1.2133 0.9981
-0.8606 -0.6414 -0.4069 -0.0878 0.1816 0.4502 0.8433
-0.6552 -0.8795 -1.0845 -1.3308 -1.5062 -1.6636 -1.8415

Columns 2528 through 2534

0.8543 0.6911 0.5124 0.1726 -0.0425 -0.2662 -0.5294
1.0627 1.2720 1.4885 1.8130 2.0031 2.1527 2.3033
-1.9170 -1.9631 -2.0010 -1.9855 -1.9606 -1.8865 -1.7738

Columns 2535 through 2541

-0.6973 -0.8511 -1.1100 -1.2921 -1.4232 -1.5793 -1.7058
2.3687 2.4214 2.4677 2.4662 2.4278 2.3520 2.2381
-1.6714 -1.5703 -1.3577 -1.1742 -1.0046 -0.7727 -0.5323

Columns 2542 through 2548

-1.7894 -1.8618 -1.9191 -1.9111 -1.8751 -1.8185 -1.7145
2.1068 1.9373 1.6983 1.5028 1.2914 1.0468 0.7310
-0.3174 -0.0755 0.2208 0.4082 0.5837 0.7717 0.9836

Columns 2549 through 2555

-1.5630 -1.4319 -1.2882 -1.0278 -0.8543 -0.6828 -0.4316
0.3695 0.1530 -0.0590 -0.4159 -0.6076 -0.7939 -1.0358
1.1935 1.2790 1.3472 1.4437 1.4619 1.4767 1.4674

Columns 2556 through 2562

-0.1669 0.1460 0.4304 0.6661 0.9407 1.1904 1.5071
-1.2563 -1.4734 -1.6374 -1.7521 -1.8646 -1.9461 -2.0023
1.4232 1.3274 1.2069 1.0861 0.9240 0.7557 0.4952

Columns 2563 through 2569

1.6979 1.8663 2.0175 2.1775 2.2945 2.3963 2.4471
-2.0165 -1.9993 -1.9701 -1.8940 -1.8126 -1.6608 -1.5092
0.3186 0.1331 -0.0473 -0.2835 -0.4820 -0.7355 -0.9379

Columns 2570 through 2576

2.4507 2.4289 2.3656 2.2514 2.1369 2.0064 1.8208
-1.3186 -1.1325 -0.8930 -0.5917 -0.4147 -0.2395 0.0012
-1.1321 -1.2964 -1.4725 -1.6597 -1.7222 -1.7669 -1.8220

Columns 2577 through 2583

1.5830 1.3363 1.0681 0.8226 0.5686 0.2617 0.0089

0.2670 0.5114 0.7298 0.8946 1.0356 1.1924 1.2941
-1.8500 -1.8477 -1.7979 -1.7172 -1.6042 -1.4541 -1.3030

Columns 2584 through 2590

-0.2330 -0.5187 -0.7561 -0.9853 -1.2479 -1.4166 -1.5639
1.3782 1.4476 1.4730 1.4645 1.4172 1.3552 1.2787
-1.1453 -0.9289 -0.7168 -0.4792 -0.1693 0.0613 0.2852

Columns 2591 through 2597

-1.7081 -1.8347 -1.9488 -2.0080 -2.0422 -2.0433 -1.9960
1.1739 1.0324 0.8346 0.6672 0.4888 0.2675 -0.0015
0.5342 0.8023 1.1142 1.3409 1.5534 1.7758 1.9975

Columns 2598 through 2604

-1.9450 -1.8752 -1.7865 -1.6520 -1.4869 -1.2869 -1.0651
-0.1729 -0.3408 -0.5120 -0.7141 -0.9203 -1.1207 -1.3024
2.1179 2.2161 2.2985 2.3661 2.4072 2.4076 2.3676

Columns 2605 through 2611

-0.8527 -0.6548 -0.3212 -0.1227 0.0723 0.2663 0.4891
-1.4362 -1.5343 -1.6851 -1.7374 -1.7674 -1.7805 -1.7682
2.2889 2.1890 2.0063 1.8600 1.6951 1.5141 1.2791

Columns 2612 through 2618

0.6878 0.9240 1.0760 1.2030 1.3229 1.4363 1.4969
-1.7277 -1.6426 -1.5495 -1.4287 -1.2882 -1.0720 -0.8799
1.0400 0.7186 0.4734 0.2257 -0.0347 -0.3643 -0.6170

Columns 2619 through 2625

1.5225 1.5120 1.4649 1.4065 1.2167 1.1101 1.0152
-0.6603 -0.2946 -0.0828 0.1221 0.5954 0.7858 0.9710
-0.8622 -1.2174 -1.3822 -1.5285 -1.8122 -1.8959 -1.9862

Columns 2626 through 2632

0.8019 0.6381 0.4638 0.2869 0.0734 -0.1326 -0.4236
1.2838 1.4896 1.6757 1.8442 2.0149 2.1557 2.3040
-2.0856 -2.1277 -2.1394 -2.1311 -2.0883 -2.0231 -1.8804

Columns 2633 through 2639

-0.5790 -0.7363 -0.8914 -1.0925 -1.2495 -1.4079 -1.5348
2.3573 2.3839 2.3952 2.3791 2.3356 2.2435 2.1218
-1.7783 -1.6476 -1.5038 -1.2866 -1.0862 -0.8356 -0.5870

Columns 2640 through 2646

-1.6345 -1.6934 -1.7312 -1.7462 -1.7123 -1.6515 -1.5423
1.9554 1.7869 1.5792 1.3199 1.0856 0.8443 0.4965
-0.3209 -0.0935 0.1520 0.4263 0.6267 0.8072 1.0458

Columns 2647 through 2653

-1.4301 -1.3022 -1.1333 -0.9191 -0.6891 -0.4758 -0.2440
0.2488 -0.0001 -0.2814 -0.5768 -0.8441 -1.0599 -1.2705
1.1813 1.3022 1.4147 1.4960 1.5332 1.5357 1.5145

Columns 2654 through 2660

0.0068 0.3207 0.5859 0.8462 1.1478 1.3526 1.5397
-1.4757 -1.6949 -1.8544 -1.9792 -2.0900 -2.1430 -2.1766
1.4689 1.3742 1.2684 1.1330 0.9422 0.7904 0.6369

Columns 2661 through 2667

1.7206 1.8967 2.0628 2.1845 2.2766 2.3540 2.3710
-2.1881 -2.1721 -2.1181 -2.0360 -1.9223 -1.7425 -1.5955
0.4675 0.2755 0.0553 -0.1485 -0.3543 -0.6115 -0.7755

Columns 2668 through 2674

2.3623 2.3286 2.2447 2.1381 1.9954 1.7769 1.6058
-1.4334 -1.2265 -0.9502 -0.7128 -0.4632 -0.1261 0.0662
-0.9289 -1.1021 -1.2946 -1.4253 -1.5322 -1.6508 -1.6720

Columns 2675 through 2681

1.4270 1.1736 0.9039 0.6585 0.3736 0.0297 -0.1775
0.2480 0.4934 0.7177 0.9018 1.0854 1.2752 1.3563
-1.6750 -1.6670 -1.6216 -1.5603 -1.4591 -1.3048 -1.1787

Columns 2682 through 2688

-0.3805 -0.7469 -0.9453 -1.1358 -1.3912 -1.6206 -1.7907
1.4252 1.5164 1.5370 1.5459 1.5238 1.4575 1.3766
-1.0447 -0.7696 -0.5917 -0.4101 -0.1326 0.1630 0.4140

Columns 2689 through 2695

-1.9277 -2.0545 -2.1505 -2.2205 -2.2367 -2.2300 -2.1942
1.2791 1.1493 0.9891 0.7489 0.5899 0.4216 0.2189
0.6486 0.9052 1.1615 1.4716 1.6469 1.8084 1.9752

Columns 2696 through 2702

-2.1007 -2.0085 -1.8843 -1.7142 -1.5365 -1.3447 -1.0352
-0.0436 -0.2270 -0.4085 -0.6220 -0.8009 -0.9705 -1.2085
2.1442 2.2355 2.2928 2.3362 2.3374 2.3152 2.2437

Columns 2703 through 2709

-0.8367 -0.6260 -0.4276 -0.1627 0.0578 0.3435 0.5986
-1.3188 -1.4107 -1.4841 -1.5616 -1.6040 -1.6293 -1.6185
2.1555 2.0368 1.9117 1.7243 1.5463 1.2858 1.0199

Columns 2710 through 2716

0.7984 0.9531 1.1240 1.2724 1.4349 1.5114 1.5551
-1.5638 -1.4988 -1.3998 -1.2808 -1.0724 -0.9064 -0.7082
0.7654 0.5457 0.2758 0.0084 -0.3625 -0.6050 -0.8469

Columns 2717 through 2723

1.5817 1.5707 1.5400 1.4191 1.3435 1.2479 1.0672
-0.4698 -0.2256 0.0194 0.4244 0.6159 0.8087 1.1133
-1.1119 -1.3450 -1.5594 -1.8435 -1.9594 -2.0566 -2.1805

Columns 2724 through 2730

0.9293 0.8064 0.6109 0.4465 0.2288 0.0020 -0.2133
1.3047 1.4743 1.6845 1.8345 1.9863 2.1143 2.2012
-2.2340 -2.2807 -2.2954 -2.2810 -2.2151 -2.1163 -1.9879

Columns 2731 through 2737

-0.3758 -0.6380 -0.7705 -0.9032 -1.0603 -1.2311 -1.4077
2.2439 2.2838 2.2647 2.2215 2.1573 2.0507 1.8854
-1.8681 -1.6458 -1.4943 -1.3183 -1.0969 -0.8196 -0.4777

Columns 2738 through 2744

-1.4579 -1.4899 -1.5514 -1.5379 -1.5298 -1.4913 -1.4192
1.7473 1.5977 1.3124 1.1173 0.9286 0.6541 0.3714
-0.2894 -0.1078 0.2389 0.4206 0.6012 0.8372 1.0478

Columns 2745 through 2751

-1.2974 -1.1653 -1.0048 -0.7975 -0.5966 -0.3837 -0.1035
0.0761 -0.1808 -0.4451 -0.7481 -0.9989 -1.2321 -1.4932
1.2214 1.3461 1.4498 1.5455 1.5955 1.6158 1.5967

Columns 2752 through 2758

0.1194 0.3474 0.5664 0.8141 1.0461 1.3332 1.4877
-1.6759 -1.8434 -1.9890 -2.1245 -2.2250 -2.3011 -2.3246
1.5564 1.4961 1.4226 1.3104 1.1789 0.9679 0.8369

Columns 2759 through 2765

1.6314 1.8098 1.9689 2.0735 2.1467 2.2061 2.2126
-2.3265 -2.2997 -2.2271 -2.1330 -2.0007 -1.8144 -1.6499
0.6951 0.4900 0.2582 0.0595 -0.1460 -0.3918 -0.5626

Columns 2766 through 2772

2.1966 2.1590 2.0812 1.9724 1.8278 1.6265 1.4428
-1.4743 -1.2555 -0.9830 -0.7328 -0.4750 -0.1554 0.0754
-0.7223 -0.9036 -1.0982 -1.2395 -1.3528 -1.4711 -1.5182

Columns 2773 through 2779

1.2494 1.0254 0.7708 0.5145 0.2469 -0.0633 -0.3164
0.2971 0.5322 0.7672 0.9742 1.1610 1.3457 1.4683
-1.5465 -1.5576 -1.5380 -1.4887 -1.4080 -1.2824 -1.1519

Columns 2780 through 2786

-0.5671 -0.8459 -1.1453 -1.3995 -1.6006 -1.8089 -1.9868
1.5664 1.6487 1.6988 1.7097 1.6931 1.6462 1.5721
-0.9992 -0.8028 -0.5535 -0.3102 -0.0925 0.1626 0.4147

Columns 2787 through 2793

1.5927 1.3867 1.1636 0.8804 0.6029 0.3469
-2.1597 -2.2573 -2.3263 -2.3793 -2.3953 -2.3752 -
0.0935
2.3107 -0.1635 0.1033 0.3671 0.6576 0.9017 1.0853
1.4415 1.3296 1.1997 1.0417 0.8164 0.6162
1.2449
0.4018
0.7182 0.9277 1.1266 1.3376 1.5789 1.7590 Columns 2843 through 2849
1.9089

-1.1590 -1.0220 -0.8670 -0.7002 -0.4766 -0.2530
Columns 2794 through 2800 0.0432
-0.2885 -0.5281 -0.7605 -0.9877 -1.2456 -1.4793 -
-2.2050 -2.0539 -1.9081 -1.7423 -1.5244 -1.2591 - 1.7397
1.0184 1.4475 1.5501 1.6275 1.6879 1.7221 1.7323
0.1608 -0.0894 -0.2837 -0.4693 -0.6860 -0.9104 -
1.6964
1.0764
2.0442 2.1433 2.1918 2.2116 2.2104 2.1696 Columns 2850 through 2856
2.0947

0.2774 0.5105 0.8242 0.9982 1.1618 1.4084
Columns 2801 through 2807 1.5978
-1.9247 -2.0748 -2.2352 -2.3024 -2.3582 -2.4011 -
-0.7973 -0.4315 -0.2303 -0.0380 0.3616 0.5296 2.3990
0.6957 1.6473 1.5643 1.4110 1.3041 1.1964 0.9927
-1.1992 -1.3845 -1.4455 -1.4878 -1.5705 -1.5579 -
0.8011
1.5445
1.9965 1.8160 1.6758 1.5258 1.2089 1.0283 Columns 2857 through 2863
0.8488

1.7286 1.8553 1.9608 2.0723 2.1053 2.1097
Columns 2808 through 2814 2.0993
-2.3661 -2.2943 -2.1902 -2.0115 -1.8700 -1.7079 -
0.9843 1.1300 1.2604 1.4509 1.5392 1.6151 1.5198
1.6846 0.6376 0.4390 0.2295 -0.0608 -0.2353 -0.4018 -
-1.4934 -1.4272 -1.3455 -1.1776 -1.0418 -0.8959 - 0.5794
0.6814
0.5092 0.2972 0.0851 -0.2734 -0.4973 -0.7192 - Columns 2864 through 2870
1.0032

2.0550 1.9660 1.8561 1.7293 1.5787 1.3893
Columns 2815 through 2821 1.1811
-1.2552 -0.9271 -0.7045 -0.4836 -0.2505 0.0104
1.7176 1.7161 1.6758 1.5942 1.4889 1.3796 0.2720
1.2264 -0.7999 -1.0389 -1.1516 -1.2457 -1.3282 -1.3997 -
-0.4431 -0.1925 0.0793 0.3683 0.6287 0.8449 1.4530
1.0906
-1.2745 -1.5236 -1.7551 -1.9625 -2.1176 -2.2245 - Columns 2871 through 2877
2.3170

0.8786 0.6850 0.4781 0.1601 -0.0665 -0.2842 -
Columns 2822 through 2828 0.5972
0.6035 0.7815 0.9477 1.1818 1.3165 1.4394
1.0378 0.8494 0.6857 0.4847 0.2717 0.0066 - 1.5818
0.2358 -1.4820 -1.4665 -1.4258 -1.3420 -1.2500 -1.1553 -
1.3414 1.5474 1.6960 1.8459 1.9741 2.0973 0.9846
2.1743
-2.3792 -2.3968 -2.3817 -2.3306 -2.2458 -2.1039 - Columns 2878 through 2884
1.9385

-0.8529 -1.0953 -1.4611 -1.6440 -1.7985 -2.0808 -
Columns 2829 through 2835 2.1648
1.6716 1.7299 1.7656 1.7612 1.7419 1.6314
-0.4163 -0.5840 -0.7708 -0.9534 -1.1592 -1.2698 - 1.5624
1.3576 -0.8187 -0.6346 -0.3045 -0.1171 0.0566 0.4494
2.1944 2.1886 2.1606 2.1046 2.0027 1.8901 0.6024
1.7510
-1.7780 -1.6046 -1.3898 -1.1511 -0.8435 -0.6203 - Columns 2885 through 2891
0.3933

-2.2548 -2.3758 -2.4205 -2.4312 -2.4239 -2.3605 -
Columns 2836 through 2842 2.2804
1.5000 1.3083 1.1863 1.0534 0.8895 0.6361
-1.4292 -1.4899 -1.5306 -1.5379 -1.5047 -1.4322 - 0.4267
1.3385 0.7548 1.0675 1.2342 1.3778 1.5344 1.7244
1.8538

Columns 2892 through 2898

-2.1583 -1.9967 -1.8272 -1.6430 -1.4047 -1.1299 -0.8664

0.2051 -0.0453 -0.2415 -0.4315 -0.6514 -0.8713 -1.0427

1.9532 2.0420 2.0687 2.0745 2.0561 2.0012 1.9091

Columns 2899 through 2905

-0.6413 -0.3619 -0.0997 0.1843 0.4550 0.7104 0.9244

-1.1591 -1.2849 -1.3778 -1.4558 -1.5016 -1.5106 -1.4858

1.8004 1.6469 1.4775 1.2715 1.0466 0.8001 0.5614

Columns 2906 through 2912

1.1199 1.3546 1.4889 1.6014 1.7027 1.7833 1.8402

-1.4305 -1.3196 -1.2174 -1.0973 -0.9532 -0.7620 -0.5323

0.3106 -0.0349 -0.2715 -0.5041 -0.7495 -1.0213 -1.3079

Columns 2913 through 2919

1.8531 1.8343 1.7823 1.6644 1.5713 1.4712 1.3512

-0.2548 -0.0211 0.2059 0.5242 0.7139 0.8927 1.0727

-1.5983 -1.8132 -1.9883 -2.1885 -2.2851 -2.3639 -2.4239

Columns 2920 through 2926

1.1938 1.0137 0.7931 0.5578 0.3516 0.1620 -0.1076

1.2629 1.4438 1.6210 1.7735 1.8723 1.9395 2.0108

-2.4567 -2.4576 -2.4140 -2.3313 -2.2239 -2.1015 -1.9032

Columns 2927 through 2933

-0.3540 -0.5382 -0.7081 -0.8879 -1.0530 -1.2413 -1.3165

2.0395 2.0235 1.9838 1.9143 1.8120 1.6421 1.5028

-1.6855 -1.4854 -1.2757 -1.0264 -0.7590 -0.4008 -0.1863

Columns 2934 through 2940

-1.3762 -1.4494 -1.4738 -1.4829 -1.4505 -1.3898 -1.2891

1.3507 1.1252 0.9222 0.7133 0.4097 0.1523 -0.1206

0.0255 0.3242 0.5516 0.7696 1.0407 1.2375 1.4097

Columns 2941 through 2947

-1.0984 -0.9688 -0.8398 -0.5397 -0.3844 -0.2263 0.0261

-0.5261 -0.7421 -0.9468 -1.3344 -1.5099 -1.6817 -1.9089

1.6245 1.7109 1.7865 1.8740 1.8943 1.9080 1.8828

Columns 2948 through 2954

0.2860 0.4850 0.6861 0.9022 1.1565 1.2948 1.4231

-2.1019 -2.2290 -2.3336 -2.4178 -2.4740 -2.4803 -2.4598

1.8159 1.7441 1.6475 1.5155 1.3175 1.1855 1.0367

Columns 2955 through 2961

1.5428 1.6675 1.7668 1.8728 1.9066 1.9145 1.9049

-2.4168 -2.3351 -2.2272 -2.0426 -1.8824 -1.6945 -1.4788

0.8740 0.6676 0.4604 0.1699 -0.0242 -0.2199 -0.4262

Columns 2962 through 2968

1.8557 1.7785 1.6630 1.4418 1.2928 1.1495 0.8643

-1.1677 -0.9192 -0.6569 -0.2073 -0.0006 0.1941 0.5580

-0.6881 -0.8593 -1.0061 -1.2345 -1.2922 -1.3436 -1.4223

Columns 2969 through 2975

0.6910 0.5168 0.2623 -0.0066 -0.3327 -0.5672 -0.7996

0.7375 0.9160 1.1431 1.3507 1.5612 1.6927 1.8040

-1.4285 -1.4328 -1.4053 -1.3440 -1.2285 -1.1255 -1.0044

Columns 2976 through 2982

-1.0364 -1.2894 -1.5265 -1.7845 -1.9538 -2.0991 -2.2293

1.9023 1.9799 2.0328 2.0459 2.0374 1.9997 1.9430

-0.8659 -0.6905 -0.5063 -0.2614 -0.0835 0.0994 0.2863

Columns 2983 through 2989

-2.3474 -2.4224 -2.4547 -2.4507 -2.4057 -2.3226 -2.2019

1.8316 1.7088 1.5542 1.3914 1.1765 0.9364 0.7029

0.5158 0.7136 0.9005 1.0593 1.2292 1.3862 1.4991

Columns 2990 through 2996

-2.0594 -1.8821 -1.6651 -1.4206 -1.1843 -0.9373 -0.6034

0.4924 0.2657 0.0177 -0.2267 -0.4280 -0.6121 -0.8302

```
   1.5670    1.6164    1.6475    1.6473    1.6124    1.5494
1.4336

 Columns 2997 through 3003

  -0.3784   -0.1567    0.1114    0.4069    0.7197    0.9059
1.0835
  -0.9463   -1.0514   -1.1694   -1.2711   -1.3476   -1.3681   -
1.3735
   1.3248    1.2082    1.0580    0.8643    0.6279    0.4622
0.2900

 Columns 3004 through 3010

   1.2477    1.4005    1.6012    1.7687    1.8824    1.9633
2.0227
  -1.3662   -1.3434   -1.2660   -1.1559   -1.0396   -0.9087   -
0.6530
   0.1185   -0.0571   -0.3353   -0.6128   -0.8429   -1.0545   -
1.3698

 Columns 3011 through 3017

   2.0332    2.0262    1.9419    1.8836    1.8246    1.6327
1.5149
  -0.5345   -0.4288   -0.1234   -0.0348    0.0435    0.2950
0.3786
  -1.4987   -1.5974   -1.8184   -1.8488   -1.8681   -1.9276   -
1.8934

 Columns 3018 through 3024

   1.3842    1.1774    0.9594    0.7632    0.5236    0.2437
0.0455
   0.4606    0.6037    0.7250    0.8069    0.8926    0.9801
1.0120
  -1.8449   -1.7811   -1.6844   -1.5701   -1.4163   -1.2238   -
1.0575

 Columns 3025 through 3031

  -0.1452   -0.3379   -0.5371   -0.7260   -0.9185   -1.0882   -
1.2037
   1.0289    1.0321    1.0138    0.9718    0.9018    0.8155
0.7245
  -0.8837   -0.6942   -0.4768   -0.2458    0.0167    0.2727
0.4792

 Columns 3032 through 3038

  -1.2880   -1.3818   -1.4020   -1.3975   -1.3766   -1.3071   -
1.2120
   0.6387    0.4514    0.3391    0.2195    0.0734   -0.1469   -
0.3427
   0.6493    0.9305    1.0629    1.1780    1.3032    1.4540
1.5547

 Columns 3039 through 3045

  -1.0700   -0.9178   -0.7362   -0.4494   -0.2878   -0.1335
0.2169
  -0.5465   -0.7159   -0.8707   -1.1052   -1.1379   -1.1354   -
1.2553
   1.6165    1.6337    1.6069    1.5546    1.4258    1.2689
1.0384

 Columns 3046 through 3047

   0.3821    0.5113
  -1.1905   -1.0891
   0.8085    0.5778

>>
```

PUBLICATIONS

(1) Aneke J. I., Ezechukwu O. A. and Onuegbu J. C. (2018). Multi-resolution Analysis & Neural Network Based Fault Diagnosis & Location in Transmission Lines. Journal of Engineering and Applied Science (JEAS), 1119-8109.

(2) Aneke Jude I., Ezechukwu O. A., and Ebune R. U. (2018). "Artificial Intelligence Based Fault Detection and Classification In Transmission Lines" Iconic Research And Engineering Journals Volume 2 Issue 4 Page 38-46

(3) Aneke Jude I., Ezechukwu O. A., and Uwaechi P. C. (2018). "Analysis of Neural Network Back-propagation Algorithm" Iconic Research and Engineering Journals Volume 2 Issue 4 Page 33-37

(4) Aneke J. I., Ezechukwu O. A. and Ndubisi M. A. (2018). Wavelet Transform based Fault Diagnosis & Classification in Transmission Lines. Scholars Journal of Engineering and Technology (SJET). Vol-6, Issue-9; ISSN 2321-435X (Online) & ISSN 2347-9523

(5) Aneke Jude I., Ezechukwu O. A. and Tagbo P. I. (2018) Hybrid Pattern Recognition and Multi-Resolution Analysis (MRA) Based Fault Location in Power Transmission Lines. American Journal of Engineering Research (AJER). www.ajer.org(e)ISSN 2320-0847 & (p)ISSN 2320-0936

(6) Aneke Jude I., Ezechukwu O. A. and Tagbo P. I. (2018) Back-Propagation Algorithm Based Fault Location Prediction in Transmission Lines. International Journal of Engineering Inventions.www.ijeijournal.com. (e)ISSN: 2278-7461 (p)ISSN: 2319-6491.