IMPROVED FAULT DIAGNOSIS IN ANALOG CIRCUITS USING HYBRID NEURAL NETWORK AND EVOLUTIONARY TECHNIQUES

PROJECT REPORT

Submitted by

DEEPAK B

Register No: 13MAE02

in partial fulfillment for the requirement of award of the degree

of

MASTER OF ENGINEERING

in

APPLIED ELECTRONICS

Department of Electronics and Communication Engineering

KUMARAGURU COLLEGE OF TECHNOLOGY

(An autonomous institution affiliated to Anna University, Chennai)

COIMBATORE - 641 049

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report titled **"IMPROVED FAULT DIAGNOSIS IN ANALOG CIRCUITS USING HYBRID NEURAL NETWORK AND EVOLUTIONARY TECHNIQUES"** is the bonafide work of **DEEPAK B [Reg. No. 13MAE02]** who carried out the research under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Ms. SHANTHI M,

PROJECT SUPERVISOR

Department of ECE

Kumaraguru College of Technology

Coimbatore-641 049

SIGNATURE

Dr. RAJESWARI MARIAPPAN

HEAD OF THE DEPARTMENT

Department of ECE

Kumaraguru College of Technology

Coimbatore-641 049

The Candidate with university **Register No. 13MAE02** was examined by us in the project viva –voice examination held on...........................

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

My very first gratitude goes to God Almighty for giving me life, strength and the enablement to carry out this work.

I wish to express my deep sense of gratitude to **Dr.R.S.Kumar PhD,** Principal, Kumaraguru College of Technology, Coimbatore, for providing the facilities to conduct this study.

I express my humble gratitude to **Dr.Rajeswari Mariappan PhD**, Head of the Department, Electronics and Communication Engineering, Kumaraguru College of Technology, Coimbatore, for facilitating conditions for carrying out the research work smoothly.

I wish to express my sincere thanks to **Mrs.S.Sasikala M.tech**, Associate Professor, PG Project Co-ordinator, Department of Electronics and Communication Engineering, Kumaraguru College of Technology, Coimbatore, for her valuable suggestions.

I am indeed grateful to my project guide, **Ms.M.Shanthi M.S,** Associate Professor, Department of Electronics and Communication Engineering, Kumaraguru College of Technology, Coimbatore, for her immense contribution, guidance, support and constructive criticism not only during this project but also during this two years of my master program.

Lastly, I would like to thank my friends and family for standing by me and encouraging me throughout the project.

ABSTRACT

System on Chip (SOC) having both digital and analog circuits has become increasingly prevalent in integrated circuit manufacturing industry. Diagnosis of analog faults are indeed challenging as well as interesting and motivating behind the project work. With better testing methodology, large analog circuit design is also feasible for low cost. This report proposes a new transfer function based component level fault diagnosis methodology for analog circuits using artificial intelligence technique. Neural Networks are one of analytical tools that can be used for fault classification. Architecture selection for a neural network depends on various factors such as selection of the optimal number of hidden nodes, selection of the relevant input variables and selection of optimal connection weights. This report presents hybridization model that combines Genetic Algorithm (GA) and Back Propagation network (BPN) where GA is used to initialize and optimize the connection weights of BPN. State variable filter and Sallen–Key Band pass filter are the circuits used as the circuit under test. Gain, Quality factor and Frequency are the parameters used to spot the faults in the filter circuits. The fault dictionary is generated with fault and fault free conditions. With the faulty input the neural network is trained and detects the fault. The results prove that, GA-optimized BPN approach has outperformed the BPN approach without GA optimization.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ACRONYMS	ABBREVIATIONS
AMS	Analog and Mixed Signals
ANFIS	Adaptive Neuro-Fuzzy Inference System
ANN	Artificial Neural Network
BP	Back Propagation
BPN	Back Propagation Network
BP-NN	Back Propagation Neural Network
CUT	Circuit Under Test
ELM	Extreme Learning Model
GA	Genetic Algorithm
GA-BP	Genetic Algorithm – Back Propagation
HMM	Hidden Markov Model
IC	Integrated Circuits

LNA	Low Noise Amplifier
LPO	Low Pass Output
LS-SVM	Least Square Support Vector Machine
MSE	Mean Square Error
NN	Neural Network
PDF	Probability Density Function
SAT	Simulation After Test
SBT	Simulation Before Test
SKBPF	Sallen-Key Band Pass Filter
SVF	State Variable Filter
SVM	Support Vector Machine
VCVS	Voltage-Controlled Voltage-Source

CHAPTER 1

INTRODUCTION

Electronic tests are system dependent and are classified as digital, analog and mixed signal. Current methodologies for the testing of digital circuits are well developed. Some digital testing are D-Algorithm, Level sensitive scan design [LSSD], Build-in logic block observer [BILBO]. Automatic testing tools software also available for digital which automatically generate a test analysis for digital circuits. Some automatic testing software for digital are HP quick test professional, selenium, IBM Rational functional tester, silk test, win runner, WATIR, etc.

By contrast, methodologies for the testing of analog circuits remain relatively under developed due to complex nature of analog signals [1]. Despite the translation of many analog electronic functions into their digital equivalents, there still exists a need to incorporate analog sections on many chips. Therefore, the importance of analog testing cannot be underrated and there is a requirement to develop strategy, which will allow the analog and digital parts of the circuit to be tested simultaneously.

Analog and mixed-signal (AMS) integrated Circuits (IC) are gaining popularity in applications such as:

- Consumer electronics
- Biomedical equipments
- Wireless communication
- Networking
- Multimedia
- Automotive process control
- Real-time control system

With such wide applications, AMS ICs will constitute the bulk of future electronic devices, making it imperative to research AMS testing. New analog testing and fault diagnosis methodologies need to be compatible with existing digital test methods and be practical in compromising and test overhead.

1.1. FAULT DIAGNOSIS IN ANALOG CIRCUITS

Defects of analog circuits during the manufacturing process are caused by environmental defects and process variation or technological process inappropriateness.

There are certain classes of faults that occur and develop at slower rate in a circuit which is called as soft faults or parametric faults and these soft faults still continue to operate at poor effectiveness levels [2]. They do not change the circuit topology but cause the circuit to operate outside its allowable range of operation. Soft faults are due to distinction of one or more circuit component values outside the tolerance range. Performance of the circuit degraded when the tolerance range is exceeded. Soft fault recognition and identification of electronic circuits has been an active research topic in modern years. Fault diagnosis is often considered to be a two-stage process: fault detection and fault identification [3]. The Simulation Before Test approach (SBT) and the Simulation After Test approach (SAT) are two major approaches for fault diagnosis. For the SAT approach, fault analysis is obtained by analyzing the circuit components from the measured responses of the CUT [4]. For the SBT approach, compares the circuit responses associated with the predefined fault values in the dictionary to locate the faults. This SBT approach ensures a short test time even for complex circuits.

Diagnosis of the faults in analog circuits using the neural networks and the genetic algorithm were developed based on phenomena found in nature. Both of them have been widely used to solve a variety of computationally intensive problems. When combined they yield advantages over many conventional approaches. The purpose of this work is to apply a genetic algorithm to determine the optimal connection weights of an arbitrary neural network.

1.2. ARTIFICIAL NEURAL NETWORK

The concept of an artificial neural network (ANN) is based on the human brain. Human brain is made up of billions or millions of neurons which are interconnected by synapses [5]. Similarly, an artificial neural network is composed of many computational units which are also called neurons. The interconnections of the neurons dictate the characteristics of both a neural network and a brain. ANN has three major advantages: the ability to learn, parallelism and the ability to generalize.

The human brain is highly parallel in nature since each neuron may communicate with several others concurrently. The parallelism is perfect for complicated and complex tasks such as data pattern classification, which would be much more complicated to realize in a serial fashion. Neural networks are arranged in a similar fashion, and are therefore ideal for pattern classification and similar applications which can exploit parallelism.

Another major advantage of neural networks is their ability to learn by training. Training consists of supplying the neural network with many training samples, each of which consists of a set of inputs and the desired set of outputs. The back-propagation algorithm is a popular method used to train neural networks. Through an iterative learning process the synaptic weights are modified and eventually the neural network is trained to produce outputs close to those desired.

Neural networks can be trained to solve different types of application problems, such as pattern classification, function approximation, filtering, controlling, etc. Since only training samples are mandatory, the real relationship connecting the inputs and outputs does not require to be known. This is a benefit for many applications, particularly when the input/output relationship is extremely complex. Moreover, neural networks are capable to simplify and generate the exact results for inputs which are not found in the training sample set.

Before a neural network can be trained, topology (how they are connected), its size (how many neurons), learning rate (the speed of the back-propagation algorithm) and several other parameters have to be selected. Normally, more complicated functions require for larger neural networks consisting of more neurons and more synapses than those required for simpler functions. Major struggle can happen if the parameters are not selected correctly: over fitting and unacceptable error.

If a neural network is too small for a given application, it may never be able to learn the desired function and thus produce an unacceptably high error. An unacceptable error may also occur if the learning rate of the training algorithm is selected incorrectly.

Finally, if a neural network is too large for a particular problem, it may learn the training samples too fine and not be able to simplify to inputs outside the training data set. Selecting the suitable neural network parameters is more of an art than a science and usually turns into a trial-and-error.

1.3. GENETIC ALGORITHM

The genetic algorithm (GA) also has its roots in nature, and is based on Charles Darwin's theory of natural selection [6]. In Darwin's theory, individuals in a population of reproductive organisms inherit traits from their parents during each generation. Each individual's genome represents one's phenotype (i.e., the physical characteristics), and is made up of many genes (the actual genetic makeup). Over time, desirable traits become more common than undesirable ones since individuals with desirable traits are more likely to reproduce or replicate. The GA follows natural selection to a certain extent closely.

In the genetic algorithm, each gene is usually represented as a binary number which is encoded to represent several phenotypes. A population of individuals is originally created with all of their genotypes randomly selected, with each individual representing a potential solution to the problem. Once the initial population is created, it is sorted by fitness. The algorithm designer can choose how the fitness value is calculated, with a higher fitness value representing a better solution. During each generation, the parents (two individuals) are randomly selected to reproduce, with the more fit individuals getting selected. The chromosomes of the two parents are combined to create a new offspring in a process known as crossover.

During crossover, the offspring's genotype is created by combining the genes of its parents in a random mode. After crossover, some of the bits in the offspring's genes may be flipped at random in the process of mutation, which also occurs in natural world. At last, the offspring's fitness value is evaluated. If the offspring's fitness is better than the worst parent (individual) currently in the population, then the new offspring replaces that individual. The population repeatedly improves its overall fitness for the duration of each generation until finally the peak individual is optimized to an acceptable level.

With today's extremely fast computer processors, running genetic algorithms for hundreds of generations is possible in a reasonable amount of time (on the order of minutes, hours, or days).

1.4. OVERVIEW OF THE REPORT

This report is organized as follows. Chapter 2 discusses about the literature survey. Detailed descriptions of the fault diagnosis methods are found in chapter 3. Chapter 4 outlines some necessary background information including a description of the neural networks and the genetic algorithm. Chapter 5 briefly describe about the circuit under test (CUT). The proposed BP-NN model and hybrid GA-BP model is discussed in chapter 6. Chapter 7 shows the simulation results of the project. The final conclusions and future work are discussed in chapter 8.

CHAPTER 2

LITERATURE SURVEY

2.1. Analog Circuits Fault Detection Using Cross-Entropy Approach (27 Jan 2013).

This paper presents a novel method that can detect component faults in analog circuits. Because the probability density function (PDF) of output voltage (current) is sensitive to the components of the circuit, the cross-entropy between the good circuit and the bad circuit is employed to detect component faults in analog circuits based on the autoregressive (AR) model [7]. In the proposed approach, the value of each component of the circuit under test (CUT) is varied within its tolerance limit using Monte Carlo simulation. The minimal and maximal bounds of the cross-entropy are found for fault-free circuit. While testing, the cross-entropy is obtained. If cross-entropy lies outside the tolerance limit then the CUT is declared faulty. The effectiveness of the proposed method is demonstrated via the second order Sallen-key band pass filter circuit and continuous-time low pass state-variable filter circuit.

2.2. Diagnosis of Incipient Faults in Weak Nonlinear Analog Circuits (13 April 2013).

Aiming at the problem to diagnose incipient faults in weak nonlinear analog circuits, an approach is presented in this paper. The approach calculates the fractional Volterra correlation functions beforehand [8]. The next step is to use the fractional Volterra correlation functions and different angle parameters of the fractional wavelet packet transform to extract the fault signatures. Meanwhile, the computational complexity is analyzed. Then the variables of the fault signatures are constructed, which are used to form the observation sequences of the hidden Markov model (HMM). HMM is used to accomplish the fault diagnosis. The simulations show that the presented method can significantly improve the incipient fault diagnosis capability.

2.3. Diagnosis of Local Spot Defects in Analog Circuits (October 2012).

In this paper a method for diagnosing local spot defects in analog circuits. The method aims to identify a subset of defects that are likely to have occurred and suggests giving them priority in a classical failure analysis [9]. For this purpose, the method relies on a combination of multiclass classifiers that are trained using data from fault simulation. The method is demonstrated on an industrial large-scale case study. The device under consideration is a controller area network transceiver used in automobile systems. This device demands high-quality control due to the reliability requirements of the application wherein it is deployed. The diagnosis problem is discussed by taking into consideration the realities of this case study.

2.4. A Neuro-Fuzzy Inference System through Integration of Fuzzy Logic and Extreme Learning Machines (October 2007).

This paper investigates the feasibility of applying a relatively novel neural network technique i.e., extreme learning machine (ELM), to realize a neuro-fuzzy Takagi–Sugeno–Kang fuzzy inference system [10]. The proposed method is an improved version of the regular neuro-fuzzy Takagi–Sugeno–Kang fuzzy inference system. For the proposed method, first, the data that are processed are grouped by the *k*-means clustering method. The membership of arbitrary input for each fuzzy rule is then derived through an ELM, followed by a normalization method. At the same time, the consequent part of the fuzzy rules is obtained by multiple ELMs. At last, the approximate prediction value is determined by a weight computation scheme. For the ELM-based Takagi–Sugeno–Kang fuzzy inference system, two extensions are also proposed to improve its accuracy. The proposed methods can avoid the curse of dimensionality that is encountered in back-propagation and hybrid adaptive neuro-fuzzy inference system (ANFIS) methods.

2.5. A New Analog Circuit Fault Diagnosis Method Based on Improved Mahalanobis Distance (21 December 2012).

This paper presents a new analog circuit fault diagnosis method based on improved Mahalanobis Distance. The Mahalanobis Distance is improved according to the characteristics of analog circuit, and then introduced into analog circuit fault detection [11]. First, the circuit testability was analyzed, and the relation of ambiguity groups was determined on the basis of the test matrix, and then the separable potential faulty components under the assumption of single fault were also determined. Finally, the suspicious components could be classified using the improved Mahalanobis Distance according to the feature values of the test points, so as to reduce the number of classes and enhance the speed when classifying faults.

2.6. Soft Fault Classification of Analog Circuits Using Network Parameters and Neural Networks (11 April 2013).

A new method to identify component faults in analog circuits is proposed using network parameters like driving point impedance, transfer impedance, voltage gain and current gain [12]. Using Monte-Carlo simulation each component of the circuit is varied within its tolerance limit and samples of each network parameter are found for fault free circuit. Similarly all possible single faults are introduced and the corresponding samples of network parameters are found. Fault classification is done through neural network. The proposed method is validated through second order Sallen-key band pass filter. Numerical results are presented to clarify the proposed method and prove its efficiency.

2.7. Analog Circuit Fault Detection Using Location of Poles (11 August 2011).

A method for detection of parametric faults occurring in linear analog circuits based on location of poles of the Circuit Under Test (CUT) is proposed. In the proposed method, the value of each component of the CUT is varied within its tolerance limit using monte-carlo simulation [13]. The upper and lower bounds of magnitude, phase angle, real part and imaginary part of all poles of the CUT are obtained. While testing, the locations of poles are obtained. If any one or more of the poles lies outside the tolerance limit then the CUT is declared faulty. The effectiveness of the proposed method is validated through two benchmark circuits like second order sallen-key band pass filter and fourth order leapfrog low pass filter.

2.8. Parametric Fault Testing of Non-Linear Analog Circuits Based on Polynomial and V-Transform Coefficients (28 August 2012).

This paper is an exposition of recent advances made in polynomial coefficient and V-transform coefficient based testing of parametric faults in linear and non-linear analog circuits. V-transform is a nonlinear transform that increases the sensitivity of polynomial coefficients with respect to circuit component variations by three to five times [14]. In addition, it makes the original polynomial coefficients monotonic. Using simulation, the proposed test method is shown to uncover most parametric faults in the range of 5–15 % on a low noise amplifier (LNA) and an elliptic filter benchmark. Diagnosis of parametric faults clearly illustrates the effect of enhanced sensitivity through V-transform. Finally, we report an experimental validation of the polynomial coefficient based test scheme, with and without V-transform, using the National Instruments' ELVIS bench-top test-bed. The result demonstrates the benefit of V-transform.

2.9. An Approximate Calculation of Ratio of Normal Variables and Its Application in Analog Circuit Fault Diagnosis (14 June 2013).

The challenging tolerance problem in fault diagnosis of analog circuit remains unsolved. To diagnose the soft-fault with tolerance effectively, a novel diagnosis approach based on the ratio of normal variables and the slope fault model was proposed. Firstly, the approximate distribution function of the ratio of normal variables was deduced and the basic approximate conditions were given to improve the approximation accuracy [15]. The conditional monotonous and continuous mapping between the ratio of normal variables and the standard normal variable was proved. Based on the aforementioned proved mapping, the estimation formulas of the range of the ratio of normal variables were deduced. Then, the principle of the slope fault model for linear analog circuit was presented. After the contrastive analysis of the typical methods of handling tolerance based on the slope fault model, the ratio of normal variables and the slope fault model were combined and a test-nodes selection algorithm based on the basic approximate conditions of ratio of normal variables was designed, by which the computation can be reduced greatly.

2.10. Fault Diagnosis of Analog Circuits Using Systematic Tests Based on Data Fusion (19 September 2012).

An analog fault diagnosis approach using a systematic step-by-step test is proposed for fault detection and location in analog circuits with component tolerance and limited accessible node [16]. First, by considering soft faults and component tolerance, statistics-based fault detection criteria are established to determine whether a circuit is faulty by measuring accessible node voltages. For a faulty circuit, fuzzy fault verification is performed using the accessible node voltages. Furthermore, using an approximation technique, the most likely faulty elements are identified with a limited number of circuit gain measurements at selected frequencies. Finally, employing the D-S evidence theory, synthetic decision is made to locate faults according to the results of fault verification and estimation. Unlike other methods which use a single diagnosis method or a particular type of measurement information, the proposed approach makes use of the redundancy of different types of measurement information and the combined use of different diagnosis methods so as to improve diagnosis accuracy.

2.11. Diagnostics of Analog Circuits Based on LS-SVM Using Time-Domain Features (29 May 2013).

Most researchers use wavelet transforms to extract features from a time domain transient response from analog circuits to train classifiers such as neural networks (NNs) and support vector machines (SVMs) for analog circuit diagnostics [17]. In this paper, we have proposed some new feature selection methods from a time-domain transient response, and compared the diagnostic results based on a least squares SVM (LS-SVM) using different time-domain feature vectors. First, we have improved two traditional feature selection methods: (a) using the mean and standard deviation in wavelet transforms features, and (b) using the mean, standard deviation, skewness, kurtosis, and entropy in statistical property features. Then, a conventional time-domain feature vector based on the impulse response properties of a control system has been proposed. The simulation experiments for a leapfrog filter and a nonlinear rectifier show that: (i) the two improved methods have better accuracy than the traditional methods; (ii) the proposed conventional time-domain feature vector is effective in the diagnostics of analog circuits—over 99 % for both of the two example circuits; (iii) the proposed diagnostic method can diagnose soft faults, hard faults, and multi-faults, regardless of component tolerances and nonlinearity effects.

CHAPTER 3

FAULT DIAGNOSIS METHODOLOGY

3.1. TESTING OF ANALOG

Testing techniques are available in past three decades for digital circuitry. The main reason for this is the ease of formulating the test generation as a mathematical problem due to the discrete signal and time values [18]. The distinction between what does and what does not work is crisp and clear for digital circuitry.

For analog circuitry, generation of optimal test signals based on design topology is still not fully automated. As opposed to the digital approach based on the gate-level net-list, analog testing still relies mainly on a black-box approach. A similar test generation solution for analog testing became necessary with the increasing integration of analog and digital functionality on one chip. The analog test community has also been aiming at a solution comparable to that in digital, but the analog version of the problem is not solvable by similar analytical techniques. The main reasons for this are:

- In Digital values (0 & 1)'s but in analog infinite number of signal values are possible.
- The time variation properties of analog signals bring an extra dimension to the problem.
- One-to-one link is possible in Digital but not possible in analog circuits. Given particular topology, there is no general way of determining which part of the functionality is of interest and what the related performance limits.
- Propagation of fault effect to the output is not simple as digital because of two reason:
  - ➢ Fault cannot be modeled in one direction as digital. Fault effect propagates in all directions and calculation of propagation pattern becomes more complex.
  - ➢ In analog, the information that fault present in certain nodes does not readily comprise signal value information for that node, making time consuming calculations of signal values are necessary.

The obstacles presented above have prevented analog test generation from being applied in practice. Unfortunately, a satisfying solution to the analog problem has not been found to this day.

The alternative of solving the analog test generation is based on two methods

1. Specification based testing – checking whether the specification are met.
2. Functional testing – checking the functioning of the circuit with a standard input.

3.2. TEST LEVELS

Test can be performed at several levels:

1. Wafer level
2. Package level
3. Module level
4. System level
5. Field level

Test can be classified into three methods,

1. Fault detection
2. Fault location
3. fault prediction

3.3. APPROACH

There are different approaches for fault diagnosis methodology.

1. Fault verification approach
2. Parameter identification approach
3. Fault dictionary approach
4. Approximation approach

The current (fault dictionary) approach is to detect manufacturing faults in analog electronic circuits using functional tester.

3.4. FAULT DIAGNOSIS METHOD

Analog fault diagnosis methods are generally classified into two methods

1. Simulation-after-test (SAT)
2. Simulation-before-test (SBT)

3.4.1. Simulation-After-Test (SAT)

SAT methods focus on parameter identification and fault verification and they are very efficient for soft fault diagnosis because they are based on linear network models.

a) Parameter Identification

Parameter identification technique is to formulate sufficient number of independent equations from the measurements to determine all component values. A component value that lies outside the design tolerance range specification is identified as a faulty component. Saeks et al [19] proposed a method to determine parameter values using voltage and current measurement when a single excitation is applied. If multiple current excitation are applied to a network and voltage measurements are used to identify network parameters is proposed by Biernacki and Bandler [20]. Above methods is deal with DC domain or single frequency excitation. The multi-frequency techniques include research on the test point selection and test frequency selection. Their measure provides more information on the degree of difficulties about the testability. The time domain approach includes formulating testable equations, which are solvable from time domain measurements.

b) Fault Verification

The fault verification methods use almost the same equations as are used in the parameter identification approaches, except that in the fault verification approaches, circuit components are partitioned into two classes, a fault-free class (class1) and faulty class (class2). It is assumed that all components in class1 are fault free and all faults are localised in class2. Using the measurement data and nominal characteristics of all circuit components, test equations are formulated and expressed as functions of deviations of class2 components. Test equations can be satisfied only if all faults are indeed localised in class2.

No matter what kind of testability measures are used, whether it is based on frequency domain, time domain or topology, the advantage is that the measure tells whether the CUT is testable or diagnosable. However, the computational complexity is a difficult problem to overcome. In manufacturing testing, this problem becomes more severe. SBT methods provide a compromise by shifting the computational burden to simulation before test.

3.4.2. Simulation-Before-Test (SBT)

c) Fault Dictionary

In this report, the fault detection of single element component failure is drawn and used to generate the fault dictionary. Different types of measurements were used in the literature to construct the dictionary. The wide used measurements are node voltage, magnitude and phase of node voltages, voltage/current measurements. Power supply current and voltage measurements are also used in linear bipolar ICs for fault detection.

Artificial intelligence and neural network methods have been widely used in analog fault diagnosis, especially in SBT methodologies

A fault dictionary constructs a look up table, which lists each faulty case and corresponding nominal case for comparison purpose. The objectives of fault detection must be clear because they are critical aspects for deciding the fault detection or diagnosis capability of the dictionary. They also have an impact on the size of the dictionary and impose a limitation on the dictionary approach. Too broad fault coverage may end up with prohibitively large number of combinations which may not be realisable algorithmically, while too narrow fault coverage may not meet the quality target. The anticipated faults and nominal circuit of the CUT need to be simulated in order to develop sets of stimuli and response to detect and isolate faults. To generate a reasonable fault list, physical failures and failure modes have to be related and suitable fault models have to be developed.

3.5. ANALOG FAULT MODELLING

Faults in analog are generally classified in to the following two categories

i. Catastrophic faults

It is also known as hard faults. Catastrophic faults are all those changes to the circuit that cause the circuit to fail catastrophically. These faults include shorts, open or large variations of the design parameters.

ii. Parametric faults

It is also known as soft faults. Parametric faults are those changes that cause performance degradation of the circuit. These faults are due to the process fluctuations. These faults involve parameters deviations from their nominal value that can consequently quit their tolerance band.

3.6. FAULT DIAGNOSIS PROCEDURE

The general idea is to test the response of the given circuit. Deviations in circuit parameters caused by any fault will affect the output response, either in its amplitude or phase.

One problem central to testing is the determination of an optimal test pattern fulfilling the following essential requirements:

— Detection of (ideally) all defects assumed in the fault model,

— Ease of the generation/storage (low overhead) and

— Compactness (short test stimulus generation time).

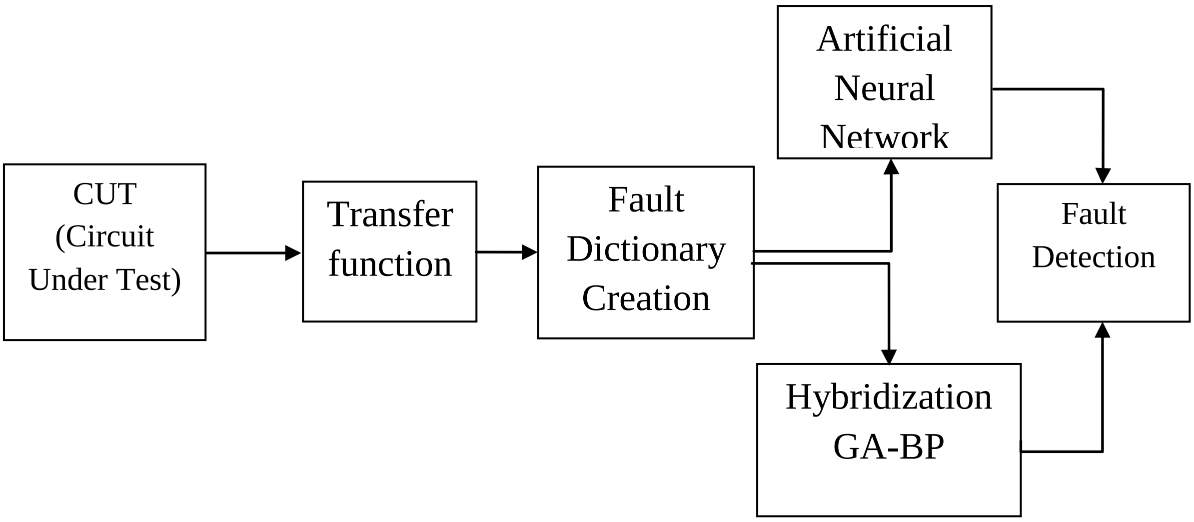


Fig.3.1. Architecture of a Typical Fault Diagnosis

In this report, two filter circuit's State variable filter and sallen-key band pass filter are consider as circuit under test. The circuit is mathematically represented using transfer function. The transfer function contains the specified parameters such as gain (K), pole selectivity (Q) and frequency (f) of the circuit. Fault dictionary stores the possible faults injected in the component values lying outside their nominal range. Artificial Neural Network (ANN) is used as a pattern classifier with its training and testing phase to detect the fault in the circuits.

Hybridization of genetic algorithm and back propagation neural network (GA-BP) is done for improving the fault detection accuracy. GA is used to obtain optimum connecting weights and biases and then ANN is trained with optimum connecting weights and biases so that mean square value (MSE) is reduced.

CHAPTER 4

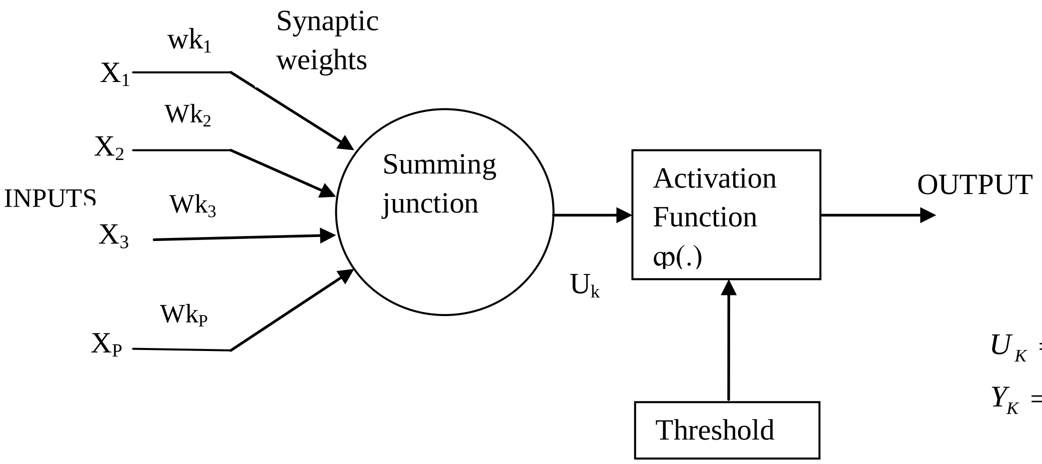
AN OVERVIEW OF ARTIFICIAL NEURAL NETWORK AND GENETIC ALGORITHM

4.1. ARTIFICIAL NEURAL NETWORK BASICS

Neural networks are easier to understand if they are broken down into their core components. This section explains the basics of neural networks. First, the nature of the neuron is explored (the elementary unit of a neural network). Next, the different ways in which neurons can be connected are shown [21]. Finally, neural network training (the way in which a neural network learns) is examined.

4.1.1. The Neuron

Neural networks (NN) are made from as few as one to as many as hundreds of elementary units called neurons. A neuron produces an output from its arbitrary inputs [22]. As shown in figure 4.1, each neuron is made up of the following: inputs, synaptic weights, a bias, a summing junction, a local induced field, an activation function, and a single output.



$$U_K = \sum W_K * X_J$$
$$Y_K = \sum \phi(U_K - \theta_K)$$

Fig.4.1. Model of an Artificial Neuron

4.1.2. Network Architecture

The output of a neuron can be connected to the input of another neuron. In fact, one neuron’s output can be connected to any number of other neurons’ inputs, allowing numerous possible ways of combining neurons to form a neural network. Neural networks are commonly organized in a layered fashion, in which neurons are organized in the form of layers. There are three kinds of layers: input layer, hidden layers, and output layer. The input layer is made up of the input nodes of the neural network. The output layer consists of neurons which produce the outputs of the network. All layers which do not produce outputs, but instead produce intermediate signals used as inputs to other neurons, are considered hidden layers.

Figure 4.2 shows a simple neural network, which consists of an input layer and an output layer (circles represent neurons and arrows represent synapses). In this network, three neurons process the three inputs to produce three outputs.

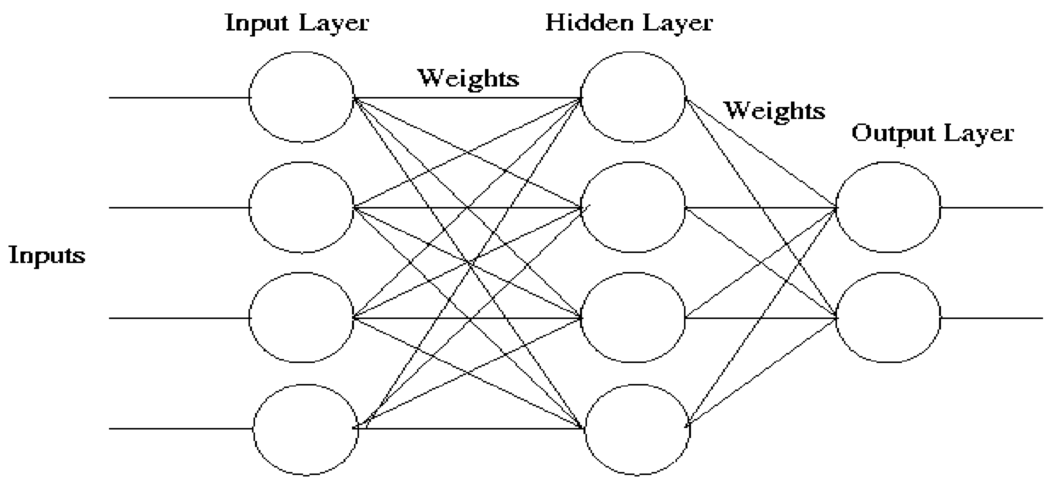


Fig.4.2. Simple NN Architecture

Figure 4.3 shows a multi-layer network. This network consists of all three layers: an input, hidden, and output layer. When designing the architecture of a neural network, there is no limit on the number of layers or the number of neurons within each of those layers. Some complex tasks require architectures that contain multiple hidden layers.

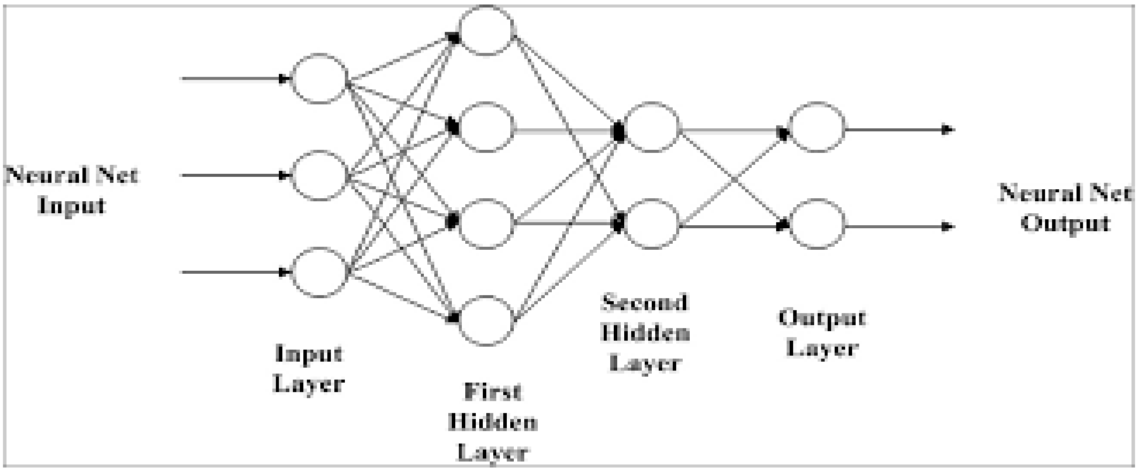


Fig.4.3. Multi-Hidden Layer NN

4.1.3. Training

Neural networks can be trained in several different ways, the most common of which is called the back-propagation algorithm. In neural network training, error represents the difference between the produced outputs and the desired outputs. In order to train a network, it must be trained with a set of training samples (training set). Each training sample in the training set is made up of a set of inputs and the desired set of outputs. The training set should be a representative collection of input/output samples (all possible samples if available).

Initially, all synaptic weights are chosen at random throughout a neural network. Back-propagation learning is an iterative process, which modifies the synaptic weights to minimize error in the each training iteration. During the each iteration, the input set (from the training sample) is fed into the network. The produced outputs are compared to the expected outputs (from the training sample) and the error is computed. The error is used to modify the weights throughout the network in order to bring the outputs closer to their expected values.

Depending on the complexity of the application, a neural network requires many epochs of training before it produces an acceptably low error. An epoch consists of training a network with the entire training set.

A set of samples, sometimes referred to as the test set, is used to determine the mean squared error (MSE). If possible, test set should contain a different set of samples than the training set in order to test the network's ability to generalize. Generalization is the ability of a neural network to produce accurate results for inputs not found in the training set. The mean squared error for a single sample (ε_j) is calculated as shown in equation 4.1, where n is the number of outputs. The MSE for the entire evaluation set (E_{MSE}) is an average of the individual sample MSE values, as shown in equation 4.2, where m is the total number of samples in the evaluation set and j is the index.

$$Error, \varepsilon_j = \frac{\frac{1}{2}\sum_{i=1}^{n}(desiredoutput_i - actualoutput_i)^2}{n} \tag{4.1}$$

$$E_{mse} = \frac{\sum_{j=1}^{m}(\varepsilon_j)}{m} \tag{4.2}$$

The total number of training epochs can vary, and usually relates with the maximum allowed E_{MSE}. The training continues until E_{MSE} has reached an acceptable value. However, networks can sometimes reach a local minimum in the error space (as opposed to the global minimum). In this case, the error cannot be lowered any further and the neural network is unacceptable. If a neural network has reached a local minimum, the only possible recourse is to restart the process with a new neural network that either has a different architecture, new randomly selected weights, or both.

4.1.4. Back Propagation Algorithm

The back-propagation algorithm (BP) is used to modify the synaptic weights throughout a neural network in order to minimize error. It is an iterative process, which modifies the network one training sample at a time. During the each iteration the error signal travels backwards through the network, starting at the output neurons and ending at the input synapses.

The correction for a weight is defined as

$$\Delta w_{j,i} = \eta \delta_j y_j \tag{4.3}$$

Where η is the learning rate, $w_{j,i}$ is the change in weight connecting neuron i in layer L to neuron j in layer L+1, δ_j is the local gradient, and y_i is the output of neuron i. The learning rate affects how much a weight will change based on the error and can be chosen to be any real number. The local gradient is the error signal that travels backwards through the network and is based on activation function, as well as whether the neuron in question is an output and non-output neuron. Only the correction functions for sigmoid and tanh activation functions are shown, as they are the only functions used in this report.

For output neurons using the sigmoid activation function

$$\delta_j = a(d_j - o_j)o(1 - o_j) \tag{4.4}$$

Where a is the sigmoid parameter, d_j is desired output, and o_j is the actual output. The correction function for output neurons using the sigmoid activation function is therefore

$$\Delta w_{j,i} = a\eta(d_j - o_j)o_j(1 - o_j)y_i \tag{4.5}$$

For non-output neurons using the sigmoid activation function

$$\delta_j = ay_j(1 - y_j)\sum_k \delta_k w_{k,j} \tag{4.6}$$

Where *yj* is the output of neuron δ $_{j,k}$ is the local gradient of neuron k in the next layer, and *k j w*, is the weight connecting neuron j with each of the neurons in the next layer. The correction function for non-output neurons using the sigmoid activation function is

$$\Delta w_{j,i} = a\eta y_j(1 - y_j)y_i \sum_k \delta_k w_{k,j} \tag{4.7}$$

For output neurons using the tanh activation function

$$\delta_j = a(d_j - o_j)(1 - o_j)(1 + o_j) \tag{4.8}$$

Where a is the tanh slope. The correction function for output neurons using the tanh activation function is

$$\Delta w_{j,i} = a\eta(d_j - o_j)(1 - o_j)(1 + o_j)y_i \tag{4.9}$$

For non-output neurons using the tanh activation function.

The correction function for non-output neurons using the sigmoid activation function is

$$\delta_j = a(1 - y_j)(1 + y_j)\sum_k \delta_k w_{k,j} \tag{4.10}$$

Therefore

$$\Delta w_{j,i} = a\eta(1 - y_j)(1 + y_j)y_i \sum_k \delta_k w_{k,j} \tag{4.11}$$

4.2. GENETIC ALGORITHM

The genetic algorithm (GA) is based on Charles Darwin's theory of evolution and can be used to solve a wide variety of problems. In general, a GA is defined by an iterative process, comprised of six key stages: generating initial population, evaluation, ranking, selection, crossover, and mutation [23]. This iterative process resembles what occurs when organisms reproduce in nature. The first step necessary to understand the GA is to learn about the overall structure and terminology of the GA. The next step is to learn about each of the six stages, some of which require a more detailed explanation than others.

4.2.1. Structure and Terminology

Each genetic algorithm can differ, but all genetic algorithms have a few things in common. General structure of GA is shown in fig.3.4. Each GA has a population certain size, which is made up of individuals [24]. Each individual represents a potential solution to the problem the GA is trying to solve.

The first step in GA involves randomly generating the initial population. Each individual in the population (plus the newly created individual starting in the second iteration) is given a fitness value in the evaluation step. All of the individuals are then ranked based on their fitness. In the selection step, two individual are selected to reproduce based on their rank. The two selected individuals' genotypes are used to produce a new individual in the crossover step. Finally, the new individual's genes may be mutated (bits are randomly flipped) and the next iteration begins.

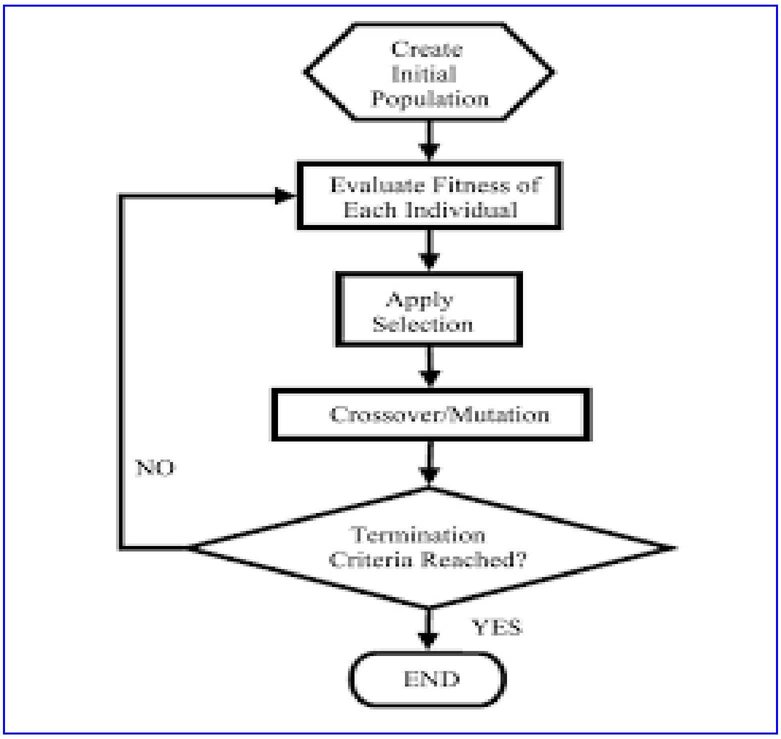


Fig.4.4. Flow Diagram of Genetic Algorithm

4.2.2. Initial Population and Fitness Evaluation

The basis of the GA is that the population of a fixed size is improving with the each iteration of the algorithm. An initial population of size *P* is created by generating P individuals with randomly selected genes.

After the initial population has been generated, the population must be evaluated. Evaluation is one of the most important steps in the GA process, since it defines what constitutes an individual as ‘fit’. At the end of the evaluation, each individual will have a fitness value. The method for calculating the fitness of an individual is completely up to the GA designer, with the requirement that an individual with higher fitness represent a better solution than an individual with lower fitness.

The evaluation step also occurs after a new individual has been generated, and is used to maintain the population at a constant size *P*. Like in the theory of evolution, the more fit individuals survive: the least fit individual in the population is replaced if the new individual has a higher fitness value.

4.2.3. Crossover and Mutation

There are two commonly used ways of generating new individuals in GA: mutation and crossover. As shown in fig.4.4, the order and inclusion of mutation and crossover can be arbitrarily made by a GA designer. Mutation is the simpler of the two processes and is examined first.

In mutation, an arbitrary number of bits are flipped in an individual’s genome. The probability of a bit flipping, the selection of which bits will be flipped, and the number of bits to be flipped are all parameters that a GA designer can adjust to produce a better solution. If crossover is not used, then new individuals are generated by mutating a single individual selected in the selection stage. Mutation can be applied randomly as an independent operator on any individual in the population, or just to the offspring produced with crossover.

The more complex crossover process is commonly used in combination with mutation. In crossover, the genes of two selected parents are combined to produce an offspring, similar to the reproduction of organisms in nature. There are several commonly used forms of crossover: 1-point crossover, multi-point crossover, and uniform crossover.

CHAPTER 5

CIRCUIT UNDER TEST (CUT)

In this report, two circuits are considered for fault diagnosis method. First circuit is state variable filter (SVF) and another circuit is sallen key band pass filter (SKBPF).

5.1. STATE VARIABLE FILTER (SVF)

5.1.1. SVF Circuits

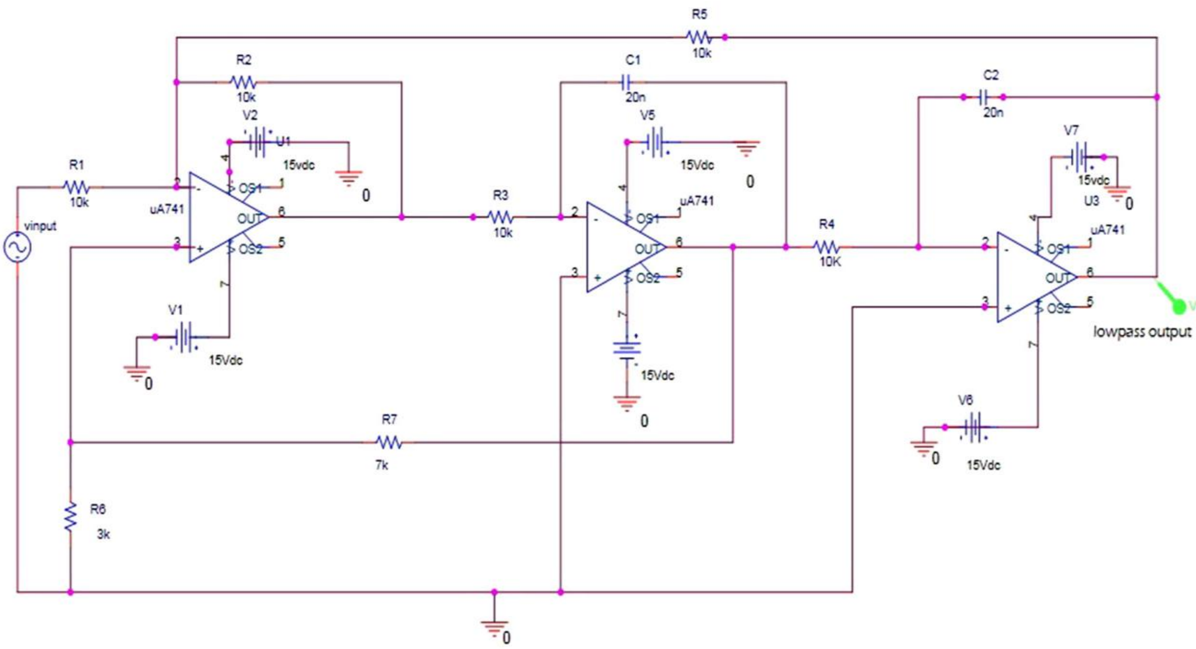


Fig.5.1. State Variable Filter Circuits

A state variable filter is a type of active filter. It consists of one or more integrators, connected in feedback configuration. Any Linear time invariant system can be described as a state-space model, with n state variables for an nth-order system. The low pass and high pass output's are phase inverted while the band pass output maintains the in phase. The gain of each output is independently variable. Due to temperature variation, component value may vary but must be in tolerance limit.

A state variable filter (SVF) can't be used as a different filters from a single set of hardware. A SVF is just an active filter circuit that incorporates operational amplifiers, capacitors and resistors to form an integrator and a feedback network to create resonance peaks for boosting or cutting certain frequencies.

The State-Variable filter offers low pass, high pass, band pass, and band-reject all within one block. An added advantage over bi-quad section filters is that only one coefficient is needed, rather than their five coefficients. For the algorithm including pole selectivity (Q), there's an additional coefficient for its control.

5.1.2. SVF Transfer Function

The nominal values of the circuit components [18] are:

R1 = R2 = R3 = R4 = R5 = 10kΩ;

R6 = 3kΩ;

R7 = 7kΩ;

C1 = C2 = 20nF.

All the parameters were assigned ±5% tolerance.

The voltage transfer function of the second-order SVF (Fig 5.1.), considering its low-pass output (LPO) is given by

$$\frac{\mathrm{VLPO}}{\mathrm{Vinput}} = \frac{-\mathrm{R5}}{\mathrm{R1}}\left[\frac{\frac{\mathrm{R2/R5}}{\mathrm{R3C1R4C2}}}{\mathrm{s}^2 + \frac{\left(1+\frac{\mathrm{R2}}{\mathrm{R5}}+\frac{\mathrm{R2}}{\mathrm{R1}}\right)\mathrm{s}}{\left(1+\frac{\mathrm{R7}}{\mathrm{R6}}\right)\mathrm{R3C1}} + \frac{\mathrm{R2/R5}}{\mathrm{R3C1R4C2}}}\right] \tag{5.1}$$

Comparing the equation.5.1 with second order low-pass filter transfer function, we get the following relations for k, $\dot{\omega}_0$ and Q.

$$\text{Gain,} \qquad \mathrm{K} = \frac{\mathrm{R5}}{\mathrm{R1}} \tag{5.2}$$

$$\text{Pole frequency,} \quad \dot{\omega}_0 = \sqrt{\frac{\mathrm{R2/R5}}{\mathrm{R3C1R4C2}}} \tag{5.3}$$

$$\text{Pole selectivity, } \mathrm{Q} = \sqrt{\left(\frac{R3C1}{R4C2}\right)\left(\frac{R2}{R5}\right)}\,\frac{1+\frac{R7}{R6}}{1+\frac{R2}{R5}+\frac{R2}{R1}} \tag{5.4}$$

Therefore for the LPO of filter with nominal values of the components yields k= 1.0, Q = 1.11 and f_o = 796HZ.

5.1.3. Fault Dictionary Creation

The procedure for fault dictionary generation is shown in fig 5.2. The transfer function is simulated with faults injected to the components [25]. The fault injection is done to the extent of ±50% deviation from nominal value with a step size of 5%. Single fault are introduced to one component at a time with other fault free components taking different random values within their tolerance.

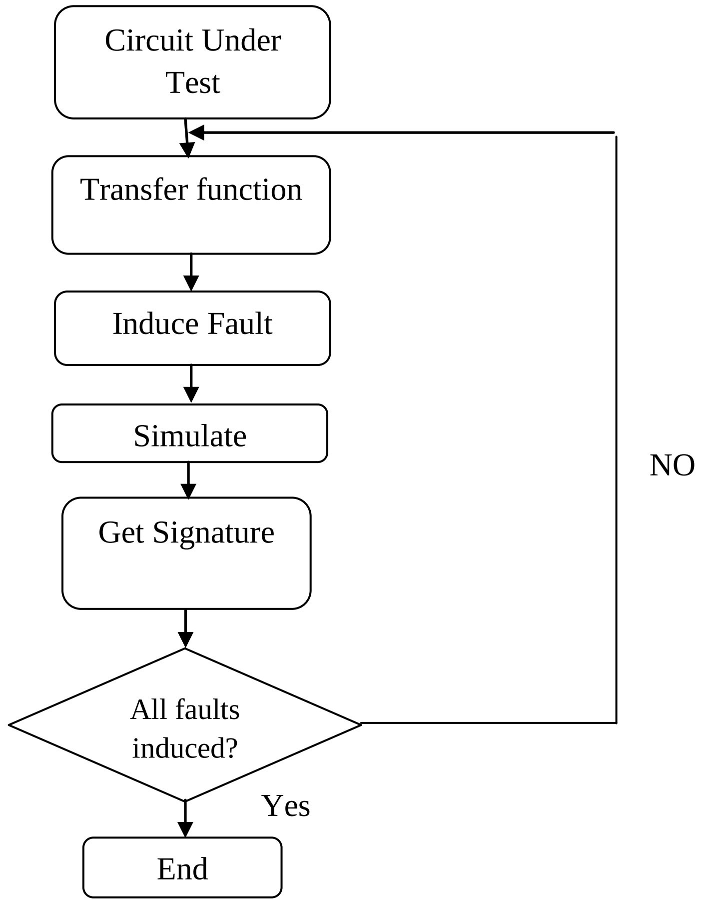


Fig.5.2. Fault Dictionary Generation-Flow Diagram

Table 5.1. Fault Dictionary samples

ANN Input				ANN Target
Fault injected in Component	Gain (k)	Pole selectivity (Q)	Pole frequency (f_o)	Fault detected
R1+15%	0.90941	1.1432048	796.2776	1
	0.909191	1.145129	795.1583	
	0.909389	1.143981	795.2185	
	0.906775	1.143891	798.3586	
	0.90769	1.144.32	796.4914	
	0.910507	1.147908	796.1246	
	0.909431	1.14824	735.1509	
	0.909603	1.146166	796.0002	
	0.909251	1.14508	796.2286	
	0.910388	1.143313	794.4788	
R1+20%	0.872328	1.164124	795.1364	
	0.872123	1.159479	794.6936	
	0.867851	1.159794	796.7802	
	0.869872	1.157294	796.8162	
	0.870619	1.169578	795.3414	
	0.870069	1.161836	795.8468	
	0.872095	1.162064	795.1361	
	0.869524	1.162294	795.1274	
	0.870863	1.156976	794.5657	
	0.834129	1.169535	796.9395	

Fault dictionary is generated injecting fault to all component and evaluating the parameter K, Q, f. The input sample of size 1854×4 is obtained. Fault dictionary are separated into train samples and test samples randomly. Samples of fault dictionary for component resistor R_1 with fault index 1 and capacitor C_2 with fault index 9 are shown in table 5.1, and table 5.2, to be used in ANN classifier for fault detection.

Table 5.2. Fault Dictionary samples

ANN Input				ANN Target
Fault injected in Component	Gain (k)	Pole selectivity (Q)	Pole frequency (f_o)	Fault detected
C2-15%	1.006933	1.169294	838.7723	9
	1.004368	1.173732	838.2972	
	1.004238	1.169415	838.1235	
	1.001462	1.172438	837.8346	
	0.996817	1.175384	839.3517	
	1.003823	1.176335	836.4943	
	1.003063	1.16539	839.0912	
	0.999961	1.173119	837.4293	
	1.000102	1.170809	840.5106	
	1.000922	1.175286	839.3955	
C2-20%	1.000335	1.204263	861.9353	
	0.996622	1.200059	863.0687	
	1.001686	1.202249	863.2027	
	0.997482	1.206427	863.0931	
	0.997623	1.206345	864.8744	
	0.999166	1.209003	864.2478	
	1.001493	1.200372	864.3618	
	1.001164	1.206592	863.6756	
	1.005795	1.201868	863.3667	
	1.000705	1.207985	863.6945	

5.2. SALLEN KEY BAND PASS FILTER (SKBPF)

5.2.1. SKBPF Circuit

The Sallen–Key filter used to implement second-order active filter that is particularly valued for its simplicity. It is a degenerate form of a voltage-controlled voltage-source (VCVS) filter topology. A VCVS filter uses a super-unity-gain voltage amplifier with practically infinite input impedance and zero output impedance to implement a 2-pole low-pass, high-pass, or band-pass response. The super-unity-gain amplifier allows for very high Q factor and pass-band gain without the use of inductors [18]. A Sallen–Key filter is a variation on a VCVS filter that uses a unity-gain amplifier (i.e., a pure buffer amplifier with 0 dB gain). Because of its high input impedance and easily selectable gain, an operational amplifier in a conventional non-inverting configuration is often used in VCVS implementations. Implementations of Sallen–Key filters often use an operational amplifier configured as a voltage follower; however, emitter or source followers are other common choices for the buffer amplifier.

VCVS filters are relatively resilient to component tolerance, but obtaining high Q factor may require extreme component value spread or high amplifier gain. Higher-order filters can be obtained by cascading two or more stages.

The band-pass case of the Sallen-Key filter has a severe limitation. The value of Pole selectivity (Q) determines the gain of the filter, that is, it cannot be set independently, as it can with the low-pass or high-pass cases. The schematic of the Sallen-Key band-pass filter circuit are shown fig.5.3.

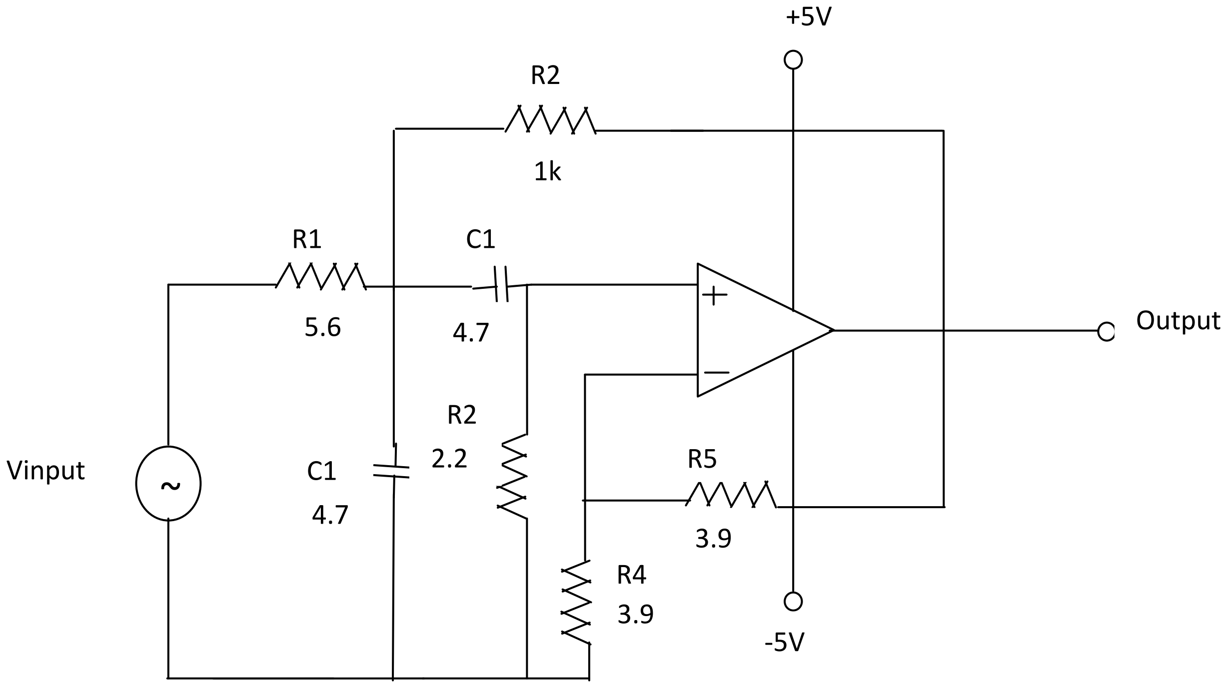


Fig.5.3. Sallen and Key Band Pass Filter Circuit

5.2.2. SKBPF Transfer Function

The nominal values of the circuit components are given below:

$R_1 = 5.6k\Omega$;

$R_2 = 1k\Omega$;

$R_3 = 2.2k\Omega$;

$R_4 = R_5 = 3.9k\Omega$;

$C_1 = C_2 = 4.7nF$.

All the components were assigned ±5%.

The voltage transfer function of the sallen- key band pass filter circuit is given by

$$H(s) = \frac{V_o(s)}{V_{in}(s)}$$

$$= \frac{\frac{ks}{R_1C_1}}{s^2 + \left(\frac{1}{R_1C_1} + \frac{1}{R_3C_2} + \frac{1}{R_3C_1} + \frac{1-k}{R_2C_1}\right)s + \frac{R_1+R_2}{R_1R_2R_3C_1C_2}} \tag{5.5}$$

Comparing equation.5.5 with second order BPF transfer function, we get the following relations for K, ώ0, and Q.

$$Gain, K = \frac{k}{R_1C_1} \tag{5.6}$$

$$PoleFrequency, \omega_p = \sqrt{\frac{R_1+R_2}{R_1R_2R_3C_1C_2}} \tag{5.7}$$

$$Poleselectivity, Q_p = \frac{\sqrt{\frac{R_1+R_2}{R_1R_2R_3C_1C_2}}}{\frac{1}{R_1C_1} + \frac{1}{R_3C_1} + \frac{1}{R_3C_2} + \frac{1-k}{R_2C_1}} \tag{5.8}$$

Therefore for the SKBPF of the filter with nominal values of the components yields k = 75,987, Q = 8.34 and f_o = 25HZ.

5.2.3. Fault Dictionary Creation

The procedure for the creation of fault dictionary is as same as SVF fault creation. The transfer function is simulated with faults injected to the components. The fault injection is done to the extent of ±50% deviation from nominal value with a step size of 5%. Single fault are introduced to one component at a time with other fault free components taking different random values within their tolerance.

Table 5.3. Fault Dictionary Samples

ANN Input				**ANN Target**
Fault Injected in Component	**Gain**	**Pole selectivity**	**freq**	**Fault Index**
R1+5%	72206.5	10.4648	24861.68	1
	72468.2	8.65264	24596.22	
	72452.3	9.61235	24561.92	
	72386.7	7.71696	24649.35	
	72265.8	10.7935	24585.49	
	72539.8	9.61000	24486.47	
	72265.3	9.25718	24636.08	
	72408.4	16.8320	24955.77	
	72573.4	7.49430	24666.27	
	72161.2	10.3563	24645	
R1+10%	69007.6	8.721377	24576.05	
	69197.11	8.825022	24429.72	
	69143.26	13.54071	24793.58	
	69060.86	9.881653	24920.91	
	69136.2	10.43722	24629.85	
	69169.81	8.976105	24429.01	
	69141.01	7.135413	24434.82	
	69255.5	12.06381	24806.14	
	69065.08	14.17858	24690.2	
	69089.22	10.76941	24732.67	

Fault dictionary is generated injecting fault to all component and evaluating the parameter K, Q, f. The input sample of size 1408×4 is obtained. Fault dictionary are separated into train samples and test samples randomly. Samples of fault dictionary for the component resistor $R_1$with fault index 1and capacitor C_1 with fault index 6 are shown in table 5.3 and table 5.4 to be used in ANN classifier for fault detection.

Table 5.4. Fault Dictionary Samples

ANN Input				ANN Target
Fault Injected in Component	**Gain**	**Pole selectivity**	**freq**	**Fault Index**
C1-50%	151279	-3.25884	35006.89	6
	151693.79	-4.381850	34577.621	
	151599.9	-4.11038	35146.44	
	151675.3	-3.4097	34931.66	
	151935.4	-3.32773	35310.29	
	152411.8	-4.93658	34803.7	
	152019.2	-3.64331	34997.42	
	151711.8	-4.24674	34995.43	
	151170	-3.10716	35360.46	
	151245.7	-4.01401	34764.09	
C1-45%	138085	-4.42296	33215.69	
	137969	-4.71895	33555.13	
	138135.3	-4.9726	33463.16	
	138519.5	-5.11997	33248.53	
	137550.7	-4.13017	33802.65	
	138248.8	-4.50096	33442.49	
	138348.9	-5.18737	33004.70	
	137542.6	-5.57255	33156.69	
	137642.	-4.690375	33677.30	
	137703.7	-4.43746	33665.2	

CHAPTER 6

PROPOSED BP-NN MODEL AND HYBRID GA-BP MODEL

6.1. PROPOSED BP-NN MODEL

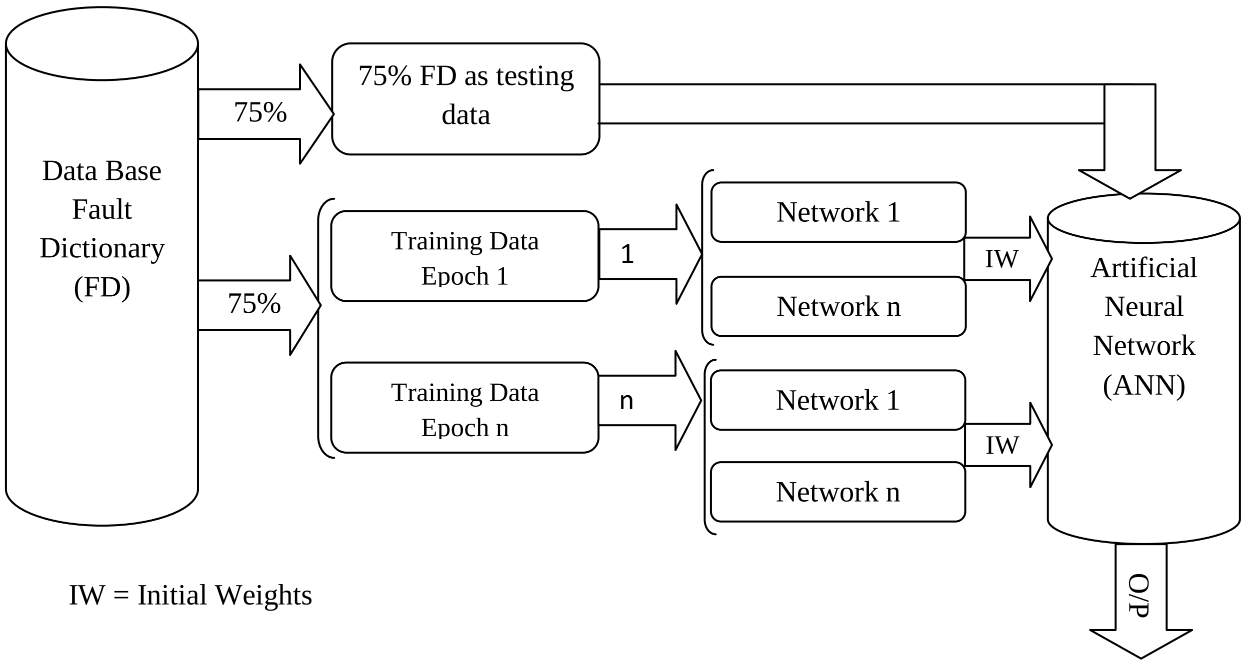


Fig.6.1. Functioning of ANN

Fault dictionary is created for two circuits State variable filter (SVF) and Sallen-key band pass filter (SKBPF). Fault dictionary are separated randomly into training set and testing set. Training set contains 25% of fault dictionary and remaining 75% as testing sets. Each fault dictionary data contains ANN inputs and ANN targets (desired outputs). Number of inputs and outputs for NN is designed depending on the components in the circuits. First train samples are given to ANN for training the neural network. BP algorithm regularly initializes all the weights of the network with minute random values; accordingly it takes the risk of being trapped in the local minimum. After training the network, test samples are given to the network and the trained networks correctly detects the particular fault injected. The effectiveness of correct classification is indicated by fault detection rate. Hidden layer neurons are varied and tested because hidden layers neuron plays a major role to achieve better performance. Fault detection rate is calculated with the output of the neural network, setting tolerance limit ±2%.

Fault detection rate = (No. of Correct classification) / (Total no. of test samples). (6.1)

6.2. HYBRID GA-BP MODEL

Neural networks offer many advantages in a variety of applications, but are ineffective if they are not properly designed. GA has a directed stochastic search, makes it a very robust and universal tool for almost any optimization problem which can be expressed in a reasonably small set of parameters. There are several ways for combination of GA and BP algorithms that have been used in many articles. Such as using GA algorithm to determine the optimal structure of ANN, weight optimization in back propagation network (BPN) algorithm and determine the number of hidden layers in the ANN. In this report hybrid GA-BP model is used to optimize the initial weights of the artificial neural network using genetic algorithm. Neural network is unstable with different results because of a small change in training data sets. Therefore the training data sets and testing data sets were defined for each artificial neural network as shown in Figure 6.1 and then the genetic algorithm would find the optimum initial weights of ANN as shown in Figure 6.2.

6.2.1. INITIAL POPULATION

In genetic algorithms, the binary code and the real code are the primary schemes to describe a chromosome. But, because the binary-coded scheme is neither necessary nor beneficial and according to the advantages of intuitiveness, resolution, and facility (i.e., need not to decode) for real code, the study used the real coded method for describing the chromosomes. There were 10 chromosomes generated in each generation, and each of chromosomes encoded with weights and biases representing the genes. The length of the chromosomes depends on the architecture. For each of architecture a number of connections weight vary according to the number of hidden layers and neurons in the hidden layers.

The range of lower bound and upper bound for weight based on trial and error basis. At first, the range between −2 and 2 was used, because all of the weights fell in this range after training by back-propagation neural network. But later, the study tried to set the value between −1 and 1 to compare with back-propagation neural network. And last, because the crossover operator could search over the initial range, we tried to narrow the range again between −0.5 and 0.5.

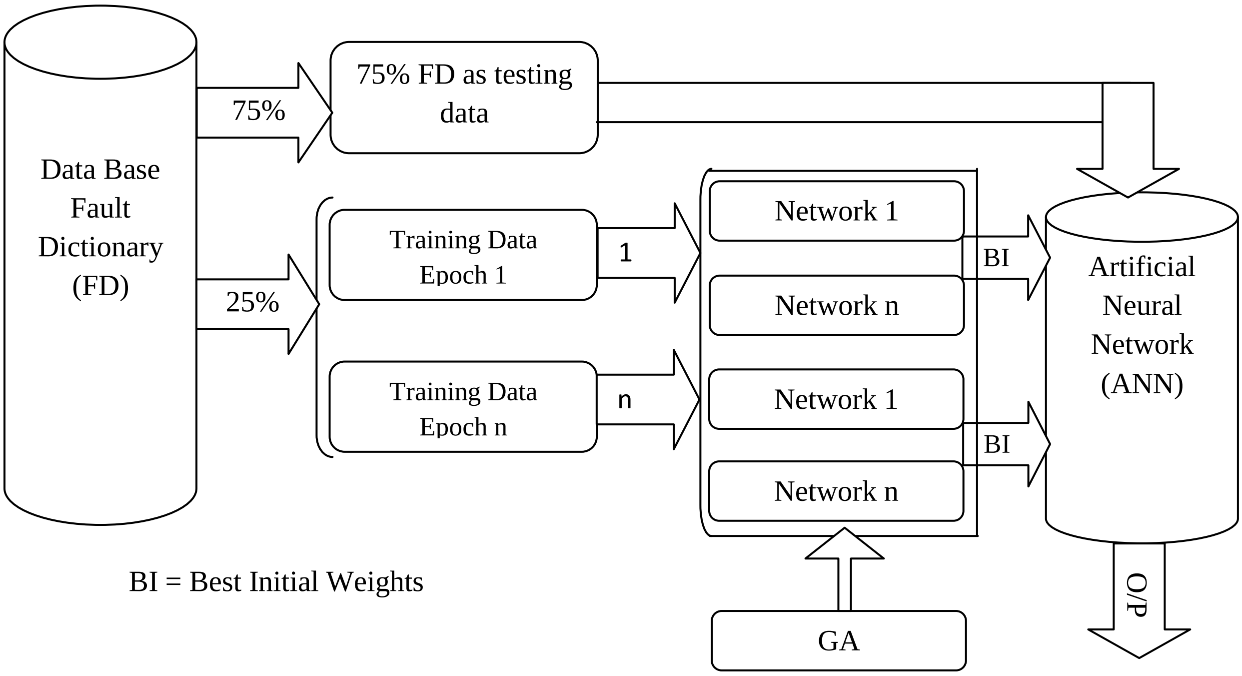


Fig.6.2. Functioning of GA-BP Model

6.2.2. Fitness Function Evaluation

Each individuals of the current population is evaluated by fitness function which is the mean square error value of the neural networks. The BPN network is initially trained with random connection weight value and MSE is calculated. The mean square error (MSE) value represents how the solution is fit for the problem. The optimum weights are obtained after GA runs for 100 generations. During the testing phase, best obtained value from GA is fed into the network and again MSE is evaluated. The fitness function is

$$Error, \varepsilon_j = \frac{\frac{1}{2}\sum_{i=1}^{n}(desiredoutput_i - actualoutput_i)^2}{n} \quad 6.2.$$

W is a vector constituted by all the weights and biases involved in the network, and n is the number of output units. In this scheme, an initial weight vector W_0 is iteratively adapted according to the following recursion to find an optimal weight vector. The positive constant of ŋ is learning rate.

$$W_{K+1} = W_k - \eta \frac{\partial(\varepsilon_j)}{\partial(W)} \quad 6.3.$$

6.2.3. GA Operators

The three operators of the genetic algorithm, are used in artificial systems are Mutation, Crossover and Selection. The evolution usually starts from a population of randomly generated individuals and is an iterative process, with population in each-iteration called a generation. The more fit individuals are stochastically selected from the current population, and each individual's genome is modified (recombined and possibly randomly mutated) to form a new generation. The new generation of candidate solutions is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of generations has been produced, or a satisfactory fitness level has been reached for the population.

6.3. HYBRID GA-BP ALGORITHM

The GA algorithm requires initial settings for its run. The initial setting values of GA are shown in table.6.1. BP can converge quickly on the local optima and also regularly initializes all the weights of the network with minute random values; accordingly it takes the risk of being trapped in the local minimum. Now, the GA provides the back propagation neural network (BP-NN) better initial values. Subsequently, both the training efficiency and developing speed can be improved. So the fundamental idea of the hybrid GA-BP algorithm is straightforward and simple, GA algorithm is used to explore the optimal combination of all the neural network parameters and then BP algorithm is used to find the correct classification of the each parameter. The framework of GA-BP scheme is shown as fig.6.3.

Table 6.1. Parameters of Genetic Algorithm

Parameter	**Value**
Population size	10
Generation	100
Cross over probability	0.8
Mutation probability	0.2
Stall Gen limit	20
Elite count	2

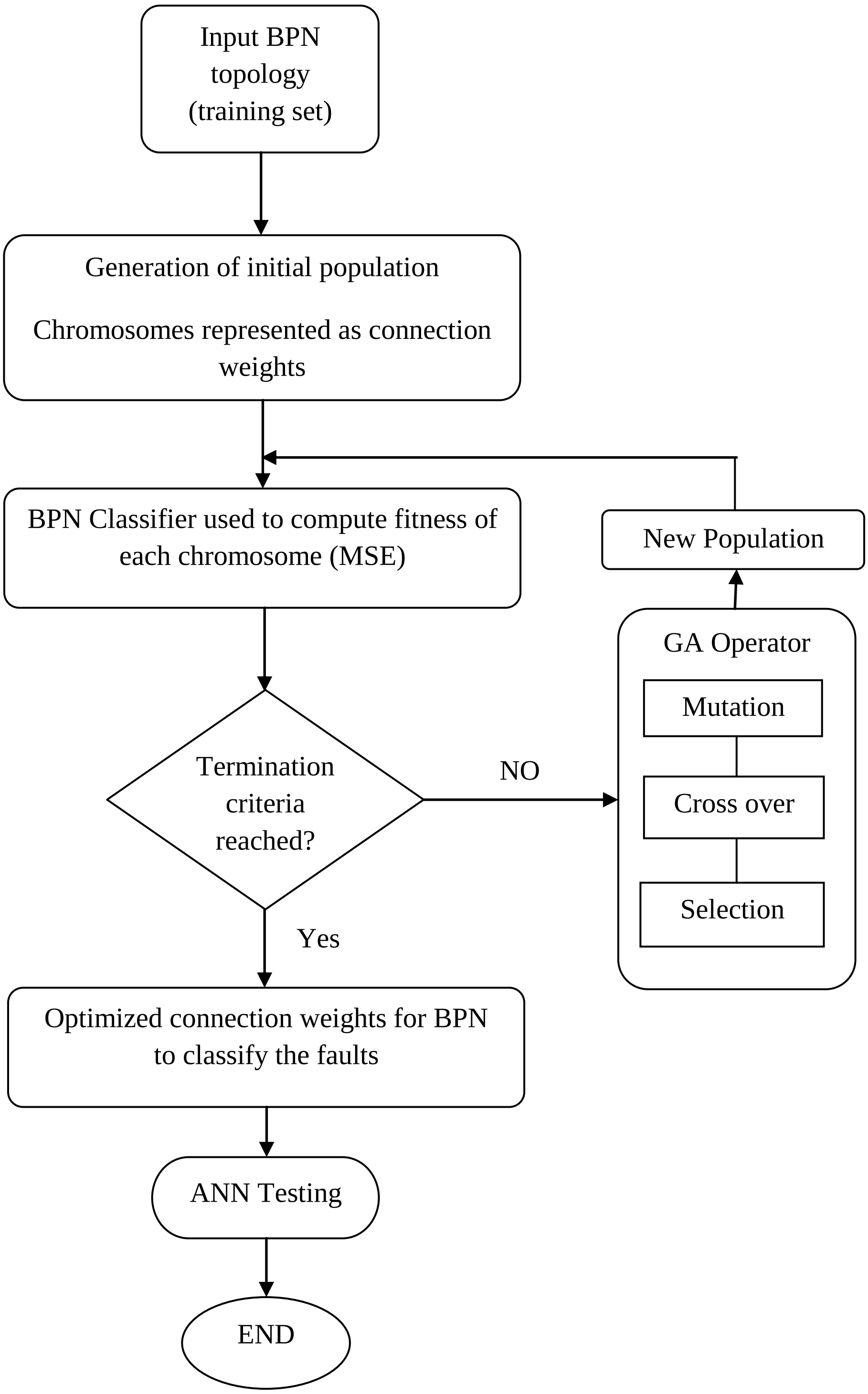


Fig.6.3. Flow diagram of GA-BP Model

- Population is initialized with 10 individuals representing weights and bias values of designing the NN specific architecture and the length of chromosomes contains specific values of weight and biases for particular architecture.
- Each individual is evaluating for its quality through MSE as the fitness function.
- Termination criteria are fixed as 100 generation.
- Initially the NN is trained with train sets and MSE calculated.
- GA is invoked which gives the optimum value of weight value and using this weights NN is tested with test sets and MSE is minimized for improved performance.

CHAPTER 7

SIMULATION RESULTS

7.1. NEURAL NETWORK PERFORMANCE FOR SINGLE FAULT

Fault detection is carried by fixing the feed forward neural network architecture with suitable hidden layer neuron without hybridization. The weight and bias values automatically determined and fed into neural network along with train samples. After ANN trained, the separate test samples are fed to the network and faults which are correctly identified are observed. The fault detection rate of SVF and SKBPF is estimated for each component and is tabulated in table.7.1 and table.7.2 respectively.

Table.7.1. Performance of BPN (SVF)

Network topology	Epoch	Number of samples	Time (secs)	R1	R2	R3	R4	R5	R6	R7	C1	C2
4-15-12-16-9	60	1407	10s	0.41	0.46	0.49	0.26	0.57	0.40	0.24	0.60	0.73
4-16-6-9-9	52	1407	9s	0.43	0.57	0.26	0.31	0.26	0.63	0.10	0.32	0.56
4-14-6-17-9	116	1407	18s	0.47	0.80	0.34	0.47	0.59	0.41	0.34	0.56	0.48
4-12-8-20-9	102	1407	15s	0.59	0.54	0.47	0.58	0.64	0.49	0.17	0.43	0.60
4-11-8-6-9	89	1407	12s	0.54	0.56	0.27	0.33	0.47	0.38	0.14	0.45	0.47

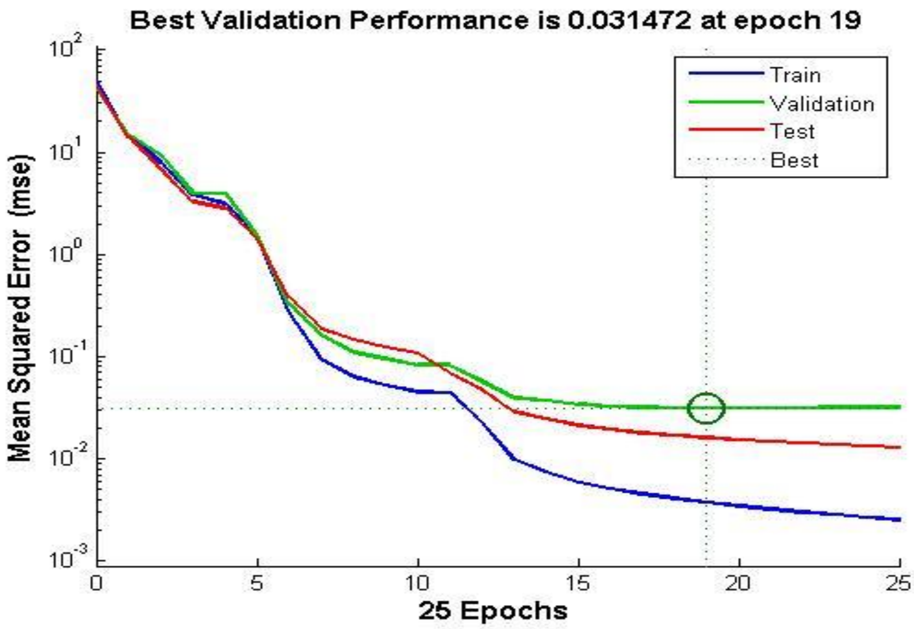


Fig.7.1. MSE Value of each epoch (SVF)

The overall Mean square error (MSE) value of each iterations (epochs) are calculated. The 4-5-12-16-9 sample architecture is designed and mean square error (MSE) values are plotted in graph (Fig.7.1). Best performance (MSE) in epoch 19 is 0.031472.

Table.7.2.Performance of BPN (SKBPF)

Network topology	Epoch	No. of samples	Time (secs)	R1	R2	R3	R4	R5	C1	C2
4-15-12-16-7	29	1090	7	0.3910	0.5556	0.4843	0.2368	0.1899	0.0655	0.0265
4-15-18-12-7	33	1090	10	0.0833	0.4248	0.2579	0.2171	0.1853	0.0854	0.066
4-15-9-16-7	36	1090	12	0.6923	0.7972	0.6352	0.5987	0.2911	0.0065	0.0662
4-12-12-16-7	34	1090	11	0.2179	0.3856	0.2579	0.1645	0.2089	0.0844	0.0922
4-15-15-18	24	1090	7	0.7244	0.9804	0.6289	0.7303	0.6582	0.0779	0.1722

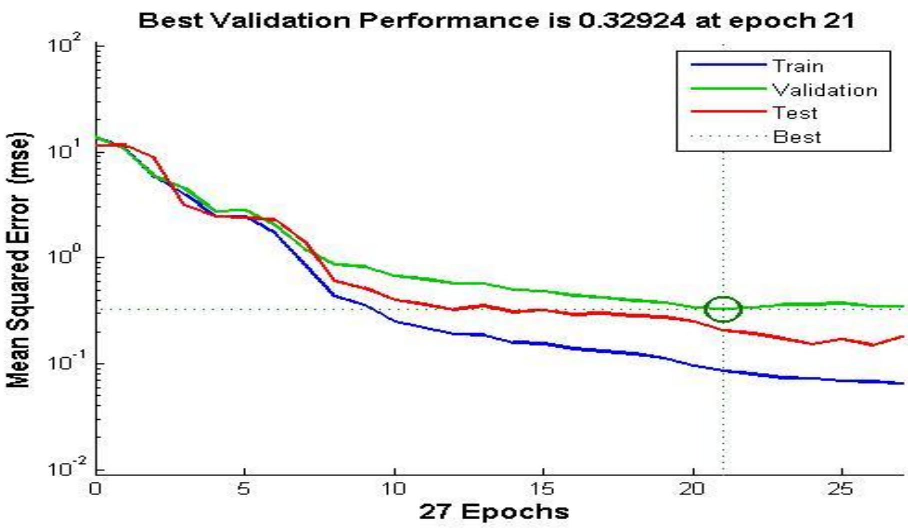


Fig.7.2. MSE Value of each epoch (SKBPF)

7.2. HYBRID GA-BP PERFORMANCE FOR SINGLE FAULT

The weight and bias values determined after 100 generation using GA for the same neural architecture is fed into neural network along with train samples. After ANN trained, the separate test samples are fed to the network and faults which are correctly identified are observed. The fault detection rate is estimated for each component and is tabulated in table 7.3 and table.7.4 respectively. GA-BP model outperformed BPN.

Table.7.3. Performance of GA-BP (SVF)

Network topology	**Epoch**	**Number of samples**	**Time (secs)**	**R1**	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**	**C1**	**C2**
4-15-12-16-9	56	1407	10s	0.72	0.85	0.78	0.57	0.62	0.55	0.49	0.70	0.83
4-16-6-9-9	55	1407	10s	0.78	0.80	0.45	0.45	0.67	0.73	0.45	0.54	0.86
4-14-6-17-9	102	1407	14s	0.75	0.82	0.54	0.52	0.79	0.54	0.51	0.45	0.58
4-12-8-20-9	112	1407	15s	0.62	0.64	0.56	0.78	0.84	0.60	0.32	0.61	0.85
4-11-8-6-9	95	1407	15s	0.69	0.65	0.57	0.47	0.65	0.55	0.48	0.65	0.68

Validation

Fault is injected in one component R_1 to the extent of ±50% variation along with 5% step size. The sample size for testing is 150×4(for one component). When this testing sample is fed to GA-BP network, the expected output is fault index 1. The correct classification done by the implemented model is identified and fault detection rate calculated by the equation 7.1 for R_1 is 72% of GA-BP which is much higher than the value 41% obtained without GA hybridization. Comparison of GA-BP and BPN are shown in fig.7.5 and fig.7.6 for SVF and SKBPF respectively.

Performance samples = (No. of correct classification) / (Total No. of samples)   (7.1)

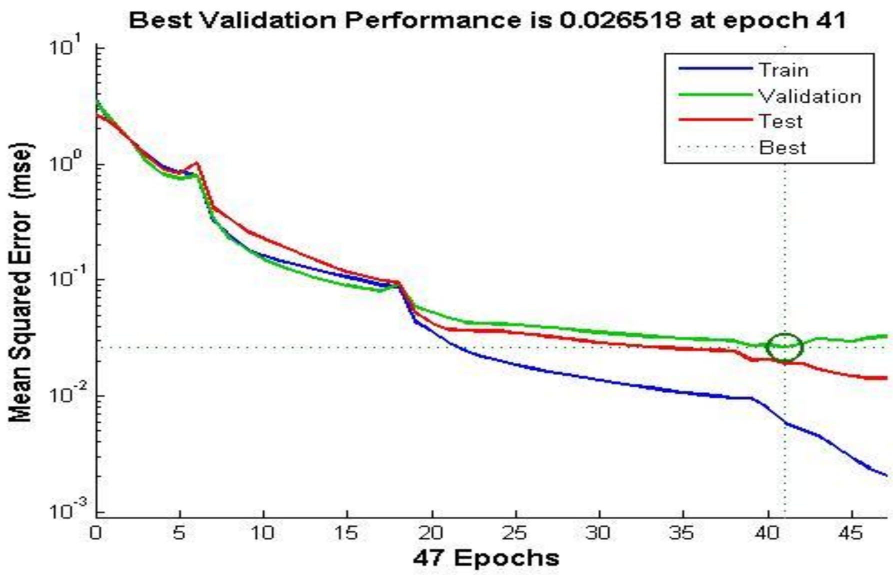


Fig.7.3. MSE Value of each epoch (SVF) with GA Optimization

Table7.4.Performance of GA-BP (SKBPF)

Network topology	Epoch	No. of samples	Time (secs)	R1	R2	R3	R4	R5	C1	C2
4-15-12-16-7	31	1090	8	0.80	0.98	0.86	0.92	0.83	0.24	0.20
4-15-18-12-7	37	1090	13	0.92	0.94	0.89	0.82	0.75	0.24	0.33
4-15-9-16-7	33	1090	10	0.87	1.00	0.95	0.89	0.79	0.31	0.11
4-12-12-16-7	30	1090	8	0.48	0.99	0.84	0.47	0.43	0.25	0.01
4-15-15-18-7	33	1090	10	0.78	0.96	0.93	0.92	0.89	0.30	0.36

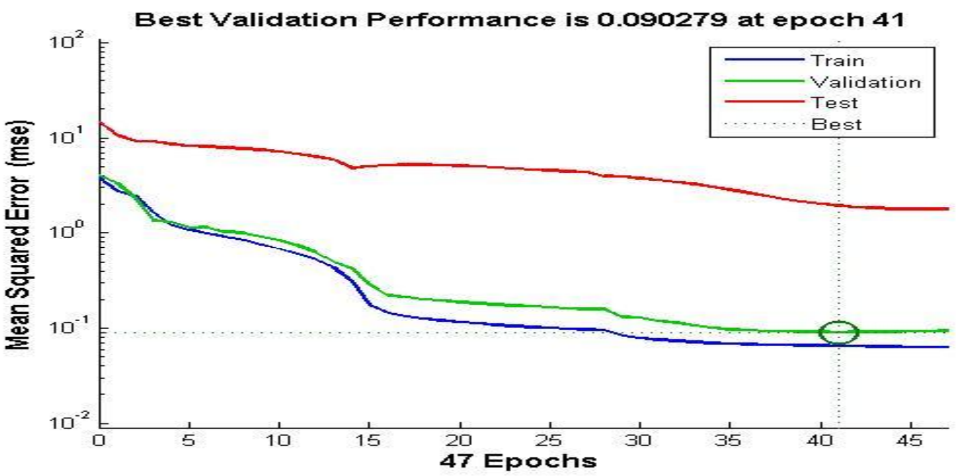


Fig.7.4. MSE Value of each epoch (SKBPF) with GA Optimization

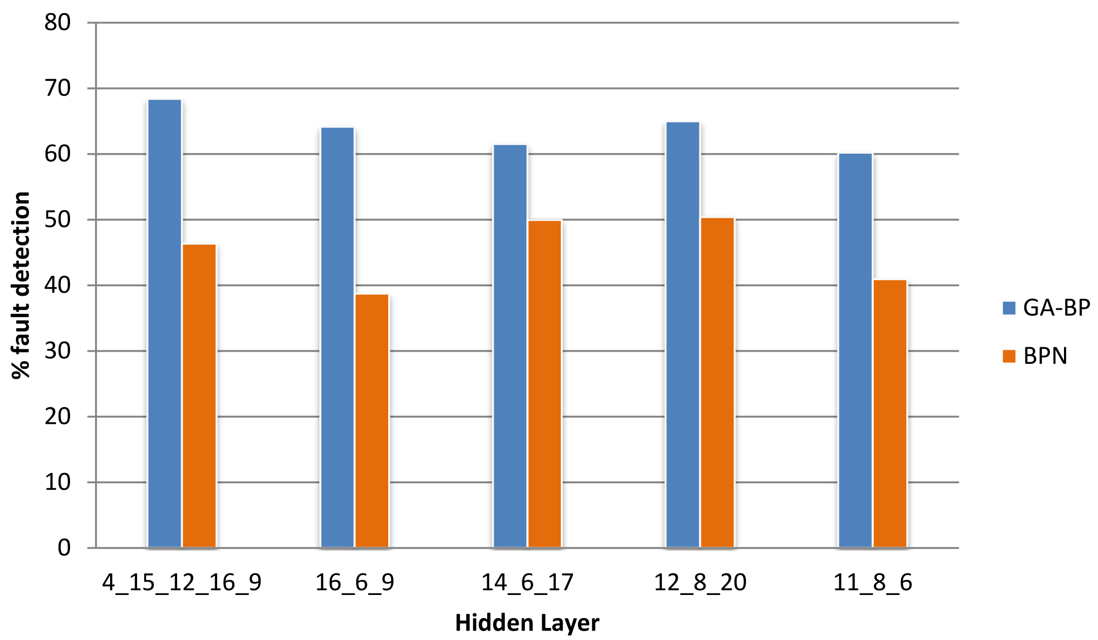
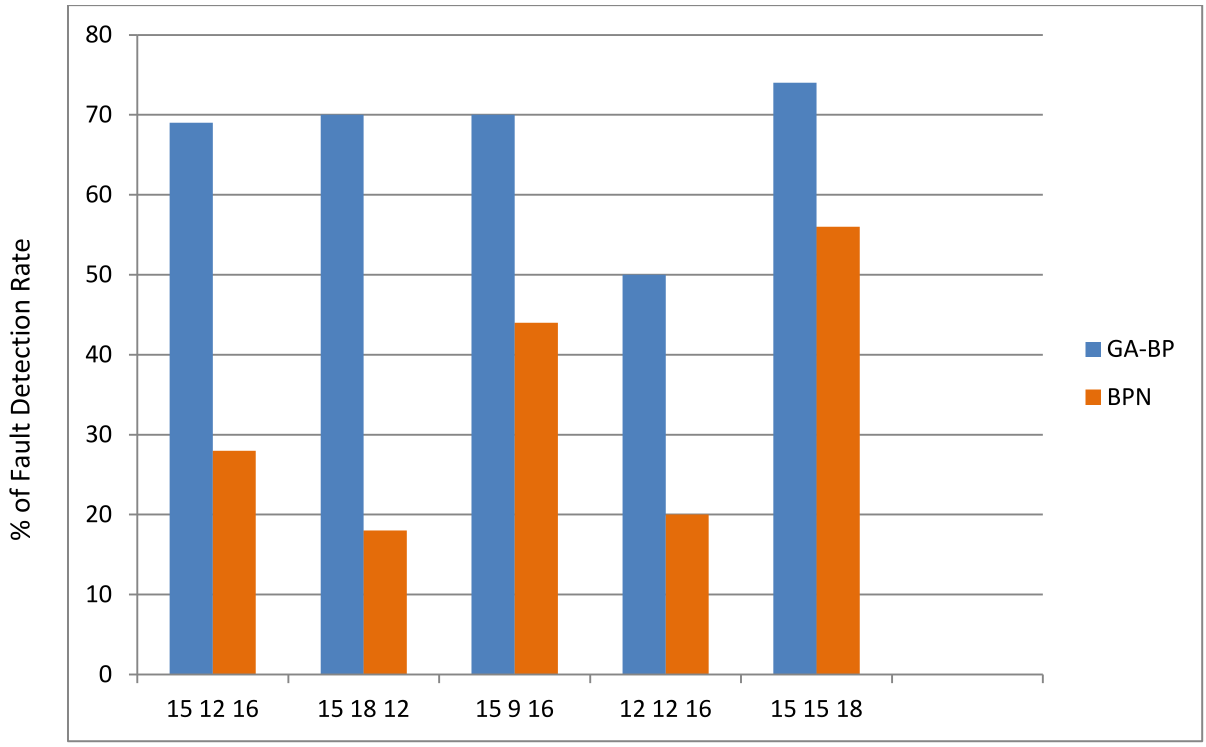


Fig.7.5.Comparison of GA-BP and BPN (SVF)



Hidden Layer

Fig.7.6.Comparison of GA-BP and BPN (SKBPF)

7.3. DOUBLE FAULT DETECTION

SKBPF is considering for double fault detection as it contain lesser number of component. Fault is injected for two components at a time. Fault dictionary is created for all components and resulting a sample size of 50400×6 along with fault index. The fault dictionary is used for BPN and GA-BP model. The results obtained for GA-BP is much higher than the BPN models. Performance of the double faults for both BP-NN and GA-BP of a single architecture (5-15-12-16-7) are shown in table 7.5.

Table.7.5 Performance of GA-BP and BPN (SKBPF)

Fault injected Components	Fault detection rate	
	GA-BP	BP-NN
R1-R2	0.8927	0.6637
R1-R3	0.7611	0.6692
R1-R4	0.5123	0.5257
R1-R5	0.6628	0.4733
R1-C1	0.4011	0.4135
R1-C2	0.4363	0.3523
R2-R3	0.8841	0.8449
R2-R4	0.7986	0.7405
R2-R5	0.8896	0.7460
R2-C1	0.6452	0.5035
R2-C2	0.4615	0.4743
R3-R4	0.7232	0.5139
R3-R5	0.8129	0.8027

Table.7.5 Performance of GA-BP and BPN (SKBPF) (Contd.)

R3-C1	0.3660	0.1296
R3-C2	0.4001	0.3093
R4-R5	0.5679	0.3620
R4-C1	0.3130	0.1493
R4-C2	0.3022	0.3168
R5-C1	0.3235	0.2957
R5-C2	0.3139	0.1439
C1-C2	0.9214	0.7394

The fault detection performance is improved with the hybrid model and the no of epochs taken is considerably reduced to 120 from 335 for BPN model. Time taken is also minimized to 2 min 22 sec where the BPN model it is doubled.

CHAPTER 8

CONCLUSION AND FUTURE WORK

The proposed work describes the use of GA to train neural network. The weight in different layers of the network is optimized using genetic algorithm. The relative difference between the fault detection rate of ANN and GA-BP model is analyzed for fault diagnosis of two circuits SVF and SKBPF. The experimental results show that architecture 4-15-12-16-9 gave good results for single fault in SVF circuit. The average fault detection rate is 40% without optimization and 71% with GA optimization. Similarly for SKBPF, 4-15-18-12-7 architecture produced better results. The average fault detection rate is 36% without optimization and 80% with GA optimization. The work can be extended with other evolutionary algorithm like particle Swarm Optimization (PSO), Artificial Bee Colony (ABC) algorithm. Neural network training can be implemented with optimization for deciding the number of internal layers and number of neurons in each layer.

REFERENCES

[1] H. Spence, N. Carmichael, and C. Camargo, "Automatic analog fault simulation," in Proceedings of the AUTOTESTCON Conference, pp. 17–22, Dayton, Ohio, USA, September 1996.

[2] R. W. Liu, Selected Papers on Analog Fault Diagnosis, IEEE Press, New York, NY, USA, 1987.

[3] M. Catelani and A. Fort, "Soft fault detection and isolation in analog circuits: Some results and a comparison between a fuzzy approach and radial basis function networks," IEEE Transactions on Instrumentation and Measurement, vol. 51, no. 2, pp. 196–202, 2002.

[4] V. Manikandan and N. Devarajan,"SBT Approach towards Analog Electronic Circuit Fault Diagnosis" Hindawi Publishing Corporation, Active and Passive Electronic Components, Volume 2007, Article ID 59856.

[5] S. Hsu, et al., "Comparative analysis of five neural network models," Remote Sensing Reviews, vol. 6, pp. 319–329, 1992.

[6] David E. Gold berg, "Genetic Algorithms in Search, Optimization, and Machine Learning" Addison Wesley; 1 edition The University of Alabama (1 January 1989).

[7] Xifeng Li, Yongle Xie, "Analog Circuits Fault Detection Using Cross-Entropy Approach" © Springer Science + Business Media New York 2013.

[8] Yibing Shi · Yong Deng ·Wei Zhang, "Diagnosis of Incipient Faults in Weak Nonlinear Analog Circuits" © Springer Science + Business Media New York 2013.

[9] Ke Huang, Salvador Mir, "Diagnosis of Local Spot Defects in Analog Circuits" IEEE TRANSACTIONS ON INSTRUMENTATION AND MEASUREMENT, VOL. 61, NO. 10, OCTOBER 2012.

[10] Zhan-Li Sun, Kin-Fan Au, and Tsan-Ming Choi,, "A Neuro-Fuzzy Inference System Through Integration of Fuzzy Logic and Extreme Learning Machines" Ieee Transactions On Systems, Man, And Cybernetics—Part B: Cybernetics, Vol. 37, No. 5, October 2007.

[11] Han Han & Houjun Wang & Shulin Tian & Na Zhang, "A New Analog Circuit Fault Diagnosis Method Based On Improved Mahalanobis Distance" Springer Science + Business Media New York 2012.

[12] A. Kavithamani & V. Manikandan & N. Devarajan, "Soft Fault Classification of Analog Circuits Using Network Parameters and Neural Networks" Springer Science + Business Media New York 2013.

[13] Ashok, Kavithamani & Venugopal Manikandan & Nanjundappan Devarajan, "Analog Circuit Fault Detection Using Location Of Poles" Springer Science + Business Media, LLC 2011.

[14] Suraj Sindia, Vishwani D, Agrawal, Virendra Singh, "Parametric Fault Testing Of Non-Linear Analog Circuits Based On Polynomial And V-Transform Coefficients" © Springer Science+Business Media, LLC 2012.

[15] Yongcai Ao · Yibing Shi ·Wei Zhang · Yanjun Li, "An Approximate Calculation of Ratio of Normal Variables and Its Application in Analog Circuit Fault Diagnosis" © Springer Science + Business Media New York 2013.

[16] Minfang Peng, Chi K. Tse, Meie Shen, Kai Xie, "Fault Diagnosis Of Analog Circuits Using Systematic Tests Based On Data Fusion" © Springer Science + Business Media, LLC 2012.

[17] Bing Long, Min Li, HoujunWang, Shulin Tian, "Diagnostics of Analog Circuits Based on LS-SVM Using Time-Domain Features" © Springer Science + Business Media New York 2013.

[18] Prithviraj Kabisatpathy, Alok Barua and Satyabrota Sinha, "Fault Diagnosis of Analog Integrated Circuits" Springer International Edition 2008.

[19] R. Seaks and R.W. Liu, "Fault isolation via component simulation" IEEE Transactions on Circuit Theory, Vol, CT-19,pp, 634-640, nov 1972.

[20] R.M. Biernacki and J.W.Bandler, "Post Production parameter identification of Analog circuits" proceedings, IEEE International Symposium on circuits and System, pp.1082-1086, Apr 1980.

[21] S. Haykin, "Neural Networks a comprehensive Foundation, prentice Hall, 2nd edition, 1998.

[22] Christopher M.Bishop, "Neural Networks for pattern Recognition" Oxford University Press, 1995.

[23] Morimoto, T. (2006). Genetic algorithm. In S. S. Sablani, M.S. Rahman, A. K. Datta, & A.S. Mujumdar (Eds.), Handbook of Food and bioprocess modeling techniques (pp. 405–434). New York: CRC Press.

[24] Koehn P. (1994). Combining Genetic Algorithms and Neural Networks. A Thesis Presented for the Master of Science Degree The University of Tennessee, Knoxville.

[25] W. Hochwald and J.D. Bastin, “A DC Approach for Analog Fault Dictionary Determination” IEEE transactions on circuits and systems, Vol. CAS-26, pp, 523-529, July 1979.

CONFERENCE PUBLICATION

The paper presented on **Soft Fault Diagnosis in Analog Circuit using Hybrid Genetic Algorithm and Neural Network** in international conferences ICIIECS'15 at **Karpagam College of Engineering.**