EVALUATION OF STOCHASTIC OPTIMISATION ALGORITHMS FOR INDUCTION MACHINE WINDING FAULT IDENTIFICATION

Mohamoud Omran A. Alamyal

B.Sc., M.Sc.

A thesis submitted for the degree of

Doctor of Philosophy

June, 2012

School of Electrical and Electronic Engineering

Newcastle University

United Kingdom

ABSTRACT

This thesis is concerned with parameters identification and winding fault detection in induction motors using three different stochastic optimisation algorithms, namely genetic algorithm (GA), tabu search (TS) and simulated annealing (SA).

Although induction motors are highly reliable, require low maintenance and have relatively high efficiency, they are subject to many electrical and mechanical types of faults. Undetected faults can lead to serious machine failures. Fault identification is, therefore, essential in order to detect and diagnose potential failures in electrical motors. Conventional methods of fault detection usually involve embedding sensors in the machines, but these are very expensive. The condition monitoring technique proposed in this thesis flags the presence of a winding fault and provides information about its nature and location by using an optimisation stochastic algorithm in conjunction with measured time domain voltage, stator current data and rotor speed data. This technique requires a mathematical ABCabc model of the three-phase induction motor.

The performance of the three stochastic search methods is evaluated in this thesis for their use to identify open-circuit faults in the stator and rotor windings of a three-phase induction motor. The proposed fault detection technique is validated through the use of experimental data collected under steady-state operating conditions.

Time domain terminal voltages and the rotor speed are used as input data for the induction motor model while the outputs are the calculated stator currents. These calculated currents are compared to the measured currents to produce a set of current errors that are integrated and summed to give an overall error function. Fault identification is achieved by adjusting the model parameters off-line using the stochastic search method to minimise this error function. The estimate values for the winding parameters give the best possible match between the performance of the faulty experimental machine and its mathematical ABCabc model. These estimates of the values of the motor winding parameters are used in the detection of the development of faults by identifying both the location and the nature of the winding fault. The effectiveness of the three stochastic methods to identify stator and rotor winding faults are compared in terms of the required computation resources and their success rates in converging to a solution.

ACKNOWLEDGEMENTS

First of all, my deepest gratitude is due to God, the most graceful and the most merciful for his blessings.

I would like to express my sincere gratitude to my supervisor *Dr. Bashar Zahawi* for his guidance, advices and support throughout the duration of this work.

Thanks also go to my second supervisor *Dr. Damian Giaouris* for his help and support. Moreover, I would like to thank my friends and colleagues at Newcastle University for the wonderful discussions and friendly working atmosphere during my stay here.

I deeply appreciate the scholarship awarded to me to do this research degree by the Ministry of Higher Education of Libyan Government.

Lastly, I would like to thank my parents, my wife and my lovely children for their support, care and love.

June, 2012

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF PRINCIPAL SYMBOLS AND ABBREVIATIONS

Below is a list of symbols and abbreviation that are used throughout this thesis. When possible the meaning of each symbol is given in the text when the symbol is first used.

R_{sA}, R_{sB}, R_{sC}	Stator Resistances
R_{ra}, R_{rb}, R_{rc}	Rotor Resistances
V_{sA}, V_{sB}, V_{sC}	Stator voltages
I_{sA}, I_{sB}, I_{sC}	Stator Currents
I_{ra}, I_{rb}, I_{rc}	Rotor Currents
L_{sA}, L_{sB}, L_{sC}	Stator Self Inductances
M_{ss}	Mutual Inductance between pairs of stator windings
L_{ra}, L_{rb}, L_{rc}	Rotor Self Inductances
M_{rr}	Mutual inductance between pairs of rotor windings
$M_{sr} = M_{rs}$	Mutual inductance between stator/rotor winding
L_m	Stator to Rotor Mutual Inductance
w	Angular rotor speed
θ_r	Rotor position angle
DC	Direct Current
GA	Genetic Algorithm

SA	Simulated Annealing
TS	Tabu Search
ACO	Ant Colony Optimisation
PSO	Particle Swarm Optimisation
BFO	Bacterial Forging Optimisation
L_s	Tabu list
L_{st}	Length of the string.
T	Temperature
T_s	Initial temperature
P_A	Acceptance probability
P_s	Swap probability
AI	Artificial intelligence
ANNs	Artificial neural networks
FLS	Fuzzy logic systems
MCSA	Motor current signature analysis
SOA	Stochastic optimisation algorithms

CHAPTER 1

INTRODUCTION

The three-phase induction motor is used in a wide variety of applications because of its simple, rugged construction, easy maintenance, low cost and good operating characteristics. It is an electromechanical energy device in which the energy is converted from electrical to mechanical form. Although induction motors are highly reliable and have relatively high efficiency, they are subject to many types of faults. The ability to test the integrity of the motor through this process results in lower maintenance costs and an overall lowered risk of malfunction. Conventional methods of condition monitoring are based on a variety of technologies including vibration analysis and current signature analysis.

Condition monitoring of induction motor is usually applied to detect various types of electrical and mechanical faults. It is important to be able to detect faults while they are still developing. This called incipient failure detection. A fault that is not identified in the initial stage may become catastrophic and the induction motor may suffer severe damage. Thus, undetected motor faults may result in motor failure and complete shutdown of the machine. Such shutdowns are very costly in terms of lost production time, wasted raw materials and maintenance costs. Condition monitoring is necessary for identifying machine defects and their location. Knowledge of the motor condition allows the operator to review the physical state of the motor so as to prevent machine damage by stopping the process and carrying out the required maintenance at an appropriate time. The application of condition monitoring in plants results in savings in maintenance costs, and improved safety. Therefore, many condition monitoring techniques have been proposed and applied to the fault detection of three-phase induction motors [1-15].

A new method for detecting induction motor faults based on global random optimisation methods has recently been developed [16]. This technique has the potential to identify a wide variety of faults without the need for knowledge of various fault signatures. However, this method requires the measurement of the rotor position angle θ, limiting the potential use of this approach because of the implied extra cost and complexity. In this work, this technique is developed and is also applied for inverter-fed induction motor using different stochastic methods; no rotor position is required for the proposed technique. The proposed technique uses only terminal voltages, stator currents and rotor speed data obtained during steady state with load disturbances.

Fault identification is implemented by adjusting the induction machine model parameters off-line, using a stochastic search method; Genetic Algorithm (GA), Tabu Search (TS) and Simulated Annealing (SA) to estimate values of the winding parameters which are those that give the best possible match between the performance of the faulty experimental machine and its mathematical ABCabc model. The changes in these parameters help in the detection of the development of faults, thus identifying both the location and the nature of the winding fault.

1.1 Overview of thesis

The thesis consists of eight chapters and three appendices. The work presented in the thesis investigates the performance of genetic algorithm, tabu search and simulated annealing for parameter identification and fault identification in induction motors. The thesis is structured as follows:

Chapter 1 gives a general introduction of the main work. Chapter 2 provides a review of literature in this area of research. The literature review contains an overview of condition monitoring and fault diagnostics of induction machines to show what has been done by other researchers, as well as a discussion of induction motor failures and methods of detection of motor faults. A brief overview of stochastic optimisation methods are also presented in this chapter.

In chapter 3, three standard stochastic optimisation methods (GA, TS and SA) are described and their ability to locate global optima demonstrated using simple function. This chapter is divided into three parts - part one presents the GA, part two presents the TS, and part three presents the SA.

Chapter 4 gives details of the condition monitoring scheme and experimental machine set-up used for parameter identification and, to emulate the presence of machine winding faults in this investigation.

Experimental verification of the GA algorithm for the induction machine parameter identification and fault detection are presented in Chapter 5. For parameter identification, the performance of the identification scheme is demonstrated with measured data obtained from a healthy machine at steady-state and the electric parameters obtained using this method and the other two methods are evaluated and compared with parameters obtained from IEEE standard tests. For fault detection, the GA algorithm was used in conjunction with steady-state loaded data sets from a faulty machine. This is to demonstrate the proposed fault identification method using Matlab/Simulink as a software platform.

Chapter 6 presents the application of TS algorithm using the same experimental data and conditions for same faults and for parameter identification while Chapter 7 presents the application of SA algorithm. Chapter 8 concludes the work presented in the thesis and an overall discussion of the results. Also presents some suggestions for future work followed by the list of references and the Appendices. Source codes for the GA, TS and SA algorithms can be found in Appendix A, B and C respectively

1.2 Contributions

The main aim of the research work is to develop a technique [16] for detecting induction motor winding faults by estimating its parameters. The proposed technique is based on the application of GA, TS and SA algorithms to identify the machine parameters which relate to a given set of measured current and voltage waveforms at steady state condition for supply-fed and inverter-fed induction machine. The required model for this identification scheme is very simple and the used stochastic algorithms are easy to implement with the help of Matlab/Simulink environment. The contributions of this research are summarised as follows:

- Develop a technique for the induction machine parameter identification and fault detection
- Demonstrate the application of this technique using three stochastic optimisation algorithms (GA, TS and SA) for parameter identification and fault detection
- Evaluate the performance of the three stochastic algorithms when used in conjunction with machine experimental data sets acquired under steady-state conditions
- The performance of the three stochastic algorithms when used in conjunction with an inverter-fed induction machine is also examined, using simulation data only.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

In this chapter, the literature on condition monitoring of electrical machine is reviewed. Induction motors like other rotating electrical machines are subject to mechanical and electromagnetic forces which could lead to the development of a fault. The widespread use of electric motors is becoming ever more common in various industries, resulting in an increased demand for fault detection methods. There is a wide variety of research conducted in motor condition monitoring and fault diagnostics, and there are many different ideas and techniques for performing motor condition monitoring.

One of the simplest methods to protect electrical machines is by using different types of protection relays, which sense serious disruptions of the current flowing in the windings and operate to trip or disconnect the machine once a fault such as an over-voltage, an over-current or an earth-fault has occurred. Conventional techniques for induction machine conditional monitoring usually involves sensors embedded in the machine which is very expensive. These sensors are used to measure and detect data such as temperature and vibration and help detect developing faults [17]. Additionally, these conventional methods of fault detection are not favourable for smaller-sized electric machines. They are widely used in larger machines but they fall short in their application in smaller machines as a result of limitations such as the size of the sensing-device and financial considerations [18]. Furthermore, sensors are also limited in their ability to detect some kinds of faults. The stator current monitoring can provide the same indications without requiring any access to the motor. This technique uses results of spectral analysis of the stator current of an induction motor for fault detection [19].

The major faults in electrical machines can be classified as follows [6]

- Stator faults resulting in the opening or shorting of one or more stator coils or phase windings,
- Broken rotor bars or cracked end-rings,
- Static and/or dynamic air-gap eccentricities,
- Bent shaft,
- Shorted rotor field winding,
- Bearing and gearbox failures.

These faults produce one or more of the following symptoms:

- Unbalanced voltages and line currents,
- Increased torque pulsation,
- Decreased average torque,
- Increased losses and reduction in efficiency,
- Excessive heating,
- Vibrations.

Different techniques of fault identification have been developed and used effectively to detect the machine faults at an early stage using different machine quantities, such as current, voltage, speed, efficiency, temperature and vibrations. To identify the above faults, the diagnostic methods may involve several fields of science and technology [6]. Due to the prevalent increase in automated machines alongside the reduction in human supervision of system operations, the need for condition monitoring is paramount [20].

Many methods have been developed for the purpose of detecting mechanical and electrical faults in induction motors, either directly or indirectly, such as motor current signature analysis [21], vibration monitoring and analysis of the negative sequence components of the stator current. The monitoring and fault detection of electrical machines has moved in recent years from traditional techniques towards artificial intelligence (AI) techniques such as artificial neural networks (ANNs) and fuzzy logic systems (FLS) [22]. Heuristic optimisation techniques which are dependent on the idea of neighbourhood searches are also used in the fault detection of induction motors [16, 23]. Many papers have been published presenting methods for induction motor fault identification by using different techniques [24-35].

2.2 Induction Machine Faults

Although induction motors are reliable electric machines, they are subject to many electrical and mechanical types of faults and these failures can be classified as been the result of internal or external factors. Internal factors originate from the motor itself and arise in one of the three main induction machine components; the stator, the rotor or the bearings. There are a few failure types in the induction motor caused by external factors such as cooling, insulation, environmental and manufacturing problems.

Electrical faults include short circuits in stator windings, open circuits in stator windings, and open circuits in rotor windings, while mechanical faults include bearing failures and rotor eccentricities. The risk of failure can be decreased if these faults are recognised and corrected. In general, these faults can be classified as stator related, rotor related and bearing related faults with the percentages of total failure [36] as shown in Figure 2.1,

- Stator winding faults 38%
- Bearing failures 40%
- Rotor faults 10%
- Other faults 12%

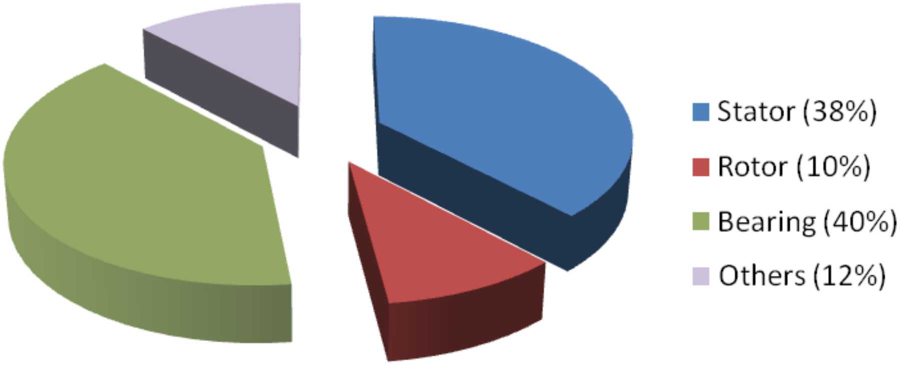


Figure 2.1. Types of induction machine faults [36].

2.2.1 Electrical faults

The most common faults related to the stator winding of induction motors are turn-to-turn, phase-to-phase, coil-to-coil and coil-to-ground faults. The broken bar and end ring faults of squirrel cage rotors can also occur. Furthermore, short circuit of rotor laminations is also a common fault.

2.2.1.1 Stator faults

The stator faults are identified in terms of health and quality of the insulation between the turns and phases of the individual turns and coils inside the motor. Stator faults of induction motor represent 38% of total induction machine failures as shown in Figure 2.1. The stator faults, resulting in the opening or shorting of one or more stator phase windings as illustrated in Figure 2.2. The stator winding is subjected to various stresses due to high temperature, mechanical vibrations, and voltage spikes. The thermal stresses of induction motor cause insulation failures and short circuit winding faults. For every 10^o C increase above the stator winding temperature limit, the life of the insulation is reduced by 50%. Stator winding faults are often caused by insulation failure between two adjacent turns in a coil. This is called a turn-to-turn fault or shorted turn.

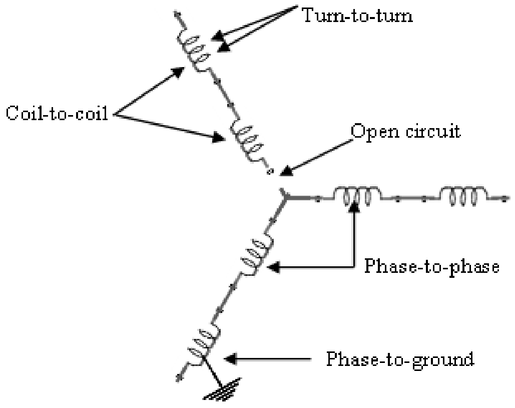


Figure 2.2. Possible failure modes in induction machine stator windings.

2.2.1.2 Rotor Faults

The failures in the rotor are motivated by a combination of various stresses such as electromagnetic, thermal, dynamic, environmental and mechanical which act on the rotor [37]. These faults such as a broken rotor bar, cracked rotor end-rings, short circuit of rotor laminations and open circuit [38-40] represent 10% of the most commonly reported faults as shown in Figure 2.1.

2.2.2 Mechanical Faults

Mechanical stresses are caused by overloads and sudden load changes, which can produce bearing faults and rotor bar breakage. These faults are referred to rotor faults because they are related to the moving parts of the machine. Mechanical faults in the rotor include eccentricity (static or dynamic) and misalignment faults. Stator eccentricity and core slacking are the major types of mechanical faults in the stator and these faults produce problems such as vibration and noise.

2.2.2.1 Bearing Faults

In the induction motor, the bearings on both sides of the rotor shaft allow the rotor to spin freely inside the stator. Rolling element bearings consist of two rings - an inner and an outer, between which a set of balls or rollers rotate in raceways. The temperature of a bearing should not exceed a certain amount in order to protect the grease and the bearing itself. A schematic view of a typical rolling element bearing is shown in Figure 2.3. Bearing faults [41-43] account for over 40% of all machine breakdowns. Bearing faults (inner raceway defects, outer raceway defects and ball defects) cause machine vibration. This vibration results in air gap eccentricity. The first signs of deterioration are noisy bearings. Any oscillations in air gap length can cause variation in flux density which can affect the machine inductances and generate harmonics in stator currents.

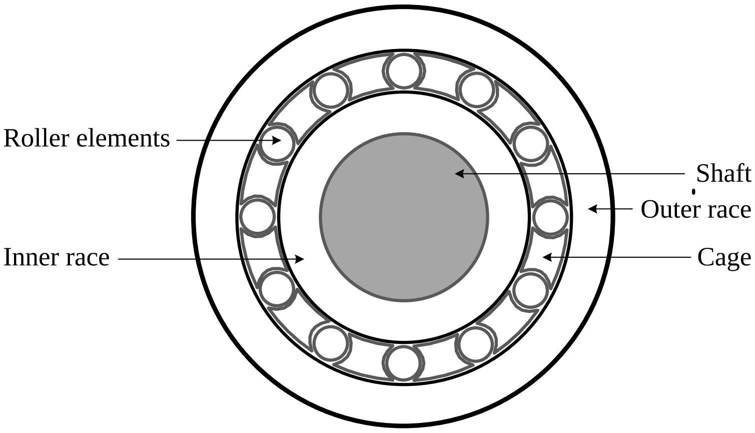


Figure 2.3. A schematic view of rolling element bearings

2.2.2.2 Eccentricity fault

The space between the stator and the rotor in the induction motor is called the air gap. If any damage happens to the bearing, the rotor becomes eccentric causing a degree of static or dynamic eccentricity. Static eccentricity can be caused by the incorrect positioning of the stator or the rotor [44, 45]. Dynamic eccentricity occurs when the centre of the rotor is not at the centre of rotation and the minimum air gap revolves with the rotor. Dynamic eccentricity could be caused by a bent shaft, mechanical resonances at critical speeds, or bearings and movement. The combined static and dynamic eccentricity is called mixed eccentricity. Bearing faults lead to air gap eccentricity and affects the resultant magnetic field. This also causes an increase in vibration as the shaft dynamics are affected by the altered air gap. For static eccentricity the position of minimum radial air gap length is fixed in space while dynamic eccentricity is a function of space and time (see Figure 2. 4 to Figure 2.6).

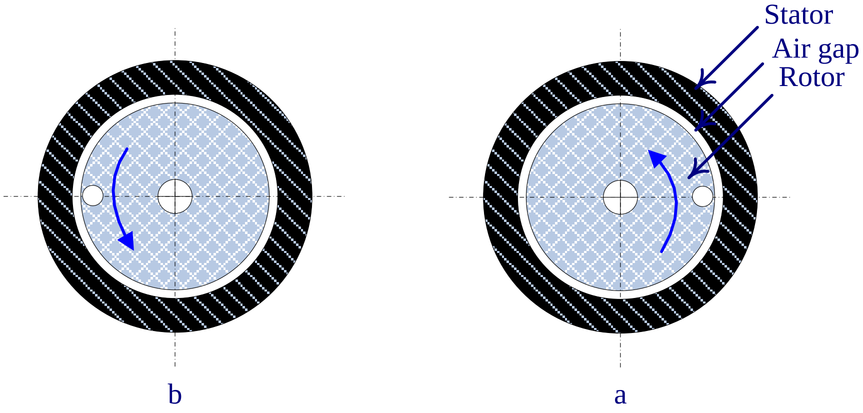


Figure 2. 4. Cross section of a healthy induction motor; (a) Initial position (b) After half a revolution.

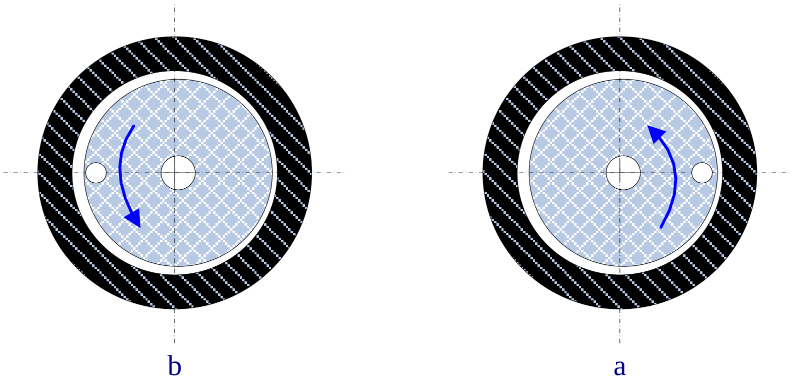


Figure 2.5. Static eccentricity; (a) Initial position (b) After half a revolution.

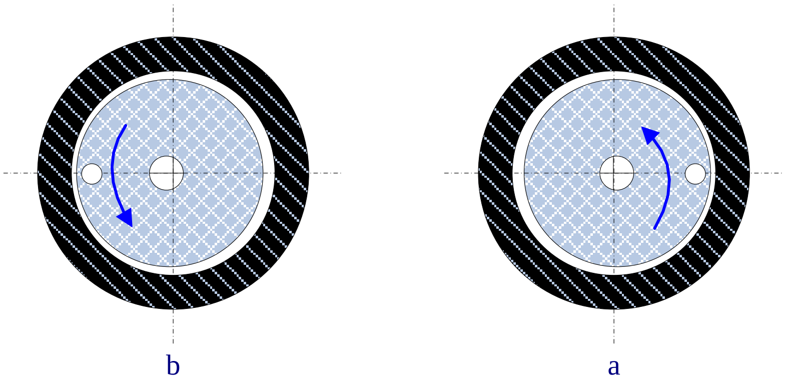


Figure 2.6. Dynamic eccentricity; (a) Initial position (b) After half a revolution.

2.2.3 Other Faults

A small number of failures in induction motors are linked to external factors. These faults can be caused by environmental, cooling, installation or manufacturing problems.

2.3 Fault detection Methods

There are numerous methods of induction motor fault diagnosis were developed in the last decades such as motor current signature analysis, temperature measurements, vibration monitoring, chemical analysis, artificial intelligence and stochastic optimisation techniques. These diagnostic methods may involve several different fields of science and technology.

2.3.1 Motor current signature analysis

MCSA is one of the most powerful methods of online fault diagnosis due to its low cost and simplicity [21]. Motor current signature analysis (MCSA) is based on current monitoring of an induction motor. The MSCA uses current spectrum of the machine for locating characteristic fault frequencies. Fast Fourier Transform and Wavelet transform are used to analyse motor current signature by identifying fault spectrum and extracting unique features for fault diagnosis.

Stator current contains unique fault frequency components that can be used for detection of various faults of motor. When a fault is present, the frequency spectrum of the line current becomes different from the healthy machine. The motor faults could be diagnosed through the comparison of recorded stator current signals and the reference signals in the frequency domain. MCSA can easily detect the common machine fault such as rotor fault, short winding fault, bearing fault, air gap eccentricity fault, etc [46, 47]. Schoen and et al. addressed the application of MSCA for detection of rolling-element bearing damage [48].

2.3.2 Artificial Intelligence diagnoses

The main idea behind Artificial Intelligence (AI) is to mimic natural human intelligence in the form of a computer program to tackle problems that are hard to solve by traditional methods

2.3.2.1 Neural Networks

Artificial neural networks are modelled on the neural connections in the human brain. Each artificial neuron accepts several inputs, applies preset weights to each input and generates a non-linear output based on the result. The neurons are connected in layers between the inputs and outputs. The application of artificial intelligence methods [49, 50], like neural networks, is rather easy to develop and to perform. Neural networks can be applied when the information about the process is obtained by measurements, which later can be used in the training procedures of neural nets. This method is an on-line technique and doesn't use a mathematical model. Neural detectors can be designed using the data acquired from simulation or experimental tests [22].

2.3.2.2 Fuzzy Logic

Fuzzy logic utilizes human knowledge by giving the fuzzy or linguistic descriptions a definite structure. A fuzzy logic approach may help to diagnose induction motor faults [51, 52]. In fact, fuzzy logic is reminiscent of human thinking processes and natural language enabling decisions to be made based on vague information. Fuzzy logic allows items to be described as having a certain membership degree in a set. This allows a computer, which is normally constrained to 1 and 0, to delve into the continuous realm.

2.3.3 Park's Vector Approach

This method is based on the visualization of the motor current Park's vector representation which is based on the identification of a specified current pattern obtained from the transformation of the three-phase stator currents to an equivalent two-phase system [53].

A healthy machine shows a perfect circle in Park's vector representation while an elliptical pattern is observed if the machine is faulty. Park's vector method can be used for diagnosing many types of induction motor faults such as air gap eccentricity, stator winding short-circuits, occurrence of rotor cage fractures, open wound rotor and bearing damages [54-56]. Cruz [53] has used Park's vector approach to detect several types of rotor faults. The main advantage of this method is that the change in the shape of the line current phasor can be clearly observed, making it easy to diagnose machine faults. The main disadvantage is that it is not effective for load faults and broken rotor bar faults.

2.3.4 Stochastic optimisation search methods

These optimisation techniques depend on the idea of neighbourhood search and there is no mathematical model of the system involved. Heuristic techniques seek near optimal solutions without being able to guarantee either feasibility or optimality of the solution [57]. The principles of heuristic techniques are easy to understand, implement and use. The heuristic optimisation is very popular in applications. Every method uses a different search rule of finding optimal or near optimal solutions. The space of all feasible solutions is called the search space. Each and every point in the search space represents one possible solution. Therefore each possible solution is represented by one point in the search space. In effect, this technique searches through the solution space and moves in the direction of a known solution until a suitable solution is found or time bound is elapsed.

Every stochastic method has different mechanism to escape from local minimum.

Many stochastic algorithms have been developed in recent years to address optimisation problems. The most popular algorithms are:

- Genetic Algorithms (GA)
- Ant Colony Optimisation (ACO)
- Simulated Annealing (SA)
- Particle Swarm Optimisation (PSO)
- Tabu Search (TS)
- Bacterial Forging Optimisation (BFO)

Many researchers have used these techniques; Zakaria [16] has used SA algorithm for induction machine fault detection. Ethny [58] has also used PSO and compare it with SA and BFO algorithms for induction machine fault identification. A tabu search algorithm has been used by Montane [59] for the vehicle routing problem. Cai [60] is applied GA for speed estimation of induction motor. In this study, GA, TS and SA algorithms were used because they are some of the most widely used algorithms and have proven to be very effective and robust in a wide range of applications. A brief description of the three algorithms used in this thesis is presented in the following subsection,

2.3.4.1 Tabu search

Tabu Search (TS) is a new a stochastic optimisation procedure and has traditionally been used for optimisation problems [61]. TS has the power to avoid being trapped in local minima by using a tabu list. This tabu list constitutes the short-term memory which records any repeated solutions as a forbidden move. At each iteration, a set of candidate moves is extracted from the neighbourhood for evaluation and the best move is selected as a new solution. If the new solution is not tabu, it is accepted as a current solution, even if it is not a better solution. The tabu list is then updated with a new set of solutions. The advanced mechanisms of TS include the uses of intensification and diversification; by using the intensification mechanism, the algorithm does a more comprehensive exploration of attractive regions which may direct it to a local optimal point and by using the diversification mechanism, the search is moved to previously unvisited regions to avoid cycling. This process of TS continues until the stopping criterion has been reached whereby the determined optimal solution for that specific problem is selected.

2.3.4.2 Genetic Algorithm

Genetic algorithm (GA) is a stochastic search technique inspired by the mechanism of evolution, and natural selection. GA is successful at avoiding local minima and has been proven to be effective in solving difficult combinatorial optimisation problems. Traditional optimisation techniques use a single candidate and use repeated search techniques. However, the GA approach searches a population of candidates across several areas of a solution space simultaneously. The population consists of individuals or chromosomes which can be represented by strings of real or binary numbers. This population represents points in the solution space. The chromosomes evolve through successive iterations called generations. A new set of solutions, called offsprings, are created in a new generation. The basic operators in GA are selection, crossover and mutation. During each generation, the individuals are evaluated according to their objective and fitness functions. Following several generations, the algorithm then converges to the best chromosome that represents the optimal or near optimal to the problem.

2.3.4.3 Simulated Annealing

Simulated annealing (SA) is an optimisation technique which has been widely used in large combinatorial optimisation problems. SA mimics the annealing process for crystalline solids. The annealing process starts with melting the solid by heat treatment and slowly decreasing the annealing temperature. The SA algorithm views the cost function being minimised as equivalent to the energy state of a physical system, and the process of reaching equilibrium is equivalent to repeatedly accepting or rejecting changes in energy from one state to another. The algorithm starts from a randomly generated initial point and simulates a walk through the solution space; a candidate configuration is accepted if its cost is less than the current configuration, while deteriorating steps are only accepted with a certain probability. As the temperature parameter is decreasing, the algorithm accepting only good solutions until converges to a solution very close to optimality [62].

2.4 Summary

The most prevalent faults in induction motors are described in this chapter. The use of electric motors is becoming ever more common in various industries, resulting in an increased demand for fault detection methods. This chapter has provided a general review of existing induction motor fault detection methods. Relevant information about stochastic search algorithms used in this study has also been presented. In the next chapter, the three stochastic algorithms (Genetic Algorithms, Tabu Search and Simulated Annealing) will be explained in more details.

CHAPTER 3

STOCHASTIC OPTIMISATION ALGORITHMS

3.1 Introduction

This chapter introduces the concepts involved in the various stochastic optimisation algorithms (SOA) that have been investigated. Optimisation problems are defined by a set of solutions and an objective function associated with each solution. The goal when addressing an optimisation problem is to optimise the objective function to find solutions that are optimal or near-optimal in a reasonable amount of time. Conventional optimisation methods suffer from the problem of local minima trapping. This problem can be circumvented by using stochastic optimisation methods such as Genetic Algorithm (GA), Simulated Annealing (SA) and Tabu Search (TS). Many efforts have been directed toward developing efficient heuristic algorithms. SOA have been applied to a wide variety of combinatorial optimisation problems with great success. Several stochastic search techniques have been proposed in the literature. Figure 3.1 shows a simple classification of some stochastic optimisation methods according to their algorithmic structure [63]. In this work, three standard stochastic optimisation methods: genetic algorithm, tabu search and simulated annealing are used to identify induction machine winding faults. These stochastic algorithms are implemented using the MATLAB language on a PC.

When using stochastic algorithms, the search process is repeated until a stopping criterion has been reached. There are various kinds of termination conditions such as the use of a fixed number of iterations, fixed number of generations, fixed amount of time, when the objective value reaches a pre-specified value, or when there is no improvement of the objective function. The simplest form of stopping criterion is a fixed number of iterations or generations. The search is terminated once a preset maximum number of iterations have been reached. The best solution found in this period will be the result of the search.

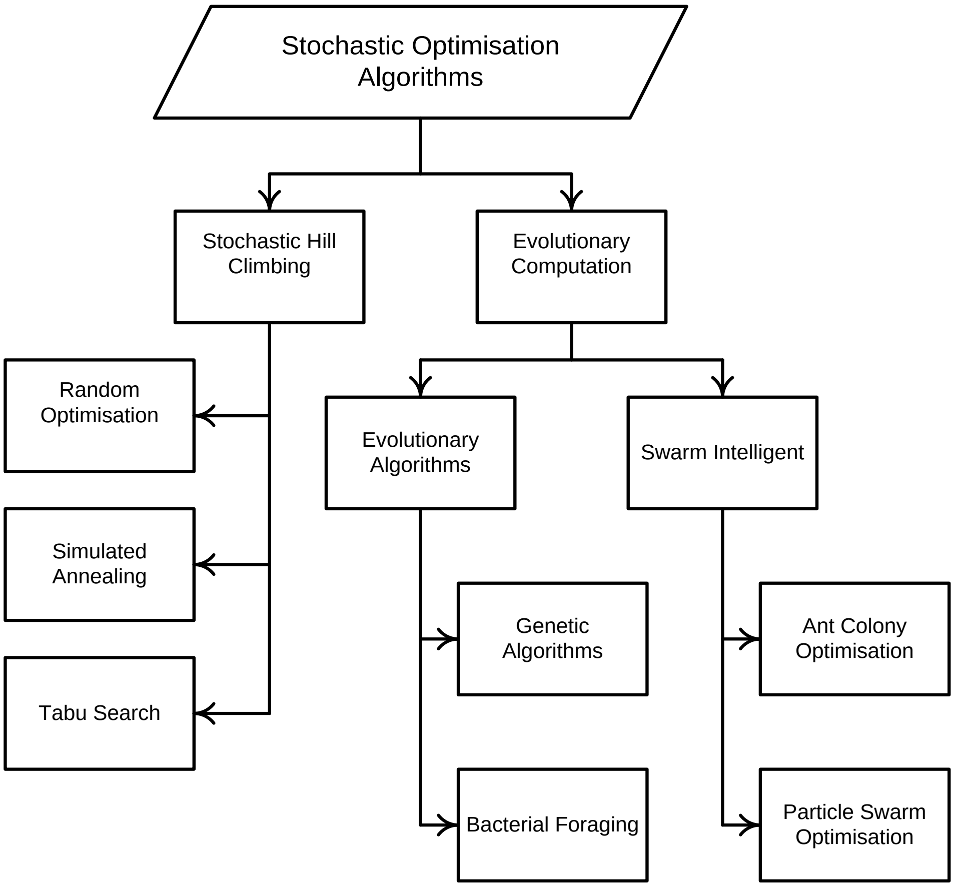


Figure 3.1. Classification of stochastic optimisation algorithms [63].

3.2 Genetic Algorithm

3.2.1 Introduction

Genetic algorithm (GA) is a stochastic search technique inspired by the mechanisms of evolution and natural selection. Like other evolutionary algorithms, GA is a population-based metaheuristic* optimisation algorithm that uses biology-inspired mechanisms such as mutation, crossover, natural selection and survival of the fittest (Figure 3.2). GA was introduced in 1975 by a team led by John Holland and was later developed by De Jong Goldberg and many other researchers. There are many different GA algorithms but the basic idea, which is based on Darwin's theory of survival of the fittest, is the same [64, 65]. GA is successful in avoiding local minima and has proved to be effective in solving difficult combinatorial optimisation problems. GA is efficient, reliable and robust. It finds widespread applications in system optimisation problems in science, economy and many other fields. GA has been successfully applied to numerous combinatorial problems such as the travelling salesman problem [66], scheduling problems [67], graph colouring [68] and many others. Traditional optimisation techniques use a single candidate and repeated search techniques. The GA approach on the other hand searches a population of candidates across several areas of a solution space simultaneously. The population consists of individuals or chromosomes which can be represented by strings of real or binary numbers. GA uses fitness functions for evaluation rather than derivatives. This technique is generally able to find the optimal or near-optimal solutions to the considered optimisation problems. One of the drawbacks of GA is their complex computational requirements as they can be very slow in some applications. This problem can be overcome by using faster computers. Sastry et al. give a good introduction to genetic algorithms [69]. In this section, the basic principle of genetic algorithm is described, followed by a simple example showing the use of a genetic algorithm to find the optimum parameters of a given problem.

* The term metaheuristic derives from the composition of two Greek words: heuristic means "to find" while the prefix meta means "beyond, in an upper level." It refers to the set of strategies that guide the iterative search process.

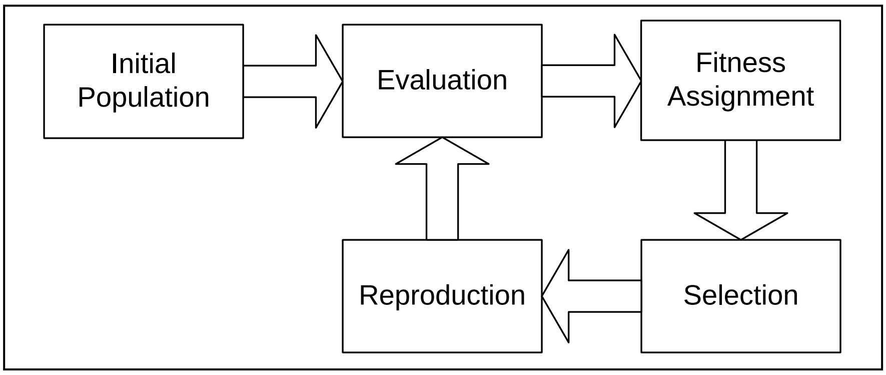


Figure 3.2. The basic cycle of Evolutionary Algorithms.

3.2.2 Basic concept of genetic algorithm

In Genetic Algorithm only two kinds of operations are executed: evolutionary operations (selection) and genetic operations (crossovers and mutations). The GA starts with a random population of potential individuals, each representing one possible solution to the problem. A population is made up of a set of individuals, and evolution from one generation to the next takes place by the deletion of existing individuals and the creation of new ones. During each generation, the individuals are evaluated according to the objective and fitness functions. After obtaining the fitness of all individuals, a selection process is used to choose individuals for reproduction. Individuals with higher fitness should have a higher probability of being selected as parents so that the more successful individuals will have more chances to mate and generate offspring. The least fit individuals in each population are then replaced by the offspring so that the population size remains constant and another generation starts. Through an iterative process, the population evolves towards better regions of the search space. After many generations, the algorithm reaches convergence towards the best chromosome, or the individual which signifies the optimal solution or the nearest optimal solution to the problem [70]. The algorithm stops once the termination criteria is met. These processes are described in more detail in the following subsections. The basic steps of simple genetic algorithm can be described as follows:

Step 1: Generate an initial population randomly.

Step 2: Evaluate the fitness value of each individual in the population.

Step 3: Select parent individuals from a population that have higher fitness values to generate new offspring.

Step 4: Crossovers and mutations are applied to the selected offspring.

Step 5: Replace the worst part of the population by new offspring.

Step 6: If the stopping criterion has been reached, stop; ***otherwise*** save the best solution and proceed to ***Step 2***.

Figure 3.3 shows the flowchart for the GA algorithm.

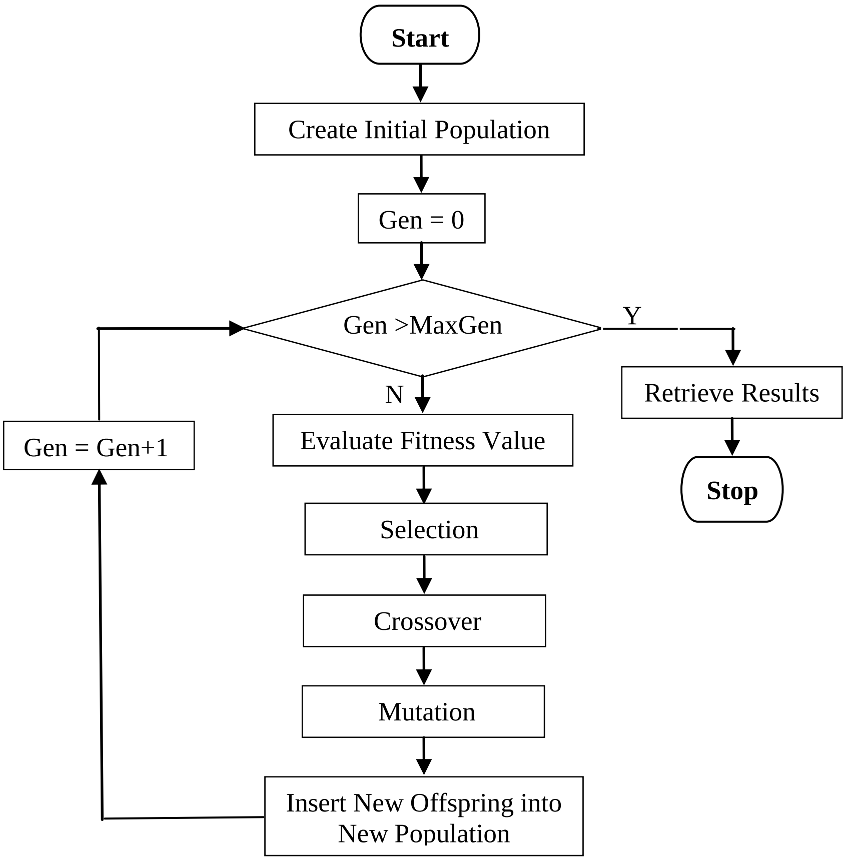


Figure 3.3. GA flowchart.

3.2.2.1 Initialisation

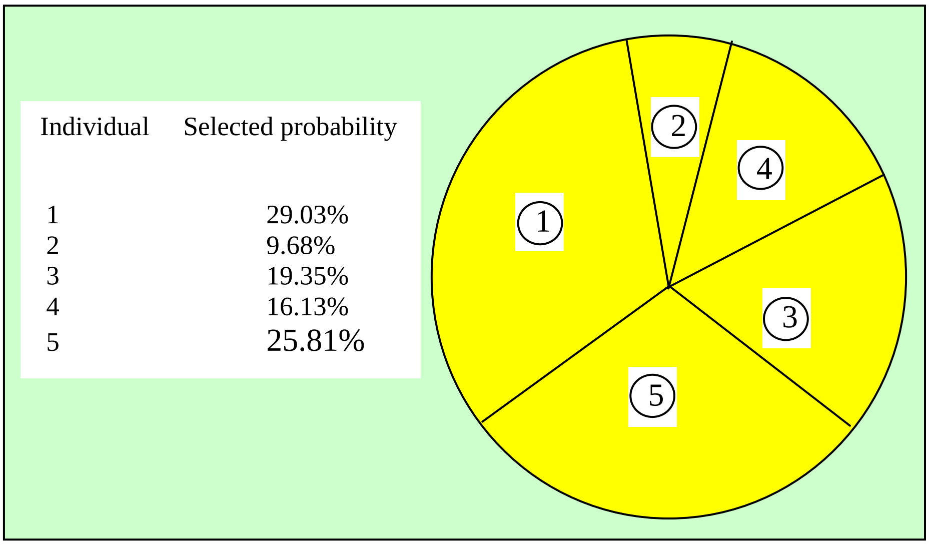
To start the GA algorithm, a specified number of individuals, a maximum number of generations and the type of chromosome coding are chosen. The chromosomes are encoded as either binary or real numbers. However, by using binary strings, the chromosomes may become long, due to the number of digits that may be required to form one single variable. By using real numbers, the chromosome length is reduced and becomes easier to understand, as each allele or gene of the chromosome or individual is represented by its real value. The initial set of random solutions is known as the initial population. The number of individuals in the population is called the population size which is usually recommended to be between 30 and 100 [71]. It has been noted that too large a population slows down the optimisation, while a small population does not utilise the genetic operators effectively. The crossover and mutation rates for the reproduction function should also be defined as should the selection method.

3.2.2.2 Objective function and fitness functions

The objective function is a function associated with an optimisation problem that would determine the merit of the solution. For example, the function indicates that the best individuals would be the ones with the lowest value of objective function in the problem of minimisation and vice versa in the case of maximisation. Every individual in the population has its gene representation, called its code, and performance evaluation, called its fitness value; the fitness calculation is a measure of how good a particular solution to the problem is, so that different solutions can be compared based on the values obtained from the fitness function. This means that individuals with higher fitness value will have higher probability of being selected as candidates for the next generation.

3.2.2.3 Selection

A selection operator is executed to choose parents from the current population (set of solutions) for generating offspring. This process is based upon their fitness values. This is achieved by assigning better solutions a higher probability of being chosen for recombination. It is also very important to allow a few less-fit individuals to increase the diversity of the population. A number of different selection strategies have been implemented in different algorithms including roulette wheels, local tournaments, various ranking schemes, etc. A simple reproduction operator is the roulette wheel selection where each individual in a population has a roulette wheel slot sized in proportion to its fitness; for example, if there are five individuals in the population with fitness values of 9, 3, 6, 5 and 8, the corresponding probabilities for these individuals are 29.03%, 9.68%, 19.35%, 16.13% and 25.81%, respectively. Figure 3.4 shows the selection probability for the 5 individuals. Individual 1 is the most fit individual and occupies the largest interval, whereas individual 2 is the weakest and correspondingly occupies the smallest interval. In order to select an individual, a random number is generated within the range (0, Sum). The variable Sum is determined as the sum of the row fitness values over all the individuals in the current population [72]. The individual whose segment of the wheel spans the number is selected. This process is repeated several times until the required number of individuals has been selected with the selection probability of an individual being proportional to its fitness value [73].



Individual	Selected probability
1	29.03%
2	9.68%
3	19.35%
4	16.13%
5	25.81%

Figure 3.4. An example of roulette wheel selection.

3.2.2.4 Crossover

The process of crossover, which represents mating (recombination) of two parent individuals, is carried out by exchanging parts of their chromosomes to create new offspring. Each offspring shares genes with its parents. The process of crossover is applied with a certain probability referred to as the crossover rate. The crossover rate generally ranges from 0.25 to 0.95. The crossover operation is necessary to ensure convergence of the GA to an optimal solution. After selecting a pair of parents, the algorithm implements crossover only if a randomly generated variable is greater than the crossover rate, otherwise, the parents remain unchanged. Each offspring inherits genes from its parents. This is achieved by slicing the chromosomes and crossing over their genes. The simplest and most popular implementations of this process are one-point crossover and two-point crossover where the chromosome string of each parent organism is randomly split at one or two points, respectively. The crossover point is chosen at random in the range of [1, L_{st} -1] where L_{st} is the length of the string. To generate new offspring using one-point crossover, a cut point is chosen randomly and the genetic information to the left of the cut of point in parent A are combined with the genetic information to the right of the cut of point in parent B. A second offspring can also be generated by using the right hand side string of parent A with the left hand side string of parent B. For a two-point crossover, two cut of points are set randomly and the section between the two points is taken from one parent and combined with the outer sections from the other parent. In the following example, the crossover point is set after the second bit in the case of one-point crossover. In the case of two-point crossover, the two crossover points are set after the second and fifth bits respectively. This operation is illustrated in Figure 3.5.

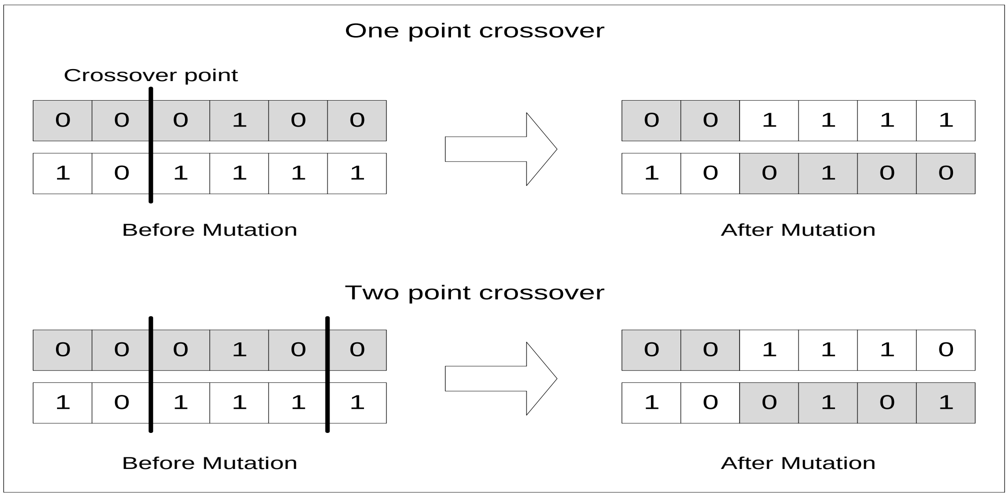


Figure 3.5. The GA crossover operation.

3.2.2.5 Mutation

Mutation takes place by randomly changing a few bits in the chromosome of the resulting offspring. This prevents the solutions from falling into a local optimum and can also help in exploring new regions of the solution space. The mutation operation does not occur as frequently as the crossover function. This is achieved by using a low mutation probability (a value fixed throughout the whole search process) that represents how often parts of a chromosome will be mutated. First, the mutation point is selected randomly (Figure 3.6). After mutation, the resulting solutions become the current population. In the case of binary encoding, mutation is simulated by flipping bits at random, using a low probability in the range [0.001- 0.05]; for this work, a mutation probability of 0.05 was chosen. A gene can be mutated by swapping its value from 0 to 1 or 1 to 0. For real-value encoding, the mutation operator can be implemented by random replacement (i.e., the original value is replaced with a new randomly generated value). An illustration of the process of mutation in GA is shown in Figure 3.6. The mutation operator is more important at the final stages of the optimisation process when the majority of the individuals in the population possess similar qualities.

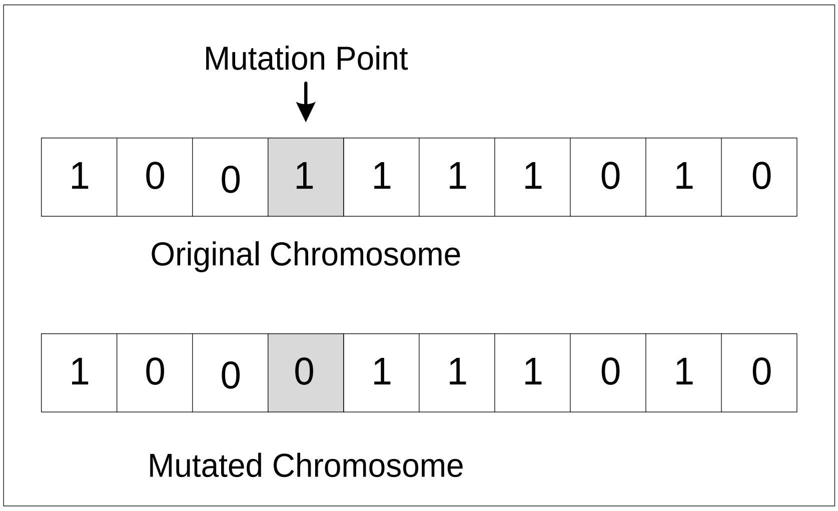


Figure 3.6. Illustration of mutation in Genetic Algorithms.

3.2.3 Reinsertion

The fitness of the new individuals is determined once a new population has been produced. A generation gap may occur where fewer individuals are produced and this may cause a difference between the old and new population sizes. Some of the new individuals need to be reinserted in the old population to maintain the original population size.

3.2.4 Application of GA to simple example

In this Section, the GA algorithm was implemented using Matlab on the following function (Equation 3.1) which has one global minimum of (0.846) at *x* = 2.388 as can be seen in Figure 3.7.

$$f(x) = 0.0116x^4 - 0.2473x^3 + 1.7927x^2 - 4.9426x + 5.4150 \tag{3.1}$$

The task of the GA algorithm is to optimise the variable x to minimise the objective function and to confirm the ability of GA to avoid becoming trapped in the local minimum of 1.5 at *x* = 8. The search range for variable x is set from -20 to 20. The Matlab code for this exercise is given in Appendix A1. Figure 3.8 shows that the GA algorithm terminates after 50 generations at the global minimum. The computational time was 1.965 sec and the value of *x* is 2.388.

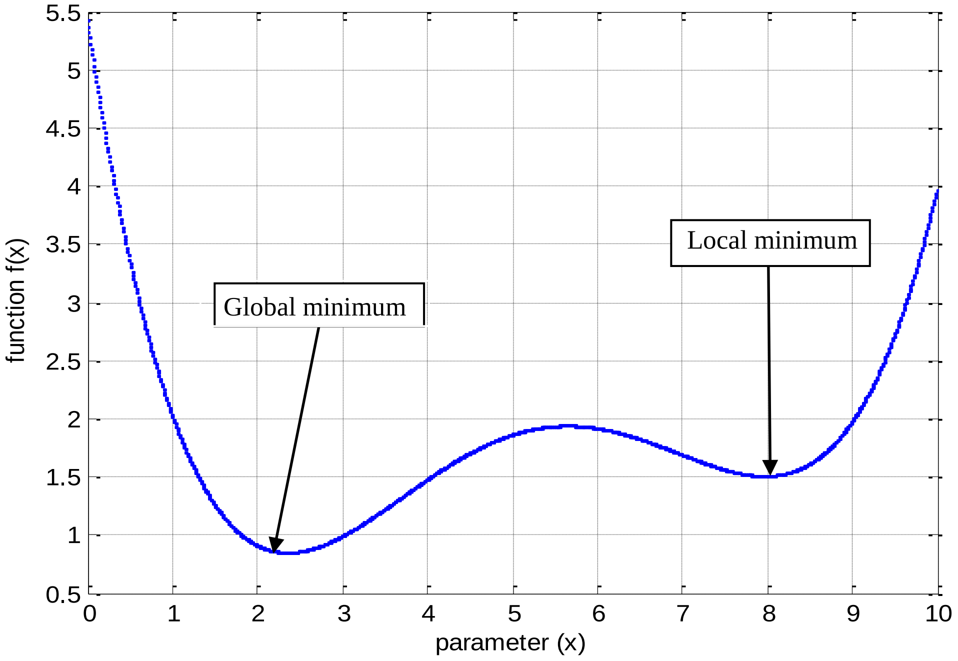


Figure 3.7. Function with a local minimum.

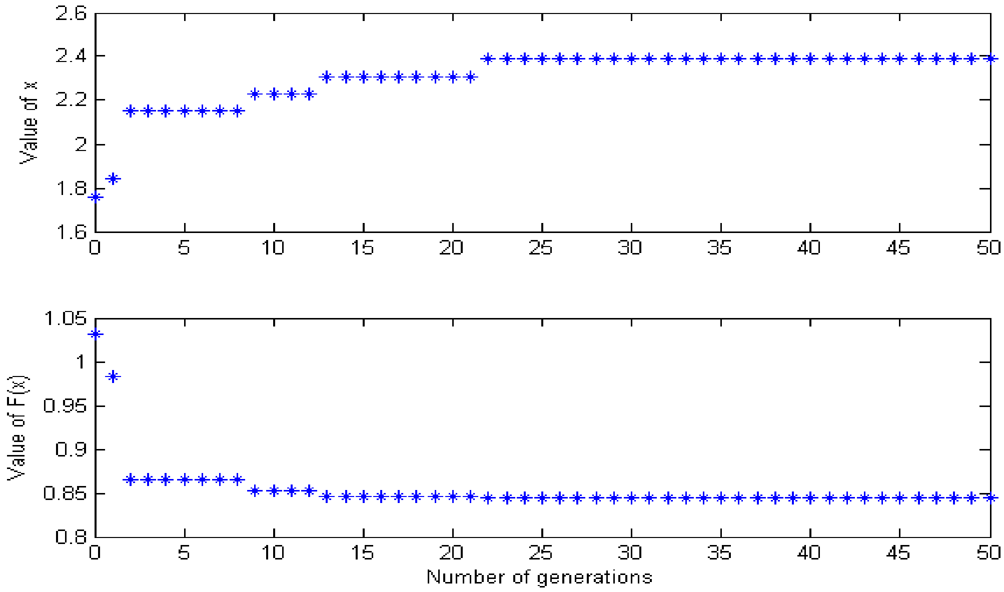


Figure 3.8. Value of potential solution and objective function obtained by GA.

3.3 Tabu Search

3.3.1 Introduction

Tabu Search (TS) is a stochastic optimisation procedure originally proposed by F. Glover in the early 1980s. TS is a non-random metaheuristic algorithm [74] which has been applied with success to a number of complex problems in science and engineering [75, 76]. TS has traditionally been used for combinatorial optimisation problems [61] and differs from the other optimisation techniques in the use of memory. A simple tabu search algorithm is based on short-term memory with a tabu list and an aspiration criteria as well as more advanced techniques such as intensification and diversification. TS has the power to avoid being trapped in local minima by using a tabu list. This tabu list constitutes the short-term memory which records any repeated solutions as forbidden moves, thus preventing the algorithm from cycling to a recently visited solution. The use of memory in the search process is a major factor in the success of the tabu search method. The role of the memory can change as the algorithm proceeds. The efficiency of the TS method is strongly dependent on the proper selection of its parameters, i.e. initial solution, neighbourhood, tabu list and stopping condition.

3.3.2 Tabu Search Procedure

The basic idea here is to randomly generate an initial solution using a normal distribution with a suitable mean and standard deviation; then the procedure moves to a new solution in the neighbourhood that improves the objective function (i.e. a set of moves are defined that may be applied to the current solution to produce a list of candidate solutions using a Mean Gaussian Distribution with a suitable variance). At each iteration, a set of candidate moves is extracted from the neighbourhood for evaluation. If a move produces the best overall value for the objective function, it becomes the candidate move and is selected as the new solution. If the new solution is not tabu, it is accepted as a current solution for the next iteration. Otherwise, its aspiration criterion is checked. This criterion is introduced in tabu search routines to determine when tabu activation rules can be overridden. Tabu may prohibit moves leading to attractive solutions even when there is no danger of cycling; this problem can be overcome by using aspiration criteria. The aspiration criterion is performed to override the tabu status of a move; however, if the new solution is tabu and the aspiration criterion is not satisfied, the move is not performed. The simplest method of achieving this is to allow the move even if it is tabu when the current solution is better than the best solution.

If the condition of the aspiration criterion is satisfied, it becomes the current solution. The tabu list is always updated with a new set of solutions. After a predefined number of iterations, old moves are removed from the tabu list. The overall procedure iterates until one of the stopping criteria is satisfied. The final state is then the optimal solution selected for that particular problem. The general TS procedure is defined below.

Step 1: Generate a random initial solution $x_{initial}$, Set x_{best} = $x_{initial}$ = $x_{current}$. Assign the tabu list memory and set the aspiration criterion.

Step 2: Generate trial solutions in the neighbourhood of the current solution *N(X)*.

Step 3: Compute the objective function for the trial solutions and compare them to the best solution objective function value. If a better solution is obtained then x_{best} = x_{trial} .

Step 4: If x_{trial} is not tabu, set $x_{current} = x_{trial}$, update the tabu list and go to ***Step 7***. If x_{trial} is tabu, go to ***Step 5***.

Step 5: If the condition of the aspiration criterion is satisfied, then the tabu status is overridden, set $x_{current} = x_{trial}$, update tabu list and go to ***Step 7***. **Otherwise**, go to

Step 6: Go to ***Step 3*** to check all trial solutions.

Step 7: Check whether the stopping criterion is satisfied. If the answer is yes (e.g i > max iteration), then stop. **Otherwise**, go to ***Step 2.***

These steps are illustrated in the flowchart of Figure 3.9

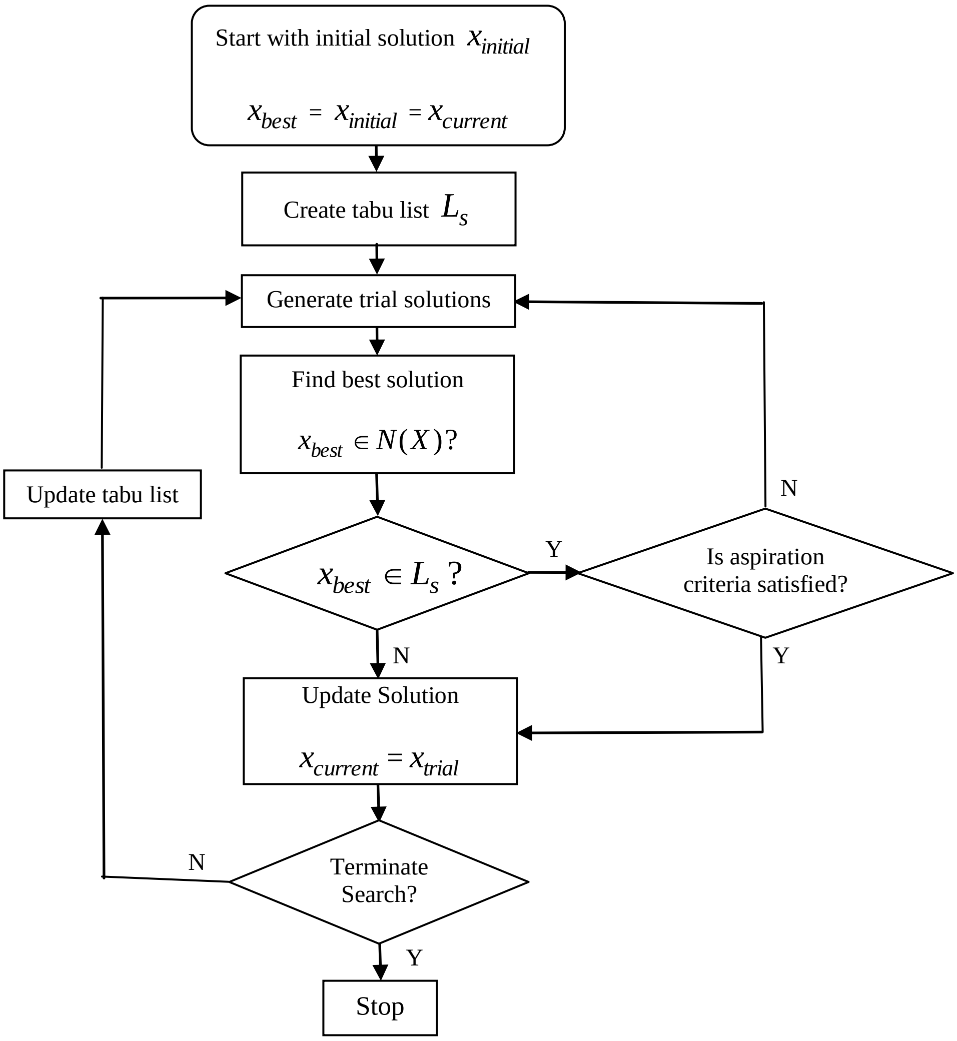


Figure 3.9. The flowchart for tabu search algorithm.

3.3.3 The Tabu List

One of the most important aspects of tabu search is the tabu list. The tabu search algorithm overcomes the problem of entrapment in a local optimum by avoiding a previously visited solution. In this way, the algorithm is forced to explore new regions of the solution space. This is achieved by using memory restriction referred to as a tabu list. The tabu list is updated after each iteration by adding a new element to the bottom of the list. When the tabu list becomes full, the oldest element on the list drops from the top. The duration for which a solution remains on the tabu list is called the tabu tenure. The length of the tabu list (L_s) is an important parameter in most tabu search algorithms. Short tabu lists may not prevent cycling (i.e. entrapment in a local optimum), resulting in information loss, while long tabu lists may excessively prevent neighbourhood searches so that moves are limited to some extent. Glover [61] suggested that 7 would be a good value for L; this value has been empirically found to be effective in many applications. Anderson et al. [77] reported that list lengths can be between 7 and 15.

3.3.4 Intensification and diversification

The tabu search alternates between intensification and diversification strategies through the use of memory structures and candidate lists [61]. Long-term memory helps to implement diversification and intensification mechanisms. Both intensification and diversification are considered advanced functions of tabu search and they can be added to a basic tabu search with a short memory and aspiration criterion. The intensification mechanism helps the tabu search to explore specific areas more thoroughly while the diversification mechanism moves the search to unvisited regions of the search space in order to avoid cycling.

3.3.5 Application of TS to a simple example

In this section, the TS algorithm described above is implemented in Matlab to locate the global minimum of the function expressed by Equation (3.1).

The algorithm was run for 50 iterations using a tabu list size of seven and a neighbourhood size of eight. Figure 3.10 shows the accepted configurations. The final function value obtained is for a global minimum of 0.844 at x = 2.3642, with 1.133 sec computational time. By using the tabu list, the algorithm can avoid getting stuck in a local minimum. Matlab code for this exercise is given in Appendix B1.

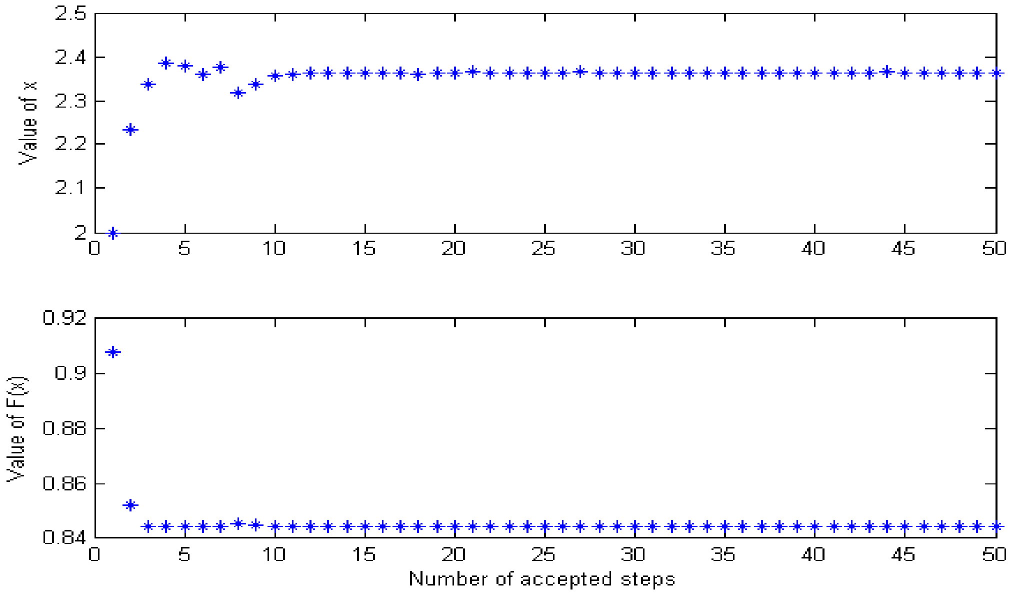


Figure 3.10. Value of potential solution and objective function obtained by TS.

3.4 Simulated annealing

3.4.1 Introduction

Simulated annealing (SA) is one of the most commonly used stochastic global optimisation algorithms. The SA algorithm, which originally took its motivation from the simulation of the physical annealing of melted metals, was introduced by Kirkpatrick et al. in 1983 [78]. SA mimics the annealing process for crystalline solids. The annealing process starts by melting the solid by heat treatment and cooling it down by slowly decreasing the annealing temperature. The SA algorithm views the cost function being minimised as equivalent to the energy state of a physical system, and the process of reaching equilibrium is equivalent to repeatedly accepting or rejecting changes in energy from one state to another. Typically, the temperature is high at the initial stage and decreases as moves are accepted. The adjustment of the temperature should be gradual enough to ensure that the global minimum is reached slowly; otherwise, the annealing process may converge to a local minimum [79]. The drawback of applying SA is that it requires a long period of computation time.

3.4.2 Basic concepts of simulated annealing

The algorithm starts from a randomly generated initial point and simulates a walk through the solution space; a candidate configuration is accepted if its cost is less than the current configuration, while deteriorating steps are only accepted with a certain probability. The probability function used has an exponential decay which mimics the cooling of a crystal solution. The ability to perform uphill moves allows simulated annealing to avoid the system getting stuck in a meta-stable state representing a local minimum of energy. The process contains the following two stages:

- The starting value for temperature needs to be high enough for the solid to melt.
- The temperature then needs to be decreased carefully until the ground solid state is reached.

The difference between the evaluation value of the current solution and the evaluation value of the new solution is called the energy difference ΔE (considered here as cost function). If ΔE is negative or equal to zero, the new solution is accepted and replaces the current solution. If the difference is positive, the new solution may be accepted according to the Metropolis acceptance probability which is lowered gradually as the algorithm proceeds [62]. The following probability is calculated in performing the acceptance test,

$$P_{m\,=}\ \exp\left[\frac{\Delta E}{K_B T}\right] \tag{3.2}$$

Where T represents the current temperature value and K_B is a physical constant known as Boltzmann constant. The temperature is decreased only if the candidate solution is accepted. The total process of SA algorithm is repeated until a frozen state is achieved at T = 0. If the cooling is not performed sufficiently slowly, the solid will be frozen into a meta-stable state rather than into a ground state [62]. The procedure is finished when the searching process of the last temperature is over. In the SA algorithm, each trial involves the following steps:

Step 1: Select an initial temperature and set the current configuration.

Step 2: Generate a new configuration from the neighbourhood using a random perturbation.

Step 3: Calculate the difference between the costs of the current and the new configurations.

Step 4: Decide whether the new configuration should be accepted using the Acceptance probability (P_A) and Swap probability (P_s). Replace the current configuration with the new one if this is the case.

Step 5: Reduce the temperature according to the cooling scheme and return to ***Step 2***. The process then repeats itself until a minimum criterion or some other limit such as time or maximum iterations is met. In this way, the final configuration with minimum cost can be obtained.

The basic flowchart of the Simulated Annealing algorithm used throughout this research is shown in Figure 3.11.

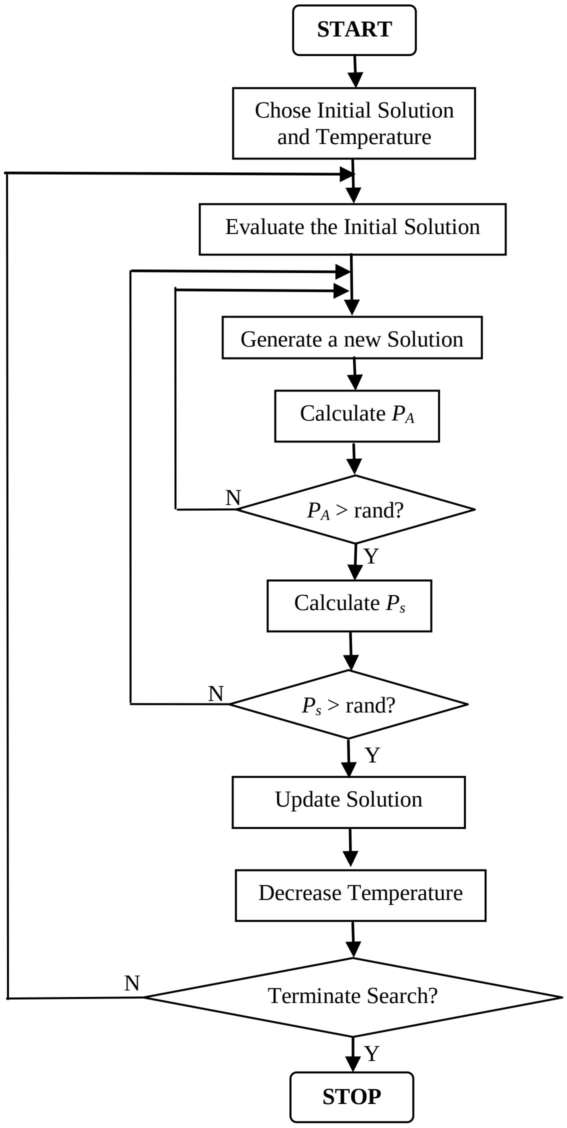


Figure 3.11. The flowchart of the Simulated Annealing algorithm.

3.4.3 SA procedure

SA starts at an initial random state (configuration) X and an initial temperature T_s equal to the cooling temperature T. The algorithm then generates a new configuration Y by applying a perturbation mechanism in the neighbourhood of the first configuration X. The perturbation mechanism is a method of exploring the neighbourhood of the current solution by creating small changes in the current solution. At each iteration, the objective function values for the two solutions (the current solution and a newly generated solution) are compared. Improving solutions are always accepted, to increase the rate of convergence; the configurations must be within the allowed solution space and are defined by the relation,

$$range = \sqrt{\sum_{N=1,2,...} \left[\frac{(Y_{N\max} - Y_{N\min})^2}{N} \right]} \tag{3.3}$$

Where *N* is the number of solution parameters.

The displacement, which is the distance between the two configurations *X* and *Y*, is calculated as follows,

$$displacement = \sqrt{\sum_{N=1,2,....} \left(\frac{(Y_N - X_N)}{N} \right)} \tag{3.4}$$

The acceptance probability (P_A) is then calculated according to the relation,

$$P_A = \exp\left[-\left(\frac{displacement}{range} \right) \times \left(\frac{T_S}{T} \right) \right] \tag{3.5}$$

The P_A is compared with a randomly generated number in the interval [0-1]. If the random number is greater than the acceptance probability, the solution *Y* is rejected and the generation mechanism is repeated. When the random number is smaller than the acceptance probability, the solution *Y* is accepted. When the annealing temperature *T* is high, there is a high probability of all values of displacement accepting any potential solution in the search space. The ability to perform uphill moves allows simulated annealing to avoid being trapped in a local optimal. As cooling proceeds, *T* is reduced according to annealing temperature schedule and P_A is reduced for a given displacement. The search is then concentrated on potential solutions which are very close to the existing solution.

The calculation error ΔE between the *X* and *Y* configurations **is** used to calculate the swap probability function (P_s),

$$P_S = \frac{1}{\left(1 + \exp\left[\left(\frac{E_{(n+1)} - E_n}{E_n}\right) \times \left(\frac{T_S}{T}\right)\right]\right)} \tag{3. 6}$$

where $E_{(n+1)}$ corresponds to configuration Y and E_n corresponds to configuration X.

The P_s is then compared to a random number in the interval [0 -1]. If the random number is greater than P_s, the original solution X is retained, and if it less than P_s, the new solution Y is accepted and X is replaced by Y. At the same time E_n is replaced by $E_{(n+1)}$ as the new objective function value and the process starts again. When $E_{(n+1)} < E_n$, the swap probability is greater than 0.5. At a lower temperature, the P_s tends towards 1, and any improved solution is generally accepted. In the case of $E_{(n+1)} > E_n$, the swap probability falls below 0.5. So, there is always a probability that a worse solution will be swapped. However, with the decrease in the annealing temperature, the probability of such an occurrence is reduced.

3.4.4 Cooling schedule

The performance of SA is very sensitive to the choice of cooling schedule [80]. The algorithm begins with a high initial temperature T_s which is then lowered gradually until convergence to steady state occurs. T_s must be high enough to allow moves to almost neighbourhood state. The acceptance probability P_A is proportional to the temperature T of the annealing process. Initially, when the temperature is high, the algorithm is likely to accept all solutions, while a low temperature only allows the acceptance of better-quality solutions; finally, when the temperature tends to zero, no deteriorations are accepted. If an appropriate cooling schedule is used, the SA algorithm will reach a stable status with low system cost and a globally optimal solution. The SA process requires an initial temperature variable T_s and a cooling schedule. Many cooling schedules exist in the literature to implement SA [76, 78, 81]. The simplest and most common temperature decrement rule is:

$$T_{new} = \alpha T_{old} \tag{3.7}$$

where α is the temperature reduction rate (close to, but smaller than 1). This decrement equation was used in this work. The major drawback of this decrement function is its slow convergence, especially when the controlling temperature becomes very small. Therefore, many alternative cooling temperature schedules have been suggested in the literature [82, 83]. German and German [83] derive another cooling schedule which is inversely proportional to a logarithmic function of time

$$T(k) = \frac{T_s}{\ln(k)} \tag{3.8}$$

where *T(k)* is the temperature at iteration *k*, *k* =1, 2, 3, ...etc.

The initial temperature is often determined by trial and error [81]. The logarithmic cooling schedule is very slow and therefore requires a long time to converge to the globally optimal solution. In order to accelerate the convergence rate of SA, a faster annealing schedule was proposed by Szu and Hartley [82] which is inversely proportional to the iteration number, as follows:

$$T(k) = \frac{T_s}{k} \tag{3. 9}$$

3.4.5 Application of SA to simple examples

To show the capability of SA, the SA algorithm is implemented on the function given by Equation (3.1) in the Matlab software environment.

The initial temperature T_s was set as 40 and the reduction function is, $T_{new} = \alpha T_{old}$ (refer to code in appendix C1). Figure 3.12 shows the accepted configurations. The final function value obtained is the global minimum of 0.844 (*x* = 2.353), with 1.216 sec computational time.

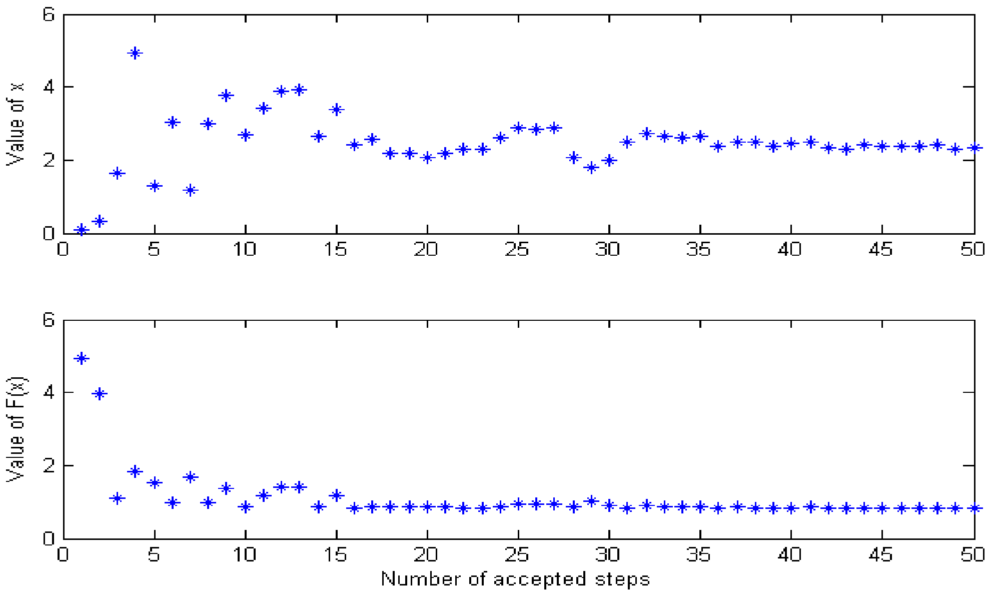


Figure 3.12. Value of potential solution and objective function obtained by SA.

3.5 Summary

In this chapter, three stochastic optimisation algorithms, GA, TS and SA, have been briefly described, and their application to find the global minimum of a mathematical function has been demonstrated. The main features of these algorithms are an ability to avoid becoming trapped at local minima and an ability to find an optimal solution, even for complex problems. Every algorithm has a different mechanism to avoid cycling. In the next chapters, these three algorithms will be used to identify different induction machine winding faults.

CHAPTER 4

CONDITION MONITORING SCHEME, EXPERIMENTAL SET-UP AND TEST DATA

4.1 Introduction

The previous chapter has illustrated the basic concepts of the three standard stochastic algorithms GA, TS and SA. This chapter gives details about the condition monitoring method presented in this thesis and the experimental set-up that were used to gather the required data (terminal voltages, stator currents and rotor speed) for the parameter estimation process.

A series of experiments were conducted to collect the data needed to verify the technique proposed for parameter identification and fault detection. Tests were carried out under steady-state operating conditions for a directly connected induction machine. A wound rotor type was used in the investigation. Rotor speed was measured using a digital tachometer. Stator currents and voltages were measured using current probe amplifiers and differential voltage probes, respectively. The current and voltage waveforms were captured using a digital oscilloscope. A general description of the experimental rig used in this investigation is given below.

4.2 Condition monitoring scheme

The proposed condition monitoring scheme is shown schematically in Figure 4.1. Experimental measurements of the three-phase stator voltages (V_A,V_B,V_C), three-phase stator currents (I_A, I_B, I_C) and speed of electric machine (ω_r) are recorded by using a digital oscilloscope and saved in the computer memory. The recorded three-phase voltages and the rotor speed are fed to the mathematical ABCabc model in order to calculate the stator currents (I_{sA}, I_{sB}, I_{sC}). As is shown in Figure 4.2, these stator currents are compared with the actual recorded stator currents (I_A, I_B, I_C) to produce a set of current errors that are integrated and summed to give an overall calculation error that is considered to be a cost function. Fault identification is then implemented to minimise the cost function in order to predict the machine condition. This is achieved by adjusting the parameters of the machine model using the stochastic algorithms until there is a close correlation between the measured and simulated data. The new set of the model parameters indicates if the machine winding is healthy or if there is a fault and also the location and the nature of this fault. The induction motor is tested to see whether it is in a healthy working condition and whether there any fault conditions. For a healthy machine, the overall error will be very small and the stochastic algorithm should identify the nominal parameters for the induction motor. It worth mentioning, that the skin effect was taken into consideration by multiplying the measured DC stator resistance of the machine by a factor of 1.1. The primary assumption made in the ABC/abc model is that of negligible magnetic saturation in the machine.

In the case of a faulty machine, there will be a large error which indicates that a winding fault of some type is present and also its location; for example, an increased value in the resistance of a rotor winding indicates a developing open-circuit fault in this circuit, and so on. This stochastic approach does not require any expert prior knowledge of the type of fault or its location.

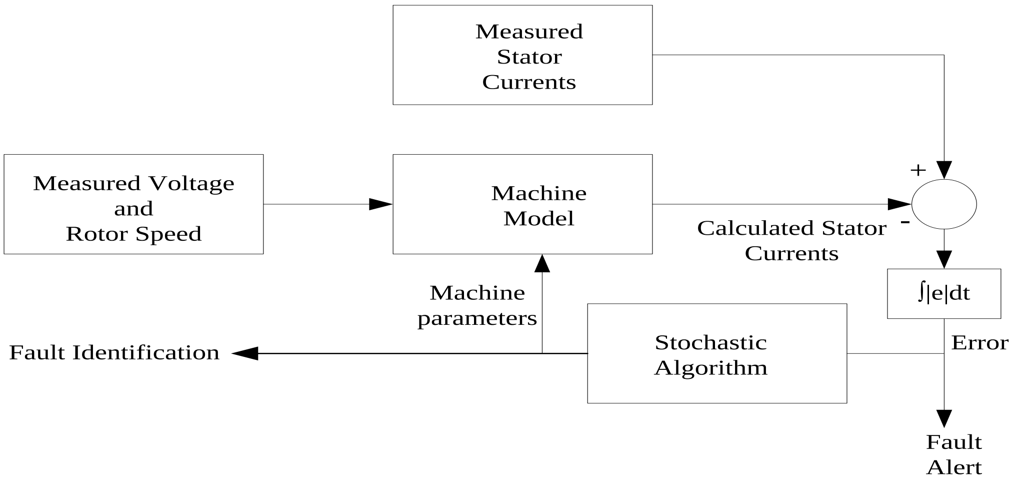


Figure 4.1. Schematic representation of the fault identification technique.

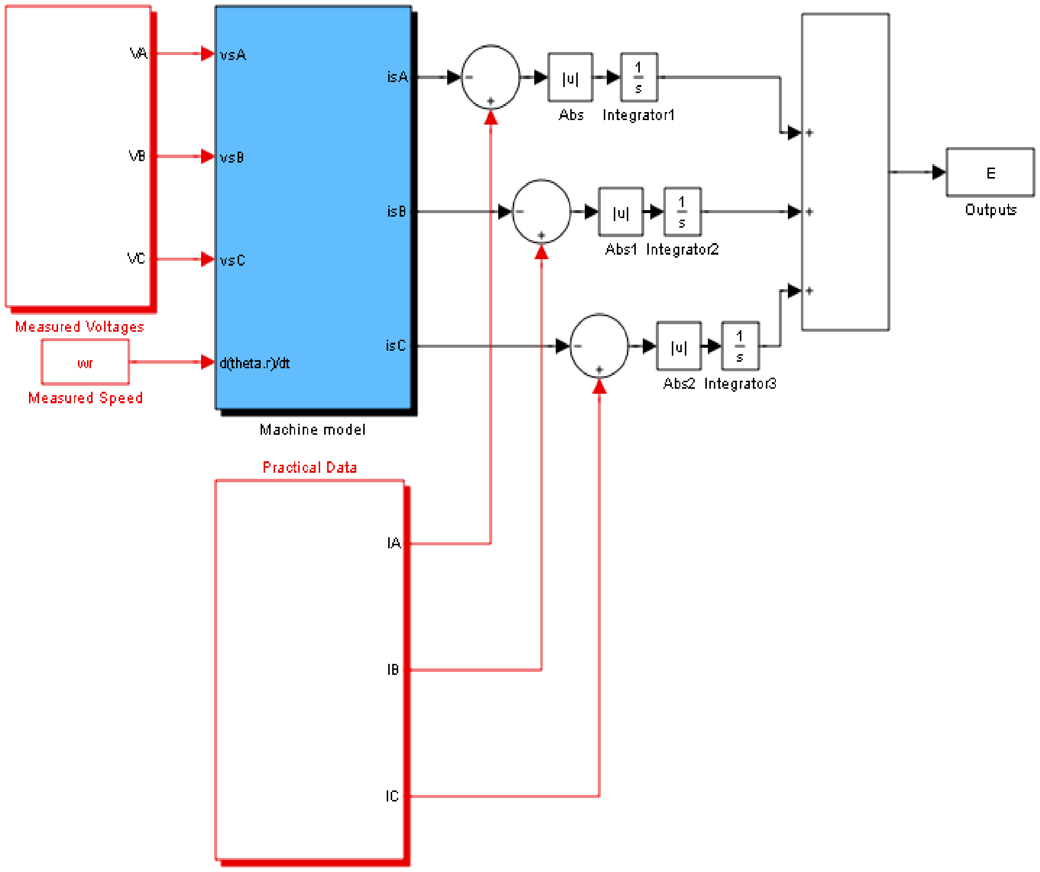


Figure 4.2. Simulink model showing machine mathematical model combined with practical data.

The mathematical ABCabc model of an induction motor is developed using Simulink software and is used with the stochastic algorithms for identifying machine winding faults. This ABCabc model is obtained from the standard machine ABCabc voltage equations and represented by Equation (4.1) [84].

$$\begin{bmatrix} V_{sA} \\ V_{sB} \\ V_{sC} \\ V_{ra} \\ V_{rb} \\ V_{rc} \end{bmatrix} = \begin{bmatrix} R_{sA} + pL_{ss} & pM_{ss} & pM_{ss} & pM_{sr}\cos\theta_r & pM_{sr}\cos\theta_{r1} & pM_{sr}\cos\theta_{r2} \\ pM_{ss} & R_{sB} + pL_{ss} & pM_{ss} & pM_{sr}\cos\theta_{r2} & pM_{sr}\cos\theta_r & pM_{sr}\cos\theta_{r1} \\ pM_{ss} & pM_{ss} & R_{sC} + pL_{ss} & pM_{sr}\cos\theta_{r1} & pM_{sr}\cos\theta_{r2} & pM_{sr}\cos\theta_r \\ pM_{sr}\cos\theta_r & pM_{sr}\cos\theta_{r2} & pM_{sr}\cos\theta_{r1} & R_{ra} + pL_{rr} & pM_{rr} & pM_{rr} \\ pM_{sr}\cos\theta_{r1} & pM_{sr}\cos\theta_r & pM_{sr}\cos\theta_{r2} & pM_{rr} & R_{rb} + pL_{rr} & pM_{rr} \\ pM_{sr}\cos\theta_{r2} & pM_{sr}\cos\theta_{r1} & pM_{sr}\cos\theta_r & pM_{rr} & pM_{rr} & R_{rc} + pL_{rr} \end{bmatrix} \cdot \begin{bmatrix} I_{sA} \\ I_{sB} \\ I_{sC} \\ I_{ra} \\ I_{rb} \\ I_{rc} \end{bmatrix} \quad (4.1)$$

where (V_{sA}, V_{sB}, V_{sC}) ,(I_{sA}, I_{sB}, I_{sC}) are stator winding voltages and currents, (V_{ra}, V_{rb}, V_{rc}), (I_{ra}, I_{rb}, I_{rc}) are rotor winding voltages and currents, (R_{sA}, R_{sB}, R_{sC}), (R_{ra}, R_{rb}, R_{rc}) are stator and rotor winding resistances, L_{ss}/L_{rr} are stator and rotor winding self inductance, M_{ss}/M_{rr} are the mutual inductance between pairs of stator and rotor windings, M_{sr} is the peak value of the rotor position dependent mutual inductance between stator and rotor winding pairs, θ_r is the rotor angle, $\theta_{r1} = \theta_r + 2\pi/3$, $\theta_{r2} = \theta_r + 4\pi/3$ and $P = d/dt$ is the differential operator.

4.3 Test rig

The experimental test rig used in this investigation consisted of a three-phase 240 V, 1.5 kW wound rotor induction machine coupled to a 3 kW DC machine used as a generator to provide the necessary load torque. Stator and rotor windings of the machine are brought out to external terminals as shown in Figure 4.3. The stator of the experimental machine consists of a single-layer 2-pole AC winding arranged in 24 slots. The terminal points of all the 12 coils are brought out to the 24 terminals. The complete arrangement is in the form of two concentric circles. The rotor of the experimental machine comprises a 2-pole full-pitch lap winding which is arranged in 36 slots and brought out to six slip rings each of which has 2 brushes [85]. The experimental rig is shown in Figure 4.4. For loading purpose, a resistor load bank is connected across the armature of the DC machine as shown in Figure 4.4. As shown in Figure 4.5, the induction motor had a star connected stator windings and a short circuited delta connected rotor winding [F_1, E_2, and D_3]. A photograph of the front plate showing winding terminations is given in Figure 4.6.

Stator voltage and current data were collected using two four channel Tektronix (MSO/DPO4000) digital storage oscilloscopes with isolated current probe amplifier (Tektronix A622), and high voltage differential probes (P5200). The data was collected and saved in (csv) file format for further analysis. Rotor speed was measured using a digital tachometer RS-445-9557.

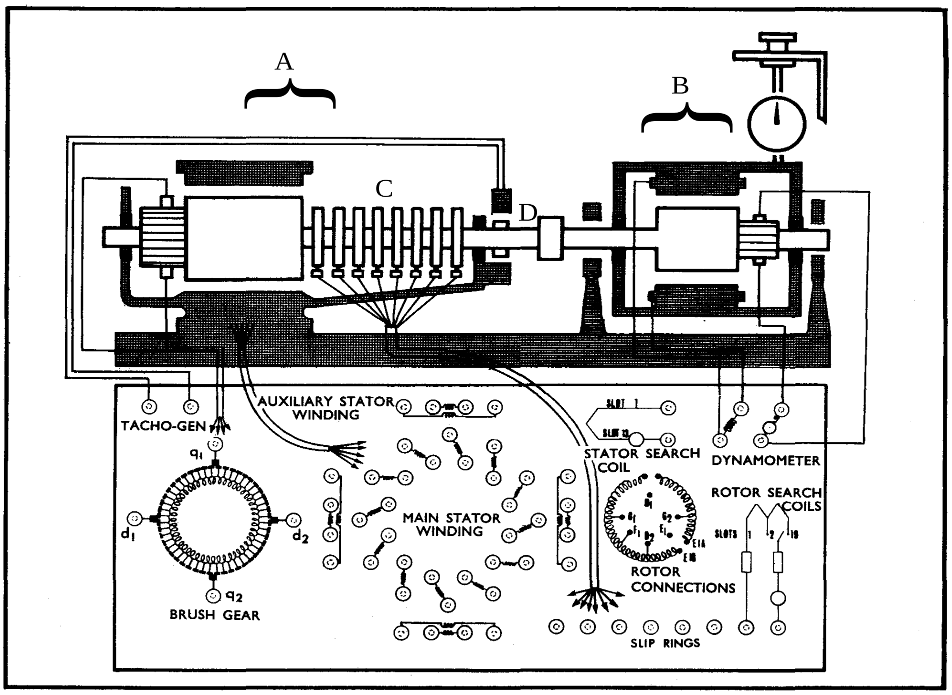


Figure 4.3. Schematic diagram of the experimental set-up [85].

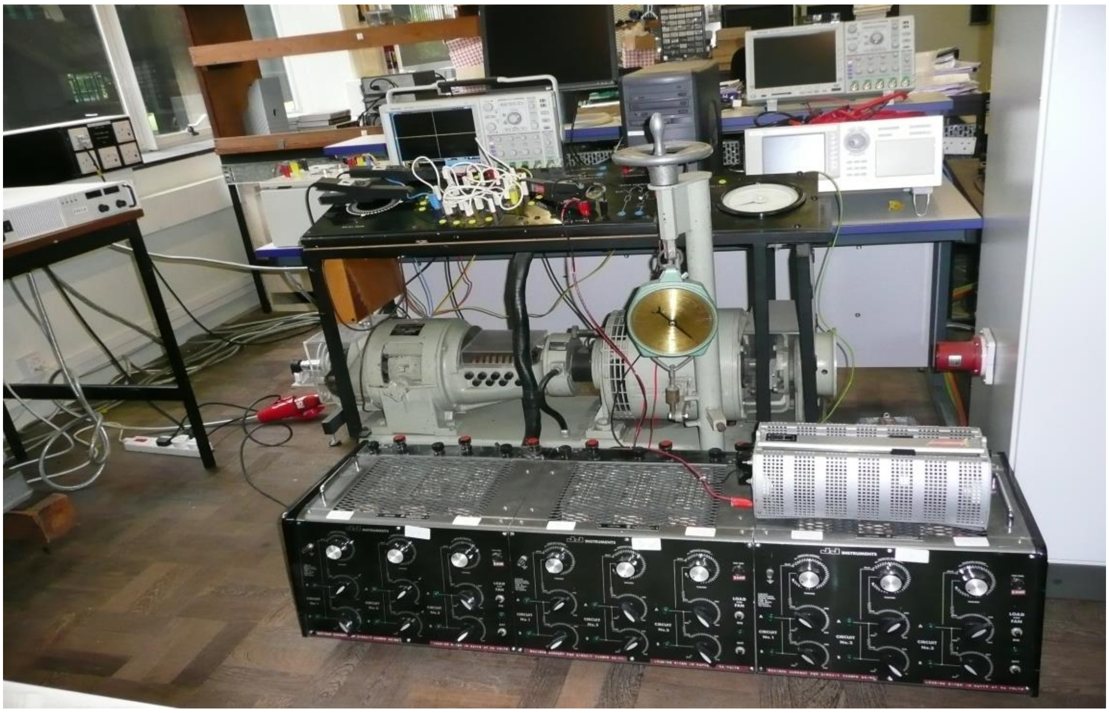


Figure 4.4. The experimental test rig.

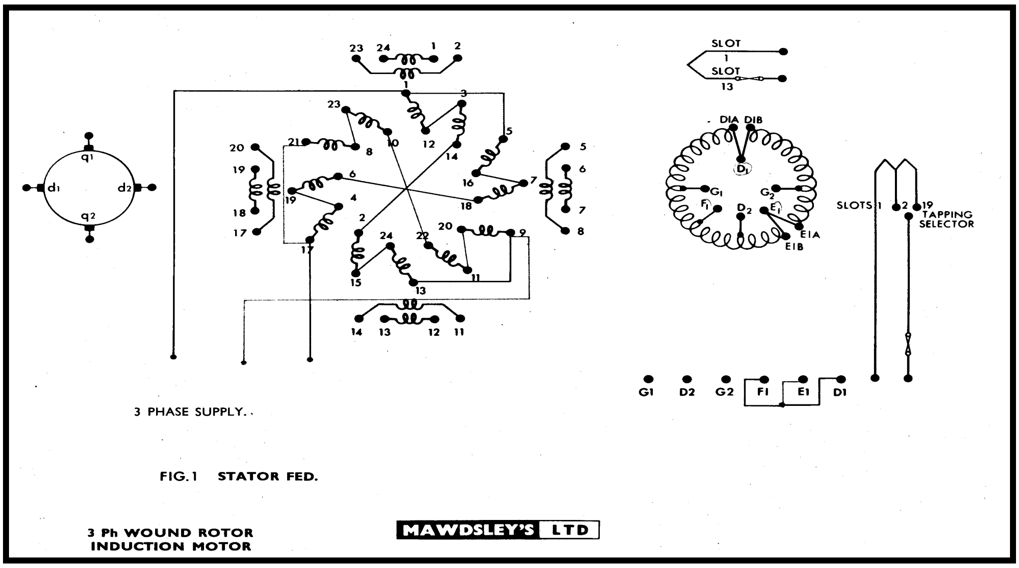


Figure 4.5. Induction machine connection diagram [85].

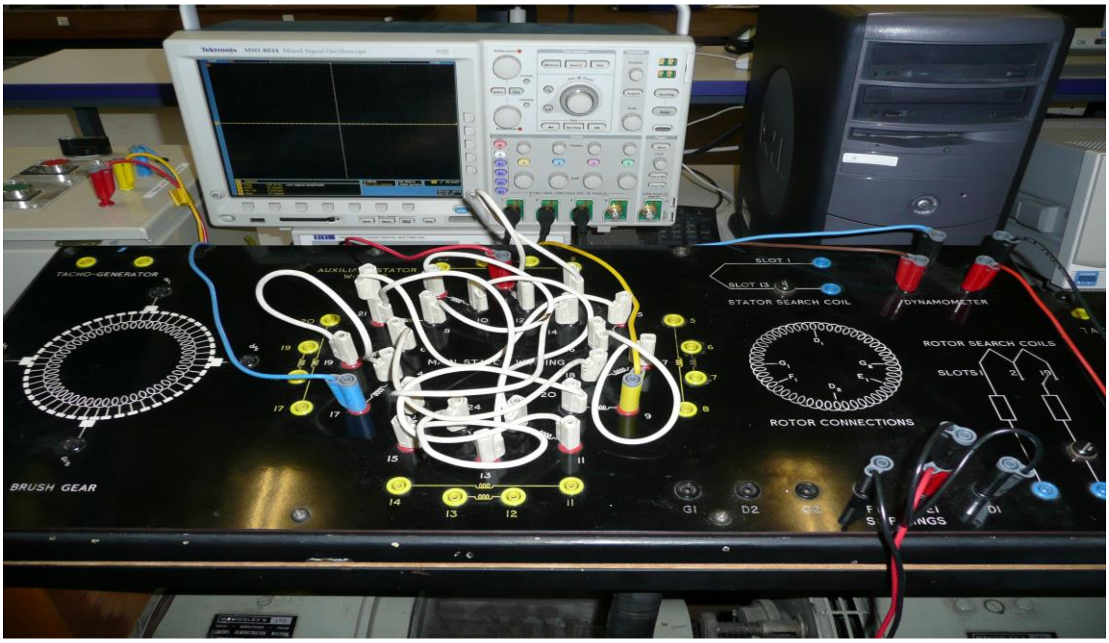


Figure 4.6. View of test machine front plate.

The nominal values of the induction machine equivalent circuit parameters were calculated using the IEEE standard [86] and then used for fault identification. Table 4.1 states the machine parameters obtained from the results of a no-load test, a locked-rotor test, and measurements of the DC resistances of the stator windings. The experiment work is conducted in a three-phase, 50 Hz, 240 V and the data is collected over a time window of 0.2 sec, and a sampling interval of 1 ms, as the machine is operating during steady-state. The calculated stator current waveforms obtained by using the IEEE standard does not agree closely with the measured one as can be noticed in Figure 4.7.

Table 4.1. Induction motor model parameters.

Stator resistances	R_s = 4. 417 (Ω)
Rotor resistances	R_r = 5.173 (Ω)
Stator self-inductances	L_{ss} = 0.691 (H)
Rotor self-inductances	L_{rr} = 0.691 (H)
Mutual inductances between the stator windings	M_{ss} = 0.334 (H)
Mutual inductances between the rotor windings	M_{rr} = 0.334 (H)
Mutual inductance between stator and rotor winding pairs	M_{sr} = 0.668 (H) $M_{rs=}\,M_{sr}$

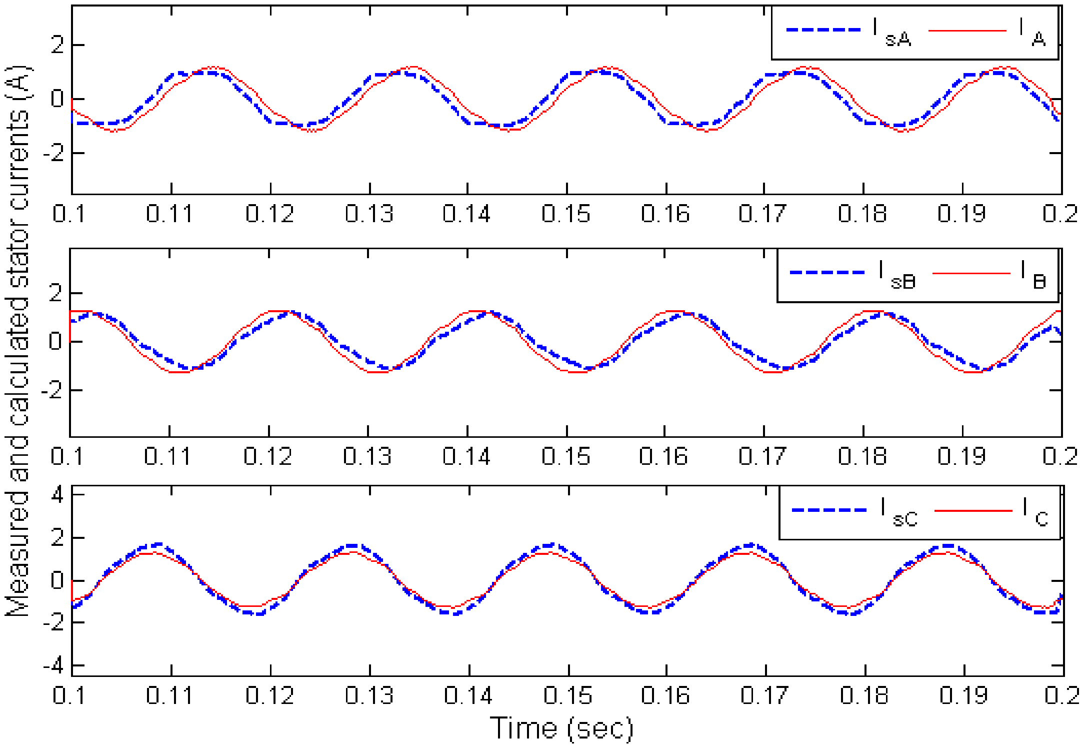


Figure 4.7. Measured (I_A, I_B, I_C) and calculated (I_{sA}, I_{sB}, I_{sC}) stator current waveforms using the estimated parameters obtained from IEEE standard tests.

4.4 Experimental results

The measurements that need to be taken are input voltages, stator currents and the rotor speed of the induction machine; this is done by using voltage differential probes (P5200), current probe amplifier (Tektronix Ac22) and a digital tachometer. Data is collected over a time window of 0.2 sec. The sampling interval is 1 ms; data is then saved in csv files format for different winding fault conditions using a digital oscilloscope. The oscilloscope was set to be triggered from the voltage channel for both current and voltage measurements.

4.4.1 Healthy Machine Tests

These tests were carried out to ensure that the stochastic algorithm is able to find the parameters of the induction motor when it is in healthy condition. Figure 4.8 shows a diagram of the healthy stator and rotor windings.

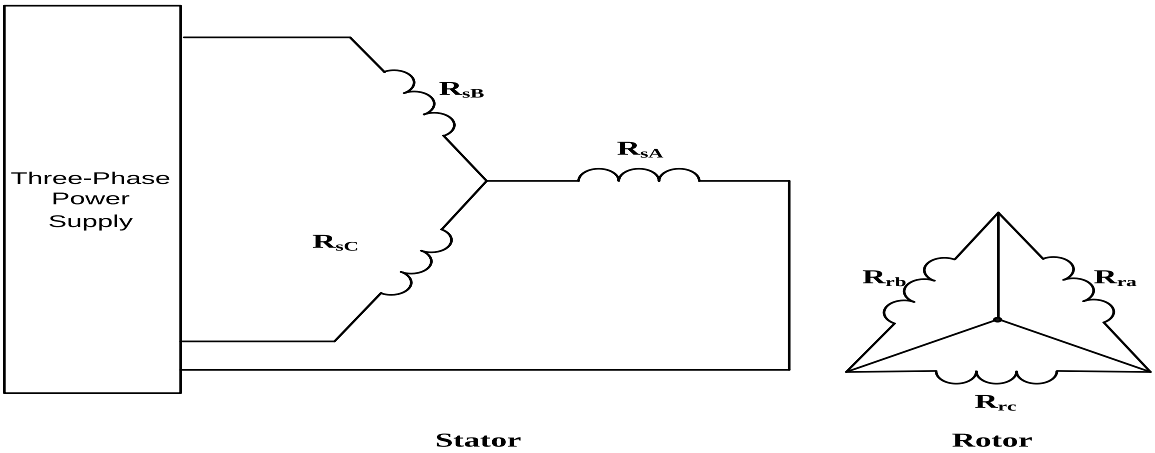


Figure 4.8. Healthy conditions.

The machine was operating at steady state with load of about 50% of full load at a speed ω_r of 2878 rpm. Figure 4.9 and Figure 4.10 show the measured voltage and current waveforms obtained from the induction motor at steady-state when it is fed from the three phase mains supply.

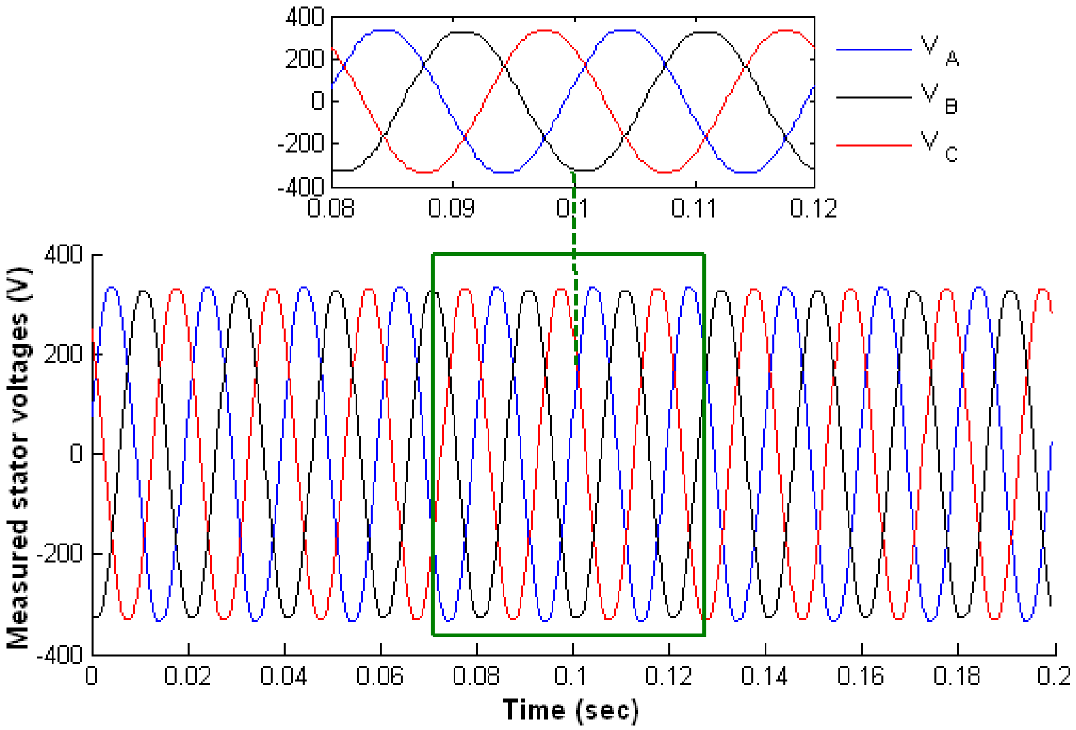


Figure 4.9. Measured stator voltage waveforms; supply-fed machine, healthy conditions.

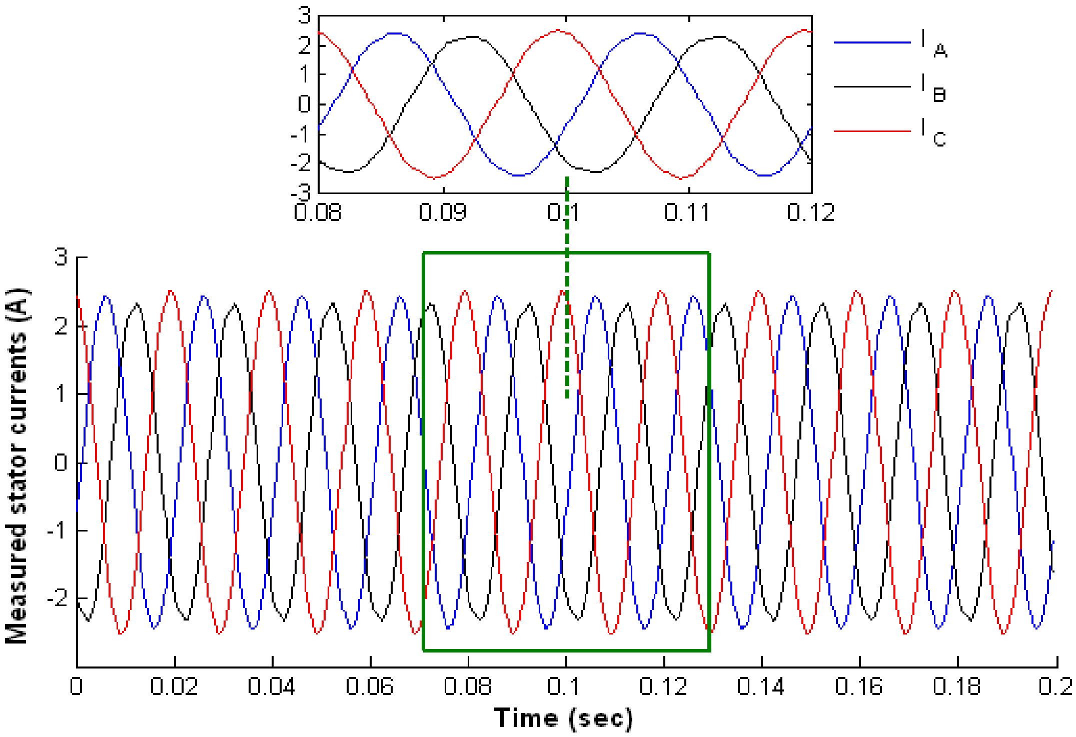


Figure 4.10. Measured stator current waveforms; supply-fed machine, healthy conditions.

4.4.2 Stator open-circuit winding fault

As shown in Figure 4.11, an external resistance (7 Ω) was placed in series with a phase winding B to emulate an open-circuit stator winding fault. The selected 7 Ω resistance value represents the smallest value to produce a noticeable open-circuit fault in the machine. Figure 4.12 and Figure 4.13 show measured supply voltage waveforms and stator currents waveforms.

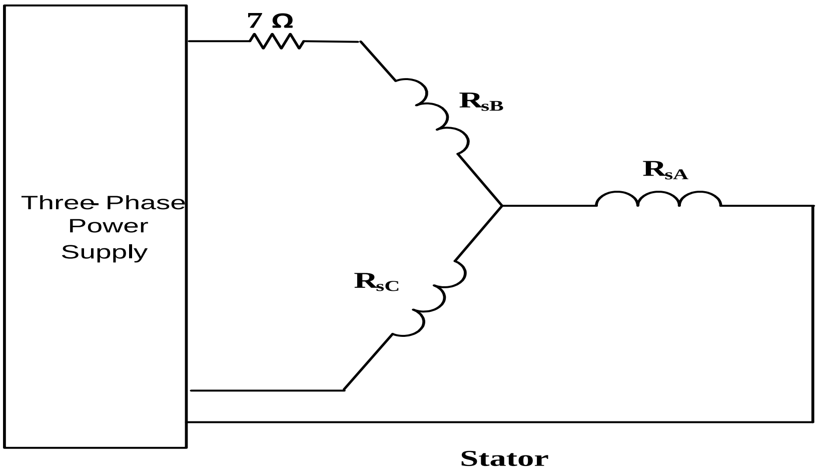


Figure 4.11. Developing stator winding open-circuit fault; test circuit.

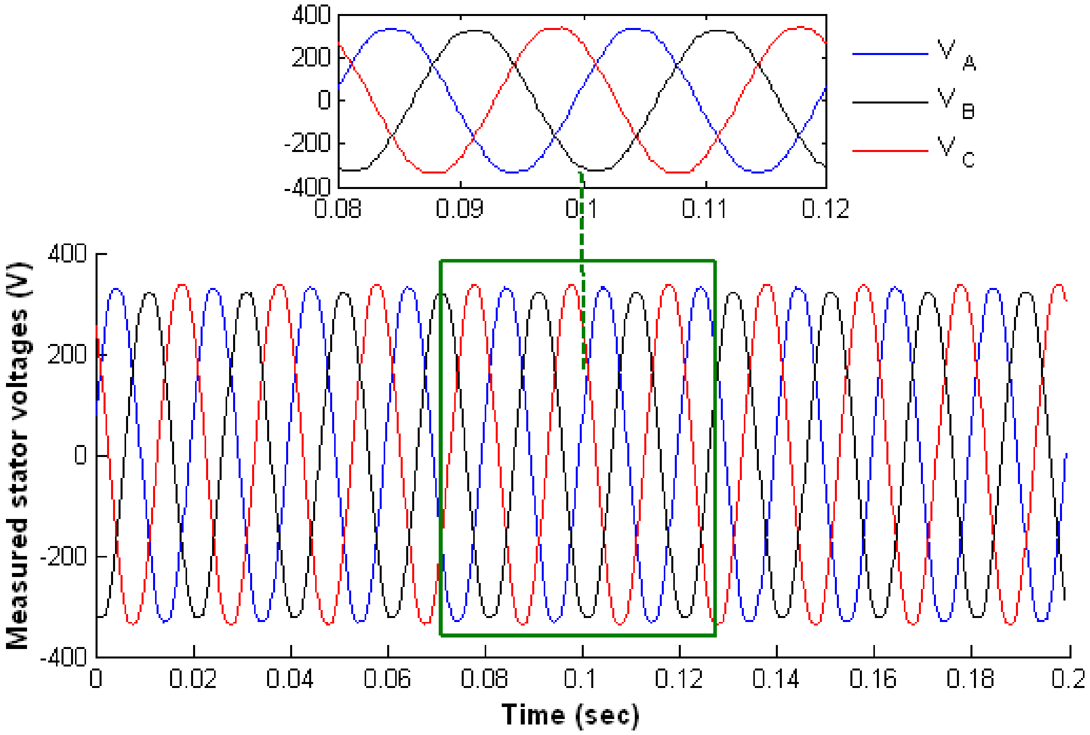


Figure 4.12. Measured stator voltage waveforms; developing stator open-circuit winding fault.

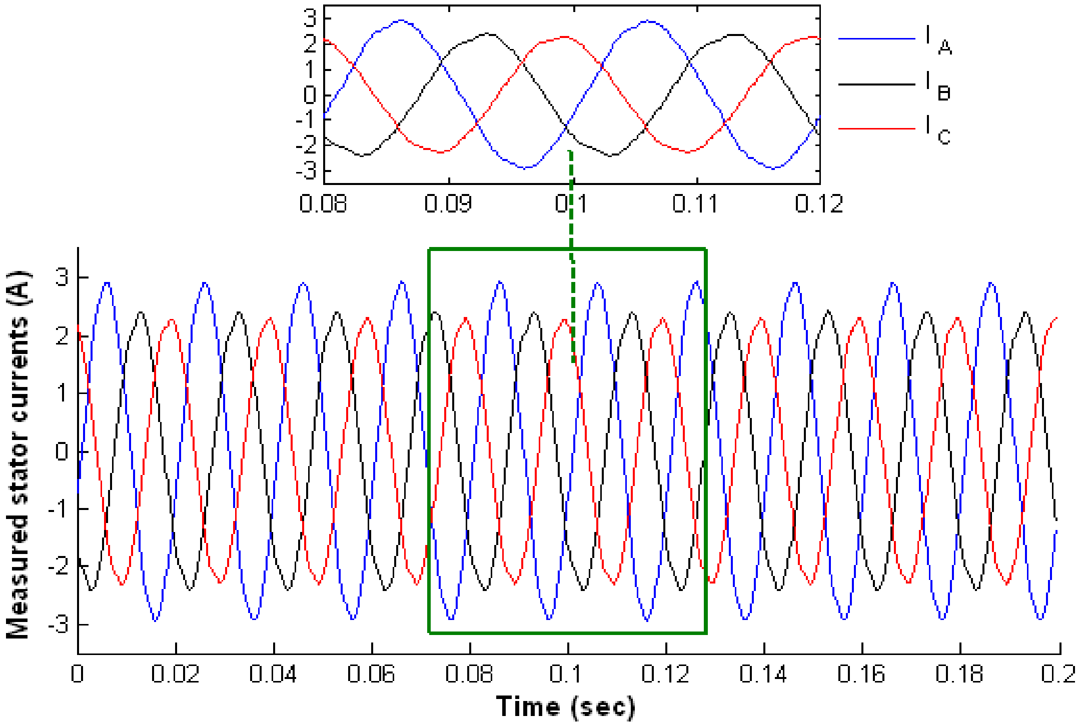


Figure 4.13. Measured stator current waveforms; developing stator open-circuit winding fault.

4.4.3 Rotor open-circuit winding fault

As shown in Figure 4.14, an external resistance (7 **Ω**) was placed in series with the line connected to the two ends of the a-b rotor delta windings, to emulate an open-circuit rotor winding fault.

Figure 4.15 and Figure 4.16 show the measured voltage waveforms and the measured current waveforms obtained from the faulty motor fed from three phase supply.

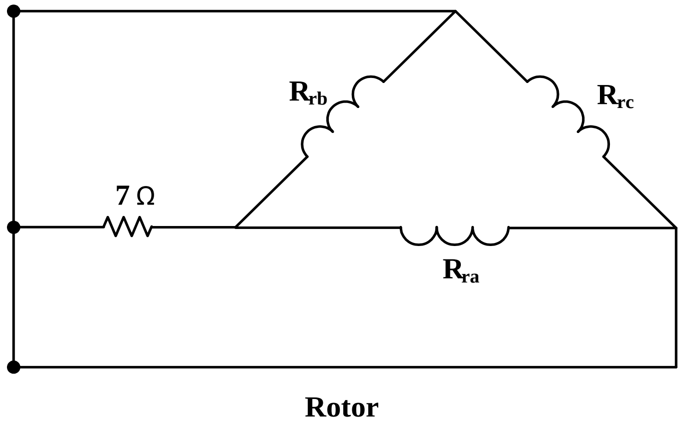


Figure 4.14. Developing rotor winding open-circuit fault; test circuit.

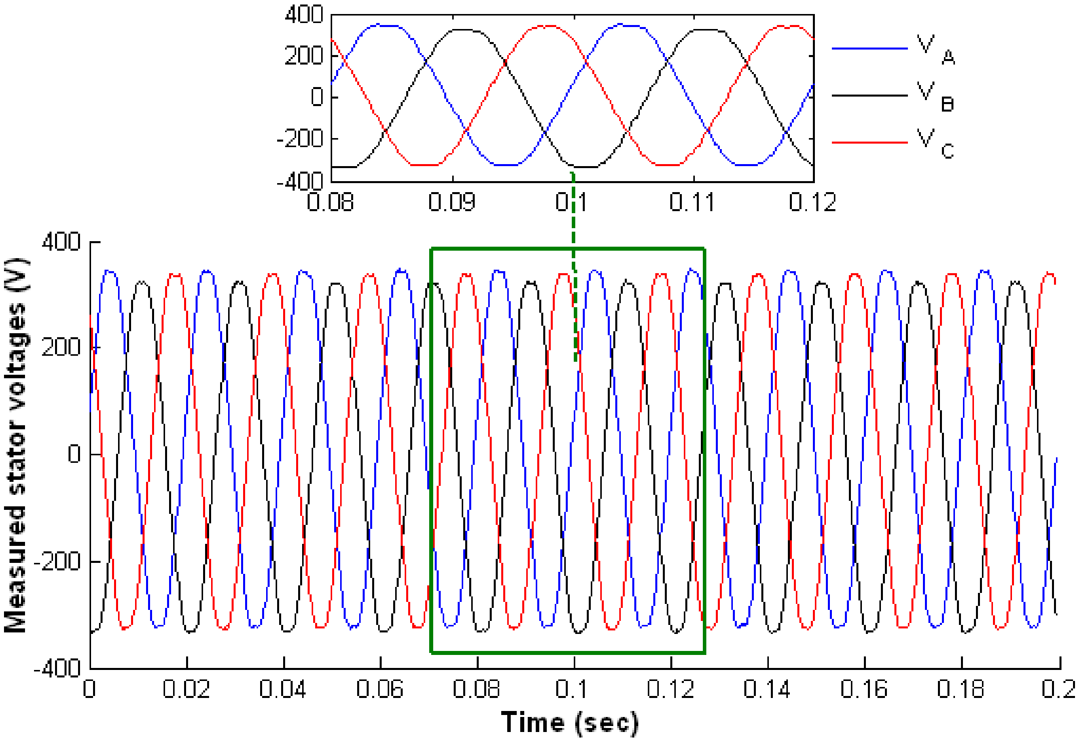


Figure 4.15. Measured stator voltage waveforms; developing rotor open-circuit winding fault.

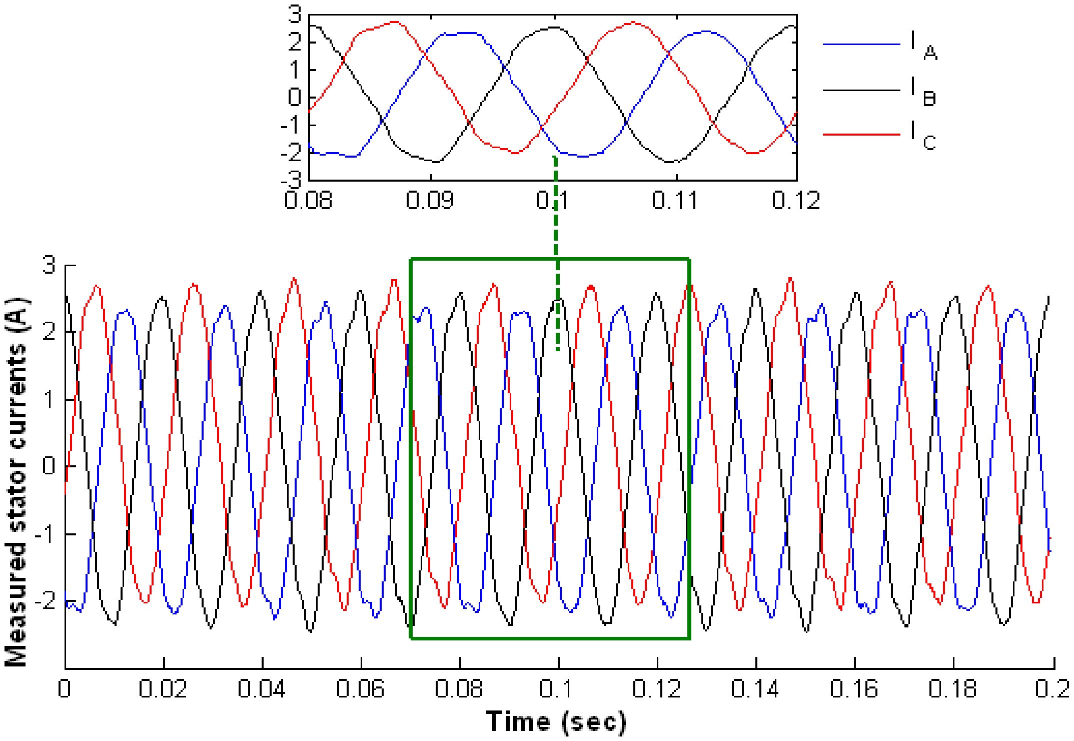


Figure 4.16. Measured stator current waveforms; developing rotor open-circuit winding.

4.5 Summary

This Chapter has described the condition monitoring scheme and the experimental test rig used in this investigation. The laboratory machine set employed in this study was used to emulate different stator and rotor winding faults. Stator voltage and current waveforms were collected to be used by stochastic optimisation algorithms (GA, TS and SA) for induction machine faults detection and parameter identification, the verification of these algorithms and a the results of the parameter estimation process are presented in the next three Chapters.

CHAPTER 5

USE OF GENETIC ALGORITHM FOR INDUCTION MOTOR FAULT DETECTION AND PARAMETER IDENTIFICATION

5.1 Introduction

The basic concept of the use of genetic algorithm (GA) as an optimisation tool has been explained in chapter 3. This chapter deals with the application of GA in fault detection and parameter identification. This stochastic method is implemented using the induction motor ABCabc model for estimating values of induction machine parameters and identifying machine winding faults. The Matlab/Simulink software environment is used to implement this stochastic algorithm. The performance of the identification scheme is demonstrated with measured steady-state induction machine data. The measured waveforms are the three terminal voltages, three stator currents and rotor speed, collected over a time window of 0.2 sec using a sampling interval of 1 ms.

Before the GA could be used with the experimental test data, two important aspects should be considered: the coding of chromosomes and defining the evolution criteria. The chromosomes can be encoded as either binary or real values. In this study, the parameters are encoded with real values to alleviate errors in decimal-to-binary and binary-to-decimal conversions.

The evolution criteria deal with evaluating each chromosome’s fitness using an appropriate measure. In this study, performance is assessed by using the integral of absolute error (IAE) where the best parameters are associated with the smallest IAE. The parameters of the GA algorithm are shown in Table 5.1.

Table 5.1. GA parameter values.

Description	Value
Population size	12
Crossover rate	0.7
Mutation rate	0.05
Generation gap	0.9
Precision of variables	20
Number of generations	100

5.2 Induction machine parameter identification using GA

The measured waveforms obtained from the healthy machine tests (Figure 4.9 and Figure 4.10) are used here to show that this technique can be used for induction motor parameter identification. The parameters obtained using this technique are compared with the parameters obtained using IEEE standard tests.

The objective of the identification is to determine a vector of seven parameters, which represent the resistances and inductances of the motor,

$$\text{Chromosome} = (R_s, R_\mathrm{r}, L_{ss}, L_{rr}, M_{ss}, M_{rr}, M_{sr})$$

The definition of the problem is formulated so as to find the parameters with a given set of measurements:

- Three-phase stator voltages and currents

- Rotor speed of the motor

Each chromosome represents a candidate solution for the parameters that can be applied to the induction motor model to calculate the cost function. The search space is empirically set to maximise the effectiveness algorithm (see Table 5.2). After the encoding and evolution criteria are chosen, the step to crossover and mutation can be carried out as outlined in Chapter 3. The GA code is provided in Appendix A2.

Table 5.2. Search space for machine parameters.

R_s *(Ω)*	R_r *(Ω)*	L_{ss} *(H)*	L_{rr} *(H)*	M_{ss} *(H)*	M_{rr} *(H)*	M_{sr}*(H)*
1-20	1-20	0.01-1.2	0.01-1.2	0.01-1.2	0.01-1.2	0.01-1.2

Figure 5.1 and Figure 5.2 illustrate the evolution of the estimated stator and rotor winding resistances, self and mutual inductances parameters obtained using GA while Table 5.3 gives the final estimated values of these parameters.

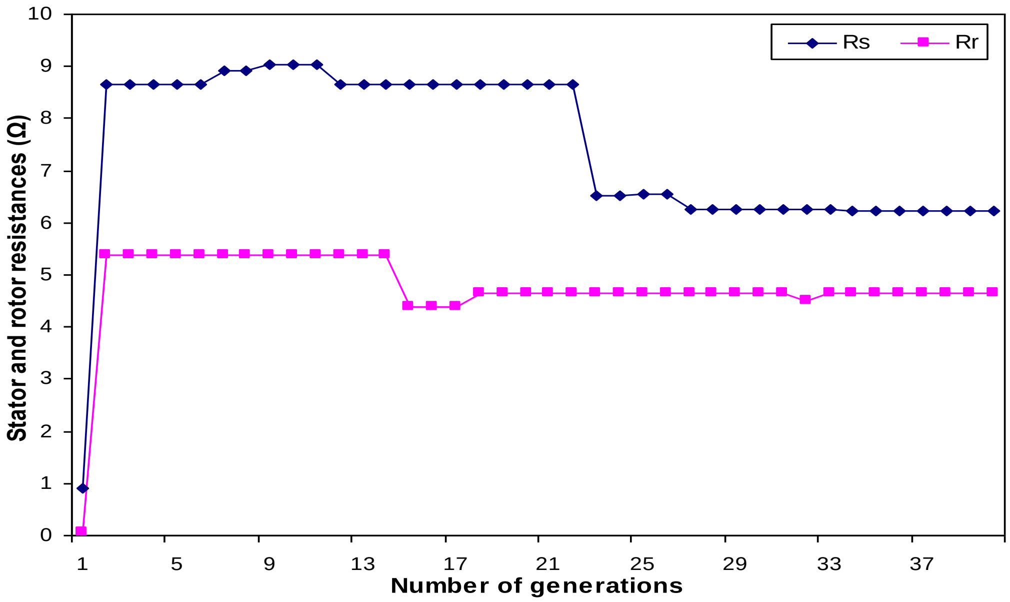


Figure 5.1. Estimated stator and rotor resistances obtained using GA.

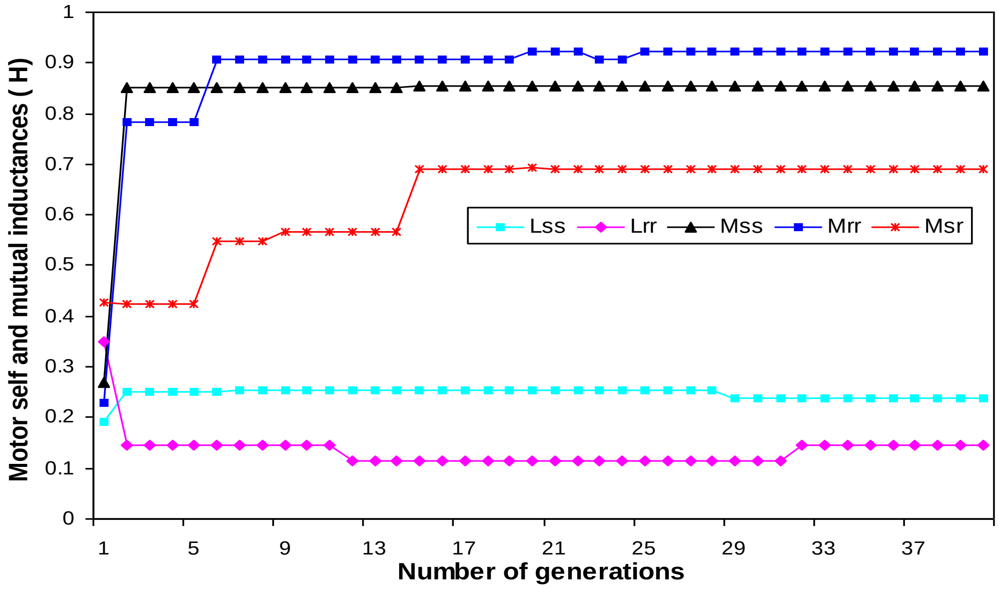


Figure 5.2. Estimated stator and rotor self and mutual inductances obtained using GA.

Table 5.3. Final values of machine parameters obtained using GA.

R_s (Ω)	R_r (Ω)	L_{ss} (H)	L_{rr} (H)	M_{ss} (H)	M_{rr} (H)	M_{sr} (H)
6.241	4.638	0.238	0.146	0.853	0.924	0.689

Around 30 investigations of potential solutions are required to obtain convergence of machine parameters. The error corresponding to the existing best solution is shown in Figure 5.3. The calculation error falls from maximum value of 0.164 A.s to 0.021 A.s. It can be seen from Figure 5.4 that the measured (I_A, I_B, I_C) and calculated (I_{sA}, I_{sB}, I_{sC}) stator currents using the final parameter values obtained by the GA algorithm are in good agreement. The parameter values obtained through these methods are compared with those obtained from IEEE standard tests. Based on the results, the conclusion can be drawn that the GA algorithm is able to provide reliable results.

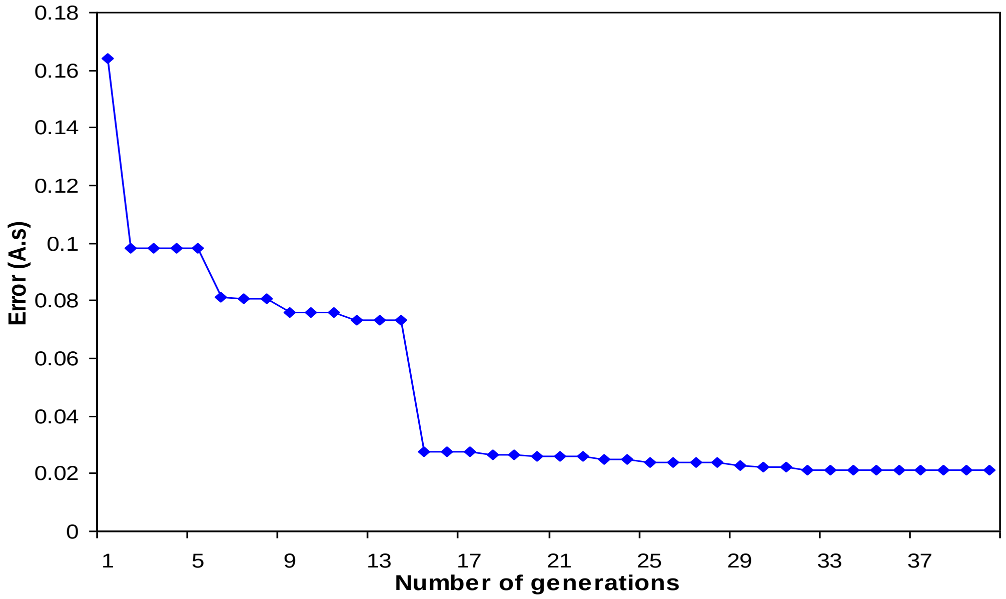


Figure 5.3. Estimated calculation error for parameter estimation obtained using GA.

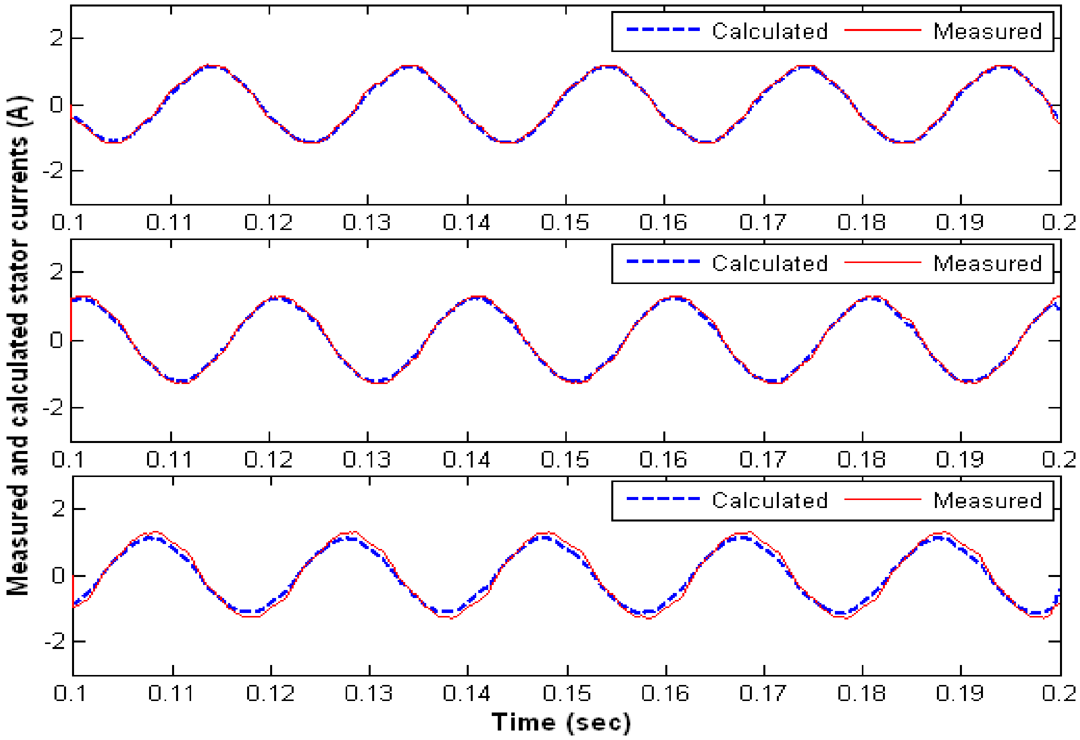


Figure 5.4. Measured (I_A, I_B, I_C) and calculated (I_{sA}, I_{sB}, I_{sC}) stator current waveforms using the estimated parameters obtained from GA.

5.3 Winding fault detection

When trying to identify open-circuit winding faults, the number of variables needs to be reduced to six parameters, namely the three stator resistances (R_{sA}, R_{sB}, R_{sC}) and three rotor resistances (R_{ra}, R_{rb}, R_{rc}), in order to allow the algorithm to converge within the designated stop criteria. The other machine parameters are maintained at the values identified earlier based on the GA identification process (Table 5.3). The computational complexity of the analysis is related to the number of variables, the size of the search space and the operating characteristics of the algorithm itself. By repeating an experiment a number of times it is possible to determine the average convergence time and the success rate of the algorithm.

5.3.1 Supply-fed induction motor

In these tests, the induction motor is supplied directly from the 50 Hz three-phase supply. The 2-pole induction motor has a nominal phase voltage of 240 V. Tests are carried out emulating stator and rotor open-circuit winding fault conditions.

5.3.1.1 Stator open-circuit winding fault

This experiment is conducted by replicating a stator open-circuit fault. A developing stator open-circuit winding fault is emulated by connecting a 7-Ω resistor in series with a stator phase winding (winding B), as shown in Figure 5.5. Then, the GA algorithm is implemented in conjunction with the measured waveforms to estimate the machine winding parameters as a means of detecting this fault, as previously described.

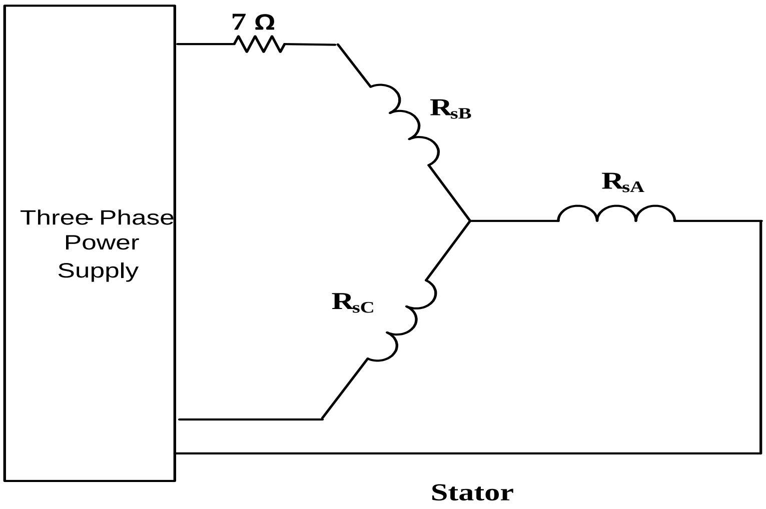


Figure 5.5. Developing stator winding open-circuit fault; test circuit.

The six winding resistances (R_{sA}, R_{sB}, R_{sC}, R_{ra}, R_{rb}, R_{rc}) are the parameters to be optimised. The type and location of the fault are identified by adjusting the ABCabc model parameters.

As shown in Figure 5.6, there is a much higher estimated value of winding resistance (13.937 Ω) in stator phase B while the other estimated stator resistances are at approximately their nominal values, indicating the presence of a stator open circuit winding fault. At the same time, the estimated three rotor resistances are all at their nominal values (Figure 5.7), indicating a healthy state for the rotor winding. Figure 5.8 shows the corresponding values of the IAE function.

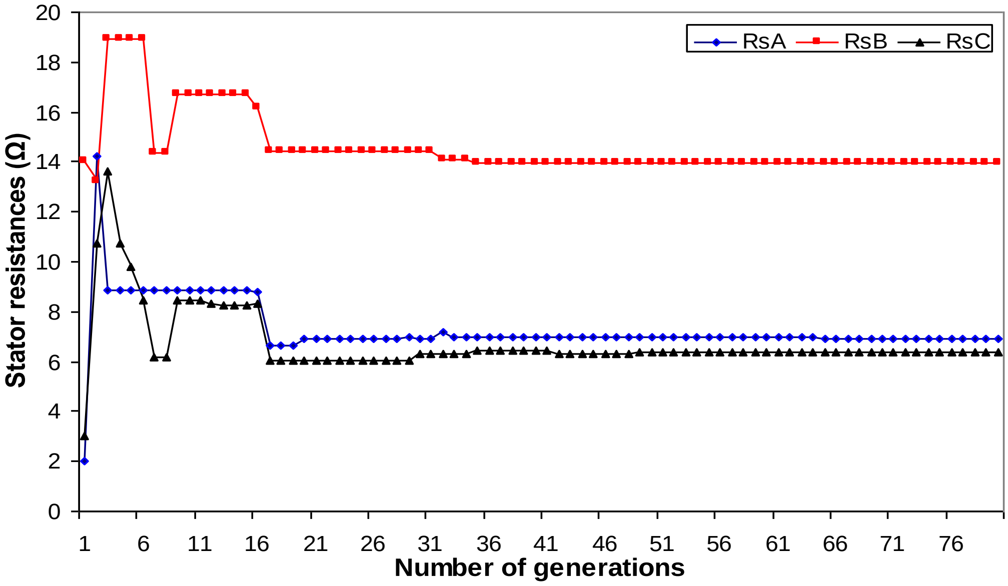


Figure 5.6. Estimated stator resistances obtained using GA for operation of induction motor with stator open-circuit fault.

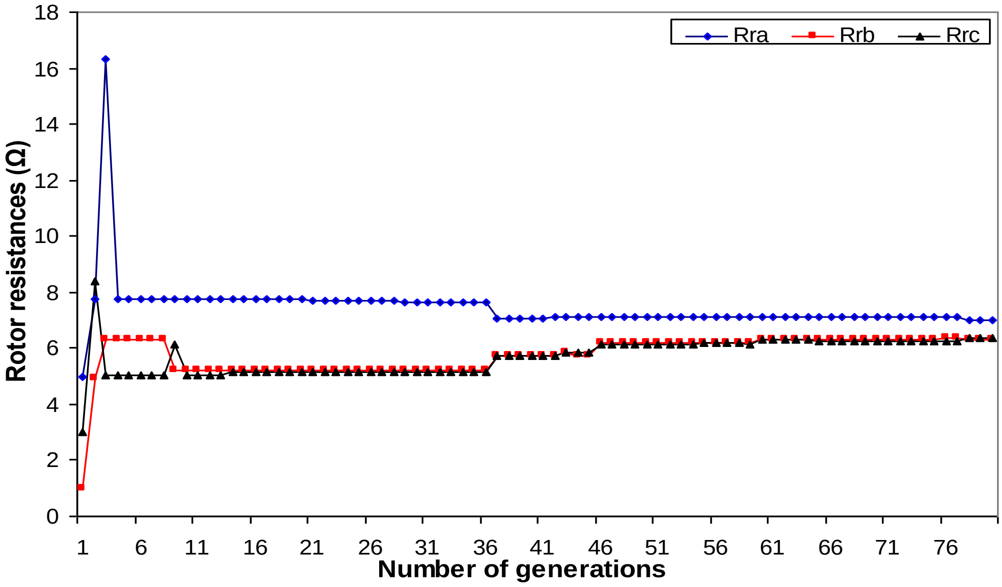


Figure 5.7. Estimated rotor resistances obtained using GA for operation of induction motor with stator open-circuit fault.

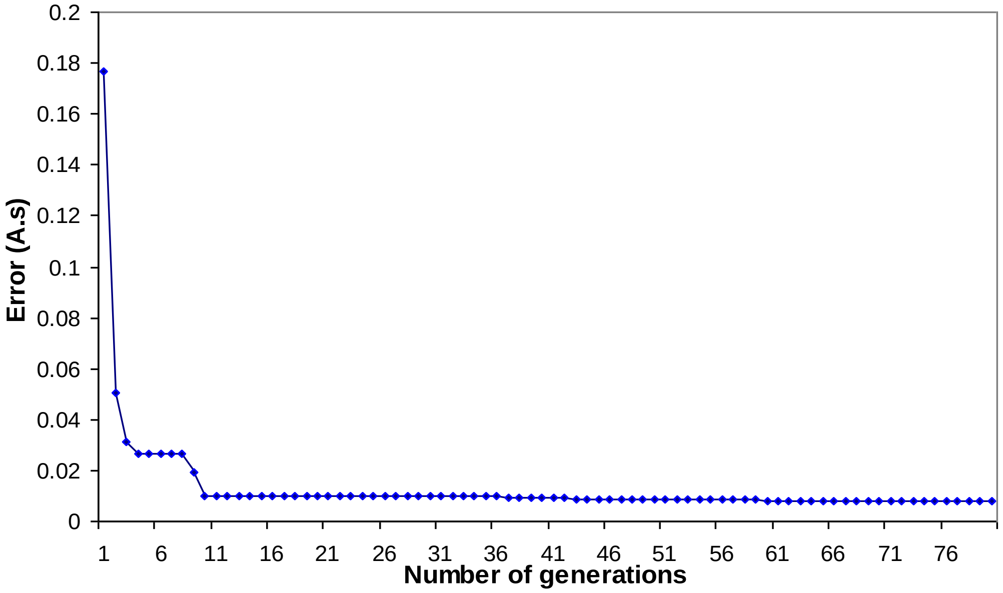


Figure 5.8. Estimated calculation error obtained using GA for operation of induction motor with stator open-circuit fault.

The number of investigations of potential solutions required to obtain convergence with this data set was 16. The final values of the stator and rotor resistances obtained at the end of the GA optimisation process are listed in Table 5.4.

Table 5.4. Final values of winding resistances obtained using GA with stator open-circuit fault.

R_{sA} (Ω)	R_{sB} (Ω)	R_{sC} (Ω)	R_{ra} (Ω)	R_{rb} (Ω)	R_{rc} (Ω)
6.937	13.964	6.369	6.991	6.311	6.387

Figure 5.9 shows the measured (I_A, I_B, I_C) and calculated (I_{sA}, I_{sB}, I_{sC}) stator currents using the final parameter values obtained by the GA algorithm revealing good agreements between the two current waveforms. This gives confidence in the ability of the GA algorithm not only to identify the presence of the open-circuit winding fault but also to accurately estimate the resulting parameter values.

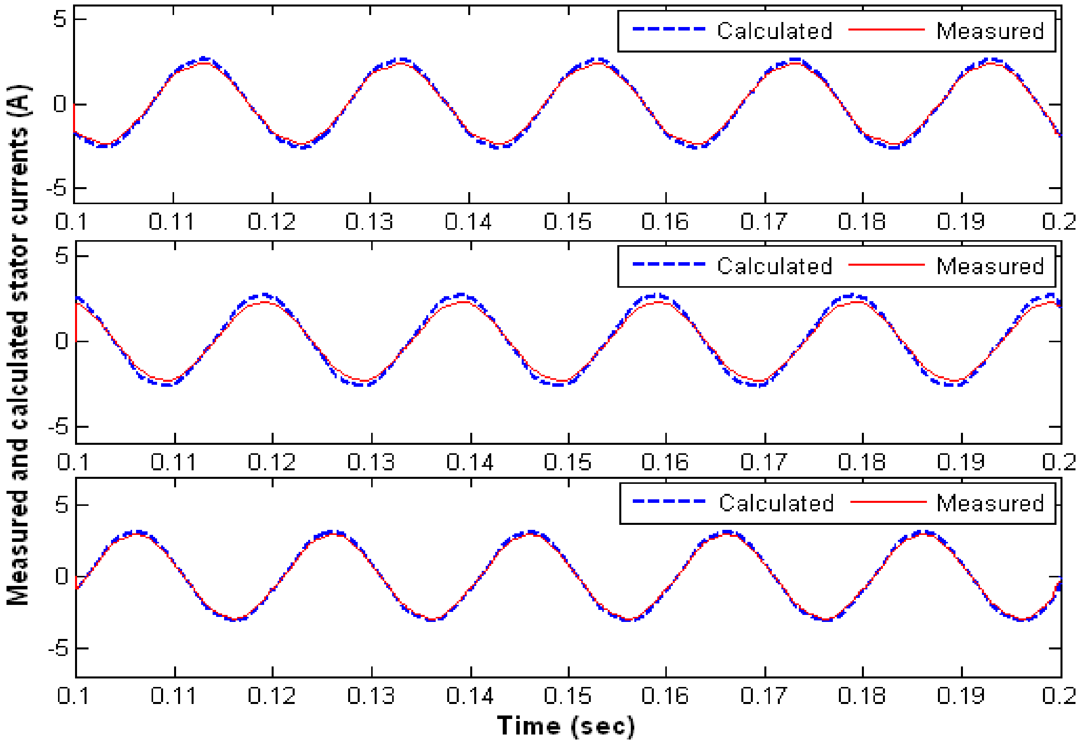


Figure 5.9. Measured (I_A, I_B, I_C) and calculated (I_{sA}, I_{sB}, I_{sC}) stator current waveforms using the estimated resistances obtained from GA for operation of induction motor with stator open-circuit fault.

5.3.1.2 Rotor open-circuit winding fault

A developing open-circuit rotor winding fault is emulated by connecting an external 7-Ω resistor in series with the line connected to the two ends of the $R_{rb} - R_{ra}$ rotor delta windings. This arrangement is shown in Figure 5.10.

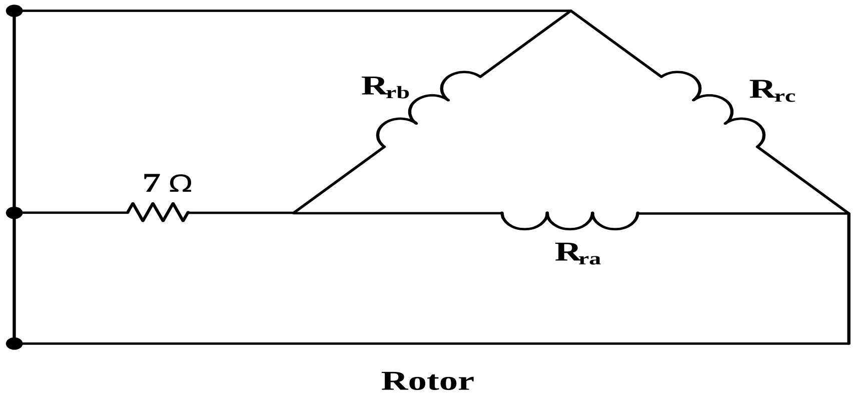


Figure 5.10. Developing rotor winding open-circuit fault; test circuit.

The GA algorithm is implemented to identify the presence of a developing rotor winding open-circuit fault based on the experimental measurements. In this test, the six winding resistances (R_{sA}, R_{sB}, R_{sC}, R_{ra}, R_{rb}, R_{rc}) are again the parameters to be optimised in order to minimise the IAE.

Figure 5.11 shows the estimated stator resistances during the optimisation process. It is clear from this figure that there is a rotor winding open-circuit fault as indicated by the high values of R_{ra} and R_{rb} compared with R_{rc}. On the other hand, the three stator resistances give nearly the same values; showing the healthy state of the stator (Figure 5.12).

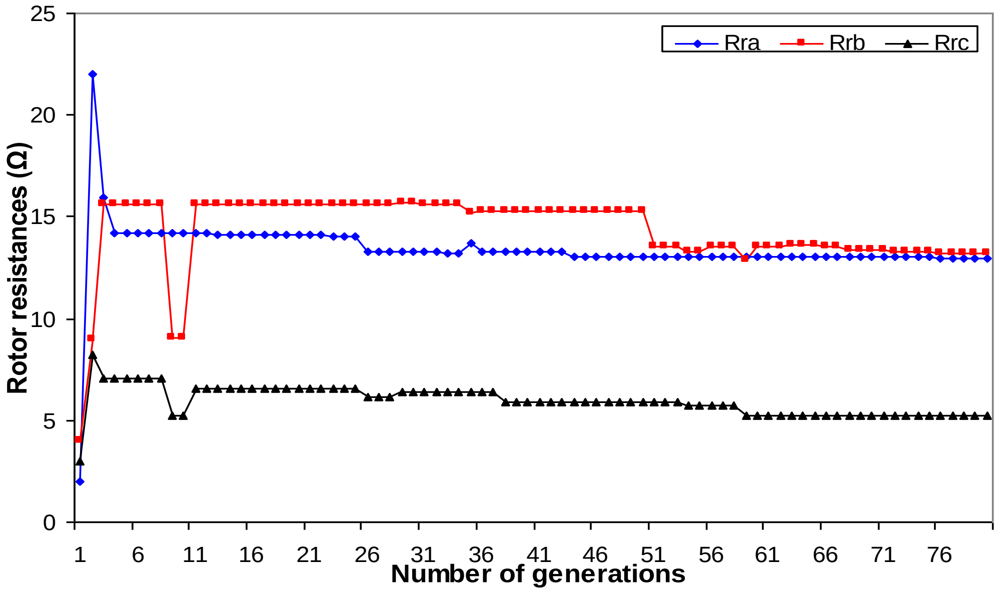


Figure 5.11. Estimated rotor resistances obtained using GA for operation of induction motor with rotor open-circuit fault.

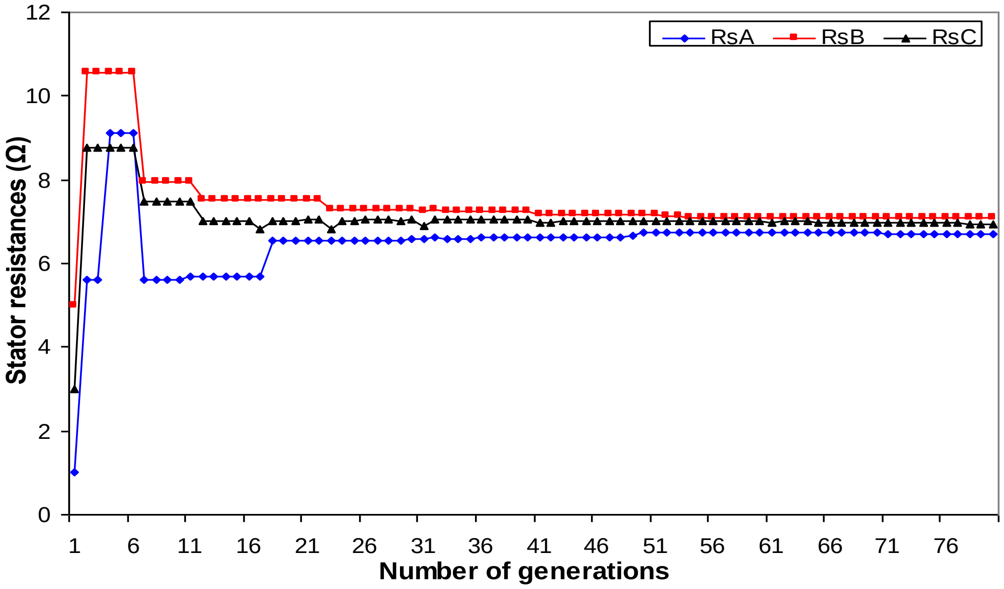


Figure 5.12. Estimated stator resistances obtained using GA for operation of induction motor with rotor open-circuit fault.

The number of investigations of potential solutions required to obtain convergence with this data set was 20. Figure 5.13 shows the error corresponding to the best solution under this rotor open-circuit fault condition. The error falls from a maximum value of 0.843 A.s to 0.449 A.s. The final estimated values of the stator and rotor resistances are listed in Table 5.5. The measured (I_A, I_B, I_C) and calculated (I_{sA}, I_{sB}, I_{sC}) stator currents waveforms when using the final parameter values obtained by the GA algorithm are shown in Figure 5.14.

Table 5.5. Final values of winding resistances obtained using GA with rotor open-circuit fault.

R_{sA} (Ω)	R_{sB} (Ω)	R_{sC} (Ω)	R_{ra} (Ω)	R_{rb} (Ω)	R_{rc} (Ω)
6.695	7.074	6.953	12.982	13.232	5.216

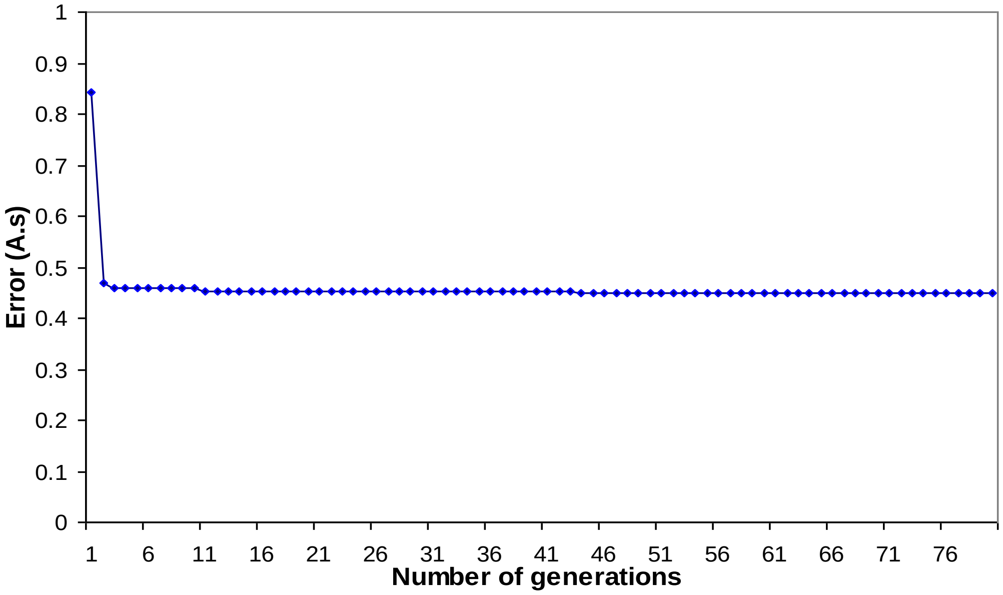


Figure 5.13. Estimated calculation error obtained using GA for operation of induction motor with rotor open-circuit fault.

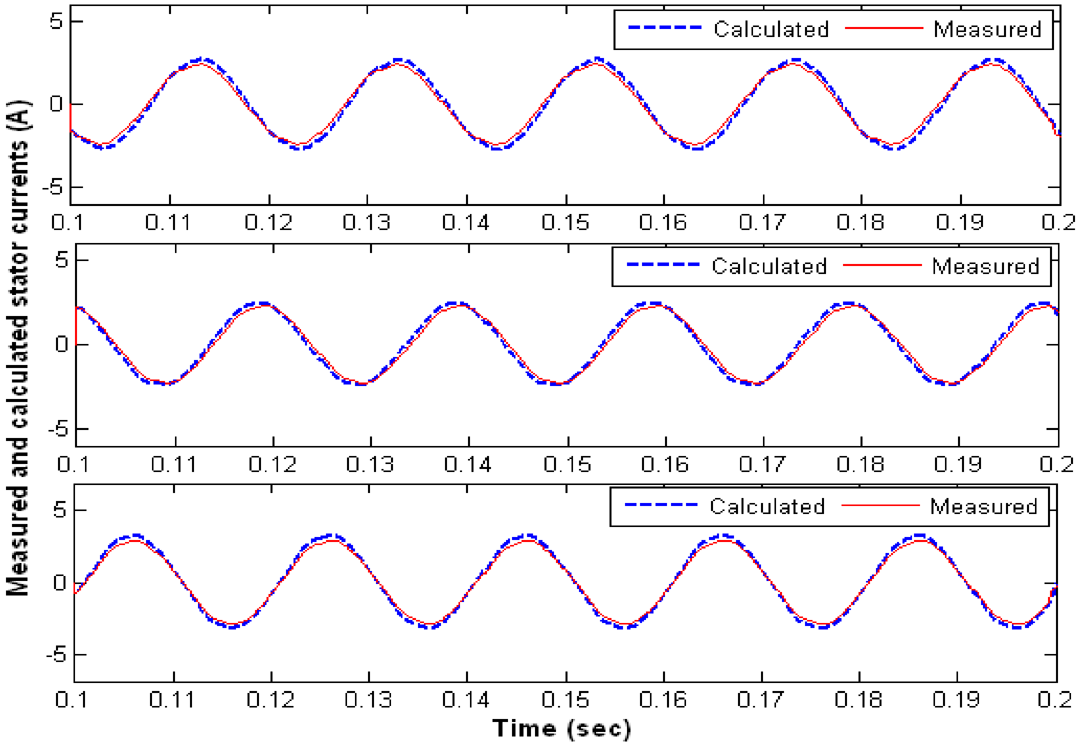


Figure 5.14. Measured (I_A, I_B, I_C) and calculated (I_{sA}, I_{sB}, I_{sC}) stator current waveforms using the estimated resistances obtained from GA for operation of induction motor with rotor open-circuit fault.

5.3.2 Inverter-fed induction motor

Measured PWM inverter output voltage waveforms are not used in this part of the study because it was not possible to sample the waveform at a sufficiently high rate to avoid aliasing effects. To demonstrate the operation of the GA algorithm when used in conjunction with an inverter fed induction motor, a simple computer model of the inverter PWM output voltage waveform is used to supply the model of induction motor to obtain one set of results (machine current waveforms, I_A, I_B, I_C) while the second set of results (machine currents I_{sA}, I_{sB}, I_{sC}) is supplied from the induction machine model with the PWM modulating signal (with the appropriate adjustment for the magnitude of the voltages) used as the input voltage (see Figure 5.15). The task of the stochastic algorithm is to estimate the values of the stator and rotor resistances values which give the currents (I_{sA}, I_{sB}, I_{sC}) that produce the minimum error value between the two sets of currents.

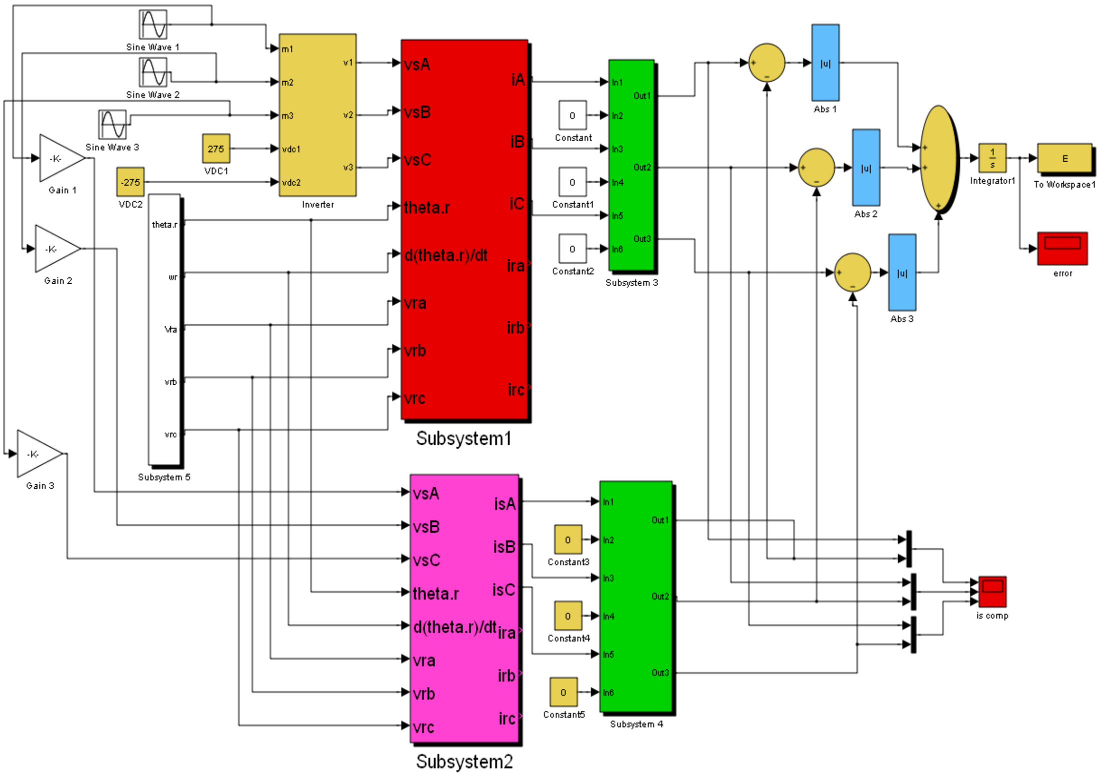


Figure 5.15 Simulink model arrangement used to assess the operation of the stochastic algorithms in conjunction with an inverter-fed machine

5.3.2.1 Stator open-circuit winding fault

A developing stator open-circuit winding fault is once again emulated by connecting an external resistance in series with a stator phase. Then a test is conducted to estimate the machine winding parameters as previously described.

In this test, the stator was supplied at a frequency of 40 Hz which is equivalent to a synchronous speed of 2400 rpm. It can be seen from Figure 5.16 that there is an obvious increase in the value of R_{sB} to 12.923 Ω while the other estimated stator resistances are approximately at their nominal values. There is no change in rotor resistance values because no fault was introduced into those windings (see Figure 5.17). The corresponding values of the IAE function are shown in Figure 5.18. The calculation error has a maximum of 0.0438 A.s before settling down to 0.008 A.s.

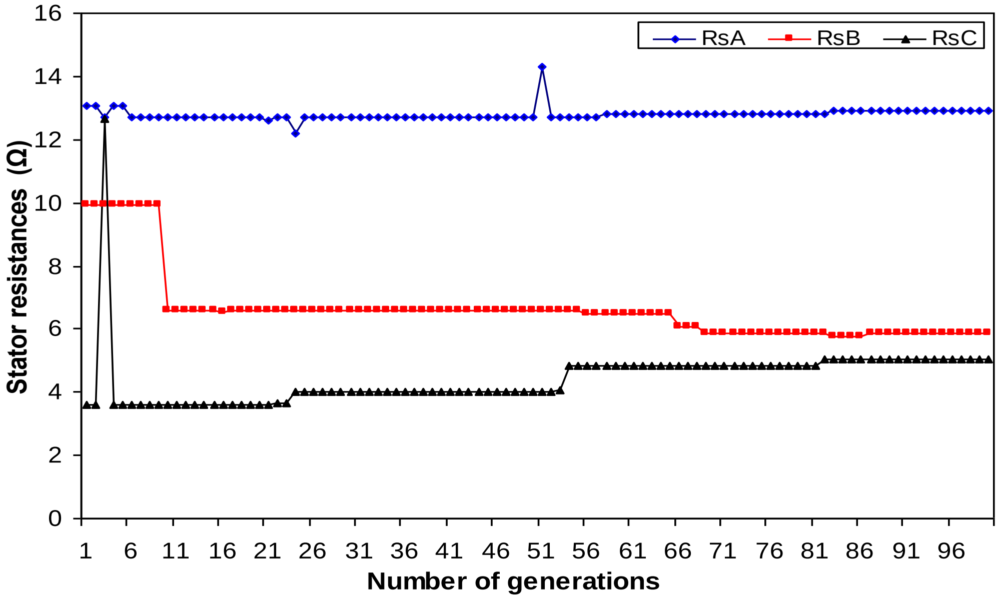


Figure 5.16. Estimated stator resistances obtained using GA for operation of inverter-fed induction motor with stator open-circuit fault at 40 Hz stator frequency.

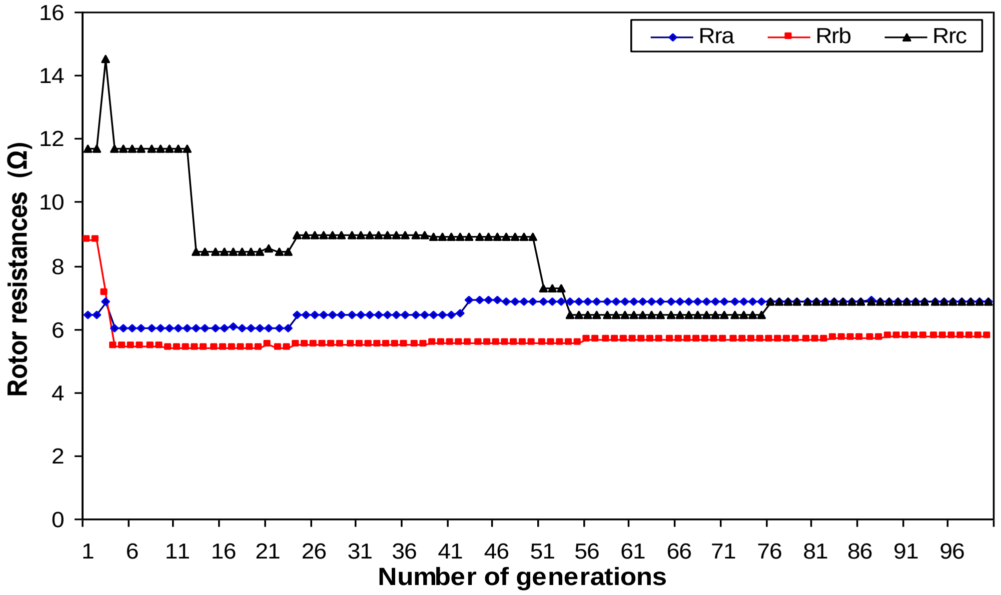


Figure 5.17. Estimated rotor resistances obtained using GA for operation of inverter-fed induction motor with stator open-circuit fault at 40 Hz stator frequency.

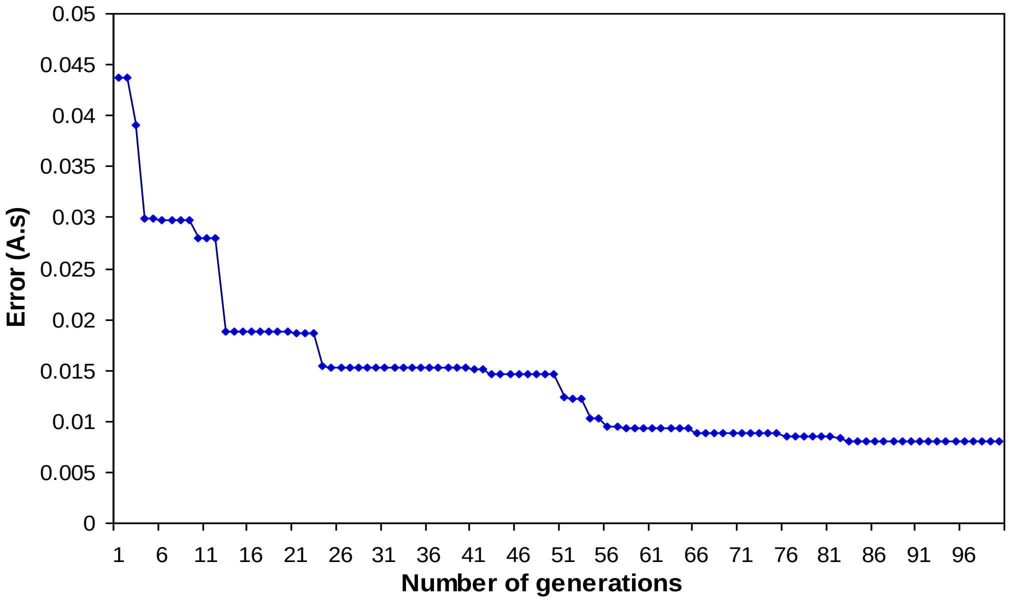


Figure 5.18. Estimated calculation error obtained using GA for operation of inverter-fed induction motor with stator open-circuit fault at 40 Hz stator frequency.

About 50 investigations of potential solutions were required to obtain convergence with this data set. The stator and rotor resistances obtained at the end of the optimisation process are shown in Table 5.6. Figure 5.19 shows that the two sets of stator currents (I_A, I_B, I_C) and (I_{sA}, I_{sB}, I_{sC})**,** using the final parameter values obtained by the GA algorithm, are in good agreement.

Table 5.6. Final values of winding resistances obtained using GA with stator open-circuit fault at 40 Hz stator frequency.

R_{sA} (Ω)	R_{sB} (Ω)	R_{sC} (Ω)	R_{ra} (Ω)	R_{rb} (Ω)	R_{rc} (Ω)
12.922	5.864	5.058	6.898	5.768	6.868

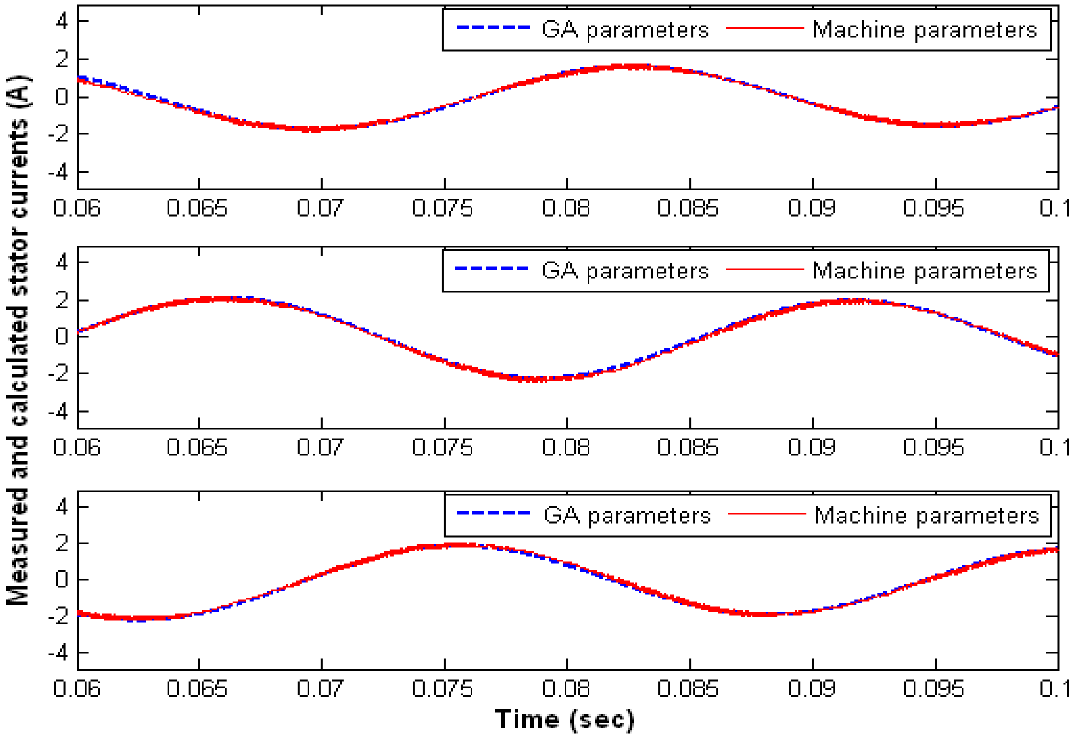


Figure 5.19. Stator current waveforms (I_A, I_B, I_C) and (I_{sA}, I_{sB}, I_{sC}) for inverter-fed induction motor with stator open-circuit fault; GA algorithm, 40 Hz stator frequency.

5.3.2.2 Rotor open-circuit winding fault

A rotor winding open circuit fault was emulated and tests conducted to estimate the machine winding resistances, as previously described for the stator fault results.

Figure 5.20 and Figure 5.21 show the results when the stator was run at a frequency of 40 Hz (equivalent to a synchronous speed of 2400 rpm). Clearly, the GA algorithm can successfully identify the presence of the rotor winding fault as indicated by the high values of R_{rb} and R_{ra} compared with R_{rc} while the stator resistances remain at approximately their nominal values. The number of investigations of potential solutions required to obtain convergence with this data set was 16. The error corresponding to the existing best solution is shown in Figure 5.22.

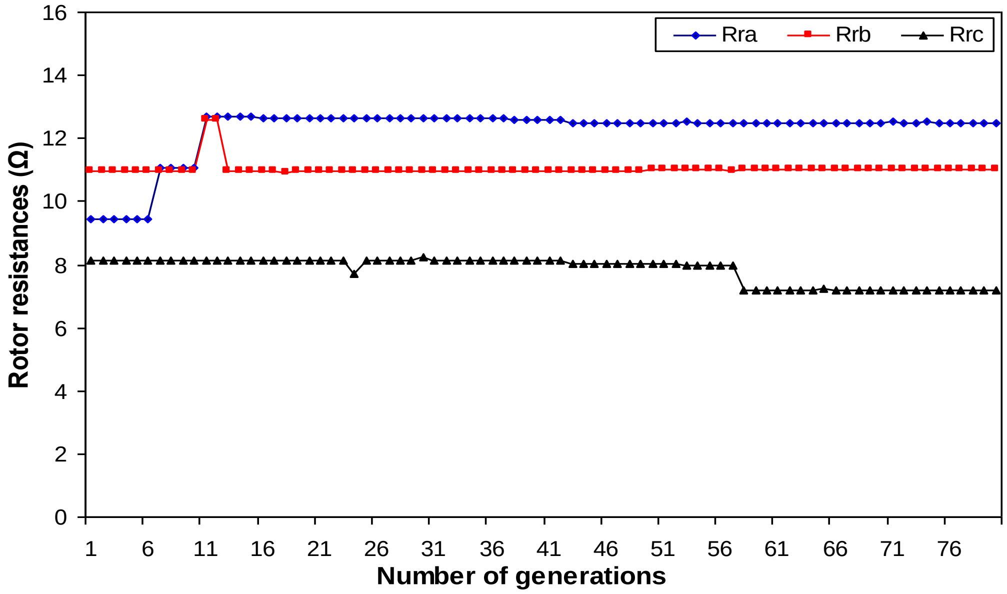


Figure 5.20. Estimated rotor resistances obtained using GA for operation of inverter-fed induction motor with rotor open-circuit fault at 40 Hz stator frequency.

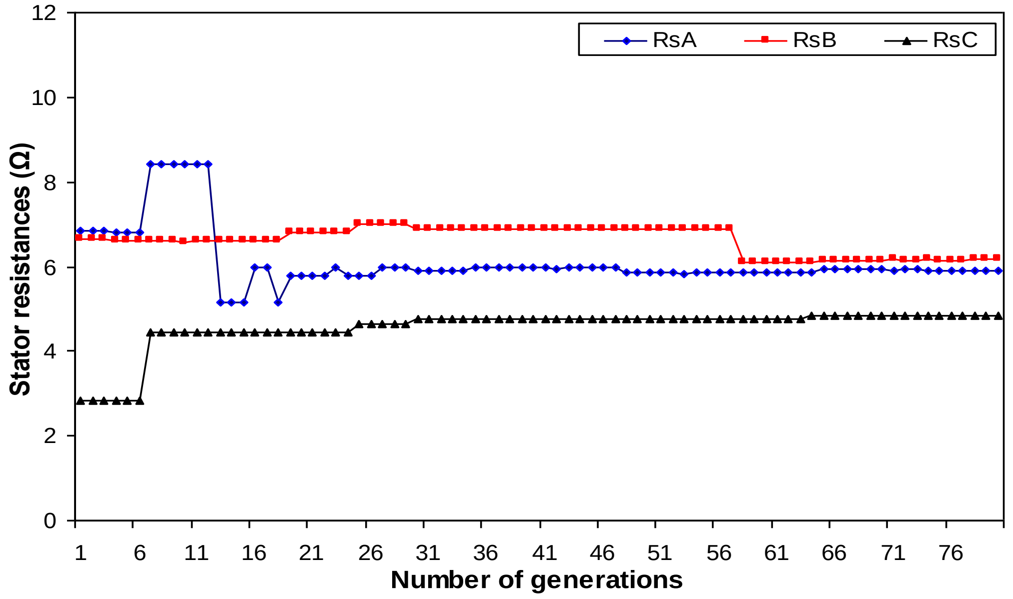


Figure 5.21. Estimated stator resistances obtained using GA for operation of inverter-fed induction motor with rotor open-circuit fault at 40 Hz stator frequency.

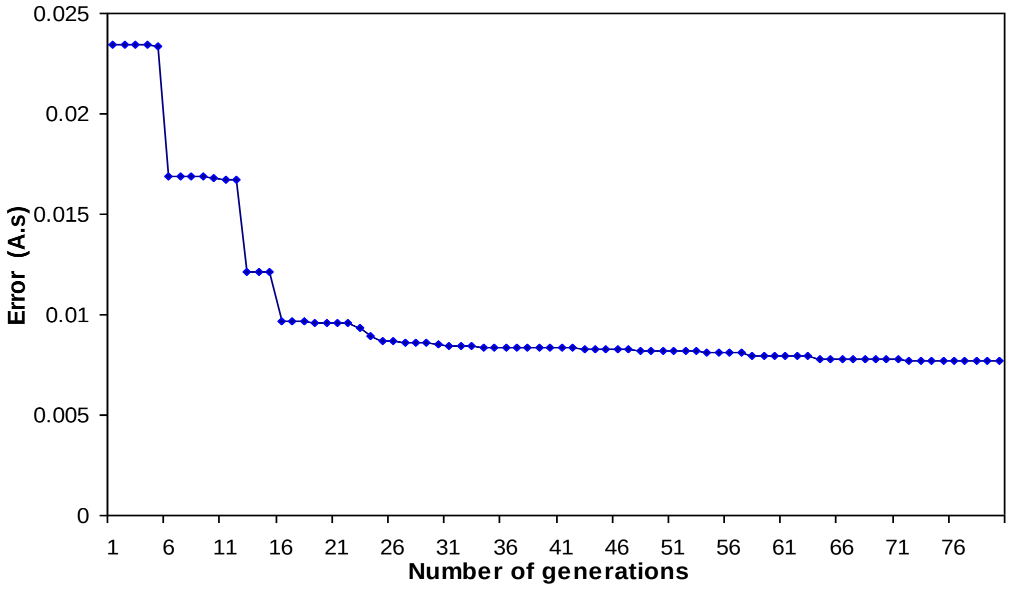


Figure 5.22. Estimated calculation error obtained using GA for operation of inverter-fed induction motor with rotor open-circuit fault at 40 Hz stator frequency.

The stator and rotor resistance values obtained at the end of the optimisation process are listed in Table 5.7. It is clear from Figure 5.23 that there is reasonable agreement between the two sets of currents (I_A, I_B, I_C) and (I_{sA}, I_{sB}, I_{sC}).

Table 5.7 Final values of winding resistances obtained using GA with rotor open-circuit fault at 40 Hz stator frequency.

R_{sA} (Ω)	R_{sB} (Ω)	R_{sC} (Ω)	R_{ra} (Ω)	R_{rb} (Ω)	R_{rc} (Ω)
5.912	6.166	4.836	12.498	10.997	7.195

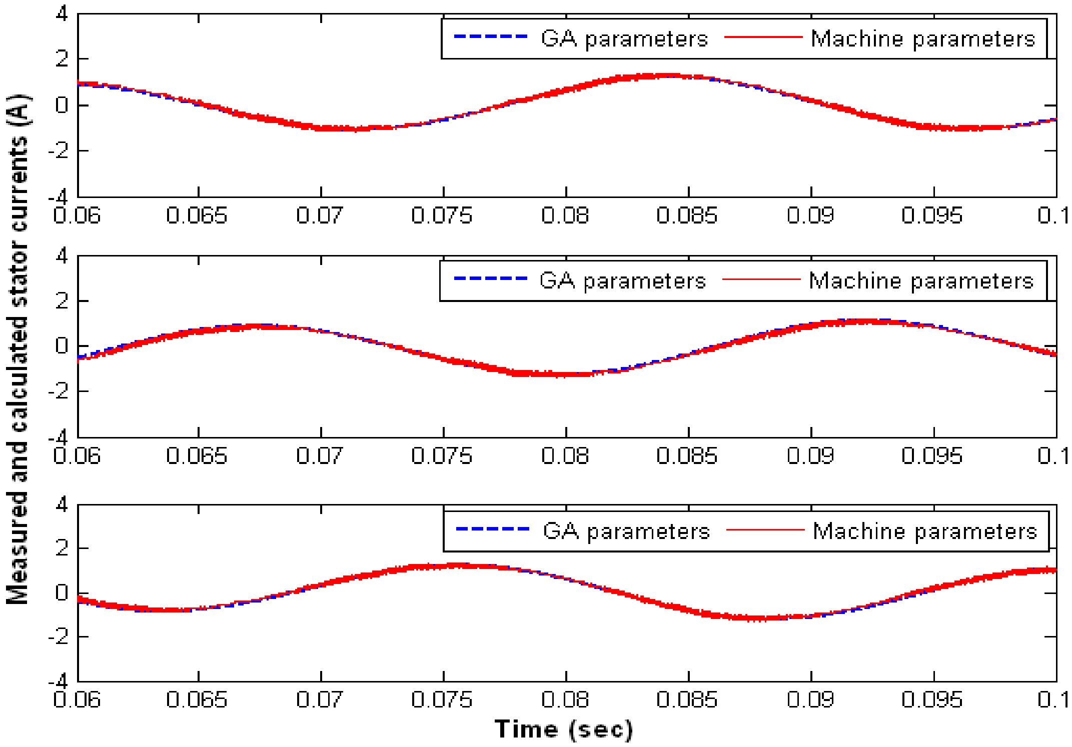


Figure 5.23. Stator current waveforms (I_A, I_B, I_C) and (I_{sA}, I_{sB}, I_{sC}) for inverter-fed induction motor with rotor open-circuit fault; GA algorithm, 40 Hz stator frequency.

5.4 Summary

In this chapter, GA is used in fault detection and the determination of equivalent circuit parameters of the induction motor. Based on the results obtained, it can be concluded that the GA algorithm is successful in terms of machine parameter estimation and also identifying the presence and location of an open-circuit winding fault of an induction motor. The success rate of the algorithm is calculated as the number of successes in finding the optimal solution in after a number of trials; for example, if the algorithm converges to the global optimum 14 times out of 20 trials, the success rate is 70%. In this investigation the probability of success for the GA algorithm for parameter identification and stator faults was found to be about 70% while the success rate for identifying rotor faults was found to be about 60%. In the next two chapters, the same process will be repeated using the TS and the SA algorithms.

CHAPTER 6

USE OF TABU SEARCH FOR INDUCTION MOTOR FAULT DETECTION AND PARAMETER IDENTIFICATION

6.1 Introduction

This Chapter reports the performance of the Tabu Search (TS) algorithm for parameter identification and identifying stator and rotor open-circuit winding faults using the experimental test data and technique used previously.

As mentioned in Chapter 3, the idea of TS is to allow the acceptance of non-improved solutions in order to avoid being trapped in local optima. To prevent a return to recently visited solutions, memory is used to record the moves made in the recent past of the search. The memory list with restricted moves is called a Tabu List. The TS algorithm is implemented first to estimate the parameters of the induction motor and then to detect the presence of a developing open-circuit fault in the stator and rotor winding of the induction motor when it is either supply-fed or inverter-fed. At the commencement of optimisation, the TS algorithm requires an initial set of values for the six parameters (R_{sA}, R_{sB}, R_{sC}, R_{ra}, R_{rb}, R_{rc}). The initial solution can be formed as a random generated number and a new solution can be obtained from the perturbation mechanism. The perturbation mechanism is defined as the method of creating new solutions from the current solution.

6.2 Induction machine parameter identification using TS

In this section, TS is employed to identify the induction motor parameters. The data used to determine these parameters are measured stator currents, stator voltages and rotor speed as previously described when illustrating the use of the GA algorithm. To implement the TS algorithm, random values are first chosen as current solutions at the beginning of the iteration process which represent a set of solution candidates for the parameters (R_s, R_r, L_{ss}, L_{rr}, M_{ss}, M_{rr}, M_{sr}). These random solutions can be applied to the induction motor model to calculate the cost function, and then it is possible to obtain new solutions using the perturbation mechanism.

If the new solution is not tabu, it will become the current solution even if it is worse than the previous one. This is done by using a memory that allows TS to explore the search space while preventing cycles in the search. The search space in TS has been set as with the GA algorithm. The length of the tabu list is a very important parameter in TS and is chosen in this study as 7 [61]; if the tabu list is too long the quality of the solution starts to deteriorate. The implementation of the TS algorithm has been described in Chapter 3 and shown in a flow chart in Figure 3.3. The code comprising the TS algorithm is included in Appendix B2.

Around 20 investigations of potential solutions are required within the search space to obtain convergence of machine parameters when the algorithm finally settles on the final values of the machine parameters, which are shown in Table 6.1.

Table 6.1. Final values of machine parameters obtained using TS algorithm.

R_S (Ω)	R_r (Ω)	L_{ss} (H)	L_{rr} (H)	M_{SS} (H)	M_{rr} (H)	M_{sr} (H)
5.601	7.510	0.896	0.507	0.185	0.771	0.751

Figure 6.1 and Figure 6.2 show the results of the search. The TS algorithm has a shorter convergence time than the GA algorithm. The TS algorithm needed about 15 evaluations of potential solutions to obtain convergence. Figure 6.3 shows how the

integral absolute error (IAE) decreases from a maximum value of 1.214 A.s to the minimum of 0.032 A.s. using 540 sec of computational time.

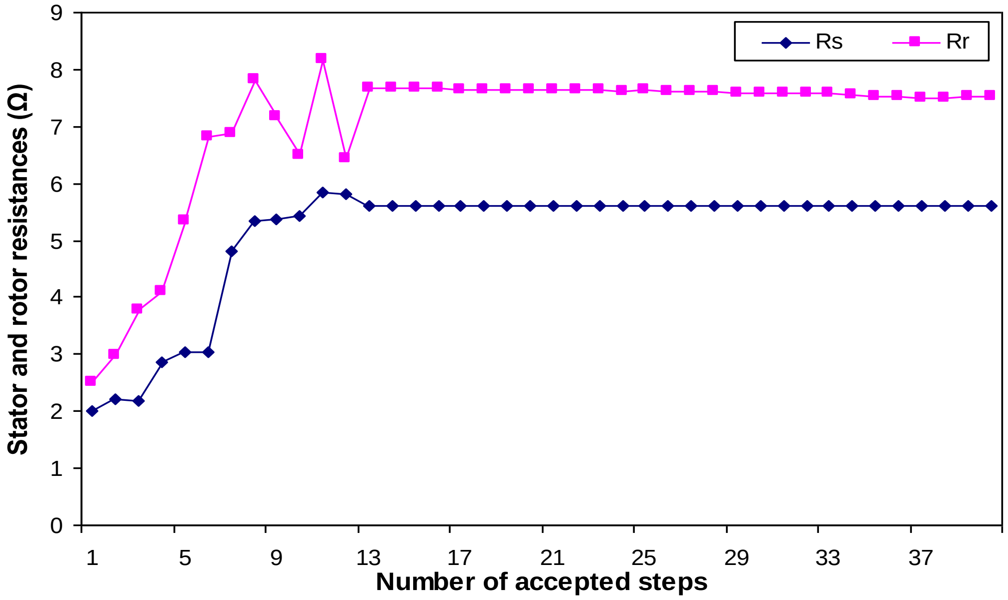


Figure 6.1. Estimated stator and rotor resistances obtained using TS algorithm.

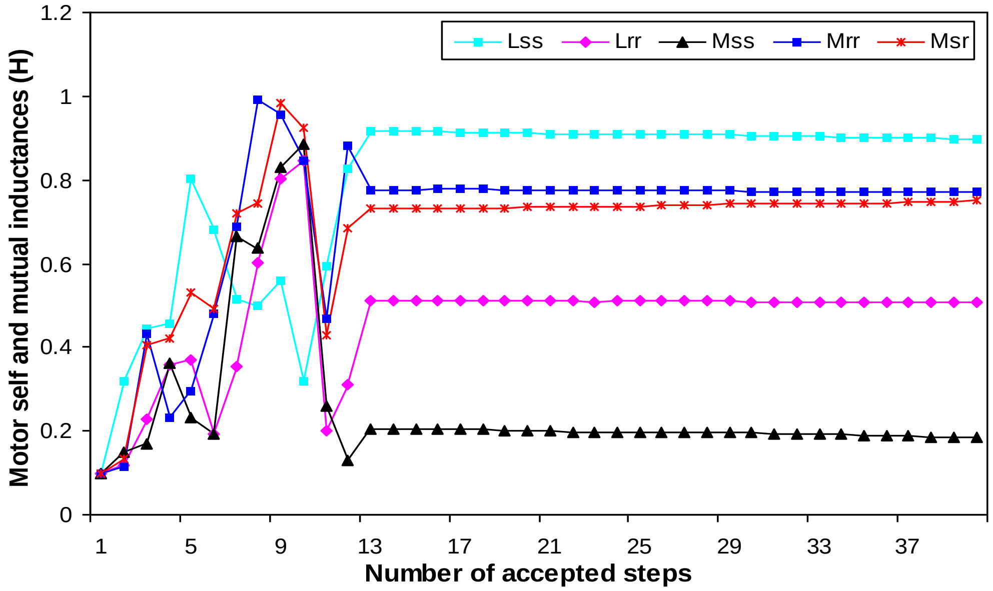


Figure 6.2. Estimated stator and rotor self and mutual inductances obtained using TS.

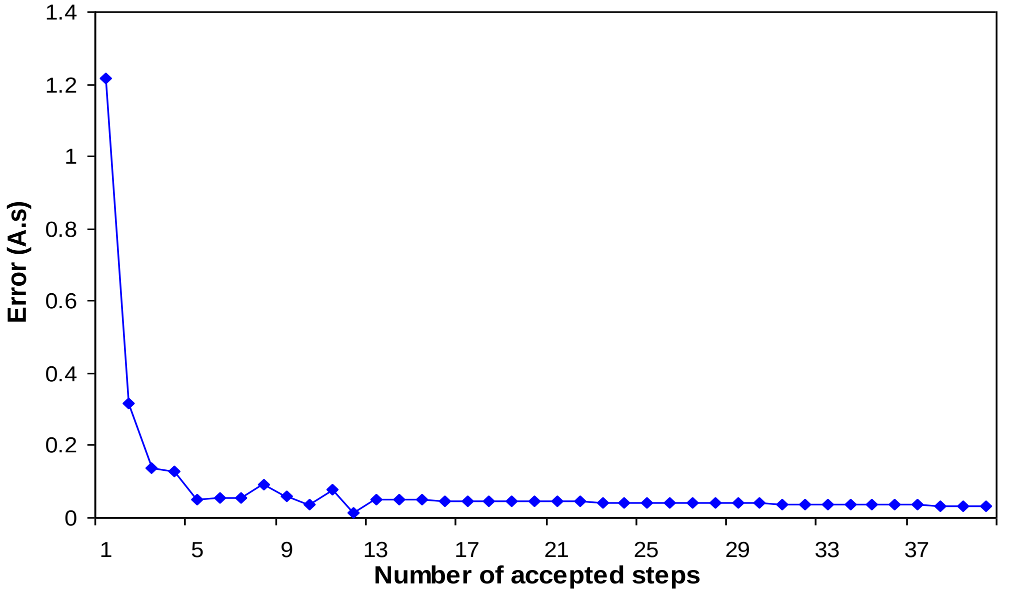


Figure 6.3. Estimated calculation error for parameter estimation obtained using TS.

Figure 6.4 shows good agreement between the measured and calculated stator current waveforms. These results strongly suggest that the TS algorithm is able to identify machine parameters with good accuracy.

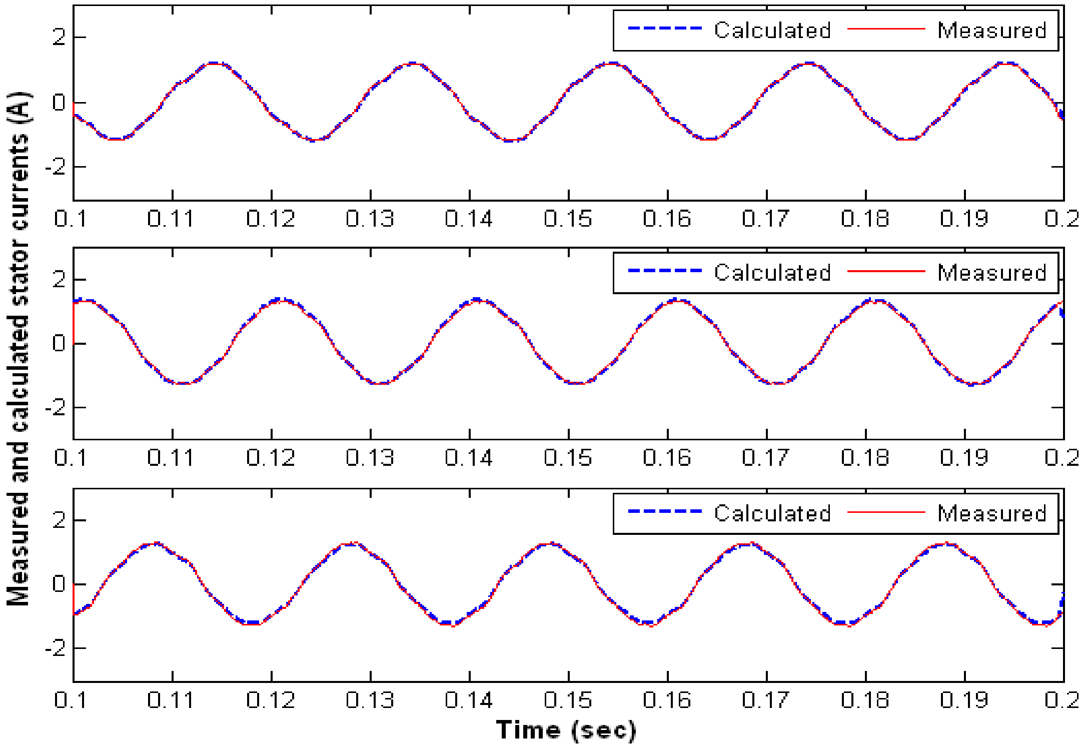


Figure 6.4. Measured (I_A, I_B, I_C) and calculated (I_{sA}, I_{sB}, I_{sC}) stator current waveforms using the estimated parameters obtained from TS.

6.3 Winding Fault detection

As described in Chapter 5, when using GA to detect the presence of a developing open-circuit in the stator and rotor windings, the same experimental test data was used by TS algorithm to estimate the values of the six winding resistances (R_{sA}, R_{sB}, R_{sC}, R_{ra}, R_{rb}, R_{rc}) of the induction motor that satisfy the condition of minimum integral absolute error (IAE). The many trial runs carried out identify that the stator and rotor resistances should be investigated while the other parameters are maintained at the values identified earlier based on the TS identification process (Table 6.1).

6.3.1 Supply-fed induction motor

6.3.1.1 Stator winding open-circuit fault

Figure 6.5 shows that the TS algorithm identifies the presence of the stator winding fault based on the resistance values obtained from the estimation. The fact that R_{sB} has a high value, 14.863 Ω compared to R_{sA} and R_{sC}, locates the stator winding fault in phase B and identifies it as an open-circuit fault. Alternatively, the estimated rotor resistances R_{sA} and R_{sC}, are approximately nominal values, as shown in Figure 6.6. This indicates that these rotor windings are in a healthy state. By optimising the parameters of the ABCabc model a very small relative value of IAE was obtained. Figure 6.7 shows the integral absolute error (IAE) decreases from a maximum value of 0.139 A.s to the minimum of 0.039 A.s.

The number of investigations of potential solutions required to obtain convergence with this data set was 30. The final estimated values of the stator and rotor resistances are listed in Table 6.2

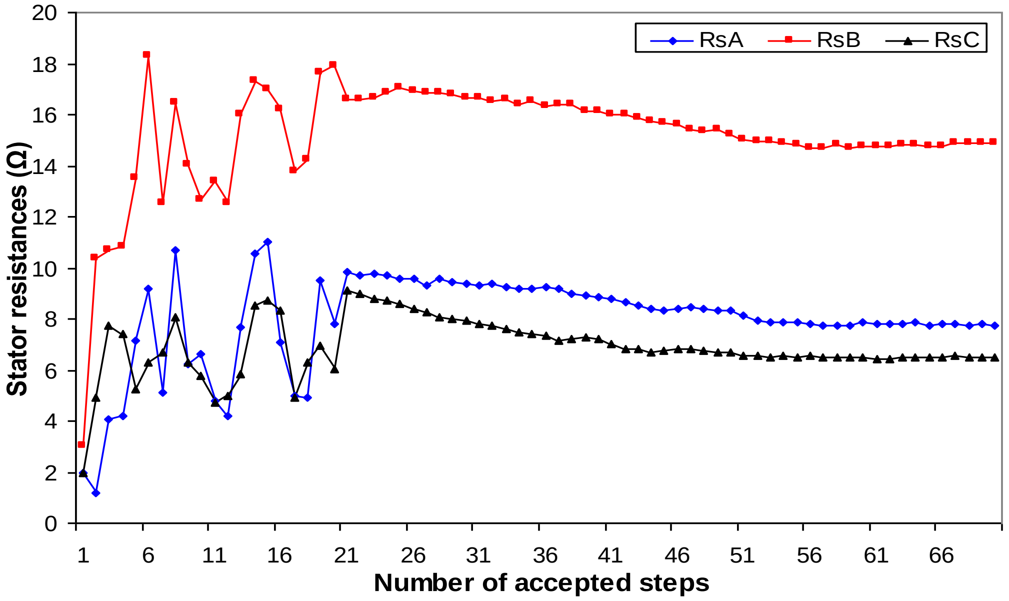


Figure 6.5. Estimated stator resistances obtained using TS for operation of induction motor with stator open-circuit fault.

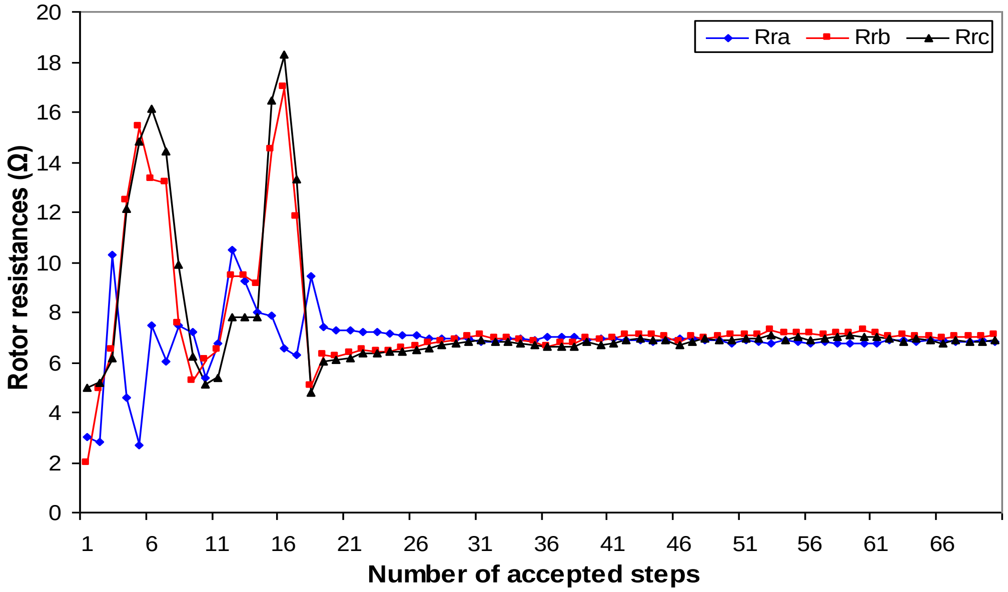


Figure 6.6. Estimated rotor resistances obtained using TS for operation of induction motor with stator open-circuit fault.

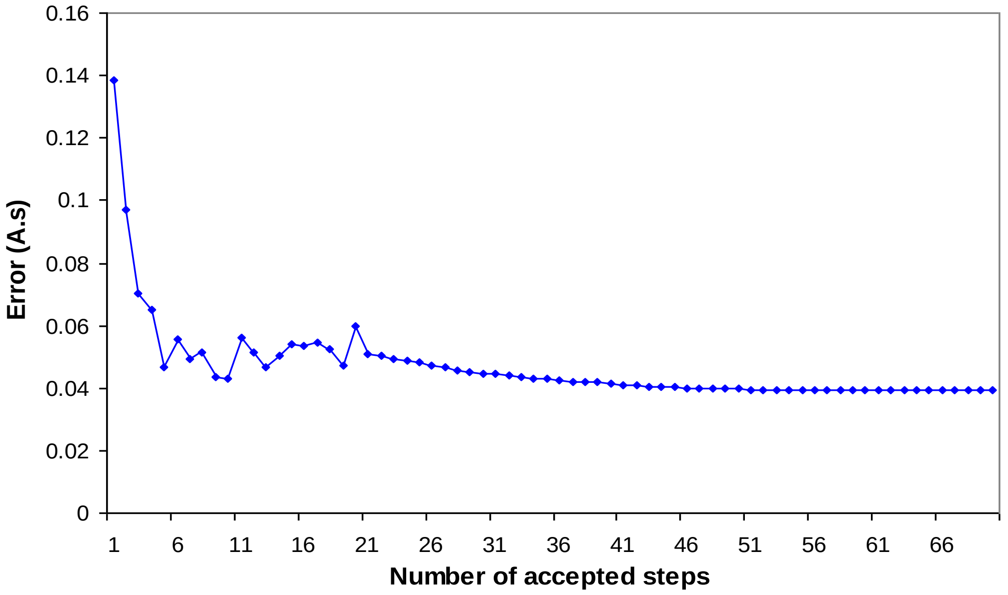


Figure 6.7. Estimated calculation error obtained using TS for operation of induction motor with stator open-circuit fault.

Table 6.2. Final values of winding resistances obtained using TS with stator open-circuit fault.

R_{sA} (Ω)	R_{sB} (Ω)	R_{sC} (Ω)	R_{ra} (Ω)	R_{rb} (Ω)	R_{rc} (Ω)
7.752	14.863	6.476	6.802	7.062	6.904

Figure 6.8 show good agreement between the measured stator currents (I_A, I_B, I_C) produced by a motor with a developing opened-circuit fault in the stator winding and the calculated stator currents (I_{sA}, I_{sB}, I_{sC}) when using the final TS parameter values.

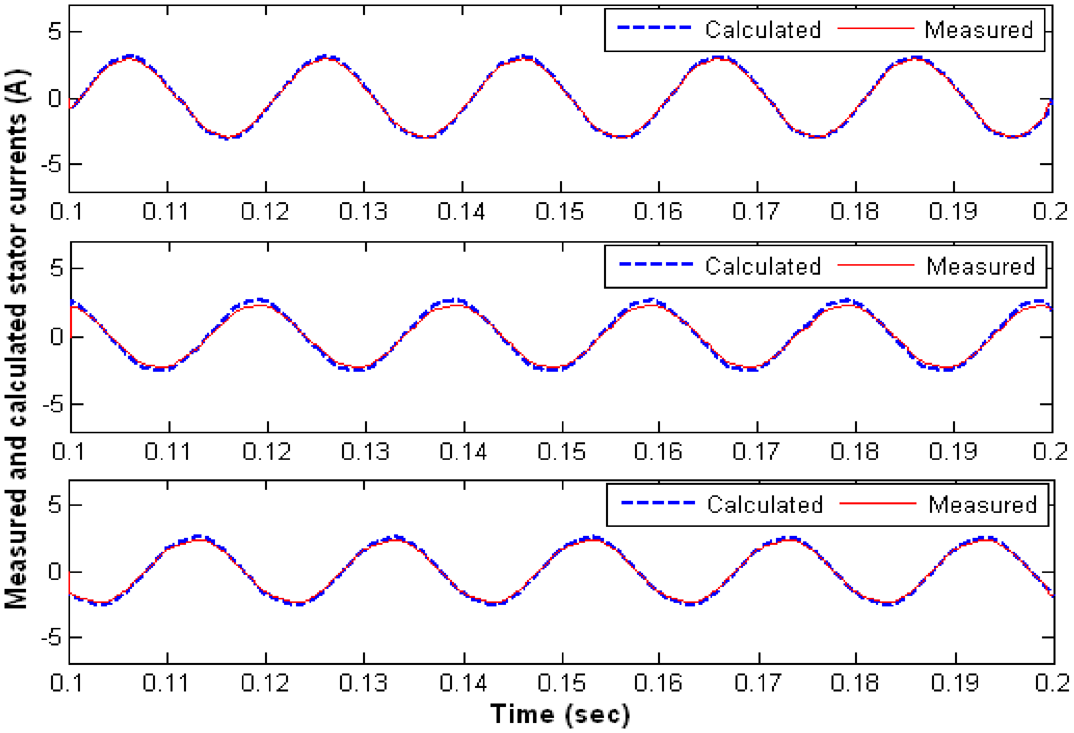


Figure 6.8. Measured (I_A, I_B, I_C) and calculated (I_{sA}, I_{sB}, I_{sC}) stator current waveforms using the estimated resistances obtained from TS for operation of induction motor with stator open-circuit fault.

6.3.1.2 Rotor winding open-circuit fault

In this test, The TS algorithm is implemented to identify the presence of a developing rotor winding open-circuit fault based on the same experimental measurements. The six winding resistances (R_{sA}, R_{sB}, R_{sC}, R_{ra}, R_{rb}, R_{rc}) are again the parameters to be optimised so that the IAE is minimised. It can be seen from Figure 6.9 that there is an open-circuit fault in the rotor phases A and B; this is because of the high values R_{ra} and R_{rb} compared with R_{rc} while the stator resistances are in a healthy state (Figure 6.10).

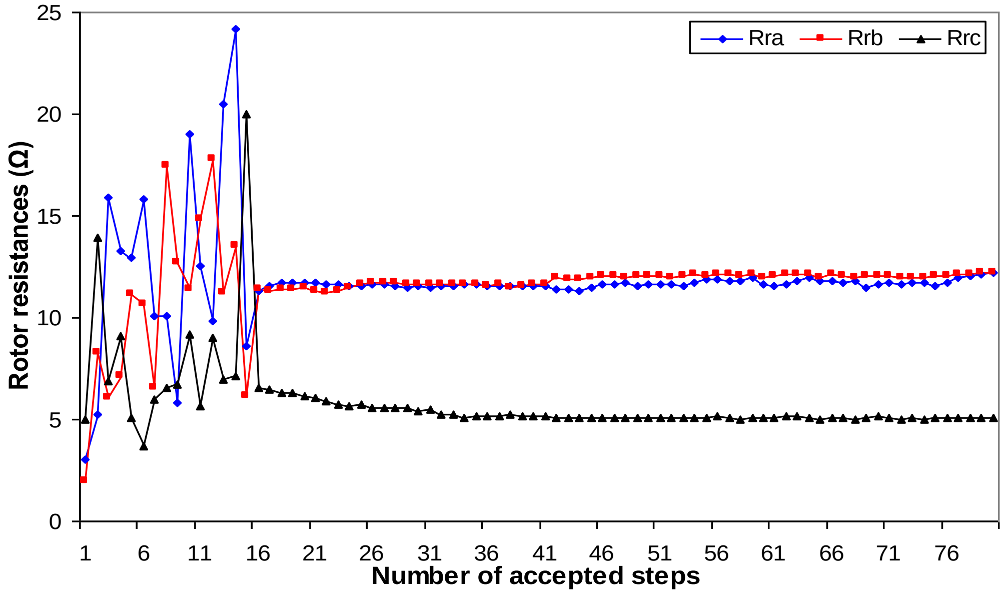


Figure 6.9. Estimated rotor resistances obtained using TS for operation of induction motor with rotor open-circuit fault.

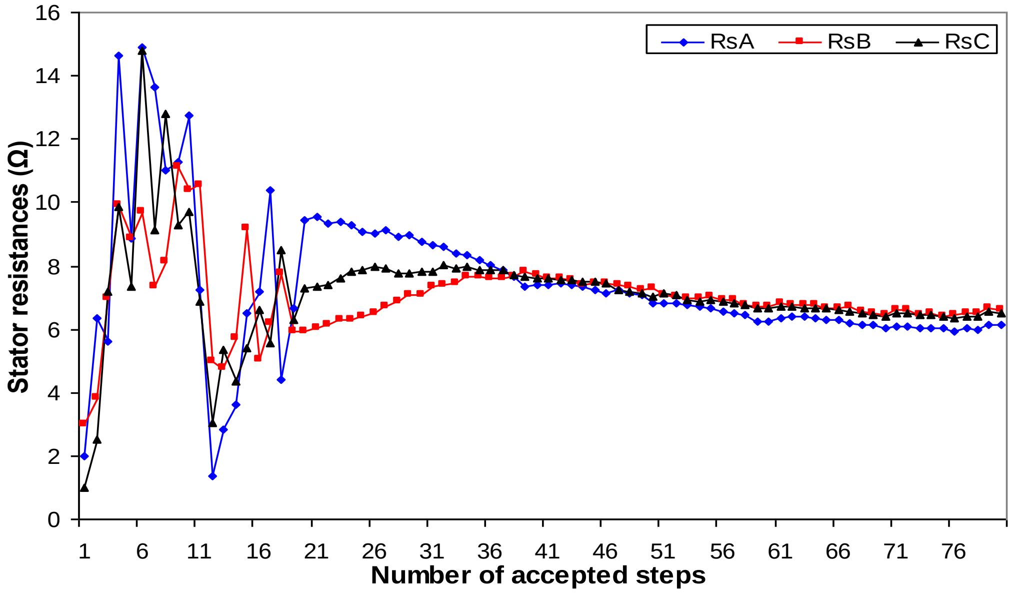


Figure 6.10. Estimated stator resistances obtained using TS for operation of induction motor with rotor open-circuit fault.

The number of investigations of potential solutions required to obtain convergence with this data set was 36. The calculated IAE error falls from a maximum value of 0.368 A.s to 0.007 A.s as shown in Figure 6.11. The Final values of the estimated stator and rotor resistances are listed in Table 6.3.

Figure 6.12 compares the measured (I_A, I_B, I_C) and calculated (I_{sA}, I_{sB}, I_{sC}) stator currents obtained by the TS algorithm and show reasonable agreement between the two current waveforms.

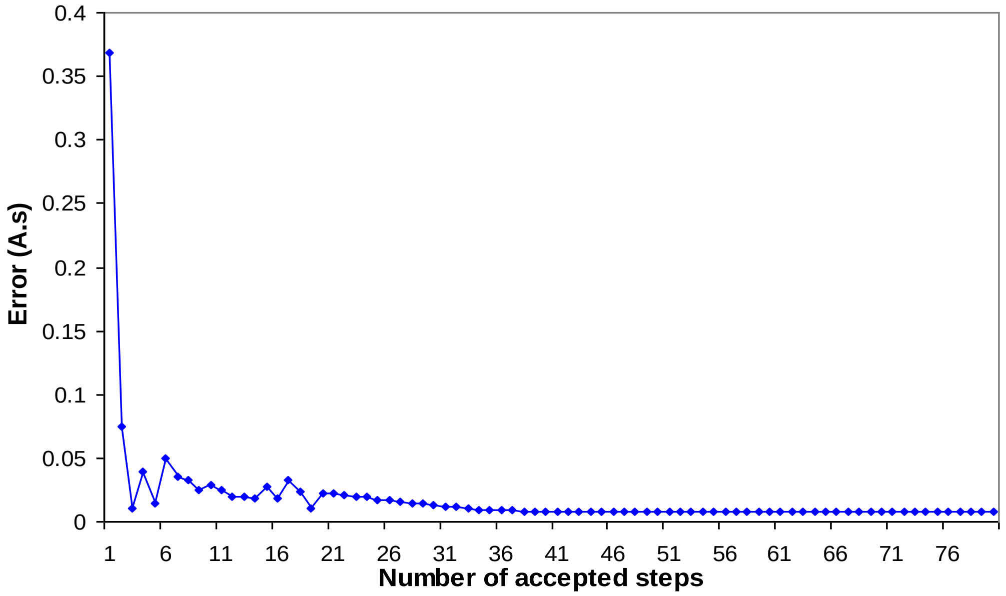


Figure 6.11. Estimated calculation error obtained using TS for operation of induction motor with rotor open-circuit fault.

Table 6.3. Final values of winding resistances obtained using TS with rotor open-circuit fault.

R_{sA} (Ω)	R_{sB} (Ω)	R_{sC} (Ω)	R_{ra} (Ω)	R_{rb} (Ω)	R_{rc} (Ω)
6.147	6.618	6.525	12.240	12.222	5.054

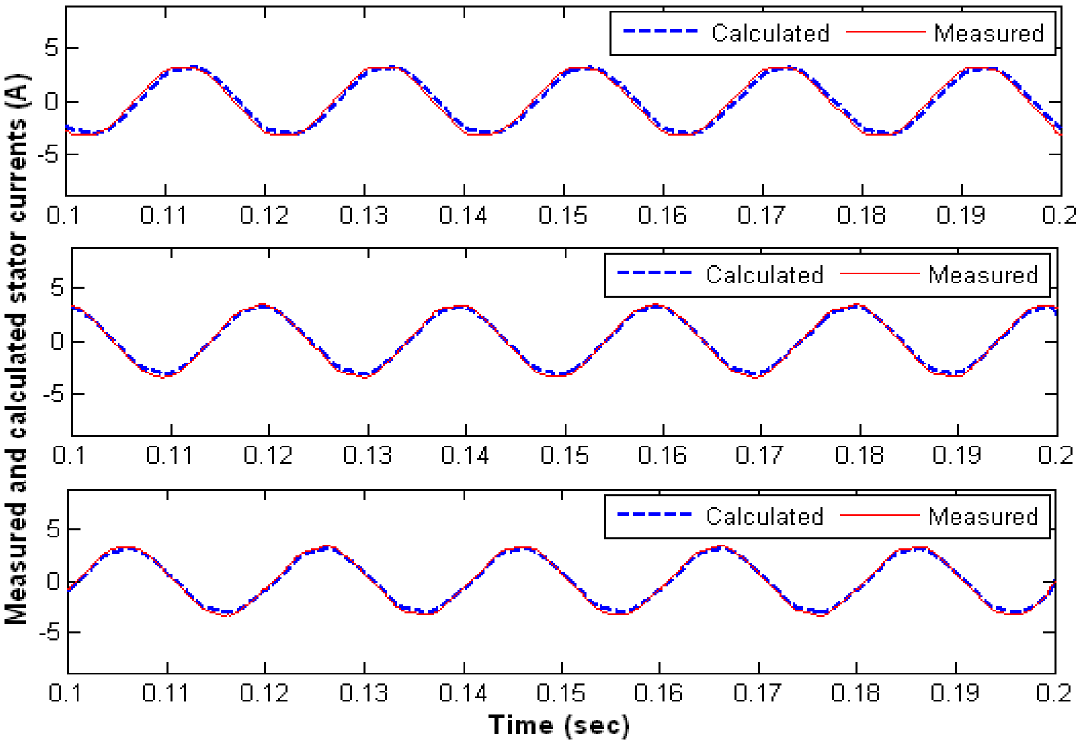


Figure 6.12. Measured (I_A, I_B, I_C) and calculated (I_{sA}, I_{sB}, I_{sC}) stator current waveforms using the estimated resistances obtained from TS for operation of induction motor with rotor open-circuit fault.

6.3.2 Inverter-fed induction motor

6.3.2.1 Stator winding open-circuit fault

In this simulation test, the stator was supplied at a frequency of 40 Hz which is equivalent to a synchronous speed of 2400 rpm (see section 5.3.2 for a description of the test arrangement). The TS algorithm uses the stator currents (I_{sA}, I_{sB}, I_{sC}) to estimate the stator and rotor winding resistances (R_{sA}, R_{sB}, R_{sC}, R_{ra}, R_{rb}, R_{rc}). It can be seen from Figure 6.13 that the stator resistance of phase A is high at 15.472 Ω compared to the other two resistances which are at approximately their nominal values. Figure 6.14 shows that all identified rotor resistances are approximately at the same value, close to their nominal values, confirming that rotor resistances are not affected by the presence of the open-circuit fault in the stator windings. The type and location of the fault was identified by adjusting the ABCabc model parameters until the IAE was minimised.

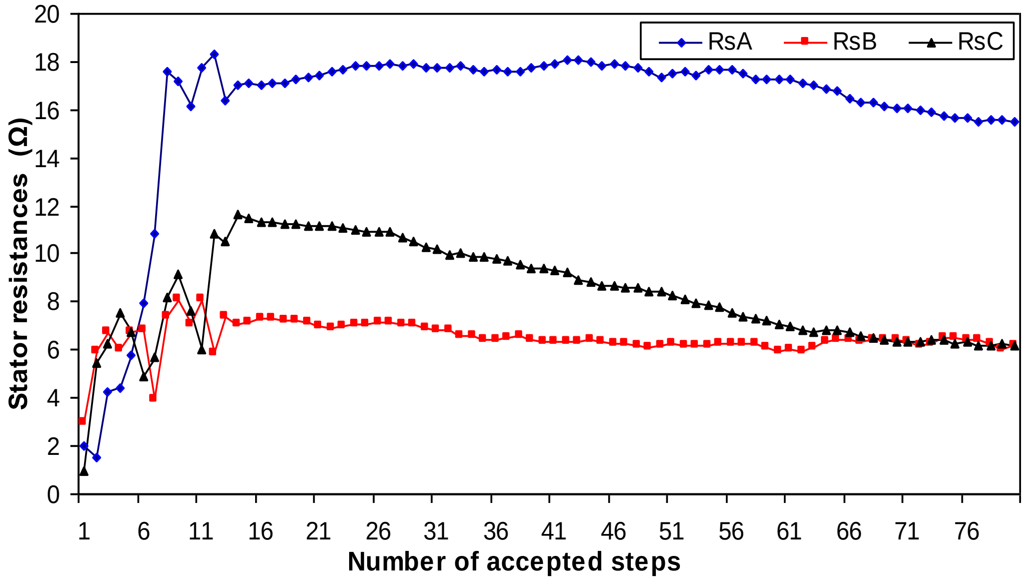


Figure 6.13. Estimated stator resistances obtained using TS for operation of inverter-fed induction motor with stator open-circuit fault at 40 Hz stator frequency.

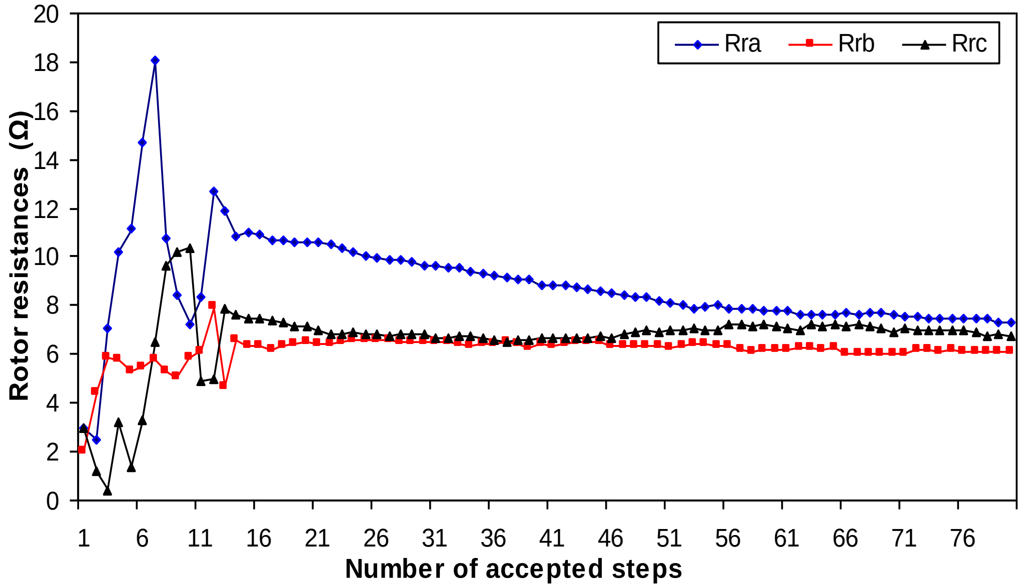


Figure 6.14. Estimated rotor resistances obtained using TS for operation of inverter-fed induction motor with stator open-circuit fault at 40 Hz stator frequency.

The number of investigations of potential solutions required to obtain convergence with this data set was 20. The corresponding values of the IAE function are shown in Figure 6.15. The final values of the stator and rotor resistances obtained at the end of the optimisation process are listed in Table 6.4. The stator currents (I_A, I_B, I_C) show reasonable agreement with (I_{sA}, I_{sB}, I_{sC}) when using the final parameter values obtained by the TS algorithm, as illustrated in Figure 6.16.

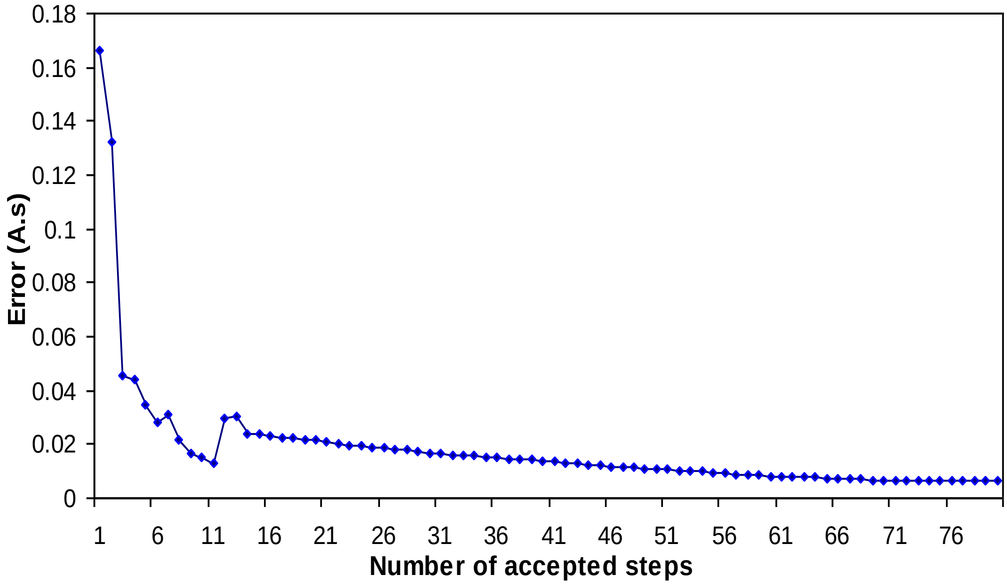


Figure 6.15. Estimated calculation error obtained using TS for operation of inverter-fed induction motor with stator open-circuit fault at 40 Hz stator frequency.

Table 6.4. Final values of winding resistances obtained using TS with stator open-circuit fault at 40 Hz stator frequency.

R_{sA} (Ω)	R_{sB} (Ω)	R_{sC} (Ω)	R_{ra} (Ω)	R_{rb} (Ω)	R_{rc} (Ω)
15.472	6.151	6.165	7.297	6.089	6.759

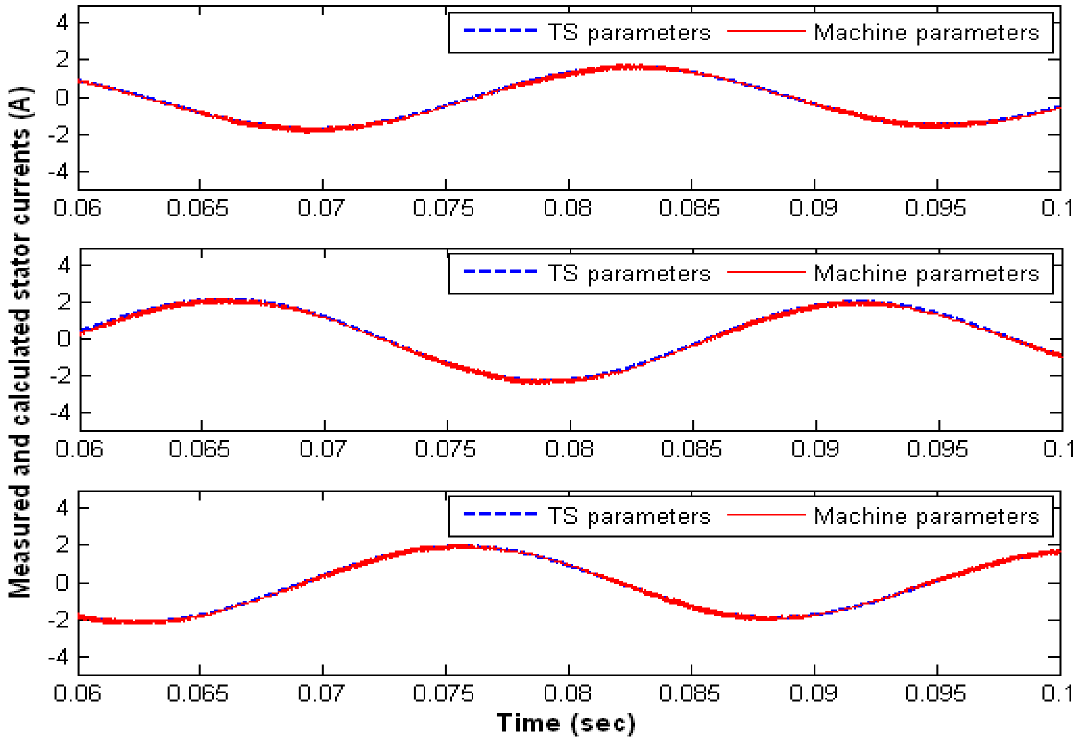


Figure 6.16. Stator current waveforms (I_A, I_B, I_C) and (I_{sA}, I_{sB}, I_{sC}) for inverter-fed induction motor with stator open-circuit fault; TS algorithm, 40 Hz stator frequency.

6.3.2.2 Rotor winding open-circuit fault

In this simulation test, the stator was supplied at a frequency of 40 Hz (equivalent to a synchronous speed of 2400 rpm). As previously stated, stator currents (I_A, I_B, I_C) and (I_{sA}, I_{sB}, I_{sC}) are calculated and used to identify the presence of a developing rotor winding open-circuit fault. Figure 6.17 shows high values of R_{rb} and R_{ra} compared with R_{rc}, and indicates that there is a fault has occurred at phases A and B. On the other hand, the stator resistances remain at approximately their nominal values, see Figure 6.18

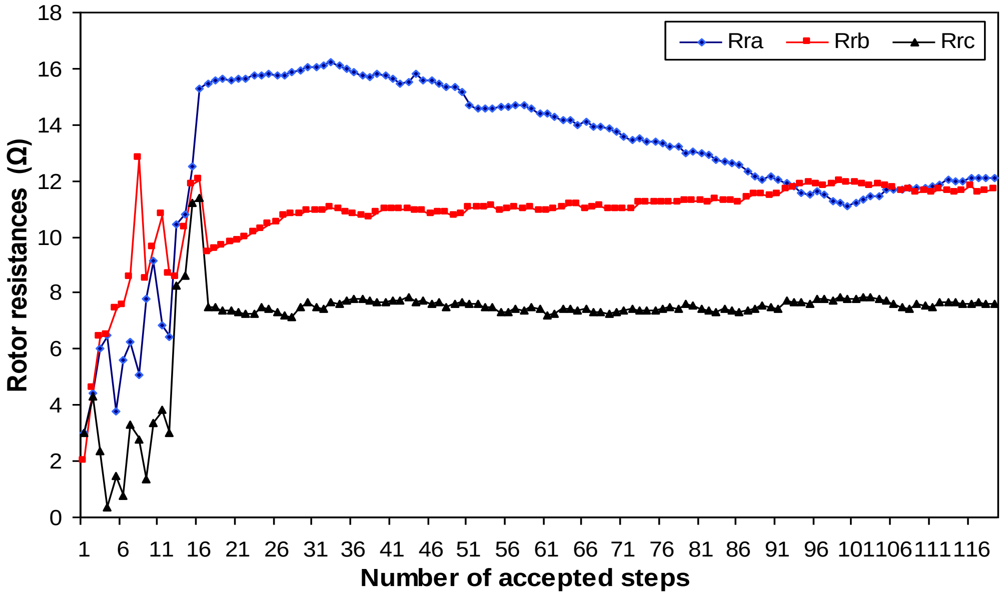


Figure 6.17. Estimated rotor resistances obtained using TS for operation of inverter-fed induction motor with rotor open-circuit fault at 40 Hz stator frequency.

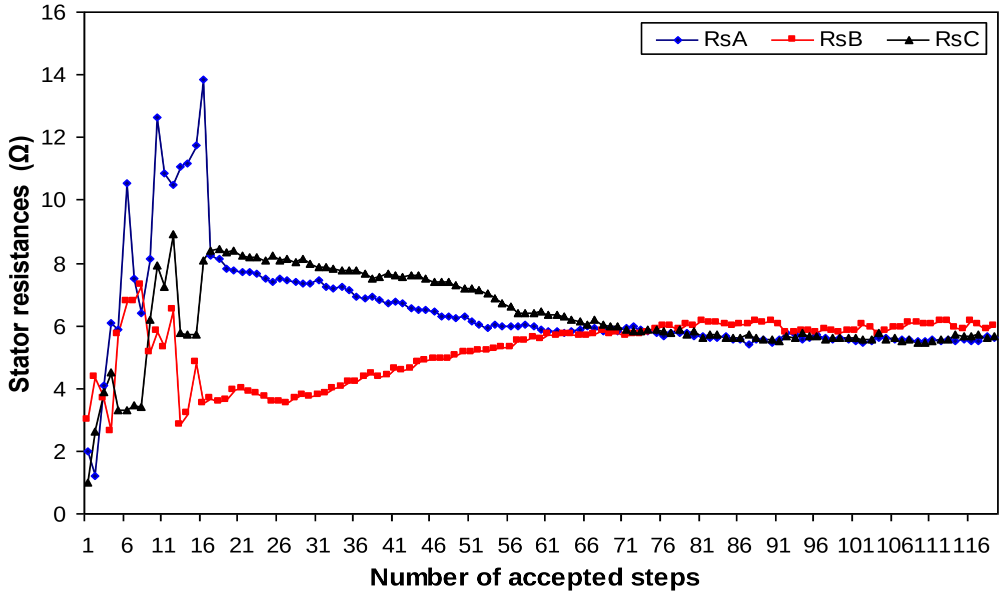


Figure 6.18. Estimated stator resistances obtained using TS for operation of inverter-fed induction motor with rotor open-circuit fault at 40 Hz stator frequency.

Table 6.5 shows the final values of the estimated stator and rotor resistances. The number of investigations of potential solutions required to obtain convergence with this data set was 60. Figure 6.19 shows the error corresponding to the best solution under this rotor open-circuit fault condition. As it can be seen from Figure 6.20, that there is a good agreement between (I_A, I_B, I_C) and (I_{sA}, I_{sB}, I_{sC}), calculated using the final TS parameter values.

Table 6.5. Final values of winding resistances obtained using TS with rotor open-circuit fault at 40 Hz stator frequency.

R_{sA} (Ω)	R_{sB} (Ω)	R_{sC} (Ω)	R_{ra} (Ω)	R_{rb} (Ω)	R_{rc} (Ω)
5.595	5.975	5.664	12.071	11.682	7.588

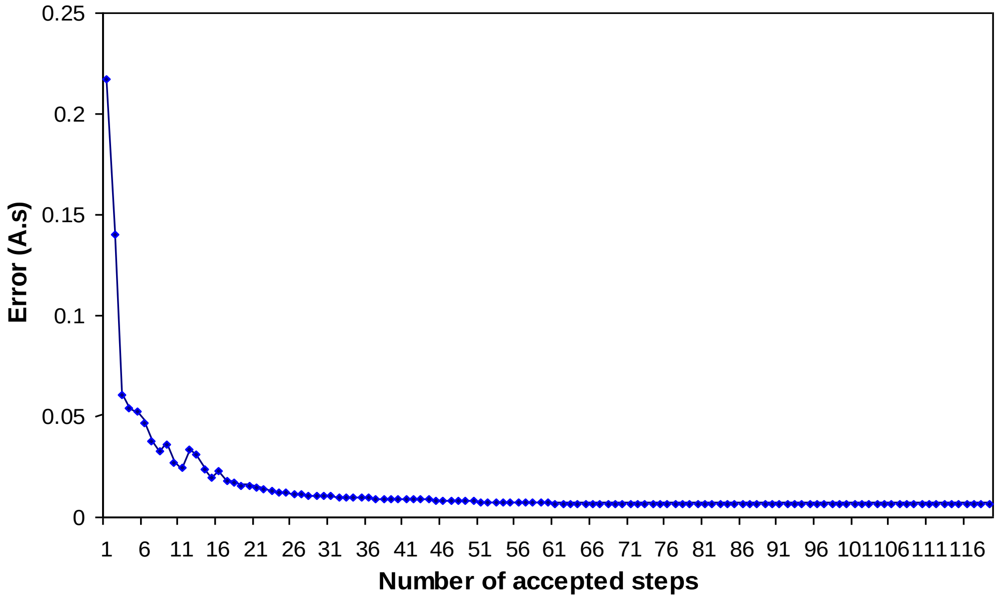


Figure 6.19. Estimated calculation error obtained using TS for operation of inverter-fed induction motor with rotor open-circuit fault at 40 Hz stator frequency.

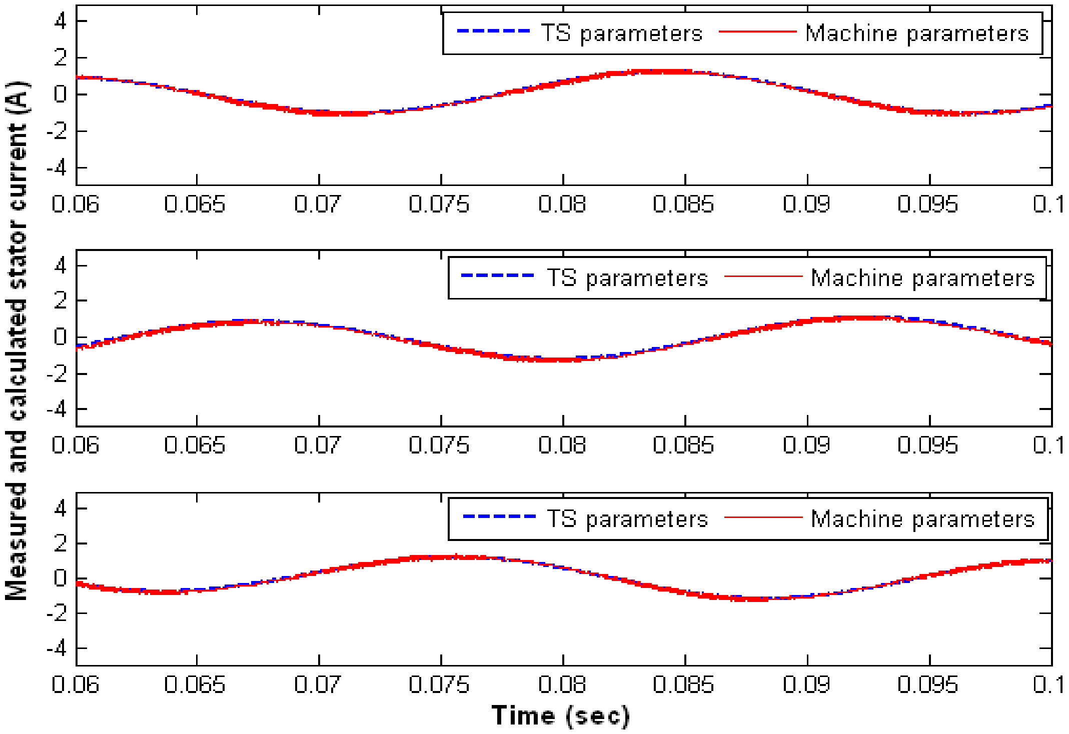


Figure 6.20. Stator current waveforms (I_A, I_B, I_C) and (I_{sA}, I_{sB}, I_{sC}) for inverter-fed induction motor with rotor open-circuit fault; TS algorithm, 40 Hz stator frequency.

6.4 Summary

This chapter has demonstrated the capability of TS in parameters identification and for identifying the presence of stator and rotor open-circuit faults in an induction motor. The results given in this chapter show that the TS is successful in identifying the parameters of an induction motor with a higher accuracy

TS had success rate of about 80% in identifying and locating the presence of open-circuit fault in both the stator and rotor windings of an induction motor in reasonable time. The next chapter presents the implementation of SA algorithm in parameter identification and fault detection.

CHAPTER 7

USE OF SIMULATED ANNEALING FOR INDUCTION MOTOR FAULT DETECTION AND PARAMETER IDENTIFICATION

7.1 Introduction

Simulated annealing (SA) is implemented using the same experimental test data and technique used previously with the GA and TS algorithms for fault detection and induction motor parameter identification. For fault detection, SA is used to detect the presence of a developing open-circuit fault in the stator and rotor winding when the motor is either supply-fed or inverter-fed. The initial temperature in the SA algorithm is set to 30 degrees.

Because the SA algorithm accepts poor solutions at high annealing temperatures, the integral absolute error (IAE) is initially at a high value. As the annealing temperature is lowered, the IAE gradually decreases as the search is concentrated in the area around the solution. In SA, the following steps are required:

- An initial solution for the stator and rotor resistances (R_{sA}, R_{sB}, R_{sC}, R_{ra}, R_{rb}, R_{rc}) is chosen and the cost function is evaluated.

- A new solution is generated from the current one by applying a generation mechanism (randomly).

- The difference in cost functions between the two solutions is calculated to decide whether or not the new solution can be accepted.

- The decision to accept new solutions is based on an acceptance criterion.

7.2 Induction machine parameter identification using SA

As explained in the previous chapters when illustrating the use of the GA and TS algorithms, the solution candidate for the parameters (R_s, R_r, L_{ss}, L_{rr}, M_{ss}, M_{rr}, M_{sr}) is applied to the induction motor model to calculate the cost function. The search space for SA is set as in GA and TS algorithms and the initial temperature, T_s was set to 30 degrees. The SA algorithm uses the acceptance probability, swap probability and cooling strategy to guide the optimisation process. The SA algorithm uses a random perturbation mechanism to generate a new solution from the current solution.

The acceptance probability, swap probability and the decrement function of temperature used in this work are given by Equations (3.4), (3.5) and (3.6), respectively, in Chapter 3. The SA code is provided in Appendix C2.

In this investigation, the SA fails to obtain convergence. These experiments are repeated for different types of temperature schedules and different values of the initial temperature but there are no improvements in the results. It is noted that increasing the number of iterations does not have an effect on the final solution.

7.3 Winding fault detection

The many trial runs carried out identify that the stator and rotor resistances should be investigated while the other machine parameters are maintained at the values identified earlier based on the TS identification process (Table 6.1).

7.3.1 Supply-fed induction motor

7.3.1.1 Stator winding open circuit fault

In this test, the same experimental test data is used as was described previously when illustrating the use of the GA and TS algorithms for stator winding open-circuit fault.

Figure 7.1 shows the estimated stator winding resistances obtained by the SA algorithm. From these results, it can be seen that there is an open-circuit fault in stator phase B while the other stator resistances are at approximately their nominal values. The three estimated rotor resistance are at their nominal values indicating a healthy state of the rotor windings (see Figure 7.2). Hence, it may be concluded that the SA algorithm can identify the presence of an open-circuit stator winding fault.

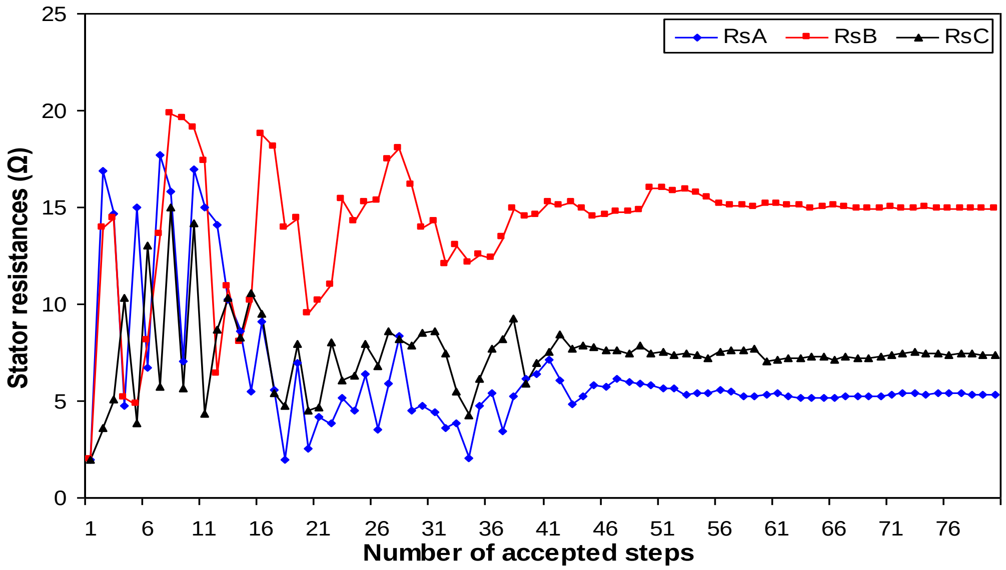


Figure 7.1. Estimated stator resistances obtained using SA for operation of induction motor with stator open-circuit fault.

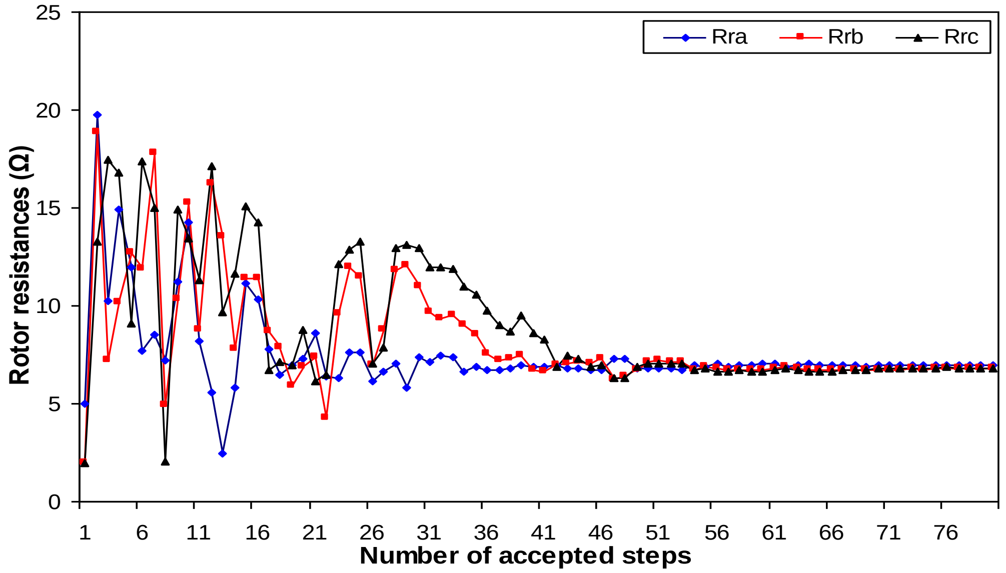


Figure 7.2. Estimated rotor resistances obtained using SA for operation of induction motor with stator open-circuit fault.

The final values of the estimated stator and rotor resistances obtained by the SA algorithm are listed in Table 7.1. Figure 7.3 shows the error function corresponding to the best solution under the stator open- circuit fault condition, the error falls from a maximum value of 0.129 A.s to 0.016 A.s.

The number of investigations of potential solutions required to obtain convergence with this data set was 40. Figure 7.4 shows that the measured (I_A, I_B, I_C) and calculated (I_{sA}, I_{sB}, I_{sC}) stator currents using the final SA parameter values are in a very good agreement.

Table 7.1. Final values of winding resistances obtained using SA with stator open-circuit fault.

R_{sA} (Ω)	R_{sB} (Ω)	R_{sC} (Ω)	R_{ra} (Ω)	R_{rb} (Ω)	R_{rc} (Ω)
5.363	14.933	7.386	6.955	6.835	6.808

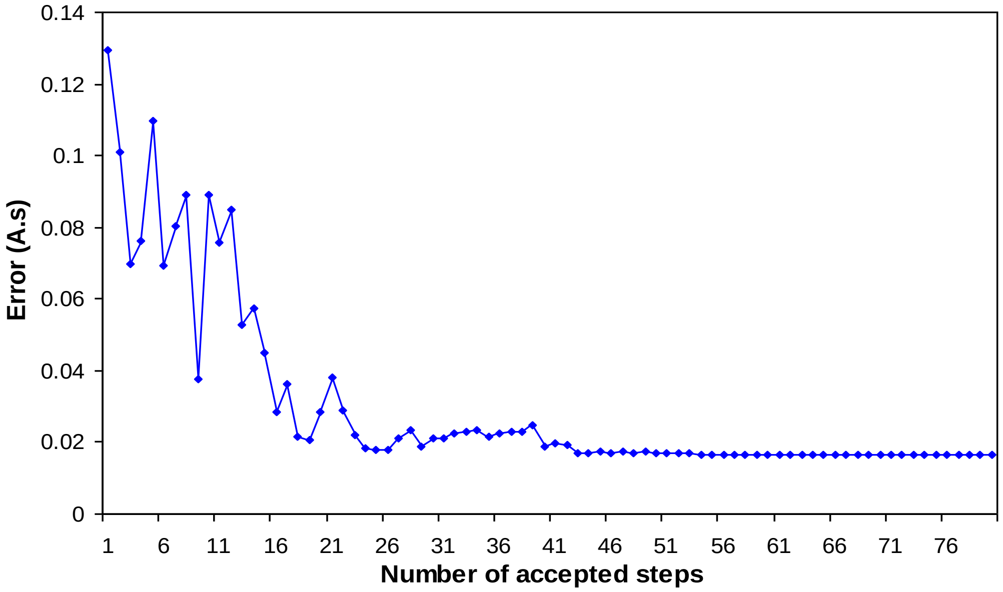


Figure 7.3. Estimated calculation error obtained using SA for operation of induction motor with stator open-circuit fault.

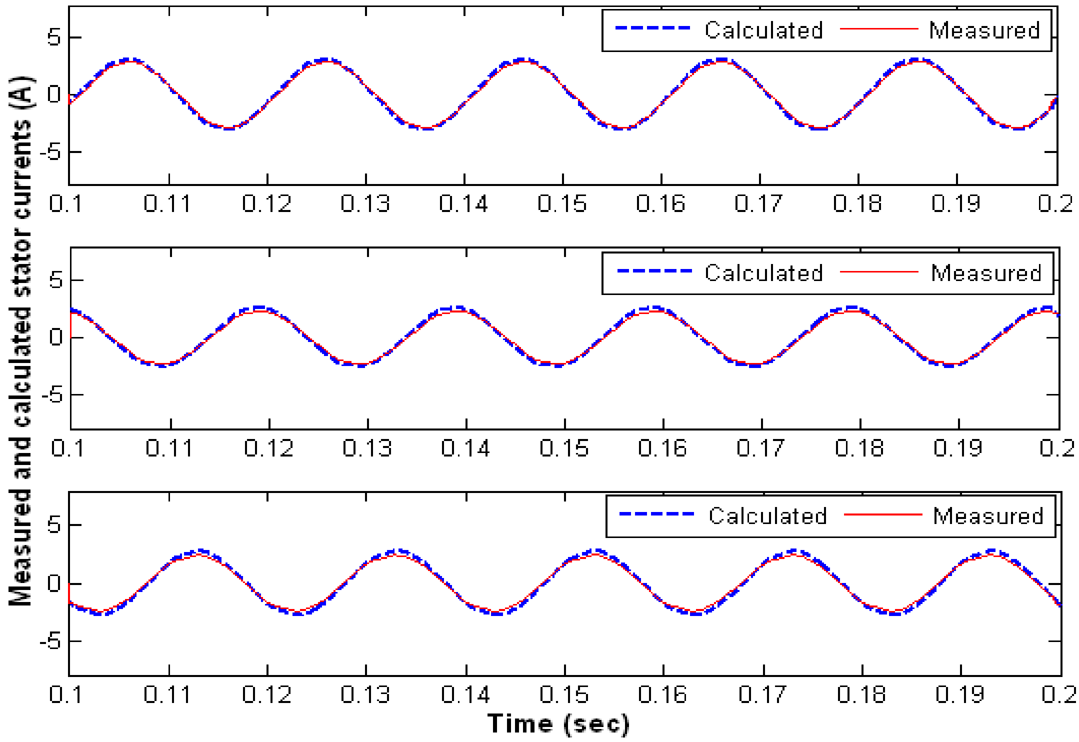


Figure 7.4. Measured (I_A, I_B, I_C) and calculated (I_{sA}, I_{sB}, I_{sC}) stator current waveforms using the estimated resistances obtained from SA for operation of induction motor with stator open-circuit fault

7.3.1.2 Rotor winding open-circuit fault

The performance of the SA algorithm in identifying open-circuit faults in the rotor using the same experimental test data is presented in this section. Figure 7.5 and Figure 7.6 show the resulting estimated rotor and stator resistances, respectively. The value of R_{rc} is equal to 5.769 Ω, and the other rotor resistances are at relatively high values (R_{ra} = 11.091 Ω , R_{rb} = 11.084 Ω); this indicates that there is an open circuit fault in both phases A and B while the stator resistances may be considered healthy.

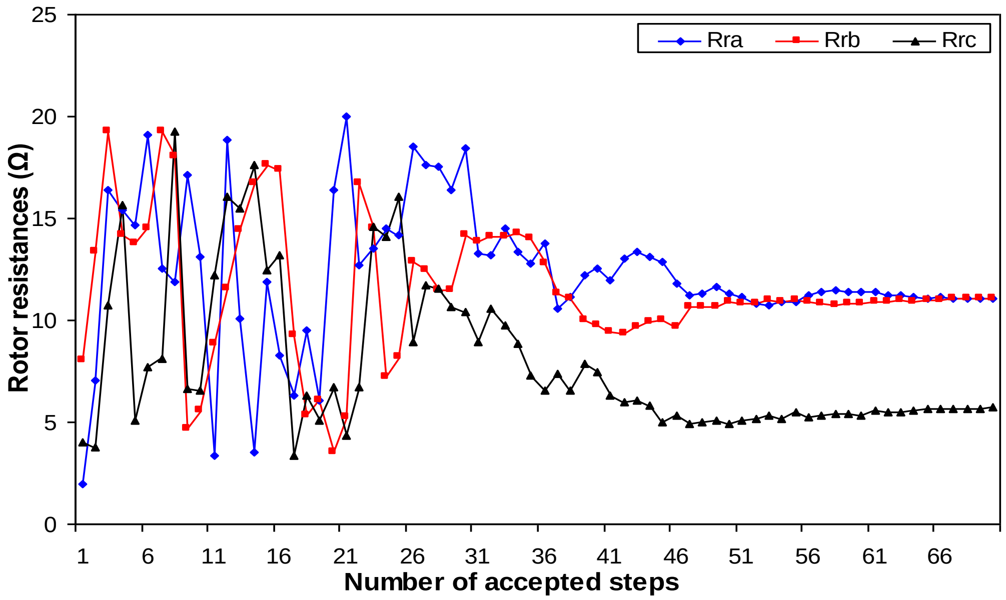


Figure 7.5. Estimated rotor resistances obtained using SA for operation of induction motor with rotor open-circuit fault.

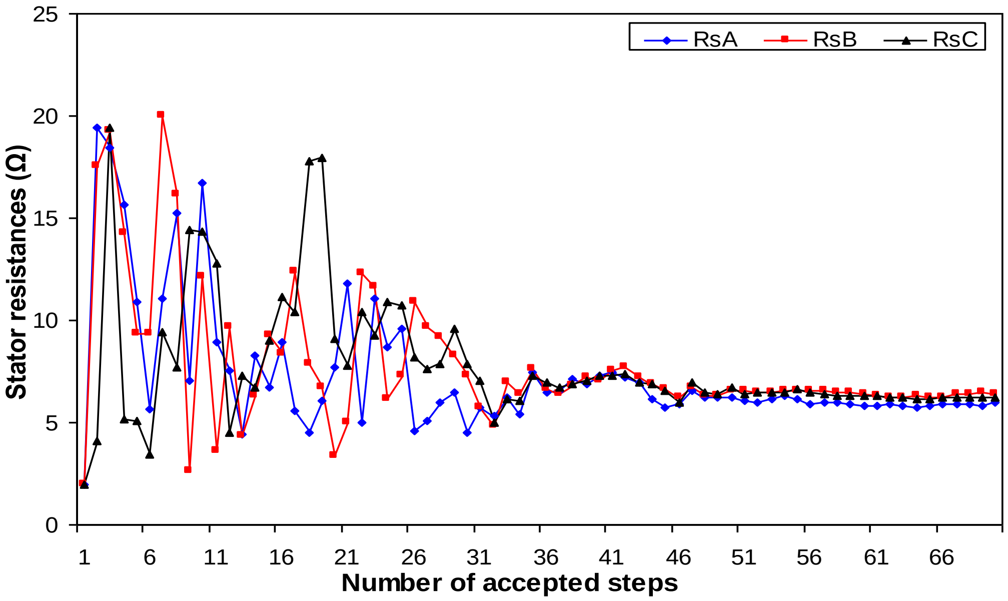


Figure 7.6. Estimated stator resistances obtained using SA for operation of induction motor with rotor open-circuit fault.

Around 50 investigations of potential solutions are required to obtain convergence. The values of the estimated stator and rotor resistances obtained at the end of the optimisation process are listed in Table 7.2. Figure 7.7 shows the value of the error corresponding to the number of steps when determining the best solution under the rotor open-circuit fault condition. The error falls from a maximum value of 0.854 A.s to 0.458 A.s.

Figure 7.8 compares the measured (I_A, I_B, I_C) and calculated (I_{sA}, I_{sB}, I_{sC}) stator currents using the final SA parameter values, and shows good agreement between the two current waveform sets.

Table 7.2. Final values of winding resistances obtained using SA with rotor open-circuit fault.

R_{sA} (Ω)	R_{sB} (Ω)	R_{sC} (Ω)	R_{ra} (Ω)	R_{rb} (Ω)	R_{rc} (Ω)
5.970	6.420	6.259	11.091	11.084	5.769

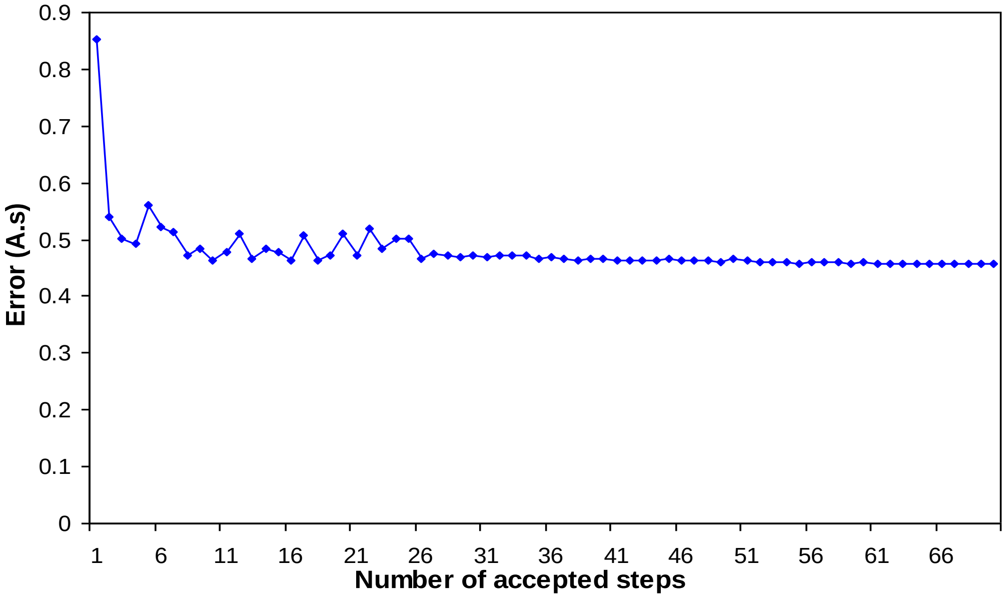


Figure 7.7. Estimated calculation error obtained using SA for operation of induction motor with rotor open-circuit fault.

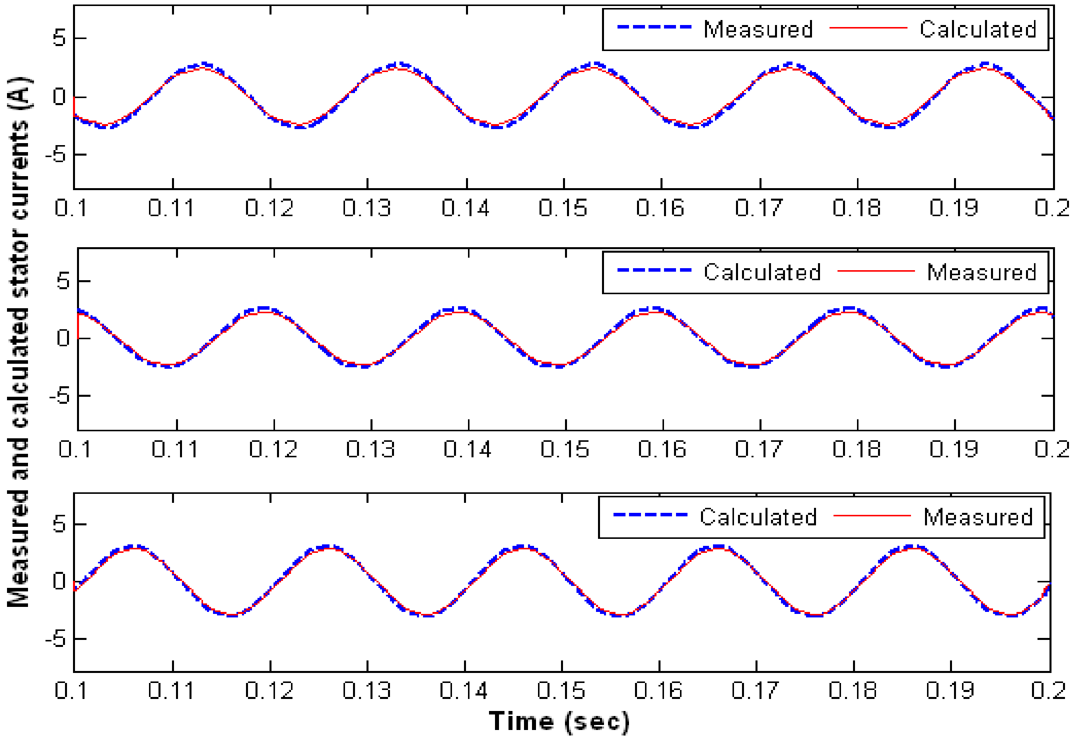


Figure 7.8. Measured (I_A, I_B, I_C) and calculated (I_{sA}, I_{sB}, I_{sC}) stator current waveforms using the estimated resistances obtained from SA for operation of induction motor under a rotor open-circuit fault.

7.3.2 Inverter-fed induction motor

7.3.2.1 Stator winding open-circuit fault

Using the same simulation test described previously in section 5.3.2, Figure 7.9 shows that convergence occurred after around 46 investigations of potential solutions when using the 40 Hz waveforms. There was an obvious increase in value of stator resistance to a steady level of 12.453 Ω for phase A, while the values of the other two resistances were approximately at their nominal values. Figure 7.10 shows that the rotor resistances also have nominal values indicating a healthy state for the rotor windings. The maximum IAE is 0.134 A.s while the final value of IAE is 0.016 A.s, see Figure 7.11.

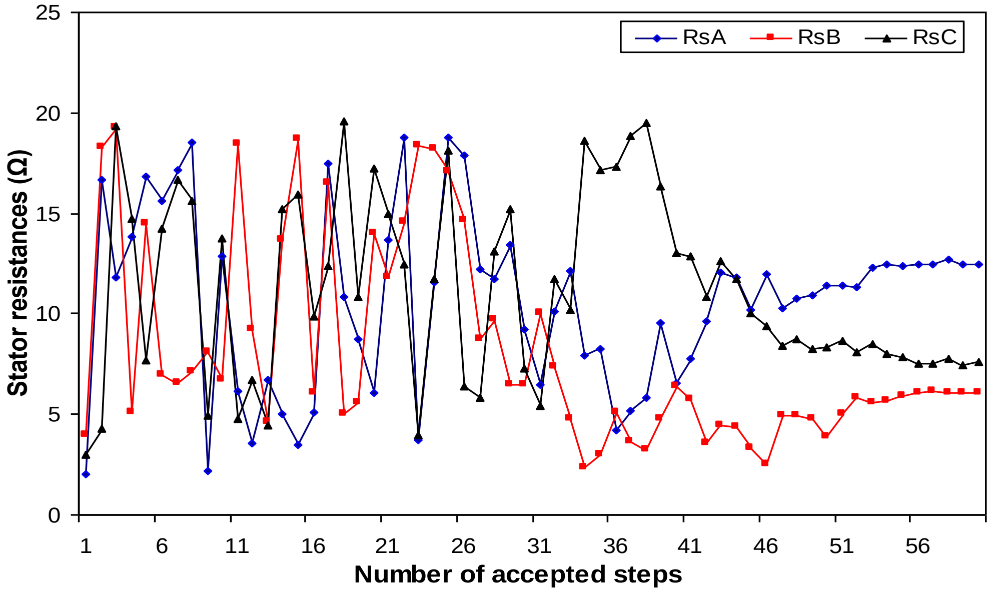


Figure 7.9. Estimated stator resistances obtained using SA for operation of inverter-fed induction motor with stator open-circuit fault at 40 Hz stator frequency.

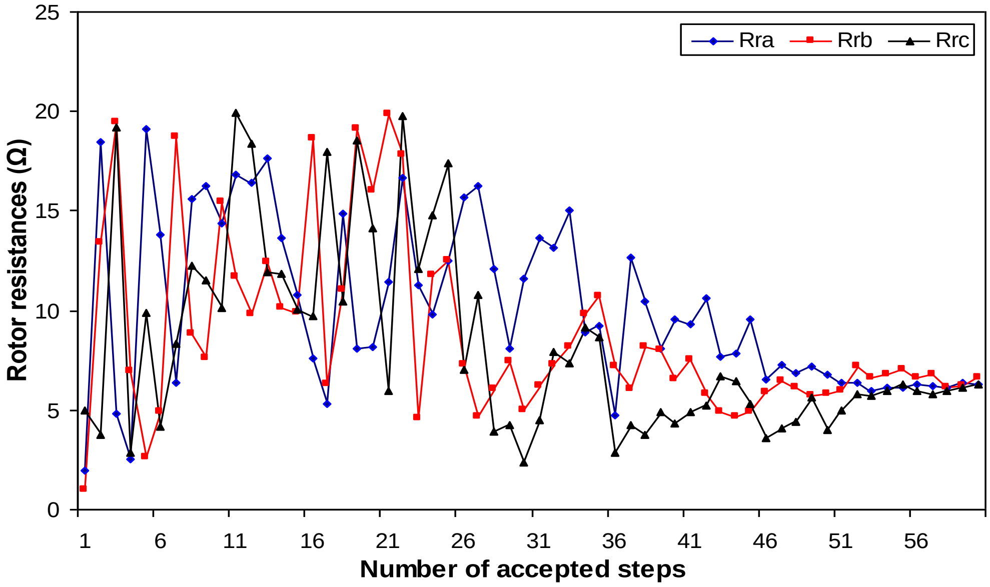


Figure 7.10. Estimated rotor resistances obtained using SA for operation of inverter-fed induction motor with stator open-circuit fault at 40 Hz stator frequency.

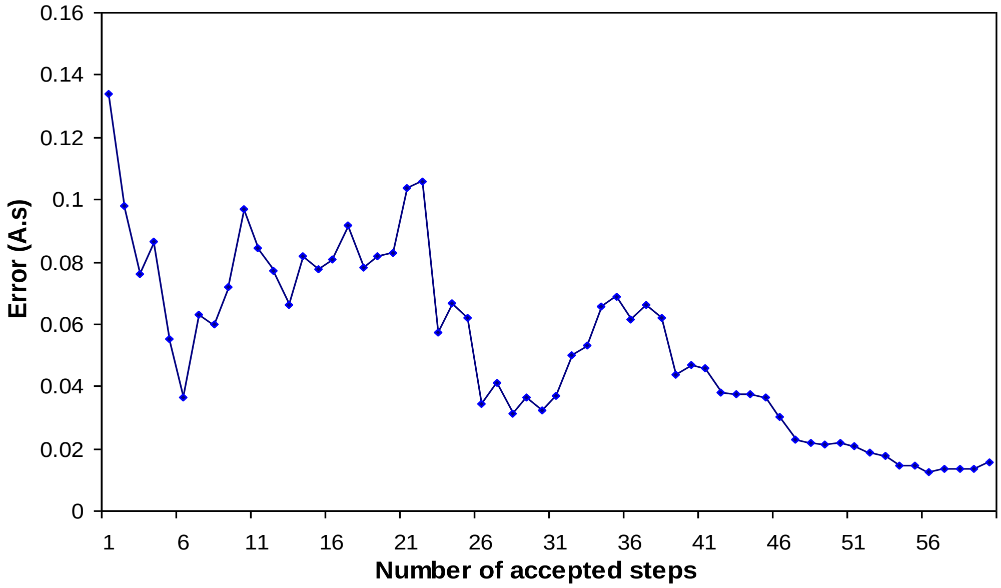


Figure 7.11. Estimated calculation error obtained using SA for operation of inverter-fed induction motor with stator open-circuit fault at 40 Hz stator frequency.

The final estimated values of the stator and rotor resistances are listed in Table 7.3. Figure 7.12 shows (I_A, I_B, I_C) and (I_{sA}, I_{sB}, I_{sC}), calculated using the final SA parameter values. The results indicate reasonable agreement between the two current waveforms.

Table 7.3. Final values of winding resistances obtained using SA with a stator open-circuit fault at 40 Hz stator frequency.

R_{sA} (Ω)	R_{sB} (Ω)	R_{sC} (Ω)	R_{ra} (Ω)	R_{rb} (Ω)	R_{rc} (Ω)
12.453	6.084	7.623	6.320	6.622	6.265

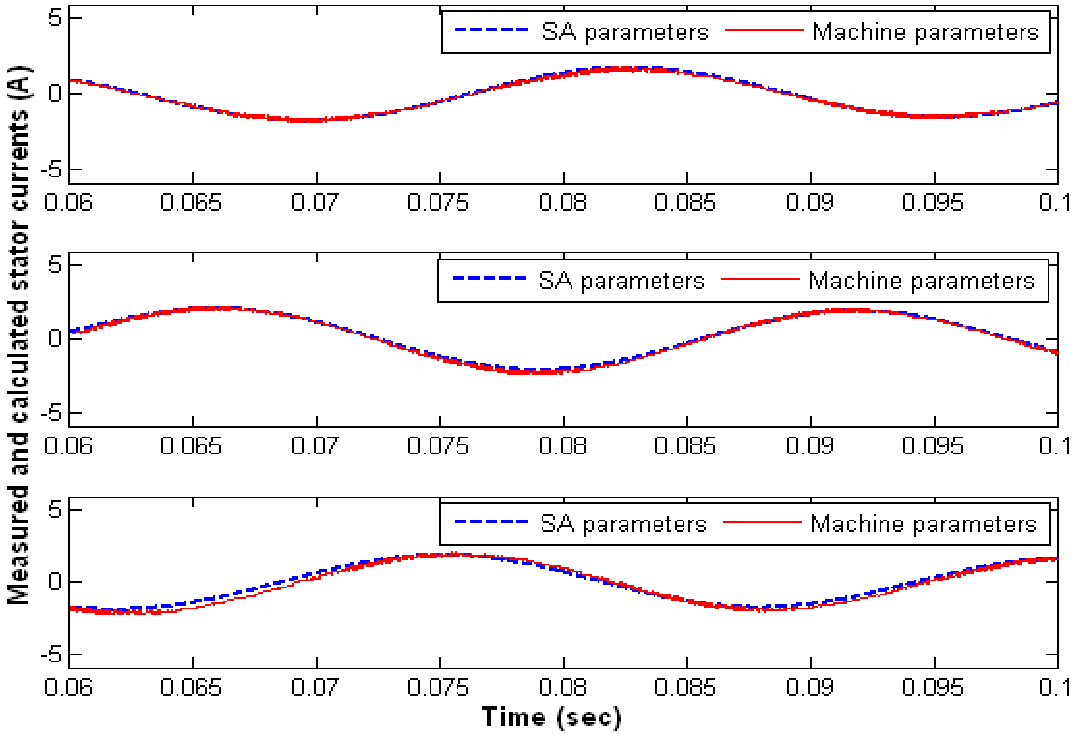


Figure 7.12. Stator current waveforms (I_A, I_B, I_C) and (I_{sA}, I_{sB}, I_{sC}) for inverter-fed induction motor with stator open-circuit fault; SA algorithm, 40 Hz stator frequency.

7.3.2.2 Rotor winding open-circuit fault

In this investigation, the SA algorithm fails to obtain convergence and is stuck after number of steps of potential solutions. These tests were repeated for different types of temperature schedules and different values of the initial temperature but there were no improvements in the results. Figure 7.13 and Figure 7.14 show the results when the stator was supplied at a frequency of 40 Hz. Although the error falls from a maximum value of 0.166 A.s to 0.014 A.s., the final values of the stator and rotor resistances estimated during the optimisation process were not correct (see Figure 7.15 and Table 7.4).

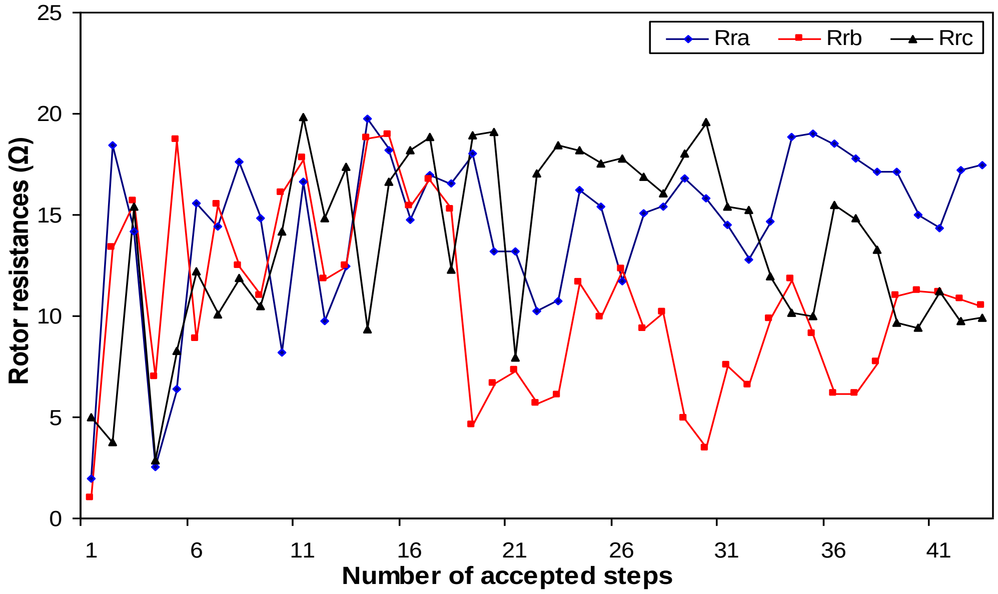


Figure 7.13. Estimated rotor resistances obtained using SA for operation of inverter-fed induction motor with rotor open-circuit fault at 40 Hz stator frequency.

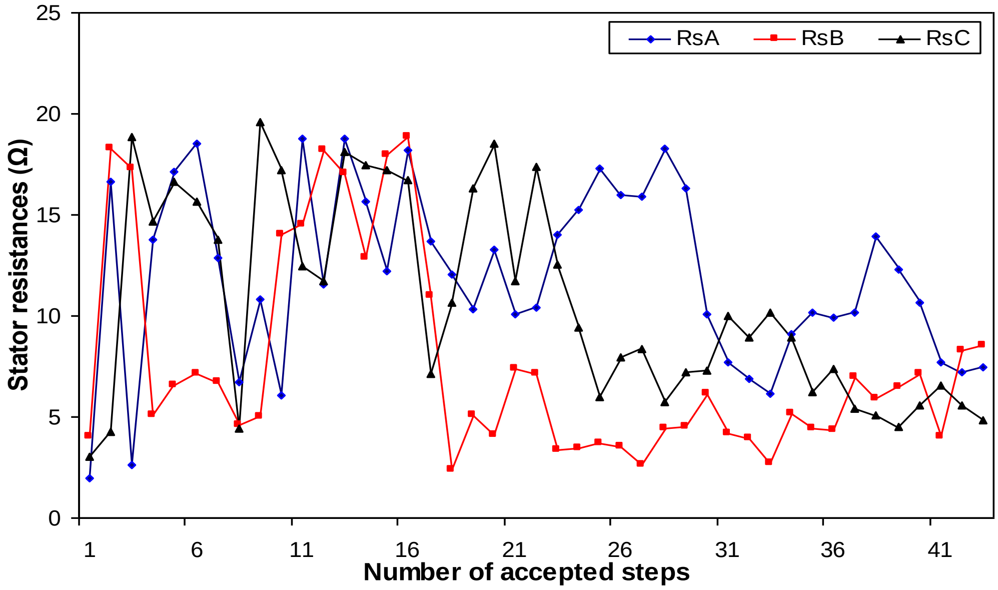


Figure 7.14. Estimated stator resistances obtained using SA for operation of inverter-fed induction motor with rotor open-circuit fault at 40 Hz stator frequency.

Table 7.4. Final values of winding resistances obtained using SA with rotor open-circuit fault at 40 Hz stator frequency.

R_{sA} (Ω)	R_{sB} (Ω)	R_{sC} (Ω)	R_{ra} (Ω)	R_{rb} (Ω)	R_{rc} (Ω)
7.425	8.494	4.834	17.456	10.499	9.951

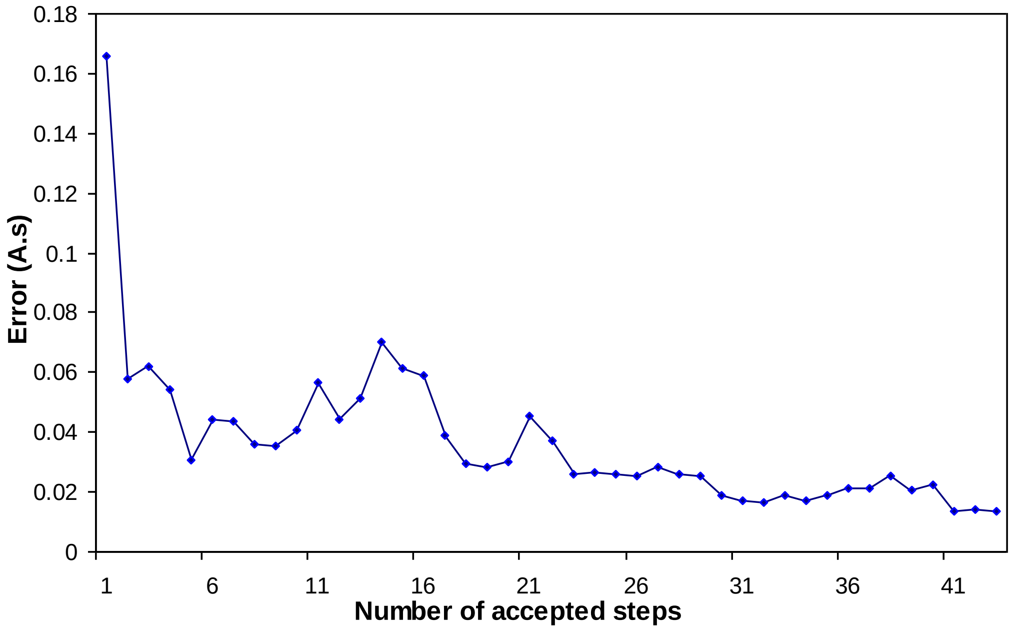


Figure 7.15. Estimated calculation error obtained using SA for operation of inverter-fed induction motor with rotor open-circuit fault at 40 Hz stator frequency.

7.4 Summary

SA, like other stochastic search methods, does not always converge to an optimum solution. The results presented in this chapter show that SA often failed when used to identify the presence of stator and rotor open-circuit faults. The SA probability of success was found to be about 60% for stator winding faults and 50% for rotor winding faults.

CHAPTER 8

THESIS CONCLUSION AND FUTURE WORK

8.1 Conclusion

This thesis has been concerned with an identification scheme for fault detection in induction motors using three different stochastic optimisation algorithms: genetic algorithm (GA), tabu search (TS) and simulated annealing (SA). The aim of this work is to demonstrate the proposed technique for parameter identification and stator and rotor winding fault detection using GA, TS and SA algorithms where the optimum solution is determined by the criterion of the integral of the absolute error and best performance is associated with the smallest value of IAE.

The data required for the proposed method are motor terminal voltages, stator currents and rotor speed obtained during steady state operation. Tests were carried out for a supply-fed induction machine. The data acquired are then processed off-line using the GA, TS and SA algorithms used in conjunction with an ABCabc induction motor model to determine the effective motor parameters. A series of experiments were conducted to collect the data needed to verify the technique proposed for parameter identification and fault detection. The experimental test rig used in this investigation consisted of a three-phase 240 V, 1.5 kW wound rotor induction machine coupled to a 3 kW DC machine used as a generator to provide the necessary load torque.

In this technique, the calculated stator currents obtained from the ABCabc induction motor model are compared with the actual measured stator currents to produce a set of current errors that are integrated and summed to give an overall calculation error that is then used as the cost function for the stochastic optimisation algorithm. Fault identification is then implemented by minimising the cost function in order to predict the machine condition. This is achieved by adjusting the parameters of the machine model using the stochastic algorithms until there is a close agreement between the measured and simulated data. The new set of the model parameters indicates if the machine winding is healthy or if there is a fault and also the location and the nature of this fault.

Firstly, these stochastic algorithms were used in conjunction with steady-state data obtained from healthy induction motor to determine the equivalent circuit parameters of this motor. Based on the results obtained, it can be concluded that GA and TS algorithms were successful in identifying the parameters of the induction motor while the SA algorithm failed to converge. Experiments for SA were repeated for different types of temperature schedules and different values of the initial temperature but there were no improvement in the results. It is noted that increasing the number of iterations did not have an effect on the final solution. TS is most successful in terms of accuracy and computational time achieving a success rate of about 80% while the GA algorithm achieves a success rate of about 70%.

Secondly, open-circuit stator winding fault was tested. To emulate this winding fault, an external 7 Ω resistance was connected in series with a phase winding. In this investigation, all the three stochastic algorithms were able to identify the presence and location of an open-circuit winding fault in the stator of the induction motor, when fed directly from the three phase supply or via a voltage-fed variable-frequency, variable-voltage converter. The probability of success for the GA, TS and SA algorithms were found to be about 70%, 80% and 60% respectively. TS and GA have faster computational time to converge while SA required a relatively long computational time because of its' one agent search feature.

Thirdly, a rotor open-circuit winding fault which was emulated by connecting a 7 Ω resistor in series with the line connected to the two ends of the a-b rotor delta; the stochastic algorithms were able to identify the presence and location of an open-circuit winding fault in the rotor of the induction motor. TS had a success rate of about 80% with less computational time. The probability of success for the GA and SA algorithms were found to be about 60% and 50%, respectively.

Finally, it can be concluded that the proposed scheme is able to confirm that the three-phase induction motor windings are healthy or indicate that there is a winding fault and can identify the type and location of fault without the need for knowledge of various fault signatures. The three stochastic algorithms used in this investigation are very simple to implement using the Matlab/Simulink environment.

8.2 Future work

To extend the work presented in this thesis to cover other machine faults such as short-circuit winding faults and mechanical faults. Machine inductances and other model parameters should be included in this optimisation process.

The identification scheme described in this thesis is basically an off-line technique. It needs to be devolved and possibly combined with other fault detection schemes to be used for on-line condition monitoring.

REFERENCES

[1] R. Sharifi and M. Ebrahimi, "Detection of stator winding faults in induction motors using three-phase current monitoring," *ISA Transactions,* vol. 50, pp. 14-20.

[2] G. H. Muller and C. F. Landy, "A novel method to detect broken rotor bars in squirrel cage induction motors when interbar currents are present," in *Power Engineering Society General Meeting, 2003, IEEE,* 2003.

[3] M. J. Castelli, J. P. Fossati, and M. T. Andrade, "New methodology to faults detection in induction motors via MCSA," in *Transmission and Distribution Conference and Exposition: Latin America, 2008 IEEE/PES*, 2008, pp. 1-6.

[4] Y. B. Ivonne, D. Sun, and Y. K. He, "Fault diagnosis using Neural-Fuzzy technique based on the simulation results of stator faults for a three-phase induction motor drive system," in *ICEMS 2005: Proceedings of the Eighth International Conference on Electrical Machines and Systems*, 2005, pp. 1966-1971.

[5] L. Tong and H. Jin, "A novel method for induction motors stator interturn short circuit fault diagnosis by wavelet packet analysis," in *Electrical Machines and Systems, 2005. ICEMS 2005. Proceedings of the Eighth International Conference on*, 2005, pp. 2254-2258 Vol. 3.

[6] S. Nandi, H. A. Toliyat, and L. Xiaodong, "Condition monitoring and fault diagnosis of electrical motors-a review," *Energy Conversion, IEEE Transactions on,* vol. 20, pp. 719-729, 2005.

[7] M. Wolkiewicz, C. T. Kowalski, and M. KamiÅ„ski, "Detection of inter-turn short circuits of induction motor with application of symmetrical components and neural networks," *Wykrywanie zwarÄ‡ zwojowych silnika indukcyjnego z wykorzystaniem skÅ‚adowych symetrycznych oraz sieci neuronowych,* vol. 88, pp. 276-281.

[8] N. Mariun, M. R. Mehrjou, M. H. Marhaban, and N. Misron, "An experimental study of induction motor current signature analysis techniques for incipient broken rotor bar detection," in *International Conference on Power Engineering, Energy and Electrical Drives*.

[9] V. Rashtchi, E. Rahimpour, and S. Fazli, "Genetic algorithm application to detect broken rotor bar in three phase squirrel cage induction motors," *International Review of Electrical Engineering,* vol. 6, pp. 2286-2292.

[10] T. Kato, K. Inoue, and K. Yoshida, "Diagnosis of stator-winding-turn faults of induction motor by direct detection of negative-sequence currents," *IEEJ Transactions on Industry Applications,* vol. 131, pp. 1346-1353+12.

[11] G. R. Bossio, C. H. De Angelo, P. M. De La Barrera, J. A. Solsona, G. O. GarcÃ-a, and M. I. Valla, "Stator winding fault detection in induction motor drives using signal injection," in *SDEMPED 2011 - 8th IEEE Symposium on Diagnostics for Electrical Machines, Power Electronics and Drives*, pp. 92-97.

[12] I. B. A. Bazine, S. Tnani, T. Poinot, G. Champenois, and K. Jelassi, "On-line detection of stator and rotor faults occurring in induction machine diagnosis by parameters estimation," in *SDEMPED 2011 - 8th IEEE Symposium on Diagnostics for Electrical Machines, Power Electronics and Drives*, pp. 105-112.

[13] M. Davudi, S. Torabzad, and B. Ojaghi, "Application of Adaptive Neuro Fuzzy Inference Systems for induction motor fault detection and diagnosis," *Australian Journal of Basic and Applied Sciences,* vol. 5, pp. 961-971.

[14] R. P. S. Ventura, A. M. S. Mendes, and A. J. M. Cardoso, "Fault detection in multilevel cascaded inverter using Park's Vector approach with balanced battery power usage," in *Proceedings of the 2011 14th European Conference on Power Electronics and Applications, EPE 2011*.

[15] J. Wang, R. X. Gao, and R. Yan, "Broken-rotor-bar diagnosis for induction motors," *Journal of Physics: Conference Series,* vol. 305.

[16] K. P. Zakaria, P. P. Acarnley, and B. Zahawi, "Condition monitoring of an induction machine using a stochastic search technique," in *Power Electronics, Machines and Drives, 2006. PEMD 2006. The 3rd IET International Conference on*, 2006, pp. 42-46.

[17] P. J. T. a. J. Penman, *Condition Monitoring and Electrical Machines*. New York, Wiley: Research Studies Press Ltd, 1987.

[18] V. N. Ghate and S. V. Dudul, "Optimal MLP neural network classifier for fault detection of three phase induction motor," *Expert Systems with Applications,* vol. 37, pp. 3468-3481.

[19] M. E. H. Benbouzid, "A review of induction motors signature analysis as a medium for faults detection," *IEEE Transactions on Industrial Electronics,* vol. 47, pp. 984-993, 2000.

[20] A. Siddique, G. S. Yadava, and B. Singh, "A review of stator fault monitoring techniques of induction motors," *IEEE Transactions on Energy Conversion,* vol. 20, pp. 106-114, 2005.

[21] M. E. H. Benbouzid, "Review of induction motors signature analysis as a medium for faults detection," in *IECON Proceedings (Industrial Electronics Conference)*, 1998, pp. 1950-1955.

[22] F. Filippetti, G. Franceschini, C. Tassoni, and P. Vas, "Recent developments of induction motor drives fault diagnosis using AI techniques," *IEEE Transactions on Industrial Electronics,* vol. 47, pp. 994-1004, 2000.

[23] S. A. Ethny, P. P. Acarnley, B. Zahawi, and D. Giaouris, "Induction machine fault identification using particle swarm algorithms," in *2006 International Conference on Power Electronics, Drives and Energy Systems, PEDES '06*, 2006.

[24] M. Iorgulescu, R. Beloiu, and M. O. Popescu, "Vibration monitoring for diagnosis of electrical equipment's faults," in *Optimization of Electrical and Electronic Equipment (OPTIM), 2010 12th International Conference on*, pp. 493-499.

[25] G. G. Acosta, C. J. Verucchi, and E. R. Gelso, "A current monitoring system for diagnosing electrical failures in induction motors," *Mechanical Systems and Signal Processing,* vol. 20, pp. 953-965, 2006.

[26] N. Mehla and R. Dahiya, "An Approach of Condition Monitoring of Induction Motor Using MCSA," *International Journal of Systems applications, Engineering & Development,* vol. 1, 2007.

[27] H. Tian, Y. Bo-Suk, and L. Jong Moon, "A new condition monitoring and fault diagnosis system of induction motors using artificial intelligence algorithms," in *Electric Machines and Drives, 2005 IEEE International Conference on*, 2005, pp. 1967-1974.

[28] A. M. Trzynadlowski and E. Ritchie, "Comparative investigation of diagnostic media for induction motors: a case of rotor cage faults," *IEEE Transactions on Industrial Electronics*, vol. 47, pp. 1092-1099, 2000.

[29] S. Nandi, H. A. Toliyat, and X. Li, "Condition monitoring and fault diagnosis of electrical motors - A review," *IEEE Transactions on Energy Conversion,* vol. 20, pp. 719-729, 2005.

[30] F. C. Trutt, J. Sottile, and J. L. Kohler, "Condition monitoring of induction motor stator windings using electrically excited vibrations," in *Conference Record - IAS Annual Meeting (IEEE Industry Applications Society),* 2002, pp. 2301-2305.

[31] M. S. N. Said, M. E. H. Benbouzid, and A. Benchaib, "Detection of broken bars in induction motors using an extended Kalman filter for rotor resistance sensorless estimation," *Energy Conversion, IEEE Transactions on,* vol. 15, pp. 66-70, 2000.

[32] I. Bazine, S. Bazine, S. Tnani, and G. Champenois, "On-line broken bars detection diagnosis by parameters estimation," in *Power Electronics and Applications, 2009. EPE '09. 13th European Conference on*, 2009, pp. 1-7.

[33] P. V. Goode and C. Mo-Yuen, "Neural/fuzzy systems for incipient fault detection in induction motors," in *Industrial Electronics, Control, and Instrumentation, 1993. Proceedings of the IECON '93., International Conference on*, 1993, pp. 332-337 vol.1.

[34] M. Seera, L. Chee Peng, D. Ishak, and H. Singh, "Fault Detection and Diagnosis of Induction Motors Using Motor Current Signature Analysis and a Hybrid FMM&#x2013;CART Model," *Neural Networks and Learning Systems, IEEE Transactions on,* vol. 23, pp. 97-108.

[35] R. M. Tallam, T. G. Habetler, R. G. Harley, D. J. Gritter, and B. H. Burton, "Neural network based on-line stator winding turn fault detection for induction motors," in *Industry Applications Conference, 2000. Conference Record of the 2000 IEEE,* 2000, pp. 375-380 vol.1.

[36] M. L. S. Sin, Wen Liang Ertugrul, Nesimi, "Induction machine on-line condition monitoring and fault diagnosis - a survey," 2003.

[37] E. Serna and J. M. Pacas, "Detection of rotor faults in field oriented controlled induction machines," in *Conference Record - IAS Annual Meeting (IEEE Industry Applications Society),* 2006, pp. 2326-2332.

[38] K. R. Cho, J. H. Lang, and S. D. Umans, "Detection of broken rotor bars in induction motors using state and parameter estimation," *IEEE Transactions on Industry Applications,* vol. 28, pp. 702-709, 1992.

[39] A. H. Bonnett and G. C. Soukup, "ROTOR FAILURES IN SQUIRREL CAGE INDUCTION MOTORS," in *Conference Record - IAS Annual Meeting (IEEE Industry Applications Society),* 1986, pp. 1735-1744.

[40] R. Supangat, N. Ertugrul, W. L. Soong, D. A. Gray, C. Hansen, and J. Grieger, "Broken rotor bar fault detection in induction motors using starting current analysis," in *2005 European Conference on Power Electronics and Applications*, 2005.

[41] J. Cibulka, M. K. Ebbesen, and K. G. Robbersmyr, "Bearing fault detection

in induction motor-gearbox drivetrain," *Journal of Physics: Conference Series*, vol. 364.

[42] L. Wang, H. Wang, Y. Cai, and S. Wang, "Fault diagnosis system of rolling bearing based on wavelet analysis," in *Applied Mechanics and Materials*. vol. 166-169, pp. 951-955.

[43] J. T. Cheng, L. Ai, and W. Xiong, "Rolling bearing fault diagnosis based on the hybrid algorithm of particle swarm optimization with neighborhood operator," in *Proceedings - 2012 International Conference on Computer Science and Electronics Engineering, ICCSEE 2012*, pp. 24-26.

[44] P. Vas, *Electrical Machines and Drive: A space-vector theory approach*. New York, USA: Oxford University Press, 1992.

[45] D. Hyun, S. Lee, J. Hong, S. Bin Lee, and S. Nandi, "Detection of Airgap Eccentricity for Induction Motors Using the Single-Phase Rotation Test," *IEEE Transactions on Energy Conversion*.

[46] B. Trajin, J. Regnier, and J. Faucher, "Indicator for bearing fault detection in asynchronous motors using stator current spectral analysis," in *Industrial Electronics, 2008. ISIE 2008. IEEE International Symposium on*, 2008, pp. 570-575.

[47] J. R. Cameron, W. T. Thomson, and A. B. Dow, "Vibration and current monitoring for detecting airgap eccentricity in large induction motors," *IEE Proceedings B: Electric Power Applications,* vol. 133, pp. 155-163, 1986.

[48] R. R. Schoen, T. G. Habetler, F. Kamran, and R. G. Bartheld, "Motor bearing damage detection using stator current monitoring," *IEEE Transactions on Industry Applications,* vol. 31, pp. 1274-1279, 1995.

[49] B. K. N. Rao, P. Srinivasa Pai, and T. N. Nagabhushana, "Failure diagnosis and prognosis of rolling - Element bearings using artificial neural networks: A critical overview," *Journal of Physics: Conference Series,* vol. 364.

[50] M. K. Rad, M. Torabizadeh, and A. Noshadi, "Artificial neural network-based fault diagnostics of an electric motor using vibration monitoring," in *Proceedings 2011 International Conference on Transportation, Mechanical, and Electrical Engineering, TMEE 2011*, pp. 1512-1516.

[51] V. P. Mini, S. Sivakotaiah, and S. Ushakumari, "Fault detection and diagnosis of an induction motor using fuzzy logic," in *Proceedings - 2010 IEEE Region 8 International Conference on Computational Technologies in Electrical and Electronics Engineering, SIBIRCON-2010*, pp. 459-464.

[52] S. E. Zouzou, W. Laala, S. Guedidi, and M. Sahraoui, "A fuzzy logic approach for the diagnosis of rotor faults in squirrel cage induction motors," in *2009 International Conference on Computer and Electrical Engineering, ICCEE 2009*, 2009, pp. 173-177.

[53] S. M. A. Cruz and A. J. M. Cardoso, "Rotor cage fault diagnosis in three-phase induction motors by extended Park's Vector approach," *Electric Power Components and Systems*, vol. 28, pp. 289-299, 2000.

[54] A. J. Marques Cardoso, S. M. A. Cruz, J. F. S. Carvalho, and E. S. Saraiva, "Rotor cage fault diagnosis in three-phase induction motors, by Park's vector approach," in *Conference Record - IAS Annual Meeting (IEEE Industry Applications Society)*, 1995, pp. 642-646.

[55] A. J. Marques Cardoso, E. S. Saraiva, M. L. Sousa Mateus, and A. L. Ramalho, "On-line detection of airgap eccentricity in 3-phase induction motors, by Park's Vector approach," in *Electrical Machines and Drives,*

1991. Fifth International Conference on (Conf. Publ. No. 341), 1991, pp. 61-66.

[56] J. L. H. Silva and A. J. M. Cardoso, "Bearing failures diagnosis in three-phase induction motors by extended Park's vector approach," in *Industrial Electronics Society, 2005. IECON 2005. 31st Annual Conference of IEEE,* 2005, p. 6 pp.

[57] A. Kangas, J. Kangas, and M. Kurttila, *Decision Support for Forest Management* vol. 16: Springer 2008.

[58] S. A. Ethni, B. Zahawi, D. Giaouris, and P. P. Acarnley, "Comparison of particle swarm and simulated annealing algorithms for induction motor fault identification," in *IEEE International Conference on Industrial Informatics (INDIN),* 2009, pp. 470-474.

[59] F. A. T. MontanÃ© and R. D. GalvÃ£o, "A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service," *Computers and Operations Research,* vol. 33, pp. 595-619, 2006.

[60] L. Cai, Y. Zhang, Z. Zhang, C. Liu, and Z. Lu, "Application of genetic algorithms in EKF for speed estimation of an induction motor," in *PESC Record - IEEE Annual Power Electronics Specialists Conference,* 2003, pp. 345-349.

[61] F. G. a. M. Laguna, *Tabu Search*. Norwell , Massachusetts: Kluwer Academic Publishers, 1997.

[62] E. Aarts and J. Korst, *Simulated Annealing and Boltzmann Machines*. Essex , U.K: John Wiley & Sons Ltd, 1989.

[63] T. Weise, *Global Optimization Algorithms - Theory and Application*: Thomas Weise, 2008.

[64] D. E. Goldberg and J. H. Holland, *Genetic Algorithms in Search, Optimisation, and Machine learing* vol. 3: Addison - Wesley, 1989.

[65] M. Mitchell, *An Introduction to Genetic Algorithms*: The MIT Press, 1996.

[66] J. J. Grefenstette, "Incorporating problem specific knowledge into genetic algorithms " in *Genetic Algorithms and Simulated Annealing*: Morgan Kauffmann, 1987.

[67] Y. Yuehong, Y. Jianfeng, and C. Zhaoneng, "A genetic algorithm based approach to flowshop scheduling," in *Intelligent Control and Automation, 2004. WCICA 2004. Fifth World Congress on*, 2004, pp. 3019-3021 Vol.4.

[68] L. D. Davis, *Handbook of Genetic Algorithms*. New York , USA: Van Nostrand Reinhold, 1991.

[69] E. K. Burke and G. Kendall, *Search Methodologies*. Nottingham . U.K, 2005.

[70] D. Goldberg, K. Deb, and D. Thierens, "Toward a better understanding of mixing in genetic algorithms," *Journal of Society of Instrument and Control Engineers,* vol. 32, No.1, pp. 10-16, 1993.

[71] Y. H. Song and M. R. Irving, "Optimisation techniques for electrical power systems Part 2 Heuristic optimisation methods," *Power Engineering Journal,* vol. 15, pp. 151-160, 2001.

[72] S. Papadamou and G. Stephanides, "Improving technical trading systems by using a new MATLAB-based genetic algorithm procedure," *Mathematical and Computer Modelling,* vol. 46, pp. 189-197, 2007.

[73] D. A. Coley, *An introduction to genetic algorithms for scientists and engineers*. Singapore: World Scientific Publishing Co. Pte. Ltd, 1999.

[74] F. Glover, "Future paths for integer programming and links to artificial

intelligence," *Computers and Operations Research,* vol. 13, pp. 533-549, 1986.

[75] E. Costamagna, A. Fanni, and G. Giacinto, "A tabu search algorithm for the optimisation of telecommunication networks," *European Journal of Operational Research*, vol. 106, pp. 357-372, 1998.

[76] F. Wen and C. S. Chang, "Transmission network optimal planning using the tabu search method," *Electric Power Systems Research,* vol. 42, pp. 153-163, 1997.

[77] C. A. Anderson, K. Fraughnaugh, M. Parker, and J. Ryan, "Path assignment for call routing: An application of tabu search," *Annals of Operations Research*, vol. 41, pp. 299-312, 1993.

[78] S. Kirkpatrick, C. D. Gelatt Jr, and M. P. Vecchi, "Optimization by simulated annealing," *Science,* vol. 220, pp. 671-680, 1983.

[79] X. Zhu, Y. Huang, and J. Doyle, "Genetic algorithms and simulated annealing for robustness analysis," in *Proceedings of the American Control Conference*, 1997, pp. 3756-3760.

[80] T. El-Ghazali, *Metaheuristics from design to implementation* New Jersey , U.S.A: John Wiley & Sons, Inc. , Publication, 2009.

[81] Y. G. Saab and V. B. Rao, "Combinatorial optimization by stochastic evolution," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 10, pp. 525-535, 1991.

[82] M. Sen and P. L. Stoffa, *Global Optimisation Methods in Geophysical Inversion*. Texas , USA: Elsevier Science B. V., 1995.

[83] X. Yao, "A New Simulated Annealing Algorithm," *Inter. J. Computer Math,* vol. 56, pp. 161-168, 1995.

[84] P. Vas, *Parameter Estimation, Condition Monitoring, and Diagnosis of Electrical Machines*. Oxford: Oxford University Press, 1993.

[85] Mawdsley, *Operating & Maintenance Instructions For The Students Demonstration Set*. Gloucestershire, England: Mawdsley's LTD, 1964.

[86] I. Standards, *IEEE Standard Test Procedure for polyphase Induction Motors and Generators*. New York, USA, 2004.

APPENDIX

CODE FOR GA, TS AND SA ALGORITHMS

A1- M.file code for GA algorithm (simple example)

```
% This program is for simple example (one variable)
clc
clf
clear all
NIND = 10;                         % Number of individuals
MAXGEN = 50;                      % Maximum no. of generations
NVAR =1;                           % No. of Variables
PRECI = 40;                         % Precision of variables
GGAP = 0.9;                        % Generation gap
% Build Fielf Descriptor
FieldD = [rep([PRECI],[1,NVAR]);rep([-
5;5],[1,NVAR]);rep([1;0;1;1],[1,NVAR])];
% Initialise population
Chrom = crtbp(NIND , NVAR*PRECI);
gen=0;                 % counter
x=bs2rv(Chrom,FieldD);
          for i=1:NIND


Chr=x(i,:);
a=Chr(1,1);

% Evaluate initial population
%ObjV(i) = objfun16(a(i));

%objfun12(a);

function ObjVal=objfun12(a); % if is one parameter (a)

ObjVal=0.0116*a^4-0.2473*a^3+1.7927*a^2- 4.9426*a+5.4150;

ObjV(i) = objfun12(a);
          end
ObjV=ObjV';
% Generational loop

  while gen < MAXGEN

% Assign fitness values to entire population
FitnV = RANKING(ObjV);
```

```
% Select individuals for breeding
SelCh = select('sus',Chrom,FitnV,GGAP);

% Recombine individuals (crossover)
SelCh = recombin('xovsp',SelCh,0.7);

% Apply mutation
SelCh = mut(SelCh);

% Evaluate offspring, call objective function
w=bs2rv(SelCh,FieldD);
for i=1:(NIND-1)
Chr=w(i,:);
a=Chr(1,1);

%ObjVSel(i) = objfun16(a(i));
ObjVSel(i) = objfun12(a);
end


% Reinsert offspring into population
[Chrom ObjV]= reins(Chrom,SelCh,1,1,ObjV,ObjVSel');
y=bs2rv(Chrom,FieldD);
 minV= min(ObjV);
for p=1:NIND
    if minV==ObjV(p)
        ind=p;
    end
end

figure(1)
k=min(ObjV);
k1=y(ind,1);
subplot(2,1,1),plot(gen,k1,'*'),hold on
subplot(2,1,2),plot(gen,k,'*'),hold on
% Increment counter
gen = gen+1;

end
 toc
          k1(end)

          k(end)
```

A2- M.file code for GA algorithm (fault identification)

```
%This test is for a healthy machine with six parameters
clear all
clc
clf
tic
NIND =12;                         % Number of individuals
MAXGEN =151;                      % Maximum no. of generations
NVAR =6;                          % No. of Variables
PRECI = 20;                       % Precision of variables
GGAP =0.9;                        % Generation gap
%-------------------------------------------------------------------
% Build Fiele Descriptor

FieldD =
[rep([PRECI],[1,NVAR]);rep([2;30],[1,NVAR]);rep([1;0;1;1],[1,NVAR])];
FieldDR=[2 2 2 2 2 2;30 30 30 30 30 30];
%-------------------------------------------------------------------
% Initialise population

Chrom = crtbp(NIND,NVAR*PRECI);
Gen=1               % counter
RsAm=1;RsBm=5;RsCm=RsC(n);Rram=3;Rrbm=2;Rrcm=7;
%-------------------------------------------------------------------
% Other machine parameters

LsAm=0.896;LsBm=LsAm;LsCm=LsAm;
%------------------------------
MssABm=0.185;MssACm=MssABm;MssCBm=MssABm;MssBAm=MssABm;MssBCm=MssABm
;MssCAm=MssABm;
%------------------------------
Mrrabm=0.771;Mrracm=Mrrabm;Mrrbam=Mrrabm;Mrrbam=Mrrabm;Mrrbam=Mrrabm
;Mrrbcm=Mrrabm;Mrrcam=Mrrabm;Mrrcbm=Mrrabm;
%------------------------------
MsrBam=0.751;MsrAam=MsrBam;MsrAbm=MsrAam;MsrAcm=MsrAam;MsrBam=MsrAam
;MsrBbm=MsrAam;MsrBcm=MsrAam;MsrBam=MsrAam;MsrBbm=MsrAam;MsrBcm=MsrA
am;MsrCam=MsrAam;MsrCcm=MsrAam;MsrCbm=MsrAam;
%------------------------------
MrsaAm=0.751;MrsaBm=MrsaAm;MrsaCm=MrsaAm;MrsaBm=MrsaAm;MrsaCm=MrsaAm
;MrsbAm=MrsaAm;MrsbBm=MrsaAm;MrsbCm=MrsaAm;MrscAm=MrsaAm;MrscBm=Mrsa
Am;MrscCm=MrsaAm;
%-----------------------------
Lrcm=0.507;Lram=Lrcm;Lrbm=Lrcm;
%-------------------------------------------------------------------
%Experimental data, armature voltages

V1=csvread('vp1.csv',2,0,[2,0,999,1]);TimeVSA=V1(:,1);min(TimeVSA);T
imeVSA=TimeVSA-min(TimeVSA);VSA=V1(:,2);

V2=csvread('vp2.csv',2,0,[2,0,999,1]);TimeVSB=V2(:,1);min(TimeVSB);T
imeVSB=TimeVSB-min(TimeVSB);VSB=V2(:,2);
V3=csvread('vp3.csv',2,0,[2,0,999,1]);TimeVSC=V3(:,1);min(TimeVSC);T
imeVSC=TimeVSC-min(TimeVSC);VSC=V3(:,2);
%-------------------------------------------------------------------
%Experimental data, armature currents
```

```
IA=csvread('p1.csv',2,0,[2,0,999,1]);TimeISA=IA(:,1);min(TimeISA);Ti
meISA=TimeISA-min(TimeISA);ISA=IA(:,2);
IB=csvread('p2.csv',2,0,[2,0,999,1]);TimeISB=IB(:,1);min(TimeISB);Ti
meISB=TimeISB-min(TimeISB);ISB=IB(:,2);
IC=csvread('p3.csv',2,0,[2,0,999,1]);TimeISC=IC(:,1);min(TimeISC);Ti
meISC=TimeISC-min(TimeISC);ISC=IC(:,2);
%-------------------------------------------------------------------
% Rotor speed
 wr=301.5
%-------------------------------------------------------------------
sim('Healthy_Machine');
k1=RsAm;k2=RsBm;k3=RsCm;k4=Rram;k5=Rrbm;k6=Rrcm;k7=E;
figure(1)
plot(Gen,k1,'bx',Gen,k2,'rh',Gen,k3,'go'),ylabel('RsA,RsB,RsC');hold
on
figure(2)
plot(Gen,k4,'bx',Gen,k5,'rh',Gen,k6,'go'),ylabel('Rra,Rrb,Rrc');hold
on
figure(3)
plot(Gen,k7,'bx'),ylabel('Error');hold on

mem(Gen,1:7)=[k1,k2,k3,k4,k5,k6,k7];
Gen=2              % counter
x=bs2rv(Chrom,FieldD);
for i=1:NIND
Chr=x(i,:);
RsAm=Chr(1,1);RsBm=Chr(1,2);RsCm=Chr(1,3);Rram=Chr(1,4);Rrbm=Chr(1,5
);Rrcm=Chr(1,6);
sim('Healthy_Machine');
    ObjV(i)=E;
          end
ObjV=ObjV';
minV= min(ObjV);
for p=1:NIND
    if minV==ObjV(p)
        ind=p;
    end
end
k1=x(ind,1);k2=x(ind,2);k3=x(ind,3);k4=x(ind,1);k5=x(ind,2);k6=x(ind
,3);k7=min(ObjV);
figure(1)
plot(Gen,k1,'bx',Gen,k2,'rh',Gen,k3,'go'),ylabel('RsA,RsB,RsC');hold
on
figure(2)
plot(Gen,k4,'bx',Gen,k5,'rh',Gen,k6,'go'),ylabel('Rra,Rrb,Rrc');hold
on
figure(3)
plot(Gen,k7,'bx'),ylabel('Error');hold on
mem(Gen,1:7)=[k1,k2,k3,k4,k5,k6,k7];
Gen=3
% Generational loop

 while Gen < MAXGEN
%-------------------------------------------------------------------
%Assign fitness values to entire population

FitnV = RANKING(ObjV);
%-------------------------------------------------------------------
% Select individuals for breeding
```

```
SelCh = select('sus',x,FitnV,GGAP);
%-------------------------------------------------------------------
% Recombine individuals (crossover)

SelCh = recombin('xovsp',SelCh,0.7);
%-------------------------------------------------------------------
% Apply mutation

SelCh = mutbga(SelCh,FieldDR);
%-------------------------------------------------------------------
%Evaluate offspring, call objective function

for i=1:(NIND-1)
Chr=SelCh(i,:);
RsAm=Chr(1,1);RsBm=Chr(1,2);RsCm=Chr(1,3);
sim('Healthy_Machine');
ObjVSel(i) = E;
end
%-------------------------------------------------------------------
% Reinsert offspring into population
[x ObjV]= reins(x,SelCh,1,1,ObjV,ObjVSel');
 minV= min(ObjV);
for p=1:NIND
    if minV==ObjV(p)
        ind=p;
    end
end
k1=x(ind,1);k2=x(ind,2);k3=x(ind,3);k4=x(ind,4);k5=x(ind,5);k6=x(ind
,6);k7=min(ObjV);
figure(1)
plot(Gen,k1,'bx',Gen,k2,'rh',Gen,k3,'go'),ylabel('RsA,RsB,RsC');hold
on
figure(2)
plot(Gen,k4,'bx',Gen,k5,'rh',Gen,k6,'go'),ylabel('Rra,Rrb,Rrc');hold
on
figure(3)
plot(Gen,k7,'bx'),ylabel('Error');hold on
mem(Gen,1:7)=[k1,k2,k3,k4,k5,k6,k7];
%-------------------------------------------------------------------
%Increment counter

Gen = Gen+1
          end
 toc
```

B1 - M.file code for TS algorithm (simple example)

```
%using of tabu search for one variable.
clear all
clc
clf
tic
format short
i_max =300;% maximum iterations
n=1;
nn=1;
A(n)=2;

 ObjV(n)= 0.0116*A(n)^4-0.2473*A(n)^3+1.7927*A(n)^2-
4.9426*A(n)+5.4150;

   C(n)=min(ObjV);
%  initialize tabu list
N=7; % proposed by Glover
TL=zeros(N,1);

k=1; % counter works every 10 iterations for changing the value of
standrad deviation 'S'
S=0.2; % Standard deviation
f=1;    % counter for best solution

                 k1=A(n);

                 k8=C(n);
figure (1)
plot(f,k1,'bx'),hold on
figure (2)
plot(f,k8,'bx'),hold on


 BS(f,:)=[k1,k8]% Best solution

    TL(N,:)=C(n);


while (n <=i_max)
 w=abs([Normrnd(A(n),S); Normrnd(A(n),S); Normrnd(A(n),S);
Normrnd(A(n),S);Normrnd(A(n),S) ;Normrnd(A(n),S) ;Normrnd(A(n),S);No
rmrnd(A(n),S)])


 for i=1:8

Chr=w(i,:);
a=Chr(1,1);

%---------------------

ObjVsel(i) = objfun12(a);
```

```
end

     minV= min(ObjVsel);
for p=1:8
     if minV==ObjVsel(p)
        ind=p;
      end

end

   C(n+1)=min(ObjVsel);
   A(n+1) =w(ind,1);


 if nn >20
     if (k/nn)> 0.2
      S=0.1;

      elseif (k/nn)==0.2
      S=0.5;

      else
      S=1;
     end
       nn=nn-21;
      k=0;
 end

 xx=true;
 for g=1:N
        if C(n+1)==TL(g,1)
         xx=false;
         end

 end

 if xx==true

       for j=1:N-1                  % Swap the element inside the TL
       TL(j,1)=TL(j+1,1);
       end
       TL(N,1)=C(n+1);

            if C(n+1) < C(n)

                  k=k+1

                C(n)= C(n+1);
                 A(n)=A(n+1);

                 % ---------------

                 k1=A(n+1);
                 k8= C(n+1);

                  f=f+1
```

```
figure (1)
plot(f,k1,'bx'),hold on
figure (2)
plot(f,k8,'bx'),hold on

                 BS(f,:)=[k1,k8];% Best solution
            end

  end

% Increment counter
       n=n+1;
       nn=nn+1;
end
 toc
k1(end)
k8(end)
TL;
```

B2- M.file code for TS algorithm (fault identification)

```
% using of tabu search for healthy machine with six variables for
clear all
clc
clf
tic
format short
i_max =170;% maximum iterations
n=1;
nn=1;
RsA(n)=2;RsB(n)=3;RsC(n)=1;Rra(n)=3;Rrb(n)=5;Rrc(n)=1;
RsAm=RsA(n);RsBm=RsB(n);RsCm=RsC(n);Rram=Rra(n);Rrbm=Rrb(n);Rrcm=Rrc
(n);
%-------------------------------------------------------------------
% Other machine parameters

LsAm=0.896;
LsBm=LsAm;LsCm=LsAm;
%------------------------------
MssABm=0.185;MssACm=MssABm;MssCBm=MssABm;MssBAm=MssABm;MssBCm=MssABm
;MssCAm=MssABm;
%------------------------------
Mrrabm=0.771;Mrracm=Mrrabm;Mrrbam=Mrrabm;Mrrbam=Mrrabm;Mrrbam=Mrrabm
;Mrrbcm=Mrrabm;Mrrcam=Mrrabm;Mrrcbm=Mrrabm;
%------------------------------
MsrBam=0.751;MsrAam=MsrBam;MsrAbm=MsrAam;MsrAcm=MsrAam;MsrBam=MsrAam
;MsrBbm=MsrAam;MsrBcm=MsrAam;MsrBam=MsrAam;MsrBbm=MsrAam;MsrBcm=MsrA
am;MsrCam=MsrAam;MsrCcm=MsrAam;MsrCbm=MsrAam;
%------------------------------
MrsaAm=0.751;MrsaBm=MrsaAm;MrsaCm=MrsaAm;MrsaBm=MrsaAm;MrsaCm=MrsaAm
;MrsbAm=MrsaAm;MrsbBm=MrsaAm;MrsbCm=MrsaAm;MrscAm=MrsaAm;MrscBm=Mrsa
Am;MrscCm=MrsaAm;
%-----------------------------
Lrcm=0.507;
Lram=Lrcm;Lrbm=Lrcm;
%-------------------------------------------------------------------
%Experimental data, armature voltages

V1=csvread('vp1.csv',2,0,[2,0,999,1]);TimeVSA=V1(:,1);min(TimeVSA);T
imeVSA=TimeVSA-min(TimeVSA);VSA=V1(:,2);
V2=csvread('vp2.csv',2,0,[2,0,999,1]);TimeVSB=V2(:,1);min(TimeVSB);T
imeVSB=TimeVSB-min(TimeVSB);VSB=V2(:,2);
V3=csvread('vp3.csv',2,0,[2,0,999,1]);TimeVSC=V3(:,1);min(TimeVSC);T
imeVSC=TimeVSC-min(TimeVSC);VSC=V3(:,2);
%-------------------------------------------------------------------
%Experimental data, armature currents

IA=csvread('p1.csv',2,0,[2,0,999,1]);TimeISA=IA(:,1);min(TimeISA);Ti
meISA=TimeISA-min(TimeISA);ISA=IA(:,2);
IB=csvread('p2.csv',2,0,[2,0,999,1]);TimeISB=IB(:,1);min(TimeISB);Ti
meISB=TimeISB-min(TimeISB);ISB=IB(:,2);
IC=csvread('p3.csv',2,0,[2,0,999,1]);TimeISC=IC(:,1);min(TimeISC);Ti
meISC=TimeISC-min(TimeISC);ISC=IC(:,2);
%-------------------------------------------------------------------
% Rotor speed
 wr=301.5
```

```
%-------------------------------------------------------------------
sim('Healthy_Machine');
error(n)=E
%-------------------------------------------------------------------
%  initialize tabu list

N=7; % proposed by Glover
TL=zeros(N,1);

k=1; % counter works every 10 iterations for changing the value of
standrad deviation 'S'
S=6; % Standard deviation
f=1;   % counter for best solution
k1=RsA(n);k2=RsB(n);k3=RsC(n);k4=Rra(n);k5=Rrb(n);k6=Rrc(n);k7=error
(n);

 figure(1)
 subplot(2,1,1),plot(f,k1,'bx',f,k2,'rh',f,k3,'go'),hold on;
 subplot(2,1,2),plot(f,k7,'bx'),hold on;
 figure(2)
 subplot(2,1,1),plot(f,k4,'bx',f,k5,'rh',f,k6,'go'),hold on;
 subplot(2,1,2),plot(f,k7,'bx'),hold on;
 mem(f,1:7)=[k1,k2,k3,k4,k5,k6,k7];
 TL(N,:)=error(n)

while (n <=i_max)
%-------------------------------------------------------------------
-------
%Generate new solution
w=abc([Normrnd(RsA(n),S) Normrnd(RsB(n),S) Normrnd(RsC(n),S)
Normrnd(Rra(n),S) Normrnd(Rrb(n),S) Normrnd(Rrc(n),S);
Normrnd(RsA(n),S) Normrnd(RsB(n),S) Normrnd(RsC(n),S)
Normrnd(Rra(n),S) Normrnd(Rrb(n),S) Normrnd(Rrc(n),S);
Normrnd(RsA(n),S) Normrnd(RsB(n),S) Normrnd(RsC(n),S)
Normrnd(Rra(n),S) Normrnd(Rrb(n),S) Normrnd(Rrc(n),S);
Normrnd(RsA(n),S) Normrnd(RsB(n),S) Normrnd(RsC(n),S)
Normrnd(Rra(n),S) Normrnd(Rrb(n),S) Normrnd(Rrc(n),S);
Normrnd(RsA(n),S) Normrnd(RsB(n),S) Normrnd(RsC(n),S)
Normrnd(Rra(n),S) Normrnd(Rrb(n),S) Normrnd(Rrc(n),S);
Normrnd(RsA(n),S) Normrnd(RsB(n),S) Normrnd(RsC(n),S)
Normrnd(Rra(n),S) Normrnd(Rrb(n),S) Normrnd(Rrc(n),S);
Normrnd(RsA(n),S) Normrnd(RsB(n),S) Normrnd(RsC(n),S)
Normrnd(Rra(n),S) Normrnd(Rrb(n),S) Normrnd(Rrc(n),S);
Normrnd(RsA(n),S) Normrnd(RsB(n),S) Normrnd(RsC(n),S)
Normrnd(Rra(n),S) Normrnd(Rrb(n),S) Normrnd(Rrc(n),S)]);
for i=1:8
Chr=w(i,:);
RsAm=Chr(1,1);RsBm=Chr(1,2);RsCm=Chr(1,3);Rram=Chr(1,4);Rrbm=Chr(1,5
);Rrcm=Chr(1,6);

sim('Healthy_Machine');
ObjVsel(i) = E;
end
     minV= min(ObjVsel);
for p=1:8
     if minV==ObjVsel(p)
        ind=p;
      end
end
```

```
 error(n+1) =min(ObjVsel)
    RsA(n+1) =w(ind,1)
    RsB(n+1) =w(ind,2)
    RsC(n+1) =w(ind,3)
    Rra(n+1) =w(ind,4)
    Rrb(n+1) =w(ind,5)
    Rrc(n+1) =w(ind,6)
%--------------------------------------------------------------------
%Change value of S

 if nn >30
     if (k/nn)> 0.2
      S=0.01
      elseif (k/nn)==0.2
      S=0.05
      else
      S=0.1
     end
       nn=nn-31
      k=0
end

  xx=true;
 for g=1:N
        if error(n+1)==TL(g,1)
         xx=false;
        end
 end
 %-------------------------------------------------------------------
 % Swap the element inside the TL
 if xx==true
       for j=1:N-1
       TL(j,1)=TL(j+1,1)
       end
       TL(7,1)=error(n+1)

            if error(n+1) < error(n)
                  k=k+1
                error(n)= error(n+1);

RsA(n)=RsA(n+1);RsB(n)=RsB(n+1);RsC(n)=RsC(n+1);Rra(n)=Rra(n+1);
Rrb(n)=Rrb(n+1);Rrc(n)=Rrc(n+1);
k1=RsA(n+1);k2=RsB(n+1);k3=RsC(n+1);k4=Rra(n+1);k5=Rrb(n+1);k6=Rrc(n
+1);k7=error(n+1); f=f+1
 figure(1)
 subplot(2,1,1),plot(f,k1,'bx',f,k2,'rh',f,k3,'go'),hold on;
 subplot(2,1,2),plot(f,k7,'bx'),hold on;
 figure(2)
 subplot(2,1,1),plot(f,k4,'bx',f,k5,'rh',f,k6,'go'),hold on;
 subplot(2,1,2),plot(f,k7,'bx'),hold on;
 mem(f,1:7)=[k1,k2,k3,k4,k5,k6,k7];
            end
  end
% Increment counter
       n=n+1
       nn=nn+1
end
toc
k1(end);k2(end);k3(end);k4(end);k5(end);k6(end);k7(end);TL;
```

C1- M.file code for SA algorithm (simple example)

```
%The test for simple function with one variable

clear
clc
n=1;
k=1;
%-------------------------------------------------------------------
% Search space
a=-5:1:5;
range_a=(max(a)-min(a))^2;
%------------------------------------------------------------------
% Initial temperature

Ts=10; T=Ts;
a(n)=1;

f(n)= 0.0116*a(n)^4-0.2473*a(n)^3+1.7927*a(n)^2- 4.9426*a(n)+5.4150;

%------------------------------------------------------------------
 while n < 100
   a(n+1)=5-10*rand;

f(n+1)= 0.0116*a(n+1)^4-0.2473*a(n+1)^3+1.7927*a(n+1)^2-
4.9426*a(n+1)+5.4150;
 %---------------------------------------------------------------
 % Calculate of the displacement
    d1=(a(n+1)-a(n))^2;
    dista=d1;
 %-----------------------------------------------------------------
 % Caculation of acceptance probabilty
    r1=dista/range_a;
    r2=Ts/T;
    PA=exp(-r1*r2);
    PAr=rand;
    if PA > PAr;
 %-----------------------------------------------------------------
 % Calculation of swap probabilty
     Error=f(n+1)-f(n);
     Ps_exp=exp(Error*r2);
     Ps=1/(1+Ps_exp);
     Psr=rand;
     if Ps > Psr
            f(n)=f(n+1);  a(n)=a(n+1);
 %-----------------------------------------------------------------
 % Temperature decrement function

            T=0.9*T;
            n=n+1
        end
    end
   k=k+1
end
figure(1)
plot(f); grid
xlabel('Accepted iterations');
```

```
ylabel('f')
title('Results of simulated annealing');
hold on;

figure(2)
plot(a); grid
xlabel('Accepted iterations');
ylabel('a');
hold on;
```

C2- M.file code for SA algorithm (fault identification)

```
%This test is for healthy machine with six variables

clear all
clc
clf
tic
n=1
C=1
H=1
format short;
%-------------------------------------------------------------------
% Initial temperature
Tstr=50;T=Tstr;
%-------------------------------------------------------------------
X11=2;X12=20;X21=2;X22=20;X31=2;X32=20;X41=2;X42=20;X51=2;X52=20;X61
=2;X62=20;
%-------------------------------------------------------------------
% Search space
 rg=sqrt(((X12-X11)^2+(X22-X21)^2+(X32-X31)^2+((X42-X41)^2+(X52-
X51)^2+(X62-X61)^2)/6);

 RsA(n)=4;RsB(n)=1;RsC(n)=0.1;Rra(n)=5;Rrb(n)=2;Rrc(n)=7;
%-------------------------------------------------------------------
% Other machine parameters
LsAm=0.896;LsBm=LsAm;LsCm=LsAm;
%------------------------------
MssABm=0.185;MssACm=MssABm;MssCBm=MssABm;MssBAm=MssABm;MssBCm=MssABm
;MssCAm=MssABm;
%------------------------------
Mrrabm=0.771;Mrracm=Mrrabm;Mrrbam=Mrrabm;Mrrbam=Mrrabm;Mrrbam=Mrrabm
;Mrrbcm=Mrrabm;Mrrcam=Mrrabm;Mrrcbm=Mrrabm;
%------------------------------
MsrBam=0.751;MsrAam=MsrBam;MsrAbm=MsrAam;MsrAcm=MsrAam;MsrBam=MsrAam
;MsrBbm=MsrAam;MsrBcm=MsrAam;MsrBam=MsrAam;MsrBbm=MsrAam;MsrBcm=MsrA
am;MsrCam=MsrAam;MsrCcm=MsrAam;MsrCbm=MsrAam;
%------------------------------
MrsaAm=0.751;MrsaBm=MrsaAm;MrsaCm=MrsaAm;MrsaBm=MrsaAm;MrsaCm=MrsaAm
;MrsbAm=MrsaAm;MrsbBm=MrsaAm;MrsbCm=MrsaAm;MrscAm=MrsaAm;MrscBm=Mrsa
Am;MrscCm=MrsaAm;
%-----------------------------
Lrcm=0.507;Lram=Lrcm;Lrbm=Lrcm;
%-------------------------------------------------------------------
%Experimental data, armature voltages
V1=csvread('vp1.csv',2,0,[2,0,999,1]);TimeVSA=V1(:,1);min(TimeVSA);T
imeVSA=TimeVSA-min(TimeVSA);VSA=V1(:,2);
V2=csvread('vp2.csv',2,0,[2,0,999,1]);TimeVSB=V2(:,1);min(TimeVSB);T
imeVSB=TimeVSB-min(TimeVSB);VSB=V2(:,2);
V3=csvread('vp3.csv',2,0,[2,0,999,1]);TimeVSC=V3(:,1);min(TimeVSC);T
imeVSC=TimeVSC-min(TimeVSC);VSC=V3(:,2);
%-------------------------------------------------------------------
%Experimental data, armature currents

IA=csvread('p1.csv',2,0,[2,0,999,1]);TimeISA=IA(:,1);min(TimeISA);Ti
meISA=TimeISA-min(TimeISA);ISA=IA(:,2);
IB=csvread('p2.csv',2,0,[2,0,999,1]);TimeISB=IB(:,1);min(TimeISB);Ti
meISB=TimeISB-                     min(TimeISB);ISB=IB(:,2);
```

```
IC=csvread('p3.csv',2,0,[2,0,999,1]);TimeISC=IC(:,1);min(TimeISC);Ti
meISC=TimeISC-min(TimeISC);ISC=IC(:,2);
%-------------------------------------------------------------------
% Rotor speed
 wr=301.5
%-------------------------------------------------------------------
RsAm=RsA(n);RsBm=RsB(n);RsCm=RsC(n);Rram=Rra(n);Rrbm=Rrb(n);Rrcm=Rrc
(n);

                     sim('Healthy_Machine');
                     % This is count the number of model run
                     C=C+1;
              error(n)=E;
   minimum=0;
      k=1;
%-------------------------------------------------------------------
U1=RsA(n);U2=RsB(n);U3=RsC(n);U4=Rra(n);U5=Rrb(n);U6=Rrc(n);U7=error
(n);

figure(1)

subplot(2,1,1),plot(n,U1,'bx',n,U2,'rh',n,U3,'go'),hold on;
subplot(2,1,2),plot(n,U7,'bx'),hold on;
figure(2)

subplot(2,1,1),plot(n,U4,'bx',n,U5,'rh',n,U6,'go'),hold on;
subplot(2,1,2),plot(n,U7,'bx'),hold on;
mem(H,1:7)=[U1,U2,U3,U4,U5,U6,U7];
%-------------------------------------------------------------------
  while C <2000;

        r=rand;
        RsA(n+1)=2+ 18*rand;
        RsB(n+1)=2+ 18*rand;
        RsC(n+1)=2+ 18*rand;
        Rra(n+1)=2+ 18*rand;
        Rrb(n+1)=2+ 18*rand;
        Rrc(n+1)=2+ 18*rand;

%-------------------------------------------------------------------
% Calculate of the displacement
   dX1=RsA(n+1)-RsA(n);dX2=RsB(n+1)-RsB(n);dX3=RsC(n+1)-RsC(n);

   dX4=Rra(n+1)-Rra(n);dX5=Rrb(n+1)-Rrb(n);dX6=Rrc(n+1)-Rrc(n);
   displace=sqrt((dX1^2 + dX2^2+ dX3^2)/3);
%-------------------------------------------------------------------
% Caculation of acceptance probabilty
  PA = exp(-(displace/rg)*(Tstr/T));
      r=rand;
      if r<PA

RsAm=RsA(n+1);RsBm=RsB(n+1);RsCm=RsC(n+1);
Rram=Rra(n+1);Rrbm=Rrb(n+1);Rrcm=Rrc(n+1);

      sim('Healthy_Machine');
             C=C+1
             error(n+1)=E;
%-------------------------------------------------------------------
```

```
% Calculation of swap probabilty
  dif1=abs(error(n+1)-minimum);dif2=abs(error(n)-
minimum);dif3=(dif1-dif2)/dif2;
  Pswap=1/(1+exp(dif3*(Tstr/T)));

          if r< Pswap

error(n)=error(n+1);RsA(n)=RsA(n+1);RsB(n)=RsB(n+1);RsC(n)=RsC(n+1);
Rra(n)=Rra(n+1);
Rrb(n)=Rrb(n+1);Rrc(n)=Rrc(n+1);
%-------------------------------------------------------------------
% Temperature decrement function

        T=0.9*T;

U1=RsA(n);U2=RsB(n);U3=RsC(n);U4=Rra(n);U5=Rrb(n);U6=Rrc(n);U7=error
(n);
figure(1)
subplot(2,1,1),plot(n,U1,'bx',n,U2,'rh',n,U3,'go'),hold on;
subplot(2,1,2),plot(n,U7,'bx'),hold on;
figure(2)
subplot(2,1,1),plot(n,U4,'bx',n,U5,'rh',n,U6,'go'),hold on;
subplot(2,1,2),plot(n,U7,'bx'),hold on;
                  H=H+1
                mem(H,1:7)=[U1,U2,U3,U4,U5,U6,U7];

                            n=n+1
          end
                          end
end
toc
```