Detection and Discrimination of Inrush Current and Faults in Three-Phase Power Transformers Using Signal Processing and Machine Learning Technique

Othman T E Abdusalam

A thesis submitted to the Cardiff University in candidature for the degree of Doctor of Philosophy

Wolfson Centre for Magnetics Cardiff School of Engineering Cardiff University

Wales, United Kingdom

(May,2023)

ACKNOWLEDGEMENTS

The research presented in this thesis was conducted at the School of Engineering, specifically the Electrical and Electronic Engineering department, at Cardiff University.

I would like to express my gratitude to ALLAH for providing me with the necessary strength and guidance to complete this project. I also acknowledge the resources and opportunities made available to me by Cardiff University, without which this work would not have been possible.

I am particularly thankful to my academic supervisors, Dr. Fatih Anayi and Dr. Michael Packianather, for their unwavering support, guidance, and expertise throughout my PhD studies. Dr. Anayi's encouragement, suggestions, and understanding have been invaluable to me. I also appreciate the time and attention he has invested in my research. Dr. Packianather's assistance has been crucial to my success, and I am grateful for his guidance and patience.

Lastly, I want to express my heartfelt appreciation to my family, particularly my mother and father, for their unwavering love, support, and encouragement throughout my life.

ABSTRACT

Transformers are integral components of power systems, and their protection is critical to ensure the reliability and safety of the entire system. Internal faults in transformers are a major cause of failure, and thus, effective protection measures are essential to mitigate the risks associated with such faults. However, existing protection techniques are not infallible, and it is imperative to minimize damage to the transformer in the event of a fault, to reduce repair costs and prevent disruptions to the system. To address these issues, this thesis proposes three methods represented in the fourth study for transformer protection that are fast and highly effective in detecting both internal and external faults. The proposed techniques aim to overcome the challenge of inrush currents and saturation cases, which can lead to malfunctions in the protection system and lower the transformer's efficiency. The proposed protection technique is based on current signal processing and machine learning techniques, specifically, the use of alienation coefficients and machine learning classifiers based on practical and simulation tests. The proposed technique was successfully demonstrated using Simulink and simulations in the MATLAB software program and a laboratory model. The methods can identify and detect faults in phase A, with the same efficiency when applied to phases B and C. In the first study, the effectiveness of the alienation coefficient method in detecting common transformer faults, including cases of current overload, was demonstrated through a study where the method was applied to a transformer model created using MATLAB/SIMULINK. During steady state, it has been shown that the technique is efficient and fast at detecting faults as it detects faults between 2ms -3ms depending on the type of fault. In the second study, by using data from the laboratory, all possible faults were generated and imported to MATLAB for processing with the algorithm. The practical demonstration of the technique showed its effectiveness in detecting and eliminating internal faults within 3ms, depending on the strength of the fault current. The technique also proved to be capable of effectively differentiating between external and internal faults. Additionally, the test included examining current transformer (CT) saturation, which is a significant problem associated with protection schemes. In this study, the effectiveness of the alienation coefficient method was compared to a discrete Wavelet transform (DWT) method, both of which were applied to achieve the same objective. The proposed method was found to be faster than the DWT due to certain significant factors, with a fault being detected in just 3ms compared to the 5ms required by DWT. The third study proposes the use of three machine learning classifiers support vector machine (SVM), k-nearest neighbors (KNN) and artificial neural network (ANN), - for fault type detection in power transformers. The classifiers were trained on selected features and showed high accuracy in fault detection and classification. An optimization algorithm-based feature selection was performed to validate the effectiveness of the model, revealing that the proposed

strategies with fewer statistical features provided more accurate results. Combining the Bees algorithm with an SVM classifier resulted in maximal accuracy of 96% in fault classification, with high accuracy observed in evaluation-based testing. The fourth study proposes a novel technique for fault classification in a three-phase power transformer using SVM. The method uses two classifiers, SVM1 for fault detection and SVM2 for identifying internal and external faults. The technique achieves fault identification and detection in less than a quarter cycle of the phase current of phase A, with the same technique applied to phases B and C. The classification of faults is not affected by the type or location of the fault, and the SVM method is shown to be more accurate and faster than the conventional ANN method. The proposed technique is effective in detecting different types of faults with high accuracy under different operating conditions and provides an efficient and reliable approach for fault classification in power transformers.

LIST OF PUBLICATIONS

Journal papers:

[1] O. Abdusalam, A. Ibrahim, F. Anayi and M. Packianather, "New Hybrid Machine Learning Method for Detecting Faults in Three-Phase Power Transformers", Energies, vol. 15, no. 11, p. 3905, 2022. Available: 10.3390/en15113905.

[2] T. Alghamdi, O. Abdusalam, F. Anayi and M. Packianather, "An artificial neural network based harmonic distortions estimator for grid- connected power converter-based applications", Ain Shams Engineering Journal, p. 101916, 2022. Available: 10.1016/j.asej.2022.101916.

Conference papers:

[1] O. Abdusalam, F. Anayi and M. Packianather, "Discrimination between internal current and external fault in three phase power transformer by Using alienation coefficient," 2022 57th International Universities Power Engineering Conference (UPEC), Istanbul, Turkey, 2022, pp. 1-7, doi: 10.1109/UPEC55022.2022.9917688.

[2] O. Abdusalam, F. Anayi and M. Packianather, "Three-Phase Power Transformer Fault Diagnosis Based on Support Vector Machines and Bees Algorithm," 2023 IEEE 3rd International Conference in Power Engineering Applications (ICPEA), Putrajaya, Malaysia, 2023, pp. 145-150, doi: 10.1109/ICPEA56918.2023.10093147.

LIST OF CONTENTS

LIST OF FIGURES

CHAPTER 2

CHAPTER 3

CHAPTER 4

CHAPTER 5

CHAPTER 6

CHAPTER 7

LIST OF TABLE CAPTIONS

CHAPTER 2

CHAPTER 4

CHAPTER 5

CHAPTER 6

CHAPTER 7

LIST OF ABBREVIATIONS AND NOMENCLATURES

a	Transformer turns ratio
A	Cross-sectional area
AC	Alternating Current
AI	Artificial Intelligence
Amp	Ampere
ANN	Artificial Neural Network
B	Flux density
BA	Bees Algorithm
Bm	Susceptance
Bmax	Maximum flux density
CT	Current Transformer
DC	Direct Current
DCR	Ratio of DC
DWT	Discrete Wavelet Transform
e	Induced voltage
emf	Electromagnetic force
f	Frequency
Gc	Conductance
H	Magnetic field strength
Hz	Hertz
i	Current
I0	Exciting current or no-load current
Ic	Core loss current
Id1	Fundamental component of the differential current

Id2	Second harmonic component of the differential current
ID2	Value of normalized second harmonic component
IEEE	Institute of Electrical and Electronics Engineers
ILn	Line current
Im	Magnetizing current
IOP	Operating or differential current
IPh	Phase current
IPU	Pickup current
IRT	Restraint current
j	Imaginary part
K2	Constant coefficient of second harmonic
K3	Constant coefficient of third harmonic
K4	Constant coefficient of fourth harmonic
K5	Constant coefficient of fifth harmonic
Kc	Compensation factor
kg	Kilogram
Kh3	Threshold restraint ratio of third harmonic
KNN	K-Nearest Neighbors algorithm
kVA	Kilo Volt Ampere
L	Coil inductance
l	Length of wire
m	Number of samples per window
Max	Maximum
Min	Minimum
ms	Millisecond

mmf	Magnetomotive force
MVA	Mega Volt Ampere
N	Number of turns
n	Number of samples
Ø	Magnetic flux
Øl	Leakage flux
OLTC	On-load Tap Changer
Ømax	Maximum flux value
OMP	Orthogonal matching pursuit
Ør	Residual flux
P	Power
Pave	Average power
Pc	Core power loss
Pcu	Copper power loss
PI	Power index
Pri	Primary
Pmax	Maximum of average power
PPri	Power loss on primary side
PSec	Power loss on secondary side
Pu	Per unit
R	Resistance
r	Pearson's correlation coefficient
r2	Determination coefficient
R3rd	Ratio of third harmonic
RBF	Radial Basis Functions

Rf	Fault or protection resistor
RIFL	Ratio of Increment of Flux Linkages
RIV	Ratio of Induced Voltage
R_{Load}	Load resistance
Rm	Magnetization resistance
rms	Root mean square
Sec	second
SVM	Support vector machine
SW	Switch
t	time
T	Sampling interval
Th	Threshold value
v	Voltage
V	Volt
VA	Volt Ampere
VLn	Line votlage
Vm	Maximum voltage
VPh	Phase votlage
W	Watt
WT	Wavelet Transform
x	Coil reactance
xl	Leakage reactance
y	Exciting admittance
z	Impedance
λ	Flux linkage

ρ	Resistivity of material
σ	Standard deviation
ω	Angular frequency
Ω	Ohm
$\Omega \cdot m$	Ohm·meter

CHAPTER 1

Introduction and Objectives of the Thesis

1.1 Principles in protecting transformer

Systems to protect transformers primarily aim to minimize the extent to which transformers that develop faults are damaged in the period between the fault occurring and the transformer being repaired or a new one substituted. Transformers are costly and form a central element within power systems. Thus, there is a need for rapid and reliably performing protection relays to disconnect faulty transformers from networks and avoid greater damage occurring. In addition, this action reduces stresses upon the network system, as well as avoiding damaging neighboring components in the system. The design of power systems allows for removal of multiple components from such systems where a fault is identified, while not placing excessive strain on the system. It is essential that transformers are carefully monitored, because of the risk that a fault could lead to catastrophic failure with permanent internal harm to the apparatus [1, 2]. However, protective systems do not eliminate the risk of systems damage, but rather limit this damage as far as is practical, therefore limiting the duration of outages and the costs of repairing the system. Protective systems for transformers have different features based on the operational context and how important the transformer is considered. All transformer protection systems seek to lower fault-based outage times and make catastrophic failures less likely, thus decreasing repair costs overall. The transformer's working lifespan is progressively reduced the longer it operates in a sub-optimal condition, including when there is a fault. Based on this, effective protection is important to more rapidly isolate transformers with faults and thus reduce the time they are in operation with a fault [3].

1.2 Transformer faults

Transformers which fail due to short circuits occurring externally or internally are a key issue across the manufacturing industries. An internal fault is one which arises within the protected area of the transformer and may be a turn-to-turn or a turn-to-ground fault, for example, with any fault arising outside the protected area being classed as external. According to literature, internal faults based on short-circuiting of the internal windings account for between 7 and 8 out of every 10 occurrences of transformer failure [4]. Such internal faults in transformers must be rapidly identified and

repaired to minimize direct damage to the transformer as well as protecting the power system as a whole and thus avoiding major costs [5]. Any short-circuit or earth fault risks severely damaging the transformer's core and windings. In addition, where a fault current is high, there is the possibility that the tank's gas pressure may rise dramatically, causing the transformer to explode [6]. One of the main current issues in protecting transformers is therefore the speed and precision with which internal transformer faults are identified. In conclusion, the primary requirement for systems which are used for protecting transformers is that they have high sensitivity to any internal transformer fault and are not affected by externally occurring faults which may arise outside the protected area.

1.3 Differential protection

Differential protection systems are long established for transformer protection, with over five decades of use history, and are the main approach used in protecting transformers [7]. However, this type of system is challenged by inrush currents which are generated at the initial energization of the transformer. This is due to the imbalance they produce in the currents entering and exiting the equipment, which is an expected and transient effect, but which such systems can identify as a fault and activate the differential protective relay inappropriately. The need to distinguish genuine internal faults from the effects of inrush current is an issue which has therefore attracted a great deal of attention in the field of transformer protection, and forms the central challenge addressed in research in this area. Relays are expected to activate where an internal fault emerges but must not activate due to externally occurring faults or inrush currents [8]. What this means in practice is that current transformer saturation, inrush currents and systems operation modes decrease the accuracy of transformer protection, which can fall to as low as 75% due to this, which is unsatisfactory [9]. Generally speaking, there are 3 categories of approach to transformer protection, which are differential, pressure, and overcurrent protection, and these are selected based on the context. Differential protection is designed to identify internally occurring faults in the windings of the transformer. It uses the differential in the strength of the primary and secondary currents, which is a comparatively minor difference when the transformer is operating normally, or when a fault is present externally, and this small difference does not normally activate the relay [10]. In some conditions however, differential protection systems can malfunction, and to minimize this risk, there is a need to identify whether there is an external or an internal fault, or if differences are due to inrush current [11-12].

Differential protection methods have been developed using current second harmonics signal to address this issue. However, this approach has become less effective because of changes in the construction of transformers to reduce core power losses. These changes include for example the use of novel amorphous material, generating lower harmonics content compared to conventional

materials and therefore reducing the effectiveness of harmonics approaches, and especially the second harmonics. Despite this, and other drawbacks discussed in depth in the literature, second harmonics techniques remain in general use in transformer protection, as consensus has not been reached on a suitable replacement method. There is strong interest in developing a suitable approach however, based on the central role which transformers play within power systems, and researchers are vying to develop the most effective solution, taking into account costs, complexity and the need for rapid responsiveness. Each approach which has so far been put forward has flaws, however, as will be discussed in the literature review.

1.4 Objectives and aims of the thesis

The research endeavors to achieve several interconnected objectives and aims within the realm of transformer protection and fault detection:

Firstly, the primary aim was to employ a real transformer model within a laboratory setting to gather authentic data pertaining to inrush currents, as well as internal and external faults. This data served as the foundation for subsequent investigations. Secondly, the research sought to create and simulate all conceivable faults that might occur during transformer operation. The goal was to rigorously assess whether the proposed techniques had the capability to accurately detect these faults in real time. Thirdly, the proposed methods underwent evaluation in the laboratory environment using actual data. This step was crucial in ensuring that the testing procedures closely resembled real-world conditions, thus enhancing the reliability of the findings. Moreover, the research aimed to develop innovative methods primarily focused on the rapid detection of internal faults, including minor ones such as turn-turn faults. These types of faults pose a significant challenge, particularly when they involve only a few turns. Adhering to IEEE Standard C37.91-2000, which specifies the need to short-circuit at least 10% of the turns in a transformer winding to detect changes in terminal current, the objective was to create techniques that could efficiently address these intricate scenarios. In line with these objectives, the overarching aims of the research were to design algorithms characterized by simplicity and ease of implementation. These algorithms were intended to minimize the demand for computational resources, relying solely on the analysis of the current signal. Notably, the two major challenges in transformer protection, namely inrush currents and transformer saturation, were central to the research's focus as shown in literature review. The ultimate goal was to develop transformer protection methods that effectively addressed these challenges. The validation of these methods hinged on practical experimentation, utilizing real- data to ascertain their efficacy.

1.5 Structure of the thesis

Chapter 1. A description of the project and its objectives. Presents the aims of the study and the thesis outline.

Chapter 2. Problems related to transformers. Since transformers are the subject of this research, this chapter introduces background information about transformers, such as transformer theory, equivalent circuits, connections, and core materials. The chapter also discusses how inrush phenomena and current transformer saturation occur, along with transformer faults, since those are the problems that are aimed to be solved. Following the introduction, a discussion of relays and how they are used to protect transformers is presented, and, the differential protective relay, whose characteristics are explained.

Chapter 3. Literature reviews. The literature review chapter is an essential part of academic research that provides a comprehensive overview of previous studies on the subject. This chapter typically entails an extensive survey and evaluation of existing literature, scholarly articles, books, and other relevant sources to identify gaps, themes, and trends in the research area.

Chapter 4. Setup of Experiments and Measurements. This chapter represents the start of the practical work. This Chapter shows how the transformer was constructed, how saturated currents were generated in the laboratory and how faults were generated in the transformer model. Furthermore, it gives details regarding the equipment used for the experiments, the method of carrying out the experiments, the calculations needed, and how the acquired data are stored in preparation for further analysis.

Chapter 5. This chapter presented two parts using alienation coefficients method. The first part discussed proposed protection techniques and transformer model simulation. MATLAB/SIMULINK is used to simulate transformer models. With the proposed methods, inrush current problems, transformer saturations, and all possible faults are simulated and tested in each state and the second part discussed the practical test on the proposed techniques.

Chapter 6. In this chapter, hybrid machine learning technique is used in order to detect, classify, predict, and prevent failures from occurring during the operation of transformer. The objective of this study was to develop and apply a new algorithm to detect, classify, and predict transformer faults at their earliest stages.

Chapter 7. This chapter represents a new approach to the classification of current signals in the 3-phase transformer by Using SVM. It distinguishes between the inrush currents, internally and

externally occurring faults. The basis for the approach described here is the Support Vector Machine (SVM), and two SVM classifier types are utilized. SVM1 is applied for fault and inrush current identification for the 3-phase transformer, while SVM2 identifies faults as internally or externally occurring.

Chapter 8. Conclusion and Future Work. In this chapter, the results of testing the proposed techniques in both simulation and practical work are discussed. In addition, some recommendations for the future are present.

CHAPTER 2

Transformer and related issues

2.1 Introduction

Transformers are static electrical devices that convert AC voltage from one level to another at the same frequency by utilizing magnetic fields. The basic structure of a transformer is illustrated in Figure 2-1-a. Typically, the primary and secondary windings are connected magnetically through a ferromagnetic core, with the AC voltage source being connected to the primary winding, and the load being connected to the secondary winding. Through mutual magnetic flux, the two windings are interlinked [13]. Electromagnetic induction was discovered by Michael Faraday and Joseph Henry in 1831. However, Faraday was the first to publish his findings, thus earning him the distinction of being the first to describe this phenomenon. Faraday discovered that a magnetic field with time-varying energy induces a voltage in a winding. The basic model of a transformer is shown in Figure 2-1-a, which illustrates the principle of electromagnetic induction in action. AC power sources generate magnetic flux in the primary winding. As a result, the flux lines intersect the turns of the secondary winding and induce voltage in that winding. Faraday's original transformer, which features two copper coils that are insulated with cotton and wound around an iron core, is depicted in Figure 2-1-b [14]. This principle applies to all transformers that exist today.

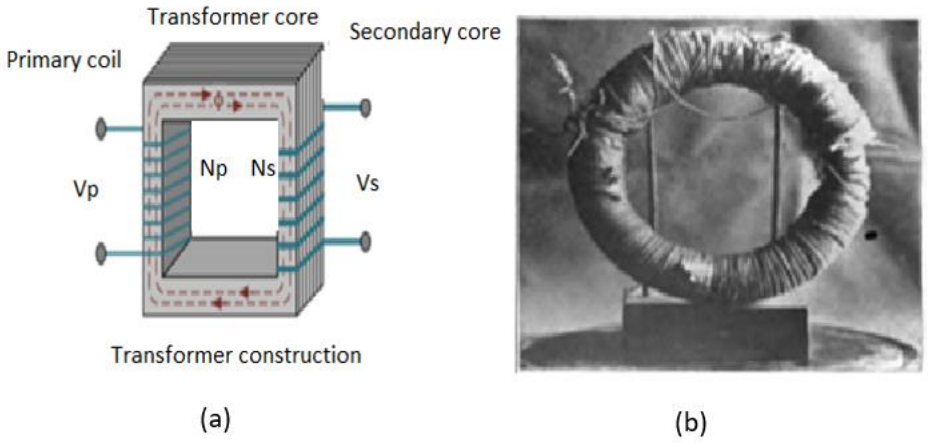

Figure 2- 1 (a) Transformer basic model (b)Figure Faraday's transformer original model [13] [14].

Transformers are a crucial component of power systems and find extensive use in transforming voltage levels. Over time, transformers have become available in various sizes and capacities and

are an integral part of modern life. Broadly, transformers can be categorized into two types based on their applications. The first category includes step-up transformers that convert low voltages to high voltages as required, while step-down transformers do the opposite. The transformer can be a step-up or a step-down transformer by setting the appropriate turn ratio, which is defined as the number of turns in the primary winding divided by the number of turns in the secondary winding. Transformers are classified based on their turn ratio.

2.1.1 Transformers core construction

Most cores are constructed from thin laminations of ferromagnetic materials. Cores are formed by stacking the laminations on top of one another. By reducing eddy currents within the core, this structure reduces losses. Two common designs of core and winding configuration have been accepted for improving magnetic coupling between primary and secondary windings and minimizing flux leakage [15].

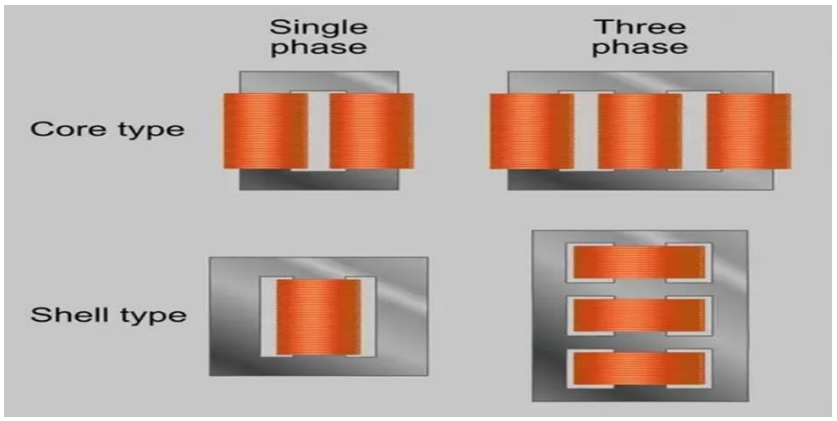

Figure 2- 2 (a) Core-type transformer (b) Shell-type transformer [15].

As shown in Figure 2-2(a,b), primary and secondary windings of the core type are wound around the periphery of the core, whereas in the shell type, the two windings are wound around one common limb within the core.

2.1.2 Theory of Transformer

The ideal transformer serves as a theoretical construct for studying transformer theory. This type of transformer is characterized by the absence of any losses or leakage flux and possesses a core with infinite magnetic permeability and electrical resistivity. The ideal transformer assumes that the entire flux of both windings is confined to the core that interlinks both windings. It is essential to note that this is not possible in practical transformers, as the aforementioned assumptions neglect the existence of certain quantities or components. Nonetheless, the ideal transformer is often employed to facilitate ease and clarity in studying the principles of transformers [16].

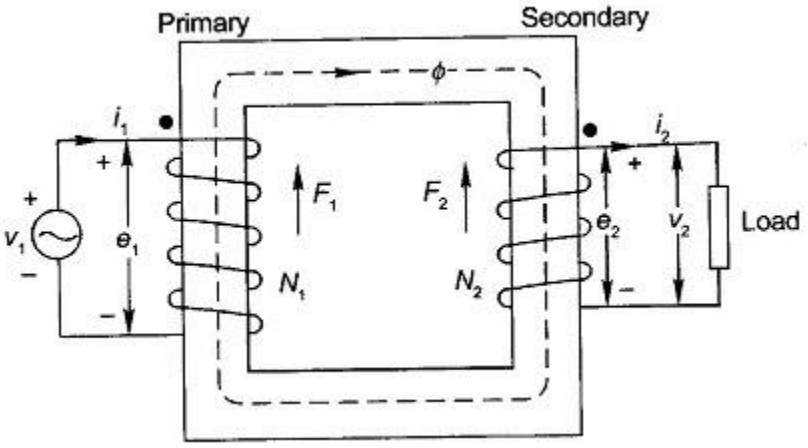

Figure 2- 3 Ideal transformer model [16].

When the primary winding of a transformer is supplied with AC voltage v_1, an electromagnetic force (emf) e_1 is induced, assuming that there is no resistance in the primary winding. This emf is produced as a result of the magnetic flux generated by the primary current i_1 flowing through the primary winding, as illustrated in Figure 2-3. The emf e_1 can be mathematically expressed in accordance with Faraday's induction principle as follows:

$$V_1 = e_1 = \frac{d\lambda_1}{dt} = N_1 \frac{d\varphi}{dt} \qquad (2-1)$$

Where the number of turns in the main winding is N_1 and flux linkage is λ_1 (λ_1=N1Ø).

The flux Ø cut all turns of winding to secondary side N_2, which induces e_2 on this side, as the presumption is that no leakage stream exists.

$$V_2 = e_2 = \frac{d\lambda_2}{dt} = N_2 \frac{d\varphi}{dt} \qquad (2-2)$$

Here with the secondary turn winding N_2 the flux linkage is λ_2.

The secondary current i_2 flows in the circuit when the load is attached to the secondary winding ends. So V_2 is equal to e_2 because, as shown in Fig 2-3, the winding resistance is ignored.

The result of dividing equation (2-1) by equation (2-2):

$$\frac{V_1}{V_2} = \frac{N_1}{N_2} = \frac{e_1}{e_2} = a \qquad (2-3)$$

Equation (2-4) can be achieved with Ampere's law in the closed circuit of magnetic flux where transformer turn ratio is a.

$$N_1 i_1 - N_2 i_2 \ = 0 \qquad (2-4)$$

Now the equation 2-2 is follow:

$$\frac{i_1}{i_2} = \frac{N_2}{N_1} = \frac{1}{a} \qquad (2-5)$$

The instantaneous input power P_1 equal the instantaneous output power P_2, i.e. because the ideal transformer does not have a power loss.

$$P_1 = P_2 \quad or \quad v_1 i_1 \ = v_2 i_2 \qquad (2-6)$$

Let v_1 and Ø the flux to be sinusoidal. The flow function can be as follows:

$$\emptyset = \emptyset_{max} \ sinwt \qquad (2-7)$$

$$B_{max} = \frac{\emptyset_{max}}{A} \qquad (2-8)$$

The A is core cross-sectional area, while $Ø_{max}$ is the maximum flux value, with an angular frequency ($\omega = 2\ \pi f$, f is the frequency). The emf e_1 can be achieved as a substitute for Ø from (2-7).

$$e_1 = N_1 \frac{d_\varphi}{dt} = \omega N_1 \emptyset_{max} \ \ cos\, wt \qquad (2-9)$$

The rms value of emf e_{rms} is then given by.

$$e_{rms} = \frac{2\pi}{\sqrt{2}} f N_1 \emptyset_{max} \qquad (2-10)$$

By replacing B_{max} (2-8) with (2-10), e_{rms} can be represented.

$$e_{rms} = 4.44 f N_1 B_{max} A \qquad (2-11)$$

When sinusoidal voltage is applied to transformer cores, equation (2-11) provides the maximum flux density.

2.1.3 Transformer on no-load

Transformers that have no load on their secondary winding are referred to as no-load transformers. However, due to the core's finite permeability and core losses, an exciting current, denoted by I_0, flows through the primary side of the transformer in this state. An equivalent circuit of an ideal transformer with an exciting circuit is depicted in Figure 2-4-a [16, 17], which demonstrates two constituents of the current, namely the magnetizing current I_m and the core loss current I_c.

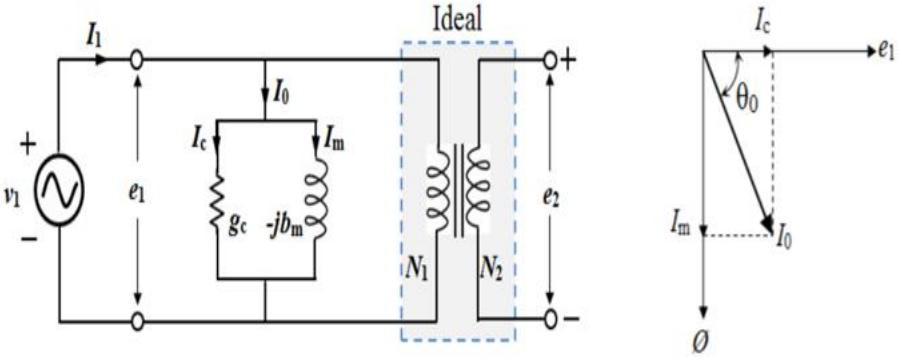

Figure 2- 4 (a) Equivalent circuit of transformers on no-load (b) Diagram of Phasor [16].

The induced voltage e_1 as well as core loss current I_c are in the same phase as in Fig 2-4-b.

$$I_c = \frac{P_C}{e_1} \qquad (2-12)$$

The primary power loss, P_C , is mainly attributed to the hysteresis and eddy current losses that occur in the transformer's core. The magnetizing current, I_m and core loss current I_c are phase shifted by 90 degrees as depicted in Figure 2-4b. The magnetization and core losses can be modeled using an equivalent circuit that comprises a resistor in parallel with an inductor, known as the exciting circuit. As there is no load connected to the secondary winding, there is no load current flowing through it.

$$I_0 = \sqrt{I2_c + I2_m} \qquad (2-13)$$

P_f (power factor) can be calculated by

$$cos\theta_0 = \frac{I_c}{I_0} \qquad (2-14)$$

y_0 represents the exciting admittance.

$$y_0 = g_c - jb_m = \frac{I_0}{e_1} \qquad (2-15)$$

Where b_m and g_c are represented susceptance and conductance of the exciting circuit.
Transformers exhibit non-sinusoidal behavior due to the nonlinearity of their magnetic properties arising from ferromagnetic circuits. This nonlinearity of the core material properties gives rise to an exciting current that generates a non-sinusoidal waveform. The exciting current is composed of two types of harmonics: the fundamental harmonic and the odd harmonics. Typically, the third harmonic

dominates the exciting current, accounting for 40% to 50% of the total current in a typical transformer.

2.1.4 Three-phase Transformers

The three-phase system, which is widely used in electrical power transmission, involves the use of three-phase transformers. These transformers distribute power across three phases using three alternating currents and voltages. At each step, one current and voltage between these phases is separated by 120 phase shifted. As illustrated in Figure 2-5a and 2-5b, a three-phase transformer is essentially a collection of three separate and equivalent one-phase transformers with a shared core. The uppercase A, B, and C are used to represent the three phases on the primary side, while the lowercase letters a, b, and c are used to represent the three phases on the secondary side. At balanced loads and voltages, the three single-phase transformers carry one-third of the total load each [19]. The primary advantage of using a three-phase transformer over three individual transformers is that it is smaller, less expensive, and requires less space. It also has only six external connections, compared to twelve in the case of three single-phase transformers. However, if one phase is disconnected, the entire transformer must be taken offline and replaced, which can be more expensive than repairing a single-phase transformer.

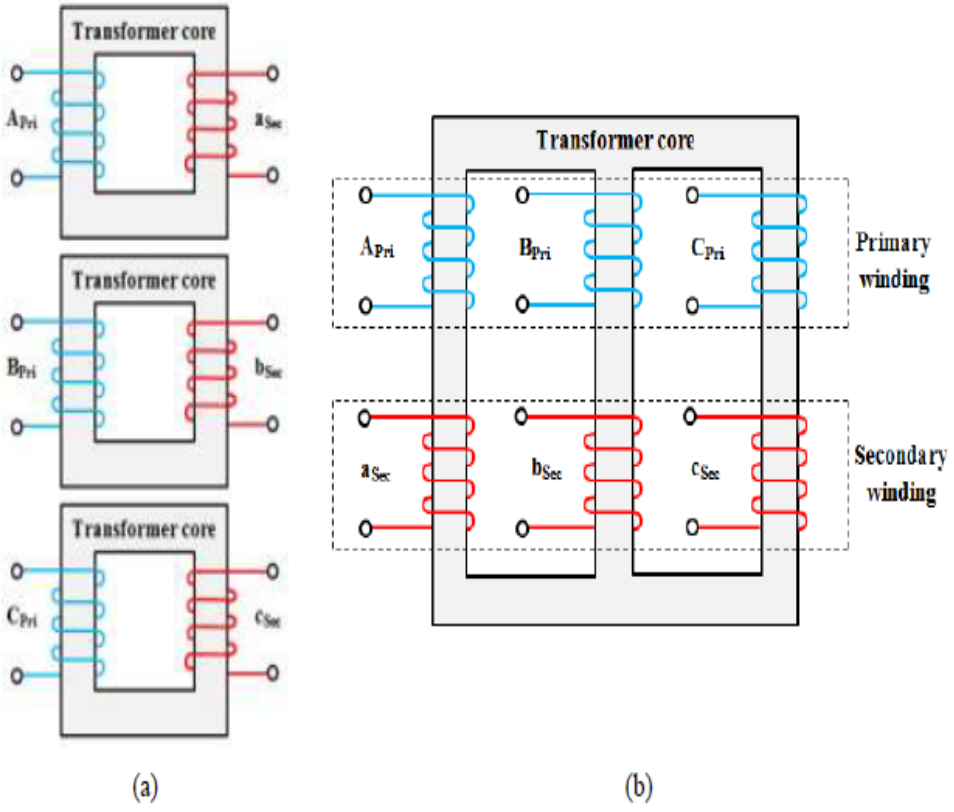

Figure 2- 5 (a) Three signal-phase transformer bank (b) Single three phase transformers [19].

2.1.5 Transformer connection

To ensure compatibility with a 3-phase power supply, the windings of a transformer must be interconnected to form a 3-phase configuration. Two types of winding connections are commonly used: Star (Y) and Delta (Δ). For a 3-phase, two-winding transformer, there are four possible connections: Y-Y, Y-Δ, Δ-Y, and Δ-Δ. However, only the Y-Y and Δ-Δ connections are commonly used due to their ease of connection, as illustrated in Figure 2-6. The neutral point, denoted as N, is the common point for interconnecting the three winding end terminals with a Star or Y connection [18].

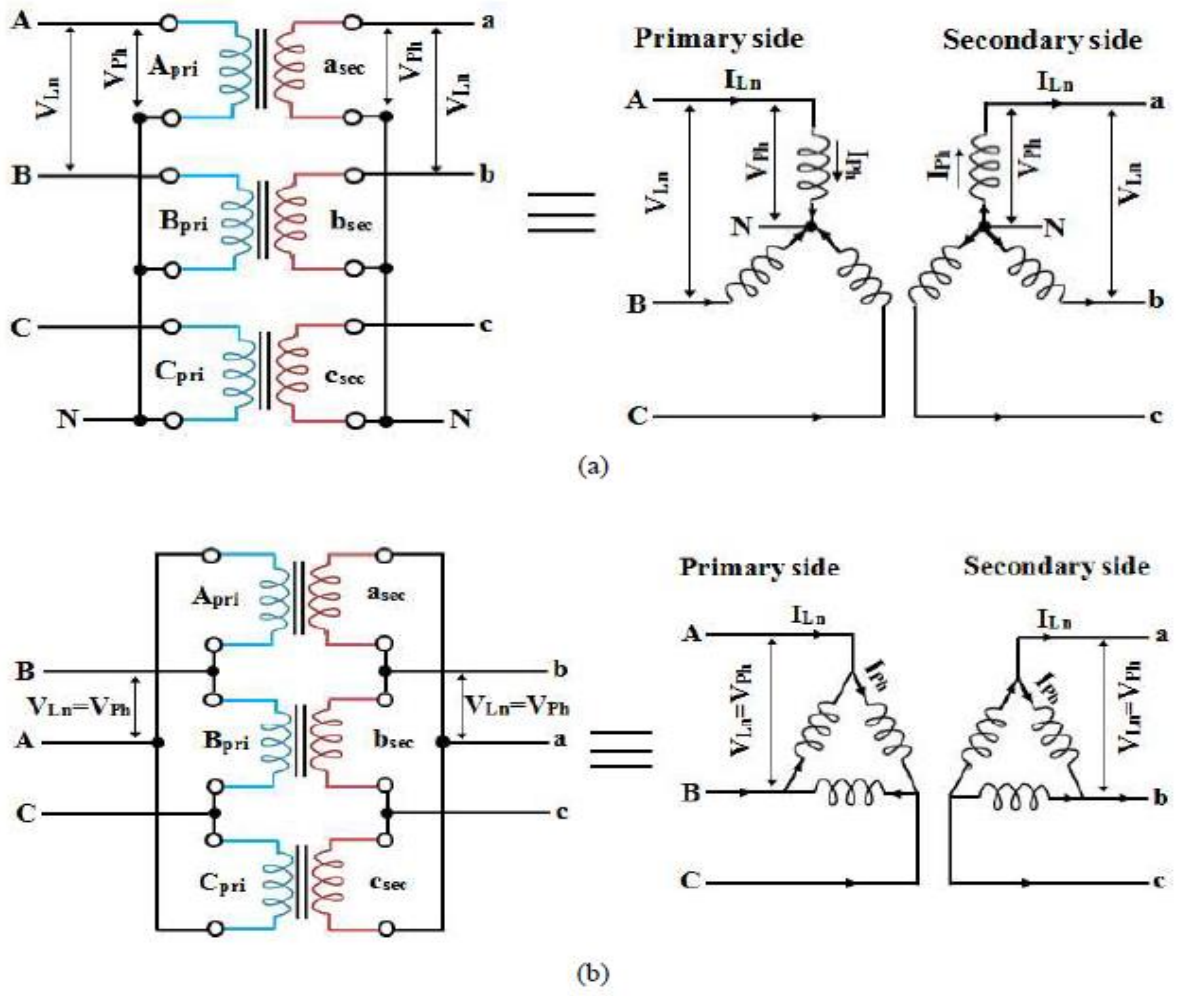

Figure 2- 6 Transformer winding connections (a) Y-Y connection (b) connection [19].

Table 2-1 presents a summary of the relationship between the line-to-line voltage V_{Ln} or current I_{Ln} and the phase voltage V_{Ph} or current I_{Ph} for both Delta and Star connections in a three-phase system.

Table 2- 1 Relationship of line-line and phase-neutral voltages and currents

Connection type	Phase voltage V_{ph}	Line voltage V_{Ln}	Phase current I_{ph}	Line current I_{Ln}
Star	$V_{ph} = \frac{1}{\sqrt{3}} V_{Ln}$	$V_{Ln} = \sqrt{3} V_{ph}$	$I_{ph} = I_{Ln}$	$I_{Ln} = I_{ph}$
Delta	$V_{ph} = V_{Ln}$	$V_{Ln} = V_{ph}$	$I_{ph} = \frac{1}{\sqrt{3}} I_{Ln}$	$I_{Ln} = \sqrt{3} I_{ph}$

2.2 B-H curve and magnetic hysteresis loop

The B-H or B-Magnetization curve is a fundamental characteristic of core materials, which varies from material to material. It shows the relationship between flux density B and magnetic field force H (B-H). The curve for steel, iron and air can be observed in Figure 2-7 [20]. Notably, each material has its own saturation point (a knee level) on the B-H curve. This point represents the magnetic field force at which flux density B remains almost constant and no longer increases. After the point of saturation, magnetic field force H continues to increase without any corresponding increase in flux density B. It is important to note that the saturation point on the B-H curve differs among materials.

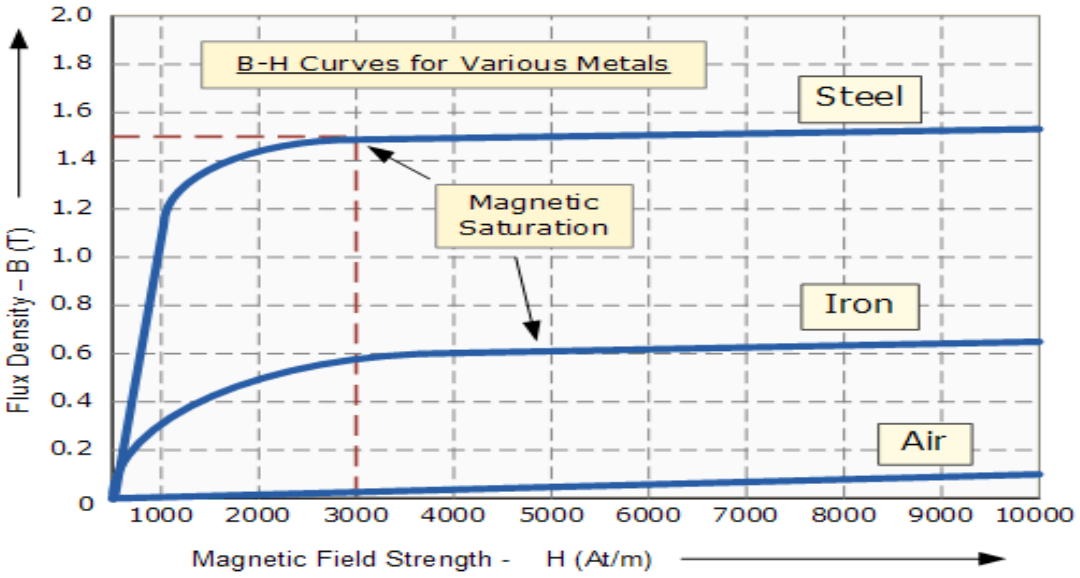

Figure 2- 7 B-H curve [20].

Figure 2-8 illustrates the behavior of a ferromagnetic core in terms of the relationship between the magnetic field strength H and the magnetic flux density B. Initially, the core is unmagnetized, with both H and B equal to zero. When a magnetizing current is applied, the magnetic field strength H increases, and the flux density B rises linearly until the point of saturation a is reached (path 0-a). Even when the magnetizing current is reduced to zero and the magnetic field strength returns to zero, the flux density does not decrease to zero due to the presence of residual flux in the core. Thus,

when the magnetic field strength is reduced to zero (path a-b), the flux density decreases to the residual flux point b. To reduce the flux density to zero, a reverse current can be applied, which produces a coercive force that demagnetizes the core (path b-c). At point c, the magnet is completely demagnetized, and the current changes direction, resulting in negative magnetization. The flux density then decreases further as the magnetic field strength becomes more negative until the point of saturation d is reached in the opposite direction (path c-d). Similarly, as the magnetizing current returns to zero (path d-e), the flux density returns to the negative residual flux at point e. The application of a stronger magnetizing current results in the coercive force demagnetizing the core in the opposite direction at point f, causing the flux density to decrease as the magnetizing current increases until the starting point of the path is reached again (path f-a). The shape of the magnetic hysteresis loop can vary depending on the properties of the material, with soft materials having narrow loops and hard materials having wider loops. The loop represents the complete cycle of magnetization and demagnetization that occurs as the alternating current changes polarity. The complete a-b-c-d-e-f-a path formed by the alternating current (AC) magnetizing current is referred to as the magnetic hysteresis loop, which forms a closed loop. In each cycle of AC magnetizing current, the direction alternates between negative and positive values on the spiral. This implies that the flux follows the same path, meaning that the core is magnetized and demagnetized in each process of applied AC voltage. The shape of magnetic hysteresis loops can be wide or narrow, indicating that the coercive forces and residual flux points can be higher or lower depending on the soft or hard materials used [20].

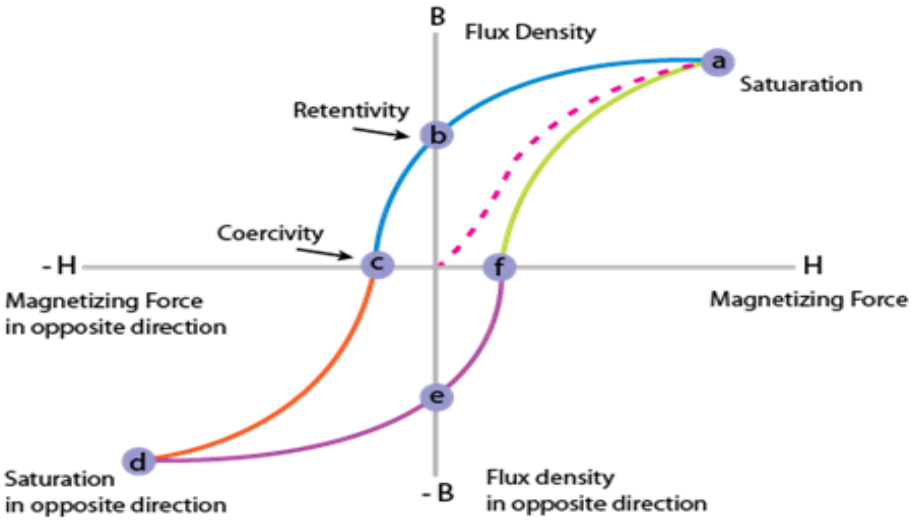

Figure 2- 8 Magnetic hysteresis loop [20].

2.3 Open-Circuit and Short-Circuit Tests in Transformers

The open circuit test and short circuit test are two common tests performed on transformers to determine their performance characteristics and to gather essential data for design and analysis. Here's an explanation of both tests:

1. Open Circuit Test:

- Purpose: The open circuit test, also known as the no-load test, is conducted to determine the core loss and magnetizing current of a transformer when it is connected to the primary side with no load (secondary side open).
- Procedure: Setup: The transformer is connected to a variable AC voltage source on its primary side, while the secondary side is left open or disconnected. Typically, a voltmeter, an ammeter, and a wattmeter are connected to measure primary voltage (V), primary current (I), and power (W) respectively.
- Test Execution: The primary voltage is gradually increased while keeping the frequency constant. Simultaneously, the primary current and power are measured.
- Data Collection: As the voltage is increased, the power consumption increases, reaching a point where it stabilizes. At this point, the transformer is operating at its rated magnetic flux, and the measured power represents the core loss.
- Interpretation: The power measured during the open circuit test represents the core loss, which consists of hysteresis and eddy current losses in the transformer's core.

2. Short Circuit Test:

- Purpose: The short circuit test, also known as the impedance test, is performed to determine the equivalent impedance (both resistance and reactance) of the transformer on its secondary side. This test helps calculate the voltage regulation and the short-circuit current of the transformer.
- Procedure: Setup: The primary side of the transformer is connected to a low-voltage source, while the secondary side is short-circuited. Instruments like a voltmeter, an ammeter, and a wattmeter are connected to measure the primary voltage (V), primary current (I), and power (W) respectively.

- Test Execution: The low-voltage source is applied to the primary winding while keeping the frequency constant. The primary current, voltage, and power are measured.
- Data Collection: The primary current and power are recorded, and the voltage is adjusted until the current stabilizes. At this point, the transformer is operating under short-circuit conditions.
- Interpretation: The primary current measured during the short circuit test allows for the calculation of the equivalent impedance of the transformer.

From the equivalent impedance, one can derive the voltage regulation (voltage drop under load conditions) and the short-circuit current that the transformer can supply. In summary, the open circuit test provides information about the core losses of the transformer, while the short circuit test helps determine the equivalent impedance and related parameters that are essential for assessing the transformer's performance under load conditions. These tests are crucial for designing, analyzing, and ensuring the efficient operation of transformers in various applications [21].

2.4 Inrush phenomenon

When an unloaded transformer is energized, it behaves like an inductor. In normal conditions, the flux Ø generated in the core lags behind the applied voltage by 90°, as illustrated in figure 2-9a. When the voltage wave v1 completes the first half-cycle, the full flux value of $Ø_m$is reached. Consequently, as the voltage rises from zero, the resulting flux has a negative maximum value.
This condition can only occur in a steady state or if the core is not magnetized and the transformer is energized at a voltage limit where the flux is zero. For such a case, the voltage and flux satisfy equation (2-1), provided core losses and primary winding resistance are ignored. Nonetheless, the correct value of magnetizing current required to generate the desired flux can be determined using the B-H curve of the transformer core [22].

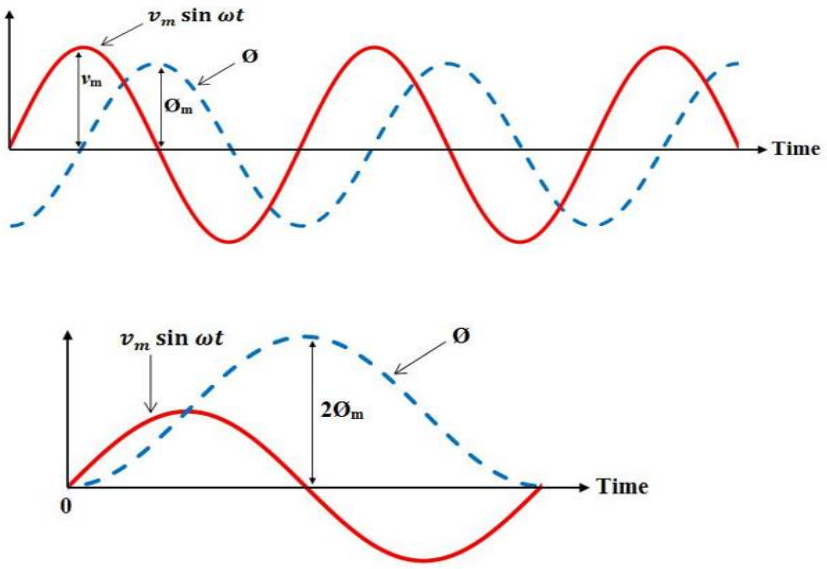

Figure 2- 9 (a) Voltage and flux. (b) Steady state on transformer [22].

Figure 2-9b have shown that, the half-cycle voltage wave, would produce a double of the maximum flux stable status $Ø=2Ø_m$. During energization of a transformer, a residual flux of $Ø_r$ in the same direction as the increasing flux (positive polarity) can lead to a significant increase in the required flux, $Ø=2Ø_m+Ø_r$, resulting in a very high current being drawn by the main winding from the source. This current is known as the magnetization inrush current, or simply inrush current. The magnitude of the inrush current is dependent on changes in the flow path and the transformer's time constant, which is typically in the range of a few milliseconds to seconds, depending on the size of the transformer. The maximum value of inrush current occurs when the required flux is equal to twice the maximum steady state flux, $Ø_m$, plus the residual flux, $Ø_r$, as shown in Figure 2-10. This is because the transformer core becomes supersaturated at this point, requiring an exciting current hundreds of times higher than the normal rated current. On the other hand, if the remanent flux in the core, $Ø_r$, is in negative value, the inrush current will be less than the maximum value, as the maximum required flux will be equal to twice the maximum steady state flux minus the residual flux, $Ø=2Ø_m- Ø_r$.

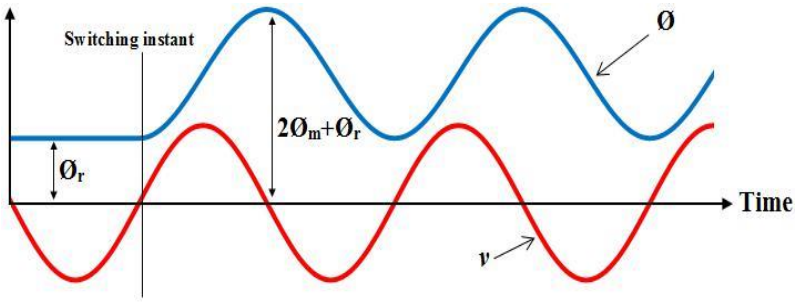

Figure 2- 10 Residual flux effect [22].

Typically, the transformer core becomes slightly saturated after reaching the maximum value of steady-state flux, $Ø_m$, which occurs around the saturation point (also known as the knee point). However, if a flux of twice the maximum steady-state flux $Ø_m$, or more, is applied along with residual flux $Ø_r$, the core may become supersaturated. This can result in an exciting current that is hundreds of times higher than the normal rated current. Figure 2-11-a illustrates how this current is formed, based on the B-H curve of the transformer core. The inrush current is dependent on the change in the path of the flux, as well as the time constant of the transformer. The time constant is typically just a few milliseconds for small transformers and several seconds for larger transformers. This inrush current is transient in nature, lasting just a few milliseconds or seconds, depending on the transformer's size, and then disappears. The typical waveform of inrush current is depicted in Figure 2-11-b.

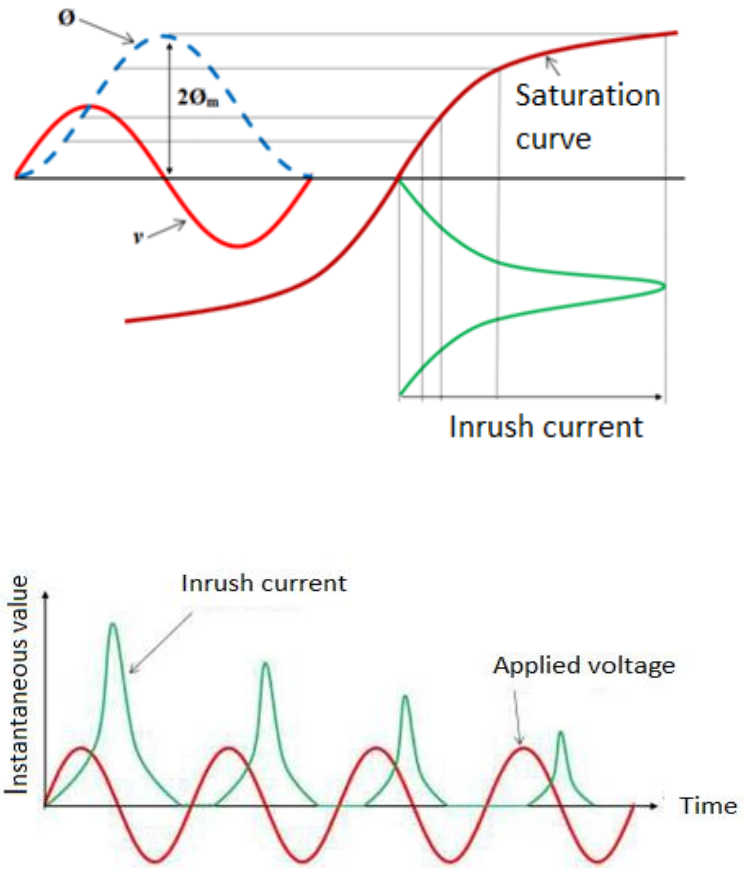

Figure 2- 11 (a) inrush current formation (b) waveform of inrush current [22].

2.4.1 Inrush current Properties

Equivalent to natural and defective currents, the inrush current has its own behavior and characteristics. The inrush present characteristics can be summarized as follows [23,24]:

1. Usually in the inrush current signal, the even, odd harmonics and DC offset occur.

2. In most cases, the waveform comprises either unipolar or bipolar cycles separated by intervals of time when the current is almost zero or very low. Dwell time and dead angle are terms used to describe this interval.

3. As a result of the large time constants involved, peak values of the unipolar waveform take a long time to decay, which is much slower than the decay of the DC component caused by fault currents.

4. Second harmonics are usually the highest in the inrush current signal.

2.4.2 Problems in transformer caused by inrush current

The inrush current can also be caused by any abrupt change in the magnetizing voltage of a transformer, in addition to its occurrence during transformer energization [25]. Inrush current is a transient phenomenon that is not typically classified as a fault due to its non-permanent and rapid nature. However, it can place significant stress on transformer windings, leading to protective relay breakdowns and circuit breaker tripping each time the inrush current occurs. Additionally, improper disconnections can result in the reduction of transformer lifespan and the development of internal faults due to winding insulation loss. Moreover, high inrush current can cause nuisance fault or interruption of breakers, leading to the need for oversized equipment. Furthermore, inrush current causes a current imbalance between the primary and secondary currents when the transformer is energized. This imbalance can impact relay functionality based on current balance, such as differential relays. As such, these relays must be blocked during inrush to prevent faulty operation. Inrush current occurs only when the transformer is turned on, resulting in unbalanced primary and secondary currents. To avoid inrush, the transformer must be turned off at the peak voltage, where the residual flux is negative. When the transformer is turned on again, the residual flux left in the core after the last shutdown is rebuilt from a negative value, and voltage waves complete positive half-cycles while flux stays below the core's knee point, eliminating the need for additional current. In practical terms, this scenario is not always possible, particularly in three-phase transformers [26].

2.5 Current transformer saturation problems

In terms of operation, the current transformer (CT) is the same as an ordinary transformer. These instrument transformers are designed to produce a secondary winding current proportional to the primary winding current to be measured. Through changing the turn ratio, CT can reduce high currents to a safer level and make them more convenient for both measuring instruments and protection devices to use.

Magnetic saturation is one of the most significant problems in power system protection transformers (CTs), particularly in differential protection transformers. In CT saturation, CT is unable to

reproduce the primary current or secondary current is not a replica of the primary current. CT core saturation caused by faults distorts the wave shape of CT output. In the event of CT saturation, protection relays response may be delayed or incorrectly operated. This problem can be solved by relay algorithms or by increasing the CT's core size to increase flux density and prevent saturation. CTs usually result in impractical, large sizes due to the requirements. Practically, there are limitations on space and costs, so saturation is inevitable. In general, current transformers operate in the linear region of their V-I curve at the lower part [27]. The purpose of this arrangement is to ensure that a transformer's core will not be susceptible to saturation during fault conditions; however, there is a possibility that CTs could be driven into saturation during severe fault conditions [28]. CTs are evaluated according to the specifications and behavior of ANSI/IEEE Standard C-57.13- 1993 under steady state and symmetrical fault conditions. There may be a high degree of DC offset and magnetic remanence in the core of a power system when a short-circuit fault occurs. Both elements cause CT cores to reach magnetic saturation [29]. By implementing simulation/modeling, we can gain a deeper understanding of CT performance under fault conditions.

2.6 Transformer failure

Transformers are essential and costly electrical components in power systems that are susceptible to faults due to inherent defects, similar to other electronic devices. A single failure in a transformer can cause a myriad of issues, including power loss, endangering individuals and the environment, as well as incurring high repair and replacement expenses.

2.6.1 General transformer failure

Transformer failures or defects are usually known as: [30, 31]

1. Winding defects triggered by a short circuit of inter-turn, phase-phase, phase-in-ground or turnover defects.
2. Core failures like core isolation failure or short-circuited laminations occur.
3. Fixed terminal faults like open lines or links.
4. On-load tap changer faults.
5. Tank flaws and other transformer components.
6. An irregular event including leakage, excess or over-tension occurs.
7. External defects that have been sustained or uncleared.

The approximate ratio of transform issues is shown in figure 2-12.

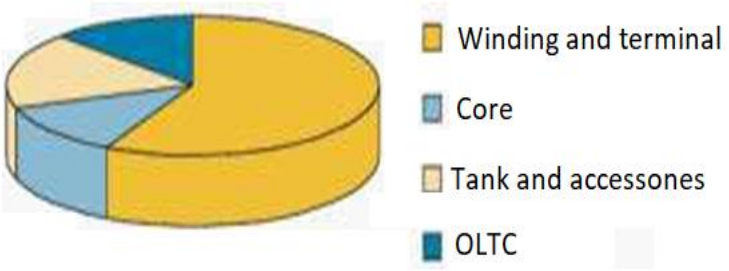

Figure 2- 12 Transformers faults statistics [30].

The potential fault occurrences in three-phase systems can be classified into several categories with varying probabilities [32]. Of these categories, single-phase (or line) to ground faults have the highest probability of occurrence. The fault types and their corresponding percentages can be found in Table 2-2.

Table 2- 2 Three phase system fault percentage

Fault type	**Single phase-to ground**	**Phase-to-phase-to ground**	**Phase-to-phase**	**Three-phase**
Percentage of fault occurrence	70%-80%	10%-17%	8%-10%	2%-3%

According to data from IEEE Standard [33-34], more than 50% of all potential faults in transformers are attributed to winding faults. Winding faults typically arise due to breakdown of insulation in the coil turn, which can be attributed to the impact of mechanical and electromagnetic forces on the windings.

2.6.2 External and internal faults

Transformers can suffer damage from winding stresses brought on by overheating, open circuits, and short circuits. While open circuits pose a relatively minor threat, transformer protection focuses primarily on short circuits, whether internal or external. External faults refer to faults occurring outside of the transformer-protected zone, while internal faults occur within it [35]. There are types of internal faults shown below:

2.5.2.1 Incipient faults

If these faults are not detected and addressed in a timely manner, there is a possibility that they may develop into more severe faults, such as single-phase faults to ground or three-phase faults.

2.5.2.2 Active faults

The main cause of such faults is thought to be the breakdown of insulation, resulting in a sudden stress on the winding. There are several factors that may cause insulation breakdown between winding coil turns or between core and windings:

- As a result of frequent exposure to high temperatures, insulation begins to degrade. Hot spots on windings will deteriorate more quickly and therefore have a shorter life expectancy than other parts.

- Oil that is to be used for insulation and cooling must be clean, dry, and free from water and other impurities. The oil can be contaminated by water or other impurities which can reduce the oil's dielectric strength and cause the oil to leak.

The occurrence of a flashover typically results in the breakdown of insulation between low voltage and ground. Active faults, such as phase-phase and phase-to-ground faults, arise when current flows through a conductor between two phases. Due to the sudden nature of these faults, a fast-response protective relay is necessary for their detection.

2.5.2.3 Earth faults

These types of faults are mostly present between earth and windings, for example, phase-to-ground or turn-ground faults, where windings touch the earth directly or by a flashover via the tank or the core to the earth.

2.5.2.4 Short Circuits

When phases are short-circuited, there is a significant current fault. The magnitude of this phenomenon is primarily determined by the leakage reactance and the impedance of the transformer.

2.5.2.5 Turn-turn faults

A transformer's turn-turn (interturn) fault is one of the most serious failures since it can significantly reduce its life expectancy. This is a minor internal fault, so it is very difficult to detect. According to IEEE Standard documents, basic protection requirements, such as sensitivity, selectivity, and speed, cannot provide protection to power transformers against minor internal faults. The difficulty in protecting transformers arises when only a few turns of the winding are short-circuited, resulting in minute changes in the magnitude of terminal currents. In order to produce a detectable change in terminal current, at least 10% of the turns in the transformer winding must be short-circuited, as specified by IEEE Standard C37.91-2000. Therefore, when only a few turns are short-circuited, an

undetectable increase in current occurs [36, 37]. This phenomenon occurs when two turns of the same winding come into direct contact due to insulation breakdown of the winding and conductors caused by vibration resulting from electromechanical forces generated by over currents and excessive external fault currents flowing through the winding. This fault leads to serious damage to the windings, including hot spots, oil heat, winding deformation, clamping, voltage sag, or power interruption. The resultant heat can cause the melting of winding turns, producing small solid copper particles that may contaminate the transformer's oil. Consequently, the damaged transformer must be removed for repair and cleaning of copper particles and soot, which requires a long outage [38,39].

2.5.2.6 Core faults

A short-circuit between laminations can cause the eddy currents to flow through them and cause overheating, which may lead to winding damaging.

2.5.2.7 Tank faults

Cooling and insulation are provided by oil-immersed transformers. In order to ensure oil quality, it should be checked for any contamination by water or other impurities such as solid copper particles, which can lower oil dielectric strength. As well as checking the oil level, the tank should be checked for leaks that may indicate poor insulation in the windings.

2.7 Relays

A relay serves the purpose of connecting to the power system in order to prevent faults or unwanted conditions within the intended protection area. The device can be either digital, analog, or numerical in nature. Alongside its primary function of protecting power system equipment, the relay plays a critical role in ensuring power service continuity and system stability [40, 41].

2.7.1 Classification of Relays

There are six types of relays according to their functions [42]:

1. **Protective Relay**. Determine whether there are faults in power lines and devices, or whether operating conditions are abnormal that could cause serious damage to the equipment used in power systems. This relay works by tripping one or more circuit breakers, and when the circuit breakers are tripped, the defective apparatus is disconnected from the system's network.
2. **Monitoring Relays.** Verification of operation of the power system or of protective devices, for example whether or not a protective relay operates during faults, etc.
3. **Reclosing Relays**. Once the protective relays issue trip signals, determine the sequence in which the circuit breakers should be closed.

4. **Regulating Relays.** When pre-set parameters of operation are exceeded, it will be activated.
5. **Auxiliary Relays.** It is basically support or supplement for other relays. This type of relay consists of a timer, an isolating relay, and a trip relay.
6. **Synchronizing Relays.** A guarantee that an appropriate interconnection condition exists between two parts of the power system.

As this study focuses on transformer protection, only protective relays will be considered.

2.7.2 Criteria of relays design

A protective relay should meet four common criteria in order to be effective. All these criteria cannot be met simultaneously, unfortunately. Therefore, these criteria need to be balanced or compromised. Compromises must be evaluated based on comparative risks [40].

1. Reliability

The concepts of security and dependability are integral to the overall reliability of relays. The security of a relay relates to its ability to avoid incorrect operations in response to any faults, while dependability refers to its capacity to properly operate in response to all faults. In practice, there can be a trade-off between these two aspects of reliability. In certain situations, improving dependability may require a reduction in security, and vice versa.

As such, an optimal balance must be struck between these two aspects. Currently, modern relays offer a high degree of reliability, which seeks to find an acceptable compromise between security and dependability. While dependability can be tested relatively easily through simulation or practical tests, verifying security is often more challenging, necessitating testing in the environment and the system being protected.

2. Speed of Operations

The speed of relay operation pertains to the rapidity with which it is able to detect faults. Swift relay response leads to a shorter fault duration, thereby reducing the damage and system instability caused by faults.

However, faster relay operation increases the likelihood of erroneous tripping. Hence, the speed of relay operation must be evaluated based on the likelihood of unwanted tripping. Employing time as a criterion is the most efficient method for distinguishing between actual and spurious issues.

3. Simplicity

It is possible to achieve all protection goals with minimal equipment and circuit breakers. However, it is important to ensure that the performance of the relay is not compromised. Therefore, it is recommended that the relay be designed with simplicity in mind.

2.8 Transformers protective relays

According to [42], it is necessary to protect transformers for many reasons as shown blow.

1. Identify the faulty device from other parts of the network infrastructure and then isolate it.
2. Keep the machine safe.
3. Minimize equipment damage.
4. Rising fire risk, which damages the adjacent equipment as well.
5. Reduce workers danger.

The following factors influence the safety of the transformer:

6. The cost of maintenance and repair.
7. Cost of outage.
8. Effect on other system components.
9. Damage risk to the adjacent machinery.

Transformer faults, such as short circuits, can occur due to internal electrical faults. Among these faults, the most prevalent one is the phase to ground fault, also known as turn to ground fault. Although less frequent, interturn faults or turn to turn faults can also occur. Interturn faults start as minor faults, and their detection and clearance may take longer. However, if left unaddressed, interturn faults can escalate into more severe faults, such as turn-ground faults. To protect transformers from internal faults, various protective measures are employed. These include fuses, overcurrent protection, pressure protection, and differential protection. These protective measures are designed to detect and isolate faults as quickly as possible to prevent further damage to the transformer [39, 43, 44].

2.8.1 Sudden pressure protection

Turn-turn faults are commonly caused by insulation breakdown in transformer windings, leading to the generation of heat in the winding turns as previously discussed in Section 2.5.2. To detect such faults, protection systems such as the Buchholz relay and rate of pressure rise relay can be employed, as the resultant heat will decompose the oil and release gas. However, these relays are not entirely satisfactory, particularly in detecting incipient faults. It is important to note that changes in pressure or static pressure due to normal transformer operation do not activate relays that rely on sudden pressure changes. Instead, these relays only operate when the rate at which gas rises in the transformer surpasses a pre-established threshold [40].

2.8.2 Protection of over current

The justification for utilizing differential protection for transformer protection diminishes when over current protection is employed. Overcurrent relays can supplement differential relays in addressing through faults. However, overcurrent relays are often vulnerable to mis-operation due to current transformer (CT) saturation or inrush current. The relay operates by tripping the circuit breaker when the input current magnitude exceeds a certain threshold level. Unless the input current surpasses the pickup current, the relay's contacts remain open. If differential protection is not a viable option, over-current protection becomes the primary method of protection. Nevertheless, when differential protection is employed, over-current protection functions as a backup measure. Fuse protection is the preferred protection approach for transformers under 10 MVA, while differential protection and over-current protection are the primary protection methods for transformers above 10 MVA, according to sources such as [45, 46].

2.8.3 Differential protection of transformers

The percentage differential protection method, which is a simplified version of basic differential protection, has been widely employed for protecting transformers over the years. As depicted in figure 2-13, differential protection involves a straightforward diagram. Essentially, this method mandates that the measured currents I_1 and I_2, acquired by current transformers (CTs), that join and leave the element requiring protection, are equal in normal conditions, i.e., $I_1=I_2$ or $I_1-I_2 = 0$. Otherwise, the element is deemed to have a fault. A relay will be triggered if this difference does not equal zero (ideally) or is not very small, referred to as the operating or differential current (I_{OP}) [47 - 48- 49].

$$I_{op} = |I_1 + I_2| \qquad (2-16)$$

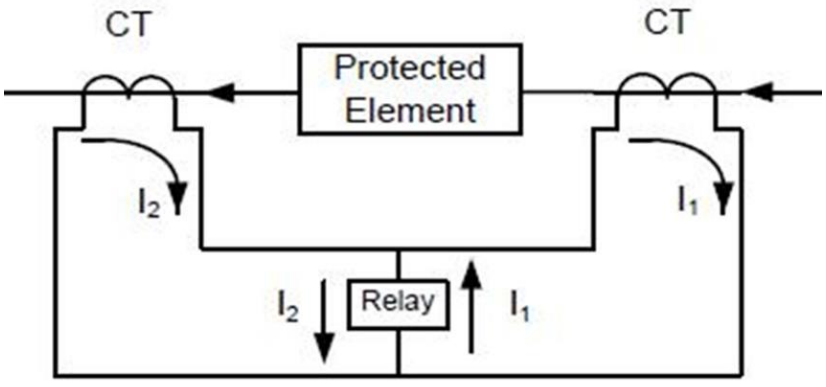

Figure 2- 13 Differential protection principals [47].

Significant differences between two currents can also occur under normal conditions when there is stray capacitance in the protected unit, non-identical current transformers, or saturation in the current transformer due to an external fault. To address this issue, a modification has been made to the basic differential protection concept. Figure 2-14 illustrates the inclusion of a restraining coil, which generates a current known as the restraint current (I_{RT}). This current can be calculated using the following equation:

$$I_{RT} = \frac{I_1 - I_2}{2} \qquad (2-17)$$

These three equations are used for the calculation:

$$I_{RT} = kc * |I_1 - I_2| \qquad (2-18)$$

$$I_{RT} = kc * (|I_1| - |I_2|) \qquad (2-19)$$

While kc Compensation factor

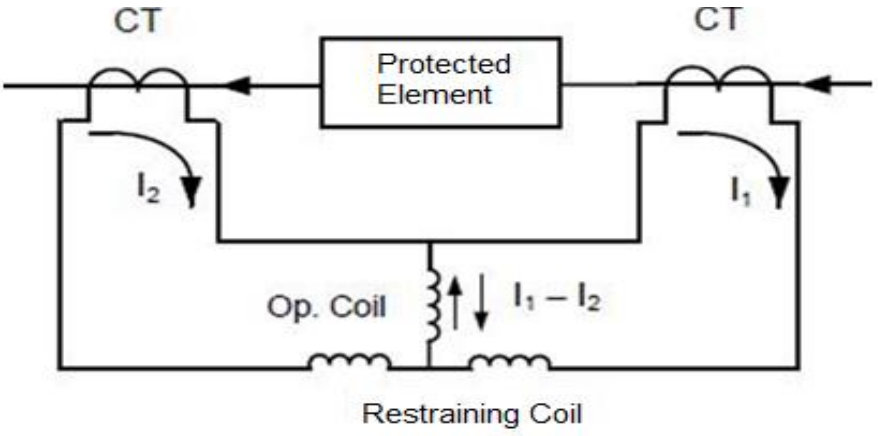

Figure 2- 14 Differential protection modified [47].

When current flows through a coil, it generates a torque that is proportional to the current. In a differential protection relay, the operating current (I_{OP}) passes through the operating coil, creating an operating torque that is opposite to the restraining torque generated by the restraint current (I_{RT}) in the restraining coil. The relay will only operate if the operating torque is greater than the restraining torque, which means $I_{OP} > I_{RT}$. However, if there is a small difference between I_1 and I_2, for reasons mentioned earlier, the relay may not operate. To address this issue, a modification has been added to the basic differential protection relay, resulting in a percentage differential relay or biased differential relay. The value of the difference between I_{OP} and I_{RT} that triggers the relay to operate or not depends on the slope, which can be set at 0%, 20%, or 40%. These slopes determine the range of difference between the two currents that is needed to either operate or restrain the relay,

as shown in figure 2-15. These percentage ratios consider the error ratio caused by non-identical values of I_1 and I_2.

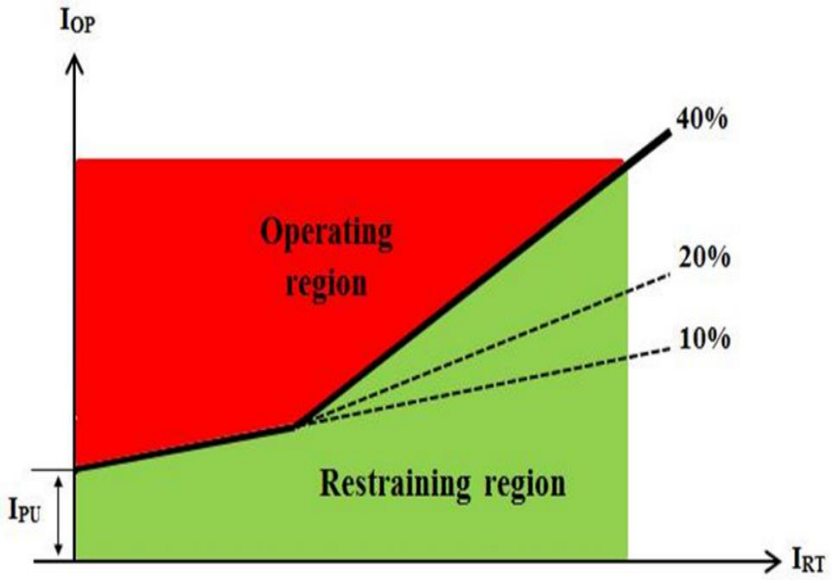

Figure 2- 15 Various slopes to determine the relay field [50].

For the relay to work, the I_{OP} value must exceed one of these ratios of the I_{RT} value. To begin to operate, the following conditions must be met:

$$I_{OP} > I_{RT} \qquad (2-20)$$

When the slope is 20%, it has been observed that the overcurrent element pickup (I_{OP}) is greater than the overcurrent element reach (I_{RT}) by more than 20%. This is attributed to the characteristics of the slope, which make external faults more secure with current transformer (CT) saturation. In Figure 2-15, the differential current I_{OP} does not initiate from a zero value but rather from a fixed value $I_{PU,}$ this value is called a minimum pickup current [50].

CHAPTER 3

Literature review

3.1 Introduction

To safeguard power transformers against relay failures, it is imperative to consider all defensive approaches during inrush current conditions. One of the major issues in transformer protection is the accurate discrimination between current inrush and internal faults. Therefore, this literature review assesses the different approaches chosen to discriminate between inrush current and internal failures in transformer protection. The review begins by introducing the concept of the approach to difference defense and explaining the current trends in transformer protection. It also discusses the conventional differential protection system that relied on second harmonic restraint. However, there are several limitations associated with the conventional differential protection scheme, which has prompted the development of various strategies to address these challenges. This literature review discusses several proposed strategies for overcoming the limitations of the conventional differential protection system. These include methods such as wavelet transform-based differential protection, negative-sequence current-based differential protection, high-impedance differential protection, and transient-based differential protection, among others. The advantages and limitations of each approach are evaluated, highlighting their respective contributions to the field of transformer protection.

In summary, this literature review provides a comprehensive overview of the various approaches used in transformer protection to discriminate between inrush current and internal faults. By considering the strengths and weaknesses of each approach, power system engineers can select the most appropriate method for their specific application to ensure the reliable and effective protection of transformers.

Many strategies for dealing with limitations have been proposed, several are listed in this literature review such as:

3.1.1 Per phase method

The per phase method is a well-established approach for mitigating harmonics in power systems. This method uses a criterion that applies to each phase independently to limit the impact of harmonics [51]. Since each phase has a different amount of residual flux, the magnitude of the 2nd harmonic will vary from phase to phase. Therefore, a threshold value is used in this method to

determine whether the 2nd harmonic content is low enough for differential protection to operate. While this method is very reliable, it may not be entirely secure, as the differential protection may trip if the 2nd harmonic content is low in one phase during energizing. Nonetheless, the per-phase method remains a valuable tool for restraining harmonics in power systems [52].

As a result of the fact that each phase has a different residual flux and has been energized at a different angle, each phase will have a different level of harmonics. In the case of a specific phase, when the second harmonic ratio exceeds a preset level, the percent differential operation will be blocked in that phase. In some cases, when the transformer is energized, the second harmonic ratio for each phase may be very low at the time of energization. Three-phase transformers may trip if there is a small ratio of the second harmonic in a phase during differential operation [53].

3.1.2 Cross-blocking method

The cross-blocking method is an alternative approach to the Per-Phase method for restraining harmonics in a system. One notable difference is that the signal restrained from one phase will prevent differential operation in all other phases. Although this technique is effective for symmetrical faults, it may cause the differential protection to trip for unsymmetrical faults. However, reliability may be compromised, particularly during energization when insulation faces high mechanical stress and inrush current is several times greater than the rated current [54].

3.1.3 Percent average blocking method

Harmonic ratio is the sum of the second harmonic ratio of three phases for the typical blocking form.

$$\text{2nd harmonic ratio } = \frac{1}{3}\left(\frac{|I_{op2ndA}|}{|I_{opA}|} + \frac{|I_{op2ndB}|}{|I_{opB}|} + \frac{|I_{op2ndC}|}{|I_{opC}|}\right) \qquad (3-1)$$

Where I_{op2ndA}, I_{op2ndB} and I_{op2ndC} are represented the second harmonic operation current for phase A, B and C respectively. The harmonic ratio method is a technique for improving the security of differential protection by calculating the sum of the second harmonic ratios of the three phases. This percentage average blocking method is considered more secure than cross-blocking methods. However, it may fail in cases where the 2nd harmonic ratio to two phases is high during a single-phase fault, which can impact the dependability of the differential protection system. During energization, a true single-phase fault may restrain differential operation if there is a high harmonic ratio in the remaining phases [55].

3.1.4 Harmonic sharing method

The harmonic summing technique combines the magnitudes of all 2nd harmonics from three phases into a single harmonic signal and calculates a 2nd harmonic ratio for each phase based on the

resultant value. A low harmonic ratio during energization indicates an internal fault, while a high harmonic ratio protects against mis operation. This technique improves the reliability of differential protection by preventing false tripping during normal conditions. This means that the tripping of a transformer in three phases takes place during an internal malfunction. With this harmonic summing technique [56], protection against mismanagement has improved.

3.1.5 Performance analysis of traditional and improved transformer differential protective relays

In [57] developed algorithms that combine both harmonic restraint and harmonic blocking methods. However, the effectiveness of the algorithm and data collection depend on the choice of fault detection coefficients. Additionally, the balance between the two methods is not optimal, as harmonic restraint is more secure for inrush currents, while harmonic blocking is better for internal faults and less time-consuming.

3.1.6 New magnetizing Inrush restraining algorithm for power transformer protection

This method extends the traditional method of second harmonic restraint. It changed the way that second harmonic ratios were traditionally defined in order to be used as restraint signals, to a complex second harmonic ratio. According to widely accepted theory, the second harmonic ratio can be calculated by dividing the magnitude of the second harmonic by the magnitude of the fundamental component of the differential current. In this case, the phase angle of the second harmonic was compared with the phase angle of the fundamental frequency component. Due to the fact that it also observed a difference in this ratio during internal faults and inrush conditions, it improved the relays' performance for internal faults [58].

3.1.7 A New digital dynamic algorithm for detection of magnetizing inrush current in transformers

This implemented a new classification scheme between internal cases and inrush. This method is based, rather than one conventional second harmonic criterion, on three factors to decide which of these instances is occurring, the conventional second harmonic material, time constant decay of the DC offset and a ratio determined by dividing the magnitude of the basic component by the magnitude of the first maximum signal present. The current signal level was estimated in a very short time using a discrete time-dynamic filter. The proposed algorithm has been used to extract the functionality of those current signals with digital samples of distorted current waveforms during inrush and internal failures. Then the three variables were tested by a specific panel of experts to distinguish between cases. The ratio between the basic variable and first peak level was based on estimates, i.e. it cannot be trusted as a difference criterion, as well as the second harmonic is used as one of the criterion in this technique [59].

3.1.8 Advances in the design of differential protection for power transformers

A standard technique for separating inrush from an internal fault is the dwell time approach. One of the properties of the inrush waveform is that in the second half-cycle its form is flat with a current value close to zero. This discontinuity duration is called dead angle or the interval of dwell t_A in Figure 3-1. Methods were suggested based on this feature. The theory proposed by them suggests that the duration of inrush current t_A is greater than the duration of internal defect t_B, as depicted in Figure 3-1. For the determination of the length of the stay the positive and negative thresholds are contrasted with the differential current.

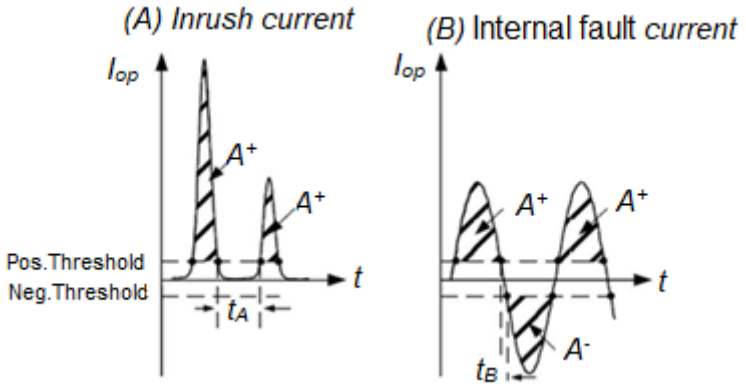

Figure 3- 1 (A) Inrush current and (B) internal fault During Dwell time [60].

As a reference time, a one-quarter interval is used to be compared with the time taken. When the differential current experiences an inrush, the relay is blocked. However, if the inrush time is shorter than the threshold time, it indicates an internal fault, and the relay trips. Conversely, if the differential current persists for longer than the threshold time, it suggests a fault external to the protected zone [60].

3.1.9 Studies for identification of the inrush based on improved correlation algorithm

The dead angle (dwell-time) approach was employed in this study as a means of restraint. By examining the flat shape that occurs in the second half cycle of inrush, the corresponding angle of the differential current in the non-current zone can be measured over one cycle (360°) to identify the presence of inrush. As the angle and its magnitude are significantly greater during inrush than in the case of an internal fault, this method can effectively distinguish between the two. However, during severe inrush current, the threshold of corresponding angle = 60° may lead to maloperations due to the influence of CT saturation on the current signal shape. Furthermore, the reverse charge of the current transformer can eliminate the dead angle, rendering this approach ineffective. Additionally, the operation time of this method is prolonged, taking up to one and a half cycles in certain fault cases [61].

3.1.10 A practical winding fault detector for power transformers

To detect internal faults, the researchers utilized the electromagnetic equations of a transformer's equivalent circuit, s. The transformer was represented using R and L matrices from standard open and short circuit tests. The equations allowed for the calculation of two voltage equations for each phase of the transformer, resulting in six voltage equations for a three-phase transformer. The researchers assumed that in normal operation, inrush, and external faults, the calculated voltages using these equations would match the measured voltages.

Tripping detector voltage = Voltage (measured) – Voltage (calculated by equations) (3-2)

Therefore, they created six tripping detectors that compared the measured voltage to the calculated voltage using the equations. In the event of an internal fault, the tripping detectors would detect a mismatch between the calculated and measured voltages. The parameters required for the equations were either experimentally obtained or derived using the "BCTRAN" routine in the Electro Magnetic Transient Program (EMTP). However, obtaining the circulating currents in Delta winding connections, which were necessary for the equations, was difficult in practice [62].

3.1.11 Digital relaying algorithm for detecting transformer winding faults

Here using only linear components in electro-magnetic equations. This has also used Delta winding currents. Furthermore, if the transformer windings are joined to the delta, each winding has a circulating current for its own components and so the factors in the equation that achieve these currents are more complicated, as these factors would include all the reference parameters for the reference phase as well as all other phases. Transformer equivalent circuits should be calculated accurately in case of core saturation in electro-magnetic equations. But this precision can hardly be proven [63].

3.1.12 Cross-correlation method

The authors proposed a technique to differentiate between inrush current and internal fault by exploiting the unique features of inrush current, such as the asymmetry of two successive half-cycles and the presence of a dead angle. These features are extracted by utilizing the short-time correlation function. However, this method requires at least one cycle of operation time. As previously mentioned in section 3.1(9), the dead angle may be eliminated due to the current transformer's reverse charge. [64].

3.1.13 Method using fuzzy logic

The inrush current and default current can be discriminated against using this fuzzy logic scheme to ensure the likely imperfect activity during magnetization of the differential relay. The percentage of the second harmonic material is greater than the amount of the working theory of this scheme.

Second harmonic content and overall harmonic deviations are less than 5 percent when the error occurs. The threshold value must therefore take into account 3 factors. Those are full distortion of harmony, second harmonic ratio, and second harmonic scale. Deciding fault factor in one phase has no impact on decision-making variables in another phase. The risk of an unwanted process of differential security is therefore reduced. These features improve the stability, reliability and sophistication of this technology [65-66].

3.1.14 Method based on ANN

Artificial Neural Networks (ANNs) have gained significant popularity in recent years as a means of improving the security of power systems, making them one of the most commonly utilized types of Artificial Intelligence (AI) techniques [67]. ANNs have been applied in several areas including transformer protection, design recognition, image processing, and power quality analysis. While some studies have focused on testing the feasibility of the ANN approach, others have integrated additional concepts with ANN, such as Principal Component Analysis (PCA) and wave-form symmetrical components. Despite the promising results obtained with ANNs, there are some key drawbacks to this approach. One issue is that the parameters of the neural network cannot be defined using specific rules, and the learning process for identifying patterns can be time-consuming. In large power systems, the topology is subject to change, and the vast number of neural networks that serve the electricity system may not reflect these changes. Furthermore, the ANN approach may not be suitable for protecting various power systems since it cannot be tailored to the specific characteristics of a particular power system. Finally, training ANNs to detect transformer faults typically requires a large dataset of fault cases, which can be computationally intensive and require significant memory, necessitating a large number of devices and processors for the system [68, 69].

3.1.15 Wavelet method

The use of Wavelet Transform (WT) in differentiating internal faults from other issues, such as inrush and external faults, has been proposed in previous studies [70-71]. Some methods have been suggested for transformer protection based on a combination of WT and Artificial Neural Networks (ANNs). WT is a statistical technique used to analyze a signal with a time-varying frequency. It extends the Fourier analysis by decomposing the signal into low- and high-frequency bands, isolating the signal's most significant component. However, the wavelet coefficients in this frequency band are not restricted to specific harmonics or fundamental elements of the desired signal, and other signals in the same frequency band are also affected. Furthermore, WT coefficients are highly sensitive to noise, especially in high detail. Therefore, in the presence of noise, the accuracy of the extracted signal is compromised. WT also requires a relatively long data duration, and in some methods, at least 1⁄4 cycle of a current waveform is needed (5ms in a 50 Hz system)

after instantaneous failures in cases of internal and external faults. Decision-making algorithms based on trial-and-error processes can be employed with data obtained from multiple trials [72,73].

3.1.16 Instant inductance equivalent (EII) based method

The Instant Inductance Equivalent (EII) is commonly used to differentiate between the presence of inrush current and internal faults. Experimental studies have been conducted to evaluate the effectiveness of this approach, revealing that the EII value for inrush current varies significantly, while the EII value for faulty conditions remains relatively constant [74]. Under normal operating conditions, when the iron core is not saturated and the magnetizing current is low, the Instantaneous magnetizing Inductance (IMI) remains constant, as it operates within the linear field of the magnetizing characteristic. However, when an internal fault occurs, there is a sudden change in the IMI, as depicted in Figure 3-2, transitioning from a non-saturation to a saturation field.

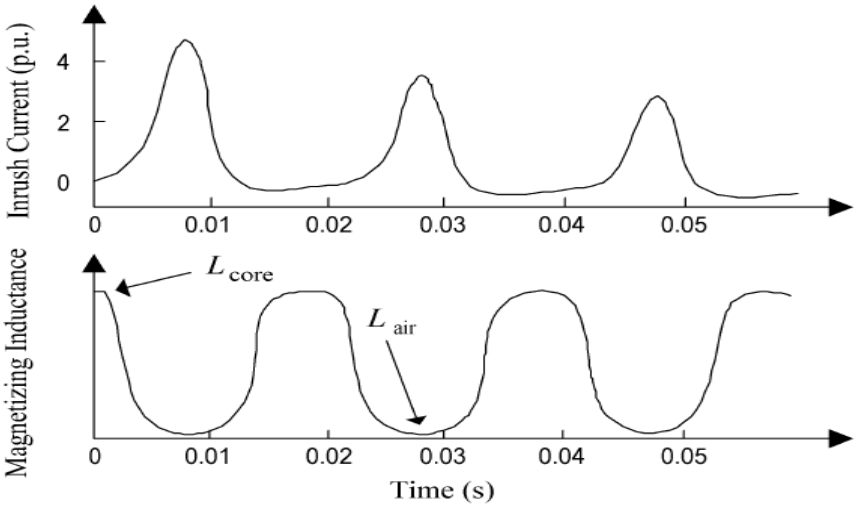

Figure 3- 2 Diagram reveals the difference in the waveform of instantaneous magnetization [74].

The determination of EII value can be carried out through two methods: indirect and direct. In the indirect method, the variation of IMI value with the fundamental frequency of 50Hz is observed when the transformer is supplied with a 50Hz source. Under normal operating conditions and internal faults, the fundamental frequency variable of IMI varies from zero to the value of EII, while inrush current does not exhibit this behavior. This criterion serves to differentiate between inrush current, regular transformer operation, and internal faults. If the magnitude of the fundamental frequency variable in EII is above the threshold or pickup value, the relay trip is blocked for inrush current, and the relay signal is sent if it exceeds the threshold value. Direct method: the RMS expression that indicates the EII variation is given as

$$\Delta L^{-}{}_{K} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}[L_k\,(i) - L^{-}{}_{K}]^2} \qquad (3-3)$$

$$L^{-}_{K} = \frac{1}{N}\sum_{i=1}^{N} L_k\,(i) \qquad (3-4)$$

In the second method of analysis, N represents the number of measurements taken in a single frequency cycle, and ΔL^{-}_{K} is a parameter used to establish a threshold for distinguishing between inrush current and internal faults. If the measured value exceeds the threshold, the relay operation is blocked as it is considered an inrush current. Conversely, if the measured value falls below the threshold, the relay operation is triggered as it is assumed to be an internal fault. This method is effective for both time and frequency domain analysis. On the other hand, the Direct method of analysis works only in the frequency domain [75].

3.1.17 Instantaneous inductance technique

Primary side transformer differential inductance is achieved in this approach by using the current signal and voltage. From every phase of the transformer, the difference in inductance is determined. For the comparison between the value of threshold and the value of differential inductance an algorithm is created. Where this value crosses the threshold value, then this is an internal fault. This technique works even when there are differences in taping of the transformer, fault resistance occurs, and saturation of the current transformer (CT) takes place [76]. The operating time for this method is also much lower (5ms, below 1/4th of the power frequency cycle).

3.1.18 Sinusoidal proximity factor method

The Sinusoidal Proximity Factor (SPF) method is an important technique for distinguishing between internal fault and inrush current in power systems. This is achieved by separating the internal fault current from the inrush current, based on the wave shape of the current signal. The waveform is assumed to be sinusoidal, with amplitude A, angular frequency ω, and phase angle θ. By standardizing and multiplying the signal, a pure wave shape can be obtained, but in practice, harmonics are present in the signal. The difference between the real signal and the pure wave shape is quantified by the SPF, which is given by the absolute difference between the standardized signal and the pure wave shape. The SPF value is close to zero for internal faults, but there is a significant difference for inrush currents. When the SPF value approaches or exceeds 0.5, the relay is triggered and the current is identified as an internal fault, rather than inrush current. This method is most effective for detecting small internal fault currents. By using the SPF method, it is possible to differentiate between different types of faults and accurately detect and isolate faults in power systems [77,78].

3.1.19 Discrimination of transformer inrush currents and internal fault currents using extended kalman filter algorithm (EKF)

In this method, an Extended Kalman Filter (EKF) algorithm is employed to accurately classify internal fault currents and inrush currents in transformers. Specifically, when the transformer is energized under normal operating conditions, the EKF algorithm is used to estimate the primary side winding current. Consequently, the Absolute Residual Signal (ARS) value, which represents the difference between the estimated current and the measured current, is zero. However, for both internal faults and inrush currents, the ARS value is non-zero. Thus, the EKF algorithm is utilized to distinguish between these two phenomena by applying a threshold level equivalent to the ARS value. Nonetheless, some fault cases may be mistaken as inrush currents since the ARS value may not have surpassed the threshold at that time. Nevertheless, it is noteworthy that the operating point is generally low, with a maximum duration of 116 milliseconds. As the severity of the fault increases, the detection time decreases since the necessary residual threshold is reached earlier. Consequently, the EKF algorithm is an effective method for accurately detecting and classifying internal fault and inrush currents in transformers, and it has proven to be a valuable tool in power system protection [79].

3.1.20 Detection and classification of internal faults in power transformers using tree based classifiers

The purpose of this study is to propose an effective method for detecting and classifying internal faults in a transformer based on Decision Trees (DT). The differential currents in phases a, b, and c belonging to the time domain, and the frequency domain have been used to extract several features from their differential currents. Three of these features are selected as evidence that the internal faults of the transformer can be distinguished from the magnetizing inrush, while another three are selected as evidence that the faults in the transformer primary and secondary can be classified. To identifying fault types, DT, Random Forest (RF), and Gradient Boost (GB) classifiers were used. This method does not deal with over-excitation, turn-to-turn faults, and inter-winding faults [80].

3.1.21 Numerical Differential Protection Algorithm for Power Transformers

The authors proposed an algorithm to identify inrush current based on the assumption that inrush current has a continuously decreasing waveform, while the peak of an internal fault is higher than the previous peak. The algorithm subtracts the RMS value of the differential current in the previous cycle from that of the current cycle. If the result is positive, it indicates an internal fault, while a negative value indicates inrush. However, this approach has limitations as it requires information from two cycles to distinguish between inrush and internal fault, and it may fail to detect an internal

fault during transformer energization. Additionally, it cannot differentiate between internal and external faults. [81].

3.1.22 A new technique for power transformer protection based on transient components

There is also a new way to differentiate between external fault and internal fault. It consists of transient components of three phases derived from both the transformer's main and secondary sides. Such components are transformed by Clark's transformation matrix into modal transient components, and a fault detection equation is then determined in the sense of new modal transition components. This equation's output sign is known as a detector. Of internal failures and external failures, it is negative. It takes more than a loop, however, to detect the case [82].

CHAPTER 4

Measurements and Experiment Setup

4.1 Transformer design

A transformer has been built at Wolfson Centre for Magnetics, School of Engineering, Cardiff University's laboratories with 4.1 cm high cores and 11 cm wide cores, as shown in Figure 4-1. A laminated core, multiple-limb, three-phase research transformer rated for 20kVA was used to test this appliance's performance. The transformer's core material was grain-oriented 3% silicon electrical steel. The bolts were tightened with a torque of 1 N.m to maintain a uniform clamping pressure on the transformer core.

Figure 4- 1 Transformer under study

4.1.1 Transformer core test

Using equation (4-1) to determine the flux density by substituting the desired value of flux density and finding the voltage corresponding to that value. In this step, a coil of one turn was used to achieve the desired flux density.

$$\text{Erms} = 4.44\text{fN1Bmax.A} \qquad (4-1)$$

In this equation, f is the presented frequency, N is the number of turns, B_{max} is the maximum magnetic field, and A is the core area.

1. In this experiment, the flux density (B) was chosen as 1.3 Tesla at 50 Hz for the transformer, and this value was selected based on Heat Generation because the transformer cores experience hysteresis and eddy current losses, which generate heat. Operating the core at a flux density of 1.3 T helps manage these losses and keeps the temperature rise within acceptable limits. Therefore, it can determine the E_{rms} by:
   E_{rms} = 4.44*50*1*1.3*0.00451=1.3 V (rms)
2. The core limb is wrapped in a single-coil, and the two terminal ends are attached to a multimeter.
3. The voltage supplier is connected to the primary winding, which is gradually increased until the multimeter displays the desired voltage across the turn as calculated. This indicates that the supplied voltage is now sufficient to achieve the specified flux density. Table 4-1 displays the corresponding voltages for the various flux densities tested on the transformer core.

Table 4- 1 Flux density corresponding to supplied voltage.

Flux density B (Tesla)	Phase voltage phase A (Vrms)	Phase voltage phase B (Vrms)	Phase voltage phase C (Vrms)
0.53	30.33	31.23	28.87
1.05	60.05	59.02	59.75
1.3	75.09	76.09	75.73
1.5	86.02	86.18	85.74
1.7	98.16	98.34	97.61

By using the fluxmeter device (Lake Shore 480) can also be determined the flux density (B) as shown in appendix A.

A comparison was conducted between the computed flux density and the flux density obtained through measurements using a fluxmeter device. In Figure 4-2, the graph illustrates the computed and measured flux densities corresponding to a range of applied voltages, varying from 0 V to 69 V.

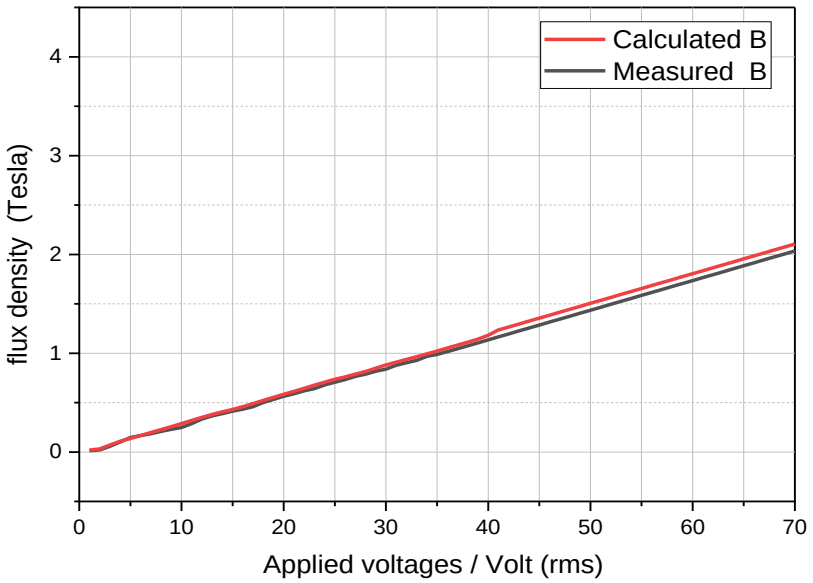

Figure 4- 2 Measured and calculated flux density (β).

The graphs are nearly overlapping. They are clearly similar. As a result, equation (4-1) has been established and may be used to determine the flux density in the transformer core.

4.1.2 Winding taps distribution

Figure 4-4 displays the taps and winding arrangements around the transformer core. Each phase comprises 120 turns, with 60 turns on both the primary (red) and secondary (black) sides. Additionally, the winding includes 25 taps that are strategically positioned: five taps at the start, middle, and end of the winding, with the remaining taps placed at intervals of five turns. These taps were intentionally installed to generate problems for transformer analysis.

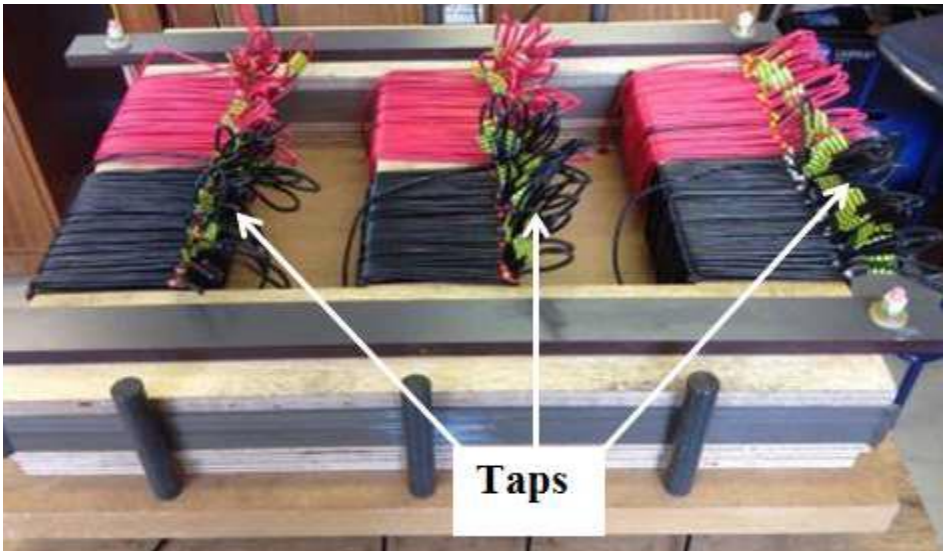

Figure 4- 3 Winding turns and taps.

4.2 Faults generation

Internal faults were intentionally created by introducing a wire connection either between taps (interturn fault) or between a tap and ground (turn-ground fault). Conversely, external faults were simulated to occur outside of the transformer. To prevent the risk of wires overheating and causing damage due to the creation of faults and safety considerations within the laboratory environment and for the sake of computational simplicity, a phase voltage of 60V was employed, and both the primary and secondary windings were designed with 60 turns each, resulting in a turn ratio of 1. It's worth noting that, for the purpose of this study, the presence of oil or a tank in the transformer setup was intentionally omitted. This omission was made because it did not influence the results, and the proposed detection technique was proven to effectively identify internal faults within the transformer, irrespective of the presence of a tank, oil, or insulation between the windings.

4.2.1 Turn-turn fault generation

Generating an interturn fault is possible, but it is unsafe as the turns to be shorted (N_x) can be damaged due to the very high circulating fault current (I_f) in the loop, as illustrated in Figures 4-5. A low wire resistance results in a high current, so a fault resistor (R_f) of 0.1Ω was added to the loop to reduce the current value flowing through the wires and protect the winding turns from damage caused by the high current. Each turn should have a voltage of 1V across it if the turns are identical. Therefore, three turns with a voltage of 2.95V and a resistance of 0.1 Ω were added to a wire resistance of approximately 0.018 Ω, which was calculated by using equation (4-2), as the turns were almost equal and manually wrapped around the core limb.

$$R = \frac{\rho \times l}{A} \qquad (4-2)$$

The wire's resistance (R) in ohms (Ω) is given by the formula R = ρ*l/A, where ρ is the material's resistivity in Ohmmeters (Ω.m), l is the wire's length in meters (m), and A is the wire's cross-sectional area in square meters (m^2). Therefore, the total resistance was calculated by adding the ware resistance to fault resistance as shown below:

$$R_{total} = R_f + R = 0.1 + 0.018 = 0.118\,\Omega \qquad (4-3)$$

In this scenario, the circulating current was 2.95/0.118 = 25 Amps, which was within the manufacturing current limit of the employed winding wire of 30 Amps. To intentionally trigger a fault at a predetermined time, the operation of opening and closing the switch (SW) was meticulously controlled through a LabVIEW program. This LabVIEW program generated a pulse

signal through a data acquisition card, subsequently activating an electro-mechanical relay. This relay was strategically connected in series with the protective resistor (Rf) within the short circuit setup. This relay functioned as a switch (SW), effectively establishing a short-circuit condition between turns. As shown in Figure 4-5, this relay was utilized as a switch (SW) to create a short-circuit between the turns.

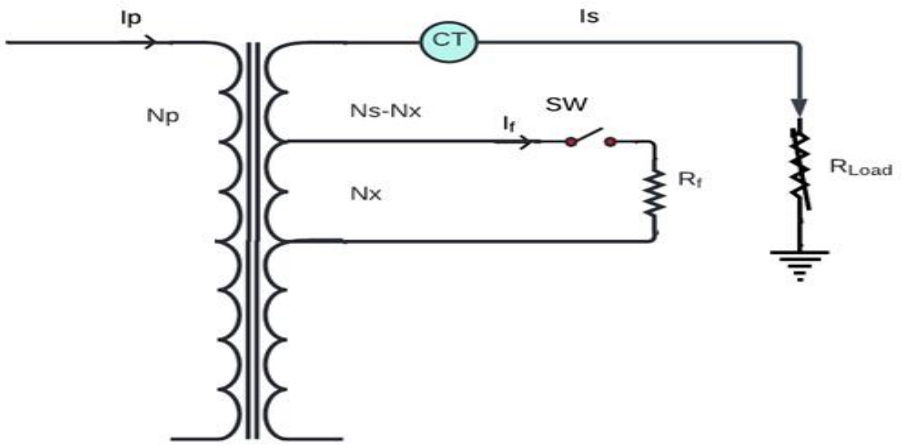

Figure 4- 4 Interturn fault representation.

The relay was not ideal since it took a few milliseconds to close after being triggered, and the designer specified this time. Therefore, specifying the exact timing of the relay's switch-on after triggering it was problematic, but the manufacturer's maximum operational period is 15ms, which means it could be switched on in less than 15ms. This flaw was created in the laboratory, as shown in Figure 4-6, to represent the interturn fault by adding R_f to the experimental circuit. In a real interturn fault (without the protection resistor Rf), the wire resistance is as low as 0.018 ohms, which means the actual current that circulates in the loop is 2.95/0.018 = 163.9 amps, which is quite high and can burn out these turns. This is precisely what happens when this issue occurs.

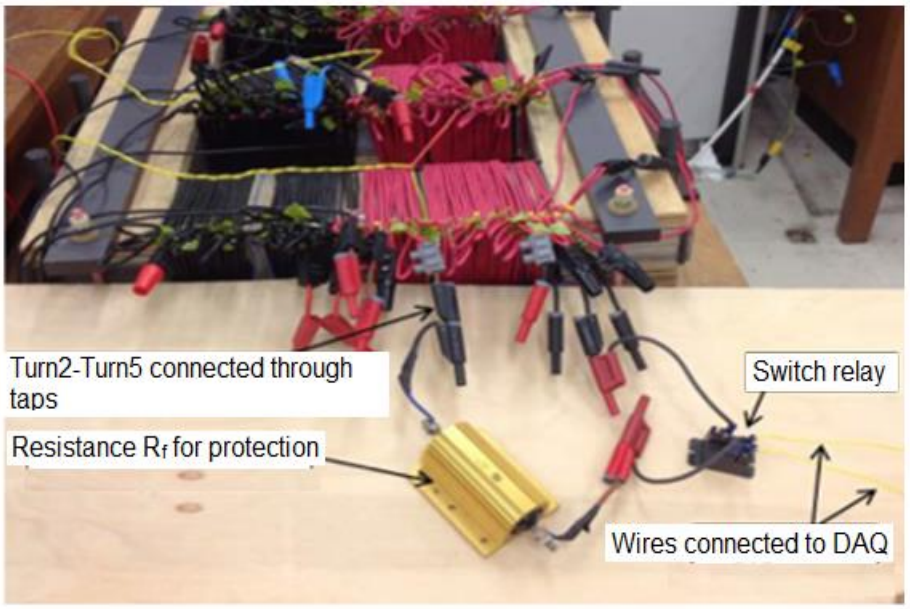

Figure 4- 5 Implementation of interturn fault in the laboratory.

4.2.2 Turn-ground fault generation

In order to simulate a turn-ground fault, multiple turns along the primary and secondary windings of all three phases were grounded at different points along the winding. The experimental setup, as shown in Figure 4-7, only displays the mid-winding turn of phrase A on the secondary side, which was wired in series with a 5Ω resistor and then grounded. The resistor was necessary to limit the high current flowing from the turn to the ground point, as without it, half of the phase voltage would be lost and the windings could be damaged due to the large current flowing through them.

To calculate the predicted current, a phase voltage of 60 volts was applied, and the induced voltage across 30 turns was assumed to be 29.97 V (preferably 30 V) with a wire resistance of approximately 0.02 ohms. This gives a predicted current of 29.97/0.02 = 1498.5 Amp for 30 turns. To protect the windings from this large current, a 5ohm resistor was added in series, which reduced the current to a safe level of 5.99 Amp. When the fault occurs on the secondary side, the majority of the current flows through just 30 turns to ground through a low-resistance wire, making the other 30 turns

redundant. The protective resistor was placed to reduce the current to approximately 6 Amp, which represents the input main current and is measured by the CT supplied.

When the fault occurs on the secondary side, the current flows through the full 60 turns, including the protective resistor. Figures 4-7 and 4-8 illustrate how this scenario can be represented, and how this experiment was conducted in the laboratory.

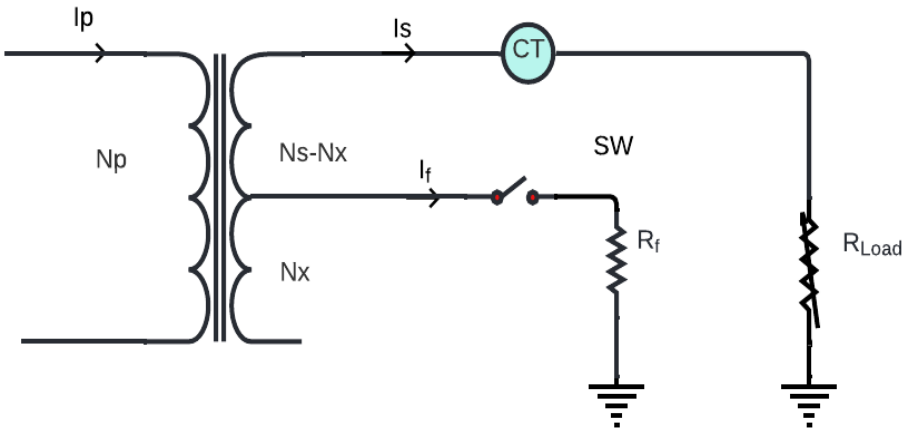

Figure 4- 6 Turn-ground fault representation.

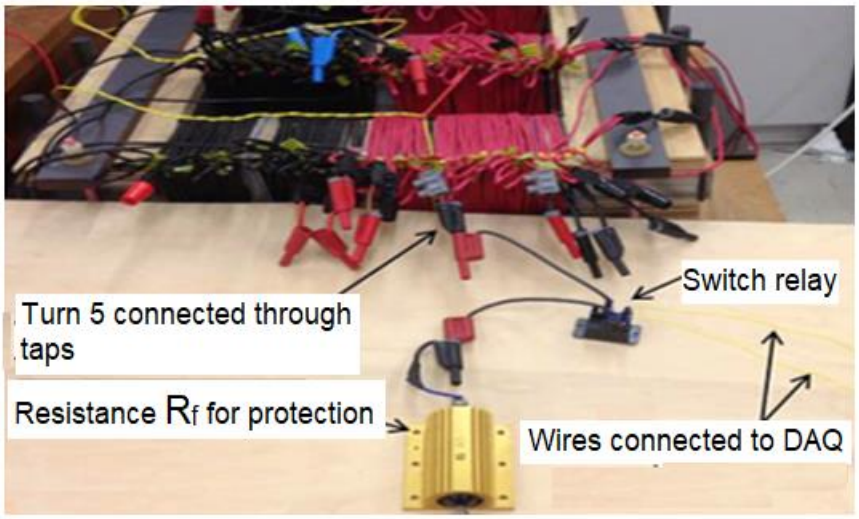

Figure 4- 7 Implementation of turn-ground fault in the laboratory.

Upon closing the fault switch (SW), a fault current of 1.93 Amp was detected by measuring the voltage across the protection resistor R_f of 5 ohms using a clamp meter, which was also recorded as 9.65 V. Additionally, the secondary current flowing through the R_{Load} was measured as 1.46 Amp

by the CT. The primary current I_p was calculated by adding the values of I_s and I_f, which was found to be 3.39 Amp which was read by the power analyzer system (LEM NORMA D 6000 is shown in Figure 4-11).

VL was 40V measured by the voltmeter.

VL = VF + voltage across 30 turns = 9.65 + 30 = 39.65 V.

4.2.3 Single phase-ground fault generation

The fault that occurred was a result of connecting the first turn to the ground through the 5ohm protection resistor R_f, which represents a ground and fault resistance. This type of fault affects the entire phase and is similar to a turn-ground fault current, but it generates the highest fault current. The phase voltage is expected to be significantly reduced when it is generated on the secondary side due to the lower resistance of the fault branch, as opposed to the high resistance of the winding where it is generated. Only a small amount of current flowed through the fault branch. The phase voltage was reduced to about 44 V because of R_f. The CTs on the primary side measure the primary current I_p, which is the sum of fault current I_f and winding current. Due to the extremely low resistance of the fault branch, this current I_f is usually very high, so a protection resistor was added to reduce it to 12 amps.

As depicted in Figure 4-9, when the fault was generated on the secondary side, the current flowing through the load I_L was very small. Instead, most of the current flowed through the fault branch. On the secondary side, only the inductive load was measured by the CT.

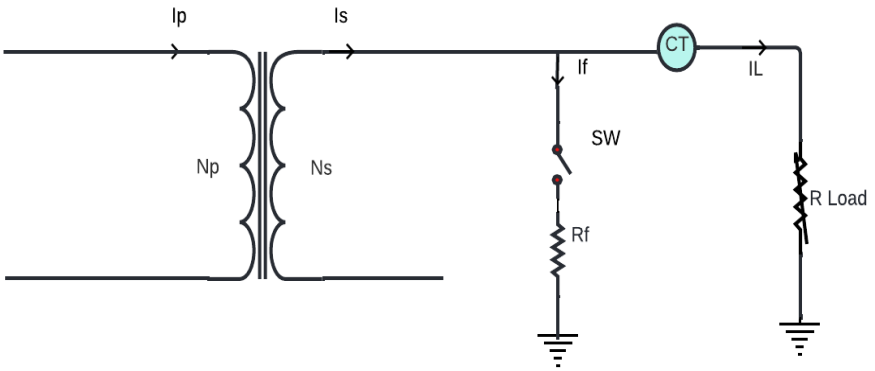

Figure 4- 8 Single phase-to-ground fault representation.

4.2.4 External fault generation

An external fault occurred beyond the protected zone of the transformer, specifically between the load and the transformer's secondary side. As a result, the CT on the secondary side measured the secondary current Is, as shown in Figure 6-10. Upon closing the switch, the phase voltage dropped from 60 V to approximately 20 V, which was equal to the value of V_f (where $V_f = V_L$). This voltage drop caused an increase in the secondary current I_s to around 5 Amps. The current then split into two branches: a high fault current I_f of 3.98 Amps that flowed through the low resistance of the fault resistor R_f (with a resistance of 5Ω), and a low load current I_L of 1 Amp that flowed through the higher resistance load R_{Load}. Hence, the total secondary current I_s was equal to the sum of I_L and I_f, which is 4.98 Amps

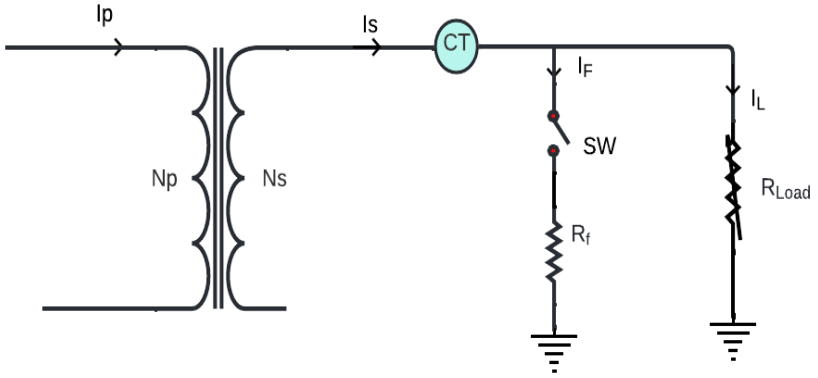

Figure 4- 9 External fault representation.

4.3 Flux density, power loss and interturn fault relationships

In the laboratory, several experiments were performed to investigate the effects of internal defects, such as interturn faults, on the transformer core's power loss and flux density. To conduct these experiments, all the necessary equipment and components were available. The power analyzer system (LEM NORMA D 6000) shown in appendix A was utilized to measure the phase currents and voltages that were provided to the transformer windings. In addition, this system allowed for the calculation of power losses. These experimental results may be valuable to researchers who are interested in examining the correlations between internal defects and the transformer's performance.

4.3.1 Flux density and power loss relationship

When a power analyzer system is connected to the secondary windings of a no-load transformer, it measures the secondary voltages and primary currents. The secondary voltage v_2 in the equivalent circuit of the no-load transformer, as depicted in Figure 4-12, is equal to the primary voltage e_1 since I_2 is zero. Consequently, the power loss (P_{Sec}) on the power analyzer system is solely due to the core

loss. However, when the power analyzer system is attached to the primary winding, it records the primary voltages and currents v_1 and I_1.

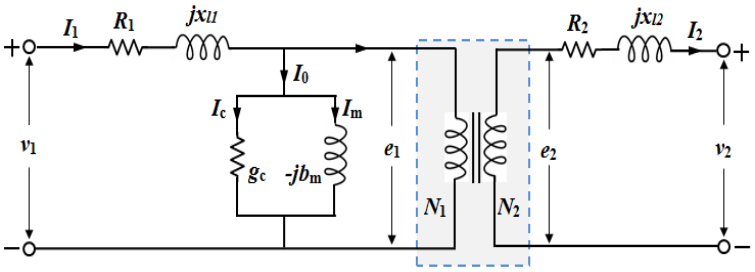

Figure 4- 10 No-load transformer equivalent circuit.

By gradually increasing the voltage supply, the flux density in the core of the transformer increased as depicted in Figure 4-13. This figure shows the core loss associated with different flux densities. The weight of the transformer core was 72.7515 kg. Typically, for a no-load transformer of this size, at a flux density of 1.7 Tesla, the core loss ranges from 1.0 W/kg to 1.2 W/kg.

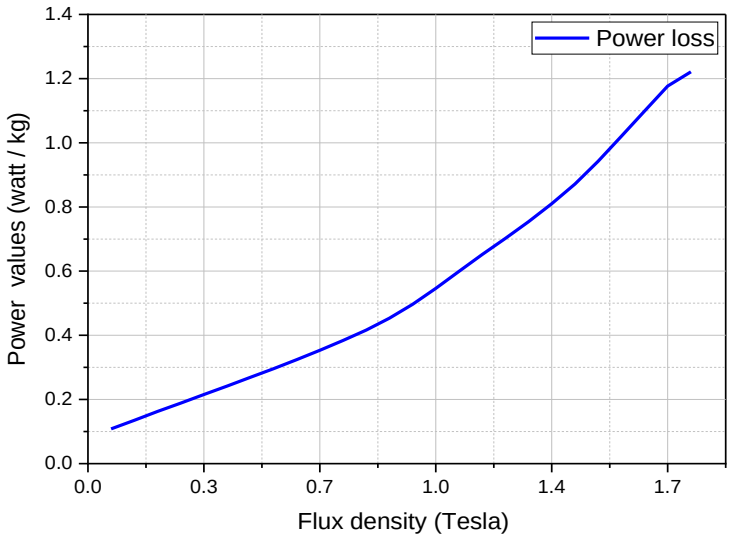

Figure 4- 11 No-load transformer core loss.

4.3.2 Power loss and interturn fault relationship

In the no-load situation, the pure copper loss (P_{cu}) can be calculated by subtracting the core loss (P_{Sec}) from the overall power loss (P_{Pri}), as shown below:

$$P_{cu} = P_{Pri} - P_{Sec} \tag{4-3}$$

where $PPri$ is the total power loss measured by the power analyzer system connected to the primary winding, and P_{Sec} is the core loss measured by the power analyzer system connected to the secondary winding. As mentioned earlier, P_{Pri} consists of both copper loss and core loss, while P_{Sec} only includes core loss.

By subtracting P_{Sec} from P_{Pri}, the copper loss has been obtained, which is due to the resistance of the copper wires in the transformer windings. This copper loss is proportional to the square of the primary current (I_1) and the resistance of the copper wires (R_1). It is important to note that the no-load situation refers to the condition when the transformer is energized without any load connected to its secondary winding. In this situation, the secondary current (I_2) is zero, and the only power consumed by the transformer is due to the core loss and copper loss. The power supply was adjusted at 30V and 60V, respectively, which corresponded to flux densities of 0.53 and 1.05 Tesla. At each set flux density, the number of short circuit turns was increased. At the two flux densities, Tables 4-2 and 4-3 indicate the rise in power losses in the no-load transformer due to the increased number of short-circuited turns.

Table 4- 2 Power losses at flux density of 0.53 Tesla

Number of short-circuited turns	**P_{Pri} (Watt)**	**P_{Sec} (Watt)**	**P_{cu} (Watt)**
0	8.04	7.95	0.09
1	9.99	9.68	0.31
2	15.4	14.7	0.7
3	24.4	23.7	0.7
4	35.3	33.7	1.6
5	48.2	46.5	1.7
6	66.5	63	3.5

Table 4- 3 Power losses at flux density of 1.05 Tesla

Number of short-circuited turns	**P_{Pri} (Watt)**	**P_{Sec} (Watt)**	**P_{cu} (Watt)**
0	29.7	29.6	0.1
1	37.1	36.8	0.3
2	58.3	56.9	1.4
3	95	93.9	1.1
4	140.2	138	2.2
5	196.2	195	1.2
6	260	258	2

As indicated in Figure 4-14, the copper loss was minimal compared to the core loss, which was larger at higher flux densities.

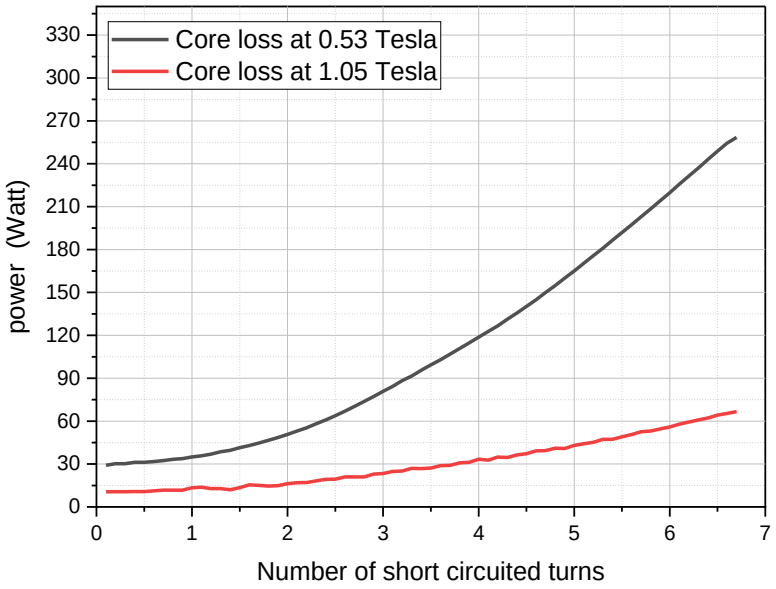

Figure 4- 12 No-load transformer core loss due to short-circuited turns.

The relationship between power losses and the number of short-circuited turns was evident, as the latter resulted in an increase in primary current. The high circulating current in the loop of short-circuited turns also resulted in power loss in the winding turn wires. If the short-circuited turns were applied for a very brief period of time to prevent damage, power loss arose from the heat in the wires, and smoke was observed coming out from the affected turns.

4.3.3 Flux density and interturn fault relationship

The distribution of flux in the core limbs of a transformer was studied by conducting tests with and without a load, in order to observe the effect of short-circuit turns.

1. The test on no-load transformer

To minimize the potential danger of a short circuit between turns, the test was conducted using maximum supply voltages of 30V and 60V. All phases were tested for interturn faults, but only the results for phase A are presented since they were consistent with those of the other phases. The short-circuited turns in phase A had an impact on the flux density in the core limb, as illustrated in Figures 4-15a and 4-15b.

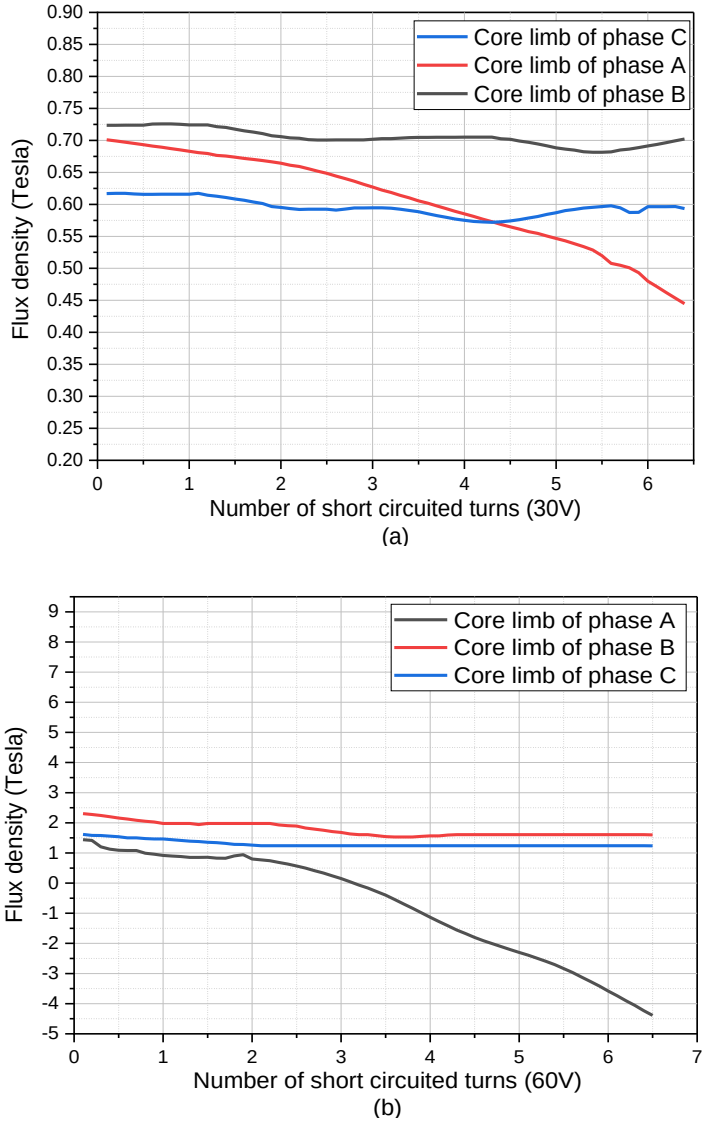

Figure 4- 13 The effect of interturn fault on flux density when no-load transformer was operated at (a) 30V and (b) 60V.

2 The test on on-load transformer

When the transformer was connected to the load, a similar test was conducted. The results, as depicted in Figures 4-16-a and 4-16-b, showed that short-circuited turns had the same impact on the flux density as observed during no-load transformer tests.

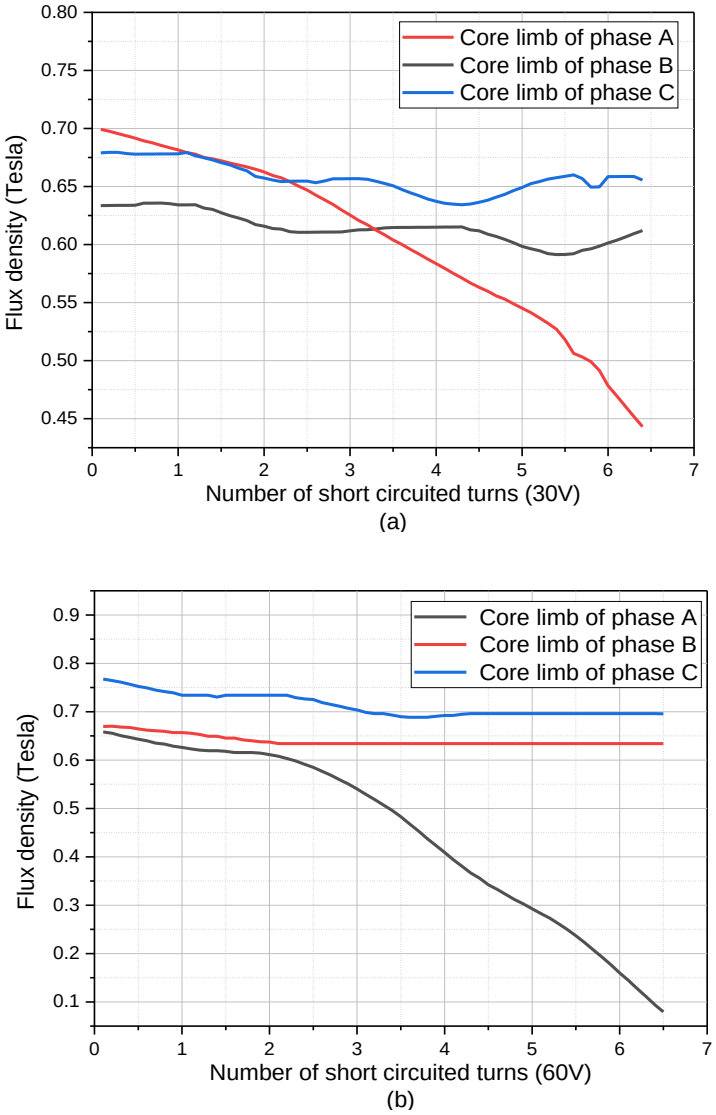

Figure 4- 14 The effect of interturn fault on flux density when on-load transformer was operated at (a) 30V and (b) 60V.

In both test scenarios, it was evident that an increase in the number of short-circuited turns led to a decrease in the flux density within the core limb enclosed by the winding of phase A, where the turns were short-circuited. The rationale behind this phenomenon lies in the behavior of an interturn fault within the primary winding. When such a fault occurs, a substantial circulating current is generated within the loop of short-circuited turns. This circulating current acts as the source of a magnetomotive force (mmf) that opposes the primary winding's magnetic field while aligning itself with the secondary winding's magnetic field. Consequently, the turn-turn fault exerts an effect that diminishes the primary flux, resulting in an increase in the primary current and a corresponding decrease in the secondary current to offset this change. In a healthy transformer, the distribution of flux density maintains a horizontal symmetry axis, passing through the core limb's center. In this configuration, the magnetic flux within the core significantly outweighs the leakage flux. However,

following the occurrence of an interturn fault, the leakage flux surrounding the short-circuited turns undergoes a substantial increase, whereas the flux within the core limb enclosed by these short-circuited turns experiences a decline. It's noteworthy that when an interturn fault transpires within the secondary winding, the current and flux density distribution exhibit analogous characteristics to those observed during a fault within the primary winding.

4.4 Laboratory model description

The laboratory models consisted of several components, which included:

1. A 3-phase transformer
2. A power supply
3. A 3-phase load
4. Six current transformers, with a turn ratio of 40/5, a rated burden of 1VA, and a Class 3 FS5
5. A switch relay, OMRON G8P series SPST-NO, 5VDC, 30A
6. A data acquisition card (6211)
7. A power analyzer, LEM NORMA D 6000
8. LabVIEW software

The connections of the system model are illustrated in Figure 4-17, which shows that a 3-phase power supply was equipped with six current transformers. On the primary side of the transformer, there were three CTs, namely CT1, CT2, and CT3. Similarly, on the secondary side, there were CT4, CT5, and CT6.

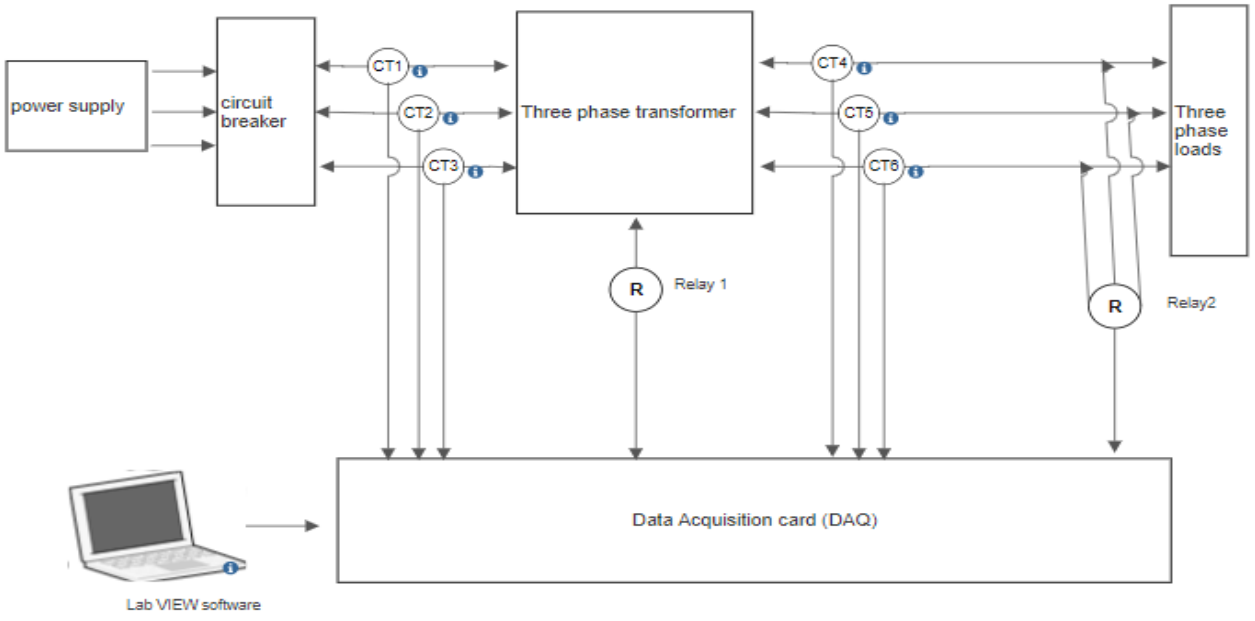

Figure 4- 15 Schematic diagram for the system model.

The power analyzer system was connected to the windings of the transformer, allowing it to measure the current and voltage values of each phase and adjust voltage and current as needed, while ensuring that the three transformer voltages were evenly balanced. The data acquisition card (DAQ) was connected to a laptop via USB cable and controlled using LabVIEW software. This software was responsible for triggering a three-phase switch for energization, switch relay 1 to generate internal faults, and switch relay 2 to generate external faults.

The transformers used in the laboratory were equipped with numbered tap points along their windings, which were used to generate internal faults. Figure 4-18 shows the connection between the DAQ system and the laboratory components. Voltage trigger signals were sent to the switch relays from the DAQ system, and reduced current signals were supplied by CTs. To be accepted by the DAQ system, the current signals were converted to voltage signals using burden resistances of 0.1 ohms. However, the LabVIEW program converted the voltage signals back to current signals. Appendix A provides an overview of the LabVIEW model which was modelled to control the practical tests.

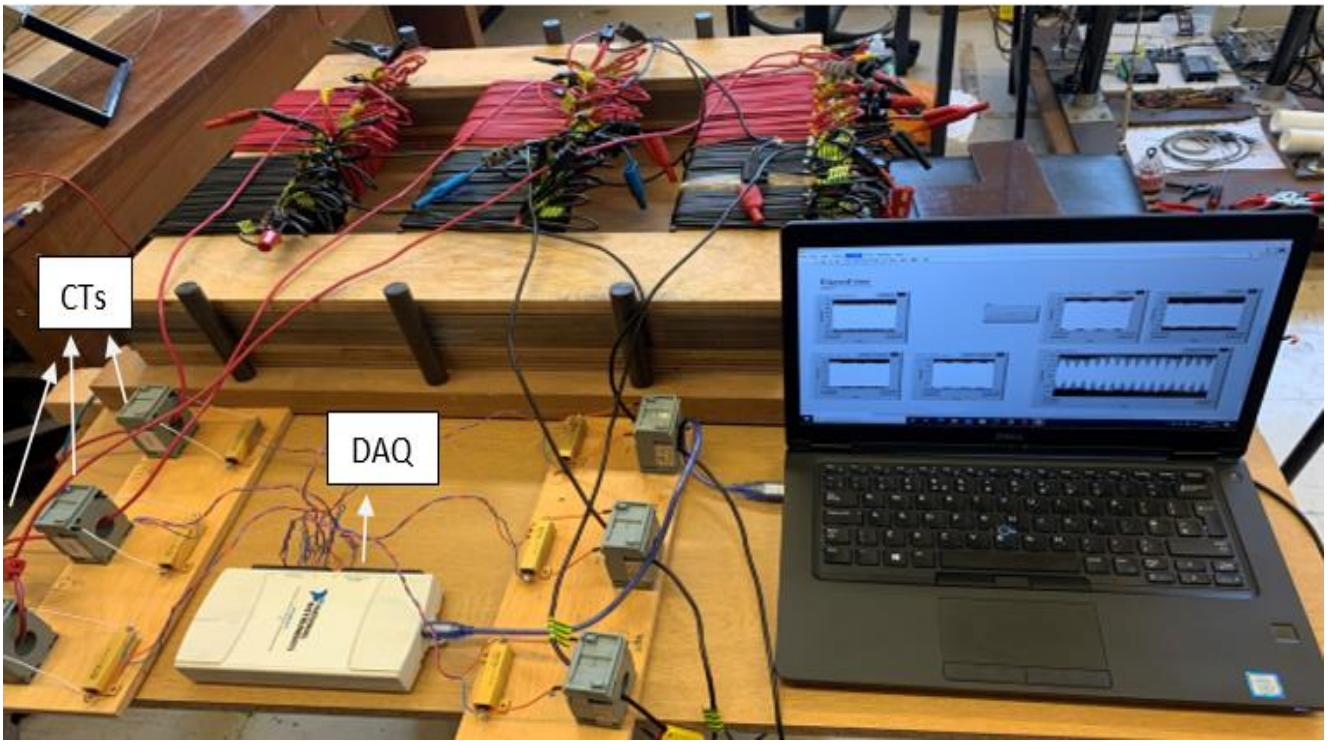

Figure 4- 16 Data acquisition card connections.

4.5 Experiment procedure

During the practical tests, the proposal technique was applied to the transformer while it was in use, following the procedure used in the simulation with MATLAB/SIMULINK as shown in chapter 5, section 5.2.2. The LabVIEW program was set to run at a sample rate of 10 kHz, which means that 200 samples were taken per cycle. To analyze the faults, the LabVIEW program ran for 10 seconds,

resulting in 100,000 samples (excluding zero-time samples) being saved into an Excel spreadsheet with seven columns. Column 1 represents the sampling time, columns 2, 3, and 4 contain the primary side data obtained from phases A, B, and C respectively, while columns 5, 6, and 7 represent the secondary side data. All signals acquired from the data acquisition card were filtered using a low pass filter with a cutoff frequency of 50 Hz and Butterworth topology order 1. This filtering process removed harmonics and noise, which was performed within the LabVIEW program. As a result, the data collected represent the fundamental frequency with the removal of noise and harmonics in the first proposed method.

4.6 Summary

In this chapter, the construction and assembly of the transformer model in the laboratory were discussed, along with the methods used to generate internal and external faults on the model. Real data was collected during the fault generation process to test the proposed analogy algorithms, which were developed as MATLAB programs. One of the most common faults observed in transformers is the turn-to-turn short circuit fault, which can cause damage to the winding turns and insulation. The experiments conducted in the laboratory demonstrated the impact of the interturn fault on the distribution of flux in the transformer limbs where the windings are wrapped, and the resulting power loss in the core of the transformer. The knowledge gained from these experiments can be valuable for further research in this field.

CHAPTER 5

Alienation Coefficient Method

5.1 The proposed method

The aim of this chapter is to introduce innovative techniques for assessing the condition of three-phase transformers using waveform Alienation Coefficient analysis. One suggested method involves calculating the Alienation coefficients, which can effectively detect the presence of internal or external faults. To protect the transformer from internal faults, a differential protection concept is utilized, where the currents entering and leaving the transformer terminals are compared. This approach relies on variations in current waveforms that occur as a result of faults. The method relies on measuring the three-phase currents that can be obtained directly from the transformer, without the need for additional equipment. This method can be used for fault detection and classifying faulty phases. For usage in advanced protection plans, the alienation strategy is appropriate in this field. The newly developed technique was applied in both MATLAB Simulink for simulation and in the laboratory for practical experimentation. Results were obtained through the combined use of MATLAB and LabVIEW software. Notably, this innovative technique is exceptionally rapid, capable of delivering conclusions in less than 3 milliseconds.

5.1.1 Alienation Coefficients concept

The coefficient of correlation evaluates the similarity of two sets of measurements (i.e., two dependent variables) obtained in the same observations [83]. The correlation coefficient indicates how much data is shared between the two variables. The range of this coefficient is –1 to +1. (inclusive). The two sets of measurements are measuring the same thing if the answer is +1. If the answer is -1, the two measurements are measuring the same object, but one is changing in the opposite direction of the other. The two sets of measurements have nothing in common if the answer is -1 [84]. The variance between any two signals is calculated using the alienation coefficient, which is obtained from the cross-correlation coefficient.

$$\text{Alienation coefficients of phaseA, Aa} = 1 - (r_a)^2 \qquad (5-1)$$

$$\text{Alienation coefficients of phase B, Ab} = 1-(r_b)^2 \qquad (5-2)$$

$$\text{Alienation coefficients of phase C, Ac} = 1-(r_c)^2 \qquad (5-3)$$

5.1.2 Cross-correlation function

The cross-correlation coefficient function is a valuable tool in signal processing, particularly for applications in pattern recognition. It is utilized to measure the similarity between two waveform signals over time. In the context of transformers, this coefficient can be employed to assess the similarity between two signals, such as voltage or current signals, obtained from two transducers [85]. The signals are sampled simultaneously and divided into windows, each containing a specific number of samples. The cross-correlation coefficient is then calculated using equation (5-4), which involves computing the coefficient between a window of one signal (x) and the opposite window of the other signal (y).

$$r = \frac{\sum_{k=1}^{n} x_k \ y_k \ - \frac{1}{n}\sum_{k=1}^{n} x_k \sum_{k=1}^{n} y_k}{\sqrt{\sum_{k=1}^{n}(x_k \ )^2 \frac{1}{n}(\sum_{k=1}^{n} x_k \ )^2}\sqrt{\sum_{k=1}^{n}(y_k)^2 \frac{1}{n}(\sum_{k=1}^{n} y_k)^2}} \qquad (5-4)$$

The new technique is based on the alienation coefficient being calculated. It is recommended that the transformer be protected against internal faults when it is in operation. This technique uses differential protection by comparing the current entering or leaving the transformer. It is calculated by analyzing the correlation between current signals produced by current transformers (CT) installed at both ends of the transformer. As a result of the new technique, the alienation coefficient is paired with a cross-correlation coefficient to improve fault discrimination and detect faults within a very short time.

Two correlation coefficients are determined in this case, a cross-correlation based on primary and secondary current signals of phases (A, B, and C) and then an alienation coefficient based on the cross-correlation between them. In order to determine whether there is a fault, these coefficients will be compared to threshold values. Figure 5-1 illustrates the proposed protection scheme for a single-phase transformer.

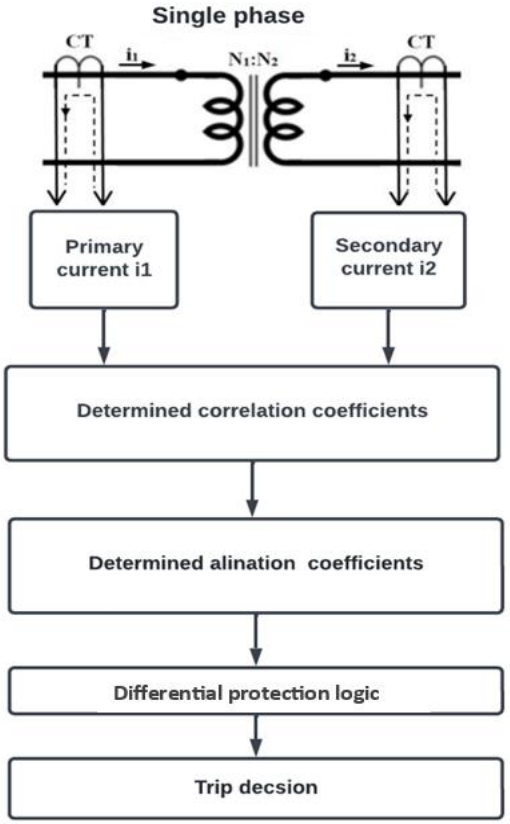

Figure 5- 1 Protection scheme for a single-phase transformer.

The cross-correlation coefficient for phase A (r_a) is calculated between every two currents (i_{ap} and i_{as}) on the primary and secondary side of the transformer, as shown in Equation (5-5), where the two windows are shifted from each other with a time interval of $h_{\Delta t}$. When the time gap between the primary and secondary signals is longer than the time interval between the primary and secondary signals, this approach is used $h_{\Delta t} = 0$, where h = 0 (h is the number of samples between the two windows which are shifted from each other and Δt is the time interval of one sample). Also, the cross-correlation coefficients r_b and r_c are given by Equations (5-6) and (5-7) respectively:

$$r_a = \frac{\sum_{k=1}^{m} i_{ap}\ (k) i_{as}(k+h\Delta t) \frac{1}{m} \sum_{k=1}^{m} i_{ap}\ (k) \sum_{k=1}^{m} i_{as}(k+h\Delta t)}{\sqrt{\sum_{k=1}^{m} (i_{ap}\ (k))^2 \frac{1}{m} (\sum_{k=1}^{m} i_{ap}\ (k))^2} \sqrt{\sum_{k=1}^{m} (i_{as}(k+h\Delta t))^2 \frac{1}{m} (\sum_{k=1}^{m} i_{as}(k+h\Delta t))^2}} \qquad (5-5)$$

$$r_b = \frac{\sum_{k=1}^{m} i_{bp}\ (k) i_{bs}(k+h\Delta t) \frac{1}{m} \sum_{k=1}^{m} i_{bp}\ (k) \sum_{k=1}^{m} i_{bs}(k+h\Delta t)}{\sqrt{\sum_{k=1}^{m} (i_{bp}\ (k))^2 \frac{1}{m} (\sum_{k=1}^{m} i_{bp}\ (k))^2} \sqrt{\sum_{k=1}^{m} (i_{bs}(k+h\Delta t))^2 \frac{1}{m} (\sum_{k=1}^{m} i_{bs}(k+h\Delta t))^2}} \qquad (5-6)$$

$$r_c = \frac{\sum_{k=1}^{m} i_{cp}\ (k) i_{cs}(k+h\Delta t) \frac{1}{m} \sum_{k=1}^{m} i_{cp}\ (k) \sum_{k=1}^{m} i_{cs}(k+h\Delta t)}{\sqrt{\sum_{k=1}^{m} (i_{cp}\ (k))^2 \frac{1}{m} (\sum_{k=1}^{m} i_{cp}\ (k))^2} \sqrt{\sum_{k=1}^{m} (i_{cs}(k+h\Delta t))^2 \frac{1}{m} (\sum_{k=1}^{m} i_{cs}(k+h\Delta t))^2}} \qquad (5-7)$$

Where (k) is the instant value for the phases A, B and C of the transformer and m is the number of samples per window. The number of samples per quarter-cycle are selected in the algorithm m = N/4 = 2000/4 = 500 samples. The alienation coefficient A_a, calculated between the two current signals, is obtained from the cross-correlation coefficient (r_a), and is given in Equation (5-1). Also, alienation coefficients A_b and A_c are given by Equations (5-2) and (5-3), respectively.

5.1.3 Detection and selection fault

A flow chart for algorithm protection based on alienation coefficients is shown in Figure 5-2. The algorithm has the following procedures:

1. Read discrete samples for three-phase current signals of the transformers.
2. Calculate input and output current values (i_{ap} (k), i_{as} (k), i_{bp} (k), i_{bs} (k), i_{cp} (k) and i_{cs} (k)) for each phase of the protected busbar.
3. Calculate the cross-correlation coefficient between the primary and secondary side of each three-phase transformer (r_a, r_b and r_c).
4. Calculate the alienation coefficients A_a, A_b and A_c using the calculated cross-correlation coefficients r_a, r_b and r_c, respectively, as given in Equations (5-1), (5-2) and (5-3).

If the value of the alienation coefficients for each phase is equal to zero, this indicates normal operation in which there is no internal fault (turn-turn or turn -ground) within the transformer. In the case that one of the values is not equal to zero, this means that there is an internal fault in this phase.

Figure 5- 2 Flowchart for Protection Algorithm Utilizing Alienation Coefficients.

Therefore, this idea is developed for use in classification between internal and external faults, and to identify which type of faulty phase is present. The method calculates the three-phase alienation coefficients A_a, A_b and A_c. The value of (r_a, r_b and r_c) is equal to one because the phase shift is zero in case of normal operation or external fault without CT saturation; hence $A_a = A_b = A_c = 0$.

The case of internal faults is different because there is no ideal operation in the grid. Thus, the threshold used is 0.09 instead of 0 to avoid this obstacle:

1.In cases of normal operation or external faults, the value of the alienation coefficients of the three phases (A, B and C) are less than or equal to threshold 0.09.

2.In case of turn to turn or turn to ground faults in three phases, the value of the alienation coefficients of three phases (A, B and C) is greater than threshold 0.09.

3.In both cases of turn to turn and turn to ground fault, one values of A_a, A_b and A_c will be greater than 0.09, as shown in Table 5-1

Table 5- 1 Alienation coefficients value of internal fault

Type of phase	**Alienation coefficients value**	**Type of fault**
Phase A	Aa >0.09	Internal fault (turn-turn or turn to ground
Phase B	Ab >0.09	
Phase C	Ac >0.09	

4.Tripping action

This case presents the action of the circuit breaker under normal operation, external and internal faults, as illustrated in following table:

Table 5- 2 Tripping action of the circuit breaker

Fault type	**Alienation Coefficients**	**Action**
Normal operation	$A_a \leq 0.09, A_b \leq 0.09 \; and \; A_c \leq 0.09$	Send signal to the circuit breaker to stay close
External fault	$A_a \leq 0.09, A_b \leq 0.09 \; and \; A_c \leq 0.09$	Send signal to the circuit breaker to open or stay close
Internal fault	$A_a > 0.09, A_b > 0.09 \; and \; A_c > 0.09$	Send signal to the circuit breaker to open

5.2 Transformer model simulation

Figure 5-3 below shows the model that was processed by the MATLAB/SIMULINK program. MATLAB was used to program the Alienation coefficients method, which was put in a function block of the model and used to process the data obtained from the six current transformers (CT_1-

CT_6) connected on both sides of the transformer. Both models and simulations are involved in the process.

5.2.1 Modelling

The MATLAB function block is employed to define the MATLAB code responsible for computing Alienation Coefficients using data stored in the memory, as illustrated in Figure 5-3 and Appendix B. This figure depicts a system featuring a three-phase power source and a circuit breaker designed to safeguard a three-phase transformer. Additionally, there are six current transformers installed on both the primary and secondary sides to collect current signal data. These data are subsequently utilized by the MATLAB code to calculate Alienation Coefficients. Depending on these coefficient values, the final block, referred to as the "fault results" block, will transmit a signal to the circuit breaker, instructing it to either remain closed or open in the event of any detected faults.

5.2.2 Simulation

The model was simulated for 0.2 seconds at a system frequency of 50 Hz, with a sampling rate of 10 kHz, meaning that a sample was taken every 0.0001 seconds. Each cycle was represented by 200 samples. To avoid losing waveform peaks, the samples were divided into windows, with each window containing 20 samples. Specifically, each window included 10 samples to the left of the peak value and 10 samples to the right of the peak value, making a total of 20 samples per window. A MATLAB program was used to program equations (5-5), (5-6) and (5-7) that computed the cross-correlation coefficients between primary and secondary current of the same phase at the same time and then computed the Alienation coefficients for each phase. Each decision for any case evaluation was based on the computed results obtained every 2 ms (20 samples multiply by 0.0001 sample interval).

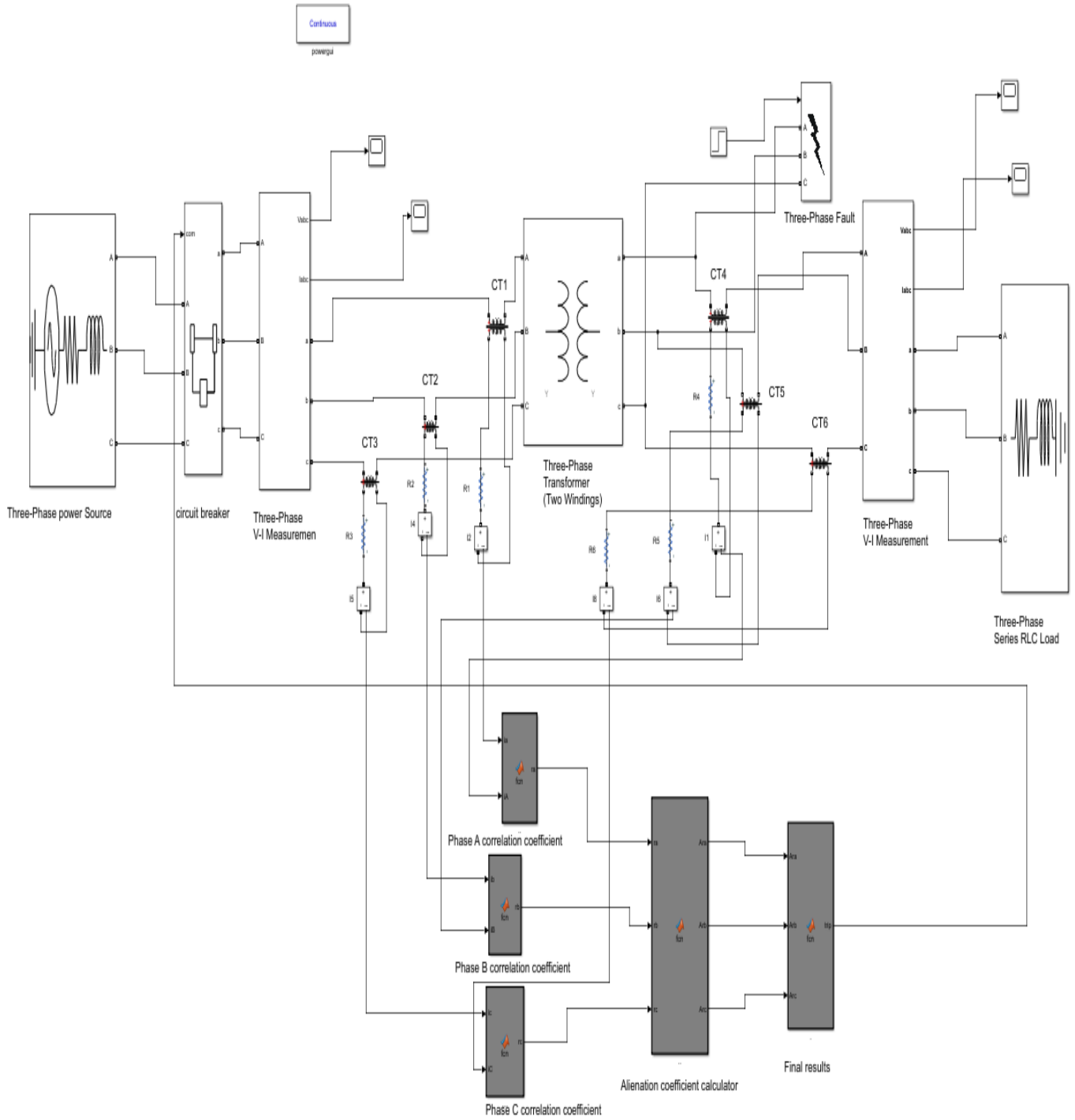

Figure 5- 3 System model for the protection technique based on correlation analysis.

5.2.3 Faults versus proposed correlation protection algorithm

The suggested method was put to the test against the following faults. All faults were produced at 0.06 second and lasted for three cycles for comparative reasons by closing specified fault switches in the system model. Only phase A will be presented because the results were not different between phases.

1. Interturn (turn-turn) fault

Turn-turn faults were represented by 2% of short-circuited turns. They occurred on the secondary side of phase A. Figure 5-4 shows the changes of the current in both sides (primary and secondary) during the fault.

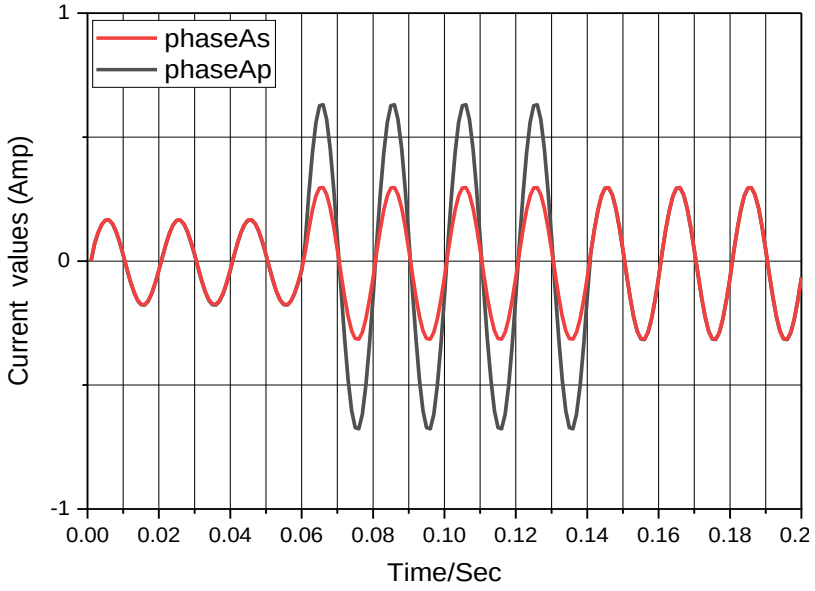

Figure 5- 4 Primary and secondary current phase A (turn-to-turn fault).

Phase B and C have applied the same strategy and the results are the same as phase A. MATLAB can calculate the Alienation Coefficient for all phases simultaneously. Each Alienation Coefficient is calculated between each of the two corresponding quarter-cycles of primary and secondary currents of phase A. The value of Aa is equal and close to zero before the fault begins. From the result above, the alienation coefficients value at turn-to-turn fault initiation are good detector to determine the faulted and the zone of fault location in three phases of the transformer; their values are greater than threshold 0.09 in case of turn-to-turn fault as shown below:

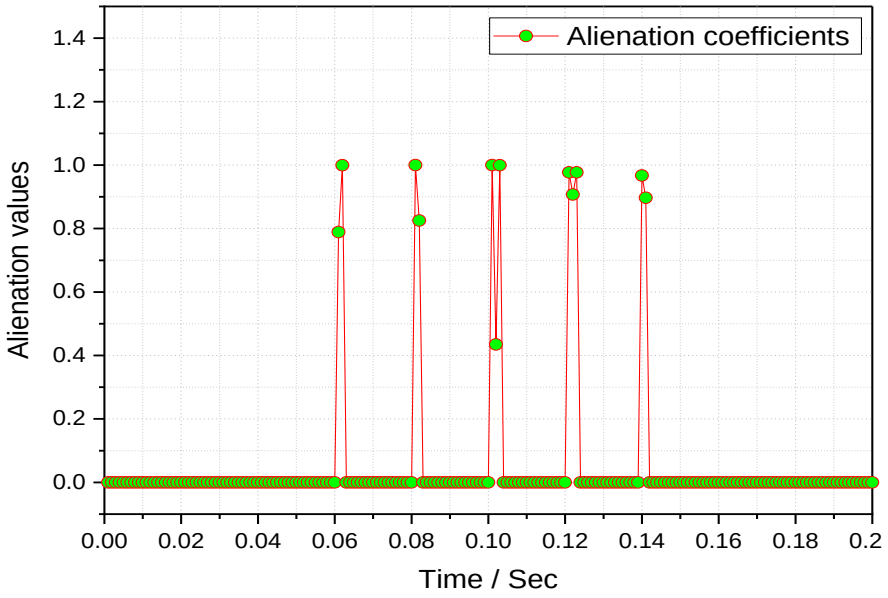

Figure 5- 5 Change in alienation coefficients during turn-turn fault in phase A.

The fault was detected at 0.062 second, according to the method displayed in Figure 5-1. Because the fault was created at 0.06 second, it was detected at 0.062 - 0.06 = 0.002 second or 2 milliseconds after the fault occurred. As illustrated in Figure 5-5, a trip signal was produced for the faulty phase A.

2. Turn-ground fault

For testing the turn to ground fault at different places along the winding of all three phases on both primary and secondary sides, different turns along the winding were grounded. Only the midturn of phase A on the secondary side is displayed because the outcomes were comparable. Primary current increased caused by the loss of half voltage on the secondary winding and a low fault resistance of 5 Ω, however secondary current reduced due to the majority of current passing via a low resistance fault branch instead of a load, as illustrated in Figure 5-6.

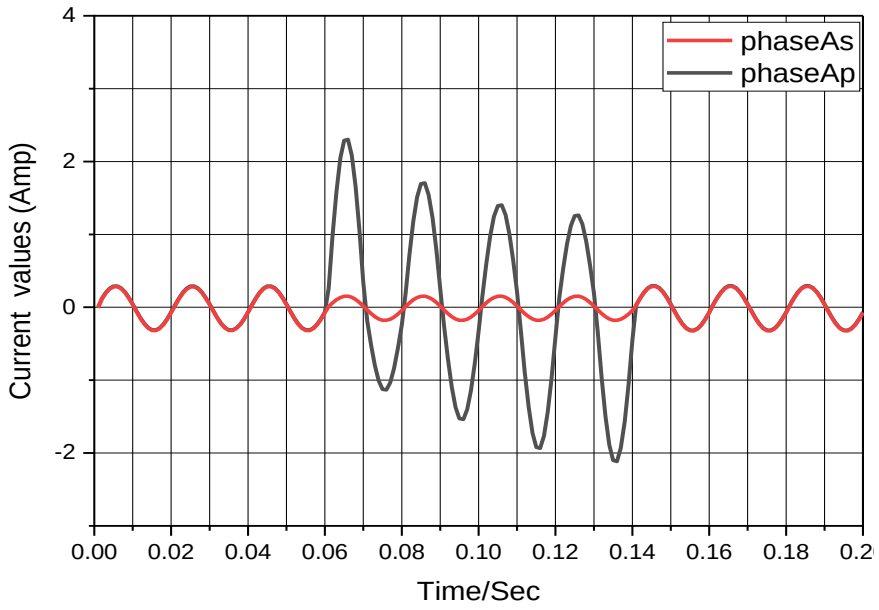

Figure 5- 6 Primary and secondary current phase A (turn-to-ground fault).

At 0.066 second, a fault causes the alienation coefficients to reach 0.95 simultaneously, as shown in Figure 5-7. After 6 milliseconds of occurrence of the fault, a trip signal was issued according to the method.

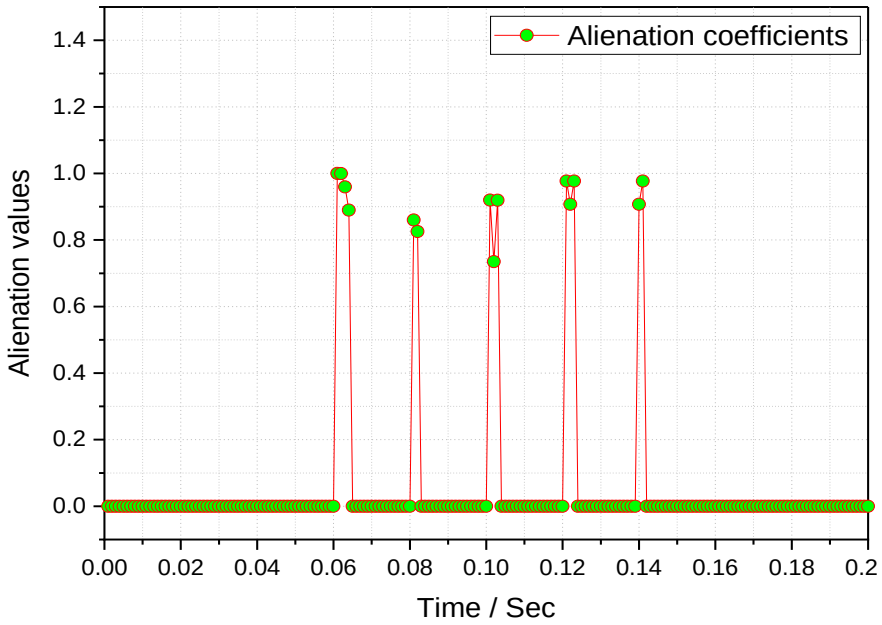

Figure 5- 7 Change in alienation coefficients during turn-ground fault in phase A.

3. Phase-to-ground fault

The first turn on secondary side of phase A was grounded to generate this fault. Thus, this phase was entirely lost. Thus, most of secondary current flowed through the fault branch while almost zero current flowed through the load as shown in Figure 5-8.

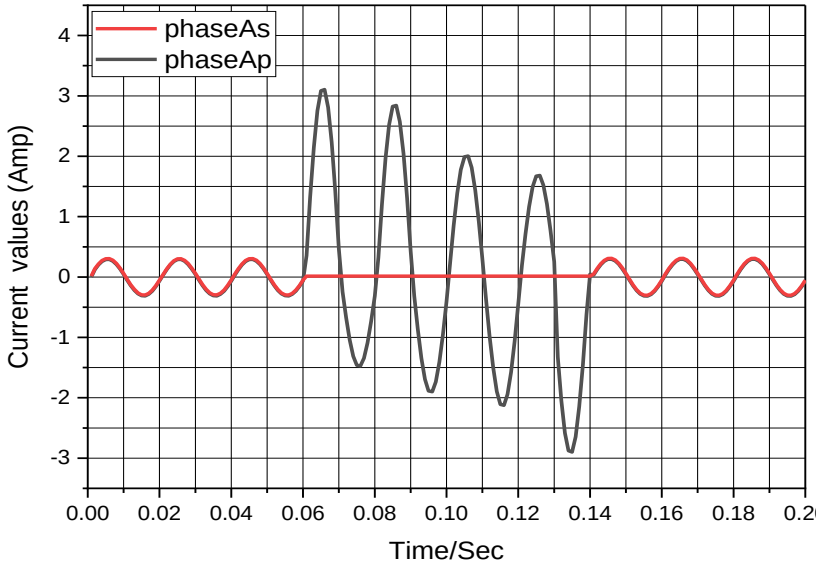

Figure 5- 8 Phase-to-ground fault in phase A.

Due to this fault, the alienation coefficients value simultaneously increased above 0.95 at 0.062 second in Figure 5-9. Consequently, the secondary current was much more impacted by the fault than the primary current.

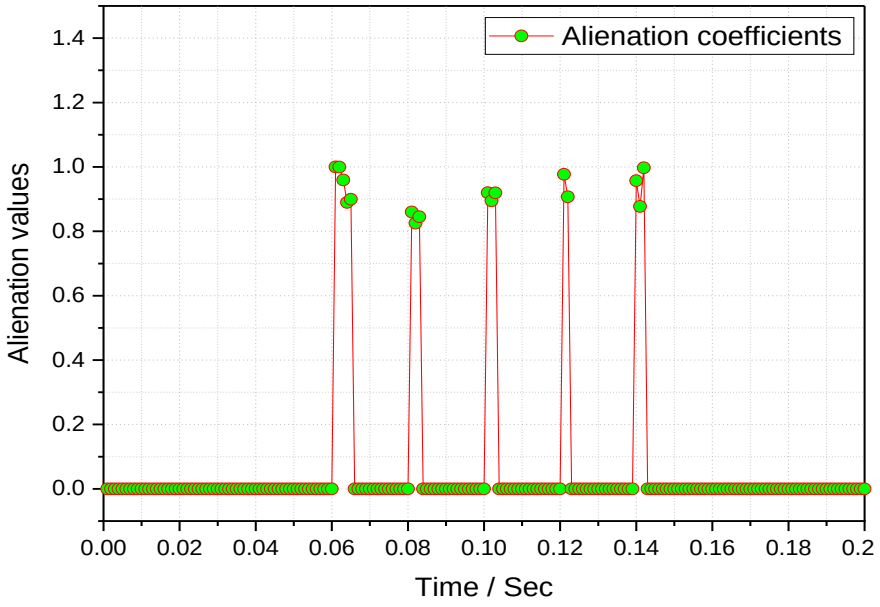

Figure 5- 9 Change in alienation coefficients during phase-to-ground fault in phase A.

Based on the method, a trip signal was issued at 0.062 seconds 0.062 - 0.06 = 0.002 second or 2ms from the instant of fault occurrence, which means that action was taken very quickly.

4. Phase-phase fault

This type of fault was generated by connecting phases A and B through a fault resistor (R_f) of 5 Ω. This fault increased the value of the primary current of phase A in the same direction as the

secondary current, while it increased the value of the primary current of phase B in the opposite direction. See Figures 5-10a and 5-10b for diagrams illustrating this phenomenon.

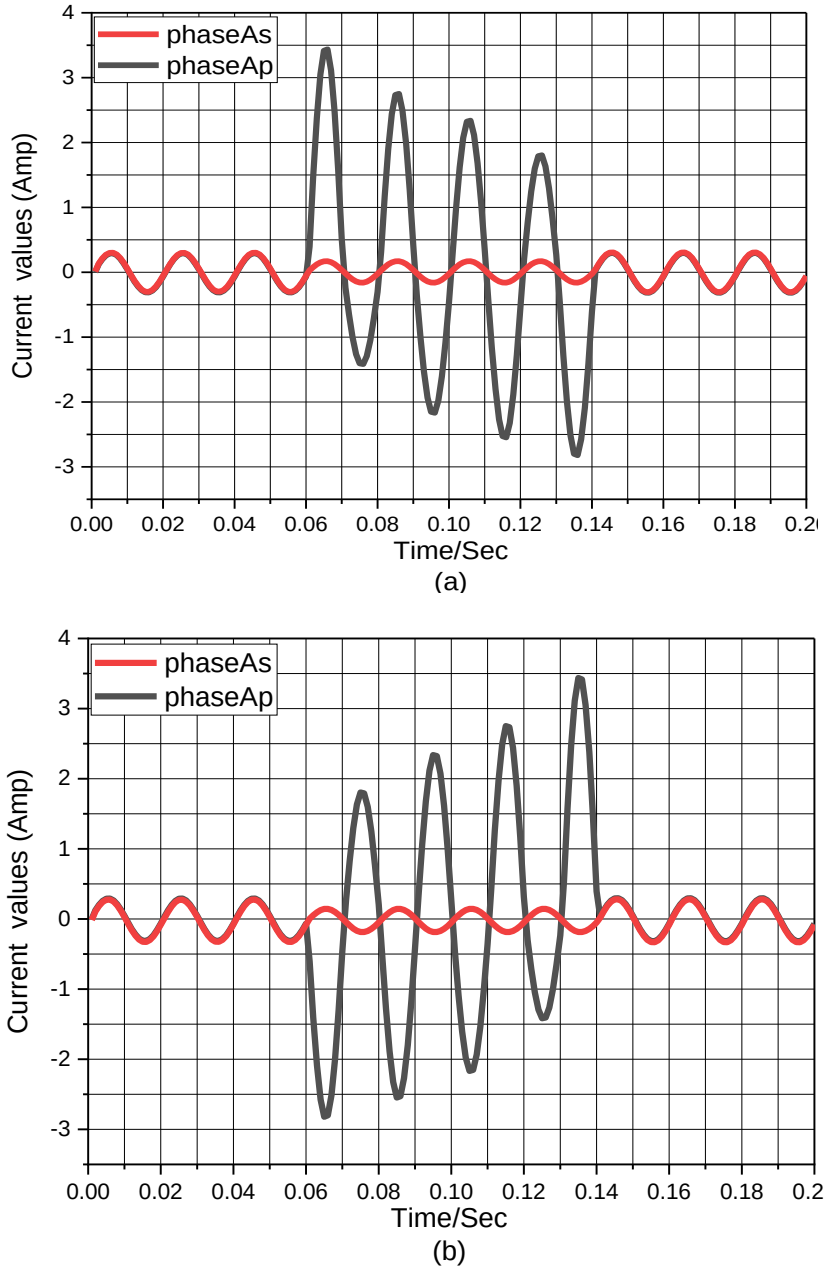

Figure 5- 10 Phase-phase fault between (a) phase A and (b) phase B.

According to Figures 5-11a and 5-11b, the values for alienation coefficients for phases A and B decreased to less than 0.95 in the first three windows after the occurrence of the fault. Because the secondary current dropped dramatically in phase A, the primary current increased pretty much in the same direction. As illustrated in Figures 5-10a and 5-10b, in phase B, the primary current mostly increased in the same direction, while the secondary current decreased as if it were reversed.

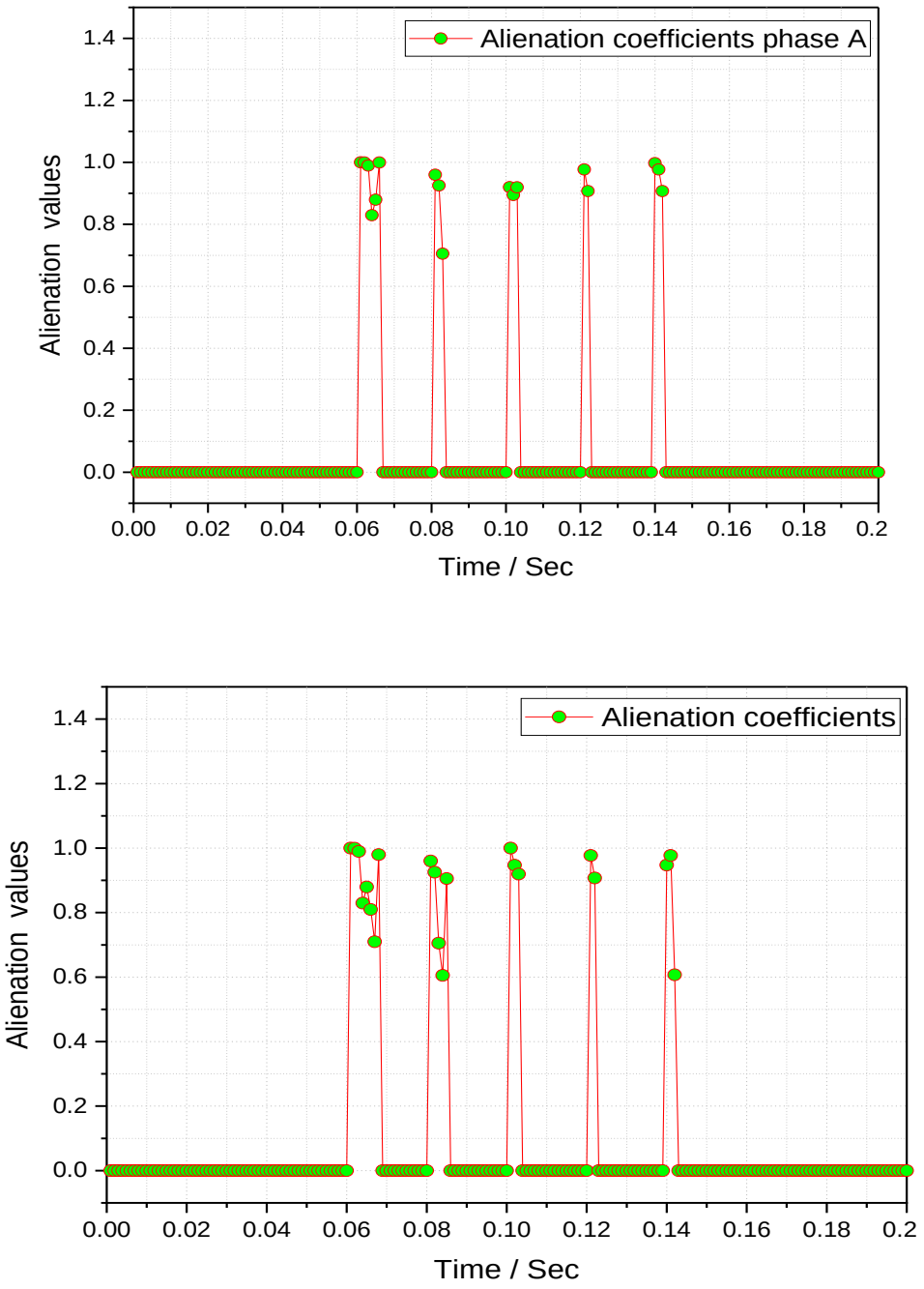

Figure 5- 11 Change in alienation coefficients of phase A and phase B during phase-phase fault.

A trip signal was issued after 0.062 seconds according to the method shown in Figure 5-1, i.e. this fault was detected after just 2ms from the occurrence of the fault. A trip signal was also issued under two different fault conditions.

5. External fault

To generate this type of fault in phase A outside transformer protected zone, the external fault switch in the Simulink model in Figure 4-2 was closed. According to Figure 5-12, both primary and secondary currents increased simultaneously since the fault occurred outside the connection between the CTs.

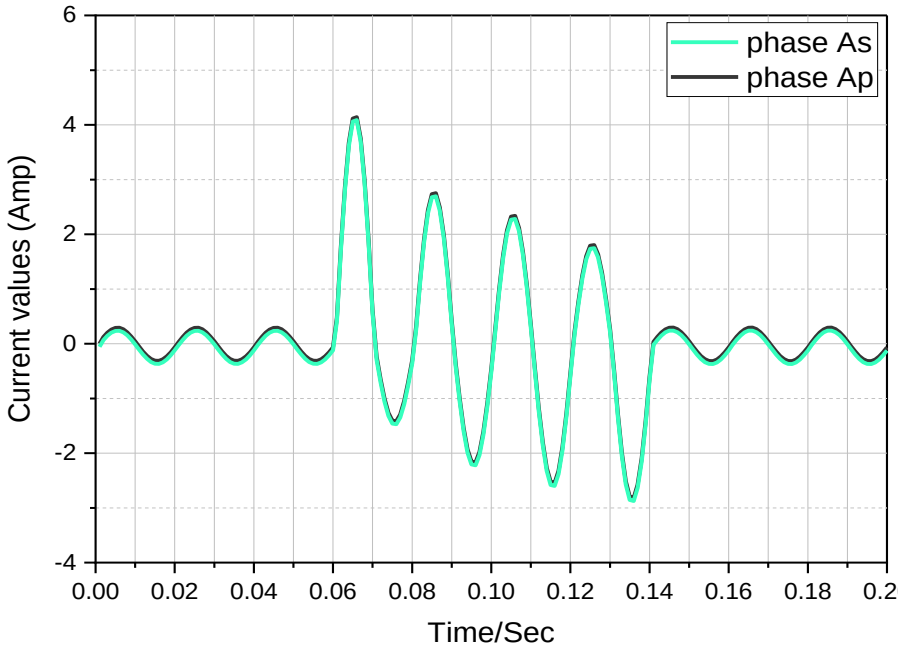

Figure 5- 12 External fault in phase A.

At 0.063 seconds, the alienation coefficients decreased to less than 0.09 due to the change in both primary and secondary currents. This is displayed in Figure 5-13 since both currents are still identical at this point. The method was likewise put to the test in phases B and C. The results were identical to those of the phase A case.

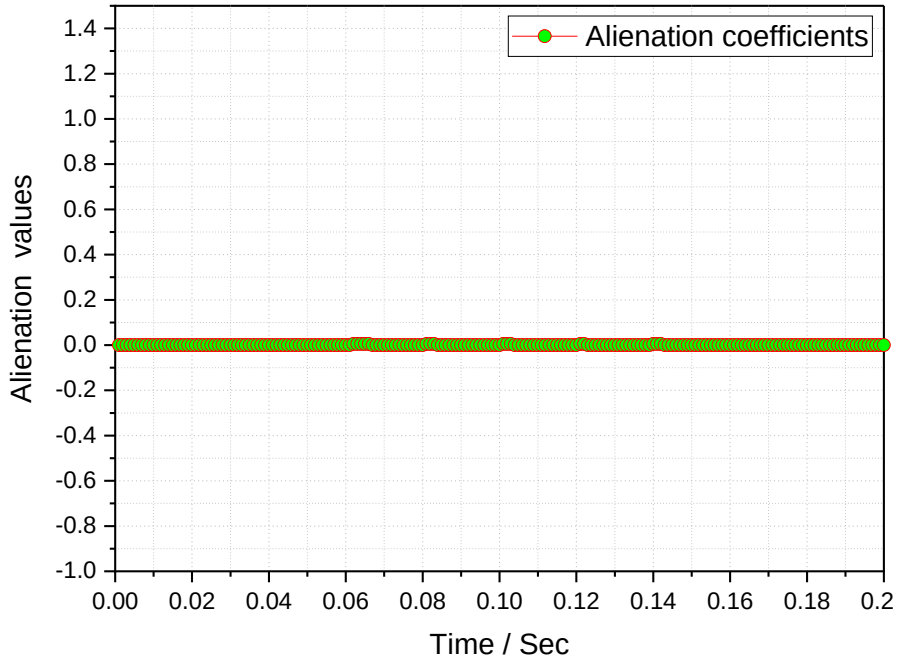

Figure 5- 13 Change in alienation coefficients of phase A.

5.2.4 Current Transformer saturation

In the case of possible CT saturation problems arising from internal and external faults, the proposed algorithm is tested. Figure 5-14 represents a protection scheme for 1- phase used to simulate CT saturation, which is caused by the following faults: -

1. Turn-ground fault with saturation of CT

To induce a turn-to-ground fault, 50% of the turns on the secondary side of phase A were grounded. The fault incidence time and duration were identical to those in the previous cases. As a result of this fault, the primary current increased, while the secondary current decreased, with the magnitude of the increase and decrease dependent on the fault intensity. Consequently, CT1 was likely to become saturated. To induce saturation in CT1, its burden was increased from 1Ω to 10Ω. Additionally, saturation characteristics can be adjusted to control the time to saturation. To illustrate this process, the burden of CT1 was fixed at 10Ω, and the saturation point was set at 10 pu. Figure 5-14 illustrates the distortion in primary current when CT1 became saturated due to the fault, while Figure 5-18-a displays the change in flux in CT1 as a result of the fault.

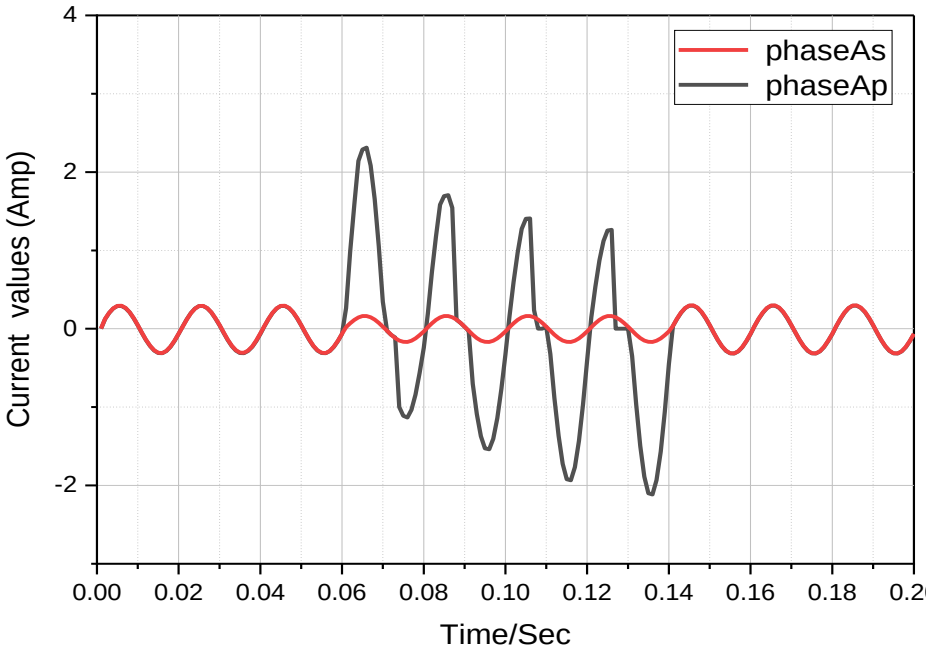

Figure 5- 14 Primary current distortion when CT1 is at saturation point.

At the time of the fault incidence, the flux reached saturation point at 0.0652 second, which is 0.0052 second following occurrence of the fault (0.0652 - 0.06 = 0.0052). After that point primary current began to distort. Distortion occurred only during the results of the simulation for the CT saturation within the fault zone, as well as the changes in alienation coefficients produced by the fault zone, are shown in Figure 5-15. Values of alienation coefficients are equal or less than 1 at 0.063 seconds

as a result of the fault. The fact that the fault was detected before the CT saturation indicates that it was detected after 3ms of the occurrence of the fault.

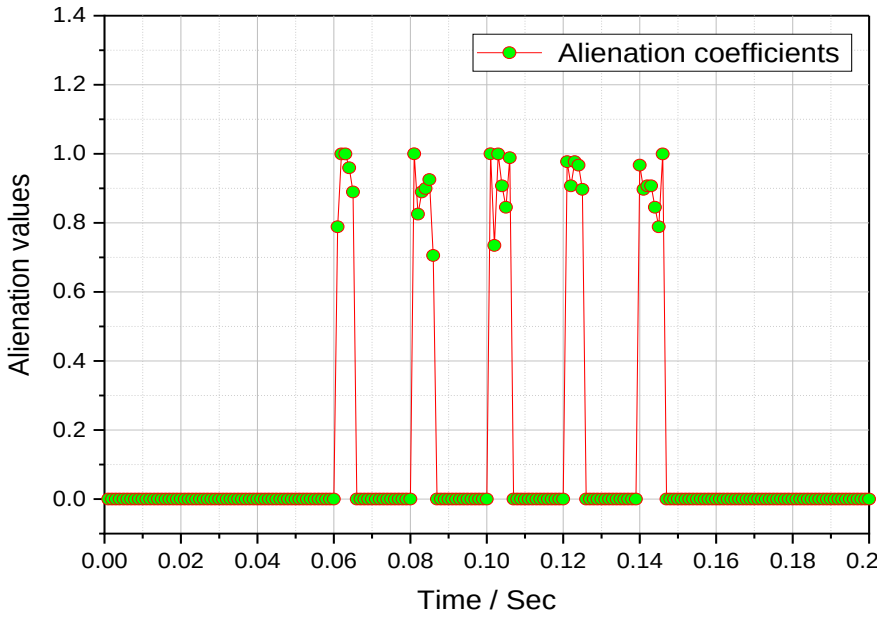

Figure 5- 15 Alienation coefficients values when Turn-ground fault occurred in phase A.

2. Phase to ground fault with saturation of CT

Figure 5-16 shows the distortion of the signal when a phase-to-ground fault occurred on the secondary side of the transformer, where all turns of phase A were grounded. This fault caused a complete loss of voltage in phase A and a significant reduction in the secondary current flowing through the load, as most of it flowed through the fault branch that had a low resistance of 5Ω. As a result of this fault, the flux in CT1 reached saturation, leading to the distortion of the signal.

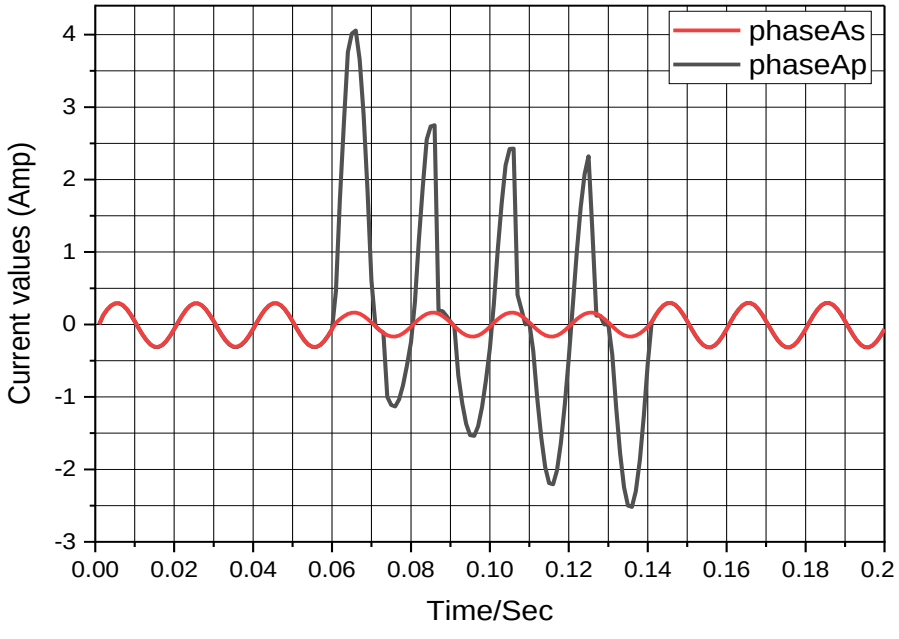

Figure 5- 16 Current signals when CT1 saturated due to phase-to-ground fault.

Figure 5-17 shows the effects of this fault on alienation coefficients. The fault had already been detected after 0.062 seconds which mean after 2ms after the fault occurrence. Because of this, the method was able to resolve this fault much faster than CT1 saturation occurred.

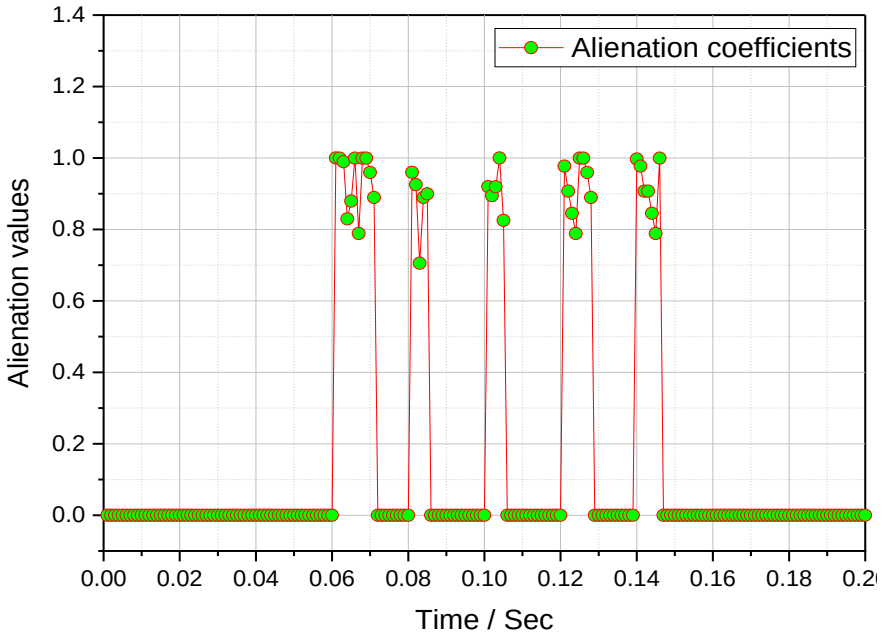

Figure 5- 17 Alienation coefficients values when phase-to-ground fault occurs in phase A.

3. External fault with CT saturation

Due to currents increasing on both sides simultaneously, if CT1 and CT4 were identical, they might both become saturated at once. The assumption is that CT1 isn't the same as CT4. Due to this, CT1 did not saturate, but CT4 which was closest to the fault did. Figure 5-18 shows that the secondary current started to distort after the flux in CT4 reached saturation at 0.068 seconds.

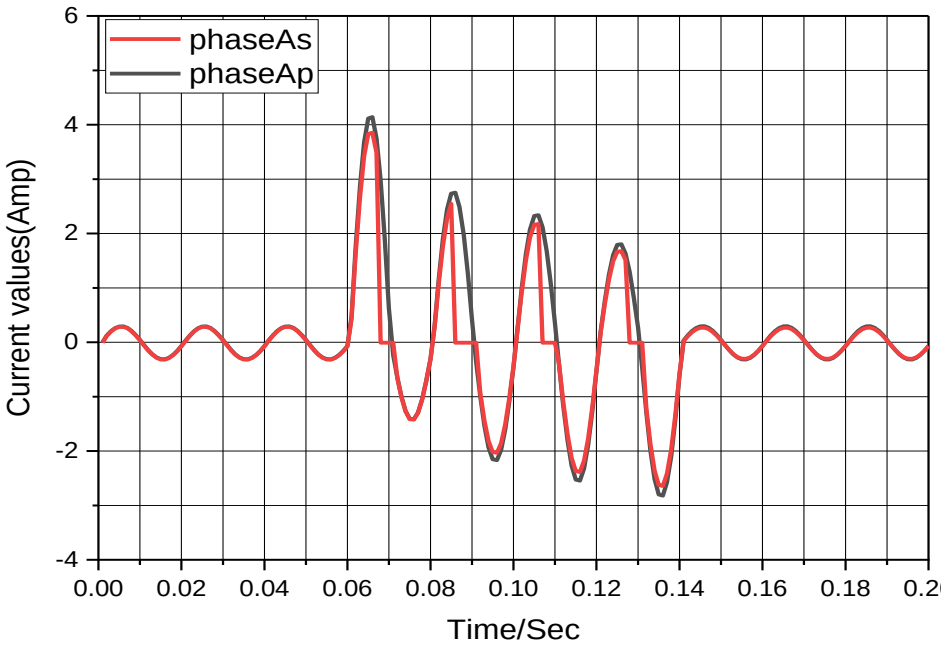

Figure 5- 18 Primary and secondary currents when CT is saturated due to external fault.

The primary and secondary currents increase together and follow both from the time of an external fault to the moment before the saturation of CTs. Due to the fault, the values of alienation coefficients will be equal to 0.005. The simulation of this case is shown in Figure 5-19. This only occurred when an external fault was present. The method, in this case will be considered as an external fault. In this case, the trip signal was not activated because of the occurrence of the fault as described earlier.

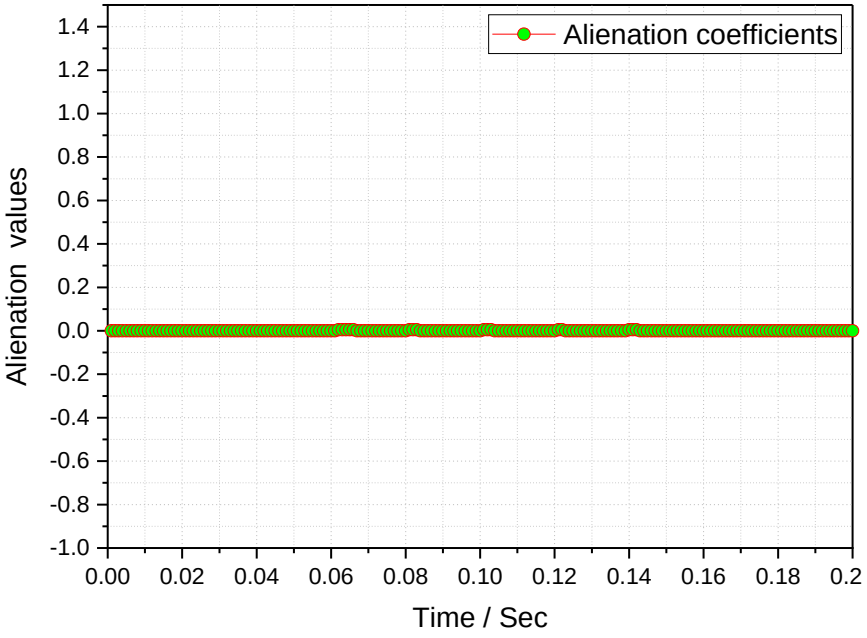

Figure 5- 19 Changes in Alienation coefficients values due to external fault with CT saturation.

5.3 The Laboratory Results

In the previous, it was shown that the faults were generated. MATLAB was used to test the proposed fault protection algorithm against each fault. This will be shown in the next section. As the same results had also been obtained from two other phases, only the results of phase A were shown here. It is worth pointing out that the actual current values in the figures are a little less than what is displayed in the figures due to the low pass filter in LabVIEW, which removed some of the data that is represented in the figures.

A. Turn to Turn Fault

There was an interturn fault along phase A's primary side due to short-circuiting of two turns. A LabVIEW program sent a command to switch relay, causing a fault to occur at approximately 8.176 seconds. In the previous section was mentioned that the great number of collected samples (100000 samples in 10 seconds) made presenting and processing difficult.

Thus, there were 2000 samples in the fault zone, starting at zero time and ending at 0.2 seconds. The samples for phase A began to be taken at zero from 2.624s, the samples for phase B began from 2.631s, and then for phase C, as it was closer to zero from 2.628s. As shown in Figure 5-20, the time shift between phases should be considered when specifying the time of the algorithm response.

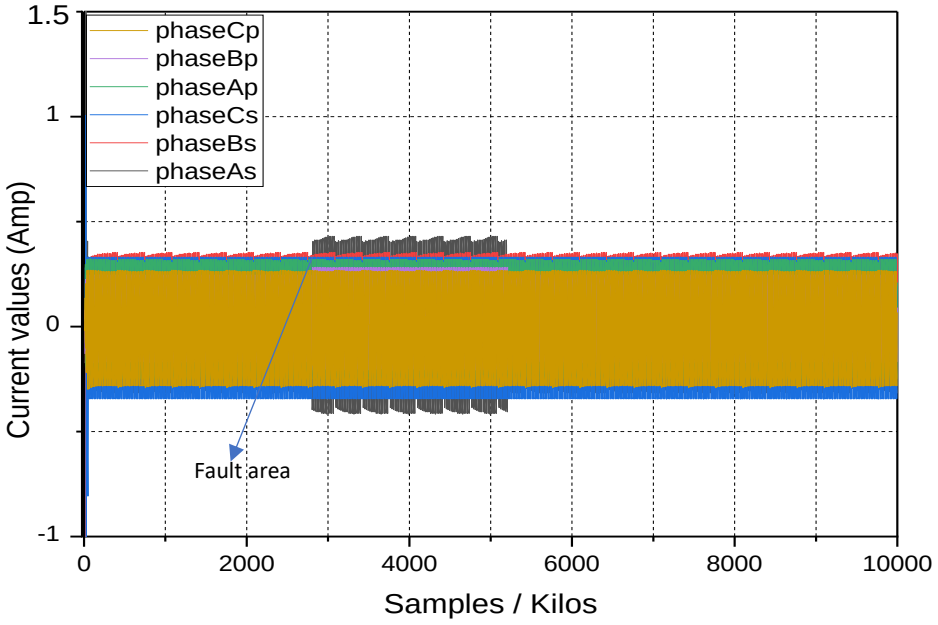

Figure 5- 20 Current signals data during creating fault (10k samples).

The samples taken are difficult to process. Hence, to clarify the fault period in the current signal, 2000 samples were taken in 0.2 seconds, as illustrated in Figure 5-21. These samples will be used to test the proposed algorithm against this type of fault in each phase.

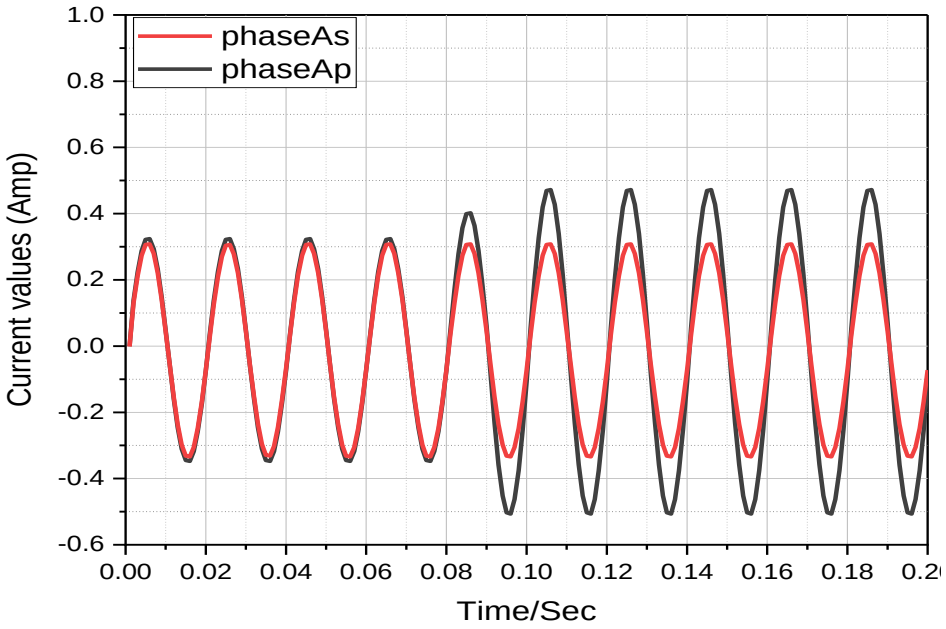

Figure 5- 21 Turn to turn fault-phase A.

Hence, this step means that the MATLAB program can determine the alienation coefficient of all phases at the same time. Each alienation coefficient is calculated between each of two corresponding quarter-cycles of the primary and secondary current for phase A. The value of Aa is equal and close to zero before the fault begins. In turn-turn faults, inception is greater than 0.09. So, this result clarifies that the values of Aa at a created fault is a suitable detector to define the type of fault location, as shown in Figure 5-22.

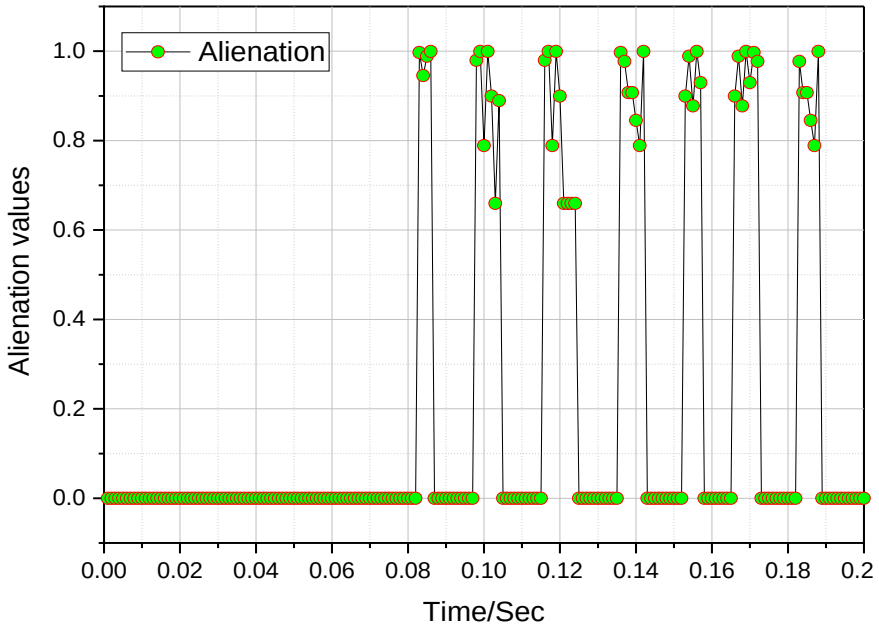

Figure 5- 22 Values of Alienation coefficients.

A fault has been detected at 0.083 seconds. Due to the fault generating at 0.08 seconds, the fault was detected at 0.083 - 0.08 = 0.003 seconds, i.e. After 3 milliseconds.

B. Turn to Ground Fault

Similar results as for the turn-to-turn fault were obtained during this stage, through choosing different turns on both primary and secondary sides to create a turn to ground fault in three phases of the transformer. The fault occurred at 3.8464 seconds on the secondary side of phase A, when the relay closed. Figure 5-23 shows the most significant change occurred in phase A current, which was the most defective phase.

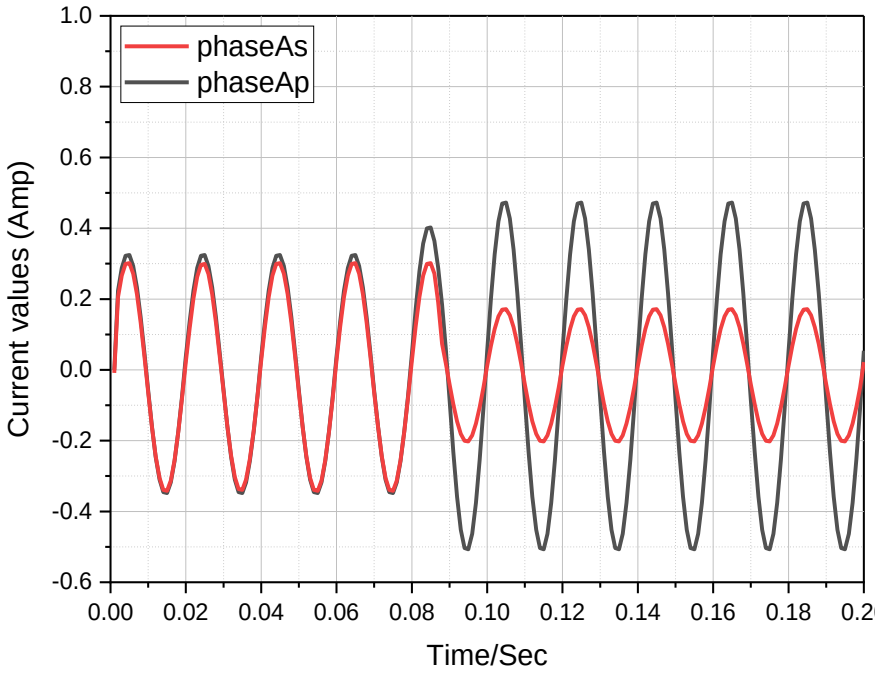

Figure 5- 23 Turn to ground fault-phase A.

The alienation coefficient is calculated between each of the two corresponding quarter-cycles of primary and secondary current for phase A. The value of Aa of phase A is close to zero before the fault beginning. In the turn-ground fault, the value of Aa is greater than 0.09 as shown in Figure 5-24. So, this result clarifies that the values of Aa at the created fault form a suitable detector to define the type of fault as a turn to ground fault.

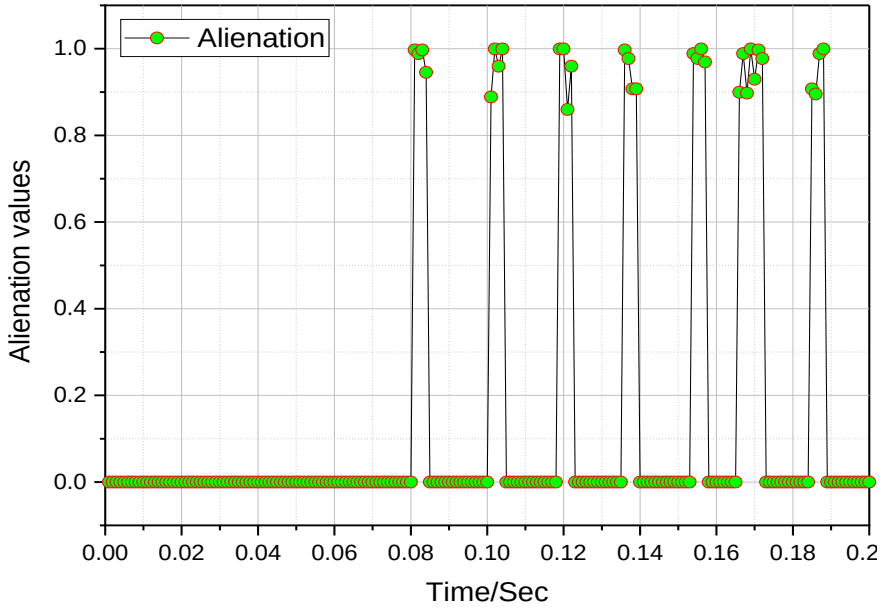

Figure 5- 24 Values of Alienation coefficients.

A fault has been detected at 0.082 seconds. Due to the fault generating at 0.08 seconds, the fault was detected at 0.082 - 0.08 = 0.002 seconds, i.e. After 2 milliseconds.

C. External Fault

An external fault was created outside the transformer protection zone and between the secondary side and the load. Therefore, the CTs on the secondary side measured the external fault current through locating CTs before the point of the external fault, as shown in Figure 5-25

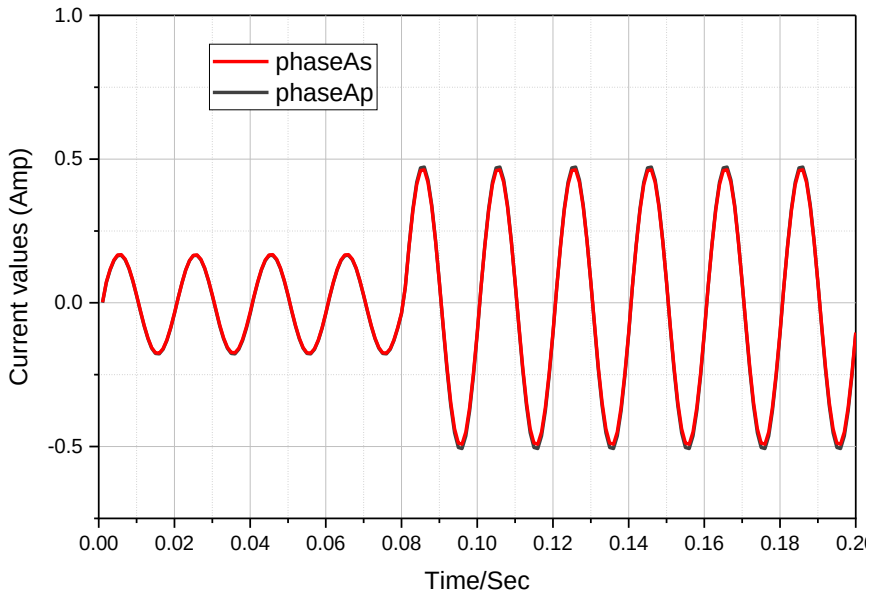

Figure 5- 25 External fault-phase A.

The alienation coefficient of phase A for the full duration of the fault zone is shown in Figure 5-26. Alienation coefficient Aa was reduced to less than 1 for the whole duration of the fault, because the same value of current increase occurred on both primary and secondary sides.

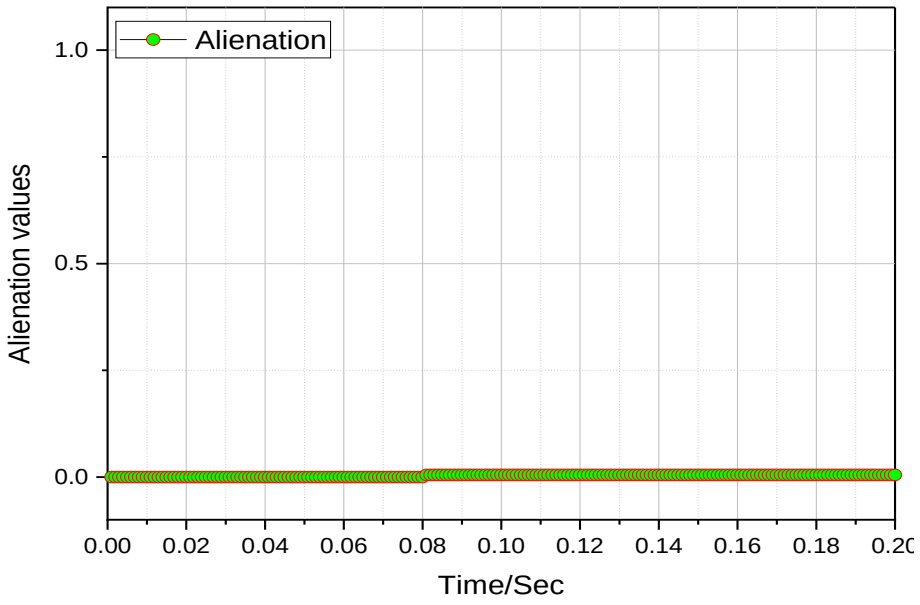

Figure 5- 26 Values of Alienation coefficients.

This happens only in the case of external faults or normal operation. In the case of this type of condition, the relay does not send a trip signal to the circuit breaker.

5.3.1 CT saturation case

The current transformer utilized in the experiment required a substantial current to reach saturation. Consequently, the faults were intentionally induced on the primary side of the transformer since this approach would result in higher fault currents. As these faults were the only ones generated in the secondary side shown in the previous section, the algorithm's response to them can also be shown here. There is always a delay between major external faults and the initial saturation of CT, indicating that even at major external faults, there is a delay. A high current is needed for the current transformer to be driven into saturation in the experiment. In normal conditions, this delay is between 3 and 5 ms, during which the primary current is not affected and is correctly transformed to the secondary side without any differential current. The differential current is assumed to occur simultaneously and instantaneously with internal fault occurrences. However, CT initially works properly. It has been tested to determine whether the method can detect faults before CT saturation begins. A proposed algorithm identified possible saturations of CTs due to faults.

1. CT saturation due to turn-ground fault

In order to increase the fault current, the 5 Ω protection resistor (R_f) was replaced with a 1 Ω. A fault occurred on the primary phase A side at the approximate position of 5.8571 second as shown in Figure 5- 27. As a result, CT1 is the most likely current transformer to have become saturated. Thus, to ensure that CT1 was saturated with current during the fault, the CT burden

was increased by setting a 1Ω resistance value. Since CT1 was saturated at the beginning of the fourth cycle, CT1 at the beginning of the fourth cycle was also distorted. In a similar way to that of the previous section, the LabVIEW program was also executed in 10 seconds. The region covering the selected case was also 0.2 seconds.

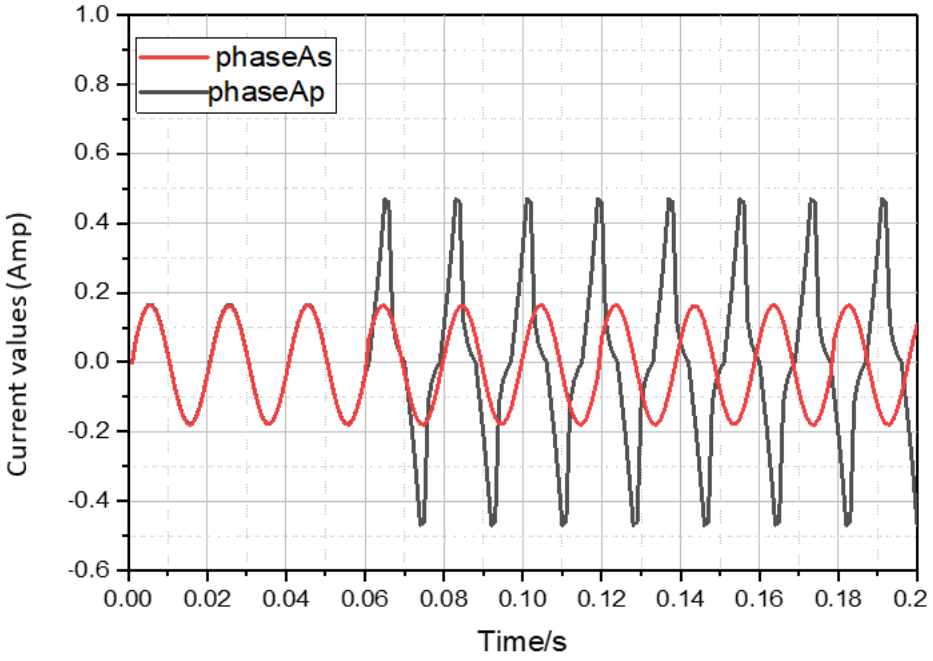

Figure 5- 27 CT1 saturation during turn-ground fault in phase A.

In Figure 5-28, Alienation coefficients are shown for the whole fault zone. The Alienation values after 0.062s increased up to 1 because of the fault. Since the algorithm had enough time to react to this fault, the saturation problem didn't occur immediately. The fault was detected through (0.062-0.06=0.002 second) after the occurrence of the fault, which corresponded to the time when the CT1 started to saturate. Hence, the algorithm was effective and fast enough to detect the fault before CT saturation occurred.

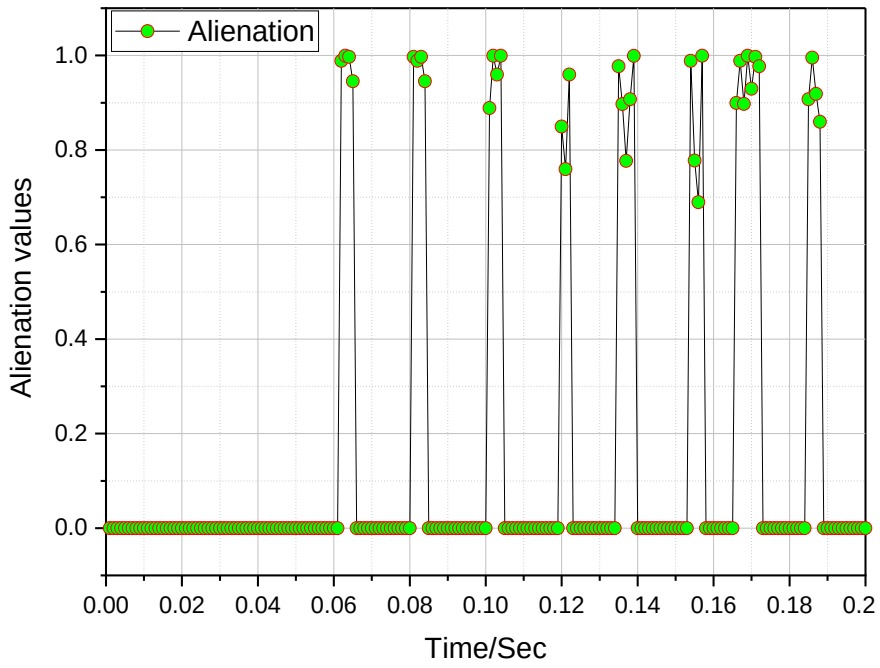

Figure 5- 28 Change in alienation coefficients due to CT1 saturation caused by turn-ground fault.

2. CT saturation due to single phase-to-ground fault

A fault occurred on phase A's primary side, where the secondary current should have been zero. Nevertheless, there was a secondary current flowing through to the load due to the presence of R_f. Nevertheless, the algorithm will respond more quickly when there is no or almost no secondary current. As can be seen in Figure 5-29, this fault occurred at the end of the third cycle, which resulted in saturation of the current transformer CT1 on the primary side, as well as distortion of the primary current. Furthermore, it is also very important to be aware of the fact that this fault occurred during the normal decrease of the current cycle.

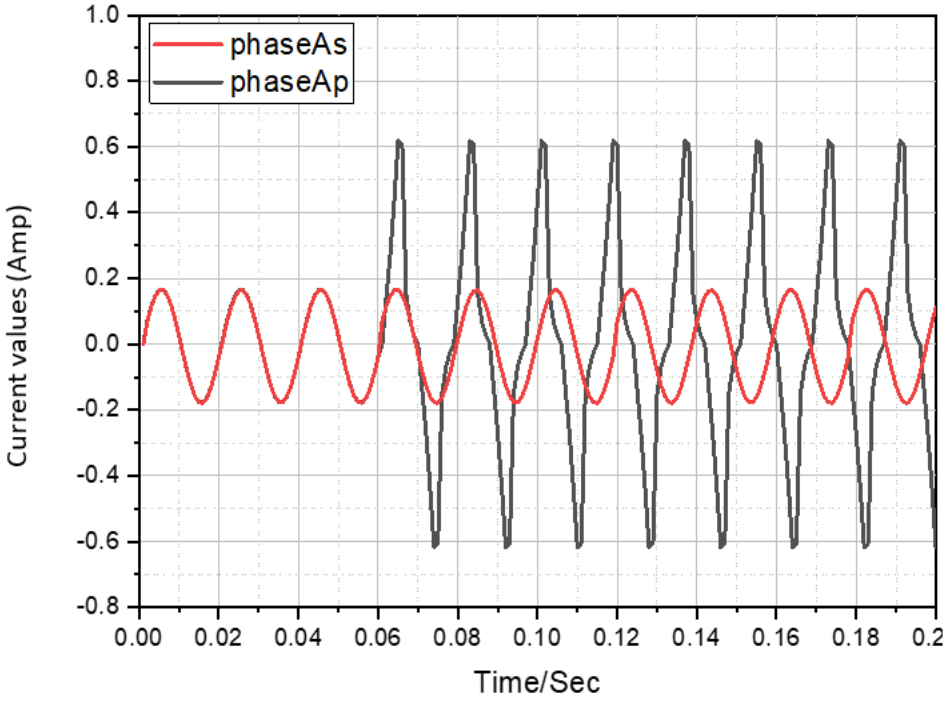

Figure 5- 29 CT1 saturation during single phase-to-ground fault in phase A.

Due to the primary side fault, the Alienation coefficients as shown in figure 5-30 increased to 1. This fault is supposed to cause the secondary current to drop to almost zero due to this fault as mentioned previously, and this can be seen by using MATLAB SIMULINK programs to model and test the fault. The fault was detected (0.0623-0.06=0.0023 second or 2.3 ms), i.e. at the point where CT1 had not yet reached saturation. As a result, the algorithm effectively dealt with the fault before the CT reached saturation.

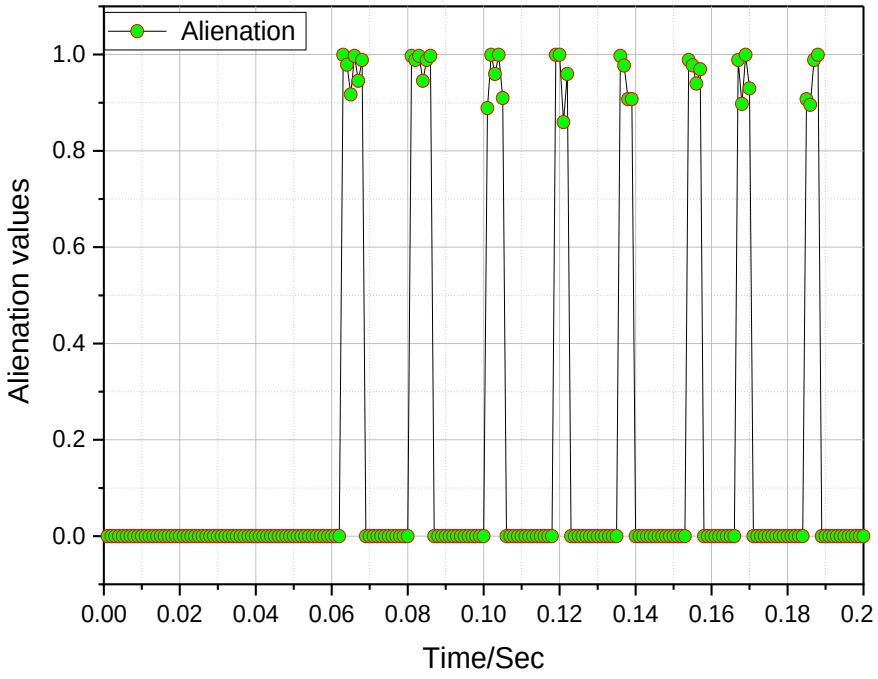

Figure 5- 30 Change in alienation coefficients due to CT1 saturation caused by phase-to-ground fault.

3.CT saturation due to external fault

In order to increase CT saturation, the external fault generated in phase A has also been increased by decreasing R_f to 2.5Ω. Since CT4 is closest to the fault point, it has been measured at approximately 26.21 Amp. Increasing burden to 4 Ω to see how the algorithm would react in this situation when an external fault occurred. The external fault occurred at 4.8 seconds. Figure 5-31 shows that the primary side was still functioning properly when CT4 on the secondary side was deeply saturated. The saturation of primary and secondary CTs does not happen simultaneously, which suggests that both CTs are not identical. then examined a case in which two CTs are identical.

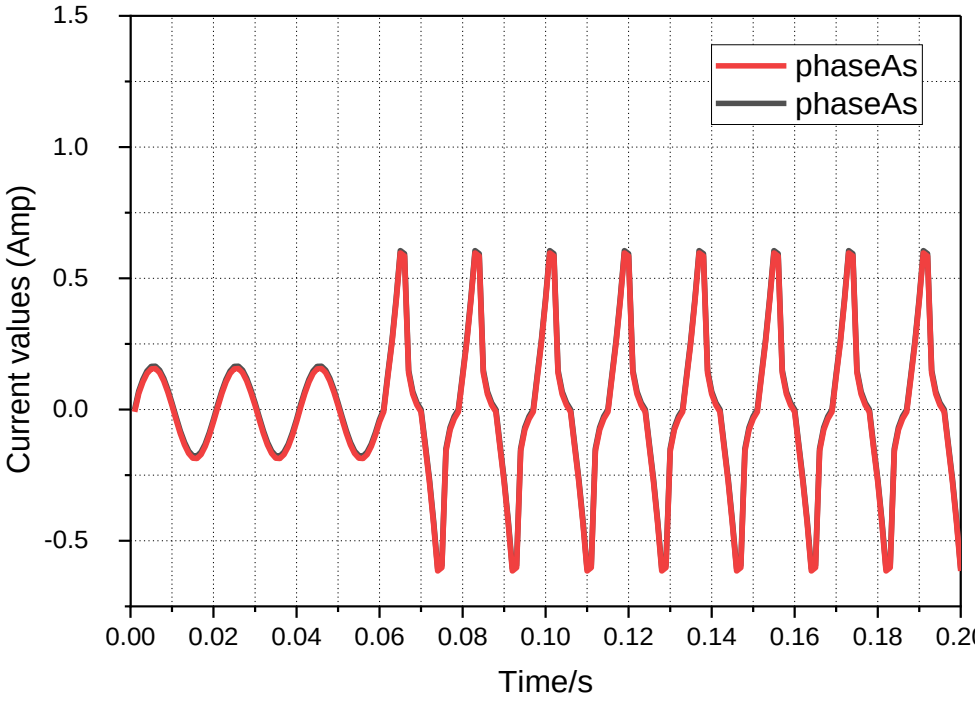

Figure 5- 31 Saturation of two identical CTs due to external fault in phase A.

According to Figure 5-32, phase A's Alienation coefficients only changed after an external fault occurred, which caused both CT4 and CT1 to be saturated simultaneously. The value of Alienation coefficients increased to less than 0.05 at 0.06. As a result of the algorithm's proper and quick processing, two CTs were simultaneously flooded, and the trip signal was not issued.

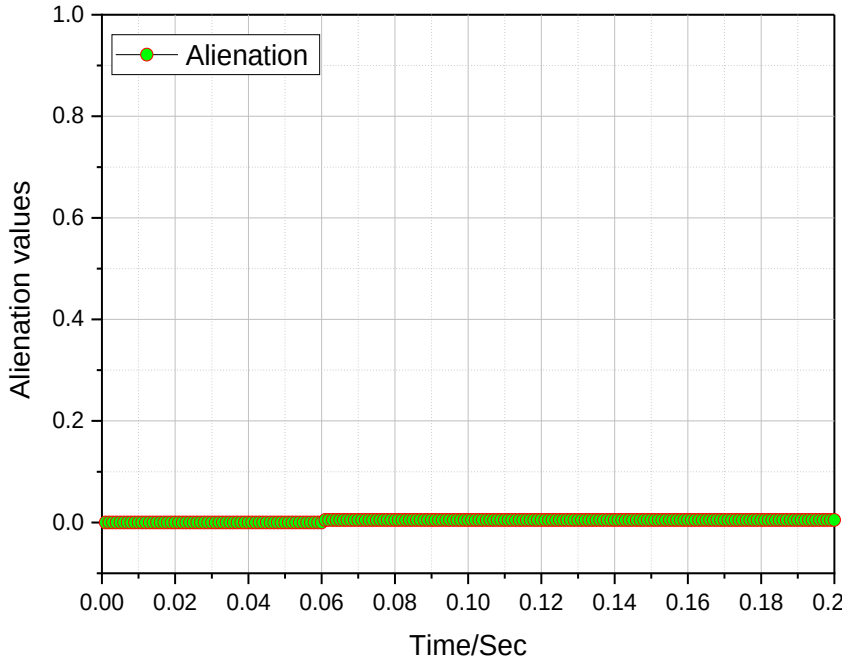

Figure 5- 32 Change in alienation coefficients due to saturation of two identical CTs.

5.4 Methods comparison

The approach described in [73] utilized discrete wavelet transforms (DWTs) to extract five scales from the current waveform. However, only coefficients from the first scale were used in the decision algorithm, as they were found to be adequate for distinguishing between internal and external faults. This study compared this method to a proposed alternative approach, as both methods were developed for the same objective. It used only 3.33% of total turns to create short-circuited turns in turn-turn faults, while 10% of total turns were used in the method comparison. The proposed method was found to be faster than DWT due to significant factors, allowing for the detection of faults in just 3ms, compared to 5ms with DWT.

5.5 Summary

The first part of this chapter demonstrates the testing of a protection method applied to a transformer that was modeled using MATLAB/SIMULINK. The technique was applied to the most common faults that occur in transformers to demonstrate how it works. During steady state, it has been shown that the technique is efficient and fast at detecting faults as it detects faults at maximum in 3ms depending on the type of fault and how severity of the fault it is. This method has been tested under situations of current overload as well. A big problem with differential protection systems is the saturation of transformers. The technique proved to be effective at solving this problem thanks to its high-speed action.

In the second part of this chapter, the effectiveness of the proposed fault detection and elimination technique was demonstrated through laboratory experiments conducted under various conditions, including steady-state operation. The results showed that the technique successfully detected and eliminated internal faults within a response time of 3ms. The method was also effective in differentiating between external and internal faults and overcoming the problem of CT saturation. The technique was able to detect turn-turn faults despite only two turns being shorted, and it is expected to be more efficient on larger transformers with more turns shorted. The method relies on waveshapes rather than current values, which makes it superior to overcurrent protection that trips based on current flow, whether caused by internal or external faults. A change in current due to a load change will not be considered an external fault, and the relay will remain stable.

CHAPTER 6

Machine learning algorithm for detecting and protecting three-phase power transformer

6.1 Introduction

In this chapter, a novel hybrid machine learning technique is proposed for the detection and protection of three-phase power transformers. The developed model is tested on various fault conditions, including inrush current and different types of current signal faults, on a laboratory-constructed transformer system that includes internal and external faults. The proposed hybrid model for fault detection in three-phase power transformers incorporates machine learning classifiers to identify faulty features via an optimal feature identification process. The data extraction process involves utilizing Discrete Wavelet Transform (DWT) and Orthogonal Matching Pursuit (OMP) to extract statistical characteristics from the samples. Additionally, the Bees algorithm (BA) is employed to create an optimized subset of the extracted features, thereby reducing the number of data points required for the model while simultaneously increasing its accuracy. The optimized feature set is used as input for three classification algorithms: K-Nearest Neighbor (KNN), Support Vector Machine (SVM), and Artificial Neural Network (ANN), to distinguish normal operating conditions from faults. The training process employs k-fold cross-validation. The proposed approach is compared with a similar approach using a genetic algorithm (GA) and the model is evaluated based on its specificity, accuracy, precision, recall, and F1 score. The experiment results show that the proposed model is suitable for fault identification in various conditions and types of faults.

6.2 Feature Extraction

The detection and continuous monitoring of mechanical equipment flaws constitute a particularly intricate and demanding endeavor within the realm of engineering and industrial maintenance. In this step, discrete wavelet transform (DWT) and orthogonal matching pursuit (OMP) were used together to extract features using MATLAB simulation software version R2021a, 1-D for DWT and 1-D wavelet for OMP. In this method, DWT and OMP are combined to extract phase differences.

DWT is employed to decompose the initial phase difference. In the next step, using the OMP pursuit method to reconstruct the phase difference using the usable components.

6.2.1 Discrete Wavelet Transform (DWT)

One of the most common types of wavelet transforms is the discrete wavelet transform. Using the time-frequency domain, an analytical method has been proposed for the processing of raw signal data [86]. Since DWT employs sub band coding, it is a faster analysis method than continuous wavelet transformations because of its sub band coding. A digital signal can also be represented in time using DWT, as it utilizes digital filtering techniques to extract the time scale of the signal. As part of the signal analysis process, a signal is passed through a variety of cutoff frequencies at various scales. DWT recently has proven its ability to extract initial characteristics in numerous applications of three-phase power transformer current failure detection among the various types of wavelets transforms [87]. By analyzing the approximate signal at each level of decomposition, the wavelets are decomposed in n levels of decomposition. This procedure is repeated until level n is reached and an analysis of each level of wavelet decomposition is performed. With wavelet decomposition, the information contained in the signal is split into two parts: the first is an approximate difference and the second a detail difference. This decomposition process further splits up A1 to create A2 and D2 as a result of further decomposition as shown in figure 6-1. Continue the decomposition until the desired level N is reached, then continue until the An, Dn level is reached.

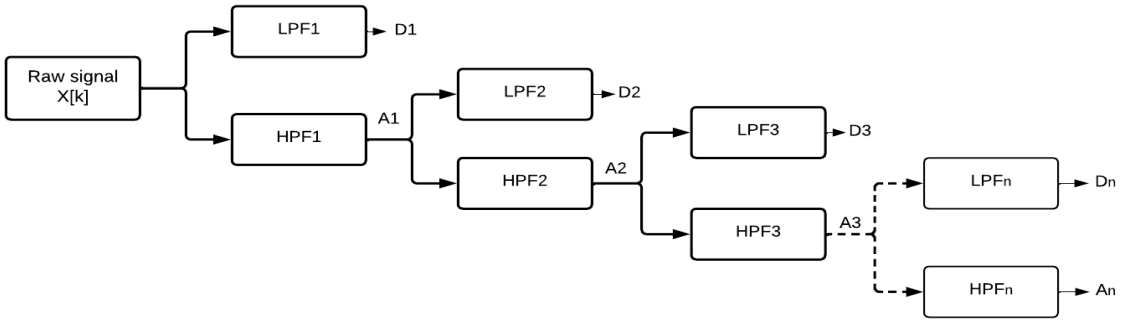

Figure 6- 1 Discrete wavelet decomposition process [86].

Referring to the figure, the steps of the decomposition process are represented by A1, A2, and A3. Additionally, LPF1, LPF2, and LPF3 indicate low pass filters, while HPF1, HPF2, and HPF3 represent high pass filters. As mentioned earlier, the decorrelation properties of DWT are a significant reason why it is an efficient method. On the other hand, it does have some drawbacks. DWT has a number of drawbacks due to the mother wavelet selection [88]. For this study, five

mother wavelets [89] were used: db7, sym3, coif4, bior6.8, and rbior6.8. Analysis of the current signals has been done through MATLAB simulations. The decomposed data has divided into six levels. Thirty features were retrieved using five of the five-mother wavelets independently in the time-frequency domain. Based on these data, the datasets were developed.

6.2.2 Orthogonal matching pursuit (OMP)

An orthogonal matching pursuit (OMP) technique is used to reconstruct a high-dimensional sparse signal based on a few noisy linear measurements. The OMP algorithm selects the column with the highest correlation with the current residuals at each stage using iterative greedy optimization. Essentially, OMP consists of combining multiple basis functions and feature waveforms to produce an overly complete vocabulary of atoms [90]. During a process of gradual iteration, the optimal atom is found to match the vibration signal. To achieve a global optimal solution in each iteration, the atoms chosen must be orthogonal. By projecting the signal in the space created by the processed atoms, the component and residue on the atoms are determined. To obtain the global optimal solution, it is necessary to ensure that the leftover signals are orthogonal to all of the selected atoms. MATLAB simulation represented five different scenarios (Sym4-Lev5, Wpsym4-Lev5, Dct, Sin, and Cos) for processing all current data signals from phase A. These are respectively symmetric wavelets with five levels and four vanished moments, wavelet packet-based symmetric with four vanished moments and five levels, discrete cosine transforms, sine sub dictionary, and cosine sub dictionary. For the simulation, several OMP parameters were carefully chosen, including a maximum iteration level of 100 and a maximum relative error of 0.01 percent. This proposed approach includes eight features: mean, median, standard deviation, median absolute deviation, mean absolute deviation, L1 norm, L2 norm, and maximal norm.

6.3 Feature Selection Algorithms

An important part of data mining is the selection of features from a large dataset or the selection of feature subsets from a large dataset, especially in high-dimensional data sets. The selection of subsets of features is one of the steps that are involved in machine learning, as it involves applying a learning algorithm to a set of features. A subset that is optimal contains the smallest number of dimensions that make the most contribution to accuracy, while the rest of the subset is discarded. As this step helps to eliminate the curse of data dimensionality, it becomes an important stage in data pre-processing. To improve the processing time and classification accuracy, two feature selection methods have been used to reduce the processing time and to improve the classification accuracy in the present study.

6.3.1 The Bees Algorithm (BA)

The 'Bees' algorithm was created by Pham in 2006. The Bees algorithm is a bee-inspired algorithm that is similar to swarm intelligence, mathematical intelligence, and metaheuristics in general. The strategy is frequently used to identify an optimal answer while avoiding problems with local optima. Because it combines neighbourhood search and random search in a unique way, this technique is excellent for combinatorial and functional optimizations. The BA is simple to implement, and it is efficient in finding the most optimal solutions [91]. A simple Bee Algorithm is depicted in Figure 6-2.

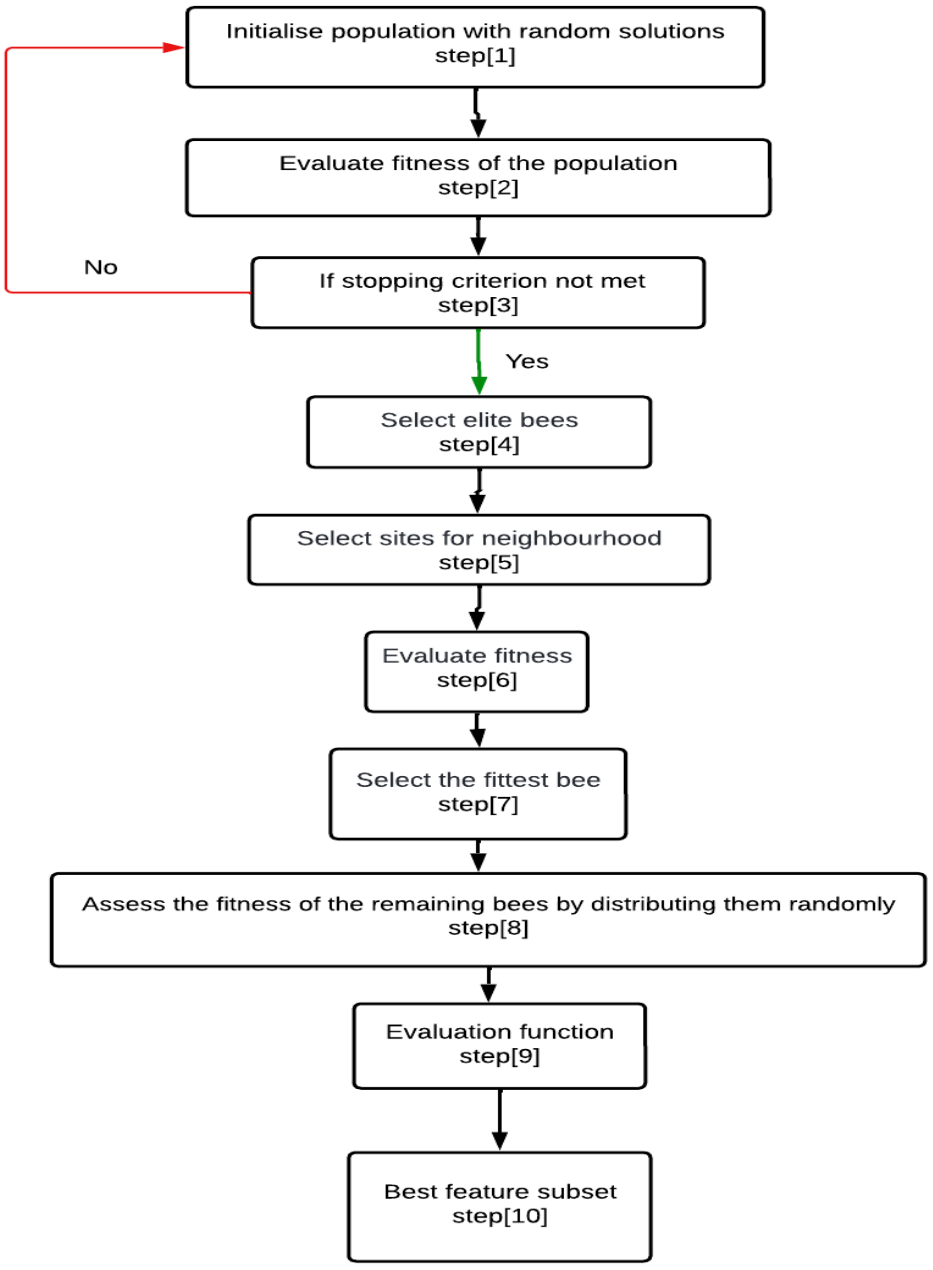

Figure 6- 2 Basic BA flowchart.

6.3.2 The Bees Algorithm Operators

The Bees Algorithm is a swarm intelligence optimization algorithm that simulates the foraging behavior of honeybees [91].

The steps involved in a Bees algorithm for optimization typically include.

1. Initialize population with random solutions:

In this step, a population of potential solutions (bees) is randomly generated to start the algorithm.

2. Evaluate fitness of the population:

Each potential solution in the population is evaluated for its fitness (i.e., how well it solves the problem at hand).

3. If stopping criterion not met. If yes go to the next step; if no, go to the first step:

The algorithm checks whether the stopping criterion has been met (e.g., maximum number of iterations, desired fitness level achieved). If the criterion has not been met, the algorithm proceeds to the next step; otherwise, it stops and returns the best solution found.

4. Select elite bees:

In this step, the best performing bees in the population are selected to be preserved for the next generation. These bees are considered "elite" because they have the highest fitness and are expected to contribute to the success of the next generation.

5. Select sites for neighborhood:

The algorithm selects a set of sites around each elite bee. These sites represent the neighborhood of each elite bee and will be used to search for new solutions in the next step.

6. Evaluate fitness:

The fitness of the potential solutions in the neighborhood of each elite bee is evaluated.

7. Select the fittest bee:

The best performing bee in each neighborhood is selected and considered for inclusion in the next generation.

8. Assess the fitness of the remaining bees by distributing them randomly:

The algorithm randomly distributes the remaining bees in the population to different neighborhoods to evaluate their fitness.

9. Evaluation function:

The evaluation function is the fitness function used to evaluate the performance of the potential solutions. It is the function that determines the quality of the solution.

10. Best feature subset:

The Bees Algorithm can be used to find the best feature subset for a given classification or regression problem. The subset of features that results in the best performance (as determined by the fitness function) is considered the best feature subset.

6.3.3 Genetic Algorithm

Developed by Holland in 1995, the genetic algorithm-based features selection approach achieves a balance between computational cost and optimal selection using a heuristic search. This method can be easily parallelized in a computer cluster and can handle tons of data without the need for any previous knowledge of the project [91].

6.3.2.1 Genetic Algorithm Operators

Genetic algorithms operate for optimization by iteratively improving a population of potential solutions through a process of selection and reproduction based on their fitness values, eventually converging to an optimal solution [92].GA operates as follows:

The steps involved in a genetic algorithm for optimization typically include:

1. Initialization: A population of potential solutions is randomly generated.
2. Evaluation: Everyone in the population is evaluated using a fitness function that measures how well it solves the problem.
3. Selection: Individuals with higher fitness values are more likely to be selected for reproduction. There are various selection methods, such as roulette wheel selection, tournament selection, and rank selection.
4. Crossover: The genetic material of two or more selected individuals is combined to create a new individual. Crossover is performed at randomly selected points in the genes.
5. Mutation: The new individual may undergo random mutations that introduce new genetic information into the population.
6. Replacement: The new individual replaces a less fit individual in the population.
7. Termination: The algorithm stops when a satisfactory solution is found or a stopping criterion is met, such as reaching a maximum number of generations or a minimum acceptable fitness level.

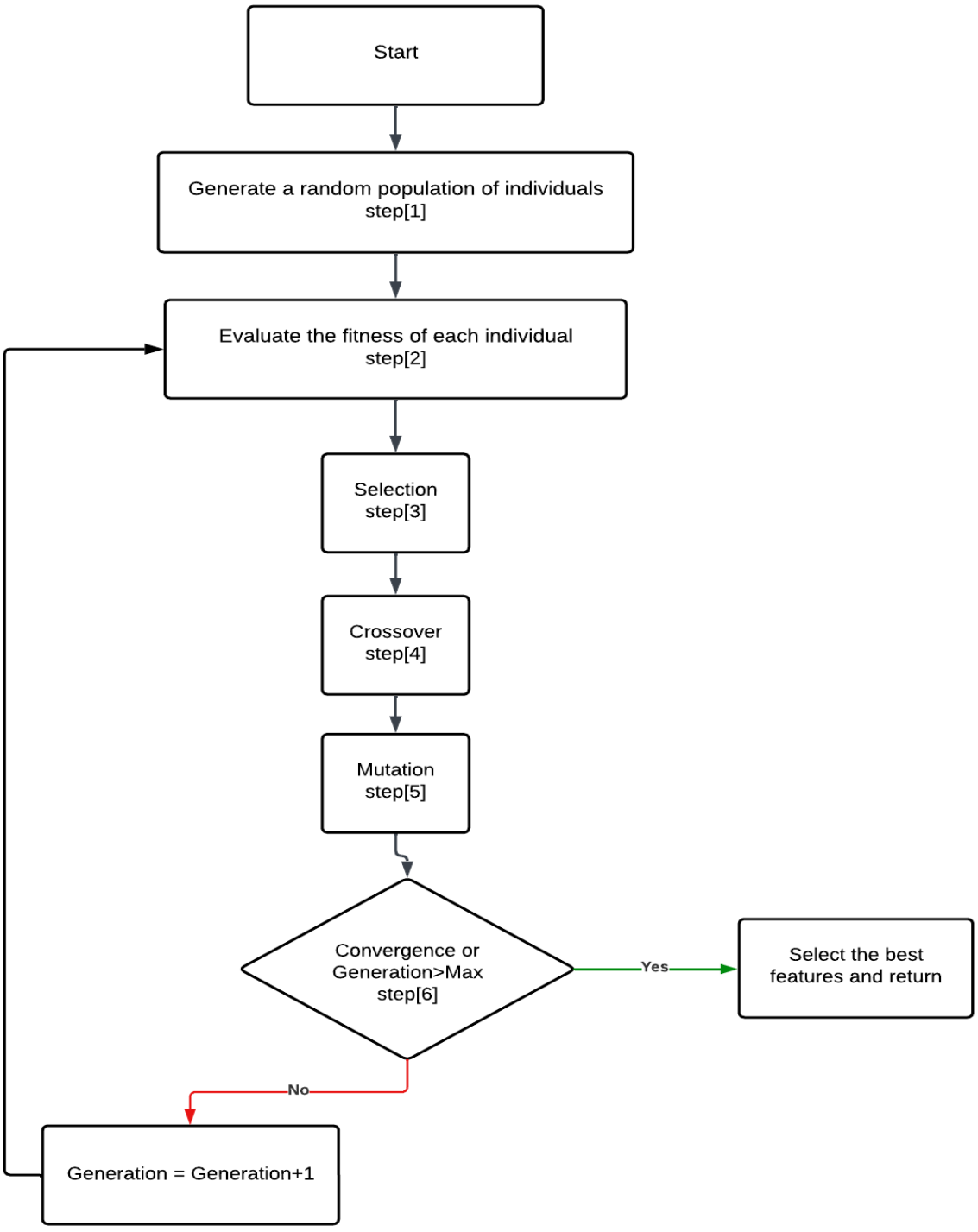

Figure 6- 3 A flow chart of genetic algorithm optimization.

Figure 6- 3 illustrates the iterative process of a genetic algorithm, where each iteration represents a generation of the population. The process is repeated until a satisfactory solution is found or a stopping criterion is met.

6.4 Classification Techniques

Various classification methods are illustrated in this section, being thoroughly explored in the subsections that follow.

6.4.1 K-Nearest Neighbour,(KNN)

KNN is the most basic machine learning algorithm. Using a similarity function to categorize unknown situations, it employs a correlation technique [93]. A predefined integer (k), which could be real or imaginary, is used to divide the dataset into clusters for training this classifier. As part of the iterative classifier process, the central data point is the centroid of the cluster. As a result of the emerging classifier, a random cluster of clusters is generated and the centroid value is continuously changed until it becomes stable. This model is then used to classify new data [94].

6.4.2 Support Vector Machine,(SVM)

The SVM is another type of classifier commonly used for classification and regression. This algorithm separates datasets into two categories: negative and positive. The proposed dataset is also trained based on statistical learning, which is expressed as a support vector [95]. Based on categorization information, the algorithm constructs the hyperplane. By creating a hyperplane, positives and negatives are spaced optimally. Kernel functions may be used for nonlinear transformations and for SVM when there are separable and non-separable features in a dataset. Multi-feature mappings make a nonlinearly separable object linearly separable [96]. This has been accomplished using linear kernels, polynomial kernels, and Gaussian radial basis functions (RBF).

6.4.3 Artificial Neural Networks,(ANN)

Artificial neural networks have been widely used in the power system protection field since 1994 since this problem falls within the current waveform pattern identification class. An ANN is typically used for pattern recognition, image processing, power quality analysis, and data compression, among other things. The ANN technique's non-algorithmic parallel-distributed architecture for information processing and the ability to make intelligent decisions are the major advantages over the conventional method. Several recent works have examined the feasibility of applying ANN to protect power transformers [97]. It is critical to note, however, that the ANNs used in this previous study were taught for a specific transformer system and would need to be retrained. In addition, the algorithms used for feature extraction are based on either the time or frequency do-main signals, rather than both, which is critical for accurately distinguishing between an internal fault and an inrush current [98].

6.5 Proposed Method

Figure 6-4 shows a flow chart illustrating the major steps of the proposed application as they will be applied to carry out this project. To create a large dataset (3000 samples), the current signals are captured using specialized sensors under various operating situations. For each signal collected, OMP and DWT are used to retrieve features.

It is crucial to determine the right number of discriminative features. The computational complexity of classification algorithms is excessive when they use many features unless certain methods of reducing data dimensionality are employed before classification. Thus, this research proposes BA-based feature selection to reduce data dimensionality by utilizing the acquired feature matrix for discriminant feature selection. Furthermore, GA is also used to compare results to those of BA. The fitness function of these methods is based on the classification error of the KNN algorithm.

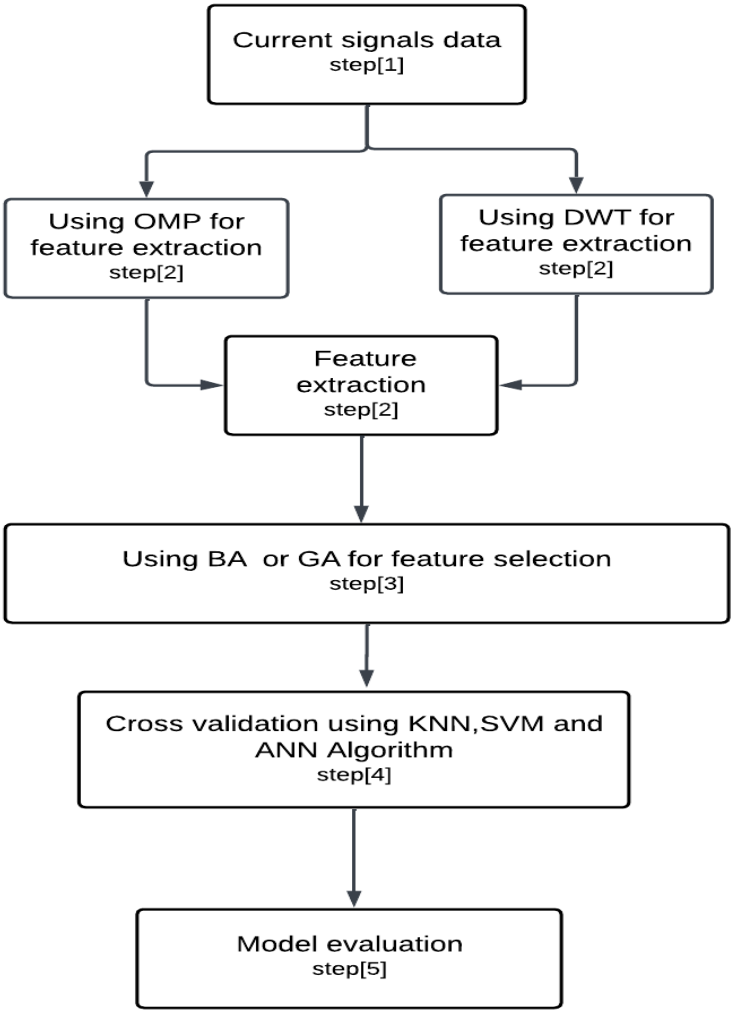

Figure 6- 4 Flow chart for proposal method.

The flow chart in Figure 6-4 illustrates the proposed application involves the following major steps:

1. Data Collection: Current signals are captured using a data acquisition card (6211) under various operating situations.
2. Feature extraction: The next step is to extract relevant features from the signals using two techniques: Discrete Wavelet Transforms (DWT) and Orthogonal Matching Pursuit (OMP).
3. Feature Selection: Feature selection: After feature extraction, the dataset is further reduced by selecting the most relevant features using two optimization algorithms: bat algorithm (BA) or genetic algorithm (GA). These algorithms identify the subset of features that have the greatest impact on fault classification accuracy.
4. Cross-validation: In order to evaluate the effectiveness of different machine learning algorithms for fault classification, the dataset is split into training and testing sets and subjected to cross-validation. Three algorithms are used: k-nearest neighbors (KNN), support vector machines (SVM), and artificial neural networks (ANN)
5. Model evaluation: Finally, the accuracy of each algorithm is evaluated based on its ability to correctly classify faults in the testing dataset. This evaluation provides a measure of the effectiveness of the pipeline in identifying and diagnosing faults in electrical equipment or machinery.

In order to classify faults, the feature matrix obtained through the feature selection process is input into three classifiers using machine learning. The classifiers are then trained using cross-validation. The training was repeated five and ten times with cross-validation techniques to fine-tune the model and ensure consistency in the results.

6.6 Assessment of the Model

The model's robustness was assessed using a variety of performance criteria, being evaluated using the F1-score, specificity, overall accuracy, prediction, sensitivity. In addition to the F1-score, specificity and accuracy indicate the performance of the model in terms of class assignment. As a positive class, precision and sensitivity measure how appropriate the model's error type is.

6.7 Results

The proposed application is evaluated in this section based on current signal experimental data. By feeding the obtained signals to the classification algorithms with the extracted features used in conjunction with the proposed strategy for selecting discriminative features, the fault detection procedure is improved. Instead of fourteen features, the Bees algorithm (BA) selected twelve

features with the best cost function, 0.083, for building the final feature matrix when the current signal was applied, as shown in Figure 6-5.

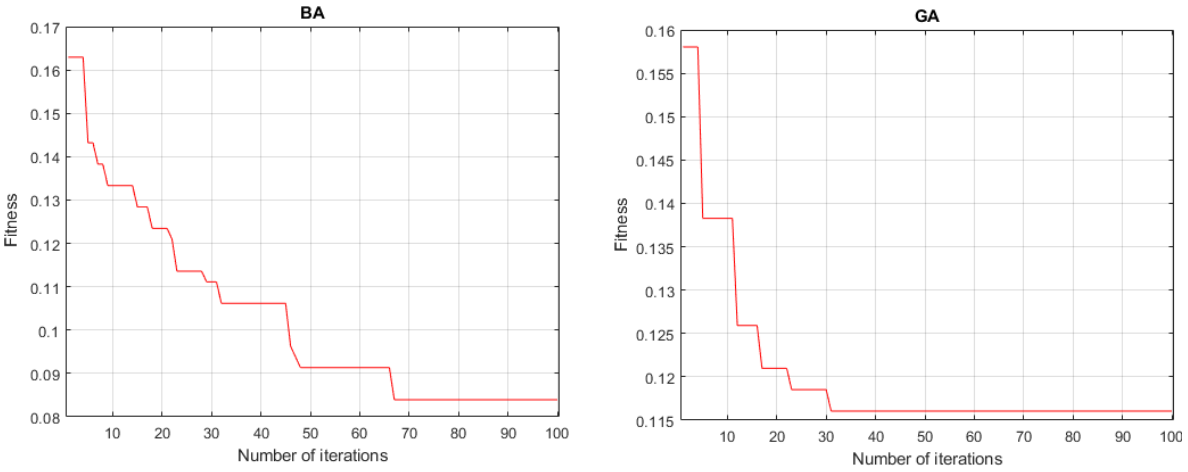

Figure 6-5 Loss curves of Bees and Genetic algorithm.

To compare the performance and superiority of the Bees algorithm technique, a similar test was run using the genetic algorithm. The original features were replaced with twenty-one features based on the implementation of GA using transformer current data. According to Table 6-1, BA selected 12 features based on the loss curve shown in Figure 6-6. GA selected 21 features based on the signal. A display of the classification results using feature selection based on BA and GA are shown in Table 6-2 and Table 6- 3 and Table 6- 4 and Table 6-5.

Table 6- 1 Number of selected features

Algorithms	Current signal data
BA	12
GA	21

In the following two steps, three machine learning algorithms, KNN, SVM, and ANN, were used for classifying the optimum feature sets selected by the BA and GA into their respective classes.

Table 6- 2 Results of 5-FoldCross-Validation with BA

5- Fold Cross -Validation	**SVM**	**ANN**	**KNN**
Accuracy	93%	62%	86%
Specificity	97%	83%	91%
Precision	85%	58%	83%
Sensitivity	85%	62%	81%

F1-Score	77%	57%	76%

Table 6- 3 Results of 10-FoldCross-Validation with BA

10-Fold Cross-Validation	**SVM**	**ANN**	**KNN**
Accuracy	96%	76%	93%
Specificity	98%	88%	91%
Precision	88%	74%	88%
Sensitivity	87%	72%	87%
F1-Score	78%	64%	85%

Table 6- 4 Results of 5-FoldCross-Validation with GA

5- Fold Cross -Validation	**SVM**	**ANN**	**KNN**
Accuracy	89%	59%	82%
Specificity	86%	85%	97%
Precision	75%	64%	82%
Sensitivity	87%	57%	75%
F1-Score	87%	58%	76%

Table 6- 5 Results of 10-FoldCross-Validation with GA

10-FoldCross-Validation	**SVM**	**ANN**	**KNN**
Accuracy	91%	73%	85%
Specificity	89%	70%	84%
Precision	86%	69%	78%
Sensitivity	77%	68%	77%
F1-Score	75%	71%	81%

When training the model using 10-fold cross-validation, the support vector machine (SVM) achieved the highest classification accuracy which was 96% using the combined with Bees algorithm (BA). When the model was trained with 5-fold cross-validation the SVM achieved again the highest accuracy by 93%. Additionally, the accuracy of the classification using the ANN classifier was 62 % and improved to 76 % by using 5-fold cross-validation. Furthermore, Tables 6-2,6-3 and 6-4,6-5 show that 10-fold cross-validation gave improved results for both Bees and GA algorithm when compared to 5-fold cross-validation. When combined with SVM and KNN

classifiers, BA produced improved results for all metric measurements. BA-ANN and BA-KNN, on the other hand, can produce identical sensitivity results and greater classification accuracy for the same data. Consequently, GA–SVM outperformed BA–SVM in terms of accuracy, sensitivity, and F1-score measures. The findings of this study provide strong evidence for the effectiveness of the proposed fault classification model, which utilizes current signals and outperforms other existing approaches. To extract the statistical characteristics of the samples, Discrete Wavelet Transform (DWT) and Orthogonal Matching Pursuit (OMP) were employed as one method for data extraction from data. The model uses the time-frequency domain to analyze the data, applying discrete wavelet transform and orthogonal matching pursuit to recover 40 distinct features per signal. The use of the Bees Algorithm (BA) and Genetic Algorithm (GA) for feature selection is crucial in reducing data dimensionality while decreasing computational complexity, which significantly improves classification performance. It is noteworthy that the BA outperformed GA in selecting fewer features, which led to higher classification accuracy. Additionally, applying a 10-fold cross-validation strategy to train the proposed models could further enhance the classification accuracy. These findings suggest that the proposed model can be a highly effective tool for fault classification in various applications, including industrial systems, power systems, and biomedical signal processing. Therefore, this study contributes to the development of effective fault classification methods that can improve the reliability and safety of various systems.

6.8 Summary

This chapter has presented a novel hybrid model which diagnoses faults in the three-phase power transformer and demonstrates its application. Algorithms were applied to optimize selection of discriminating features and thus enhance fault detection performance, and a comparison of the number of features assessed was made. The proposed application's performance was enhanced by some optimization algorithm-based selection of the discriminative features. A comparison was made with the number of executed features based on the model. A cross-validation strategy was used to train three machine learning classifiers to detect the faults that may happen in the three-phase transformer. To investigate the robustness of the proposed model, simulations of different cases of the transformer with inrush current from faults conditions were applied. Concerning the time-frequency domain, discrete wavelet transform and orthogonal matching pursuit were applied to recover 40 distinct features per signal, and the volume of necessary data was reduced by using the bees algorithm, 12 features of the current signal were chosen, while the genetic algorithm, selected 21. For fault type detection, the study applied 3 machine learning classifier approaches, ANN, SVM and KNN, training these in fault diagnosis based on the features selected, and 10-fold

cross-validation was applied. The classifiers were found to be satisfactory, a result that points to the potential of the proposed model for use in fault detection and classification. The model was validated for effectiveness by comparing optimization algorithm-based feature selection performed on the same dataset. The findings of this comparison show that the proposed strategies with fewer statistical features give comparatively more accurate results, and that the bees algorithm gave greater total accuracy while selecting fewer features. When BA was combined with an SVM classifier, maximal accuracy 96% in fault classification was seen at high accuracy on evaluation based.

CHAPTER 7

New method to detect the inrush current, internal and external fault in Three-Phase Power Transformers by Using SVM

7.1 Introduction

The goal of this chapter is to propose a new approach to the classification of current signals in the 3-phase transformer, distinguishing between inrush currents, internal faults and external faults. Support Vector Machine (SVM) creates the basis for the approach described here, and two types of SVM classifiers are employed for the analysis, SVM1 was used to identify faults and inrush currents and SVM2 was used to identify faults as they occurred either internally or externally. It has the potential to be applied to any type and position of fault, regardless of how long has passed since the fault started. Through this process, the identification of fault types for internal faults within transformers is enhanced, including turn-to-turn and turn-to-ground faults, which commonly occur within transformers. The purpose of this study is to compare this classification system with classifiers based on artificial neural networks (ANN), and to find that the SVM classifiers are more reliable and provide faster responses than the artificial neural networks.

7.2 Feature Extraction

In this part, signal processing techniques matching Discrete Wavelet Transform (DWT) was implemented for feature extraction. By Implementing the Discrete Wavelet Transform (DWT) for feature extraction, it is aimed to leverage its capabilities in uncovering relevant information from mechanical equipment signals, ultimately contributing to more effective defect detection and monitoring processes.

7.2.1 Discrete Wavelet Transform (DWT)

The discrete wavelet transform (DWT) has been used as an effective feature extraction tool to extract time-frequency features similar in shape to that of a particular wavelet function. It therefore

has an advantage over other feature extraction methods that operate in only one domain, such as the Fourier transform, or autoregressive modelling [99].

7.3 Feature Selection Algorithms

The objective of utilizing feature selection techniques is to eliminate redundant or irrelevant features from the data without compromising the information content [100]. Redundancy and irrelevance are two distinct concepts, where a correlated feature may be considered redundant if there exists another highly correlated feature with it. In this proposal, the Bees Algorithm is employed for feature selection as mentioned in chapter 6.

7.4 Support Vector Machine (SVM)

The utilization of Support Vector Machines (SVM) serves the fundamental objective of identifying a linear separating hyperplane in such a manner that the margin of separation between two classes within the data space is maximized [101-102], as elucidated in Figure 7-1. The SVM exhibits commendable performance characteristics, demonstrating effectiveness not only in scenarios where datasets are linearly separable but also in cases where such linear separability does not hold. Furthermore, the SVM algorithm has garnered notable acclaim for its ability to deliver robust results even when the dataset size is limited, underscoring its versatility and reliability [103].

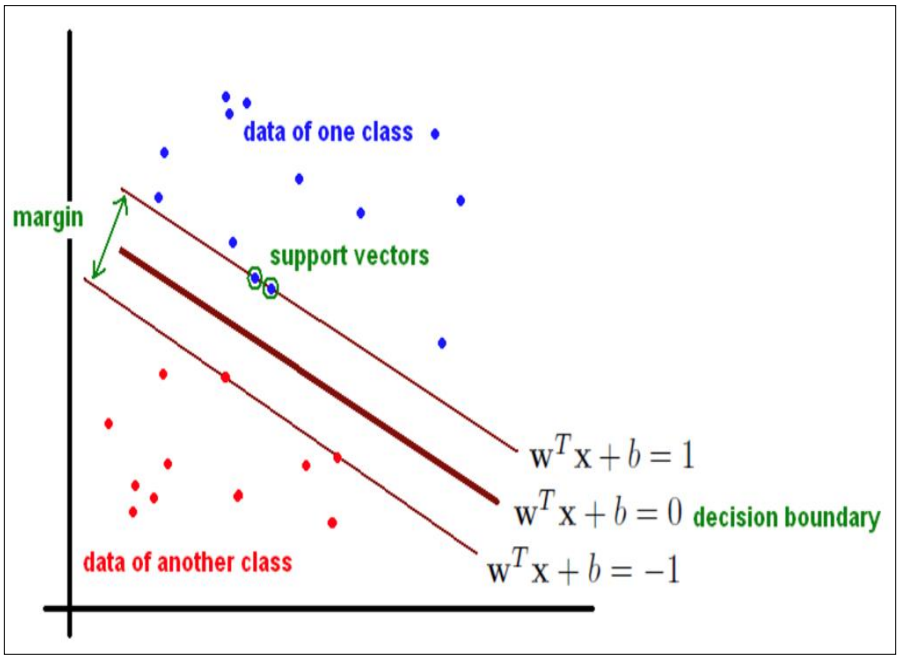

Figure 7- 1 SVM classification principles.

X represents the data space which contains various input vectors, where x is one of two different classes, and is the number of samples. The class label Yi of x equals (1 or -1). The operation of the SVM classifier will be modified according to the type of sample. The optimal separating hyper plane is represented through the hyper-plane being divided by the distance between the plane and the nearest data [103]. For linearly separable data, the hyper-plane is represented by the following equation:

$$X + b = \sum_{i=0}^{N} W_i X_i + b = 0 \qquad (7-1)$$

where W represents a normal vector on the hyper plane, and b represents the distance from the origin.

In the case of the proposed method, nonlinear classification has been used. Two types of kernels have been used to solve the problem of nonlinear classification. The most used in different applications are Polynomial and Gaussian radial basis functions [105], can be defined as below:

Polynomial: $$\mathrm{K}(\mathrm{x_i}, \mathrm{x_k}) = (\mathrm{x_i}^{\mathrm{T}}.\ \mathrm{x_k} + 1)^{\mathrm{n}} \qquad (7-2)$$

Gaussian: $$K(x_i, x_k) = exp|x_i{}^T - x_k|^2 \qquad (7-3)$$

7.5 Methodology for transformer Protection

This section illustrates the method of SVM classification for the protection of the three-phase power transformer. The proposed method utilizes two separate SVM classifiers that use three current signals measured from the three-phase transformer, and based on these signals, the SVMs decide whether the type of current signal indicates an inrush current or fault condition. The SVMs also decide which type of fault is present: internal or external. The aim of using SVM is to free the protection method from overfitting and avoid the limitations of other techniques, and therefore, to improve accuracy. The accuracy of the proposed method, as with all types of machine learning, depends on its training in terms of how representative and sufficient the training data sets are.

7.6 Proposed protection algorithm

Procedure for fault classification can be described in Figure 7-2 as following steps:

1-Read discrete samples of the three-phase currents of the transformer.

2- Taking four samples, which is less than a quarter of a cycle, this is utilized as input data to the SVM1. The result will be +1 for inrush current or -1 in the case of a fault occurrence (internal or external).

3- In the case of a fault, SVM2 classifies internal and external faults.

4- Steps 1 to 4 are repeated unless a fault is detected, and the protection relay is tripped to protect the transformer.

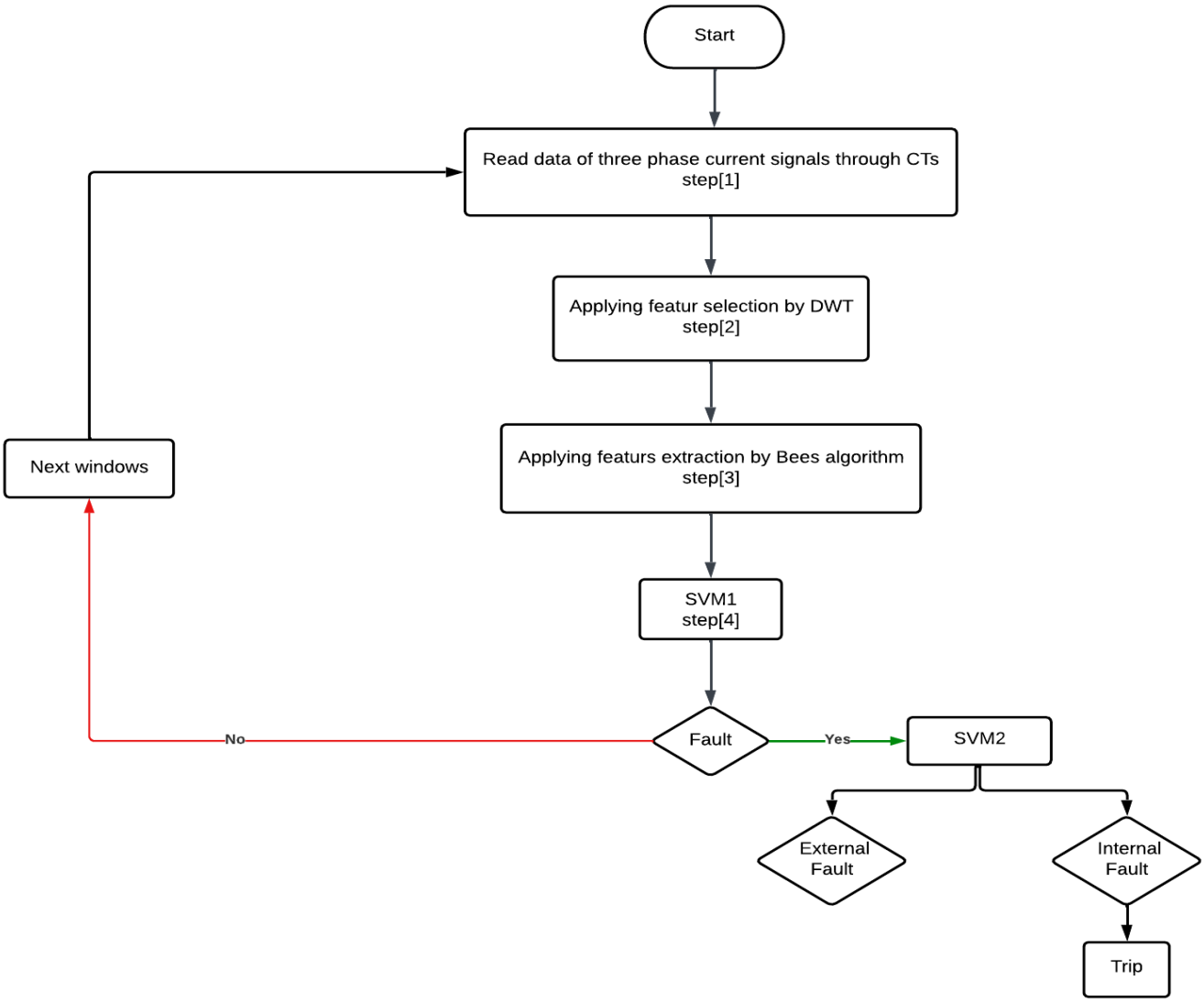

Figure 7- 2 Flow Chart for Protection Algorithm Based on Alienation coefficient.

7.7 SVM classifiers

As shown in Figure 7- 2, the proposed approach uses two SVM classifiers to discriminate between inrush current, internal faults, and external faults in three-phase transformers. SVM1 is the first classifier, and it's been trained to distinguish between inrush current and fault states. SVM2, the second classifier, is trained to distinguish between possible internal and external faults. The output of the SVM1 is +1 if the input signals for first classification indicate inrush current (normal condition); otherwise, it is – 1 in the case of a fault. The second classifier, SVM2, will produce a result of +1 for internal faults and -1 for external faults. The three recommended kernel functions were employed for training and testing in both SVMs. For the test stage, the two functions with the best performance (kernel radial basis function and polynomial kernel function) are used. Using training data, the optimum parameters for SVM 1 and 2 are determined. Testing the kernel parameters at different values yields the classification with the best performance. These variables are modified in the following pattern: kernel width γ is altered in increments of 0.05 between 0.1 and 5, with each step increasing by 0.05. The polynomial kernel (n) is adjusted by 0.5 between 1 and 10. With additions of 1, error (C) changes from 1 to 1000. Table 7-1 displays the kernel values. The accuracy of classification (CA) is used to evaluate the performance of each of the SVMs for each combination of these parameters:

$$\mathrm{CA}\% = \frac{\text{Correctly classified patterns}}{\text{total patterns}} \times 100 \qquad (7-4)$$

Based on these results, the SVM classifier with the best accuracy is chosen, and the testing procedures are carried out.

Table 7- 1 Values of the used kernel parameters

Parameters		**Values**
Kernel width parameter		0-5 in 0.05 steps
Penalty due to the error		1;1000
Order of polynomial kernel		0-10 in 0.5 steps

7.7.1 Training and testing patterns for SVMs

Because the support vector machine is related to machine learning, it must be trained to recognize the changes that may occur in the event of inrush current, external or internal transformer defects. The training and testing data in this study come from a model that was created in the lab for various scenarios. As a result, a large number of instances have been used, encompassing a wide range of operating situations and all databases that include training cases. 2000 patterns were utilized for training and testing in this work to represent various transformer operating circumstances.

7.8 Results and Discussion

The raw data may have errors; hence, data pre-processing has been used to correct these errors. To enhance the capability of SVM, the training data sometimes includes excluding add values, normalizing experimental data and data smoothing.

7.8.1 SVMs training and testing results

Various values of kernel parameters with SVMs were trained and tested using the MATLAB simulation codes. The Gaussian kernel function and polynomial kernel function had the highest accuracy from the SVM classification. Table 7-2 and Table 7-3 show the results respectively. According to these results, the maximum training accuracy obtained using polynomial kernel functions is 94.6187% for SVM1 and 85.7753% for SVM2. SVM1 reaches its highest value when C = 883, and $\gamma = 0.1$, while SVM2 reaches its highest value when C = 725 and $\gamma = 0.15$. Both values were used for the training of two classifiers, SVM1 and SVM2.

In Table 7-5, shows the accuracy of SVM classification for current signal data. To judge the validity and accuracy of the proposed system, the results from the SVMs were compared with results from the ANN. The ANN algorithm was applied to classify different types of transformer operational conditions. To train the ANN algorithm, the same data was used for training and testing. Table 7-4 and 7-5 compare the performance of both algorithms. In terms of training and testing patterns, the proposed technique using SVMs is more accurate and faster than the ANN method.

Table 7- 2 SVMs for the Gaussian kernel function during the training

SVM	kernel width parameter γ	Penalty due to the error C	Accuracy for training patterns%	Accuracy for testing patterns%
SVM1	1.5	265	64.396	78.3304
SVM2	1	104	61.8219	77.7281
SVM1	2	513	79.1165	93.1581
SVM2	6	682	75.2559	88.3249
SVM1	3	610	89.2489	94.8133
SVM2	2	595	75.7931	88.0254
SVM1	8.5	490	88.9140	95.6392
SVM2	9	463	90.5612	89.0863

Table 7- 3 process SVMs for the Polynomial kernel function during the training process

SVM	kernel width parameter γ	Penalty due to the error C	Accuracy for training patterns%	Accuracy for testing patterns%
SVM1	0.55	231	60.5127	95.7567
SVM2	0.4	437	70.6577	87.5635
SVM1	0.5	404	76.3289	91.7567
SVM2	0.35	578	73.3533	88.0711
SVM1	0.3	772	83.6723	95.2895
SVM2	0.8	85	71.2258	85.2792
SVM1	0.1	883	94.6187	98.7107
SVM2	0.15	725	85.7753	89.3401

Comparison between the proposed method and the ANN-based method is shown in Table 7- 4. The proposed SVM-based process can detect all types of test faults in a shorter amount of time.

Table 7- 4 Comparison between fault detection time (ms) for ANN and SVMs methods

Type of fault	Fault detection time (ms)	
	ANN	SVM
Inrush current	10.0106	2.1424
External fault	10.9518	2.8716
Internal fault (turn to turn fault)	11.3588	2.7151
Internal fault (turn to ground fault)	9.9508	2.4191

Table 7- 5 Comparison between Accuracy for training and testing patterns % for SVMs and ANN methods

Method	Training Accuracy %	Testing Accuracy %
SVM1	94.6187	98.7107
SVM2	90.5612	89.3401
ANN	80.7619	73.7143

As shown in Table 7-5, SVM1 and SVM2 are more accurate than ANN in training and testing patterns in a short period of time.

7.9 Case studies

In the laboratory test, the results were the same for all three phases. The purpose of this section is to outline the details of three test cases that demonstrate the accuracy and validity of the proposed system for identifying and differentiating between phase A faults.

7.9.1 Normal current (inrush current)

Inrush current in transformer arises initially when the tramsformer is first energized; the magnitude of inrush current is 10 to 15 times higher than the rated current of transformer as Figure7-3 shows three-phase current with inrush currents. Figure 7-4 shows the current in phase A.

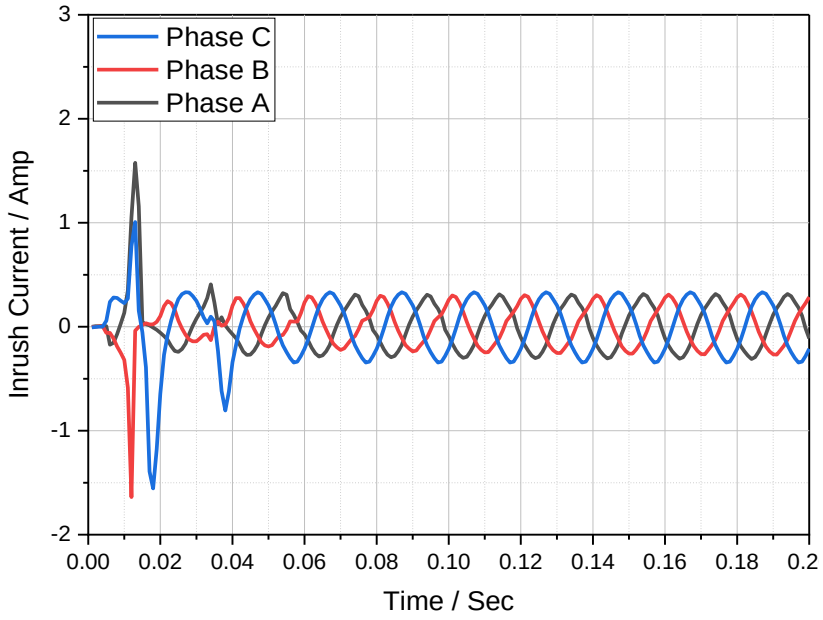

Figure 7- 3 Inrush Current in three phases.

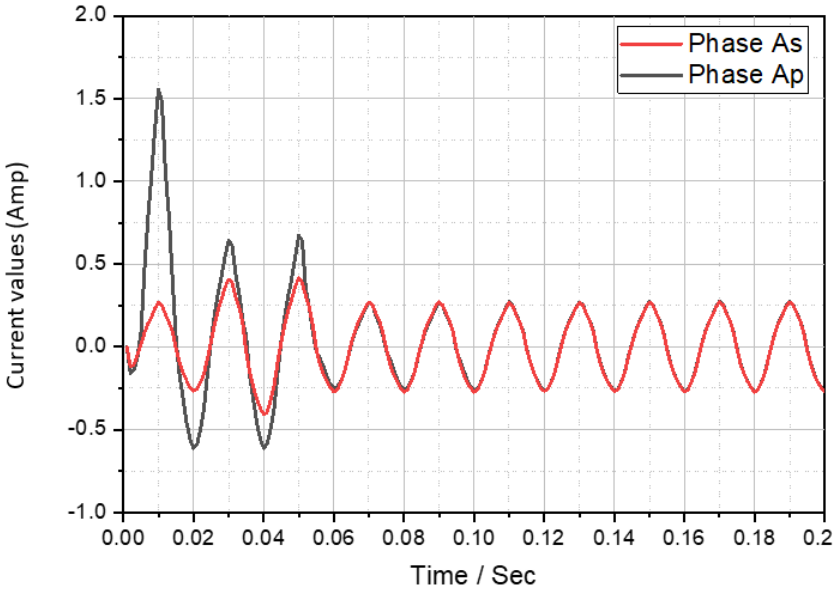

Figure 7- 4 Inrush Current in phase A.

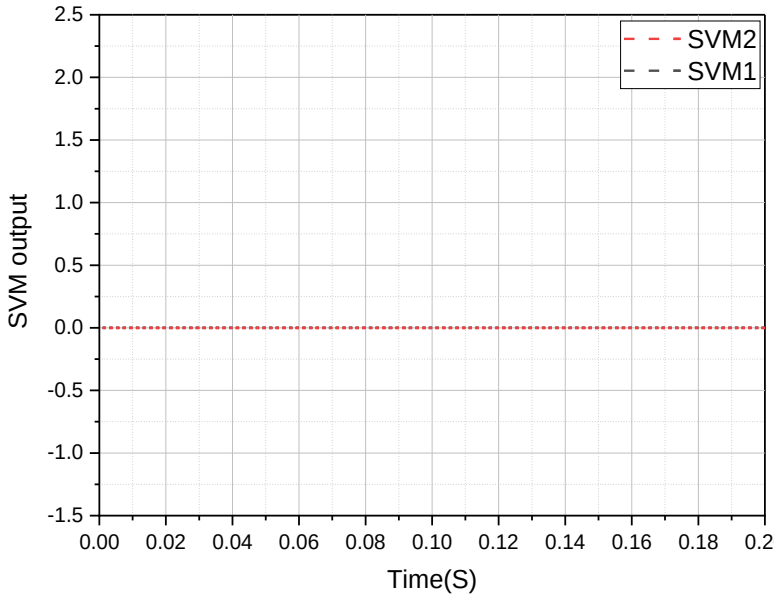

Figure 7- 5 Classification label of SVM.

Figure 7-5 shows the classification label derived from the SVMs. Output SVM1 is 0, meaning that no fault has occurred.

7.9.2 External fault

An external fault is created between the transformer and the loads at the middle of the lines shown by equivalent circuit diagram in figure 4-10. Figure 7-6 shows three phase currents with external faults. The primary and secondary currents with the external fault in phase A at t =0.06s are in figure 7-7. Figure 7-8 shows the classification label from the SVMs. The SVM1 output is +1, representing no fault happening until t =0.06s, when the fault is created, after which the output changes to -1, representing a fault. The SVM2 output becomes +1 at time 0.06s, indicating the external fault.

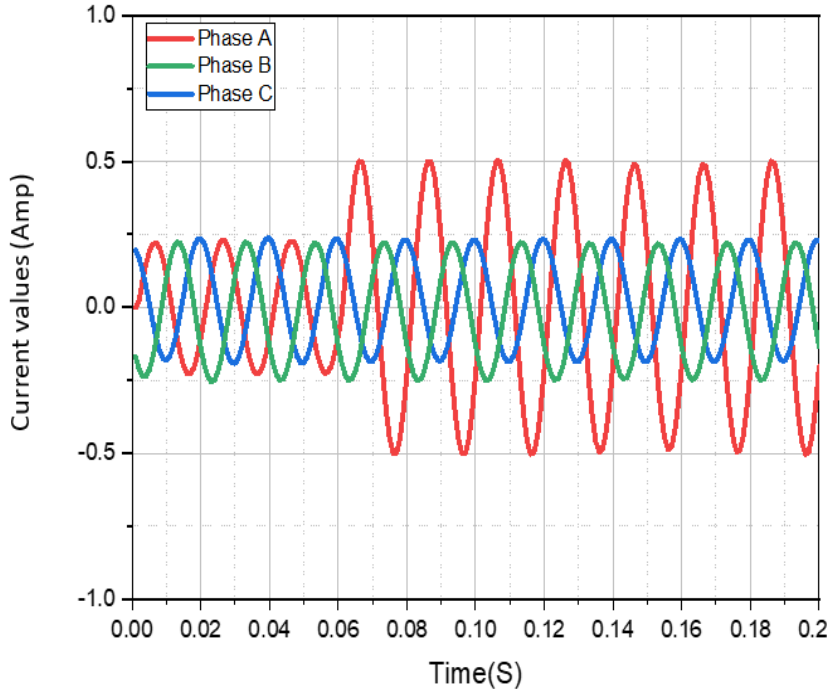

Figure 7- 6 External fault in three phases.

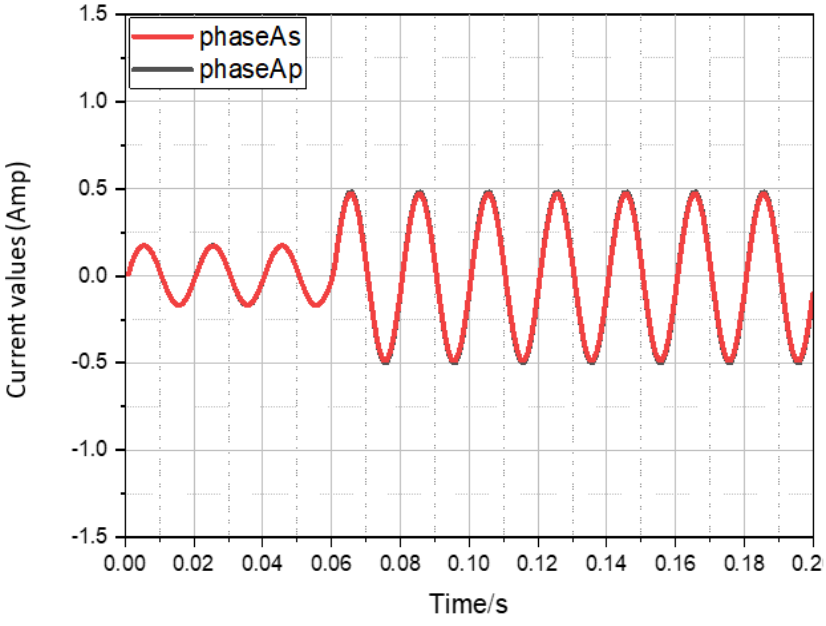

Figure 7- 7 External fault in phase A.

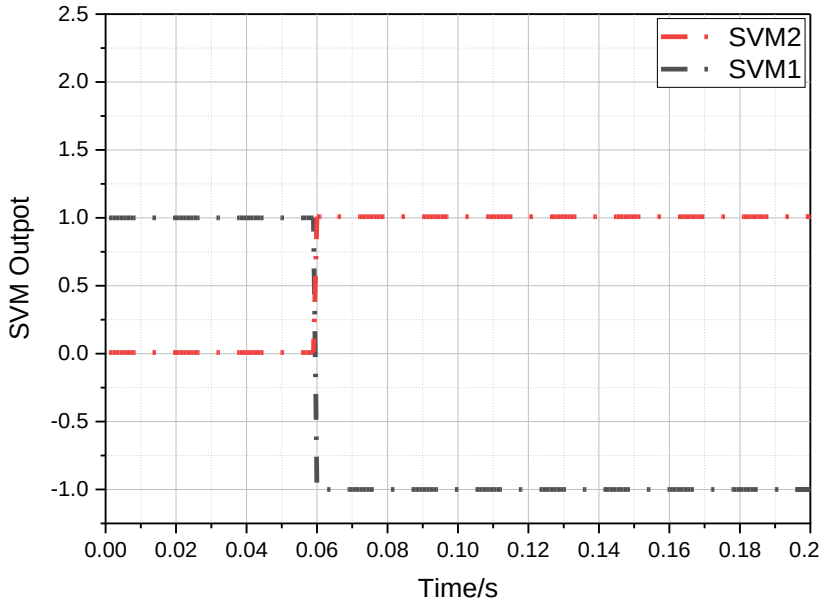

Figure 7- 8 Classification label of SVM.

7.9.3 Internal fault (turn to turn fault)

An internal fault was created between turn 2 and turn 5 on the secondary side of phase A. Figure 7-9 shows the shape of three phase currents with turn-to-turn fault. Figure 7-10 illustrated that the primary and secondary currents with a turn-to-turn fault in phase A at t =0.06s. It also shows that the currents of phase A on the primary and the secondary side of the transformer have different magnitudes, at phase As and phase Ap. The primary (phase Ap) has a higher magnitude due to the turn-to-turn fault. Figure 7-11 shows how the SVMs deal with this case. The SVM1 output has the same reaction as in case I (section 7.9.1), wherein the output is +1, representing no fault from 0 to 0.06s and then changes to -1, representing the fault's occurrence at 0.06s. The SVM2 output becomes -1 at t =0.06s, signifying that an internal fault is detected. The reaction of SVM1 and SVM2 in these cases shows the capability of the proposed method to distinguish between inrush current in normal operation, and internal and external faults.

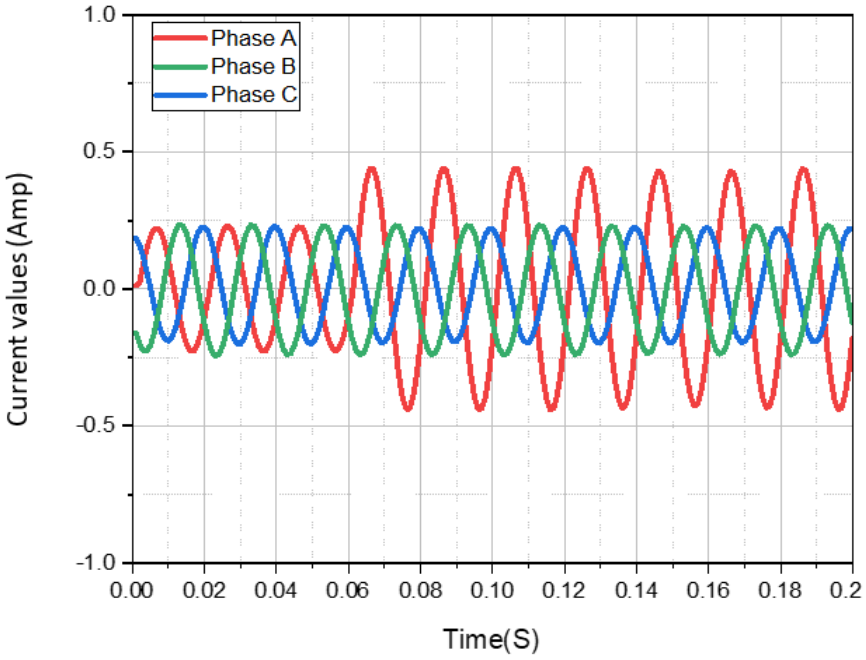

Figure 7- 9 Internal fault (Turn to turn) in three phases.

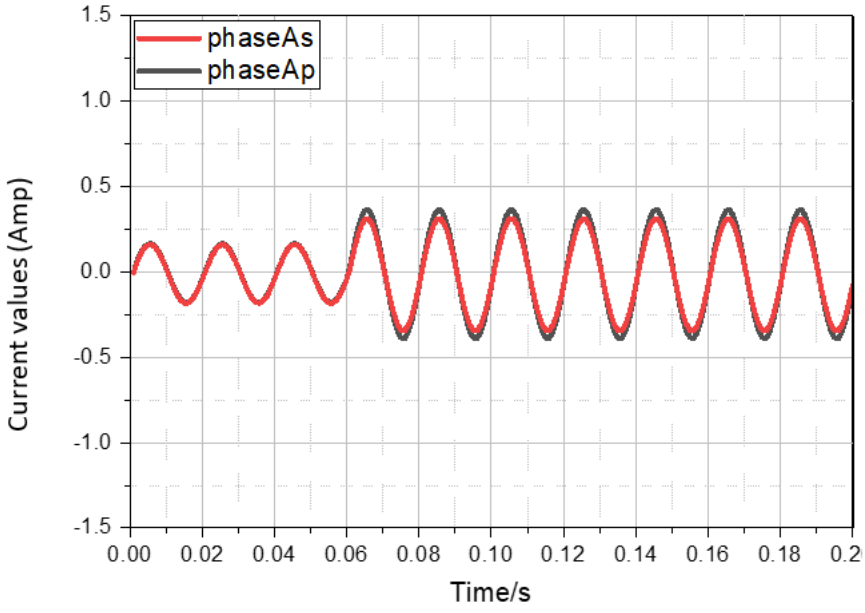

Figure 7- 10 Internal fault (Turn to turn) in phase A.

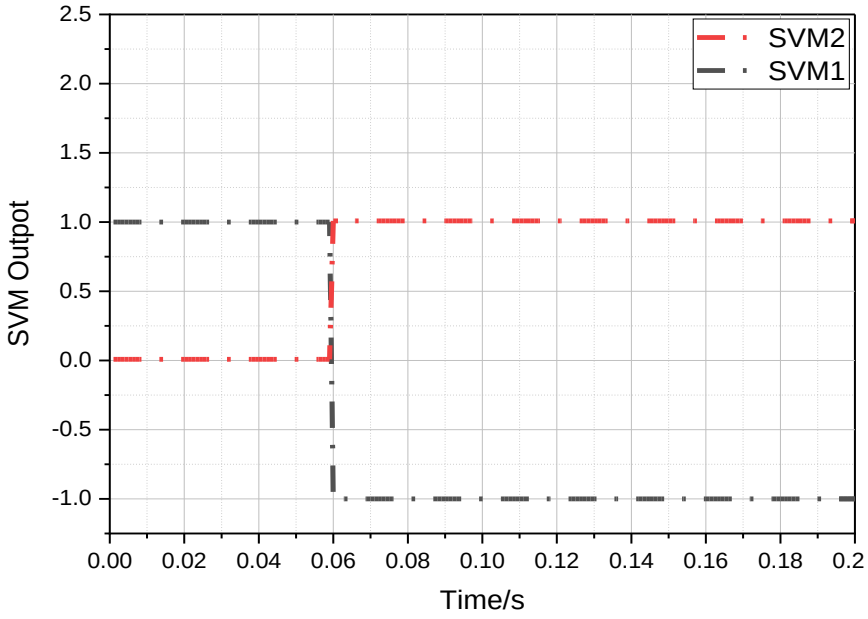

Figure 7- 11 Classification label of SVM.

7.9.4 Internal fault (turn to ground)

Similar results as for the turn-to-turn fault were obtained during this stage, through choosing different turns on both primary and secondary sides to create a turn to ground fault in phase A of the transformer. Therefore, turn 30 of phase A on the secondary side is shown in this study. To reduce the high current that would pass from turn 30 to the ground, a resistor (5 Ω) was connected in series via tap number 30 to the ground figure 7-12. As Fig 7-13 shows the primary and secondary currents with a fault were generated at 0.06s in phase A by closing the switch relay.

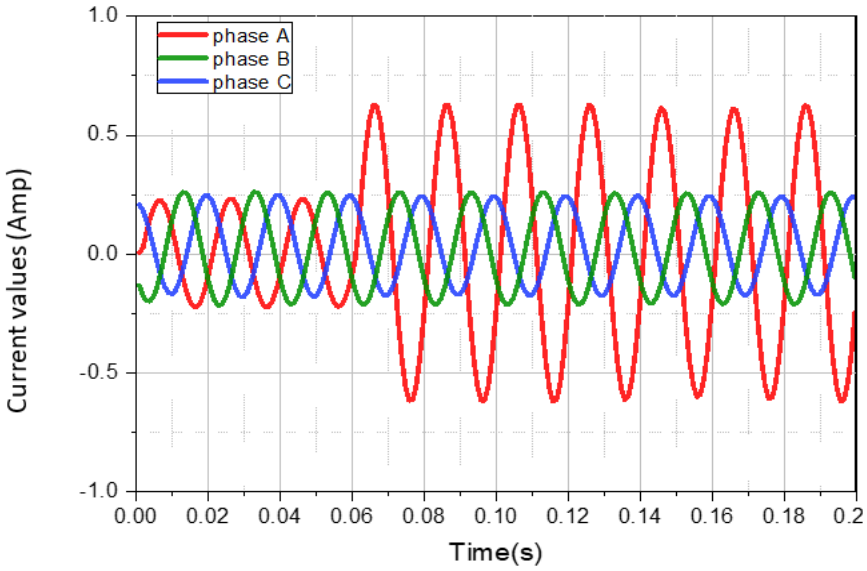

Figure 7- 12 internal fault (Turn to ground) in three phases.

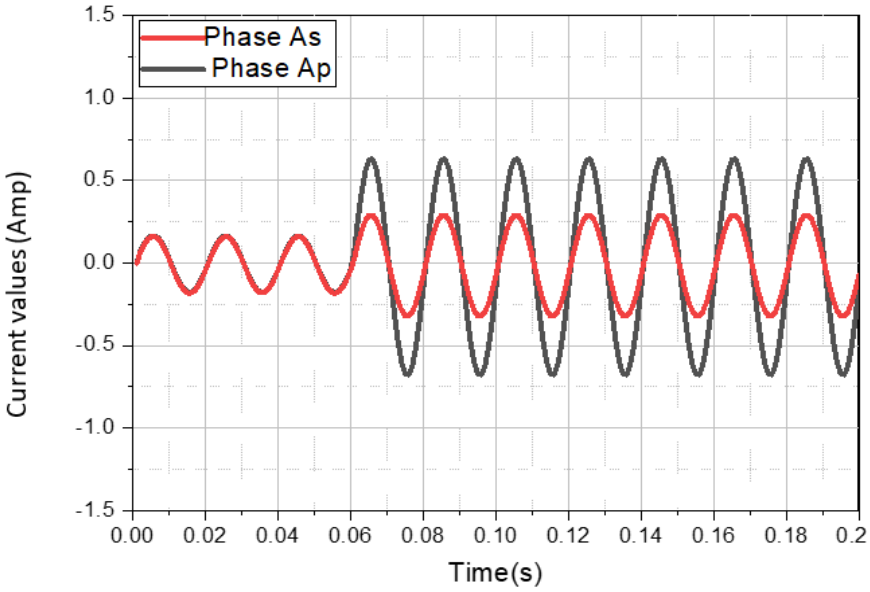

Figure 7- 13 Internal fault (Turn to ground) in phase A.

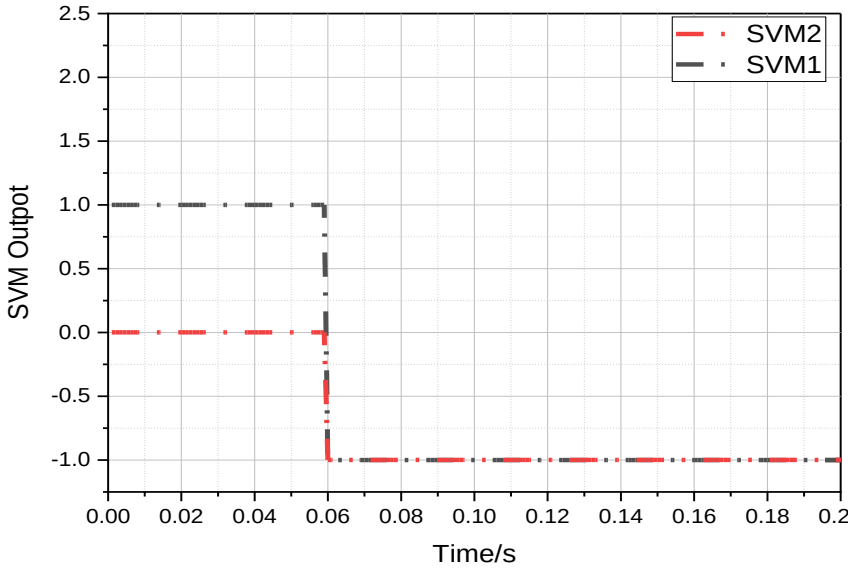

Figure 7- 14 Classification label of SVM.

Figure 7-14 shows how the SVMs deal with this case. The SVM1 output has the same reaction as in case II (section 7.9.3) where in the output of SVM1 is +1, representing no fault from 0 to 0.06s and then it changed to -1, representing the fault's occurrence at 0.06s. The SVM2 output becomes -1 at t =0.06s, signifying that a turn to ground fault is detected. The reaction of SVM1 and SVM2 in these cases also shows the ability of the proposed technique to distinguish between inrush current in normal operation, and internal and external faults.

7.10 Summary

This chapter has introduced an efficient and reliable technique based on SVM for classifying faults in a three-phase power transformer. Two classifiers have been used (SVM1 and SVM2). The first is utilized to detect whether there is a fault or not, while the second is utilized to identify if the fault is internal or external. In the proposed method, fault identification and detection were completed in less than a quarter cycle of the phase current of phase A. The same technique has been applied to phases B and C. The classification of faults was not affected by the type or location of the fault. Comparing the SVM method with the ANN method demonstrates that the SVM method is more accurate and faster. Using the proposed technique for several cases, it has been illustrated that the SVM classifier offers reliably accurate discrimination in all operating conditions. The proposed method has been shown to be very fast in detecting the different types of faults selected, with high accuracy on average for all test cases of. The proposed SVM-based

method gives a fast response compared to the ANN-based method. This makes the SVM a good candidate to replace conventional methods.

CHAPTER 8

Conclusion and future work

8.1 Conclusion

The protection and condition monitoring of three-phase transformers have become a rapidly growing field of study due to the increasing demand for reliable power systems. To enhance the effectiveness of transformer protection, many condition monitoring techniques have been proposed in the literature. However, these methods still face challenges such as inrush currents that can compromise the protection system's functionality and the transformer's efficiency.

In response to these challenges, this thesis introduces new contributions outlining three methods for transformer protection that prioritize quick and efficient detection of both internal and external faults. The methods aim to detect internal and external faults quickly and effectively while addressing the challenges of inrush currents and saturation cases that can compromise the protection system's functionality and the transformer's efficiency. Digital signal processing and Machine Learning -based techniques, which were unknown until recently, have been applied to three-phase transformer faults protection and condition monitoring. Through using digital signal processing and artificial intelligence techniques, the transformer can be protected in a faster, more secure, less expensive and more reliable way than with traditional methods.

Additionally, advanced techniques have reduced the amount of time required to detect faults. These developments will change transformer protection in the future. At the outset of this study, data on current signals were gathered through laboratory experiments on a transformer model that underwent different faults during operation. The collected data was subsequently utilized to evaluate the efficacy of the proposed methods for detecting these faults. Specifically, the proposed methods were applied in three different approaches:

- A new method of operation has been proposed for transformer operation based on a model that was executed in MATLAB/SIMULINK. An analysis of the Alienation Coefficients for the current waveforms used in this technique was used to distinguish between external and internal faults on the three-phase transformer. The proposed method was found to be

efficient and fast, detecting faults within a maximum of 3ms depending on the type and severity of the fault. The technique was also effective at detecting faults during current overload situations and addressed the issue of transformer saturation that is common with differential protection systems. Laboratory experiments were conducted to verify the ability of the proposed technique for detecting and eliminating faults. Current signals were collected under different conditions, and the results indicated that the technique was successful in detecting and eliminating internal faults. Furthermore, it achieved this with a remarkable response time of only 3ms. The method was also capable of differentiating between external and internal faults, and it overcame the issue of current transformer (CT) saturation. Furthermore, the technique was able to detect turn-turn faults even when only two turns were shorted, and it is expected to be even more effective on larger transformers with more turns shorted. Unlike overcurrent protection that trips based on current flow, whether caused by internal or external faults, the proposed method relies on waveshapes, making it superior. A change in current due to a load change will not be considered as an external fault, and the relay operation will remain stable.

- A new hybrid model is proposed to diagnose faults in three-phase power transformers. Algorithms are applied to optimize the selection of discriminating features to enhance fault detection performance, and a comparison is made between the number of features assessed. The proposed model's performance is enhanced by selecting discriminative features using optimization algorithms. The proposed hybrid model combines machine learning classifiers with an optimal feature identification process to detect faulty features. The approach uses Discrete Wavelet Transform (DWT) and Orthogonal Matching Pursuit (OMP) to extract statistical characteristics from the samples. This process helps to reduce the amount of data required for the model while also increasing its accuracy. The Bees algorithm (BA) is then utilized to create an optimized subset of the features, further improving the model's performance. The optimized feature set is used as input for three classification algorithms: K-Nearest Neighbor (KNN), Support Vector Machine (SVM), and Artificial Neural Network (ANN), to distinguish between normal operating conditions and faults. To ensure that the model is robust, the training process employs k-fold cross-validation. Overall, this hybrid model is designed to provide accurate fault detection while minimizing the number of data points required, making it a promising approach for fault

diagnosis in various applications. A cross-validation strategy is used to train three machine learning classifiers to detect faults that may occur in three-phase transformers. The classifiers are found to be accurate, indicating the potential of the proposed model for use in fault detection and classification. To assess the effectiveness of the proposed approach, it was compared to a similar method using a genetic algorithm (GA). The model's performance was evaluated based on its accuracy, specificity, precision, recall, and F1 score. The experimental results demonstrated that the proposed model is effective in identifying faults in a variety of conditions and types of faults. Overall, the proposed approach outperformed the GA-based approach, indicating that the hybrid model's use of the Bees algorithm for feature selection and optimization led to improved performance. The high accuracy and precision of the proposed approach makes it a promising option for fault detection and diagnosis in a range of practical applications. To train three machine learning classifiers to detect faults that may occur in the transformer, a cross-validation strategy is used. The proposed model's robustness is tested using simulations of different cases of the transformer with inrush current from fault conditions. The model uses the time-frequency domain to analyze the data, applying discrete wavelet transform and orthogonal matching pursuit to recover 40 distinct features per signal. The Bees algorithm is used to reduce the volume of necessary data, selecting 12 features of the current signal, while the genetic algorithm selects 21 features. For fault type detection, the study applies three machine learning classifier approaches, namely ANN, SVM, and KNN, and trains them based on the selected features, using 10-fold cross-validation. To validate the effectiveness of the proposed model, optimization algorithm-based feature selection is performed on the same dataset. The findings show that the proposed strategies with fewer statistical features give comparatively more accurate results, and that the Bees algorithm results in greater total accuracy while selecting fewer features. When the Bees algorithm is combined with an SVM classifier, maximal accuracy of 96% in fault classification is seen with high accuracy on evaluation based.

- Another proposed technique based on Support Vector Machines (SVM) has been introduced for fault classification in three-phase power transformers. The proposed method utilizes two SVM classifiers, SVM1 and SVM2, where SVM1 detects the presence of a fault, and SVM2 identifies whether the fault is internal or external. The fault detection and

identification process can be completed within a quarter cycle of phase A current and applied to all three phases. In order to evaluate the performance of SVMs in fault classification, different values of kernel parameters were trained and tested using MATLAB simulation codes. Among the different kernel functions tested, the Gaussian and polynomial kernel functions demonstrated the highest accuracy in SVM classification. The maximum training accuracy achieved using the polynomial kernel function was 94.6187% for SVM1 and 85.7753% for SVM2. Both values were used for the training of two classifiers, SVM1 and SVM2. To assess the validity and accuracy of the proposed system, the results obtained from the SVM classifiers were compared with the results obtained from an ANN algorithm. The same data was used for training and testing the ANN algorithm. The results showed that the proposed technique using SVMs is more accurate and faster than the ANN method, as evidenced by the training and testing patterns presented in Table 7-4.

The proposed method was evaluated through laboratory experiments on a transformer model with two and five short-circuited turns, which is below the minimum standard set by IEEE for detecting turn-turn faults. Despite this, the proposed method was successful in detecting these faults, even with the presence of protection resistors, which reduced the fault current value. The method's performance can be further improved by testing it on larger transformers or by inducing faults without the use of protection resistors. Many researchers have presented innovative and competitive approaches beyond what is described in the literature. The second harmonic method is the most popular and widely used method for ensuring transformer differential protection, but it has some limitations.

In summary, the proposed methods showed high levels of accuracy and repeatability. The study proposes the use of alienation coefficients and artificial intelligence to detect and track faults in three-phase transformers. The effectiveness of these techniques was found to be superior to conventional methods in terms of speed, safety, cost, and reliability. As a result, the proposed method leads to reduced detection time and improved production.

8.2 Future work

The detection and classification of faults in power systems are critical for the safe and efficient operation of power systems. For these reasons, future work can be done to improve these

techniques and adapt them for larger generators and transformers. The proposed method has the potential to improve the reliability and efficiency of power systems, as shown below:

Generator Fault Detection

The proposed correlation technique can be adapted to recognize the types of internal faults and on which side of the transformer they occur. This study has proven that the proposed methods of winding tests on small test transformers may be realistic, but they could equally well be adapted for larger test transformers, a project that can be planned in partnership with power distribution companies for future work. Generators also exhibit internal faults, turn-grounds, and turn-turns in a similar way to transformers. Usually, generator protection systems do not detect turn-turn faults. The Alienation Coefficient technique proposed can be adapted to detect these internal faults on generators as well as discriminate them from external faults.

References

[1] S. H. Horowitz and A. G. Phadke, Power system relaying, 3rd ed. John Wiley & Sons Ltd, 2008.

[2] W. H. Tang, K. Spurgeon, Q. H. Wu, and Z. J. Richardson, "An evidential reasoning approach to transformer condition assessments," IEEE Trans. Power Deliv., vol. 19, no. 4, pp. 1696–1703, 2004.

[3] Transformer Protection Principles (2015). GE Digital Energy. [Online]. Available: https://www.gedigitalenergy.com/smartgrid/ /Mar07/article5.pdf.

[4] M. R. Barzegaran, "Detecting the Position of Winding Short Circuit Faults in Transformer Using High Frequency Analysis," EURO journels, vol. 23, no. 4, pp. 644–658, 2008.

[5] W. H. Bartley, "An International Analysis of Transformer Failures," Part 1, Locomot. Winter 2004, vol. 78, no. 1, 2004.

[6] Distribution Automation Handbook (2015). ABB. [Online].Available: https://library.e.abb.com/public/74b47888d7788642c125795f00430599/DAH andbook_Section_08p07_Protection_of_HV_Transformers_757288_ENa.pdf.

[7] J. Blackburn and T. Domin, Protective Relaying: Principles and Applications, 3rd ed. CRC Press Taylor & Francis Group, LLC., 2006.

[8] M. Tripathy, R. P. Maheshwari, and H. K. Verma, "Advances in Transformer Protection: A Review," Electr. Power Components Syst., vol. 33, no. 11, pp. 1203–1209, 2005.

[9] Z. Wei-nan TANG Tao and Y. Yi-song, "Automation technology and application of power plant and transformation substation," China Electr. Power Press, 2005.

[10] S. H. Horowitz and A. G. Phadke, Power System Relaying. John Wiley & Sons, 1992.

[11] A. G. Phadke and J. S. Thorp, "A New Computer-Based Flux-Restrained Current-Differential Relay for Power Transformer Protection," IEEE Trans. Power Appar. Syst., vol. PAS-102, no. 11, pp. 3624–3628, 1983.

[12] T. S. Sidhu and M. S. Sachdev, "On-line identification of magnetizing inrush and internal faults in three-phase transformers," IEEE Trans. Power Deliv., vol. 7, no. 4, pp. 1885–1891, 1992.

[13] R. G. Carter, Electromagnetism for Electronic Engineers. Richard G. Carter & Ventus Publishing ApS, 2010

[14] Faraday's notebooks: Electromagnetic Induction (2015). RIGB. [Online]. Available: http://www.rigb.org/docs/faraday_notebooks__induction_0.pdf.

[15] Transformer Construction (2015). Electronics-tutorials. [Online]. Available: http://www.electronics-tutorials.ws/transformer/transformer-construction.html.

[16] M. S. Sarma, Electric Machines, 2nd ed. Saint Paul, USA: West publishing company, 1994.

[17] S. J. Chapman, Electric Machinery Fundamentals, 5th ed. McGraw-Hill, Inc.,2012.

[18] R. M. Del Vecchio, B. Poulin, P. T. Feghali, D. M. Shah and R. Ahuja, Transformer Design Principles: With Applications to Core-form Power Transformers, 2nd ed. CRC Press, Boca Raton, FL., 2010.

[19] Three Phase Transformers (2015). Electronics-tutorials. [Online]. Available: http://www.electronics-tutorials.ws/transformer/three-phase-transformer.htm

[20] Magnetic Hysteresis (2015). Electronics-tutorials. [Online]. Available: http://www.electronics-tutorials.ws/ electromagnetism/magnetic-hysteresis.html.

[21] S. Chattopadhyay and S. Sengupta, Basic Electrical Engineering. Oxford, U.K.: Alpha Science International, 2010.

[22] Magnetizing Inrush Current in Power Transformer (2015). Electrical4u.[Online]. Available:http://www.electrical4u.com/magnetizing-inrush-current-in-power-transformer

[23] W. K. Sonnemann, C. L. Wagner, and G. D. Rockefeller, "Magnetizing Inrush Phenomena in Transformer Banks," Trans. Am. Inst. Electr. Eng. Part III Power Appar. Syst., vol. 77, no. 3, pp. 884–892, 1958.

[24] J. Berdy, W. Kaufman and K. Winick, "A Dissertation on Power Transformer Excitation and Inrush Characteristics," in Symposium on Transformer Excitation and Inrush Characteristics and Their Relationship to Transformer Protective Relaying, Houston, TX, August 5, 1976.

[25] L. F. Blume, Transformer engineering, vol. 251, no. 4. New York: Wiley & Sons, 1951.

[26] K. Karsai, D. Kerenyi and L. Kiss, Large power transformers. New York: Elsevier, 1987.

[27] Y. Kang, U. Lim, S. Kang and Y. Kim, "Compensating algorithm for use with measurement type current transformers for protection," IEEE Russ. Power Tech, 2005.

[28] L. Kojovic, "Comparison of different current transformer modeling techniques for protection system studies," IEEE Power Eng. Soc. Summer Meet., 2002.

[29] C.57.13-1993 ANSI /IEEE Standard, Requirements For Instrument Transformers.

[30] Transformer and Transformer-feeder Protection (2015). FECIME. [Online]. Available: http://www.fecime.org/referencias/npag/chap16-254-279.pdf.

[31] Transformer Protection Principles (2015). GE Digital Energy. [Online]. Available: https://www.gedigitalenergy.com/smartgrid/ /Mar07/article5.pdf.

[32] J. Blackburn and T. Domin, Protective Relaying: Principles and Applications,3rd ed. CRC Press Taylor & Francis Group, LLC., 2006.

[33] J. J. Grainger, S. H. Lee, and A. a. El-Kib, "Design of a Real-Time Switching Control Scheme for Capacitive Compensation of Distribution Feeders," IEEE Trans. Power Appar. Syst., vol. PAS-101, no. 8, pp. 2420 – 2428, 1982.

[34] IEEE Std C37.91-2000, IEEE Guide for Protective Relay Applications to Power Transformers. 2000.

[35] P.M. Anderson, Power System Protection. John Wiley & Sons, Inc., 1999.

[36] M. R. Barzegaran, "Detecting the Position of Winding Short Circuit Faults inTransformer Using High Frequency Analysis," EURO journels, vol. 23, no. 4, pp. 644–658, 2008.

[37] Distribution Automation Handbook (2015). ABB. [Online]. Available: https://library.e.abb.com/public/74b47888d7788642c125795f00430599/DAHandbook_Section_08p07_Protection_of_HV_Transformers_757288_ENa.pdf.

[38] P.M. Anderson, Power System Protection. John Wiley & Sons, Inc., 1999.

[39] S. R. H. Ding, R. Heywood, J. Lapworth, "Why Transformers Fail," Euro TechCon 2009, Stretton, United Kingdom, 2009.

[40] IEEE Std C37.108-2002, IEEE Guide for the Protection of Network Transformers. 2002.

[41] J. Blackburn and T. Domin, Protective Relaying: Principles and Applications, 3rd ed. CRC Press Taylor & Francis Group, LLC., 2006.

[42] W. A. Elmore, protective relaying theory and applications, 2nd ed. New York: Marcel Dekker, Inc., 2000.

[43] Transformer Protection (2015). EWH. [Online]. Available: http://ewh.ieee.org/r1/boston/pes/Education/TransformerCourse2010/CourseR esources /IEEEPESTransformerProtection.pdf.

[44] S. H. Horowitz and A. G. Phadke, Power system relaying, 3rd ed. John Wiley & Sons Ltd, 2008.

[45] W. H. Bartley, “An International Analysis of Transformer Failures,” Part 1, Locomot. Winter 2004, vol. 78, no. 1, 2004.

[46] G. Rockefeller, Transformer Protection Application Guide. Basler Electric Company, 1999.

[47] H. P. Sleeper, “Ratio differential relay protection,” Electr. World, pp. 827– 831, 1927.

[48] R. E. Cordray, “Percentage Differential Transformer Protection,” Electr. Eng., vol. 50, pp. 361–363, 1931.

[49] [48]. R. E. Cordray, “Preventing False Operation of Differential Relays,” Electr. World, pp. 160–161, 1931.

[50] G. Ziegler, Numerical differential protection, 2nd ed. Erlangen, Germany: Publicis Pub, 2012.

[51] R. Hamilton, "Analysis of Transformer Inrush Current and Comparison of Harmonic Restraint Methods in Transformer Protection", IEEE Transactions on Industry Applications, vol. 49, no. 4, pp. 1890-1899, 2013. Available: 10.1109/tia.2013.2257155.

[52] M.S, Deshmukh and V.T. Barhate, “Transformer protection by distinguishing inrush and fault current with harmonic analysis using fuzzy logic’ , IEEE ICCRE (International Conference on Control and Robotics Engineering), Singapore,2016

[53] F. Zeng, Q. Liu and C. Shi, "The Discrimination of Inrush Current from Internal Fault of Power Transformer based on EMD", Energy and Power Engineering, vol. 05, no. 04, pp. 1425-1428, 2013. Available: 10.4236/epe.2013.54b270.

[54] Maya P., S. shree, Roopasree K. and K. Soman, "Discrimination of Internal Fault Current and Inrush Current in a Power Transformer Using Empirical Wavele

[55] Transform", Procedia Technology, vol. 21, pp. 514-519, 2015. Available: 10.1016/j.protcy.2015.10.038.

[56] Y. Wang, "The Influence of Exciting Inrush Current of Transformer Differential Protection", Applied Mechanics and Materials, vol. 397-400, pp. 1935-1938, 2013. Available: 10.4028/www.scientific.net/amm.397-400.1935.

[57] L. Lawhead and R. Hamilton, “Harmonic sharing for effective detection of transformer inrush condition in differential protection schemes,” in Proc. 31st Annu. Western Protective Relay Conf., Spokane, WA, USA, Oct. 19–21, 2004, pp. 1–24.

[58] X. Lin, J. Huang, L. Zeng and Z. Bo, "Analysis of Electromagnetic Transient and Adaptability of Second-Harmonic Restraint Based Differential Protection of UHV Power Transformer", IEEE Transactions on Power Delivery, vol. 25, no. 4, pp. 2299- 2307, 2010. Available: 10.1109/tpwrd.2010.2050343.

[59] A. Guzman, “Performance analysis of traditional and improved transformer differential protective relays,” SEL Tech. Pap., pp. 405–412, 2000.

[60] A. Kulidjian, B. Kasztenny and B. Campbell, “New Magnetizing Inrush Restraining Algorithm for Power Transformer Protection,” IEE Int. Conf. on Developments in Power System Protection, pp. 181–184, 2001.

[61] A. K. Al-Othman and K. M. El-Naggar, “A New Digital Dynamic Algorithm for Detection of Magnetizing Inrush Current in Transformers,” Electr. Power Components Syst., vol. 37, no. 4, pp. 355–372, 2009.

[62] A. Giuliante and G. Clough, “Advances in the Design of Differential Protection for Power Transformers,” Protective Relaying Conference, pp. 1−12, 1991.

[63] X. N. Lin, P. Liu, and O. P. Malik, “Studies for identification of the inrush based on improved correlation algorithm,” IEEE Trans. Power Deliv., vol. 17, no. 4, pp. 901–907, 2002.

[64] T. A. Kawady, H. E. Labna, and A. E. M. I. Taalab, “A practical winding fault detector for power transformers,” 2008 12th Int. Middle East Power Syst. Conf. MEPCON 2008, pp. 130–135, 2008.

[65] Y. C. Kang, B. E. Lee, and S. H. Kang, “Transformer protection relay based on the induced voltages,” Int. J. Electr. Power Energy Syst., vol. 29, no. 4, pp. 281–289, 2007.

[66] M. S. Sachdev, T. S. Sidhu, and H. C. Wood, “Digital relaying algorithm for detecting transformer winding faults,” IEEE Trans. Power Deliv., vol. 4, no. 3, pp. 1638–1648, 1989.

[67] H. Khorashadi-Zadeh, “Power transformer differential protection scheme based on symmetrical component and artificial neural network,” 7th Semin. Neural Netw. Appl. Electr. Eng., vol. C, pp. 261–265, 2004.

[68] X. Lin, J. Huang, L. Zeng and Z. Bo, "Analysis of Electromagnetic Transient and Adaptability of Second-Harmonic Restraint Based Differential Protection of UHV Power Transformer", IEEE Transactions on Power Delivery, vol. 25, no. 4, pp. 2299- 2307, 2010. Available: 10.1109/tpwrd.2010.2050343.

[69] H. Zhang, J. F. Wen, P. Liu, and O. P. Malik, “Discrimination between fault and magnetizing inrush current in transformers using short-time correlation transform,” Int. J. Electr. Power Energy Syst., vol. 24, no. 7, pp. 557–562,2002.

[70] O. Ozgonenel, U. K. Terzi, O. Akar, and U. Kurt, “Discrimination of magnetizing inrush and internal fault currents based on Stockwell transform and ann approach for transformer protection,” 2019 11th International Conference on Electrical and Electronics Engineering (ELECO), 2019. doi:10.23919/eleco47770.2019.8990377

[71] K. Erdal, Ö. Okan, U. Ömer, and D. Thomas, “PCA based protection algorithm for transformer internal faults,” Turkish J. Electr. Eng. Comput. ., vol. 17, no. 2, pp. 125–142, 2009.

[72] R. Hamilton, "Analysis of Transformer Inrush Current and Comparison of Harmonic Restraint Methods in Transformer Protection", IEEE Transactions on Industry Applications, vol. 49, no. 4, pp. 1890-1899, 2013. Available: 10.1109/tia.2013.2257155.

[73] A. Rahmati and M. Sanaye-Pasand, “A fast WT-based algorithm to distinguish between transformer internal faults and inrush currents,” Eur. Trans. Electr. Power, vol. 22, no. 4, pp. 471–490, 2011.

[74] Z. Babaei and M. Moradi, “Novel method for discrimination of Transformers faults from magnetizing inrush currents using wavelet transform,” Iranian Journal of Science and Technology, Transactions of Electrical Engineering, vol. 45, no. 3, pp. 803–813, 2021. doi:10.1007/s40998-020-00399-1

[75] J. Xie, M. A. Elizondo, F. K. Tuffner, and K. P. Schneider, “Dynamic-phasor model of transformer inrush simulation for unbalanced distribution system,” 2020 IEEE/PES Transmission and Distribution Conference and Exposition (T&D), 2020. doi:10.1109/td39804.2020.9300007

[76] Ge Baoming, A T. de Almeida, ZQionglin, and W.Xiangheng,IEEE Trans. on Power Delivery, An equivalent instantaneous inductance based technique for discrimination between inrush current and internal faults in power transformers, 20, 2473-2482, (2005).

[77] M.Taghipour, A.R.Moradi and M.Yazdani-Asrami, “Identification of magnetizing inrush current in power transformers using GSA trained ANN for educational purposes”, 2010 IEEE Conference on Open Systems (ICOs 2010), Kuala Lumpur, 2010.

[78] Z. Han, S. Liu, S.Gao, Z. Bo, UPEC IEEE,A Novel Detection Criterion for Transformer Inrush Based on Short-Window Filter Algorithm, 1-5, (2008).

[79] J. Ma, D Ye, Z. Wang, J, Wu IPEC IEEE, Identifying Inrush Current Using Sinusoidal Proximity Factor, 215-219, (2010).

[80] Gunda, S.K. and Dhanikonda, V.S. (2021) "Discrimination of transformer inrush currents and internal fault currents using extended Kalman filter algorithm (EKF)," Energies, 14(19), p. 6020. Available at: https://doi.org/10.3390/en14196020.

[81] Pani, S.R., Bera, P.K. and Kumar, V. (2020) "Detection and classification of internal faults in power transformers using tree based classifiers," 2020 IEEE International Conference on Power Electronics, Drives and Energy Systems (PEDES) [Preprint]. Available at: https://doi.org/10.1109/pedes49360.2020.9379641.

[82] A. H. Hamouda, F. Q. Al-anzi, H. K. Gad, A. Gastli, and S. Member,"Numerical Differential Protection Algorithm for Power Transformers," pp.440–445, 2013.

[83] A. M. Mahmoud, M. F. El-Naggar, and E. H. Shehab_Eldin, "A New Technique for Power Transformer Protection Based on Transient Components," Energy Procedia, vol. 14, pp. 318–324, 2012.

[84] IEEE guide for protective relay applications to .power transformers, ieee standard c37.91-2000, 2000

[85] R. S M, "Power quality issues and minimization of magnetizing inrush current by controlled switching in three phase transformers," International Journal of Engineering Trends and Technology, vol. 67, no. 10, pp. 58–65, 2019. doi:10.14445/22315381/ijett-v67i10p211

[86] E. H. Shehab-Eldin and P. G. McLaren, "Travelling wave distance protection- problem areas and solutions," IEEE Trans. Power Deliv., vol. 3, no. 3, pp. 894–902, 1988.

[87] IEEE Guide for Protective Relay Applications to Power Transformers, IEEE Standard C37.91-2000, 2000.

[88] J. Chen et al., "Wavelet transform based on inner product in fault diagnosis of rotating machinery: A review", Mechanical Systems and Signal Processing, vol. 70-71, pp. 1-35, 2016. Available:10.1016/j.ymssp.2015.08.023.

[89] Subasi, "Classification of EMG signals using PSO optimized SVM for diagnosis of neuromuscular disorders," Comput. Biol. Med., vol. 43, no. 5, pp. 576–586, 2013, doi: 10.1016/j.compbiomed.2013.01.020.

[90] Bontempo, R.N.; Bottom, W.P.; Weberp, E.U. Asynchronous motor cage fault detection through electromagnetic torque measurement. Eur. Trans. Electr. Power 1997, 17, pp.529–555.

[91] Elasha, F., Greaves, M., Mba, D., Addali, A. Application of Acoustic Emission in Diagnostic of Bearing Faults within a Helicopter Gearbox. Procedia CIRP 2015, 38, 30–36

[92] D. Pham and M. Castellani, "A comparative study of the Bees Algorithm as a tool for function optimisation", Cogent Engineering, vol. 2, no. 1, p. 1091540, 2015. Available: 10.1080/23311916.2015.1091540.

[93] Boqiang, X. Heming, L. Liling, S. Feature signal extraction of inter-turn short circuit fault in stator windings of induction motors. In Proceedings of the 2002 IEEE International Conference on Industrial Technology, Bangkok, Thailand, 11–14 December 2002; Vol. 1, pp. 97–100.

[94] Islam, R., Khan, S.A., Kim, J.M. Discriminant Feature Distribution Analysis-Based Hybrid Feature Selection for Online Bearing Fault Diagnosis in Induction Motors. J. Sens. 2016, 2016, 7145715.

[95] Meyer, H., Kühnlein, M., Appelhans, T., Nauss, T. Comparison of four machine learning algorithms for their applicability in satellite-based optical rainfall retrievals. Atmos. Res. 2016, 169, 424–433.

[96] Fawzy, H., Rady, E.H.A., Fattah, A.M.A. Comparison between support vector machines and k-nearest neighbor for time series forecasting. J. Math. Comput. Sci. 2020, 10, 2342–2359.

[97] Seshadrinath, J,. Singh, B.; Panigrahi, B.K. Investigation of vibration signatures for multiple fault diagnosis in variable frequency drives using complex wavelets. IEEE Trans. Power Electron. 2014, 29, 936–945.

[98] Z. Moravej, "Evolving Neural Nets for Protection and Condition Monitoring of Power Transformer", Electric Power Components and Systems, vol. 33, no. 11, pp. 1229-1236, 2005. Available: 10.1080/15325000590951636.

[99] J. Yu and Z. Zhang, "Fault Diagnosis of Transformer Based on RBF Neural Network", Applied Mechanics and Materials, vol. 571-572, pp. 201-204, 2014. Available: 10.4028/www.scientific.net/amm.571-572.201.

[100] M. Fatourechi, G. Birch and R. Ward, "Application of a hybrid wavelet feature se-lection method in the design of a self-paced brain interface system", Journal of Neu-roEngineering

and Rehabili-tation, vol. 4, no. 1, 2007. Available: 10.1186/1743-0003-4-11 [Accessed 8 March 2022].

[101] H. Ge, "Feature Selection with Variable Interaction", Journal of Information and Computa-tional Science, vol. 11, no. 11, pp. 3983-3991, 2014. Available: 10.12733/jics20104235.

[102] O. Alomari and Z. Ali Othman, 2022. [Online]. Available: https://www.researchgate.net/publication/267427043_Bees_Algorithm_for_feature_selection_in_Network_Anomaly_detection. [Accessed: 08- Mar- 2022].

[103] S. Abe, “Two-class support Vector Machines,” Support Vector Machines for Pattern Classification, pp. 21–112, 2010. doi:10.1007/978-1-84996-098-4_2.

[104] Nie, F., Zhu, W. And Li, X., 2020. Decision tree svm: an extension of linear svm for non-linear classification. Neurocomputing, 401, pp.153-159.

[105] Z. Moravej, S. A. Banihashemi, and M. H. Velayati, “Power Quality Events Classification and recognition using a novel support vector algorithm,” Energy Conversion and Management, vol. 50, no. 12, pp. 3071–3077, 2009. doi:10.1016/j.enconman.2009.08.007

Bibliography

[1] S. A. Saleh and M. A. Rahman, "Real-time testing of a WPT-based protection algorithm for three-phase power transformers," in *IEEE Transactions on Industry Applications*, vol. 41, no. 4, pp. 1125-1132, July-Aug. 2005, doi: 10.1109/TIA.2005.851562.

[2] S. A. Saleh and M. A. Rahman, "Testing of a Wavelet-Packet-Transform-Based Differential Protection for Resistance-Grounded Three-Phase Transformers," in *IEEE Transactions on Industry Applications*, vol. 46, no. 3, pp. 1109-1117, May-june 2010, doi: 10.1109/TIA.2010.2046293.

[3] "IEEE Guide for Protecting Power Transformers," in *IEEE Std C37.91-2008 (Revision of IEEE Std C37.91-2000)* , vol., no., pp.1-139, 30 May 2008, doi: 10.1109/IEEESTD.2008.4534870.

[4] Z. Gajic, "Use of Standard 87T Differential Protection for Special Three-Phase Power Transformers—Part II: Application and Testing," in *IEEE Transactions on Power Delivery*, vol. 27, no. 3, pp. 1041-1046, July 2012, doi: 10.1109/TPWRD.2011.2178273.

[5] X. Cheng-jun, D. Wen-liang and H. Dong-yan, "Analysis of transformer inrush under unbalanced three-phase parameters," *2011 4th International Conference on Electric Utility Deregulation and Restructuring and Power Technologies (DRPT)*, Weihai, China, 2011, pp. 894-897, doi: 10.1109/DRPT.2011.5994019.

[6] E. Vázquez, A. Conde and A. Ramírez, "A new transformer differential protection based on Principal Component Analysis," *12th IET International Conference on Developments in Power System Protection (DPSP 2014)*, Copenhagen, Denmark, 2014, pp. 1-7, doi: 10.1049/cp.2014.0114.

[7] Z. Gajic, "Use of Standard 87T Differential Protection for Special Three-Phase Power Transformers—Part I: Theory," in *IEEE Transactions on Power Delivery*, vol. 27, no. 3, pp. 1035-1040, July 2012, doi: 10.1109/TPWRD.2012.2188650.

[8] S. A. Saleh and M. A. Rahman, "Modeling and protection of a three-phase power transformer using wavelet packet transform," in *IEEE Transactions on Power Delivery*, vol. 20, no. 2, pp. 1273-1282, April 2005, doi: 10.1109/TPWRD.2004.834891.

[9] T. Zheng and Q. Liu, "A novel algorithm of calculating the circulating current on the delta side of three phase wye-delta connected transformer," *2009 Transmission & Distribution*

Conference & Exposition: Asia and Pacific, Seoul, Korea (South), 2009, pp. 1-4, doi: 10.1109/TD-ASIA.2009.5356933.

[10] M. Schuster and G. Herold, "Power based differential protection for three phase transformers," *PowerTech Budapest 99. Abstract Records. (Cat. No.99EX376)*, Budapest, Hungary, 1999, pp. 204-, doi: 10.1109/PTC.1999.826636.

[11] Z. Gajic, "Differential Protection Methodology for Arbitrary Three-Phase Power Transformers," *2008 IET 9th International Conference on Developments in Power System Protection (DPSP 2008)*, Glasgow, 2008, pp. 44-49, doi: 10.1049/cp:20080009.

[12] B. Bahmani, M. E. Jahromi and A. M. Ranjbar, "Optimizing the operation of current differential protection by power differential relay for three phase transformer," *2006 IEEE GCC Conference (GCC)*, Manama, Bahrain, 2006, pp. 1-6, doi: 10.1109/IEEEGCC.2006.5686196.

[13] V. I. Nagay, V. V. Nagay and I. V. Nagay, "The Constructing of Backup Protections of Overhead Lines Taking into Account the Effect of the Magnetizing Current of the Power Transformers," *2020 International Russian Automation Conference (RusAutoCon)*, Sochi, Russia, 2020, pp. 681-686, doi: 10.1109/RusAutoCon49822.2020.9208190.

[14] A. Aktaibi and M. A. Rahman, "A new hybrid technique of control and protection for three phase power transformers," *IECON 2012 - 38th Annual Conference on IEEE Industrial Electronics Society*, Montreal, QC, Canada, 2012, pp. 1186-1191, doi: 10.1109/IECON.2012.6388602.

[15] Gao Shibin and Chen Xiaochuan, "Principles of differential protection for three-phase to four-phase power transformer," *2004 Eighth IEE International Conference on Developments in Power System Protection*, Amsterdam, Netherlands, 2004, pp. 339-342 Vol.1, doi: 10.1049/cp:20040132.

[16] M. S. Islam and M. M. Kabir, "ANN Based Discrimination of Inrush and Fault Currents in Three Phase Power Transformer using Statistical Approaches," *2019 4th International Conference on Electrical Information and Communication Technology (EICT)*, Khulna, Bangladesh, 2019, pp. 1-6, doi: 10.1109/EICT48899.2019.9068766.

[17] E. S. Jin, L. L. Liu, Z. Q. Bo and A. Klimek, "Application of equivalent instantaneous inductance algorithm to the Y-Δ three-phase transformer," 2008 IEEE Power

and Energy Society General Meeting - Conversion and Delivery of Electrical Energy in the 21st Century, Pittsburgh, PA, USA, 2008, pp. 1-6, doi: 10.1109/PES.2008.4596319.

[18] M. Alexandrino, G. H. Flores, A. J. Guglielmi Filho, D. Tenfen and M. S. Ortmann, "Three-phase power transformer protection, automation and control specification using IEC 61850 system configuration description language," 16th International Conference on Developments in Power System Protection (DPSP 2022), Hybrid Conference, Newcastle, UK, 2022, pp. 177-182, doi: 10.1049/icp.2022.0933.

[19] "IEEE Guide for Protecting Power Transformers," in IEEE Std C37.91-2021 (Revision of IEEE Std C37.91-2008) , vol., no., pp.1-160, 29 June 2021, doi: 10.1109/IEEESTD.2021.9471045.

[20] S. A. Saleh and M. A. Rahman, "Real-time testing of a WPT-based protection algorithm for three-phase power transformers," Conference Record of the 2004 IEEE Industry Applications Conference, 2004. 39th IAS Annual Meeting., Seattle, WA, USA, 2004, pp. 2430-2436 vol.4, doi: 10.1109/IAS.2004.1348816.

[21] L. M. R. Oliveira and A. J. M. Cardoso, "Comparing power transformer turn-to-turn faults protection methods: Negative sequence component versus space vector algorithms," 2015 IEEE 10th International Symposium on Diagnostics for Electrical Machines, Power Electronics and Drives (SDEMPED), Guarda, Portugal, 2015, pp. 289-295, doi: 10.1109/DEMPED.2015.7303704.

[22] M. Bolhasani, S. S. H. Kamangar and S. Tavakoli, "Determination of distribution function of inrush current in three phase transformer using Monte Carlo method," 2012 47th International Universities Power Engineering Conference (UPEC), Uxbridge, UK, 2012, pp. 1-7, doi: 10.1109/UPEC.2012.6398660.

[23] A. B. Nagdewate and S. R. Paraskar, "Discrimination between magnetizing inrush and Interturn fault current in transformer: Hilbert transform-ANN approach," 2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC), Jalgaon, India, 2016, pp. 466-469, doi: 10.1109/ICGTSPICC.2016.7955346.

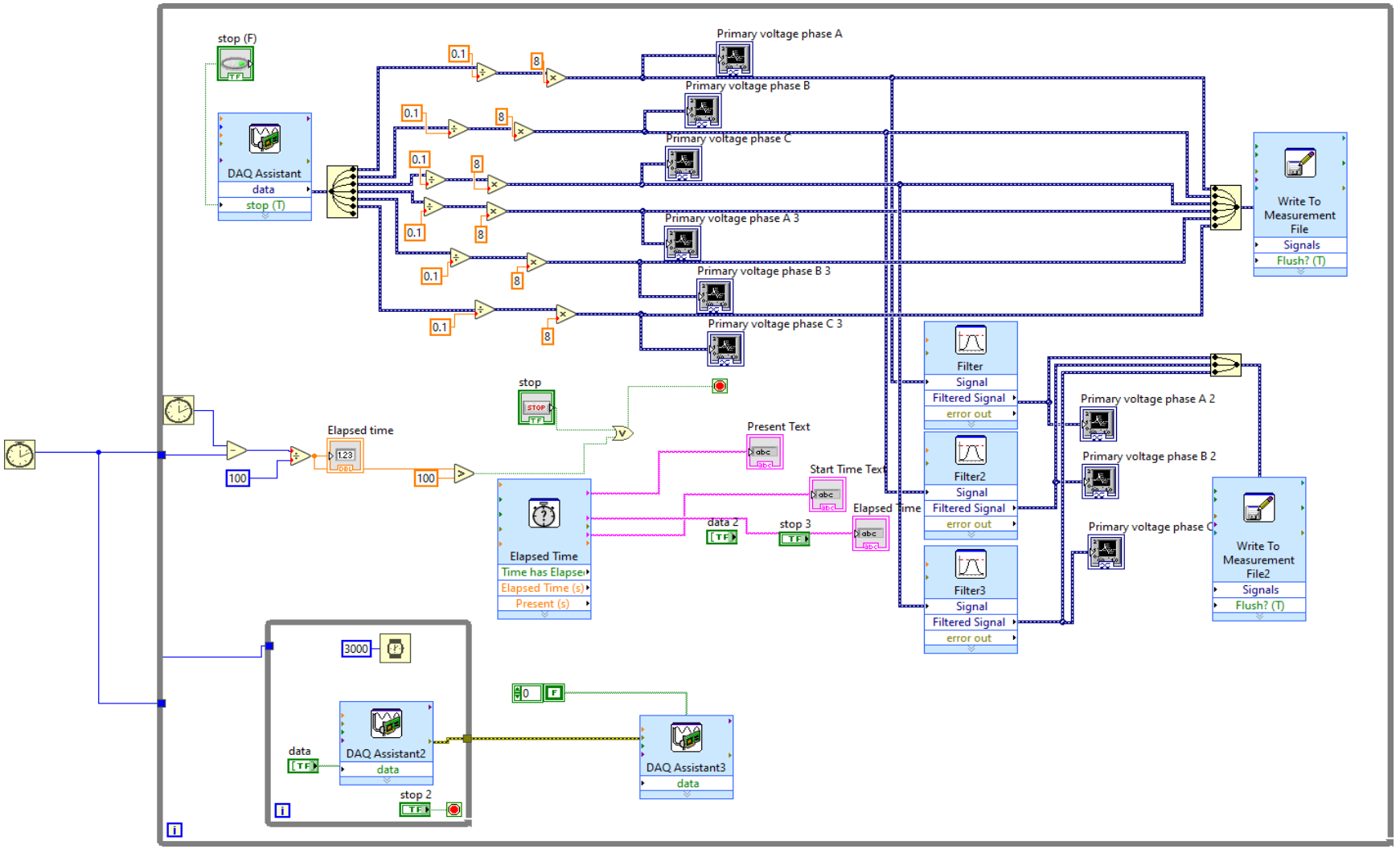

Figure 1 Lapview circuit

Figure 2 Fluxmeter

Figure 3 Power analyzer system

Alienation Coefficient Codes.

Phase A

```
function ra = fcn(ia,iA)
if all(abs(ia)<0.0001)
ia(:,:)=0;
end
if all(abs(iA)<0.0001)
iA(:,:)=0;
end
sA1=iA.*ia;
sAa=sum(sA1);
siA=sum(iA);
sia=sum(ia);
nAa=(21*sAa)-(siA*sia);
sqiA=iA.^2;
sqia=ia.^2;
ssqA=sum(sqiA);
ssqa=sum(sqia);
lAa=((21*ssqA)-siA^2)*((21*ssqa)-sia^2);
dAa=sqrt(lAa);
if dAa==0
ra=0;
else
ra=nAa/dAa;
end
```

Phase B

```
function rb = fcn(ib,iB)
if all(abs(ib)<0.0001)
ib(:,:)=0;
end
```

```
if all(abs(iB)<0.0001)
iB(:,:)=0;
end
sA1=iB.*ib;
sAa=sum(sA1);
siA=sum(iB);
sia=sum(ib);
nAa=(21*sAa)-(siA*sia);
sqiA=iB.^2;
sqia=ib.^2;
ssqA=sum(sqiA);
ssqa=sum(sqia);
lAa=((21*ssqA)-siA^2)*((21*ssqa)-sia^2);
dAa=sqrt(lAa);
if dAa==0
rb=0;
else
rb=nAa/dAa;
end
```

Phase C

```
function rc = fcn(ic,iC)
if all(abs(ic)<0.0001)
ic(:,:)=0;
end
if all(abs(iC)<0.0001)
iC(:,:)=0;
end
sA1=iC.*ic;
sAa=sum(sA1);
siA=sum(iC);
sia=sum(ic);
```

```
nAa=(21*sAa)-(siA*sia);
sqiA=iC.^2;
sqia=ic.^2;
ssqA=sum(sqiA);
ssqa=sum(sqia);
lAa=((21*ssqA)-siA^2)*((21*ssqa)-sia^2);
dAa=sqrt(lAa);
if dAa==0
rc=0;
else
rc=nAa/dAa;
end
```

Alienation

```
function [Ara, Arb,Arc ]= fcn(ra,rb,rc)
Ara=(1-ra^2);
Arb=(1-rb^2);
Arc=(1-rc^2);
%%external fualt or normal operation%%
if  Ara >=0.005 &&  Arb >=0.005 &&  Arc >=0.005
Ara=1;
Arb=1;
Arc=1;
end
%% three phase fualt %%
if  Ara<=0.005 && Arb<=0.005 && Arc<=0.005
Ara=0;
Arb=0;
Arc=0;
end
 %%singal  phase to ground (phase a ) %%
if   Ara<=0.005 && Arb >= 0.005 && Arc >= 0.005
```

```
Ara=0;
Arb=1;
Arc=1;
end
%%  singal  phase to ground (phase b ) %%
if  Ara>= 0.005 && Arb <= 0.005 && Arc >= 0.005
Ara=1;
Arb=0;
Arc=1;
end
%% singal  phase to ground (phase c ) %%
if  Ara >= 0.005 && Arb >= 0.005 && Arc <= 0.005
Ara=1;
Arb=1;
Arc=0;
end
%%  phase to phase or double phase to  ground (phase a,b)%%
if  Ara <= 0.005 && Arb <= 0.005 && Arc >= 0.005
Ara=0;
Arb=0;
Arc=1;
end
%% phase to phase or double phase to  ground (phase b,c)%%
if  Ara >= 0.005 && Arb <= 0.005 && Arc <= 0.005
Ara=1;
Arb=0;
Arc=0;
End
%%  phase to phase or double phase to  ground (phase a,c)%%
 if  Ara <= 0.005 &&  Arb >= 0.005 && Arc <= 0.005
```

Ara=0;

Arb=1;

Arc=0;

End

Machine learning algorithm for detecting and protecting three-phase power transformers codes.

```
clc
clear
close all
%% read dataset:
global TrainSet TrainLabels ValidationSet ValidationLabels modelName
%% prepare dataset + feature extraction:
DATA=xlsread('data.xlsx');
samplesPerSignal=300;
Steps=5;
k=1;
disp('Working on Feature Extraction ...');
for i=1:5
   I=DATA(:,i);
   for j=1:Steps:numel(I)-samplesPerSignal
      signals(:,k)= I(j:j+samplesPerSignal-1,:);
      FeatureVector(k,:)=[OMP_featureExtraction(signals(:,k)),...
                  DWT_featureExtraction(signals(:,k))];
      Labels(k,1)=i;
      k=k+1;
   end
end
disp('Done!');
%% Shuffle Dataset
r=randperm(numel(Labels));
FeatureVector(r,:)=FeatureVector;
Labels(r)=Labels;
%% Divide Dataset (Train - Validation):
```

```
trainRate=0.7;
NSamples=numel(Labels);
NFeatures = size(FeatureVector,2);
NTrain = round(trainRate * NSamples);
TrainSet = FeatureVector(1:NTrain,:);
TrainLabels = Labels(1:NTrain);
ValidationSet = FeatureVector(NTrain+1:end,:);
ValidationLabels = Labels(NTrain+1:end);
disp("valid choices: 'KNN'   'SVM_linear'   'SVM_polynomial'   'SVM_gaussian'  'ANN'")
modelName=input("Classifier: ");
algo=input("Feature Selection Algorithm (valid choices: 'GA' , 'BA') :");
kfold=10;

%% Optimization Parameter Settings:
maxIteration=50;
popsize=20;
nVar=NFeatures;
disp('Working on Feature Selection ...');
%% Call feature selection function
if isequal(algo,'BA')
   [best, best_accuracy]=myBA(maxIteration,popsize,nVar);
else
   [best, best_accuracy]=myGA(maxIteration,popsize,nVar);
end
%% Evaluation:
bestFeatures=find(best==1);
dataset=FeatureVector(:,bestFeatures);
[TP,TN,FP,FN,Ymodel,Ytarget]=kFoldCrossVal(dataset,Labels,kfold);
[Specificity,Accuracy,Precision,Sensitivity,F1_score]=...
       EvaluationParameters(TP,TN,FP,FN)
```

```
positiveClass4ROC=3;
figure
plotroc(Ytarget(:,positiveClass4ROC)'==positiveClass4ROC,Ymodel(:,positiveClass4ROC)'==p
ositiveClass4ROC)
```

BA

```
function [best, best_error]=myBA(MaxIt,popSize,nVar)
%% ABC Parameter Settings:
VarSize=[1 nVar];   % Decision Variables Matrix Size
VarMin=0;         % Decision Variables Lower Bound
VarMax=1;         % Decision Variables Upper Bound
%% Bees Algorithm Parameters
nSelectedSite=round(0.5*popSize);     % Number of Selected Sites
nEliteSite=round(0.4*nSelectedSite);    % Number of Selected Elite Sites
nSelectedSiteBee=round(0.5*popSize);  % Number of Recruited Bees for Selected Sites
nEliteSiteBee=2*nSelectedSiteBee;       % Number of Recruited Bees for Elite Sites
r=0.1*(VarMax-VarMin);                % Neighborhood Radius
rdamp=0.95;                           % Neighborhood Radius Damp Rate
%% Initialization

% Empty Bee Structure
empty_bee.Position=[];
empty_bee.Err=[];

% Initialize Bees Array
bee=repmat(empty_bee,popSize,1);

% Create New Solutions
for i=1:popSize
    bee(i).Position=unifrnd(VarMin,VarMax,VarSize);
    bee(i).Err=objfcn(bee(i).Position>0.5);
```

```
end

% Sort
[~, SortOrder]=sort([bee.Err]);
bee=bee(SortOrder);

% Update Best Solution Ever Found
BestSol=bee(1);

% Array to Hold Best Cost Values
BestCost=zeros(MaxIt,1);

%% Bees Algorithm Main Loop

for it=1:MaxIt

   % Elite Sites
   for i=1:nEliteSite

      bestnewbee.Err=inf;

      for j=1:nEliteSiteBee
         newbee.Position=PerformBeeDance(bee(i).Position,r);
         newbee.Err=objfcn(newbee.Position>0.5);
         if newbee.Err<bestnewbee.Err
            bestnewbee=newbee;
         end
      end

      if bestnewbee.Err<bee(i).Err
         bee(i)=bestnewbee;
```

```
    end

end

% Selected Non-Elite Sites
for i=nEliteSite+1:nSelectedSite

    bestnewbee.Err=inf;

    for j=1:nSelectedSiteBee
        newbee.Position=PerformBeeDance(bee(i).Position,r);
        newbee.Err=objfcn(newbee.Position>0.5);
        if newbee.Err<bestnewbee.Err
            bestnewbee=newbee;
        end
    end

    if bestnewbee.Err<bee(i).Err
        bee(i)=bestnewbee;
    end

end

% Non-Selected Sites
for i=nSelectedSite+1:popSize
    bee(i).Position=unifrnd(VarMin,VarMax,VarSize);
    bee(i).Err=objfcn(bee(i).Position>0.5);
end

% Sort
[~, SortOrder]=sort([bee.Err]);
```

```
    bee=bee(SortOrder);

    % Update Best Solution Ever Found
    BestSol=bee(1);

    % Store Best Cost Ever Found
    BestCost(it)=BestSol.Err;
    % Damp Neighborhood Radius
    r=r*rdamp;
    best_err=BestSol.Err;
    N_Features=sum(BestSol.Position>0.5);
    disp(['iteration ',num2str(it),...
        '/',num2str(MaxIt) ,')  best error: ',num2str(BestCost(it)),' - # of Features:
',num2str(N_Features)])

    if best_err==0 && sum(BestSol.Position>0.5)==1
        disp('Maximum Fitness Achieved.')
        break;
    end
end
best=BestSol.Position>0.5;
best_error=BestSol.Err;
figure
plot(BestCost)
ylabel('Classification Error');
grid
title('Bee Algorithm Convergence Curve');
end
```

GA

```
function [best, best_error]=myGA(maxGeneration,popsize,nvar)
```

```
crossoverRate=0.5;
mutationProbability=0.5;
mutationRate=0.1;
surviveRate=0.5;
% initial population:
for i=1:popsize
    pop(i,:)=randsrc(1,nvar,[0,1]);
end
% Optimization:
for iter=1:maxGeneration
    for i=1:popsize
        cost(i)=objfcn(pop(i,:));
        fitness(i)=1/cost(i);
    end
    probs=fitness/sum(fitness);
    [val,idx]=sort(fitness);
    best=pop(idx(end),:);
    best_error=1/val(end) ;
    N_Features=sum(best);

    disp(['iteration ',num2str(iter),...
        '/',num2str(maxGeneration) ,')  best error: ',num2str(best_error),' - # of Features:
',num2str(N_Features)])
    errplot(iter)=best_error;

    if best_error==0 && sum(best)==1
        disp('Maximum Fitness Achieved.')
        break;
    end
    % transfer (survive) half of good chromosomes to the next generation
    for i=1: round(popsize * surviveRate)
```

```
        newpop(i,:)= pop(idx(end-i+1),:);
    end
    for i=round(popsize * surviveRate)+1:popsize
        % selection:
        p1=randsrc(1,1,[1:popsize;probs]);
        p2=randsrc(1,1,[1:popsize;probs]);
        parent1=pop(p1,:);
        parent2=pop(p2,:);
        % crossover:
        child = parent1;
        child(1:ceil(nvar * crossoverRate)) = parent2(1:ceil(nvar * crossoverRate));
        % mutation:
        if rand<mutationProbability
            r = randperm(nvar , ceil(nvar * mutationRate));
            child(r)=randsrc(1,numel(r) , [0,1]);
        end
        newpop(i,:)=child;
    end
    % replacement:
    pop=newpop;
end
figure
plot(errplot)
ylabel('Classification Error');
grid
title('Genetic Algorithm Convergence Curve');
end
```

Support vector machine for detecting and protecting three-phase power transformers codes.

```
clear, clc;
close all;
raw_data = xlsread('tdata66.xlsx');
% get available data from raw data
% first column data is normal.
% second column data is internal fault.
% third column data is external fault.
real_data = [raw_data(:,1), raw_data(:,4), raw_data(:,7)];
 real_data = [raw_data(:,1), raw_data(:,4), raw_data(:,7),raw_data(:,10),raw_data(:,13)];
% make data pair for classification
x1 = [real_data(:,1)  real_data(:,2)]; % normal and external fault
x2 = [real_data(:,1), real_data(:,3)]; % normal and 5G fault
x3 = [real_data(:,1), real_data(:,4)]; % normal and 2-5G fault
x4 = [real_data(:,1), real_data(:,5)]; % normal and phaseA fault
x5 = [real_data(:,2), real_data(:,3)]; % external and 5G fault
x6 = [real_data(:,2), real_data(:,4)]; % external and 2-5G fault
x7 = [real_data(:,2), real_data(:,5)]; % external and phaseA fault

% output data
y = zeros(2*size(raw_data,1),1);
y(size(raw_data,1)+1 : end) = 1;
rng default

function [best, best_error]=myBA(MaxIt,popSize,nVar)
%% ABC Parameter Settings:
VarSize=[1 nVar];   % Decision Variables Matrix Size
VarMin=0;         % Decision Variables Lower Bound
VarMax=1;         % Decision Variables Upper Bound
```

```
%% Bees Algorithm Parameters
nSelectedSite=round(0.5*popSize);     % Number of Selected Sites
nEliteSite=round(0.4*nSelectedSite);    % Number of Selected Elite Sites
nSelectedSiteBee=round(0.5*popSize);  % Number of Recruited Bees for Selected Sites
nEliteSiteBee=2*nSelectedSiteBee;       % Number of Recruited Bees for Elite Sites
r=0.1*(VarMax-VarMin);                  % Neighborhood Radius
rdamp=0.95;                         % Neighborhood Radius Damp Rate
%% Initialization

% Empty Bee Structure
empty_bee.Position=[];
empty_bee.Err=[];

% Initialize Bees Array
bee=repmat(empty_bee,popSize,1);

% Create New Solutions
for i=1:popSize
   bee(i).Position=unifrnd(VarMin,VarMax,VarSize);
   bee(i).Err=objfcn(bee(i).Position>0.5);
end

% Sort
[~, SortOrder]=sort([bee.Err]);
bee=bee(SortOrder);

% Update Best Solution Ever Found
BestSol=bee(1);

% Array to Hold Best Cost Values
BestCost=zeros(MaxIt,1);
```

```
%% Bees Algorithm Main Loop

for it=1:MaxIt

   % Elite Sites
   for i=1:nEliteSite

      bestnewbee.Err=inf;

      for j=1:nEliteSiteBee
         newbee.Position=PerformBeeDance(bee(i).Position,r);
         newbee.Err=objfcn(newbee.Position>0.5);
         if newbee.Err<bestnewbee.Err
            bestnewbee=newbee;
         end
      end

      if bestnewbee.Err<bee(i).Err
         bee(i)=bestnewbee;
      end

   end

   % Selected Non-Elite Sites
   for i=nEliteSite+1:nSelectedSite

      bestnewbee.Err=inf;

      for j=1:nSelectedSiteBee
         newbee.Position=PerformBeeDance(bee(i).Position,r);
```

```
        newbee.Err=objfcn(newbee.Position>0.5);
        if newbee.Err<bestnewbee.Err
            bestnewbee=newbee;
        end
    end

    if bestnewbee.Err<bee(i).Err
        bee(i)=bestnewbee;
    end

end

% Non-Selected Sites
for i=nSelectedSite+1:popSize
    bee(i).Position=unifrnd(VarMin,VarMax,VarSize);
    bee(i).Err=objfcn(bee(i).Position>0.5);
end

% Sort
[~, SortOrder]=sort([bee.Err]);
bee=bee(SortOrder);

% Update Best Solution Ever Found
BestSol=bee(1);

% Store Best Cost Ever Found
BestCost(it)=BestSol.Err;
% Damp Neighborhood Radius
r=r*rdamp;
best_err=BestSol.Err;
N_Features=sum(BestSol.Position>0.5);
```

```
    disp(['iteration ',num2str(it),...
        '/',num2str(MaxIt) ,')  best error: ',num2str(BestCost(it)),' - # of Features:
',num2str(N_Features)])

    if best_err==0 && sum(BestSol.Position>0.5)==1
        disp('Maximum Fitness Achieved.')
        break;
    end
end
best=BestSol.Position>0.5;
best_error=BestSol.Err;
figure
plot(BestCost)
ylabel('Classification Error');
grid
title('Bee Algorithm Convergence Curve');
end

% %  SVM1(normal data and external fault data)
X1 = x1(:);

% linear kernel function ---------------------------------
% Optimize SVM Classifier
op_linear_model_1 = fitcsvm(X1, y, 'KernelFunction', 'Linear', ...
                    'Standardize', true,...
                    'OptimizeHyperparameters', 'auto', ...
                    'HyperparameterOptimizationOptions', ...
                    struct('UseParallel', true, ...
                    'AcquisitionFunctionName', ...
                    'expected-improvement-plus', ...
                    'MaxObjectiveEvaluations', 10));
```

```
linear_constraint1 = op_linear_model_1.BoxConstraints(1,:);
linear_kernelScale1 = op_linear_model_1.KernelParameters.Scale;
tic
linear_model_1 = fitcsvm(X1, y, 'KernelFunction', 'Linear', ...
                    'Standardize', true,...
                    'BoxConstraint', linear_constraint1, ...
                    'KernelScale', linear_kernelScale1);
t_linear1 = toc;
cv_linear_model_1 = crossval(linear_model_1);
linear_accuracy_1 = (1 - kfoldLoss(cv_linear_model_1, 'LossFun', 'classiferror'))*100
[y_pred_linear1, validationscore_linear1] = kfoldPredict(cv_linear_model_1);
confusion_matrix_linear1 = confusionmat(y, y_pred_linear1);
correct_patterns_linear1 = sum(diag(confusion_matrix_linear1));
incorrect_patterns_linear1 = confusion_matrix_linear1(1,2) + confusion_matrix_linear1(2,1);
% ------------------------------------------------------

% gaussian kernel function -----------------------------
% Optimize SVM Classifier
op_gauss_model_1 = fitcsvm(X1, y, 'KernelFunction', 'gaussian', ...
                    'Standardize', true,...
                    'OptimizeHyperparameters', 'auto', ...
                    'HyperparameterOptimizationOptions', ...
                    struct('UseParallel', true, ...
                    'AcquisitionFunctionName', ...
                    'expected-improvement-plus', ...
                    'MaxObjectiveEvaluations', 10));
gauss_constraint1 = op_gauss_model_1.BoxConstraints(1,:);
gauss_kernelScale1 = op_gauss_model_1.KernelParameters.Scale;
tic
gauss_model_1 = fitcsvm(X1, y, 'KernelFunction', 'gaussian', ...
                    'Standardize', true,...
```

```
                'BoxConstraint', gauss_constraint1, ...
                'KernelScale', gauss_kernelScale1);
t_gauss1 = toc;
cv_gauss_model_1 = crossval(gauss_model_1);
gauss_accuracy_1 = (1 - kfoldLoss(cv_gauss_model_1, 'LossFun', 'classiferror'))*100
[y_pred_gauss1, validationscore_gauss1] = kfoldPredict(cv_gauss_model_1);
confusion_matrix_gauss1 = confusionmat(y, y_pred_gauss1);
correct_patterns_gauss1 = sum(diag(confusion_matrix_gauss1));
incorrect_patterns_gauss1 = confusion_matrix_gauss1(1,2) + confusion_matrix_gauss1(2,1);
y_pred_gauss11 = predict(gauss_model_1, X1);
test_accuracy_gauss1 = sum(y_pred_gauss11 == y)/numel(y)*100
% ------------------------------------------------------
% polynomial kernel function ----------------------------
% Optimize SVM Classifier
op_polynomial_model_1 = fitcsvm(X1, y, 'KernelFunction', 'polynomial', ...
                'Standardize', true,...
                'OptimizeHyperparameters', 'auto', ...
                'HyperparameterOptimizationOptions', ...
                struct('UseParallel', true, ...
                'AcquisitionFunctionName', ...
                'expected-improvement-plus', ...
                'MaxObjectiveEvaluations', 10));
polynomial_constraint1 = op_polynomial_model_1.BoxConstraints(1,:);
polynomial_kernelScale1 = op_polynomial_model_1.KernelParameters.Scale;

poly_accuracy1 = zeros(19,2);
poly_test_pattern1 = zeros(19,2);
tic
for order = 1 : 0.5 : 2
   polynomial_model_1 = fitcsvm(X1, y, 'KernelFunction', 'polynomial', ...
                'PolynomialOrder', order, ...
```

```
                'Standardize', true,...
                'BoxConstraint', polynomial_constraint1, ...
                'KernelScale', polynomial_kernelScale1);

    cv_polynomial_model_1 = crossval(polynomial_model_1);
    polynomial_accuracy_1 = (1 - kfoldLoss(cv_polynomial_model_1, 'LossFun',
'classiferror'))*100;
    poly_accuracy1(order*2-1, 1) = order;
    poly_accuracy1(order*2-1, 2) = polynomial_accuracy_1;
    [y_pred_polynomial1, validationscore_polynomial1] =
kfoldPredict(cv_polynomial_model_1);
    confusion_matrix_polynomial1 = confusionmat(y, y_pred_polynomial1);
    correct_patterns_polynomial1 = sum(diag(confusion_matrix_polynomial1));
    incorrect_patterns_polynomial1 = confusion_matrix_polynomial1(1,2) +
confusion_matrix_polynomial1(2,1);
    poly_test_pattern1(order*2-1, 1) = correct_patterns_polynomial1;
    poly_test_pattern1(order*2-1, 2) = incorrect_patterns_polynomial1;
    disp(['polynomial SVM for internal and external fault data : ', 'Order : ', num2str(order), ',
Penalty due to the error : ',num2str(polynomial_constraint1), ', Accuracy : ',
num2str(polynomial_accuracy_1), '%'])
end
t_polynomial1 = toc/19;
polynomial_model_1 = fitcsvm(X1, y, 'KernelFunction', 'polynomial', ...
                'Standardize', true,...
                'BoxConstraint', polynomial_constraint1, ...
                'KernelScale', polynomial_kernelScale1);

cv_polynomial_model_1 = crossval(polynomial_model_1);
polynomial_accuracy_1 = (1 - kfoldLoss(cv_polynomial_model_1, 'LossFun', 'classiferror'))*100
%
[polynomial_accuracy_1, num] = max(poly_accuracy1(:,2));
```

```
polynomial_order1 = poly_accuracy1(num, 1);
correct_patterns_polynomial1 = poly_test_pattern1(num, 1);
incorrect_patterns_polynomial1 = poly_test_pattern1(num, 2);
y_pred_polynomial11 = predict(polynomial_model_1, X1);
test_accuracy_polynomial1 = sum(y_pred_polynomial11 == y)/numel(y)*100
% ------------------------------------------------------
% %  SVM2(normal and 6G fault data)
X2 = x2(:);
% linear kernel function ---------------------------------
% Optimize SVM Classifier
op_linear_model_2 = fitcsvm(X2, y, 'KernelFunction', 'Linear', ...
                    'Standardize', true,...
                    'OptimizeHyperparameters', 'auto', ...
                    'HyperparameterOptimizationOptions', ...
                    struct('UseParallel', true, ...
                    'AcquisitionFunctionName', ...
                    'expected-improvement-plus', ...
                    'MaxObjectiveEvaluations', 10));
linear_constraint2 = op_linear_model_2.BoxConstraints(1,:);
linear_kernelScale2 = op_linear_model_2.KernelParameters.Scale;
tic
linear_model_2 = fitcsvm(X2, y, 'KernelFunction', 'Linear', ...
                    'Standardize', true,...
                    'BoxConstraint', linear_constraint2, ...
                    'KernelScale', linear_kernelScale2);
t_linear2 = toc;
cv_linear_model_2 = crossval(linear_model_2);
linear_accuracy_2 = (1 - kfoldLoss(cv_linear_model_2, 'LossFun', 'classiferror'))*100
[y_pred_linear2, validationscore_linear2] = kfoldPredict(cv_linear_model_2);
confusion_matrix_linear2 = confusionmat(y, y_pred_linear2);
correct_patterns_linear2 = sum(diag(confusion_matrix_linear2));
```

```
incorrect_patterns_linear2 = confusion_matrix_linear1(1,2) + confusion_matrix_linear1(2,1);
% ------------------------------------------------------

% gaussian kernel function -----------------------------
% Optimize SVM Classifier
op_gauss_model_2 = fitcsvm(X2, y, 'KernelFunction', 'gaussian', ...
                    'Standardize', true,...
                    'OptimizeHyperparameters', 'auto', ...
                    'HyperparameterOptimizationOptions', ...
                    struct('UseParallel', true, ...
                    'AcquisitionFunctionName', ...
                    'expected-improvement-plus', ...
                    'MaxObjectiveEvaluations', 10));
gauss_constraint2 = op_gauss_model_2.BoxConstraints(1,:);
gauss_kernelScale2 = op_gauss_model_2.KernelParameters.Scale;
tic
gauss_model_2 = fitcsvm(X2, y, 'KernelFunction', 'gaussian', ...
                    'Standardize', true,...
                    'BoxConstraint', gauss_constraint2, ...
                    'KernelScale', gauss_kernelScale2);
t_gauss2 = toc;
cv_gauss_model_2 = crossval(gauss_model_2);
gauss_accuracy_2 = (1 - kfoldLoss(cv_gauss_model_2, 'LossFun', 'classiferror'))*100
[y_pred_gauss2, validationscore_gauss2] = kfoldPredict(cv_gauss_model_2);
confusion_matrix_gauss2 = confusionmat(y, y_pred_gauss2);
correct_patterns_gauss2 = sum(diag(confusion_matrix_gauss2));
incorrect_patterns_gauss2 = confusion_matrix_gauss2(1,2) + confusion_matrix_gauss2(2,1);
y_pred_gauss12 = predict(gauss_model_2, X2);
test_accuracy_gauss2 = sum(y_pred_gauss12 == y)/numel(y)*100
% ------------------------------------------------------
% ------------------------------------------------------
```

```
% polynomial kernel function ----------------------------
% Optimize SVM Classifier
op_polynomial_model_2 = fitcsvm(X2, y, 'KernelFunction', 'polynomial', ...
                    'Standardize', true,...
                    'OptimizeHyperparameters', 'auto', ...
                    'HyperparameterOptimizationOptions', ...
                    struct('UseParallel', true, ...
                    'AcquisitionFunctionName', ...
                    'expected-improvement-plus', ...
                    'MaxObjectiveEvaluations', 10));
polynomial_constraint2= op_polynomial_model_2.BoxConstraints(1,:);
polynomial_kernelScale2 = op_polynomial_model_2.KernelParameters.Scale;

poly_accuracy2 = zeros(19,2);
poly_test_pattern2 = zeros(19,2);
tic
for order = 1 : 0.5 : 2
   polynomial_model_2 = fitcsvm(X2, y, 'KernelFunction', 'polynomial', ...
                        'PolynomialOrder', order, ...
                        'Standardize', true,...
                        'BoxConstraint', polynomial_constraint2, ...
                        'KernelScale', polynomial_kernelScale2);

   cv_polynomial_model_2 = crossval(polynomial_model_2);
   polynomial_accuracy_2 = (1 - kfoldLoss(cv_polynomial_model_2, 'LossFun',
'classiferror'))*100;
   poly_accuracy2(order*2-1, 1) = order;
   poly_accuracy2(order*2-1, 2) = polynomial_accuracy_2;
   [y_pred_polynomial2, validationscore_polynomial2] =
kfoldPredict(cv_polynomial_model_2);
   confusion_matrix_polynomial2 = confusionmat(y, y_pred_polynomial2);
```

```
    correct_patterns_polynomial2 = sum(diag(confusion_matrix_polynomial2));
    incorrect_patterns_polynomial2 = confusion_matrix_polynomial2(1,2) +
confusion_matrix_polynomial2(2,1);
    poly_test_pattern2(order*2-1, 1) = correct_patterns_polynomial2;
    poly_test_pattern2(order*2-1, 2) = incorrect_patterns_polynomial2;
    disp(['polynomial SVM2 for internal and external fault data : ', 'Order : ', num2str(order), ',
Penalty due to the error : ',num2str(polynomial_constraint2), ', Accuracy : ',
num2str(polynomial_accuracy_2), '%'])
end
t_polynomial2 = toc/19;
polynomial_model_2 = fitcsvm(X2, y, 'KernelFunction', 'polynomial', ...
                    'Standardize', true,...
                    'BoxConstraint', polynomial_constraint2, ...
                    'KernelScale', polynomial_kernelScale2);

cv_polynomial_model_2 = crossval(polynomial_model_2);
polynomial_accuracy_2 = (1 - kfoldLoss(cv_polynomial_model_2, 'LossFun', 'classiferror'))*100
%
[polynomial_accuracy_2, num] = max(poly_accuracy2(:,2));
polynomial_order2 = poly_accuracy2(num, 1);
correct_patterns_polynomial2 = poly_test_pattern2(num, 1);
incorrect_patterns_polynomial2 = poly_test_pattern2(num, 2);
y_pred_polynomial22 = predict(polynomial_model_2, X2);
test_accuracy_polynomia2 = sum(y_pred_polynomial22 == y)/numel(y)*100
% ------------------------------------------------------

% %  SVM3(normal and 2-5G fault data)
X3 = x3(:);

% linear kernel function ---------------------------------
% Optimize SVM Classifier
```

```
op_linear_model_3= fitcsvm(X3, y, 'KernelFunction', 'Linear', ...
                'Standardize', true,...
                'OptimizeHyperparameters', 'auto', ...
                'HyperparameterOptimizationOptions', ...
                struct('UseParallel', true, ...
                'AcquisitionFunctionName', ...
                'expected-improvement-plus', ...
                'MaxObjectiveEvaluations', 10));
linear_constraint3 = op_linear_model_3.BoxConstraints(1,:);
linear_kernelScale3 = op_linear_model_3.KernelParameters.Scale;
tic
linear_model_3 = fitcsvm(X3, y, 'KernelFunction', 'Linear', ...
                'Standardize', true,...
                'BoxConstraint', linear_constraint3, ...
                'KernelScale', linear_kernelScale3);
t_linear3 = toc;
cv_linear_model_3 = crossval(linear_model_3);
linear_accuracy_3 = (1 - kfoldLoss(cv_linear_model_3, 'LossFun', 'classiferror'))*100
[y_pred_linear3, validationscore_linear3] = kfoldPredict(cv_linear_model_3);
confusion_matrix_linear3 = confusionmat(y, y_pred_linear3);
correct_patterns_linear3 = sum(diag(confusion_matrix_linear3));
incorrect_patterns_linear3 = confusion_matrix_linear3(1,2) + confusion_matrix_linear3(2,1);
% ------------------------------------------------------


% gaussian kernel function -----------------------------
% Optimize SVM Classifier
op_gauss_model_3 = fitcsvm(X3, y, 'KernelFunction', 'gaussian', ...
                'Standardize', true,...
                'OptimizeHyperparameters', 'auto', ...
                'HyperparameterOptimizationOptions', ...
                struct('UseParallel', true, ...
```

```
                'AcquisitionFunctionName', ...
                'expected-improvement-plus', ...
                'MaxObjectiveEvaluations', 10));
gauss_constraint3 = op_gauss_model_3.BoxConstraints(1,:);
gauss_kernelScale3 = op_gauss_model_3.KernelParameters.Scale;
tic
gauss_model_3 = fitcsvm(X3, y, 'KernelFunction', 'gaussian', ...
                'Standardize', true,...
                'BoxConstraint', gauss_constraint3, ...
                'KernelScale', gauss_kernelScale3);
t_gauss3 = toc;
cv_gauss_model_3 = crossval(gauss_model_3);
gauss_accuracy_3 = (1 - kfoldLoss(cv_gauss_model_3, 'LossFun', 'classiferror'))*100
[y_pred_gauss3, validationscore_gauss3] = kfoldPredict(cv_gauss_model_3);
confusion_matrix_gauss3 = confusionmat(y, y_pred_gauss3);
correct_patterns_gauss3 = sum(diag(confusion_matrix_gauss3));
incorrect_patterns_gauss3 = confusion_matrix_gauss1(1,2) + confusion_matrix_gauss1(2,1);
y_pred_gauss13 = predict(gauss_model_3, X3);
test_accuracy_gauss3 = sum(y_pred_gauss13 == y)/numel(y)*100
% ------------------------------------------------------


% ------------------------------------------------------
% polynomial kernel function ----------------------------
% Optimize SVM Classifier
op_polynomial_model_3 = fitcsvm(X3, y, 'KernelFunction', 'polynomial', ...
                'Standardize', true,...
                'OptimizeHyperparameters', 'auto', ...
                'HyperparameterOptimizationOptions', ...
                struct('UseParallel', true, ...
                'AcquisitionFunctionName', ...
                'expected-improvement-plus', ...
```

```
                'MaxObjectiveEvaluations', 10));
polynomial_constraint3 = op_polynomial_model_3.BoxConstraints(1,:);
polynomial_kernelScale3 = op_polynomial_model_3.KernelParameters.Scale;

poly_accuracy3 = zeros(19,2);
poly_test_pattern3 = zeros(19,2);
tic
for order = 1 : 0.5 : 2
   polynomial_model_3 = fitcsvm(X3, y, 'KernelFunction', 'polynomial', ...
                'PolynomialOrder', order, ...
                'Standardize', true,...
                'BoxConstraint', polynomial_constraint3, ...
                'KernelScale', polynomial_kernelScale3);

   cv_polynomial_model_3 = crossval(polynomial_model_3);
   polynomial_accuracy_3 = (1 - kfoldLoss(cv_polynomial_model_3, 'LossFun',
'classiferror'))*100;
   poly_accuracy3(order*2-1, 1) = order;
   poly_accuracy3(order*2-1, 2) = polynomial_accuracy_3;
   [y_pred_polynomial3, validationscore_polynomial3] =
kfoldPredict(cv_polynomial_model_3);
   confusion_matrix_polynomial3 = confusionmat(y, y_pred_polynomial3);
   correct_patterns_polynomial3 = sum(diag(confusion_matrix_polynomial3));
   incorrect_patterns_polynomial3 = confusion_matrix_polynomial3(1,2) +
confusion_matrix_polynomial3(2,1);
   poly_test_pattern3(order*2-1, 1) = correct_patterns_polynomial3;
   poly_test_pattern3(order*2-1, 2) = incorrect_patterns_polynomial3;
   disp(['polynomial SVM for internal and external fault data : ', 'Order : ', num2str(order), ',
Penalty due to the error : ',num2str(polynomial_constraint3), ', Accuracy : ',
num2str(polynomial_accuracy_3), '%'])
end
```

```
t_polynomial3 = toc/19;
polynomial_model_3 = fitcsvm(X3, y, 'KernelFunction', 'polynomial', ...
                    'Standardize', true,...
                    'BoxConstraint', polynomial_constraint3, ...
                    'KernelScale', polynomial_kernelScale3);


cv_polynomial_model_3 = crossval(polynomial_model_3);
polynomial_accuracy_3 = (1 - kfoldLoss(cv_polynomial_model_3, 'LossFun',
'classiferror'))*100;
%
[polynomial_accuracy_3, num] = max(poly_accuracy3(:,2));
polynomial_order3 = poly_accuracy3(num, 1);
correct_patterns_polynomial3 = poly_test_pattern3(num, 1);
incorrect_patterns_polynomial3 = poly_test_pattern3(num, 2);
y_pred_polynomial33 = predict(polynomial_model_3, X3);
test_accuracy_polynomial3 = sum(y_pred_polynomial33 == y)/numel(y)*100
% ------------------------------------------------------
% % %  SVM4(normal and phaseA fault data)
% X4 = x4(:);
%
% % linear kernel function ---------------------------------
% % Optimize SVM Classifier
% op_linear_model_4= fitcsvm(X4, y, 'KernelFunction', 'Linear', ...
                    'Standardize', true,...
                    'OptimizeHyperparameters', 'auto', ...
                    'HyperparameterOptimizationOptions', ...
                    struct('UseParallel', true, ...
                    'AcquisitionFunctionName', ...
                    'expected-improvement-plus', ...
                    'MaxObjectiveEvaluations', 10));
linear_constraint4 = op_linear_model_4.BoxConstraints(1,:);
```

```
linear_kernelScale4 = op_linear_model_4.KernelParameters.Scale;
tic
linear_model_4 = fitcsvm(X4, y, 'KernelFunction', 'Linear', ...
                'Standardize', true,...
                'BoxConstraint', linear_constraint1, ...
                'KernelScale', linear_kernelScale1);
t_linear4 = toc;
cv_linear_model_4 = crossval(linear_model_4);
linear_accuracy_4 = (1 - kfoldLoss(cv_linear_model_4, 'LossFun', 'classiferror'))*100
[y_pred_linear4, validationscore_linear4] = kfoldPredict(cv_linear_model_4);
confusion_matrix_linear4 = confusionmat(y, y_pred_linear4);
correct_patterns_linear4 = sum(diag(confusion_matrix_linear4));
incorrect_patterns_linear4 = confusion_matrix_linear1(1,2) + confusion_matrix_linear1(2,1);
% ------------------------------------------------------


% gaussian kernel function -----------------------------
% Optimize SVM Classifier
op_gauss_model_4 = fitcsvm(X4, y, 'KernelFunction', 'gaussian', ...
                'Standardize', true,...
                'OptimizeHyperparameters', 'auto', ...
                'HyperparameterOptimizationOptions', ...
                struct('UseParallel', true, ...
                'AcquisitionFunctionName', ...
                'expected-improvement-plus', ...
                'MaxObjectiveEvaluations', 10));
gauss_constraint4 = op_gauss_model_4.BoxConstraints(1,:);
gauss_kernelScale4 = op_gauss_model_4.KernelParameters.Scale;
tic
gauss_model_4 = fitcsvm(X4, y, 'KernelFunction', 'gaussian', ...
                'Standardize', true,...
                'BoxConstraint', gauss_constraint1, ...
```

```
                'KernelScale', gauss_kernelScale1);
t_gauss4 = toc;
cv_gauss_model_4 = crossval(gauss_model_4);
gauss_accuracy_4 = (1 - kfoldLoss(cv_gauss_model_4, 'LossFun', 'classiferror'))*100
[y_pred_gauss4, validationscore_gauss4] = kfoldPredict(cv_gauss_model_4);
confusion_matrix_gauss4 = confusionmat(y, y_pred_gauss4);
correct_patterns_gauss4 = sum(diag(confusion_matrix_gauss4));
incorrect_patterns_gauss4 = confusion_matrix_gauss1(1,2) + confusion_matrix_gauss1(2,1);
y_pred_gauss14 = predict(gauss_model_4, X4);
test_accuracy_gauss4 = sum(y_pred_gauss14 == y)/numel(y)*100
% ------------------------------------------------------
% ------------------------------------------------------
% polynomial kernel function ----------------------------
% Optimize SVM Classifier
op_polynomial_model_4 = fitcsvm(X4, y, 'KernelFunction', 'polynomial', ...
                'Standardize', true,...
                'OptimizeHyperparameters', 'auto', ...
                'HyperparameterOptimizationOptions', ...
                struct('UseParallel', true, ...
                'AcquisitionFunctionName', ...
                'expected-improvement-plus', ...
                'MaxObjectiveEvaluations', 10));
polynomial_constraint4 = op_polynomial_model_4.BoxConstraints(1,:);
polynomial_kernelScale4 = op_polynomial_model_4.KernelParameters.Scale;

poly_accuracy4 = zeros(19,2);
poly_test_pattern4 = zeros(19,2);
tic
for order = 1 : 0.5 : 2
   polynomial_model_4 = fitcsvm(X4, y, 'KernelFunction', 'polynomial', ...
                'PolynomialOrder', order, ...
```

```
                'Standardize', true,...
                'BoxConstraint', polynomial_constraint4, ...
                'KernelScale', polynomial_kernelScale4);

    cv_polynomial_model_4 = crossval(polynomial_model_4);
    polynomial_accuracy_4 = (1 - kfoldLoss(cv_polynomial_model_4, 'LossFun',
'classiferror'))*100;
    poly_accuracy4(order*2-1, 1) = order;
    poly_accuracy4(order*2-1, 2) = polynomial_accuracy_4;
    [y_pred_polynomial4, validationscore_polynomial4] =
kfoldPredict(cv_polynomial_model_4);
    confusion_matrix_polynomial4 = confusionmat(y, y_pred_polynomial4);
    correct_patterns_polynomial4 = sum(diag(confusion_matrix_polynomial4));
    incorrect_patterns_polynomial4 = confusion_matrix_polynomial4(1,2) +
confusion_matrix_polynomial4(2,1);
    poly_test_pattern4(order*2-1, 1) = correct_patterns_polynomial4;
    poly_test_pattern4(order*2-1, 2) = incorrect_patterns_polynomial4;
    disp(['polynomial SVM for internal and external fault data : ', 'Order : ', num2str(order), ',
Penalty due to the error : ',num2str(polynomial_constraint4), ', Accuracy : ',
num2str(polynomial_accuracy_4), '%'])
end
t_polynomial4 = toc/19;
polynomial_model_4 = fitcsvm(X4, y, 'KernelFunction', 'polynomial', ...
                'Standardize', true,...
                'BoxConstraint', polynomial_constraint1, ...
                'KernelScale', polynomial_kernelScale1);

cv_polynomial_model_4 = crossval(polynomial_model_4);
polynomial_accuracy_4 = (1 - kfoldLoss(cv_polynomial_model_4, 'LossFun',
'classiferror'))*100;
%
```

```
[polynomial_accuracy_4, num] = max(poly_accuracy4(:,2));
polynomial_order4 = poly_accuracy4(num, 1);
correct_patterns_polynomial4 = poly_test_pattern4(num, 1);
incorrect_patterns_polynomial4 = poly_test_pattern4(num, 2);
y_pred_polynomial44 = predict(polynomial_model_4, X4);
test_accuracy_polynomial4 = sum(y_pred_polynomial44 == y)/numel(y)*100
% ------------------------------------------------------
% %  SVM5(external  and 6G data)
X5 = x5(:);

% linear kernel function ---------------------------------
% Optimize SVM Classifier
op_linear_model_5= fitcsvm(X5, y, 'KernelFunction', 'Linear', ...
                    'Standardize', true,...
                    'OptimizeHyperparameters', 'auto', ...
                    'HyperparameterOptimizationOptions', ...
                    struct('UseParallel', true, ...
                    'AcquisitionFunctionName', ...
                    'expected-improvement-plus', ...
                    'MaxObjectiveEvaluations', 10));
linear_constraint5 = op_linear_model_5.BoxConstraints(1,:);
linear_kernelScale5 = op_linear_model_5.KernelParameters.Scale;
tic
linear_model_5 = fitcsvm(X5, y, 'KernelFunction', 'Linear', ...
                    'Standardize', true,...
                    'BoxConstraint', linear_constraint1, ...
                    'KernelScale', linear_kernelScale1);
t_linear5 = toc;
cv_linear_model_5 = crossval(linear_model_5);
linear_accuracy_5 = (1 - kfoldLoss(cv_linear_model_5, 'LossFun', 'classiferror'))*100
[y_pred_linear5, validationscore_linear5] = kfoldPredict(cv_linear_model_5);
```

```
confusion_matrix_linear5 = confusionmat(y, y_pred_linear5);
correct_patterns_linear5 = sum(diag(confusion_matrix_linear5));
incorrect_patterns_linear5 = confusion_matrix_linear1(1,2) + confusion_matrix_linear1(2,1);
% ------------------------------------------------------

% gaussian kernel function -----------------------------
% Optimize SVM Classifier
op_gauss_model_5 = fitcsvm(X5, y, 'KernelFunction', 'gaussian', ...
                    'Standardize', true,...
                    'OptimizeHyperparameters', 'auto', ...
                    'HyperparameterOptimizationOptions', ...
                    struct('UseParallel', true, ...
                    'AcquisitionFunctionName', ...
                    'expected-improvement-plus', ...
                    'MaxObjectiveEvaluations', 10));
gauss_constraint5 = op_gauss_model_5.BoxConstraints(1,:);
gauss_kernelScale5 = op_gauss_model_5.KernelParameters.Scale;
tic
gauss_model_5 = fitcsvm(X5, y, 'KernelFunction', 'gaussian', ...
                    'Standardize', true,...
                    'BoxConstraint', gauss_constraint1, ...
                    'KernelScale', gauss_kernelScale1);
t_gauss5 = toc;
cv_gauss_model_5 = crossval(gauss_model_5);
gauss_accuracy_5 = (1 - kfoldLoss(cv_gauss_model_5, 'LossFun', 'classiferror'))*100
[y_pred_gauss5, validationscore_gauss5] = kfoldPredict(cv_gauss_model_5);
confusion_matrix_gauss5 = confusionmat(y, y_pred_gauss5);
correct_patterns_gauss5 = sum(diag(confusion_matrix_gauss5));
incorrect_patterns_gauss5 = confusion_matrix_gauss1(1,2) + confusion_matrix_gauss1(2,1);
y_pred_gauss15 = predict(gauss_model_5, X5);
test_accuracy_gauss5 = sum(y_pred_gauss15 == y)/numel(y)*100
```

```
% ----------------------------------------------------

% ----------------------------------------------------
% polynomial kernel function ----------------------------
% Optimize SVM Classifier
op_polynomial_model_5 = fitcsvm(X5, y, 'KernelFunction', 'polynomial', ...
                    'Standardize', true,...
                    'OptimizeHyperparameters', 'auto', ...
                    'HyperparameterOptimizationOptions', ...
                    struct('UseParallel', true, ...
                    'AcquisitionFunctionName', ...
                    'expected-improvement-plus', ...
                    'MaxObjectiveEvaluations', 10));
polynomial_constraint5 = op_polynomial_model_5.BoxConstraints(1,:);
polynomial_kernelScale5 = op_polynomial_model_5.KernelParameters.Scale;

poly_accuracy5 = zeros(19,2);
poly_test_pattern5 = zeros(19,2);
tic
for order = 1 : 0.5 : 2
   polynomial_model_5 = fitcsvm(X5, y, 'KernelFunction', 'polynomial', ...
                       'PolynomialOrder', order, ...
                       'Standardize', true,...
                       'BoxConstraint', polynomial_constraint5, ...
                       'KernelScale', polynomial_kernelScale5);

   cv_polynomial_model_5 = crossval(polynomial_model_5);
   polynomial_accuracy_5 = (1 - kfoldLoss(cv_polynomial_model_5, 'LossFun',
'classiferror'))*100;
   poly_accuracy5(order*2-1, 1) = order;
   poly_accuracy5(order*2-1, 2) = polynomial_accuracy_5;
```

```
    [y_pred_polynomial5, validationscore_polynomial5] =
kfoldPredict(cv_polynomial_model_5);
    confusion_matrix_polynomial5 = confusionmat(y, y_pred_polynomial5);
    correct_patterns_polynomial5 = sum(diag(confusion_matrix_polynomial5));
    incorrect_patterns_polynomial5 = confusion_matrix_polynomial5(1,2) +
confusion_matrix_polynomial5(2,1);
    poly_test_pattern5(order*2-1, 1) = correct_patterns_polynomial5;
    poly_test_pattern5(order*2-1, 2) = incorrect_patterns_polynomial5;
    disp(['polynomial SVM for internal and external fault data : ', 'Order : ', num2str(order), ',
Penalty due to the error : ',num2str(polynomial_constraint5), ', Accuracy : ',
num2str(polynomial_accuracy_5), '%'])
end
t_polynomial5 = toc/19;
polynomial_model_5 = fitcsvm(X5, y, 'KernelFunction', 'polynomial', ...
                    'Standardize', true,...
                    'BoxConstraint', polynomial_constraint5, ...
                    'KernelScale', polynomial_kernelScale5);

cv_polynomial_model_5 = crossval(polynomial_model_5);
polynomial_accuracy_5 = (1 - kfoldLoss(cv_polynomial_model_5, 'LossFun', 'classiferror'))*100
%
[polynomial_accuracy_5, num] = max(poly_accuracy5(:,2));
polynomial_order5 = poly_accuracy5(num, 1);
correct_patterns_polynomial5 = poly_test_pattern5(num, 1);
incorrect_patterns_polynomial5 = poly_test_pattern5(num, 2);
y_pred_polynomial55 = predict(polynomial_model_5, X5);
test_accuracy_polynomial5 = sum(y_pred_polynomial55 == y)/numel(y)*100
% ------------------------------------------------------
% %  SVM6(external  and 2-6G fault data)
X6 = x6(:);
```

```
% linear kernel function --------------------------------
% Optimize SVM Classifier
op_linear_model_6= fitcsvm(X6, y, 'KernelFunction', 'Linear', ...
                'Standardize', true,...
                'OptimizeHyperparameters', 'auto', ...
                'HyperparameterOptimizationOptions', ...
                struct('UseParallel', true, ...
                'AcquisitionFunctionName', ...
                'expected-improvement-plus', ...
                'MaxObjectiveEvaluations', 10));
linear_constraint6 = op_linear_model_6.BoxConstraints(1,:);
linear_kernelScale6 = op_linear_model_6.KernelParameters.Scale;
tic
linear_model_6 = fitcsvm(X6, y, 'KernelFunction', 'Linear', ...
                'Standardize', true,...
                'BoxConstraint', linear_constraint1, ...
                'KernelScale', linear_kernelScale1);
t_linear6 = toc;
cv_linear_model_6 = crossval(linear_model_6);
linear_accuracy_6 = (1 - kfoldLoss(cv_linear_model_6, 'LossFun', 'classiferror'))*100
[y_pred_linear6, validationscore_linear6] = kfoldPredict(cv_linear_model_6);
confusion_matrix_linear6 = confusionmat(y, y_pred_linear6);
correct_patterns_linear6 = sum(diag(confusion_matrix_linear6));
incorrect_patterns_linear6 = confusion_matrix_linear1(1,2) + confusion_matrix_linear1(2,1);
% -----------------------------------------------------

% gaussian kernel function -----------------------------
% Optimize SVM Classifier
op_gauss_model_6 = fitcsvm(X6, y, 'KernelFunction', 'gaussian', ...
                'Standardize', true,...
                'OptimizeHyperparameters', 'auto', ...
```

```
                    'HyperparameterOptimizationOptions', ...
                    struct('UseParallel', true, ...
                    'AcquisitionFunctionName', ...
                    'expected-improvement-plus', ...
                    'MaxObjectiveEvaluations', 10));
gauss_constraint6 = op_gauss_model_6.BoxConstraints(1,:);
gauss_kernelScale6 = op_gauss_model_6.KernelParameters.Scale;
tic
gauss_model_6 = fitcsvm(X6, y, 'KernelFunction', 'gaussian', ...
                    'Standardize', true,...
                    'BoxConstraint', gauss_constraint1, ...
                    'KernelScale', gauss_kernelScale1);
t_gauss6 = toc;
cv_gauss_model_6 = crossval(gauss_model_6);
gauss_accuracy_6 = (1 - kfoldLoss(cv_gauss_model_6, 'LossFun', 'classiferror'))*100
[y_pred_gauss6, validationscore_gauss6] = kfoldPredict(cv_gauss_model_6);
confusion_matrix_gauss6 = confusionmat(y, y_pred_gauss6);
correct_patterns_gauss6 = sum(diag(confusion_matrix_gauss6));
incorrect_patterns_gauss6 = confusion_matrix_gauss1(1,2) + confusion_matrix_gauss1(2,1);
y_pred_gauss16 = predict(gauss_model_6, X6);
test_accuracy_gauss6 = sum(y_pred_gauss16 == y)/numel(y)*100
% % % % % % % % % % % % % % % % % % % % % % % %
% ------------------------------------------------------
% -----------------------------------------------------
% polynomial kernel function -----------------------------
% Optimize SVM Classifier
op_polynomial_model_6 = fitcsvm(X6, y, 'KernelFunction', 'polynomial', ...
                    'Standardize', true,...
                    'OptimizeHyperparameters', 'auto', ...
                    'HyperparameterOptimizationOptions', ...
                    struct('UseParallel', true, ...
```

```
                'AcquisitionFunctionName', ...
                'expected-improvement-plus', ...
                'MaxObjectiveEvaluations', 10));
polynomial_constraint6 = op_polynomial_model_6.BoxConstraints(1,:);
polynomial_kernelScale6 = op_polynomial_model_6.KernelParameters.Scale;

poly_accuracy6 = zeros(19,2);
poly_test_pattern6 = zeros(19,2);
tic
for order = 1 : 0.5 : 2
    polynomial_model_6 = fitcsvm(X6, y, 'KernelFunction', 'polynomial', ...
                    'PolynomialOrder', order, ...
                    'Standardize', true,...
                    'BoxConstraint', polynomial_constraint6, ...
                    'KernelScale', polynomial_kernelScale6);

    cv_polynomial_model_6 = crossval(polynomial_model_6);
    polynomial_accuracy_6 = (1 - kfoldLoss(cv_polynomial_model_6, 'LossFun',
'classiferror'))*100;
    poly_accuracy6(order*2-1, 1) = order;
    poly_accuracy6(order*2-1, 2) = polynomial_accuracy_6;
    [y_pred_polynomial6, validationscore_polynomial6] =
kfoldPredict(cv_polynomial_model_6);
    confusion_matrix_polynomial6 = confusionmat(y, y_pred_polynomial6);
    correct_patterns_polynomial6 = sum(diag(confusion_matrix_polynomial1));
    incorrect_patterns_polynomial6 = confusion_matrix_polynomial6(1,2) +
confusion_matrix_polynomial6(2,1);
    poly_test_pattern6(order*2-1, 1) = correct_patterns_polynomial6;
    poly_test_pattern6(order*2-1, 2) = incorrect_patterns_polynomial6;
```

```
    disp(['polynomial SVM for internal and external fault data : ', 'Order : ', num2str(order), ',
Penalty due to the error : ',num2str(polynomial_constraint6), ', Accuracy : ',
num2str(polynomial_accuracy_6), '%'])
end
t_polynomial1 = toc/19;
polynomial_model_6 = fitcsvm(X6, y, 'KernelFunction', 'polynomial', ...
                    'Standardize', true,...
                    'BoxConstraint', polynomial_constraint6, ...
                    'KernelScale', polynomial_kernelScale6);


cv_polynomial_model_6 = crossval(polynomial_model_6);
polynomial_accuracy_6 = (1 - kfoldLoss(cv_polynomial_model_6, 'LossFun', 'classiferror'))*100
%
[polynomial_accuracy_6, num] = max(poly_accuracy6(:,2));
polynomial_order6 = poly_accuracy6(num, 1);
correct_patterns_polynomial6 = poly_test_pattern6(num, 1);
incorrect_patterns_polynomial6 = poly_test_pattern6(num, 2);
y_pred_polynomial66 = predict(polynomial_model_6, X6);
test_accuracy_polynomial6 = sum(y_pred_polynomial66 == y)/numel(y)*100
% ------------------------------------------------------
% %  SVM7(external and phaseA fault data)
X7 = x7(:);

% linear kernel function --------------------------------
% Optimize SVM Classifier
op_linear_model_7= fitcsvm(X7, y, 'KernelFunction', 'Linear', ...
                    'Standardize', true,...
                    'OptimizeHyperparameters', 'auto', ...
                    'HyperparameterOptimizationOptions', ...
                    struct('UseParallel', true, ...
                    'AcquisitionFunctionName', ...
```

```
                'expected-improvement-plus', ...
                'MaxObjectiveEvaluations', 10));
linear_constraint7 = op_linear_model_7.BoxConstraints(1,:);
linear_kernelScale7 = op_linear_model_7.KernelParameters.Scale;
tic
linear_model_7 = fitcsvm(X7, y, 'KernelFunction', 'Linear', ...
                'Standardize', true,...
                'BoxConstraint', linear_constraint1, ...
                'KernelScale', linear_kernelScale1);
t_linear7 = toc;
cv_linear_model_7 = crossval(linear_model_7);
linear_accuracy_7 = (1 - kfoldLoss(cv_linear_model_7, 'LossFun', 'classiferror'))*100
[y_pred_linear7, validationscore_linear7] = kfoldPredict(cv_linear_model_7);
confusion_matrix_linear7 = confusionmat(y, y_pred_linear7);
correct_patterns_linear7 = sum(diag(confusion_matrix_linear7));
incorrect_patterns_linear7 = confusion_matrix_linear1(1,2) + confusion_matrix_linear1(2,1);
% ------------------------------------------------------


% gaussian kernel function ----------------------------
% Optimize SVM Classifier
op_gauss_model_7 = fitcsvm(X7, y, 'KernelFunction', 'gaussian', ...
                'Standardize', true,...
                'OptimizeHyperparameters', 'auto', ...
                'HyperparameterOptimizationOptions', ...
                struct('UseParallel', true, ...
                'AcquisitionFunctionName', ...
                'expected-improvement-plus', ...
                'MaxObjectiveEvaluations', 10));
gauss_constraint7 = op_gauss_model_7.BoxConstraints(1,:);
gauss_kernelScale7 = op_gauss_model_7.KernelParameters.Scale;
tic
```

```
gauss_model_7 = fitcsvm(X7, y, 'KernelFunction', 'gaussian', ...
                'Standardize', true,...
                'BoxConstraint', gauss_constraint1, ...
                'KernelScale', gauss_kernelScale1);
t_gauss7 = toc;
cv_gauss_model_7 = crossval(gauss_model_7);
gauss_accuracy_7 = (1 - kfoldLoss(cv_gauss_model_7, 'LossFun', 'classiferror'))*100
[y_pred_gauss7, validationscore_gauss7] = kfoldPredict(cv_gauss_model_7);
confusion_matrix_gauss7 = confusionmat(y, y_pred_gauss7);
correct_patterns_gauss7= sum(diag(confusion_matrix_gauss7));
incorrect_patterns_gauss7 = confusion_matrix_gauss1(1,2) + confusion_matrix_gauss1(2,1);
y_pred_gauss17 = predict(gauss_model_7, X7);
test_accuracy_gauss7 = sum(y_pred_gauss17 == y)/numel(y)*100
% ------------------------------------------------------
% polynomial kernel function ----------------------------
% Optimize SVM Classifier
op_polynomial_model_7 = fitcsvm(X7, y, 'KernelFunction', 'polynomial', ...
                'Standardize', true,...
                'OptimizeHyperparameters', 'auto', ...
                'HyperparameterOptimizationOptions', ...
                struct('UseParallel', true, ...
                'AcquisitionFunctionName', ...
                'expected-improvement-plus', ...
                'MaxObjectiveEvaluations', 10));
polynomial_constraint7 = op_polynomial_model_7.BoxConstraints(1,:);
polynomial_kernelScale7 = op_polynomial_model_7.KernelParameters.Scale;

poly_accuracy7 = zeros(19,2);
poly_test_pattern7 = zeros(19,2);
tic
for order = 1 : 0.5 : 2
```

```
    polynomial_model_7 = fitcsvm(X7, y, 'KernelFunction', 'polynomial', ...
                        'PolynomialOrder', order, ...
                        'Standardize', true,...
                        'BoxConstraint', polynomial_constraint7, ...
                        'KernelScale', polynomial_kernelScale7);

    cv_polynomial_model_7 = crossval(polynomial_model_7);
    polynomial_accuracy_7 = (1 - kfoldLoss(cv_polynomial_model_7, 'LossFun',
'classiferror'))*100;
    poly_accuracy7(order*2-1, 1) = order;
    poly_accuracy7(order*2-1, 2) = polynomial_accuracy_7;
    [y_pred_polynomial7, validationscore_polynomial7] =
kfoldPredict(cv_polynomial_model_7);
    confusion_matrix_polynomial7 = confusionmat(y, y_pred_polynomial7);
    correct_patterns_polynomial7 = sum(diag(confusion_matrix_polynomial7));
    incorrect_patterns_polynomial7 = confusion_matrix_polynomial7(1,2) +
confusion_matrix_polynomial7(2,1);
    poly_test_pattern7(order*2-1, 1) = correct_patterns_polynomial7;
    poly_test_pattern7(order*2-1, 2) = incorrect_patterns_polynomial7;
    disp(['polynomial SVM for internal and external fault data : ', 'Order : ', num2str(order), ',
Penalty due to the error : ',num2str(polynomial_constraint7), ', Accuracy : ',
num2str(polynomial_accuracy_7), '%'])
end
t_polynomial7 = toc/19;
polynomial_model_7 = fitcsvm(X7, y, 'KernelFunction', 'polynomial', ...
                        'Standardize', true,...
                        'BoxConstraint', polynomial_constraint7, ...
                        'KernelScale', polynomial_kernelScale7);

cv_polynomial_model_7 = crossval(polynomial_model_7);
polynomial_accuracy_7 = (1 - kfoldLoss(cv_polynomial_model_7, 'LossFun', 'classiferror'))*100
```

```
%
[polynomial_accuracy_7, num] = max(poly_accuracy7(:,2));
polynomial_order7 = poly_accuracy7(num, 1);
correct_patterns_polynomial7 = poly_test_pattern7(num, 1);
incorrect_patterns_polynomial7 = poly_test_pattern7(num, 2);
y_pred_polynomial77 = predict(polynomial_model_7, X7);
test_accuracy_polynomial7 = sum(y_pred_polynomial77 == y)/numel(y)*100
% -----------------------------------------------------
% ----------------------------------------------------------------------------------------------------
% -----------------------------------------------------------------------------------------------------
% % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
% % % % % % % % % % % % % % % %

%% Artificial Neural Net
% normal data and external fault data
tic
ANN_1 = fitcnet(X1, y, ...
   'LayerSizes', 10, ...
   'Activations', 'relu', ...
   'Lambda', 0, ...
   'IterationLimit', 1000, ...
   'Standardize', true, ...
   'Verbose', 1);
t_ANN_1 = toc;
cv_ANN_1 = crossval(ANN_1);
ANN_accuracy_1 = (1 - kfoldLoss(cv_ANN_1, 'LossFun', 'classiferror'))*100
[y_pred_ANN_1, validationscore_ANN_1] = kfoldPredict(cv_ANN_1);
confusion_matrix_ANN_1 = confusionmat(y, y_pred_ANN_1);
correct_patterns_ANN_1 = sum(diag(confusion_matrix_ANN_1));
incorrect_patterns_ANN_1 = confusion_matrix_ANN_1(1,2) + confusion_matrix_ANN_1(2,1);
y_pred_ANN_11 = predict(ANN_1, X1);
```

```
test_accuracy_ANN1 = sum(y_pred_ANN_1 == y)/numel(y)*100
% % % % % % % % % % % % % % % % % % %
% ---------------------------------------------------
% normal data and 6G fault data
tic
ANN_2 = fitcnet(X2, y, ...
   'LayerSizes', 10, ...
   'Activations', 'relu', ...
   'Lambda', 0, ...
   'IterationLimit', 1000, ...
   'Standardize', true, ...
   'Verbose', 1);
t_ANN_2 = toc;
cv_ANN_2 = crossval(ANN_2);
ANN_accuracy_2 = (1 - kfoldLoss(cv_ANN_2, 'LossFun', 'classiferror'))*100
[y_pred_ANN_2, validationscore_ANN_2] = kfoldPredict(cv_ANN_2);
confusion_matrix_ANN_2 = confusionmat(y, y_pred_ANN_2);
correct_patterns_ANN_2 = sum(diag(confusion_matrix_ANN_2));
incorrect_patterns_ANN_2 = confusion_matrix_ANN_2(1,2) + confusion_matrix_ANN_2(2,1);
y_pred_ANN_22 = predict(ANN_2, X2);
test_accuracy_ANN2 = sum(y_pred_ANN_2 == y)/numel(y)*100
% % % % % % % % % % % % % % % % % % % % % % % % % % %
-------------------------------------
internal fault data and 2-6G fault data
tic
ANN_3 = fitcnet(X3, y, ...
   'LayerSizes', 10, ...
   'Activations', 'relu', ...
   'Lambda', 0, ...
   'IterationLimit', 1000, ...
   'Standardize', true, ...
```

```
    'Verbose', 1);
t_ANN_3 = toc;
cv_ANN_3 = crossval(ANN_3);
ANN_accuracy_3 = (1 - kfoldLoss(cv_ANN_3, 'LossFun', 'classiferror'))*100
[y_pred_ANN_3, validationscore_ANN_3] = kfoldPredict(cv_ANN_3);
confusion_matrix_ANN_3 = confusionmat(y, y_pred_ANN_3);
correct_patterns_ANN_3 = sum(diag(confusion_matrix_ANN_3));
incorrect_patterns_ANN_3 = confusion_matrix_ANN_3(1,2) + confusion_matrix_ANN_3(2,1);
y_pred_ANN_33 = predict(ANN_3, X3);
test_accuracy_ANN3 = sum(y_pred_ANN_33 == y)/numel(y)*100
% % % % % % % % % % % % % % % % % % % % % % % % % % % % %
-------------------------------------
% internal fault data and 2-6G fault data
tic
ANN_4 = fitcnet(X4, y, ...
    'LayerSizes', 10, ...
    'Activations', 'relu', ...
    'Lambda', 0, ...
    'IterationLimit', 1000, ...
    'Standardize', true, ...
    'Verbose', 1);
t_ANN_4 = toc;
cv_ANN_4 = crossval(ANN_4);
ANN_accuracy_4 = (1 - kfoldLoss(cv_ANN_4, 'LossFun', 'classiferror'))*100
[y_pred_ANN_4, validationscore_ANN_4] = kfoldPredict(cv_ANN_4);
confusion_matrix_ANN_4 = confusionmat(y, y_pred_ANN_4);
correct_patterns_ANN_4 = sum(diag(confusion_matrix_ANN_4));
incorrect_patterns_ANN_4 = confusion_matrix_ANN_4(1,2) + confusion_matrix_ANN_4(2,1);
y_pred_ANN_44 = predict(ANN_4, X4);
test_accuracy_ANN4 = sum(y_pred_ANN_44 == y)/numel(y)*100
% % % % % % % % % % % % % % % % % % % % % % % % % % % % %
```

```
% --------------------------------------
% internal fault data and 2-6G fault data
tic
ANN_5 = fitcnet(X5, y, ...
   'LayerSizes', 10, ...
   'Activations', 'relu', ...
   'Lambda', 0, ...
   'IterationLimit', 1000, ...
   'Standardize', true, ...
   'Verbose', 1);
t_ANN_5 = toc;
cv_ANN_5 = crossval(ANN_5);
ANN_accuracy_5 = (1 - kfoldLoss(cv_ANN_5, 'LossFun', 'classiferror'))*100
[y_pred_ANN_5, validationscore_ANN_5] = kfoldPredict(cv_ANN_5);
confusion_matrix_ANN_5 = confusionmat(y, y_pred_ANN_5);
correct_patterns_ANN_5 = sum(diag(confusion_matrix_ANN_5));
incorrect_patterns_ANN_5 = confusion_matrix_ANN_5(1,2) + confusion_matrix_ANN_5(2,1);
y_pred_ANN_55 = predict(ANN_5, X5);
test_accuracy_ANN5 = sum(y_pred_ANN_55 == y)/numel(y)*100
% % % % % % % % % % % % % % % % % % % % % % % % % % %
% --------------------------------------
% internal fault data and 2-6G fault data
tic
ANN_6 = fitcnet(X6, y, ...
   'LayerSizes', 10, ...
   'Activations', 'relu', ...
   'Lambda', 0, ...
   'IterationLimit', 1000, ...
   'Standardize', true, ...
   'Verbose', 1);
t_ANN_6 = toc;
```

```
cv_ANN_6 = crossval(ANN_6);
ANN_accuracy_6 = (1 - kfoldLoss(cv_ANN_6, 'LossFun', 'classiferror'))*100
[y_pred_ANN_6, validationscore_ANN_6] = kfoldPredict(cv_ANN_6);
confusion_matrix_ANN_6 = confusionmat(y, y_pred_ANN_6);
correct_patterns_ANN_6 = sum(diag(confusion_matrix_ANN_6));
incorrect_patterns_ANN_6 = confusion_matrix_ANN_6(1,2) + confusion_matrix_ANN_6(2,1);
y_pred_ANN_66 = predict(ANN_6, X6);
test_accuracy_ANN6 = sum(y_pred_ANN_66 == y)/numel(y)*100
%%%%%%%%%%%%%%%%%%%%%%%%%%
% -------------------------------------
% internal fault data and 2-6G fault data
tic
ANN_7= fitcnet(X7, y, ...
    'LayerSizes', 10, ...
    'Activations', 'relu', ...
    'Lambda', 0, ...
    'IterationLimit', 1000, ...
    'Standardize', true, ...
    'Verbose', 1);
t_ANN_7 = toc;
cv_ANN_7 = crossval(ANN_7);
ANN_accuracy_7 = (1 - kfoldLoss(cv_ANN_7, 'LossFun', 'classiferror'))*100
[y_pred_ANN_7, validationscore_ANN_7] = kfoldPredict(cv_ANN_7);
confusion_matrix_ANN_7 = confusionmat(y, y_pred_ANN_7);
correct_patterns_ANN_7 = sum(diag(confusion_matrix_ANN_7));
incorrect_patterns_ANN_7 = confusion_matrix_ANN_7(1,2) + confusion_matrix_ANN_7(2,1);
y_pred_ANN_77 = predict(ANN_7, X7);
test_accuracy_ANN7 = sum(y_pred_ANN_77 == y)/numel(y)*100

% display table 5
```

```
disp('*************** Comparison between accuracy and training time for ANN and SVMs methods ***************')
disp(['SVM1(gaussian) : Accuracy for training patterns : ', num2str(gauss_accuracy_1), ...
    ' %, Accuracy for testing patterns : ', num2str(test_accuracy_gauss1), ...
    ' %, Training time : ', num2str(t_gauss1), 's'])
disp(['SVM2(gaussian) : Accuracy for training patterns : ', num2str(gauss_accuracy_2), ...
    ' %, Accuracy for testing patterns : ', num2str(test_accuracy_gauss2), ...
    ' %, Training time : ', num2str(t_gauss2), 's'])
disp(['SVM3(gaussian) : Accuracy for training patterns : ', num2str(gauss_accuracy_3), ...
    ' %, Accuracy for testing patterns : ', num2str(test_accuracy_gauss3), ...
    ' %, Training time : ', num2str(t_gauss3), 's'])
disp(['SVM4(gaussian) : Accuracy for training patterns : ', num2str(gauss_accuracy_4), ...
    ' %, Accuracy for testing patterns : ', num2str(test_accuracy_gauss4), ...
    ' %, Training time : ', num2str(t_gauss4), 's'])
disp(['SVM5(gaussian) : Accuracy for training patterns : ', num2str(gauss_accuracy_5), ...
    ' %, Accuracy for testing patterns : ', num2str(test_accuracy_gauss5), ...
    ' %, Training time : ', num2str(t_gauss5), 's'])

disp(['SVM6(gaussian) : Accuracy for training patterns : ', num2str(gauss_accuracy_6), ...
    ' %, Accuracy for testing patterns : ', num2str(test_accuracy_gauss6), ...
    ' %, Training time : ', num2str(t_gauss6), 's'])

disp(['SVM7(gaussian) : Accuracy for training patterns : ', num2str(gauss_accuracy_7), ...
    ' %, Accuracy for testing patterns : ', num2str(test_accuracy_gauss7), ...
    ' %, Training time : ', num2str(t_gauss7), 's'])


disp(['ANN1 : Accuracy for training patterns : ', num2str(ANN_accuracy_1), ...
    ' %, Accuracy for testing patterns : ', num2str(test_accuracy_ANN1), ...
    ' %, Training time : ', num2str(t_ANN_1), 's'])
disp(['ANN2 : Accuracy for training patterns : ', num2str(ANN_accuracy_2), ...
```

```
    ' %, Accuracy for testing patterns : ', num2str(test_accuracy_ANN2), ...
    ' %, Training time : ', num2str(t_ANN_2), 's'])
disp(['ANN3 : Accuracy for training patterns : ', num2str(ANN_accuracy_3), ...
    ' %, Accuracy for testing patterns : ', num2str(test_accuracy_ANN3), ...
    ' %, Training time : ', num2str(t_ANN_3), 's'])
disp(['ANN4 : Accuracy for training patterns : ', num2str(ANN_accuracy_4), ...
    ' %, Accuracy for testing patterns : ', num2str(test_accuracy_ANN4), ...
    ' %, Training time : ', num2str(t_ANN_4), 's'])

disp(['ANN5 : Accuracy for training patterns : ', num2str(ANN_accuracy_5), ...
    ' %, Accuracy for testing patterns : ', num2str(test_accuracy_ANN5), ...
    ' %, Training time : ', num2str(t_ANN_5), 's'])
disp(['ANN6 : Accuracy for training patterns : ', num2str(ANN_accuracy_6), ...
    ' %, Accuracy for testing patterns : ', num2str(test_accuracy_ANN6), ...
    ' %, Training time : ', num2str(t_ANN_6), 's'])
disp(['ANN7 : Accuracy for training patterns : ', num2str(ANN_accuracy_7), ...
    ' %, Accuracy for testing patterns : ', num2str(test_accuracy_ANN7), ...
    ' %, Training time : ', num2str(t_ANN_7), 's'])
```