UNSUPERVISED LEARNING FOR CRITICAL EVENT DETECTION USING HISTORICAL PMU DATA

by

Dhawal Joshi

A thesis submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Master of Science in
Applied Energy and Electromechanical Systems

Charlotte

2020

Approved by:

_____
Dr. Maciej Noras

_____
Dr. Umit Cali

_____
Dr. Weimin Wang

_____
Dr. Michael Smith

ABSTRACT

DHAWAL JOSHI. Unsupervised Learning for Critical Event Detection using Historical PMU Data. (Under the direction of DR. MACIEJ NORAS)

A recent increment in power load and the fast reconciliation of sustainable power sources into the power grid requires an improvement in situational awareness. Furthermore, with this increase in power load, the aging grid infrastructure is under stress more than ever causing more event and fault occurrences on the grid. To avoid major faults and blackouts, better monitoring and optimal use of these assets has become necessary. With the advent of phasor measurement unit(PMU) technology, high resolution measurements of the power system has been made possible. However, the vast amount of data generated by PMUs brings the challenge of effectively leveraging the useful information.

This work presents unsupervised learning methods to detect critical events taking place on the grid using historical PMU measurements. The most prominent feature of this method is that it does not require prior knowledge of disturbance samples or grid topology information as opposed to the methods present in the existing literature. Different categories of events are proposed based on visual characteristics of the data and event detection using existing unsupervised anomaly detection methods have been studied. Further, a meta heuristic method particle swarm optimization is explored in order to improve the performance of one of the detector.

ACKNOWLEDGEMENTS

I would like to express the deepest appreciation to my advisor and committee chair, Dr. Maciej Noras, for his support and guidance throughout my graduate studies at University of North Carolina at Charlotte. Without his guidance and persistent help, this thesis would not have been possible.

My gratitude goes to my committee members Dr. Umit Cali, Dr. Weimin Wang and Dr. Michael Smith whose insights and contributions have been very helpful. I would also like to thank all the professors from Engineering Technology and Construction Management who directly or indirectly helped me throughout this work.

I would like to acknowledge the support of my friends Bhav and Vinayak. Their help and constructive criticism were invaluable.

Finally, I would like to thank my parents and my sister Pratyusha for the love and motivation they have given me throughout my graduate studies.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

*i*Tree Isolation Tree

ADTK Anomaly Detection Toolkit

BPA Bonneville Power Administration

DFA Detrended Fluctuation Analysis

DT Decision Tree

EML Extreme Machine Learning

FDR Frequency Data Recorder

GPS Global Positioning System

ICCP Inter-Control Center Communication Protocol

IIT Illinois Institute of Technology

KEPC Korean Electric Power Corporation

LSTM Long Short Term Memory

MSR Mean Spectral Radius

NASPI North American SynchroPhasor Initiative

NERC North American Electric Reliability Corporation

NMSU New Mexico State University

PCA Principal Component Analysis

PDC Phasor Data Concentrator

PDFA Parallel Detrended Flactuation Analysis

PMU Phasor Measurement Unit

PPCA Partitional Principal Component Analysis

PSO Particle Swarm Optimization

RMT Random Matrix Theory

RNN Recurrent Neural Networks

ROCOF Rate of Change of Frequency

RPAD Relative Phase Angle Difference

SAE Stacked Auto Encoder

SCADA Supervisory Control and Data Acquisition

SDX System Data Exchange

SSA Singular Spectral Analysis

SST Singular Spectral Transformation

SVD Singular Value Decomposition

TVA Tennessee Valley Authority

UT Austin University of Texas Austin

UT PanAm University of Texas Pan American

WAMS Wide Area Monitoring System

WECC Western Electricity Coordinating Council

CHAPTER 1: INTRODUCTION

"Situational Awareness" is the way toward understanding the components in a complex framework, observing how they carry on with changes to the framework (for example after some time), and anticipating their status as these progressions happen [2]. A recent increment in power load and the fast reconciliation of sustainable power sources into the power grid requires an improvement in situational awareness [3]. Furthermore, with this increase in power load, the aging grid infrastructure is under stress more than ever causing more event and fault occurrences on the grid.

With the advent of phasor measurement unit (PMU) technology, high resolution measurements of the power system has been made possible resulting in better observability of an electric grid. Unlike the conventional supervisory control and data acquisition (SCADA) system, whose measurement rates are significantly lower (about 1 measurement per second), PMU uses global positioning system (GPS) to time-tag the measurements which are recorded at a rate of 30 to 60 observations per second. The PMU measure voltage and current and with these measurements calculate parameters such as frequency and phase angle. Figure 1.1 shows the typical PMU data collection topology. Usually, multiple PMUs are installed at different substation locations in order to establish a Wide Area Monitoring System (WAMS). The data from these PMUs is sent to the Phasor Data Concentrator (PDC) according to the IEEE C37.118 communication standard where it is aggregated and combined with respect to the time-stamps. Finally, PDC data is sent to the network control center through Inter-Control Center Communications Protocol (ICCP) according to DIN EN 60870-6 where it is analyzed.

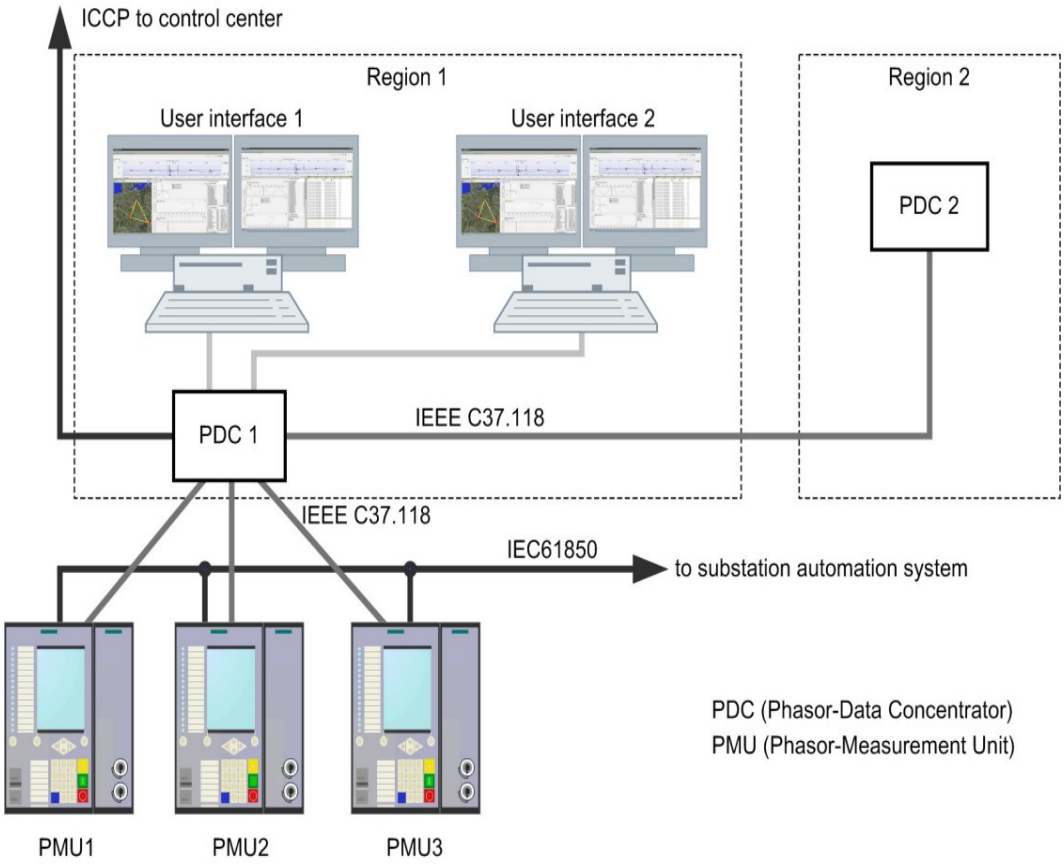


Figure 1.1: PMU Data Collection Topology [4]

Due to increase in renewable energy penetration, two way power flow and for better monitoring of the distribution system, the PMU installations are not only taking place at transmission level but also at distribution level. As of 2018 more than 1500 PMUs are installed in the United States - Table 1.1.

Table 1.1: Number of PMUs installed in the US as of 2018 [1]

Region	Number of Installations
North - East	420
Mid - West	400
South	150
Texas	120
WECC	500

Identification of system trends and behaviour such as voltage phase angle and frequency responses to factors such as variation in load, renewable energy penetration or faults on the grid is made possible by these high resolution measurements. However, due to such high number of observations and ever-increasing installations, the amount of data that needs to be handled is increasing exponentially [5]. Advancements in high-speed analytical capabilities in field of data science and machine learning opens an opportunity to efficiently leverage this data and help system operators make better-informed decisions. Nonetheless, comprehension of power system conduct as seen through PMU data is necessary.

As the measurements from the grid reveal the state of the system, this data is kept confidential and is hard to find for research. Also, the event record files which are maintained by utilities and system operators keep record of only few events which are not enough to train the supervised machine learning in order to achieve accurate event detection. Furthermore, the process of event detection can be accelerated by avoiding power flow calculations. The objective of this research is not only to understand the power system behaviour by detecting events occurring on the grid utilizing PMU data but to perform this in a purely unsupervised and data-driven manner. Various existing unsupervised learning methods for anomaly detection to extract events from the power system measurements are explored in this work. The event categories have been proposed based on the visual characteristics and the ability of various methods to detect these events is studied.

The scope of this work is restricted solely to event detection from the proposed categories in section 5.2. Although the exact time and the nature of event would be captured, cause and location of event will not be studied in this work as the grid topology information is not available. Methodology is strictly based on offline historical data as opposed to real - time data stream. Being an unsupervised method, the number of outliers that needs to be detected must be predefined. Chapter 2 will

present the available and recent literature on event detection from PMU data. In addition to that, various advanced big data and machine learning techniques used previously to analyze power system data will be discussed. Chapter 3 puts forth the thesis statement. Chapter 5 presents the research methodology for feature extraction 5.5.2.1, feature selection 5.5.2.2, detectors 5.5 and event categories 5.2 while chapter 4, will present the data description of the PMU data in use. Chapter 6 consists of experimental setup and results. Finally, in chapter 7 discussion regarding the learned insights and conclusion will be presented.

CHAPTER 2: REVIEW OF PREVIOUS RESEARCH

The task of handling the vast amount of data generated by PMUs and the development of new techniques to analyze this data has gathered interest of many researchers. A considerable amount of work has been done to monitor power systems by utilizing the information extracted from PMUs. Present literature that analyzes PMU data can be categorized into following four major clusters: -

- Dimensionality reduction
- Recovery of lost data
- Event detection
- Control and automation

The literature review in this chapter focuses mainly on event detection from synchrophasor data. However, in order to shed a light on different methods employed on PMU data, a few other studies are analyzed from the clusters mentioned above along with event detection. Research studies considered in this chapter are categorized by the methods and their use case. Usually the metrics used to test the validity of these event detection methods are as follows: -

- Comparison of results with actual event records
- Comparison of results using power flow calculations

2.1 Statistical Methods for Event Detection

One of the early work on event detection using synchrophasor data can be seen in [6]. PMU data along with North American Electric Reliability Corporation's System Data Exchange (NERC SDX) data was used to detect single line outages. NERC SDX data was used to obtain grid topology information. An algorithm was developed to detect event occurrence using edge detection method. Edge-detection operators are based upon second-order derivatives of the intensity. This is used to capture the rate of change in the intensity gradient. The method was tested on a 37 bus simulated data [7] and Tennessee Valley Authority (TVA) 500-KV line outage PMU data. Satisfactory performance of the algorithm was reported even with limited PMU coverage, presence of noise and oscillatory behavior in measurements. However, system topology information was required in this method which is not always readily available. An approach to detect high impedance faults using PMU data was proposed in [8]. This approach used probabilistic function on data converted to Guassian distribution where a null (sample in use = 60Hz) and alternative (sample in use $\neq$ 60Hz) hypothesis was tested to determine the divergence of the system from 60 Hz average to another less probable value. The method was tested on a simulated high impedance fault and effectiveness of the detecting the fault was put forth. Han et al.[9] proposed a novel real-time event detection technique based on random matrix theory (RMT) and Kalman filtering. PMU data conditioning has been done using dynamic Kalman filter. Theoretical basis for event detection was provided by RMT and mean spectral radius (MSR) has been employed as an event detector. The method was tested on IEEE 118 bus simulated data [7] and better performance of the algorithm with both normal and noisy data over the original method [10] was claimed. However, robustness of the method proposed could not be tested due to the lack of testing on real world data.

2.2 Energy Function Based Detection

Brahma et al.[11] reviewed a data-driven model and a physics-based model for real-time identification of dynamic events using PMU data. Challenges and advantages of these models were put forth on basis of data availability, data attributes and processing options. These models were tested on PMU data from New Mexico State University (NMSU) and it was concluded that shapelet method surpasses methods that use domain transformations as far as speed, accuracy and robustness against noise is concerned. Also, a new concept of monitoring change in the individual component of energy functions of the physics-based approach was explored. Negi et al.[12] proposed computation of spectral kurtosis on sum of intrinsic mode functions for event detection where the algorithm compared the maximum energy and root mean square of the energy content of present analysis segment with previous segment from the PMU data. Applied statistical indices flagged specific data, resulting in timely detection of events. Furthermore, due to the distinct nature of transient signals in different regions, the event classification was possible. Characterization algorithm represented signal characterization in terms of short-term energy and group delay. The algorithm was tested on North American SynchroPhasor Initiative (NASPI) PMU data and successful event detection and discrimination was claimed. Another use of energy function can be seen in [13] where a wavelet-based detection algorithm by using a normalized wavelet energy function that monitors the energy detail coefficient within the moving window to classify and detect events is proposed. Wavelet-based detection parameter was designed for robust detection given the non-stationary characteristics of events. The method was tested on real PMU data from Korean Electric Power Corporation (KEPC) and robust detection and zonal information of power system events was claimed.

2.3 Frequency Analysis for Islanding Detection

Frequency component of the data has been used actively to detect events. Lin et al.[14] proposed the frequency difference and the change of angle methods to detect islanding of bulk power system using frequency data recorder (FDR) data. These methods were tested on nine real cases including islanding, generation trip, load shedding and oscillations cases. Based on these cases, sensitivity analysis was performed to get the intensive interval which facilitates reasonable threshold of detecting islanding. Successful detection of islanding was claimed without false triggering for other events. However, detection could not be triggered when low power flow was transferred through lines between islands. Another application of islanding protection method can be seen in [15] where a method of loss-of-mains detection by reconsidering loss-of-mains protection as a synch-check relay is proposed. This method has a zero-detection zone and avoids nuisance tripping as opposed to the conventional methods. Thresholds to detect islanding were set based on experiments carried out under both normal operating conditions and during transient events. Accurate response of the detector to a system-wide transient was claimed.

2.4 Principal Component Analysis for Event Detection and Data Reduction

Ge et al.[16] used Principal Component Analysis (PCA) to measure the trend of data and detect abnormal data resulting from sudden changes for event detection and a second order difference method with hierarchical framework for event notification using PMU data. Further, event-oriented auto-adjustable sliding window method was used for data reduction. These methods were tested on Illinois Institute of Technology (IIT) campus micro-grid PMU data and effective event detection and data reduction was claimed. Having said that, this method uses extensive power flow calculations for which domain knowledge comes into play. Another implementation of PCA can be seen in [17] where a PCA-based method for event detection using PMU data is

proposed. In addition to detection, the method was able to identify location, magnitude and type of fault. Further, similarity of bus dynamic information could also be identified. The method was tested on 42-bus Illinois and 1511-generator system models and although effectiveness of the method was asserted, computational complexity increased with increase in data analyzed. Hence, a partitional PCA (PPCA) method was proposed alongside the method above. Case studies were implemented to demonstrate the efficacy of PPCA method. Rafferty et al.[18] proposed moving window PCA method to set threshold for event detection in order to adapt to time varying behavior of power system. Further, a method to automatically differentiate high and low frequency events for islanded and non-islanded scenarios was developed. Combining these two methods real time event detection and classification was validated using the real and simulated case studies. Although the efficacy of model to detect and classify events was claimed, inability to dis-aggregate multiple loss of load and generation events was put forth.

2.5 Automated Detection and Control

The use case of PMU data has been extended to supervisory protection and automated event diagnosis as seen in [19] where the features are extracted from the time frequency representation of PMU data with strongest disturbance signal and multi class Extreme Machine Learning (EML) classifier were used to classify events. Application of Decision trees (DT) using PMU data for response based control and to enhance relay performance has also been explored [20, 21]. Another application of DTs can be seen in [22] where an algorithm based on decision trees and random forest to identify generator event occurrences in an electric grid using PMU data is presented. A screening classifier was developed using Rate of Change of Frequency (ROCOF) from the data and a secondary classifier was developed for feature identification and extraction. Additionally, a fault map depicting the progression of event occurrence has been created for better troubleshooting of generator events. The algorithm was

tested on BPA PMU data and detection was claimed to be in real time. Having said that, increase in false positives and negatives were reported with the increase in size of the feature vector. Although, DTs are suitable for control applications, they also can be used in order to detect events from PMU data as done in [23]. However, false detection of events by DTs are reported. Also, being a supervised learning method, training is involved which is generally computationally heavy.

2.6 Advanced Big Data and Machine Learning Methods

New technologies were recently introduced to PMU information storage and processing with the advent of big data analytics such as Hadoop along with MapRedcue [24] and data cloud [25]. Khan et al.[26] proposed parallel detrended fluctuation analysis (PDFA) approach for fast event detection on huge PMU data. This approach was implemented in MapReduce model and tested on laboratory based online setup and WAMS installed on the Great Britain transmission system for offline data mining and speedup of computation compared to detrended fluctuation analysis (DFA) was claimed. This speedup in computation was analyzed with Amdahl's law and revision was proposed to enhance the capability of analyzing the performance gain in computation in cluster computing environments. For the purpose of collecting patterns or signatures of events, a variety of machine learning techniques were applied to analyze PMU data. One of the method is Neural networks. Use of neural networks for anomaly detection can be traced back to 1996 where Dasgupta and Forrest [27] proposed negative selection mechanism for anomaly detection which operates on similar lines of neural networks. Malhotra et al.[28] presented stacked Long Short Term Memory (LSTM) network approach for anomaly detection in time series data. Use of LSTM eliminates vanishing gradient problem experienced by Recurrent Neural Networks (RNN). Non-anomalous data was used for training the network which acts as a predictor for the time steps. Prospects of anomalous behavior was assessed by modelling resulting prediction errors as a multivariate Gaussian distribution. The

method was tested on four different PMU data sets and efficacy of the approach was claimed. In [29] rather than inferring outage form sensor data, AC model was used to simulate sensor responses and train neural networks. Although the above mentioned techniques are effective, training the classifier is computationally heavy and addressed in this work. Ren et al.[30] presented a dynamical machine learning method to learn nonlinear and non-stationary PMU measurements and predict system behavior in real time using state space model and Kalman filter. A second order polynomial dynamic regression model was applied on PMU data for a given time frame and the system behavior was predicted for the following time window. The method was tested on PMU data from WECC region and good accuracy and effective real time anomaly detection was claimed. Although rigorous threshold setup brought down the confidence in the confirmed event per unit, it was claimed to be adequate when considering all the units together.

2.7 Unsupervised Detection Methods

To overcome the computational exhaustive nature and requirement of disturbance samples to train the classifier, unsupervised learning methods are proposed. One of the early works on unsupervised learning using PMU data can be observed in [31] used for comprehensive clustering of disturbance events recorded by PMUs where clustering approaches such as hierarchical, partitioning, density-based, grid-based were analyzed and agglomerative hierarchical clustering approach was selected. This unsupervised learning technique was tested on disturbance files and good performance of algorithm to identify clusters from very few know class labels was claimed. Further, the algorithm was able to classify the same event into multiple clusters depending the severity. Although the method was not used to detect events, application of unsupervised learning using PMU data gave way to a new opportunity to analyze events. Another clustering method was proposed in [32] for characterizing smart grid events where an empirical study on application of unsupervised clustering on carefully selected fea-

tures from PMU data was proposed. Features were developed based on observations from voltage magnitude in the data. Different cases such as time series clustering, instantaneous clustering and cluster-specific classification were studied. Klinginsmith et al.[32] claimed better performance of clustering on instantaneous data points compared to time series clustering based on results from the case studies performed on PMU data from Bonneville Power Administration (BPA). Further, identification of unknown events without substantial training data was also claimed. Tang and Yang [33] proposed a novel three step framework for dynamic event monitoring. Every component was formulated individually in the energy function using PMU data. Stacked Auto Encoders (SAE), an unsupervised learning technique, was used to learn features from each component. Further, shared representation between modalities was learned in the feature fusion stage. Finally, a simple neural network was trained to detect and classify events. The method was tested on IEEE 39 bus system and effectiveness of dynamic event detection and classification was claimed. However, insufficient training and deterioration of online application can be caused due to increased deployment of renewable generation sources. Further, physical calculations were involved which calls for grid topology information. Zhou et al.[34] proposed an unsupervised ensemble learning method for fast event detection from PMU data. Two algorithms were developed for training the ensemble model and inferring the anomaly scores over PMU data streams. Anomalies were flagged and further analysis was performed to differentiate events and bad data. The method was tested on simulated and real-world data and efficient detection compared to standalone methods was claimed. Another method proposed by Liang et al.[35] eliminates the requirement of grid topology and physical model information where a rule-based data-driven analytics method for wide area fault detection using PMU data is proposed. Three common fault types (single line to ground, line to line and three phase faults) could be detected with fault location using two approaches, “ABC ” method and “symmetrical component method”.

Fault threshold values were created based on theoretical and synchrophasor data for each fault type. The method was tested on BPA PMU data and good detection and location accuracy for most faults was claimed. However, prior fault information was needed for effectiveness of the method.

CHAPTER 3: THESIS STATEMENT

It is evident from Chapter 2 that the methods varying from energy function analysis to advanced big data and machine learning based analysis are not only capable of successfully detecting events but also can detect its location, identify the type event and even send a corrective control action if the event is detected. Having said that, there is an extensive use of grid topology information for power flow calculations and previous fault record files to train the classifiers by many of these methods.

There is a need to avoid power flow calculations as these calculations require the knowledge of topology of the power systems which is not always readily available. Although supervised learning methods can scan for events once the classifiers are trained using labelled data, training these classifiers itself is computationally heavy and time consuming along with the fact that labels for the data are rarely available. Furthermore, methods such as classification-based and clustering-based have two major drawbacks: -

- Not optimized to detect anomalies resulting in many false alarm
- Constrained to low dimensional data (low number of features present in the data)

Unavailability of grid topology information and labeled data to train the model point towards the need for a method that does not require power flow calculations, unsupervised and the one that can provide effective detection even with low to moderate computational power. The goal of this study is to develop a PMU data analysis approach using the unsupervised learning methods for the purpose of electric grid event detection.

CHAPTER 4: DATA DESCRIPTION

The data used in this thesis was obtained from [36]. Since the PMU data reveal the state of the power system and power system being a critical infrastructure, the data is kept confidential by utilities and the independent system operators that install them. To overcome this, University of Texas at Austin(UT-Austin) introduced an independent synchrophasor network to facilitate researchers at UT-Austin with real time data and its comprehensive analysis [37].

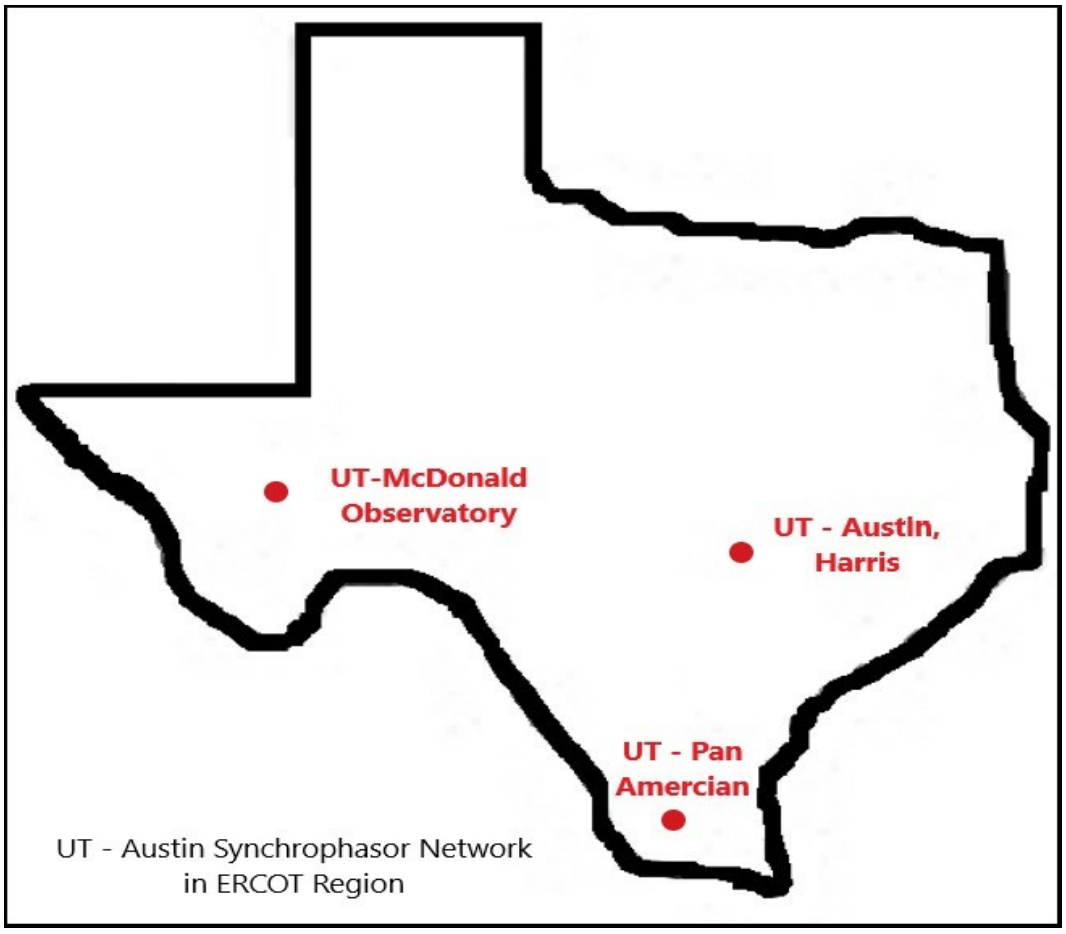


Figure 4.1: Location of PMU stations considered in this study

The data consists of five PMU stations placed within different zones of Electric Reliability Council of Texas(ERCOT) as seen in figure 4.1. However, only three stations are considered in this study due to the fact that two of the PMU stations represent same location of which one PMU measuring the customer level voltage and another PMU measuring the transmission level voltage and the frequency measurement for one of the station being inconsistent when compared to frequency measurements for rest of the stations. Following stations are considered in this study: -

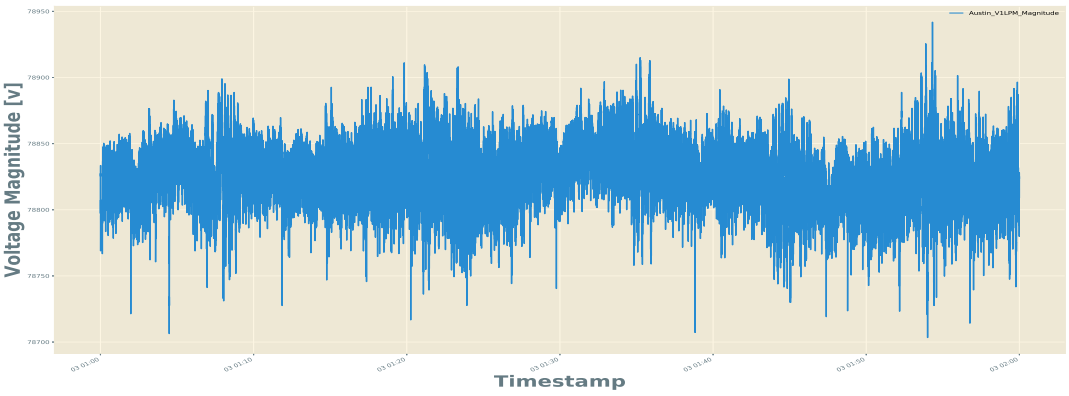
- UT Austin
- UT-McDonald
- UT-Pan America

The McDonald station is placed among various wind farms that are operated as PQ buses. PQ buses provide power at specific real and reactive power; generally wind farms are operated at unity power factor [37].

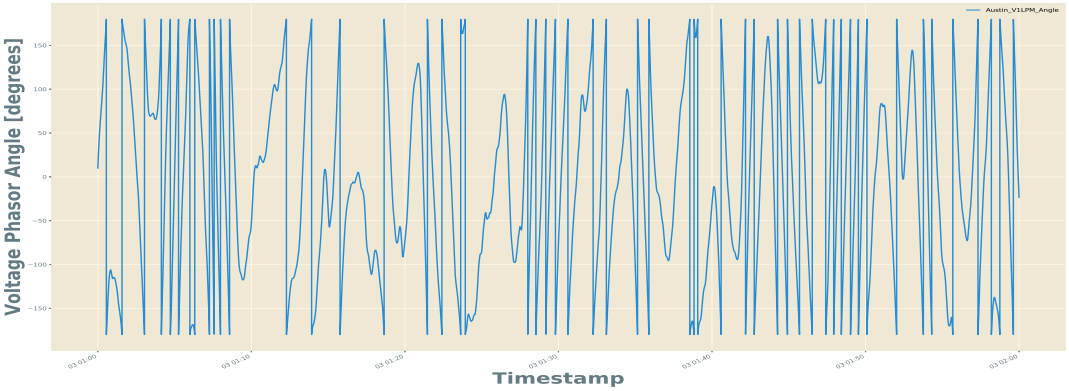
A data sets, representing an hour for January 03, 2012 has been used for event detection in this study. The data is recorded in time series for an hour with a resolution of 30 measurements per second. For each station, following measurements are available: -

- Voltage magnitude (V)
- Voltage phase angle (Degrees)
- Frequency (Hz)

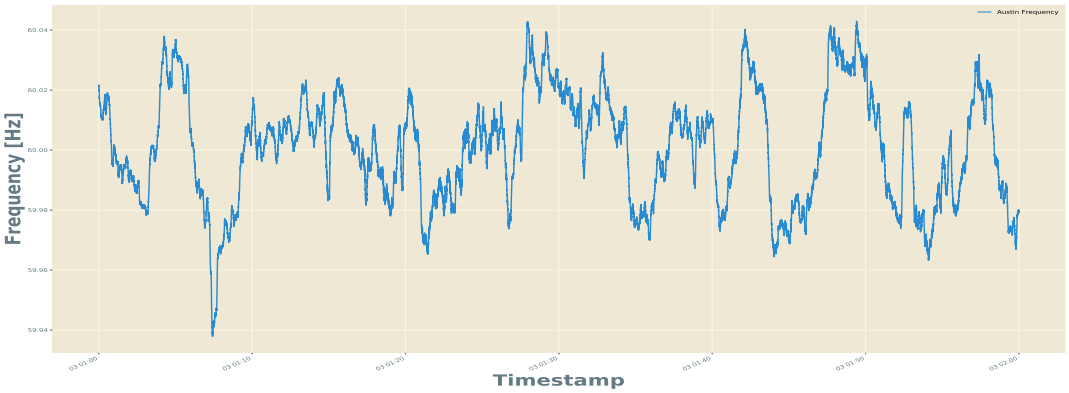
Figure 4.2 shows the time series representation for above mentioned measurements. Current and power information is not available for any of the stations.



(a) Voltage magnitude(V)



(b) Voltage phase angle(Degrees)



(c) Frequency(Hz)

Figure 4.2: Time series representation of measurements for station UT - Austin

CHAPTER 5: METHODOLOGY

This chapter introduces theoretical background on the research methodology. The introduction is not comprehensive and is only relative to building the models for event detection. Prior to the background on the methods, concept of relative phase angle difference and the proposed event categories are put forth.

5.1 Relative Phase Angle Difference

Due to the non stationary characteristics of voltage angle, relative phase angle difference(RPAD) between two station angle is used for detecting events in voltage angle data. In presence of large number PMU installations on the power system, a common reference phase angle is used for analyzing phase angle data. This reference angle can be calculated using centre of gravity concept [38]. However, RPAD is used in this study as there are small number of PMUs installed in the texas synchrophasor network. RPAD can only be used when two PMUs are located at the opposite end of the buses [37].

For the sake of convenience and to avoid any complexities, RPAD for different stations is calculated with respect to UT Austin phase angle. A first-order angle difference is used to obtain RPAD. The difference between consecutive voltage angles for other two PMU station with respect UT Austin station is used to obtain a new RPAD time-series - equation 5.1.

$$Vang_{RPAD} = Vang_{i+1} - Vang_i i = 1, 2, \ldots 108000 \tag{5.1}$$

5.2 Event Categorization

The events are categorized into three sets; impulse, transients and step change/rise or drop. Since classification is not the aim of this research, the events are categorized based on familiar visual characteristics rather than application of any specific classification method. The event categories are proposed in table 5.1.

Table 5.1: Event Categories

Voltage Magnitude	Voltage RPAD	Frequency
Impulse	Impulse	Impulse
Transient	Transient	Transient
Step Change	Step Change	Rise or Drop in Frequency

5.2.1 Impulse Events

Impulse event are characterised by high rate of change in a very short duration of time. The impulses in voltage magnitude and RPAD might be caused due to self clearing faults or due to temporary imbalance in load and generation locally [37]. Impulses in frequency are quite rare and may appear due to temporary imbalance in local load and generation. Due to the fact the RPAD can not be used to detect local events as two stations are involved, power frequency is used to detect local events at different stations.

5.2.2 Transient Events

Large events energize a single or numerous modes inside a power system these can be estimated as transients in PMU voltage information. The transients can also be seen in frequency that are generally caused by load and generation mismatch, loss of generation or loss of load. In case of voltage phase angles, the transients indicate oscillation between a large generator or group of generators near the PMU location.

5.2.3 Step Change/Rise of Drop in Frequency Events

Step change in RPAD is characterized by sudden change in a value to a higher or lower value. Larger level shifts are a result of significant changes in power flow such as a generator unit trip[37]. In this study it is assumed that larger shifts are observed when the generator unit trips and smaller level shifts are observed for transmission line trips as the exact cause is not known. In order to keep the system stable and to ensure a good power quality, the power system frequency is operated with strict thresholds (± 5 %). Hence, the Step change usually do not occur in frequency. In case frequency, sudden drops or rise in frequency are detected instead of level shifts. These types of frequency events are different to impulse events due to the fact that these events can be observed throughout the system whereas, impulse events occur locally.

Various existing unsupervised anomaly detection methods and tools are used for the task of event detection. The methodology is divided in two approaches as depicted in figure 5.1: -

- Category based method
- Generalized outlier detection method

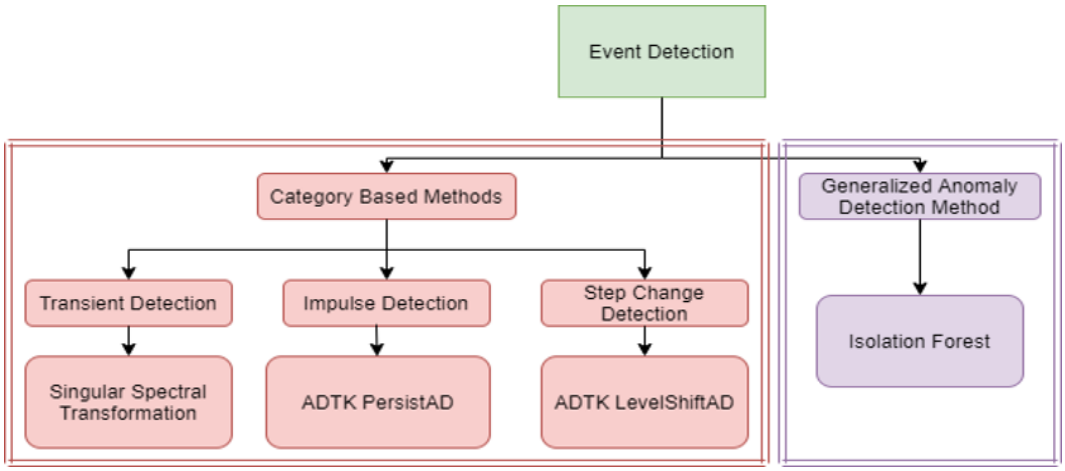


Figure 5.1: Formulation of Methodology

5.3 Hardware and Software Modules

5.3.1 Hardware Modules

This study is performed using hardware from Lambda Labs with following configuration: -

- CPU: 12 Core Intel i7 processor
- RAM: 64 GB

5.3.2 Software Modules

Task of event detection is performed using open source unsupervised learning modules and tool kits. The code was developed using Python 3.6.9 language.

banpei: *banpei* [39] is a python library with two modules; SST for change point detection and hotelling's theory for outlier detection. The SST module for change point detection is used for detecting transients in the data.

Arundo ADTK: Arundo's anomaly detection toolkit(ADTK) is a python package for unsupervised time series anomaly detection [40]. Various components from ADTK can be combined together to build a model to detect anomalies in different scenarios.

sklearn: Isolation forest for unsupervised anomaly detection module from *sklearn* [41] has been used in this study. *sklearn* is a python library which provides simple and effective tools for predictive analysis.

tsfresh: *tsfresh* [42] is a python package that automatically calculate and provide a large number of time series characteristics or features. The package also provides methods to evaluate the importance of features for regression or classification tasks.

PySwarms: PySwarms [43] is a high-level declarative interface for implementing particle swarm optimization(PSO).

5.4 Category Based Methods

5.4.1 Change Point Detection using Singular Spectrum Transformation

Singular spectral transformation (SST) is a nonlinear transformation of an original time series to a new time series that depicts an analogous anomaly metric of the original time series. Anomaly detectors are usually categorized into outlier and change-point detectors. Outliers are local points which are abruptly observed in a series of normal points. On the contrary, change-points depicts changes on a wider scale in terms of characteristics of data points. Hence, SST can be used to effectively detect start and end of transients in the PMU data in this study. This metric is characterized as the distance between two sub-spaces, which is spanned over the left singular vectors. Left singular vectors are acquired by means of the singular value decomposition (SVD) on a Hankel matrix created from strings of the original time series.

5.4.1.1 Pattern Extraction

Singular spectral analysis (SSA) is a preliminary method that is intended to achieve decomposition of a time series into a sum of interpretable factors. Factors such as trend, periodicity and noise can interpreted as representative patterns. SVD is performed on a Hankel matrix that is generated from original time series in order to perform pattern extraction. Let Z be the time series that is transformed to multi-dimensional series H.

$$Z = \{z_1, z_2, \ldots, z_K, \ldots, z_N\} \tag{5.2}$$

$$H = [H_1, H_2, \ldots, H_k] \tag{5.3}$$

where,

$H_i = (z_i, \ldots, z_{i+L-1})^T \, (1 \leq i \leq K)$

$H_i \rightarrow$ L-lagged vectors

H $\rightarrow$ L-trajectory matrix (L $\times$ K Hankel matrix - equation 5.4)

K $\rightarrow$ Window length

L $\rightarrow$ Embedding dimension

$$\mathbf{H} = \begin{pmatrix} z_1 & z_2 & \cdots & z_K \\ z_2 & z_3 & \cdots & z_{K+1} \\ \vdots & \vdots & \vdots & \vdots \\ z_L & z_{L+1} & \cdots & z_N \end{pmatrix} \tag{5.4}$$

In the next step, SVD of the Hankel matrix is performed where it is converted to sum of rank-one bi-orthogonal matrices. Squared singular values (λ_1, λ_2, $\cdots$, λ_l) are assigned to HH^T and the SVD of H is calculated - equation 5.5.

$$\mathbf{H} = \lambda \mathbf{U} \mathbf{V}^T \tag{5.5}$$

where,

$\lambda \rightarrow$ Diagonal matrix with diagonal elements as squared singular values

U $\rightarrow$ Left singular matrix

V $\rightarrow$ Right singular matrix

5.4.1.2 Change-Point Score

The change-point score gives a relative anomaly metric of time series at time t [44]. Left singular vectors is used to calculate this score. H matrix is divided into reference interval ($\mathcal{X}^{\text{ref}}$) and test interval($\mathcal{X}^{\text{test}}$). The change point score (S) is given by equation 5.6. S is a non-dimensional parameter that varies from 0 to 1.

$$S \equiv 1 - \cos \Theta \left(\mathcal{X}^{\text{ref}}, \mathcal{X}^{\text{test}} \right) \tag{5.6}$$

Where,

$\mathcal{X}^{\text{ref}} = \text{span}\{U_{\text{n}}^{\text{ref}}\}$

$\mathcal{X}^{\text{test}} = \text{span}\{U_{\text{m}}^{\text{test}}\}$

Calculation of change point score represents nonlinear transformation of original time series (Z) to a new time series (Z_c).

$$Z \rightarrow Z_c(w, L, g, m, n) \tag{5.7}$$

Where,

$w \rightarrow$ Length of consecutive sub-sequence of original time series

L $\rightarrow$ number of singular vectors

g $\rightarrow$ right singular matrix time point

m $\rightarrow$ length of test subspace

n $\rightarrow$ length of reference subspace

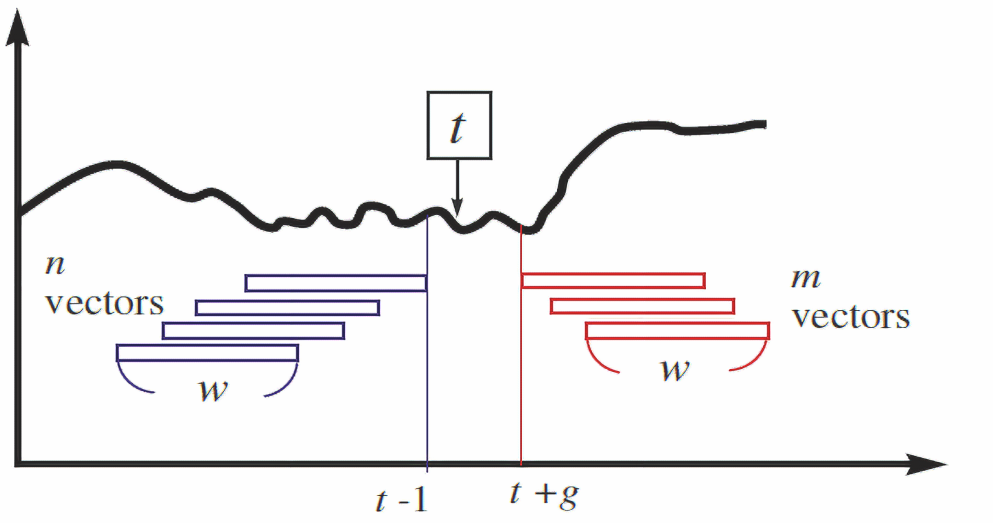


Figure 5.2: Summary of parameters used in SST [45]

5.4.1.3 Krylov Subspace Learning

Although SST is robust and flexible against heterogeneities as it does not make use of any specific generative models, the computational cost of SVD employed by SST is too high to be repeated over complete time series [46] specifically when applied over vast amount of data such as the PMU data used in this work. Ide and Tsuda [46] presented a method where w can be calculated without performing SVD using Krylov approximation resulting in process speed up. Following algorithm provides with the working of Krylov subsampling based SST.

Krylov Subspace Algorithm: *Change Point Score*(*k, C*)

Inputs: k - a positive integer($< w$), C - HH^T
j - current path length (initialized to zero for first iteration)
Output: Change point score S
1. Compute μ at each t
2. **Initialize** $r = \mu$, $\beta_0 = 1$, $q_0 = 0$, z = 0
3. **for** z = k
4. $q_z + 1 = r_z/\beta_z$
5. z $\leftarrow$ z + 1
6. $\alpha_z = q_z^T\ Cq_z$
7. $r_z = Cq_z$ - $\alpha_z\ q_z$ - $\beta_z - 1\ q_z - 1$
8. $\beta_z = ||r_z||$
9. **end for**
10. **define** Change point score (S)
11. $S \simeq 1 - \sum_{i=1}^{r} x^{(i)^2}$
12. **return** S

An open source python library *banpei* [39] is used which has two modules; SST for change point detection and hotelling's theory for outlier detection. The SST module for change point detection is used for detecting transients in the data. The parameters for SST module in *banpei* are listed in table 5.2.

Table 5.2: Parameters for *banpei*

Parameter	Description
w	Window size
m	Number of basis vectors
k	Number of columns for trajectory and test matrix
L	Lag time
is_lancos	Krylov subspace change point score

5.4.2 Arundo ADTK

PersistAD and *LevelShiftAD* from ADTK [47] is used in this work to detect impulses and level shifts respectively.

Two adjacent sliding time windows are implemented in *PersistAD* and *LevelShiftAD* and the difference between the mean or median qualities of these windows is continuously monitored. As shown in equation 5.8 A new time series with these differences over time is created and the detection is performed. At whatever point the measurements in left and right windows are significantly different, it shows a sudden change around this time point. The length of the window depends on the problem at hand. In *PersistAD*, the left window is longer than the right window so that the representative information of the near past is captured. Whereas, in *LevelShiftAD*, both windows are sufficiently long in order to detect stable status. Finally, an interquartile range is calculated given the absolute difference of the two windows and anomalies are defined based on given normal bound. The parameters for *PersistAD* are listed in table 5.3 and the parameters for *LevelShiftAD* are listed in table 5.4.

$$aggregate = mean(left\ \ window) - mean(right\ \ window) \tag{5.8}$$

$$IQR = first\ \ quartile - third\ \ quartile \tag{5.9}$$

Where,

IQR $\rightarrow$ Interquartile range

Table 5.3: Parameters for *PersistAD*

Parameter	Description	Default Value
window	Number of time points in each time window	1
c	Factor used to determine the bound of normal range based on historical interquartile range	3
side	If "both", to detect anomalous positive and negative changes; If "positive", to only detect anomalous positive changes; If "negative", to only detect anomalous negative changes	both
min periods	Minimum number of observations in each window required to have a value for that window	none
agg	Aggregation operation of the time window, either "mean" or "median"	median

Table 5.4: Parameters for *LevelShiftAD*

Parameter	Description	Default Value
window	Number of time points in each time window	10
c	Factor used to determine the bound of normal range based on historical interquartile range	6
side	If "both", to detect anomalous positive and negative changes; If "positive", to only detect anomalous positive changes; If "negative", to only detect anomalous negative changes	both
min periods	Minimum number of observations in each window required to have a value for that window	none

5.5 Generalized Outlier Detection Method

5.5.1 Isolation Forest

Attributes of anomalies are disparate from those of normal instances and they appear in relatively fewer instances compared to the normal data points. Isolation forest is an unsupervised anomaly detection method that uses these distinct properties of anomalies to isolate them from normal group. This method can not be used to detect any specific type of events but the detected events can be categorized after the detection. Ensemble of binary tree structures called isolation trees (iTree) are constructed to isolate the instances for a given data set. While the normal points are more likely to be isolated at the deeper end of the iTree, anomalies are isolated closer to the root due to their susceptibility to isolation.

Randomly generated binary trees recursively partition the data points (explained in training algorithm 2 below) and produce noticeable shorter path for anomalies as seen in figure 5.3 due to the following two reasons: -

- In anomalous region, lower number of anomalies result in smaller number of partitions and hence, tree structures result in shorter paths
- Data points with unique attributes are bound to be isolated from the get-go in the partitioning process

Therefore, when shorter paths for instances are created by the forest of random trees, there is a high probability of those instances being anomalous.

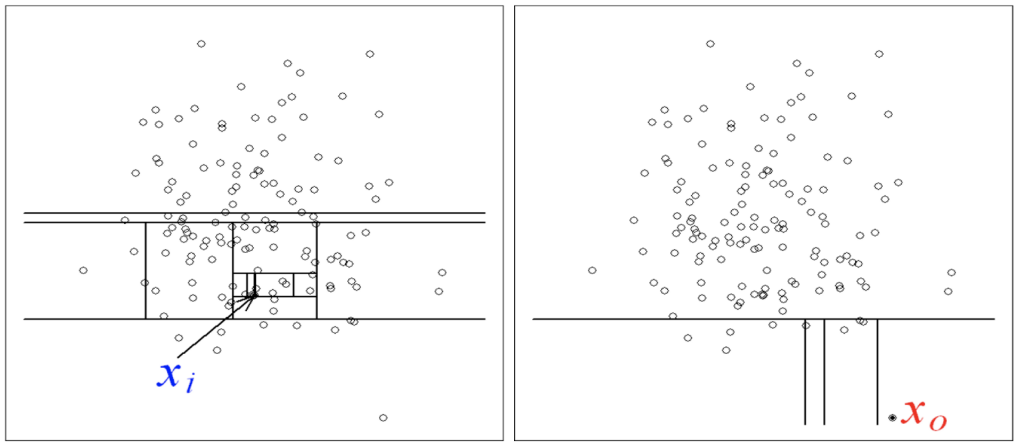


Figure 5.3: Partitioning of Normal vs. Abnormal Instances [48]

Normal instances (X_i) in general require more number of partitions as compared to the number of partitions required by abnormal instances (X_o) - Figure 5.3. As tree structures represent recurrent partitioning, partitions needed to isolate an instance corresponds to the traversal of path length from root node to a terminating node [49]. In contrast to basic density and distance based anomaly detection techniques where only scattered anomalies can be detected, isolation forest is also capable of detecting anomalies surrounded by normal instances [49]. This makes isolation forest suitable to be used in this study as there is a possibility of an event taking place around or near normal operating region.

Detection is performed in two stages; isolation trees are constructed using sub-samples of the given training set in training stage and test instances are passed through isolation tress to get an anomaly score for every instance in evaluation stage. Working of these stages is explained in algorithms below.

Training Algorithm 1: *i*Forest(Z, t, ϕ)

Inputs: Z - input data, t - number of trees, ϕ - sub-sampling size
Output: a set of t
1. **Initialize:** *forest*
2. **for** $i = 1$ to t
3. sample (Z, ϕ) $\rightarrow$ Z'
4. *forest* $\cup$ iTree (Z') $\rightarrow$ *forest*
5. **end for**
6. **return:** *forest*

Training Algorithm 2: *i*Tree(*Z'*)

Inputs: Z' - input data
Output: an iTree
1. **if** Z' isn't divisible
2. **then** return exNode $\{|Z'| \rightarrow \text{Size}\}$
3. **else**
4. let A be attributes list in Z'
5. randomly select an attribute $a \in A$
6. randomly select a split point p between the *max* and *min* values of attribute a in Z'
7. *filter*(Z', $a < p$) $\rightarrow$ Z'
8. *filter*(Z', $a \geq p$) $\rightarrow$ Z'
9. return *inNode* $\{i\text{Tree}(Z_l) \rightarrow \textit{Left}\}$,
10. $\{i\text{Tree}(Z_r) \rightarrow \textit{Right}\}$,
11. $\{a \rightarrow \textit{SplitAtt}\}$,
12. $\{p \rightarrow \textit{SplitValue}\}$
13. **end if**

Subsampling size ϕ control the training data size and is generally set to 2^8 or 256. Number of trees t control the ensemble size and the default value is set to 100 [49].

Evaluation Algorithm: *PathLengths*(*z, T, hlim, j*)
Inputs: z - an instance, T - an iTree, *hlim* - height limit, j - current path length (initialized to zero for first iteration)
Output: path length of z
1. **if** T is an external node or $j \geq hlim$
2. **then** return j + k($T.size$) {k(.) defined in equation 5.12}
3. **end if**
4. $T.splittAtt \rightarrow c$
5. **if** $z_a < T.splitValue$
6. **then** return *PathLengths*(z, $T.left$, *hlim*, $j + 1$)
7. **else** $\{z_a \geq T.splitValue\}$
8. **then** return *PathLengths*(z, $T.right$, *hlim*, $j + 1$)
9. **end if**

In evaluation algorithm, a single path length h(z) is acquired by calculating the number of edges j from root node to external node. j and adjustment *k(Size)* is returned when the predetermined height *hlim* is reached by the traversal. Anomaly score is calculated once h(z) is available from each tree of the ensemble. Average of h(z) is given by equation 5.10 [50].

$$k(\phi) = \begin{cases} 2H(\phi - 1) - 2(\phi - 1)/\phi & \text{for} \phi > 2 \\ 1 & \text{for } \phi = 2 \\ 0 & \text{otherwise} \end{cases} \tag{5.10}$$

where, H(i) is a harmonic number that can be estimated by ln(i) + 0.5772156649.

The anomaly score s of an instance z is given by equation 5.11.

$$s(z, \phi) = 2^{-\frac{E(h(z))}{k(\phi)}} \tag{5.11}$$

where, E($h(z)$) is average of h(z) from collection of iTrees. Values of anomaly score is provided by following three conditions [49].

- when E($h(z)$) $\rightarrow 0$, $s \rightarrow 1$;

- when E($h(z)$) $\rightarrow$ ϕ - 1, $s \rightarrow 0$; and
- when E($h(z)$) $\rightarrow$ k(ϕ), $s \rightarrow 0.5$.

This anomaly score would be calculated for each value from the PMU data set in this study and following would be inferred from anomaly score s: -

i. If s is close to 1, instance is an anomaly

ii. If s is smaller than 0.5, instance can be safely regarded as normal

iii. If all instances return s $\approx$ 0.5, there isn't definite presence of anomalies in the given sample

Isolation forest module from *sklearn* [41] an open source python library is used to detect the events. Anomaly score in this module is different to the one mentioned above - table 5.5. Parameters of this module are listed in table 5.6.

Table 5.5: Anomaly Score

Instance Type	Anomaly Score
Normal Instance	1
Anomaly	-1

Table 5.6: Parameters for Isolation Forest

Parameter	Description	Default Value
n_estimators	Number of base estimators in the ensemble	10
max_samples	Number of samples to draw from X to train each base estimator	auto
contamination	Amount of contamination of the data set, i.e. the proportion of outliers in the data set. Used when fitting to define the threshold on the scores of the samples If 'auto', the threshold is determined as in the original paper If float, the contamination should be in the range [0, 0.5]	auto

5.5.2 Feature Extraction and Feature Optimization

As the isolation forest is a generalized outlier detection method, features that give a more extensive characteristics of the measurement in the data needs to be extracted and used as an input to the isolation forest algorithm in order to effectively detect events. Following sections explains the methodology to extract and optimize these extracted features.

5.5.2.1 Feature Extraction

A python library *tsfresh* [42] is used to extract the features from time series PMU data. *tsfresh* provides automated time series feature extraction and and feature selection. However, only extraction would be performed using *tsfresh* as selection is designed for supervised learning [51]. Input and output objects and *scikit-learn* compatible transformer classes are provided and deployed where transformers are used to translate the data for a field into a format that can be displayed in a specific form.w It also implements the application programming interfaces of Python machine learning and data analysis frameworks such as numpy [52], pandas [53], scipy [54], keras and tensorflow [55].

The feature extraction module consists of various feature calculators and the logic to implement them effectively in time series analysis. A setting dictionary controls the number and parameters of all the extracted features. The extracted features can be distributed in five major categories.

- Statistical features (mean, variance, standard deviation, kurtosis)
- Energy based features (absolute energy)
- Logic based features (absolute sum of changes, count above)
- Value based features (max, min, range count)
- Wavelet transform based feature (continuous wavelet transform)
- Frequency transform based feature (fast fourier transform)

5.5.2.2 Feature Optimization

As features extracted from *tsfresh* act as an input to isolation forest, its performance can be improved by optimizing these extracted features. However, more than 400 features are extracted by *tsfresh* depending on the data set in use and classifying them

in terms of their importance is a challenging and time consuming process. Hence, to observe the effect of change in detection accuracy by optimizing these features, a meta heuristic optimization method particle swarm optimization (PSO) is explored in this study.

PSO is a stochastic optimization method based on swarm, proposed originally by Kennedy and Eberhart [56]. PSO algorithm is a sort of searching procedure dependent on the swarm of features, in which every individual feature is known as a particle characterized as a potential solution of the optimized problem in D-dimensional search space and it can retain the optimal position of the swarm and that of its own, just as the velocity. At every stage, the particle's data is consolidated together to modify and adjust the speed of each dimension, which is utilized to register the new position of the particle. Particles change their states continually in the multi-dimensional search space until they arrive at a optimal state, or until the given number of iterations are reached. The one of a kind association among various dimensions of the problem space are presented by means of the objective functions.

Each particle in the swarm is composed of d-dimensional vectors, where d is the dimensionality of search space [57]. The vectors are as follows: -

- Current position x_{nd}
- Previous best position p_best_{nd}
- Velocity v_{nd}

Change in velocity and position of the particles is governed by equation 5.12 and equation 5.13 respectively [58].

$$v_{nd}(i+1) = wv_{nd}(i) + c_1 \,\mathrm{rand}_1 \,(p_{nd} - x_{nd}) + c_2 \,\mathrm{rand}_2 \,(p_{g_d} - x_{nd}) \tag{5.12}$$

Where,

$n \rightarrow$ particle

$i \rightarrow$ iteration

$v_{nd}(i+1) \rightarrow$ new velocity for n^th particle

$w \rightarrow$ inertia weight

c_1 & $c_2 \rightarrow$ acceleration constant

$p_{nd} \rightarrow$ best individual position

$p_{gd} \rightarrow$ best position of all particles

$$x_{nd}(i+1) = x_{nd}(i) + v_{nd}(i+1) \tag{5.13}$$

Where,

$x_{nd}(i+1) \rightarrow$ best position after update

The working of PSO is explained in the following algorithm.

Particle Swarm Optimization

Inputs:

Output:

1. **for** each particle n
2. **for** each dimension d
3. Initialize position x_nd randomly
4. Initialize velocity v_nd randomly
5. **end for**
6. **end for**
7. Iteration i = 1
8. **do**
9. **for** each particle n
10. Calculate fitness value
11. **if** fitness value better than p_best_nd in past
12. Set present fitness value as the p_best_nd
13. **end if**
14. **end for**
15. g_best_d = particle having the best fitness value
16. **for** each particle n
17. **for** each dimension d
18. Change velocity and position of the particle according to equations 5.12 and 5.13
19. **end for**
20. **end for**
21. i = i + 1
22. **while** minimum error criteria or maximum iterations are not reached
23. **end while**

CHAPTER 6: EXPERIMENTS AND RESULTS

The task of event detection in this study is performed individually on each available parameter; voltage magnitude, voltage phase angle and frequency according to event categories listed in table 5.1. This chapter is outlined on the basis of type of event to be detected; impulse, transient and step change. Various detectors from section 5.5 are implemented to detect the events. Finally, all the detected events are aligned with respect to time to study the characteristics of different parameters in power system that are measured by PMU in case of an event and whether they match with the event record file.

It is important to note here that the event record file is not used to train the models as done in some of the methods discussed in chapter 2, but to use it as a validation measure as the physical power flow calculations are not performed to verify the occurrence of an event.

6.1 Preprocessing

Prior to the application of detectors, the PMU data needs to preprocessed in order to achieve effective analysis of the parameters evaluated and to avoid false positives.

6.1.1 Data Filtration

Voltage magnitude recorded by the PMUs is usually adulterated with noise and unwanted information due to the fact that the voltage signal is transmitted through a series of cables, equipment and parallel loads. Furthermore, the frequency range of inter-area and intra-area oscillations is below 2 Hz [59]. In order to effectively detect these oscillation events and to remove the unwanted information from the data, data needs to be filtered for high frequency content. A low pass butterworth filter is used

for this purpose. Butterworth filter provides a flat response in the output signal at the expense of a relatively wide transition region from pass-band to stop-band, with average transient characteristics. The magnitude squared butterworth function of order n is given in equation 6.1 and the transfer function is given in equation 6.2.

$$|H_a(j\omega)|^2 = \frac{1}{1 + (\omega/\omega_c)^{2n}} \tag{6.1}$$

Where,

$\omega_c \rightarrow$ cutoff frequency

$$H_a(s) = (-1)^n \prod_{K=0}^{n-1} \frac{s_k}{s - s_k} \tag{6.2}$$

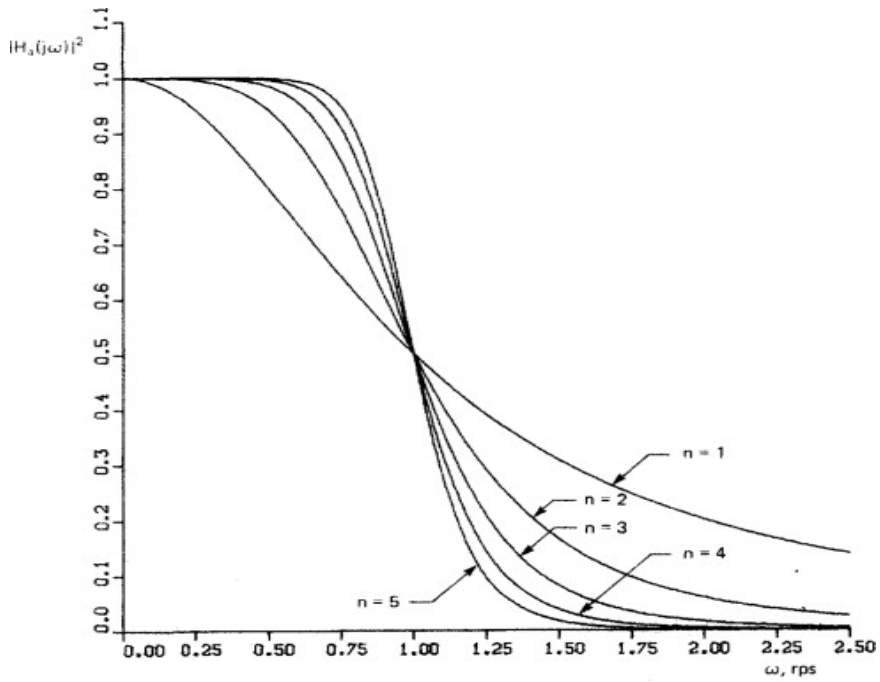


Figure 6.1: Magnitude Squared Characteristic of the Normalized Butterworth Low-pass Filter [60]

Module *butter* from python library *scipy* is used to filter out frequency content greater the 2 Hz. Parameters for this filter are listed in table 6.1.

Table 6.1: Parameters for Butterworth Filter

Parameter	Value
Order	2
Sampling rate	30 Hz
Cutoff frequency	2 Hz

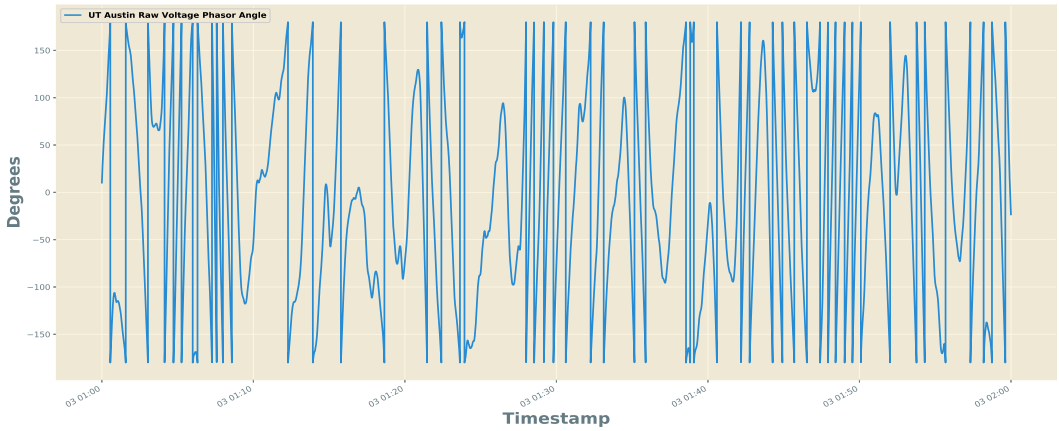
6.1.2 Angle Unwrap

The voltage phase angle recorded in the data increases or decreases depending on the speed of rotation as the rotation is not constant and not at normal frequency (60 Hz) [61]. The voltage phase angle jumps from 180 to -180 degrees when rotating at greater than normal speed as shown in figure 6.2a. In order to analyse the voltage phase angle, it is necessary to unwrap it - figure 6.2b. The unwrap process is explained in the following algorithm.

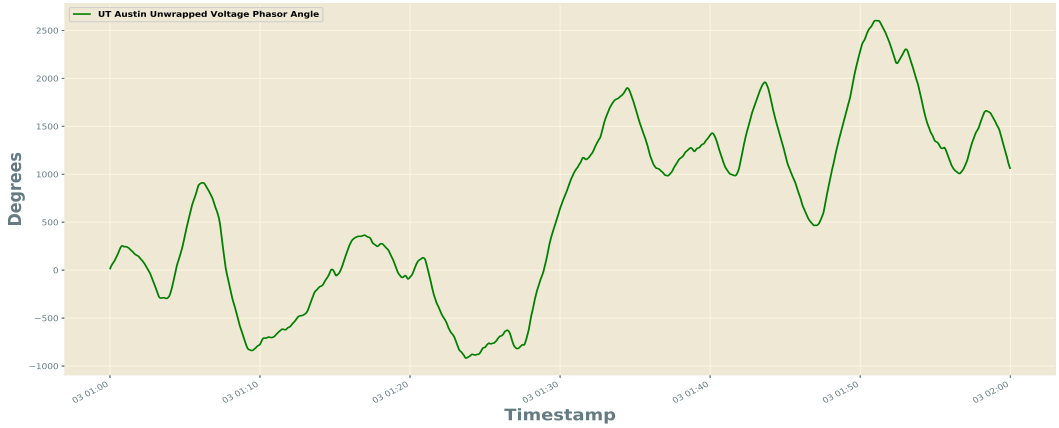
Phase Angle Unwrap

Inputs: A - Voltage phasor angle of a station
Output: Unwrapped Angle

```
angdiff = A(2:end) - A(1:end-1)
posjump = find(angdiff < -300)
negjump = find(angdiff > 300)
if isempty(posjump)
    for qt = 1 : length(posjump)
        x = [A(1 : posjump(qt)); A(posjump(qt) + 1 : end) + 360]
        A = x
    end for
end if
if isempty(negjump)
    for qt = 1 : length(negjump)
        x = [A(1 : negjump(qt)); A(posjump(qt) + 1 : end) - 360]
        A = x
    end for
end if
```



(a) Raw Voltage Phasor Angle



(b) Unwrapped Voltage Phasor Angle

Figure 6.2: Voltage Phasor Angle Unwrap Process

6.2 Detection using SST

An open source python library *banpei* 5.4.1.3 has been used to detect transient events using SST in all the available measurements by PMU. The parameters selected for *banpei* are listed in table 6.2. The detection procedure is explained in appendix A.

Table 6.2: Selected Parameters for *banpei*

Parameter	Selected Value
w	600
m	Default 2
k	300
L	150
is_lancos	True

Various combinations of the parameters were tested and the best performing parameters were selected. The number of columns in trajectory matrix and test matrix k and the lag time L are calculated using equation 6.3 and 6.4 respectively. Parameter is_lancos is set to true for performing SST in Krylov subspace 5.4.1.3 resulting in faster calculation of change point score.

$$k = \frac{w}{2} \tag{6.3}$$

Where,

$w \rightarrow$ window size

$$L = \frac{k}{2} \tag{6.4}$$

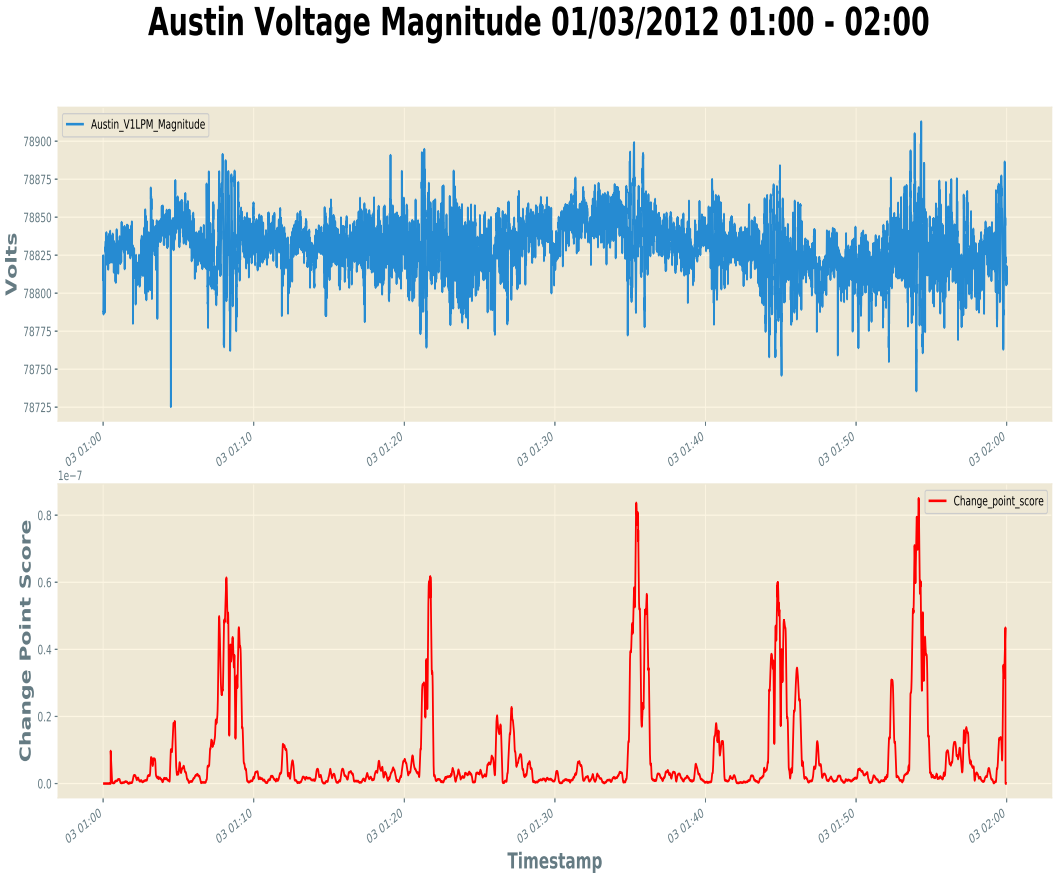
Where,

$k \rightarrow$ number of columns in trajectory matrix and test matrix

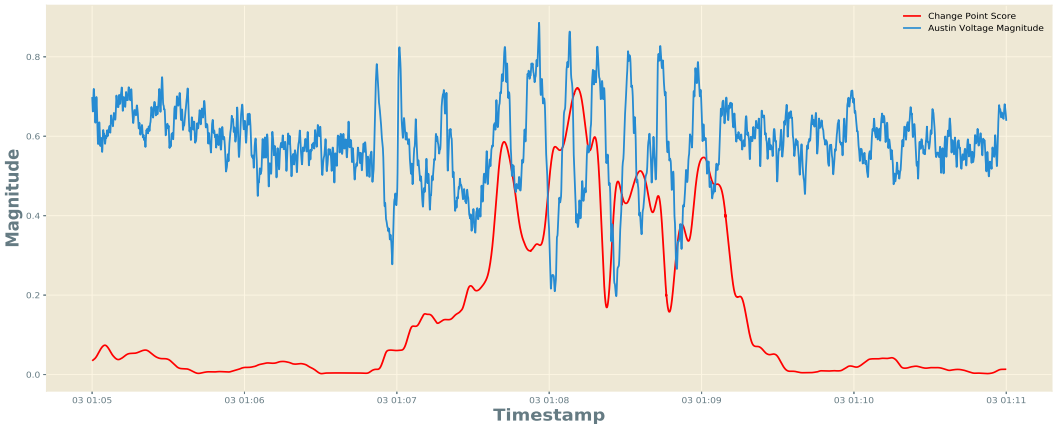
Prior to application of *banpei* the voltage magnitude and voltage angle needs to be preprocessed as per section 6.1. In case of frequency, no preprocessing is required. Once, the preprocessed data is obtained *banpei* can be directly implemented on the parameters in the data.

Figure 6.3 and figure 6.5 give the resulting plots for one of the station's voltage magnitude and frequency respectively. Figure 6.3a shows the voltage magnitude

change point score for entire hour. Whereas, figure 6.3b shows the normalized plot from minute 5 to minute 11 for the voltage magnitude and the respective change point score. In figure 6.4, the resulting plot for RPAD between two station is shown. Figure 6.4a presents the RPAD change point score for entire hour. Whereas, figure 6.4b presents the normalized plot from minute 43 to minute 45 for the RPAD and the respective change point score. Figure 6.5 shows the frequency change point score for an entire hour. Detected events from direct implementation of *banpei* on data, extracted features and from optimised set of features are listed in Table 6.3.

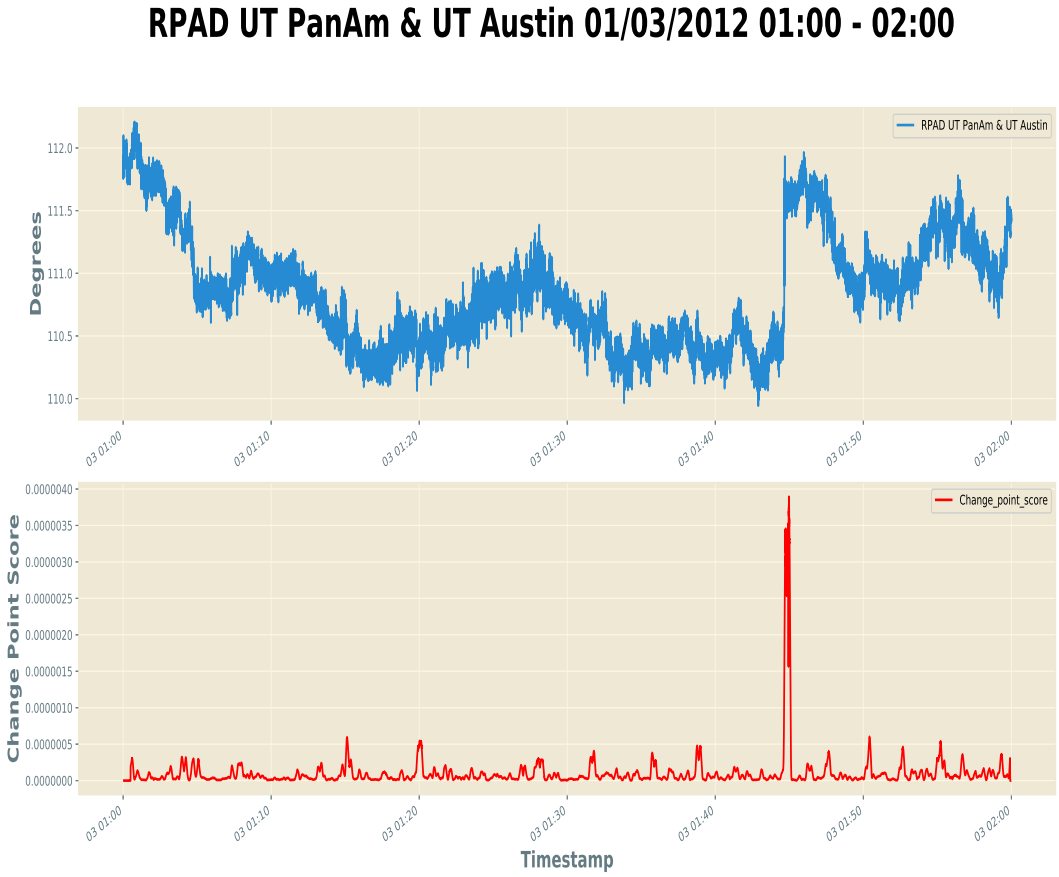


(a) Singular Spectral Transform for UT Austin Voltage Magnitude

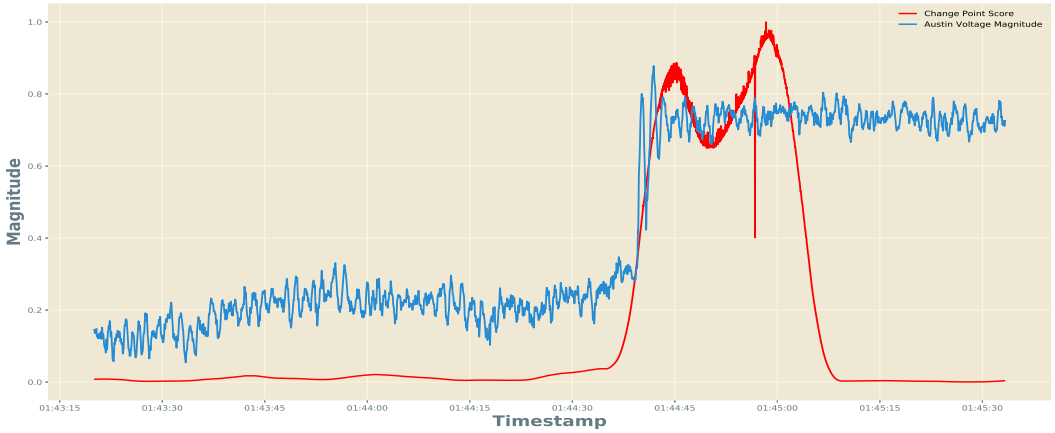


(b) Singular Spectral Transform for UT Austin Voltage Magnitude (Enlarged)

Figure 6.3: Singular Spectral Transform for UT Austin Voltage Magnitude



(a) Singular Spectral Transform for RPAD between UT Austin & UT PanAm



(b) Singular Spectral Transform for RPAD between UT Austin & UT PanAm (Enlarged)

Figure 6.4: Singular Spectral Transform for RPAD between UT Austin & UT PanAm

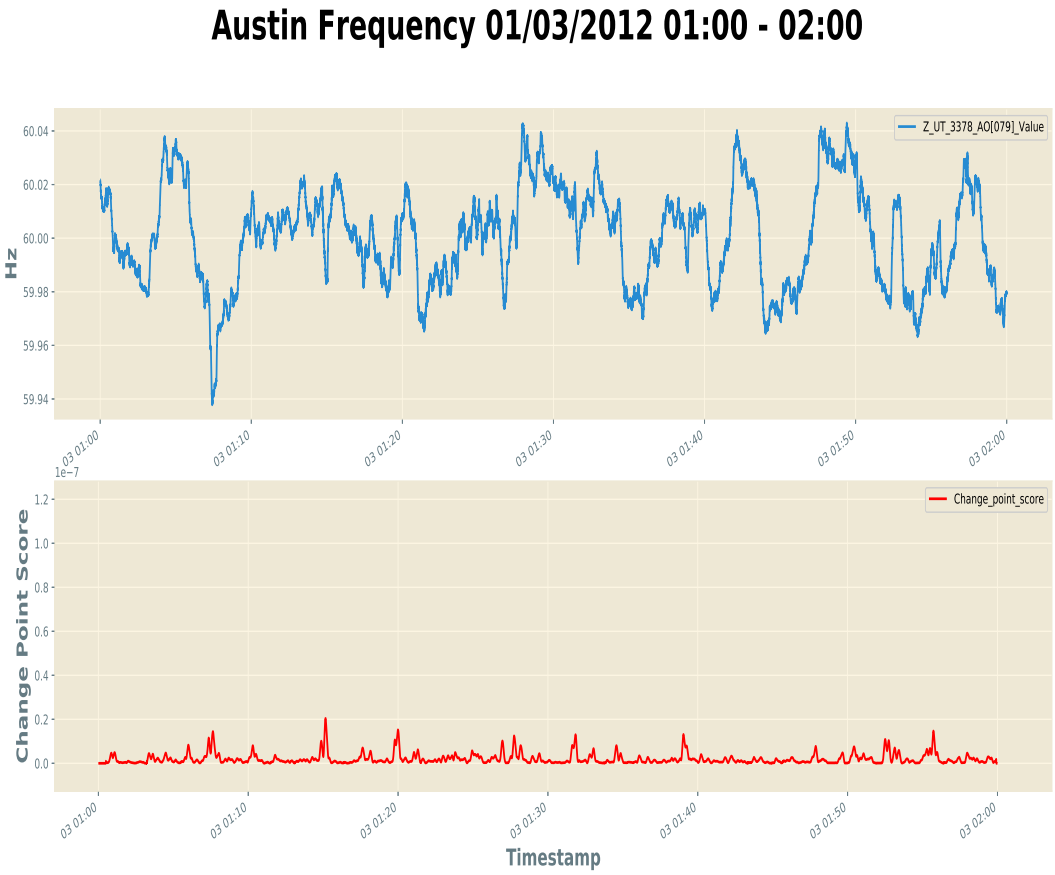


Figure 6.5: Singular Spectral Transform for UT Austin Voltage Magnitude

Table 6.3: Detected Events using SST

Parameter	Station	Number of Events Detected	Type of Event	Timestamp	Duration (seconds)
Voltage Magnitude	UT Austin	7	Transient	1:06:53	103
				1:21:00	57
				1:34:45	100
				1:44:09	87
				1:45:44	48
				1:52:05	35
				1:53:31	83
				1:59:41	14
	McDonald	0	-	-	-
	UT PanAm	0	-	-	-
RPAD	UT Austin & UT PanAm	1	Transient	1:44:36	29
	UT Austin & McDonald	0	-	-	-
Frequency	UT Austin	0	-	-	-
	McDonald	0	-	-	
	UT PanAm	0	-	-	-

The transients represent temporary imbalance of load and generation. Seven transient events were captured by *banpei* for UT Ausitn's voltage magnitude and one transient event was captured for UT Austin and UT PanAm RPAD. Table 6.3 gives the timestamp and the duration of detected events. It can be inferred that by using SST, transient events can be captured effectively without any false detection. It has been found that with larger window size, the computation time increases. However, better results are produced as the data converted to the new time series is of much higher resolution due to the formation of larger hankel's matrix (5.4.1). Therefore, a good balance of accuracy and the computational power utilization needs to be found according to the problem at hand.

6.3 Detection Using *PersistAD*

Prior to application of *PersistAD*, the data needs to be preprocessed as per section 6.1. In case of frequency, no preprocessing is required. Once, the preprocessed data is obtained *PersistAD* can be directly implemented on the parameters in the data. The parameters selected for *PersistAD* are listed in table 6.4. The detection procedure is explained in appendix B.

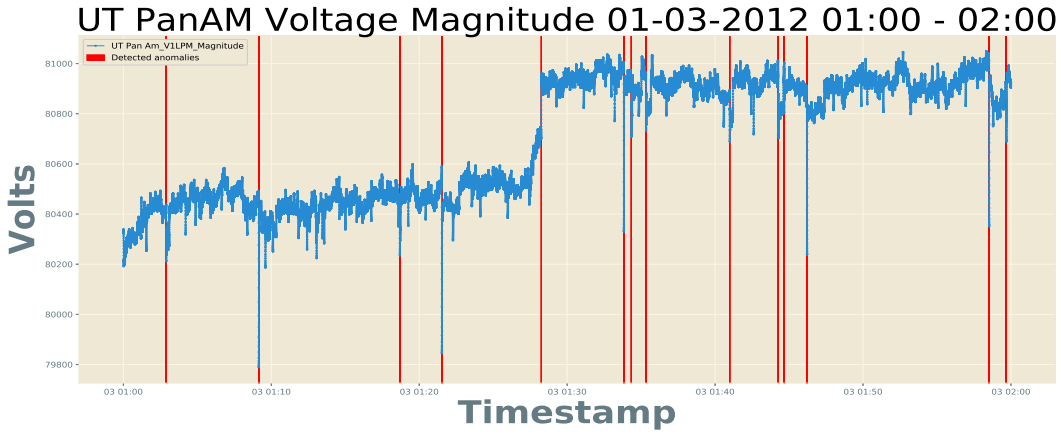
Table 6.4: Parameters for *PersistAD*

Parameter	Selected Value
window	Default: 1
c	15
side	Default: both
min periods	Default: None
agg	Default: median

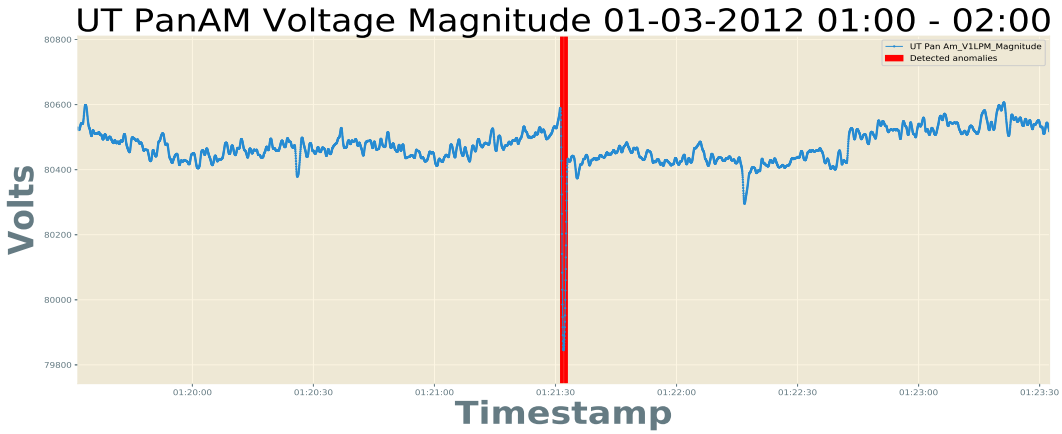
The window parameter has been set to one as the main goal of this detector is to detect impulse events in the PMU data. This allows detection of impulses with a very short time period. c is the normal bound defined with respect to the interquantile

range of the time series on which the detector is applied. *side* is set to both to detect impulses in both; positive and negative direction. Median is used as the aggregation method for the double rolling aggregate window.

The detector is applied on each of the parameter and the results are listed in table 6.5. Figure 6.6, figure 6.7 and figure 6.8 shows the detected anomalies in voltage magnitude, RPAD and frequency for one of the stations.

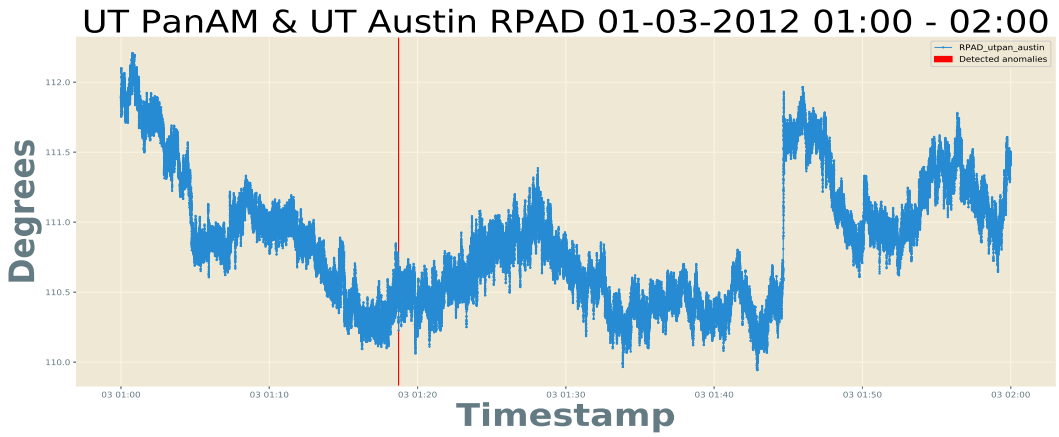


(a) Impulse Events in UT PanAm Voltage Magnitude

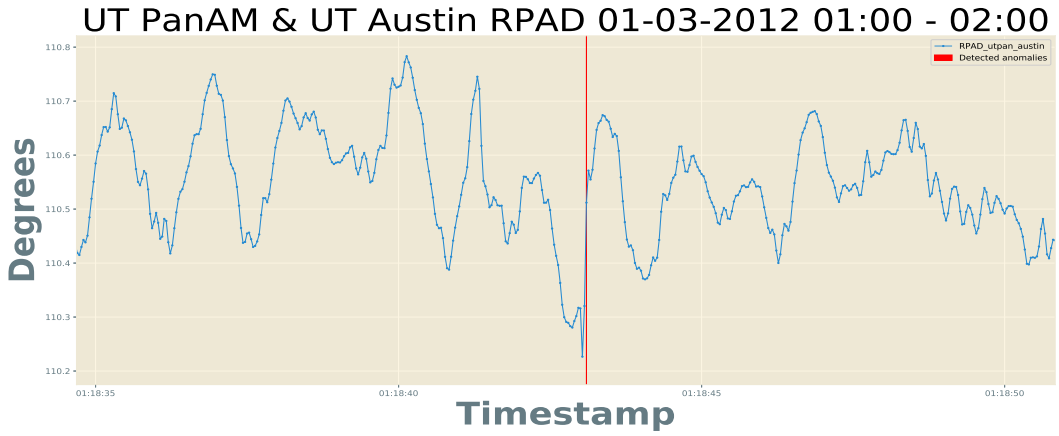


(b) Impulse Events in UT PanAm Voltage Magnitude (Enlarged)

Figure 6.6: Impulse Events in UT PanAm Voltage Magnitude

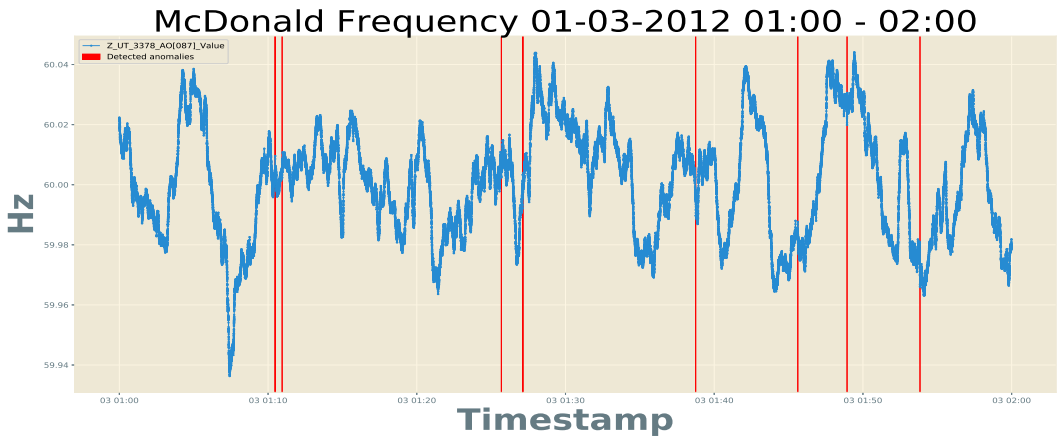


(a) Impulse Events in UT PanAm & UT Austin RPAD

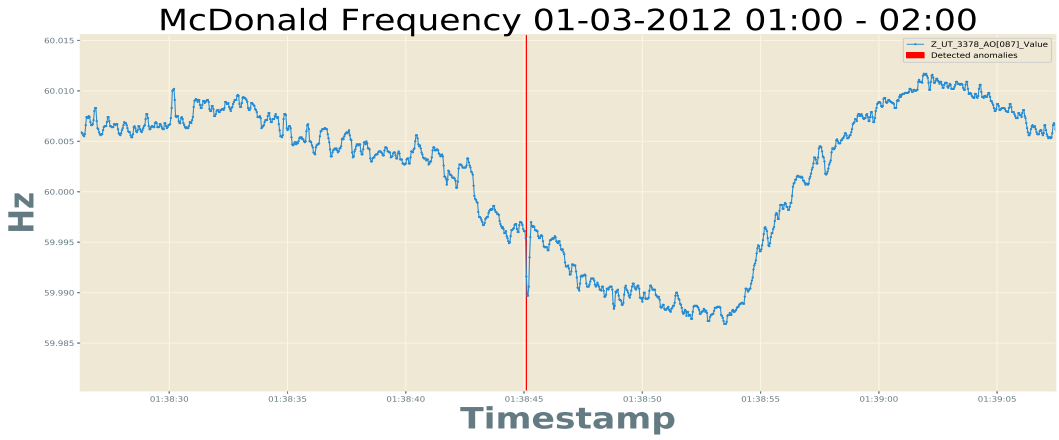


(b) Impulse Events in UT PanAm & UT Austin RPAD (Enlarged)

Figure 6.7: Impulse Events in UT Austin & UT PanAm RPAD



(a) Impulse Events in McDonald Frequency



(b) Impulse Events in McDonald Frequency (Enlarged)

Figure 6.8: Impulse Events in McDonald Frequency

The number of events, timestamp and the respective duration of the events are given in table 6.5. The number of impulses occurring on voltage magnitude has observed to be larger. Although the exact cause is unknown, transients might be the result of constant switching of loads which is reflected in the voltage magnitude. In contrast to the voltage magnitude, the RPAD has relatively low number of impulses. The speed of rotation of phasor angle varies according to the various operating conditions. Hence, the phasor angle are less susceptible to impulses and more susceptible to step changes. The frequency for station Mcdonald has eight impulse events and

the other two stations have no impulses present.

Table 6.5: Detected Events using PersistAD

Parameter	Station	Number of Events Detected	Type of Event	Timestamp	Duration (seconds)
Voltage Magnitude	UT Austin	5	Impulse	1:04:30	1
				1:14:46	1
				1:24:13	1
				1:48:46	1
				1:56:46	1
	McDonald	12	Impulse	1:07:08	1
				1:19:31	1
				1:20:42	1
				1:26:31	1
				1:27:41	1
				1:28:44	1
				1:30:04	1
				1:30:26	1
				1:38:45	2
				1:44:57	1
				1:53:46	1
				1:56:05	1
	UT PanAm	13	Impulse	1:02:53	1
				1:09:08	2
				1:18:41	1
				1:21:31	2
				1:28:14	1
				1:33:49	2
				1:35:19	1
				1:40:59	1
				1:44:16	1
				1:44:39	2
				1:46:12	2
				1:58:30	3
				1:59:40	1
RPAD	UT Austin & UT PanAm	1	Impulse	1:18:43	1
	UT Austin & McDonald	3	Impluse	1:38:45	1
				1:48:43	1
				1:53:49	1
Frequency	UT Austin	0	-	-	-
	McDonald	8	Impulse	1:10:28	1
				1:10:56	1
				1:25:41	1
				1:27:07	1
				1:38:45	1
				1:45:37	1
				1:48:55	1
				1:53:50	1
	UT PanAm	0	-	-	-

6.4 Detection Using *LevelShiftAD*

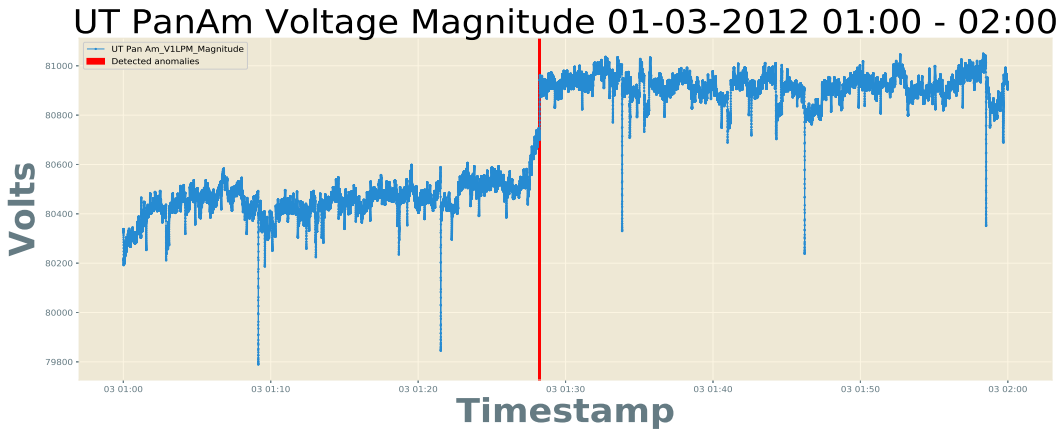
LevelShiftAD detector from an open source library ADTK 5.4.2 has been used to detect step changes in voltage magnitude and RPAD data. Sudden rise/drop events in frequency data has also been explored using this detector. The parameters used are listed in table 6.6. The detection procedure is explained in appendix B.

Table 6.6: Selected Parameters for *LevelShiftAD*

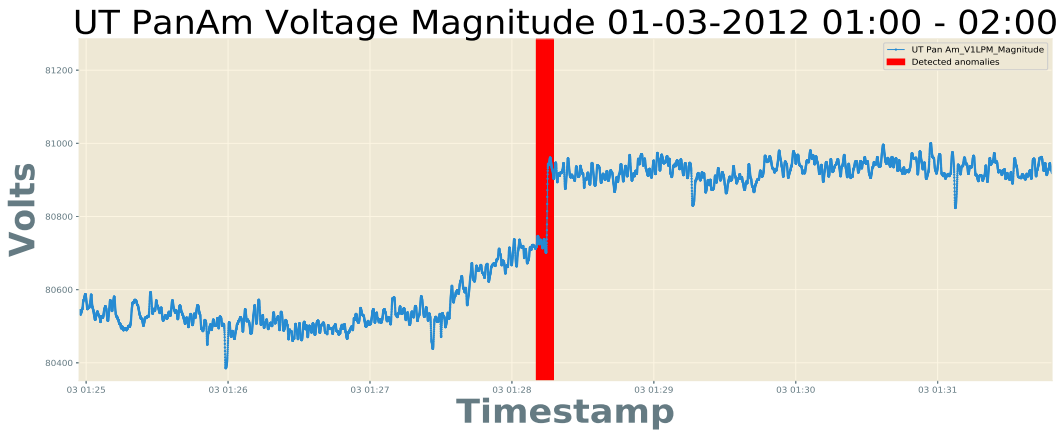
Parameter	Default Value
window	300
c	8
side	both
min periods	Default: none

The window parameter is set to 300. Unlike *PersistAD* the window for *LevelShiftAD* must be long enough to detect the step changes. c is the normal bound defined with respect to the interquantile range of the time series on which the detector is applied. *side* is set to *both* to detect impulses in both; positive and negative direction.

LevelShiftAD is applied on each parameter of the data after the preprocessing on voltage magnitude and phasor angle is performed. Figure 6.9, figure 6.10 shows the detected step changes in UT PanAm voltage magnitude and RPAD between UT PanAm and UT Austin, whereas Figure 6.11 presents the sudden rise/drop in frequency for UT Austin. The results are listed in table 6.7.

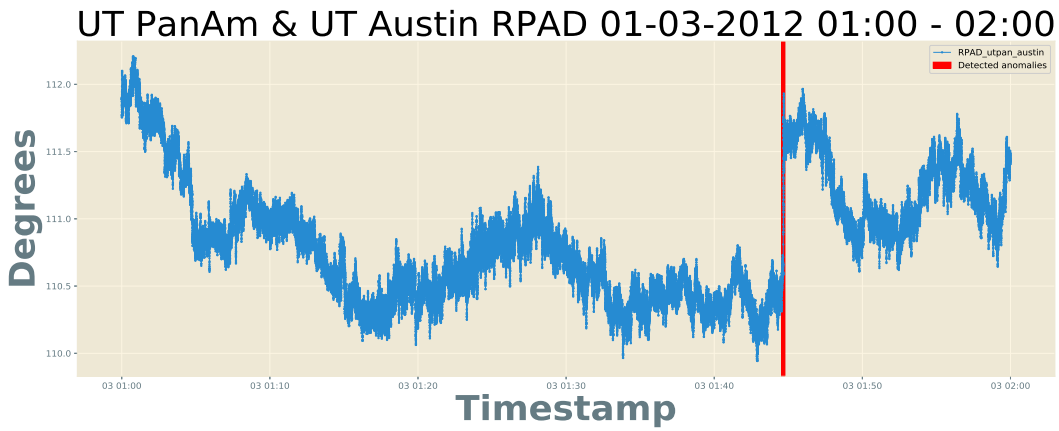


(a) Step Changes in UT PanAm Voltage Magnitude

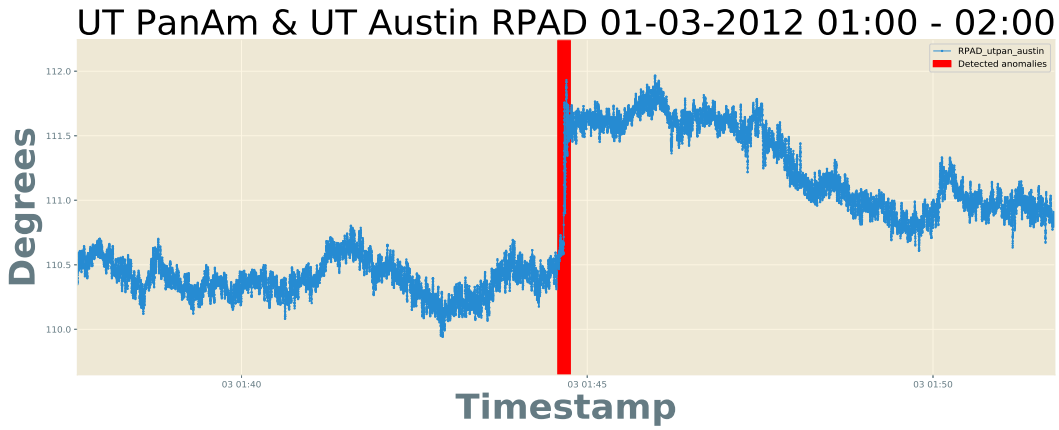


(b) Step Changes in UT PanAm Voltage Magnitude (Enlarged)

Figure 6.9: Step Changes in UT PanAm Voltage Magnitude

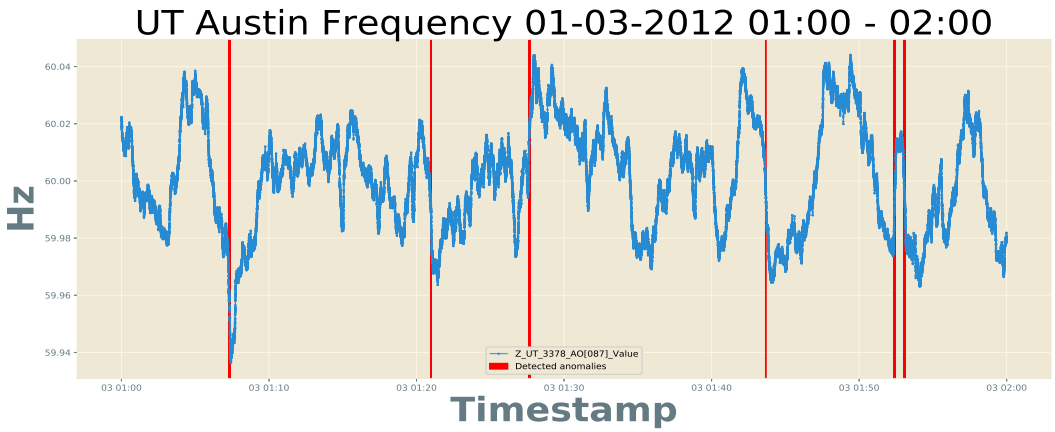


(a) Step Changes in UT PanAm & UT Austin RPAD

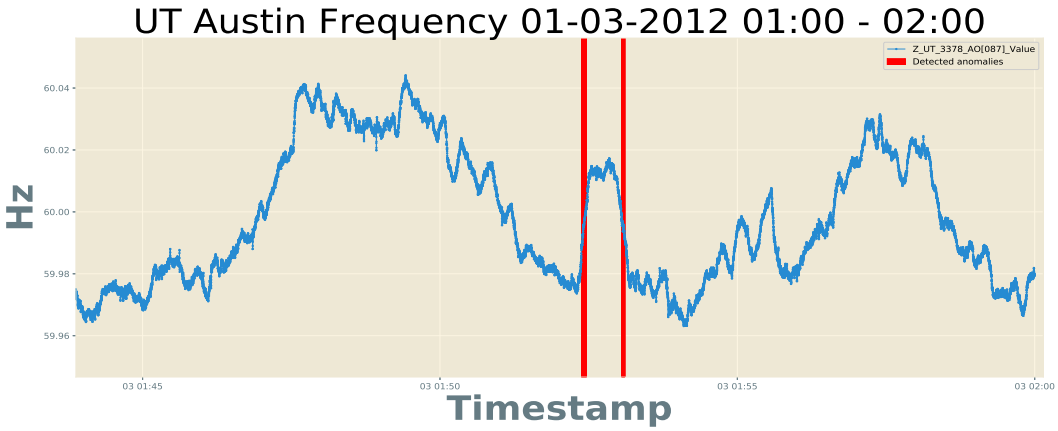


(b) Step Changes in UT PanAm & UT Austin RPAD (Enlarged)

Figure 6.10: Step Changes in UT PanAm & UT Austin RPAD



(a) Sudden Rise/Drop Events in UT Austin Frequency



(b) Sudden Rise/Drop Events in UT Austin Frequency (Enlarged)

Figure 6.11: Sudden Rise/Drop Events in UT Austin Frequency

Table 6.7: Detected Events using LevelShiftAD

Parameter	Station	Number of Events Detected	Type of Event	Timestamp	Duration (seconds)
Voltage Magnitude	UT Austin	0	-	-	-
	McDonald	0	-	-	-
	UT PanAm	1	Step Change	1:28:10	6
RPAD	UT Austin & UT PanAm	1	Step Change	1:44:34	1
	UT Austin & McDonald	0	-	-	-
Frequency	UT Austin	6	Sudden Rise/Drop	1:07:15	5
				1:20:57	1
				1:27:38	2
				1:43:39	1
				1:52:22	5
				1:53:03	3
	McDonald	6	Sudden Rise/Drop	1:07:15	5
				1:20:57	1
				1:27:38	2
				1:43:39	1
				1:52:22	5
				1:53:03	3
	UT PanAm	6	Sudden Rise/Drop	1:07:15	5
				1:20:57	1
				1:27:38	2
				1:43:39	1
				1:52:22	5
				1:53:03	3

The table 6.7 gives the number of step changes detected, their timestamp and the duration. A step change was detected in UT PanAm voltage magnitude. However, this was the only event detected by *LevelShiftAD* in voltage magnitude data. One step change was detected in UT Austin and UT PanAm RPAD data. In case of frequency, *LevelshiftAD* detected six sudden rise/drops at the same timestamps for all the stations. Hence, it can be inferred that the entire system was affected by these events.

6.5 Detection Using Isolation Forest

Isolation forest has been used to detect events in the PMU data. This detector is not designed to detect a specific type of event. Rather, it detects anomalies based on the the susceptibility of an outlier to be isolated. Hence, the detected events from isolation forest would be categorized based on the visual characteristics. The detection procedure is explained in appendix C.

Prior to application of detector on angle data, angle needs to be unwrapped as per section 6.1.2. The RPAD 5.1 of two stations with respect to UT Austin has been derived after the unwrap process and the resulting time series is used for detection. In case of voltage magnitude, the data needs to filtered to remove high frequency noise in order to achieve robust detection. The parameters selected for the isolation forest algorithm are listed in table 6.8.

Table 6.8: Selected Parameters for Isolation Forest

Parameter	Selected Value
n_estimators	Auto: 100
max_samples	Default: auto
contamination	0.001111

It was observed that due to the length of the data, high computational memory was utilized by this detector. In order to reduce the computational memory usage, a sliding window has been implemented. The length of the sliding window was set to 30 data points in one window. This length was set in such a way that each window represents one second so that if an anomaly is detected in any of the window, it can be conveniently mapped to the respective timestamp.

The parameter n_estimators and max_samples were set to default. The only parameter that has been varied in this study is contamination. As isolation forest is an unsupervised anomaly detection algorithm, the proportion of outliers in the data set

(number of outliers to be detected) must be predefined. Since, the model was implemented on each window the contamination is the proportion of anomalous windows to the total number of windows. Equation 6.5 gives formulation of contamination parameter c. The contamination parameter changes with type of measurement; voltage magnitude, RPAD and frequency and has to obtained on with a well educated guess depending upon the number of anomalies to be detected.

$$c = \frac{number\ \ of\ \ anomalous\ \ windows}{total\ \ number\ \ of\ \ windows} \tag{6.5}$$

Various features have been extracted from each window using *tsfresh* 5.5.2.1. These features have been used as an input to the isolation forest algorithm. The resulting plot has been displayed in figure 6.12 with contamination set to *auto*.

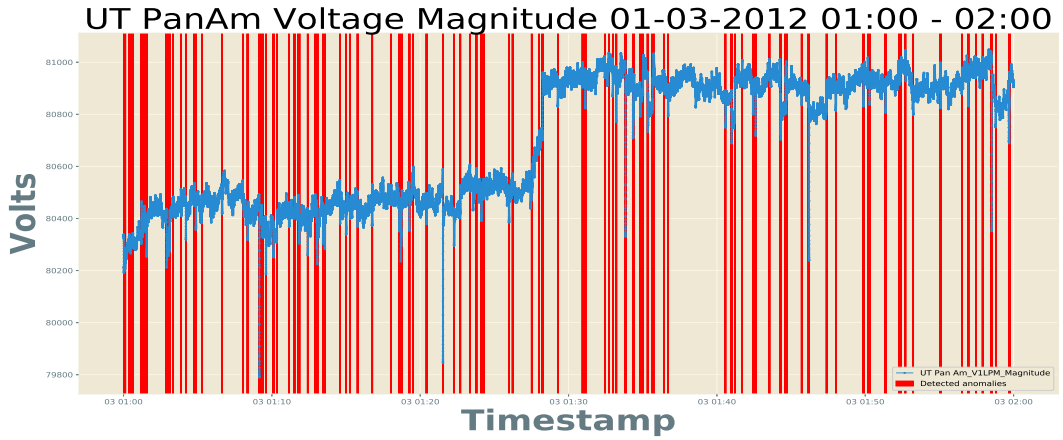


Figure 6.12: *i*Forest Implementation on UT PanAm Voltage Magnitude cāuto

Based on visual analysis of the figure 6.12, it can be inferred that with contamination set to auto, the algorithm classifies the normal data points as anomalies. With such high resolution and varying data it is difficult for the algorithm to isolate anomalies with contamination set to auto. The basis of algorithm states that the anomalies occur rarely and hence, the contamination parameter needs to be set appropriately. Figure 6.13 shows the resulting plot for UT PanAm & UT Austin RPAD

with contamination set to 0.001.

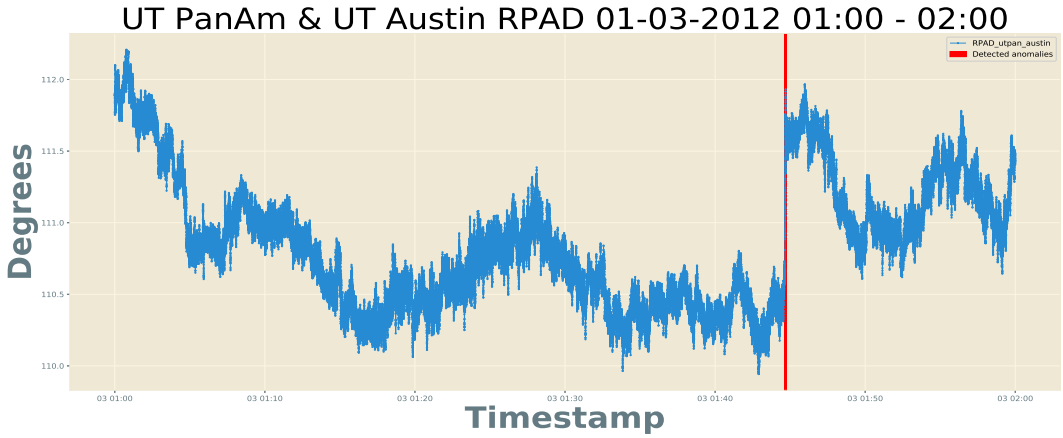


Figure 6.13: *i*Forest Implementation on UT PanAm & UT Austin RPAD $c\bar{0}.001$

In order to improve the performance of isolation forest algorithm, the extracted features from *tsfresh* can be optimized. PSO 5.5.2.2 has been implemented for this purpose. The parameters selected for PSO are listed in table 6.9. Various parameters were tested and the best performing parameters were selected. The optimization procedure is explained in appendix D.

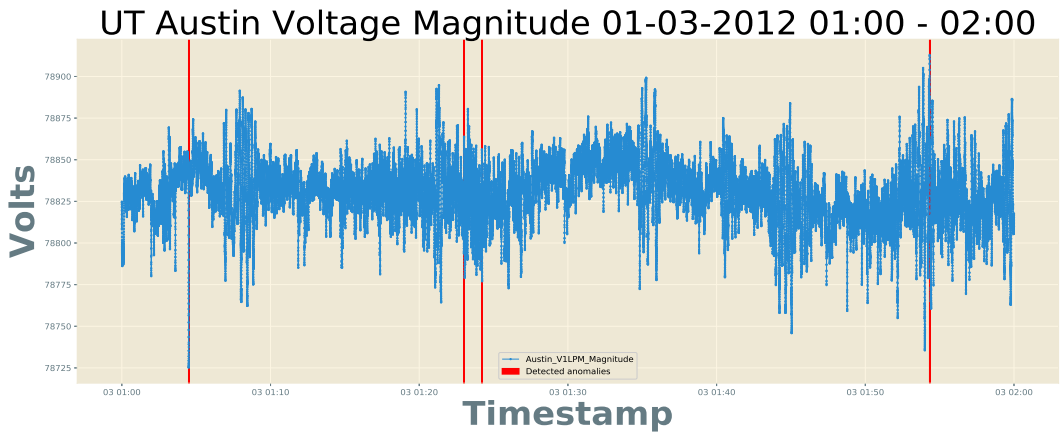
Table 6.9: Selected Parameters for Particle Swarm Optimization

Parameter	Selected Value
n_particles	30
dimensions	Number of features extracted by *tsfresh*
n_processes	30
iterations	200

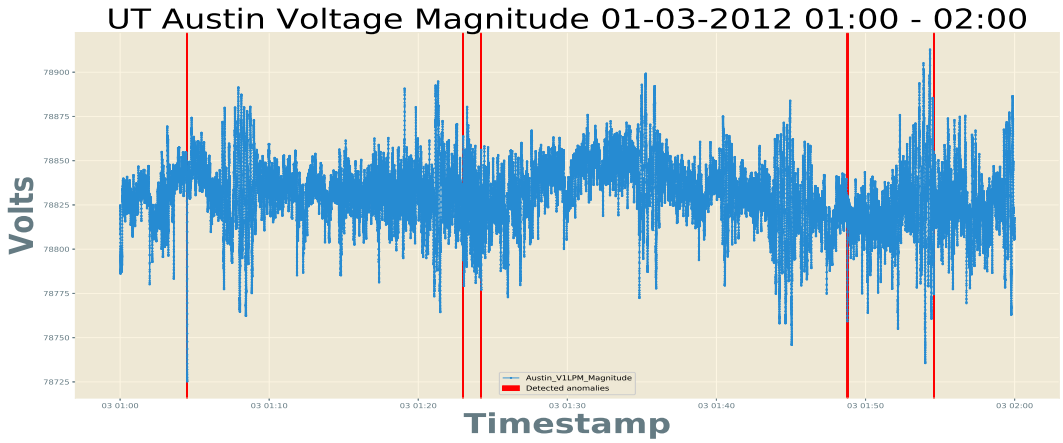
PSO needs error as an input. PSO optimises the position of features in such a way that this error is reduced. The error is formulated in equation 6.6.

$$error = \frac{actual\ \ anomalies - predicted\ \ anomalies}{actual\ \ anomalies} \tag{6.6}$$

It is necessary to have labeled data to get the actual anomalies. With error record file having just one event recorded and no other information, events detected with *PersistAD* and *LevelShiftAD* could be considered as actual anomalies. It is important to note here that the labeled data is not used to train the isolation forest model but to get a feature set that improves the detection. Figure 6.14a and figure 6.14b shows the detected events with non optimized and post optimized feature set respectively. The detected impulse events from *PersistAD* for UT Austin voltage magnitude are used as actual events.



(a) *i*Forest on UT Austin Voltage Magnitude before Optimization



(b) *i*Forest on UT Austin Voltage Magnitude after Optimization

Figure 6.14: PSO for Isolation Forest

Table 6.10 gives the error calculated using equation 6.6 and number of features for UT Austin voltage magnitude for non optimized and optimized model. Figure 6.15 shows the reduction in error with respect to the number of particle swarm iterations.

Table 6.10: PSO for *i*Forest

Isolation Forest	Number of Features	Error (%)
Non Optimized Feature	349	0.5
Optimized Features	178	0

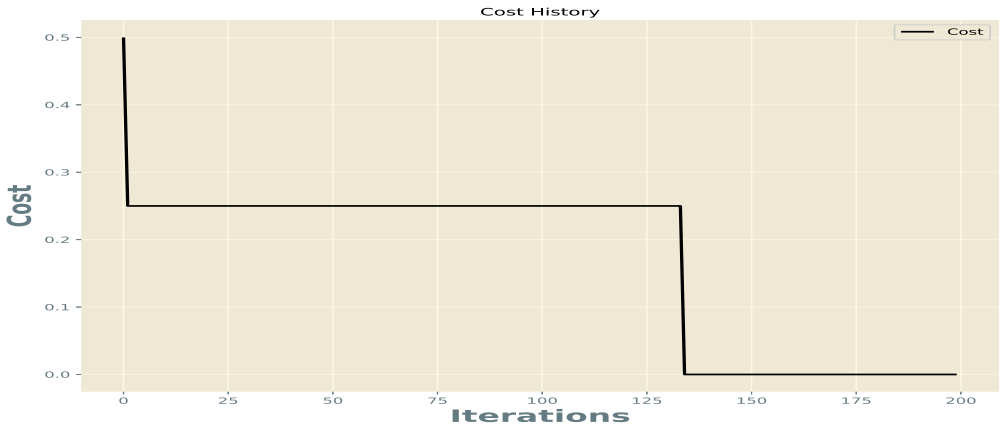


Figure 6.15: Reduction in Error with respect to the Number of Iterations

Table 6.11 shows the events detected by isolation forest. Since the actual anomalies from *PersistAD* and *LevelShiftAD* must be mapped manually, the optimization for all the parameters is not performed in this study. However, if the data is readily labeled the PSO can be be implemented for better performance as shown in table 6.10.

Table 6.11 shows number of events detected, the type of events, timestamp and the duration of the event for each parameter. It was found that some of the detected events were similar to those detected by *PersistAD* and *LevelShiftAD*. However, not all the detected events match. Isolation forest can detect both step changes as well as impulses in the PMU data. Having said that, it was learned that detected events were overly sensitive to contamination parameter. Although isolation forest is an unsupervised algorithm, for an optimum performance and to have better contamination value, data labels are required.

Table 6.11: Detected Events using Isolation Forest

Parameter	Station	Number of Events Detected	Type of Event	Timestamp	Duration (seconds)
Voltage Magnitude	UT Austin	4	Impulse	1:04:29	2
				1:23:00	1
				1:24:13	1
				1:54:19	1
	McDonald	11	Impulse	1:00:18	1
				1:08:48	1
				1:19:31	1
				1:20:42	1
				1:26:31	3
				1:28:44	1
				1:30:26	1
				1:36:14	1
				1:44:57	1
				1:45:13	1
				1:53:49	1
				1:55:41	1
	UT PanAm	9	Impulse/ Step Change	1:09:09	2
				1:21:31	1
				1:28:44	1
				1:33:49	2
				1:34:20	1
				1:44:40	1
				1:46:12	2
				1:58:31	2
				1:59:40	1
RPAD	UT Austin & UT PanAm	2	Impulse/ Step Change	1:18:43	1
				1:44:39	4
	UT Austin & McDonald	3	Impulse	1:00:18	1
				1:06:39	1
				1:38:45	1
Frequency	UT Austin	5	Sudden Rise/Drop	1:05:52	1
				1:07:20	2
				1:07:42	1
				1:15:03	1
				1:47:32	1
	McDonald	4	Sudden Rise/Drop	1:07:20	2
				1:07:25	1
				1:07:41	2
				1:27:36	1
	UT PanAm	6	Sudden Rise/Drop	1:07:22	1
				1:07:42	1
				1:28:23	1
				1:44:40	1
				1:47:32	1
				1:50:06	1

6.6 Comparison of Detected Events with Event Record File

The event record file is made available by author of the PMU data used in this study [36]. This file is a log of detected events from the detection application developed by UT Austin to monitor the Texas synchrophasor network [36]. Table 6.12 gives the event information in the record file.

Table 6.12: Events in Record File

Index	Measured Parameter	Number of Events	Type of Event	Timestamp
1	Voltage RPAD	1	Step Change	01:44
2	Frequency	0	-	-

It can be observed from table 6.12 that a step change is recorded for UT Austin and UT PanAm at 01:44. However, this was the only event detected in an entire hour by the application. Furthermore, only two measurements; RPAD and frequency were examined. This study however all three measurements available in the data are considered. Table 6.13 shows the detected events in this study for this time stamp.

Table 6.13: Detected Events at 01:44 by Various Detector

Detector	Parameter	Station	Type of Event	Timestamp	Duration (seconds)
SST banpei	Voltage Magnitude	UT Austin	Transient	01:44:09	87
	RPAD	UT Austin & UT PanAM	Transient	01:44:36	29
PersistAD	Voltage Magnitude	UT Pan Am	Implse	1:44:16	1
		UT PanAm	Impulse	1:44:39	2
LevelShiftAD	RPAD	UT Austin & UT PanAm	Step Change	1:44:34	1
Isolation Forest	Voltage Magnitude	UT PanAm	Impulse	1:44:40	1
	RPAD	UT Austin & UT PanAm	Step Change	1:44:39	4
	Frequency	UT PanAM	Rise/Drop	1:44:40	1

From table 6.13 it can be inferred that the methods used in this study were able to detect event in various parameters at the same timestamp as in the record file. The validity of all the other detected events could not be tested due to lack of information. However, based on visual analysis, the anomalous changes in the data are well captured by the detectors.

CHAPTER 7: CONCLUSIONS

With the advent of grid modernization, significant amount of grid data is generated. Sensors with high resolution measurement capabilities are replacing traditional relays and measuring systems. One such sensor is PMU which generates precise high rate measurement of the grid parameters. PMUs play a significant role in the observability of the electric grid. However, with high rate measurements, analyzing this data is seen to be a great challenge. Existing methods on data analysis and anomaly detection were explored in this study to detect event occurrences on the power grid. Furthermore, due to the lack of grid topology information and requirement of previous history of events to train the models, unsupervised learning techniques such as SST and isolation forest were used to detect the events. Unsupervised anomaly detection tool such as Arundo's ADTK was used to explore its performance on PMU data.

Historical PMU data from Texas synchrophasor network has been used to detect events. Categories based on visual characteristics have been proposed in section 5.2. The detection methods were selected based on their capability to detect the events from the proposed categories.

An open source pyhton library *banpei* was used to perform SST on time series to detect the transients. The change point score was calculated in Krylov subspace for each of the parameter. Calculating the change point score in krylov subspace significantly improves the computation time compared to the original SST method [46]. Timestamps with significantly high change point scores were marked as events. It was observed that transients were detected effectively with large window size. However, with increase in window size the computation time increases.

PersistAD and *LevelShiftAD* from Arundo's ADTK were used to detect impulse

and step change events in the data. Both the detectors use a double rolling aggregate sliding window to detect the anomalous changes. Due to continuous switching of the load impulse events are more visible in voltage magnitude rather their phasor angle and power frequency [62]. Similarly the speed of rotation of phasors changes significantly during an event resulting in step change. The impulse events detected by *PersistAD* and step change events detected by *LevelShiftAD* portrayed similar characteristics. Six sudden rise/drop events were detected in frequency measurement for all stations. The events occurred at exact same timestamps indicating a system wide event.

Based on visual analysis, SST for transients, *PersistAD* for impulses and *LevelShiftAD* for step changes as well as sudden drops or rise in frequency provided 100% detection accuracy.

Further, use of a generalized anomaly detection method, isolation forest was explored on PMU data. In contrast to the above mentioned methods, isolation forest was not used to detect any specific type of anomalies as isolation forest detects anomalies based on their susceptibility to be isolated. The anomalies detected by isolation forest were categorized on their visual characteristics after detection. It was observed that this algorithm required high computational memory due to the large length of the data. Hence, a sliding window was implemented to conveniently manage the data. Features from each window were extracted using *tsfresh* and the isolation forest was implemented on each window. Although isolation forest did not require labelled data to detect events, it was observed that labels were required in order to set an optimum contamination factor. Isolation forest was unable to detect transients in the data. However, impulses and step change were effectively detected. Further, to optimize the the features extracted by *tsfresh*, a population based metaheuristc method; PSO was implemented. PSO selects a position of features such that the error in predicted anomalies is reduced. However, for calculation of the error, the data must be labelled.

As not labels were not available, detected anomalies from *PersistAD* were used an actual anomalies. It was learned that PSO have the capability to significantly improve the detection accuracy for isolation forest.

Finally, the results obtained from the detectors were compared to the event detection methods used by UT Austin. It was observed that in contrast to the only event detected in UT Austin's record file, two of the proposed method were able to detect the same event as well as several other events. Additionally, the events in voltage magnitude were detected which is not considered in UT Austin's methods.

7.1 Future Work

In contrast to this study where the events are validated based on visual analysis, the proposed methodology can be tested on data with actual event information to examine the accuracy on quantitative grounds. In presence of labelled data, improvement in detection accuracy for isolation forest using PSO can be studied in much more detail. Possibility of an ensemble algorithm with automated parameter selection and event categorization built on the present methodology can be analysed. With continuous improvements in machine learning and deep learning methods, more optimized and advanced unsupervised machine learning and deep learning algorithms can be explored in the future. Further, application of methods used in this study can be tested on various other time series data sets. Isolation forest with particle swarm optimization can be tested on labeled data to study the improvement in detection.

REFERENCES

[1] A. G. Phadke and B. Tianshu, "Phasor measurement units, wams, and their applications in protection and control of power systems," *Journal of Modern Power Systems and Clean Energy*, vol. 6, no. 4, pp. 619–629, 2018.

[2] B. Amidan, J. Follum, K. Freeman, and J. Dagle, "Baselining pmu data to find patterns and anomalies," in *CIGRE 2015 Grid of the Future Symposium*, 2015.

[3] U. E. I. Administration, "Annual energy outlook 2020 - electricity." https://www.eia.gov/outlooks/aeo/, 2020.

[4] Siemens, "Phasor measurement unit." https://new.siemens.com/global/en/products/energy/energy-automation-and-smart-grid/protection-relays-and-control/general-protection/phasor-measurement-unit-pmu.html, 2020.

[5] A. Silverstein, "High-resolution, time-synchronized grid monitoring devices." https://www.naspi.org/sites/default/files/reference_documents/pnnl_29770_naspi_hires_synch_grid_devices_20200320.pdf, 2020.

[6] J. E. Tate and T. J. Overbye, "Line outage detection using phasor angle measurements," *IEEE Transactions on Power Systems*, vol. 23, no. 4, pp. 1644–1652, 2008.

[7] I. Power and E. Society, "Ieee pes amps dsas test feeder working group." https://site.ieee.org/pes-testfeeders/resources/, 2014.

[8] S. Kantra, H. A. Abdelsalam, and E. B. Makram, "Application of pmu to detect high impedance fault using statistical analysis," in *2016 IEEE Power and Energy Society General Meeting (PESGM)*, pp. 1–5, IEEE, 2016.

[9] F. Han, G. Taylor, and M. Li, "Towards a data driven robust event detection technique for smart grids," in *2018 IEEE Power & Energy Society General Meeting (PESGM)*, pp. 1–5, IEEE, 2018.

[10] Y. Cao, L. Cai, C. Qiu, J. Gu, X. He, Q. Ai, and Z. Jin, "A random matrix theoretical approach to early event detection using experimental data," *arXiv preprint arXiv:1503.08445*, 2015.

[11] S. Brahma, R. Kavasseri, H. Cao, N. R. Chaudhuri, T. Alexopoulos, and Y. Cui, "Real-time identification of dynamic events in power systems using pmu data, and potential applications - models, promises, and challenges," *IEEE transactions on Power Delivery*, vol. 32, no. 1, pp. 294–301, 2016.

[12] S. S. Negi, N. Kishor, K. Uhlen, and R. Negi, "Event detection and its signal characterization in pmu data stream," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 6, pp. 3108–3118, 2017.

[13] D.-I. Kim, T. Y. Chun, S.-H. Yoon, G. Lee, and Y.-J. Shin, "Wavelet-based event detection method using pmu data," *IEEE Transactions on Smart grid*, vol. 8, no. 3, pp. 1154–1162, 2015.

[14] Z. Lin, T. Xia, Y. Ye, Y. Zhang, L. Chen, Y. Liu, K. Tomsovic, T. Bilke, and F. Wen, "Application of wide area measurement systems to islanding detection of bulk power systems," *IEEE Transactions on Power Systems*, vol. 28, no. 2, pp. 2006–2015, 2013.

[15] D. M. Laverty, R. J. Best, and D. J. Morrow, "Loss-of-mains protection system by application of phasor measurement unit technology with experimentally assessed threshold settings," *IET Generation, Transmission & Distribution*, vol. 9, no. 2, pp. 146–153, 2015.

[16] Y. Ge, A. J. Flueck, D.-K. Kim, J.-B. Ahn, J.-D. Lee, and D.-Y. Kwon, "Power system real-time event detection and associated data archival reduction based on synchrophasors," *IEEE Transactions on Smart Grid*, vol. 6, no. 4, pp. 2088–2097, 2015.

[17] T. Xu and T. Overbye, "Real-time event detection and feature extraction using pmu measurement data," in *2015 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pp. 265–270, IEEE, 2015.

[18] M. Rafferty, X. Liu, D. M. Laverty, and S. McLoone, "Real-time multiple event detection and classification using moving window pca," *IEEE Transactions on Smart Grid*, vol. 7, no. 5, pp. 2537–2548, 2016.

[19] M. Biswal, S. M. Brahma, and H. Cao, "Supervisory protection and automated event diagnosis using pmu data," *IEEE Transactions on power delivery*, vol. 31, no. 4, pp. 1855–1863, 2016.

[20] Q. Gao and S. M. Rovnyak, "Decision trees using synchronized phasor measurements for wide-area response-based control," *IEEE Transactions on Power Systems*, vol. 26, no. 2, pp. 855–861, 2010.

[21] R. Dubey, S. R. Samantaray, B. K. Panigrahi, and V. G. Venkoparao, "Data-mining model based adaptive protection scheme to enhance distance relay performance during power swing," *International Journal of Electrical Power & Energy Systems*, vol. 81, pp. 361–370, 2016.

[22] J. Berrios, S. Wallace, X. Zhao, E. Cotilla-Sanchez, and R. B. Bass, "Generator event detection from synchrophasor data using a two-step time-series machine learning algorithm," in *2018 Ninth International Green and Sustainable Computing Conference (IGSC)*, pp. 1–7, IEEE, 2018.

[23] D. Nguyen, R. Barella, S. A. Wallace, X. Zhao, and X. Liang, "Smart grid line event classification using supervised learning over pmu data streams," in *2015 Sixth International Green and Sustainable Computing Conference (IGSC)*, pp. 1–8, IEEE, 2015.

[24] P. Trachian, "Machine learning and windowed subsecond event detection on pmu data via hadoop and the openpdc," in *IEEE PES General Meeting*, pp. 1–5, IEEE, 2010.

[25] S. Rusitschka, K. Eger, and C. Gerdes, "Smart grid data cloud: A model for utilizing cloud computing in the smart grid domain," in *2010 First IEEE International Conference on Smart Grid Communications*, pp. 483–488, IEEE, 2010.

[26] M. Khan, P. M. Ashton, M. Li, G. A. Taylor, I. Pisica, and J. Liu, "Parallel detrended fluctuation analysis for fast event detection on massive pmu data," *IEEE Transactions on Smart Grid*, vol. 6, no. 1, pp. 360–368, 2014.

[27] D. Dasgupta and S. Forrest, "Novelty detection in time series data using ideas from immunology," in *Proceedings of the international conference on intelligent systems*, pp. 82–87, 1996.

[28] P. Malhotra, L. Vig, G. Shroff, and P. Agarwal, "Long short term memory networks for anomaly detection in time series," in *Proceedings*, p. 89, Presses universitaires de Louvain, 2015.

[29] C.-p. Lee and S. J. Wright, "Using neural networks to detect line outages from pmu data," *arXiv preprint arXiv:1710.05916*, 2017.

[30] H. Ren, Z. Hou, and P. Etingov, "Online anomaly detection using machine learning and hpc for power system synchrophasor measurements," in *2018 IEEE International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*, pp. 1–5, IEEE, 2018.

[31] O. P. Dahal, S. M. Brahma, and H. Cao, "Comprehensive clustering of disturbance events recorded by phasor measurement units," *IEEE Transactions on Power Delivery*, vol. 29, no. 3, pp. 1390–1397, 2013.

[32] E. Klinginsmith, R. Barella, X. Zhao, and S. Wallace, "Characterizing smart grid events using clustering methods," in *Smart Cities, Green Technologies, and Intelligent Transport Systems*, pp. 75–96, Springer, 2016.

[33] Y. Tang and J. Yang, "Dynamic event monitoring using unsupervised feature learning towards smart grid big data," in *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 1480–1487, IEEE, 2017.

[34] M. Zhou, Y. Wang, A. K. Srivastava, Y. Wu, and P. Banerjee, "Ensemble-based algorithm for synchrophasor data anomaly detection," *IEEE Transactions on Smart Grid*, vol. 10, no. 3, pp. 2979–2988, 2018.

[35] X. Liang, S. A. Wallace, and D. Nguyen, "Rule-based data-driven analytics for wide-area fault detection using synchrophasor data," *IEEE Transactions on Industry Applications*, vol. 53, no. 3, pp. 1789–1798, 2016.

[36] M. Grady, "The texas synchrophasor network." https://users.ece.utexas.edu/~grady/Texas_Synchrophasor_Network.html, 2012.

[37] A. Allen, S. Santoso, and E. Muljadi, "Algorithm for screening phasor measurement unit data for power system events and categories and common characteristics for events seen in phasor measurement unit relative phase-angle differences and frequency signals," tech. rep., National Renewable Energy Lab.(NREL), Golden, CO (United States), 2013.

[38] P. Kundur, N. J. Balu, and M. G. Lauby, *Power system stability and control*, vol. 7. McGraw-hill New York, 1994.

[39] H. Tsuruta, "banpei." https://github.com/tsurubee/banpei, 2017.

[40] Arundo, "adtk." https://github.com/arundo/adtk, 2019.

[41] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[42] N. J. K.-L. A. Christ M, Braun N, "tsfresh." https://github.com/blue-yonder/tsfresh, 2018.

[43] S. K. C. Lester James V. Miranda, Aaron Moser, "pyswarms." https://github.com/ljvmiranda921/pyswarms, 2017.

[44] T. Tokunaga, D. Ikeda, K. Nakamura, T. Higuchi, A. Yoshikawa, T. Uozumi, A. Fujimoto, A. Morioka, K. Yumoto, and T. Higashimita, "Onset time determination of precursory events in time series data by an extension of singular spectrum transformation," *International Journal of Circuits, Systems and Signal Processing*, vol. 5, no. 1, pp. 46–60, 2011.

[45] T. Idé and K. Inoue, "Knowledge discovery from heterogeneous dynamic systems using change-point correlations," in *Proceedings of the 2005 SIAM International Conference on Data Mining*, pp. 571–575, SIAM, 2005.

[46] T. Idé and K. Tsuda, "Change-point detection using krylov subspace learning," in *Proceedings of the 2007 SIAM International Conference on Data Mining*, pp. 515–520, SIAM, 2007.

[47] Arundo, "adtk." https://arundo-adtk.readthedocs-hosted.com/en/stable/index.html, 2019.

[48] L. Eryk, "Outlier detection with isolation forest." https://towardsdatascience.com/outlier-detection-with-isolation-forest-3d190448d45e, 2018.

[49] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation-based anomaly detection," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 6, no. 1, pp. 1–39, 2012.

[50] B. Preis, "Data structures and algorithms with object-oriented design patterns in java," *John Wiley&Sons*, 1999.

[51] M. Christ, N. Braun, J. Neuffer, and A. W. Kempa-Liehr, "Time series feature extraction on basis of scalable hypothesis tests (tsfresh–a python package)," *Neurocomputing*, vol. 307, pp. 72–77, 2018.

[52] T. E. Oliphant, *A guide to NumPy*, vol. 1. Trelgol Publishing USA, 2006.

[53] W. McKinney *et al.*, "Data structures for statistical computing in python," in *Proceedings of the 9th Python in Science Conference*, vol. 445, pp. 51–56, Austin, TX, 2010.

[54] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. Jarrod Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. Carey, İ. Polat, Y. Feng, E. W. Moore, J. Vand erPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and S. . . Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[55] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pp. 265–283, 2016.

[56] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-International Conference on Neural Networks*, vol. 4, pp. 1942–1948, IEEE, 1995.

[57] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization," *Swarm intelligence*, vol. 1, no. 1, pp. 33–57, 2007.

[58] Y. Shi *et al.*, "Particle swarm optimization: developments, applications and resources," in *Proceedings of the 2001 congress on evolutionary computation (IEEE Cat. No. 01TH8546)*, vol. 1, pp. 81–86, IEEE, 2001.

[59] G. Rogers, *Power system oscillations*. Springer Science & Business Media, 2012.

[60] N. A. PASHTOON, "Iir digital filters," in *Handbook of Digital Signal Processing*, pp. 289–357, Elsevier, 1987.

[61] A. Allen, M. Singh, E. Muljadi, and S. Santoso, "Pmu data event detection: A user guide for power engineers," tech. rep., National Renewable Energy Lab.(NREL), Golden, CO (United States), 2014.

[62] M. A. Ibrahim, *Disturbance analysis for power systems.* John Wiley & Sons, 2011.

APPENDIX A: Transient Event Detection using SST

SST for Transient Detection: *banpei*
Inputs: w - window size
Output: Anomaly A
1. begin preprocessing data as per section 6.1
2. get w
3. calculate change point score (s) using *banpei* - section 5.4.1
4. **return** new time series with change point score t_s
5. calculate t_s mean
6. calculate t_s standard deviation
7. A = timestamps $\in t_s > 3$(standard deviation)*mean

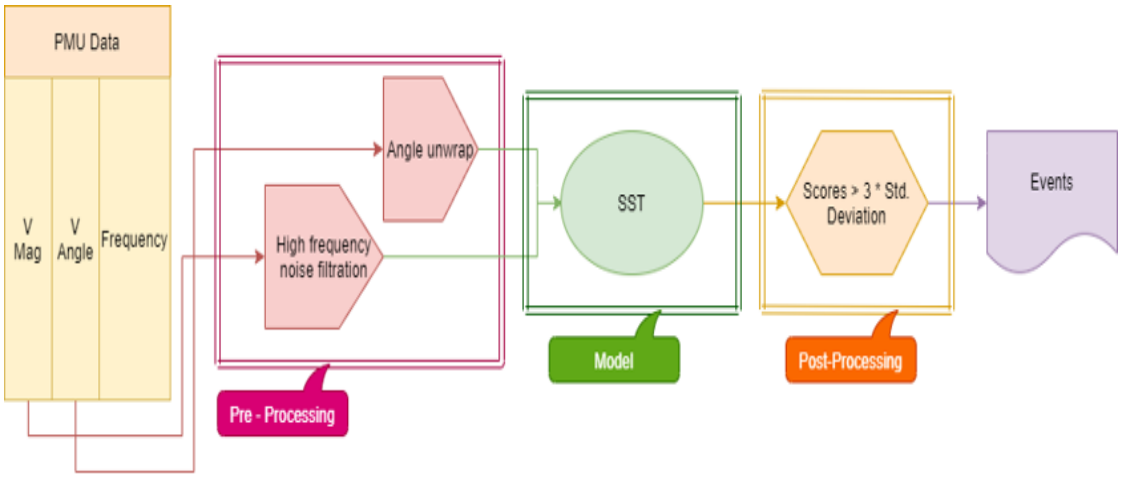


Figure A.1: Transient Event Detection

APPENDIX B: Impulse and Step Change Detection using Arundo ADTK

Arundo ADTK for Impulse and Step Change Detection: *PersistAD*, *LevelShiftAD*
Inputs: w - window size, c - normal bound
Output: Anomaly A
1. begin preprocessing data as per section 6.1
2. convert data to datetime index
3. get w
4. get c
5. select aggregation method → mean, median
6. calculate interquartile range IQR as per equation 5.9
7. A → bound > (Q1-c*IQR and Q3+c*IQR)

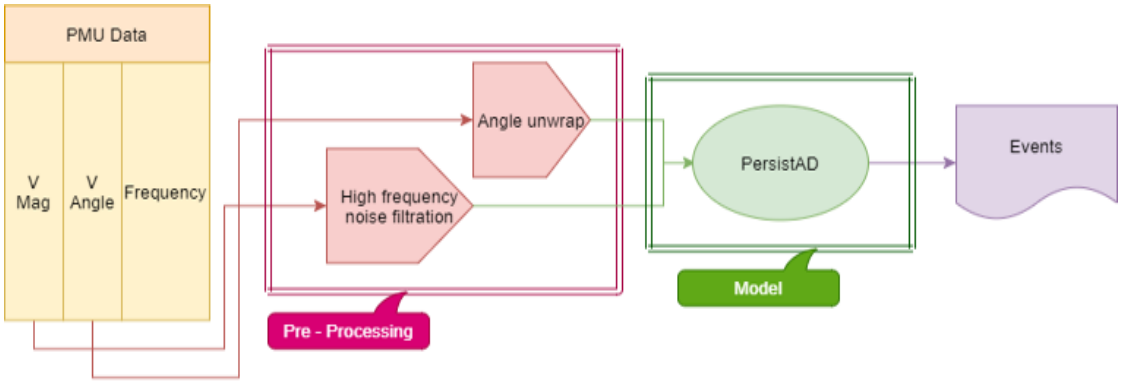


Figure B.1: Impulse Event Detection

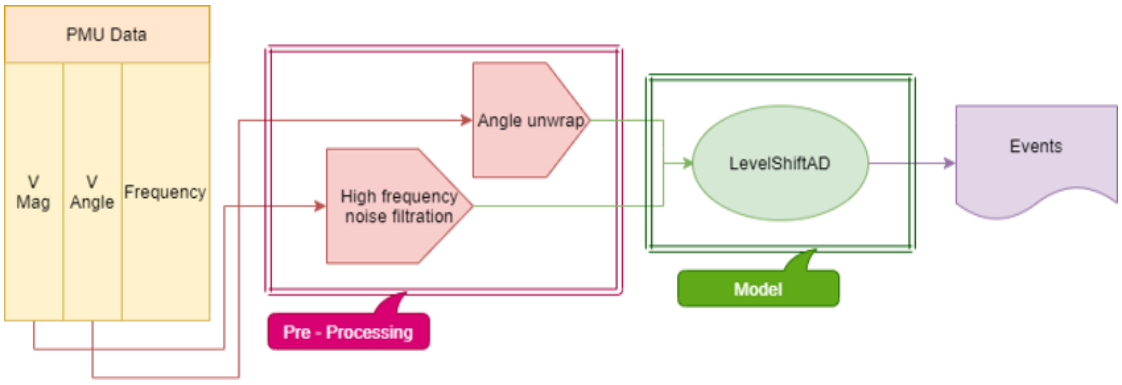


Figure B.2: Step Change Event Detection

APPENDIX C: Generalized Anomaly Detection using Isolation Forest

Generalized Anomaly Detection using Isolation Forest

Inputs: l - length of window, o - number of overlaps
c - contamination factor

Output: predicted events *pred*

1. begin preprocessing data as per section 6.1

% Distribute Data in Equal Window Lengths

2. get l
3. get o
4. shift = l - o
5. calculate number of new rows *(n)* in l
6. n = round((length(data) - l/shift)+1
7. create a list; *data_list*
8. **for**
9. initialize i where, $i \rightarrow 1$ to n+1
10. $data_{new}$ create new data set $\in$ *data_list* with given l
11. append $data_{new}$ to *data_list*
12. **end**

% Extract Features

13. Extract features using *tsfresh* form *data_list* 5.5.2.1

% Event Detection

14. get c
15. detect anomalies *pred* using isolation forest 5.5.1 with features as input

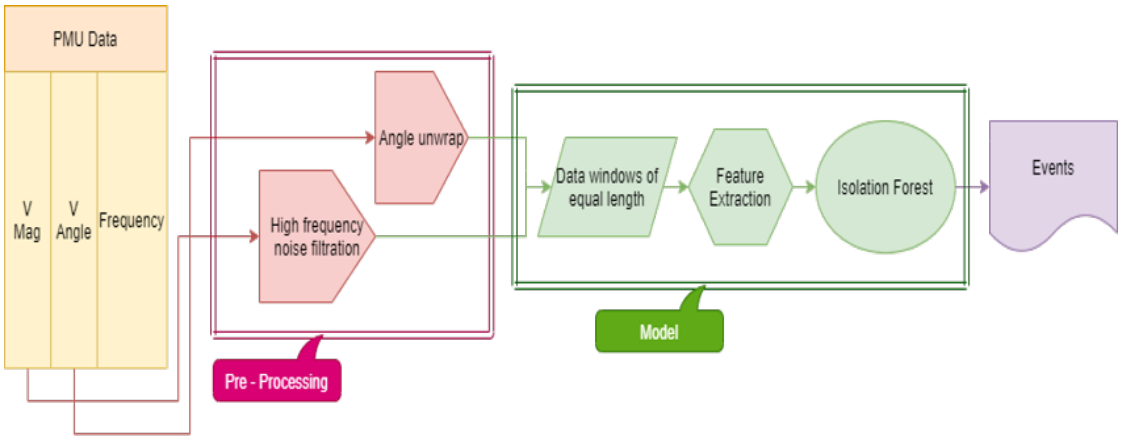


Figure C.1: Event Detection using Isolation Forest

APPENDIX D: Feature Optimization for Isolation Forest using PSO

Feature Optimization for Isolation Forest using PSO

Inputs: A_{Actual} - actual event timestamps, *pred* - events predicted using isolation forest, n - number of particles, i - number of iterations

Output: optimized feature position $f_{optimized}$

1. follow event detection procedure as in appendix C
2. **def** error
3. find common indices for A_{Actual} and *pred*
4. get length of common indices l_{common}
5. get length of actual event timestamp l_{actual}
6. error = $(l_{actual}$ - $l_{common})/l_{actual}$
7. **end**
8. **return** error
9. initialize n
10. initialize i
11. perform optimization as per 5.5.2.2
12. **return** $f_{optimized}$
13. repeat step 15 from appendix C with $f_{optimized}$ as input

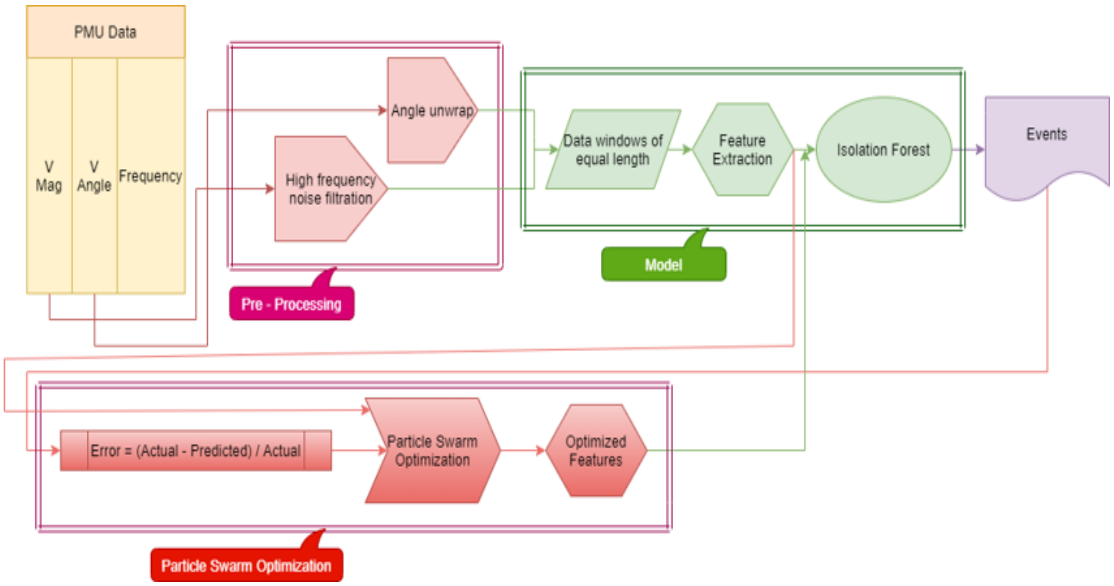


Figure D.1: Feature Optimization for Isolation Forest using PSO