7-22-2022

Development of a Configurable Real-time Event Detection Framework for Power Systems using Swarm Intelligence Optimization

Umar Farooq
Portland State University

Recommended Citation

Farooq, Umar, "Development of a Configurable Real-time Event Detection Framework for Power Systems using Swarm Intelligence Optimization" (2022). *Dissertations and Theses.* Paper 6108.
https://doi.org/10.15760/etd.7968

Development of a Configurable Real-time Event Detection Framework for Power Systems using Swarm Intelligence Optimization

by

Umar Farooq

A thesis submitted in partial fulfillment of the
requirements for the degree of

Master of Science
in
Electrical and Computer Engineering

Thesis Committee:
Robert B. Bass, Chair
John M. Acken
Mahima Gupta

Portland State University
2022

Abstract

Modern power systems characterized by complex topologies require accurate situational awareness to maintain an adequate level of reliability. Since they are large and spread over wide geographical areas, occurrence of failures is inevitable in power systems. Various generation and transmission disturbances give rise to a mismatch between generation and demand, which manifest as frequency events. These events can take the form of negligible frequency deviations or more severe emergencies that can precipitate cascading outages, depending on the severity of the disturbance and efficacy of remedial action schema. The impacts of such events have become more critical with recent decline in system inertia as they tend to exhibit larger frequency deviations and higher Rate of Change of Frequency in low inertia systems. The susceptibility of different Balancing Authorities to these events varies depending on their inertia levels. Due to the repercussions, it is indispensable to arrest such disturbances on time by activating responsive frequency control measures.

This study developed a configurable event detection framework using linear regression-based event detection algorithm with tunable parameters. Two swarm intelligence-based optimization algorithms, Grey Wolf Optimization (GWO) and Particle Swarm Optimization (PSO), were developed to dictate the algorithm parameter adjustments and make it adaptable to system conditions. The optimization algorithms tune the detection parameters according to the definition of frequency events specified by experts and enable event detection as

desired by system operators. The performance of GWO and PSO algorithms are analyzed using actual Phasor Measurement Unit (PMU) data, and the efficacy of the proposed system is demonstrated using a set of performance evaluation metrics. The proposed event detection framework is shown to be capable of detecting events with high accuracy and speed.

Dedication

To my parents, siblings, and friends.

Acknowledgements

I owe a special gratitude to the Fulbright Scholarship Program for giving me this opportunity to pursue my Master's degree at Portland State University. I am grateful to Dr. John M. Acken and Dr. Mahima Gupta for their willingness to serve on my thesis committee and share their expertise. I will seize this opportunity to present my deep and sincere gratitude to my supervisor and mentor, Dr. Robert Bass, for all his guidance, support, and knowledge sharing throughout this work. It was a valuable learning experience to work with him.

I would like to thank my parents for always being there for me. My gratitude is extended to the researchers whose papers and publications have been used to know the topic better and assisted me in performing my research work.

Table of Contents

List of Tables

List of Figures

Acronyms

ACO Ant Colony Optimization

BA Balancing Authority

CNN Convolution Neural Networks

DER Distributed Energy Resources

DWT Discrete Wavelet Transform

EA Evolutionary Algorithms

GA Genetic Algorithms

GPS Global Positioning System

GPU Graphics Processing Units

GWO Grey Wolf Optimization

NERC North American Electric Reliability Corporation

PCA Principal Component Analysis

PMU Phasor Measurement Unit

PSO Particle Swarm Optimization

RES Renewable Energy Sources

ROCOF Rate of Change of Frequency

RTAC Real-time Automation Controller

SAE Stack Auto-Encoders

SI Swarm Intelligence

1 Introduction

1.1 Problem Statement

The primary goal of system operators has always been to maintain stability of the power system and ensure continuity of supply to consumers in the event of disturbance [1]. As power systems are spread widely over large areas and exposed to weather conditions, they are vulnerable to faults and failures, which pose threats to system stability and security. Modern power systems are more vulnerable to experiencing critical situations since they are operated close to steady-state stability limits to maximize use of capital [2]. Therefore, modern power systems require sophisticated monitoring and control schemes to identify and mitigate such disturbances at an early stage. The ubiquitous unpredictability prevailing in power systems, such as loss of generation and/or transmission line outage, engenders an imbalance between supply and demand and creates hurdles for reliable operation. It is indispensable for preventing unforeseen collapse to efficiently curb every imbalance. Stable operation of power systems is dictated by maintaining frequency within permissible limits as close as possible to nominal value of 50/60 Hz, around $\pm 0.2\%$. Frequency, therefore, is a key reliability aspect of power systems and failure to maintain it within predetermined limits may lead to disruption in consumer supply, outage of generators and possibly a system breakdown [3]. Disturbances in power systems can be initiated by numerous factors such

as generator outage, transmission line tripping, lightning strike, equipment failure, human error, and substandard maintenance [4]. Table 1.1 summarizes cascading events and major blackouts that occurred globally from 2011 to 2018.

Table 1.1: Major power outages across the globe from 2011 to 2018 [5]

Region	Date	Duration (hours)	People Affected (million)	Causes
Mexico & USA	8 September 2011	12	2.7	Transmission line tripping
Brazil	4 February 2011	16	53	Transmission line fault and fluctuated power flow
India	30 July 2012	15	620	Transmission line overload
Bangladesh	1 November 2014	24	150	HVDC station outage
Pakistan	26 January 2015	2	140	Plant technical fault
Holland	27 March 2015	1.5	1	Bad weather
Turkey	31 March 2015	4	70	Power system failure
Ukraine	23 December 2015	6	230	Cyber-attack
Kenya	7 June 2016	4	10	Animal shorted the transformer
South Australia	28 September 2016	6.1	1.7	Storm and bad weather
US (NY)	1 March 2017	11	21	Cascading tripping of transmission system
US (Southeast)	10 September 2017	5	7.6	Cascading tripping of transmission system
Brazil	21 March 2018	1	10	Transmission line failure
Canada (BC)	20 December 2018	4	0.6	Heavy wind

1.2 Impact of Renewable Integration

The complexities and uncertainties associated with managing reliability of power systems have increased to an unprecedented level with the growing adoption of distributed power sources into the ever expanding power network. The primary concern in these modern power systems is the introduction of new challenges in maintaining frequency stability and grid resilience due to a reduction in the amount of reserve power. The rapid decline in system rotational inertia due to replacement of synchronous generators with inverter-based generators has further aggravated the situation [6]. The new system configuration requires advanced situational awareness capable of providing intelligent and automated actions for real-time monitoring and control of the grid. The inability to implement state-of-the-art techniques for modern problems will result in imposing conservative caps on allowable renewable energy capacity to preserve system security, hindering the transition to a more modern infrastructure. Figure 1.1 depicts an example of a frequency event.

Frequency control in a power system is the ability to normalize frequency to its rated value after a disturbance by keeping a balance of active power in the system. From a reliability perspective, frequency control capabilities are significant assets in power systems and have attained critical importance lately. Remarkable research efforts are put in this area with the recent large-scale adoption of renewable energy and Distributed Energy Resources (DER). In conventional power systems, synchronous generators were responsible for exertion of frequency control through their inertial response and governor actions. However, with major changes in the modern system topology driven by substitution of

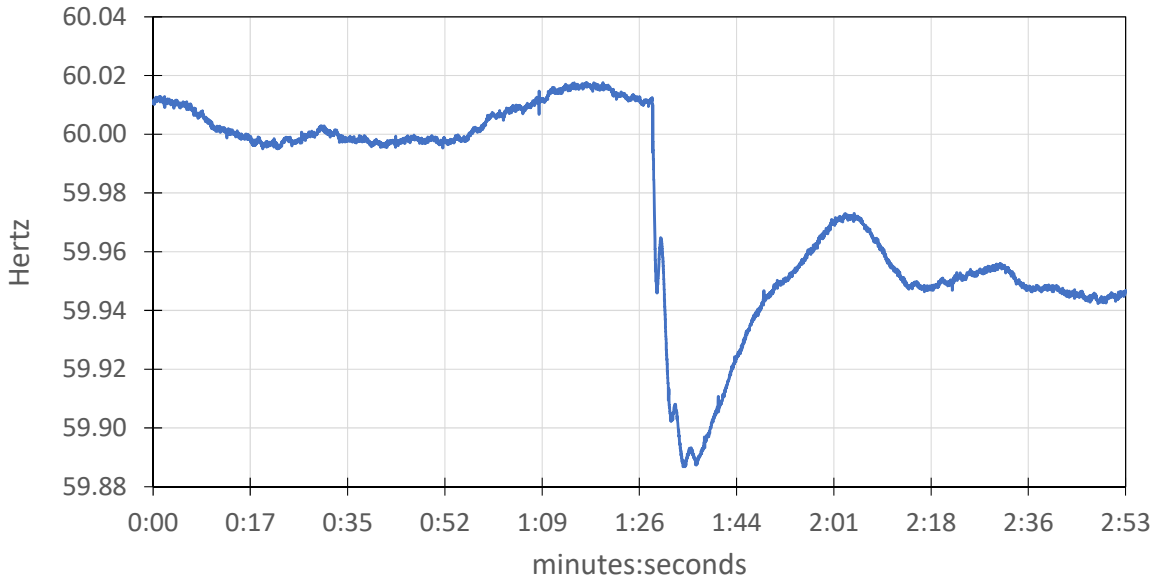

Figure 1.1: A frequency deviation measured within the U.S. Western Interconnection on January 20, 2020 at 0658. Frequency decreased from 60.01 Hz to 59.89 Hz (0.2%) in 5.9 seconds, then recovered to 59.95 Hz around 40 seconds later. Data sample rate is 60 frames per second.

synchronous generators by Renewable Energy Sourcess (RESs), it is not viable to control frequency with fewer number of conventional generators. With the decline in system rotational inertia engendered by the significant share of RESs, frequency control is becoming more of a decentralized task. To alleviate the need for conventional frequency regulation in modern power systems, attention has been drawn towards new sources of frequency control such as aggregation of demand side devices [7, 8, 9] and DER [10].

1.3 Events in Power Systems

Generally, cascading events in power systems have two stages and are not sudden events that cannot be prevented. In the first stage, there is a slowly evolving process of consecutive events, which deteriorates system operating conditions with every new successive disturbance. After occurrence of several disturbances, a transient action in the second stage

results in cascading events and ultimately system collapse. In most cases, it is too late to stop system breakdown at this point after cascading event is triggered. A possible cascading event can be averted by detecting it and taking proper control actions at an early stage [11].

The recent large-scale deployment of PMUs have enabled operators to have real-time insight into the operational state of power systems and a better situational awareness. PMUs are equipped with Global Positioning System (GPS) clocks, facilitating synchronization of multiple PMUs dispersed over a wide are.

PMU-based event detection methods can be classified into 1) signal analysis and 2) machine learning [12]. Signal analysis technique leverage signal processing methods, e.g. Wavelet Transform [13], Swing Door Trending [12]. Signal processing methods reported in literature can perform event detection but lack the ability to efficiently determine fault type, location, and are not tunable. The application of statistical feature extraction techniques by many PMU-based machine learning methods for event detection make them non-robust and ineffectual [14]. Other machine learning methods employ supervised learning and require large historical data sets with accurate labeled instances, such as Long Short-term Memory [15], Decision Trees [16], Support Vector Machines [16] and Artificial Neural Networks [17]. However the performance of supervised learning techniques depends highly on learning data and deteriorates with improper and insufficient selection of data [18].

This project aims to develop a configurable event detection framework for power systems whose parameters can be adjusted to produce results as desired by system operators. The proposed algorithm can be implemented for real-time event detection to monitor phasors

obtained from PMUs and trigger dispatchable frequency response assets upon inception of a frequency disturbance.

1.4 Objectives of the Project

High resolution measurement devices such as PMUs can contribute to numerous applications in modern power system concerning dynamic monitoring and control such as state estimation, load modeling, wide area protection, and event detection which is the focus of this project. Event detection refers to identifying occurrences of power system disturbances caused by outages or switching operations. For managing the health of critical infrastructure and maintaining stability of the power system, accurate detection of events plays a vital role to timely trigger remedial courses of action and successfully restore service. Requirements for primary frequency response (PFR) vary by jurisdiction. In Great Britain, PFR needs to activate within two seconds of the triggering event, with full provision of the requisite power within ten seconds [19]. Australian Energy Market Operator (AEMO) mandates a 5% increase in active power achieved within ten seconds of the frequency deviation from deadband, which is ± 1.5 Hz around the nominal value [19]. Therefore, detection of an event should be ensured within two seconds of its onset.

The definition of an event is not absolute; it varies for different systems depending upon critical stability limits. The North American Electric Reliability Corporation (NERC) has published Frequency Response Standard Background Document BAL-003 [20] wherein frequency events are extensively discussed but no standard definition is provided as to what

qualifies as a frequency event. Large stable interconnects with enormous synchronous inertia are less sensitive to frequency deviations. On the other hand, in smaller isolated powers systems or if the system stability is already compromised, Balancing Authorities (BAs) may be interested in arresting even minor frequency deviations to prevent unforeseen system collapse. Therefore, the detection algorithm may need to be configured for each BA to meet their specific system requirements. Contemporary event detection algorithms reported in the literature are not able to be configured and operated according to the system conditions.

The work done for this thesis facilitates the customization of the event detection algorithm by using an optimization algorithm [21]. The project has two parts. In the first part, an event detection algorithm based on calculation of frequency slope using least-sum-of-squares linear regression was developed to detect abnormal events [22]. The detection algorithm has five tunable parameters that can be adjusted to enable desired performance and has the capability to be implemented in an automation controller for online decision support using streaming PMU data. In the second part, swarm intelligence optimization was used to optimize the parameters of the detection algorithm to match the definition of frequency events as specified by experts from a BA. Two optimization algorithms were applied to this problem: Grey Wolf Optimization (GWO) and Particle Swarm Optimization (PSO). Both algorithms were tested and evaluated using a set of performance evaluation metrics.

The PMU installed in Portland State University Power Engineering Lab measures synchrophasors and archives them in a database as csv files. A set of frequency files from the archive comprised of events, non-events, and quasi-events, was presented to a

group of experts to evaluate each of the frequency plot as either an under-frequency event, over-frequency event, or non-event. The assessment took into consideration the expertise level of each expert to weigh their response for a final decision about each frequency plot. The assessment results were stored in a csv file, referred to as the human validation file, containing experts' information and a final classification for each candidate frequency plot. The human validation file defines the type of events that a BA is interested to detect. Such a human validation file can be easily created for each BA reflecting the specific needs and stability limits of that BA, and can be updated owing to the seasonal variations or changes in the system topology. Each optimization algorithm takes this human validation along with the original frequency files recorded by PMU as inputs and produces optimized parameters of the detection algorithm as output which are used for detection of events as desired by the experts. This framework facilitates the configuration of the detection algorithm to enable its operation according to the detection requirement of a BA. Once tuned, the algorithm can then monitor streaming PMU data. Performance of the optimization algorithms was validated by evaluating event detection accuracy and speed using the optimized parameters. Figure1.2 depicts the process flow diagram.

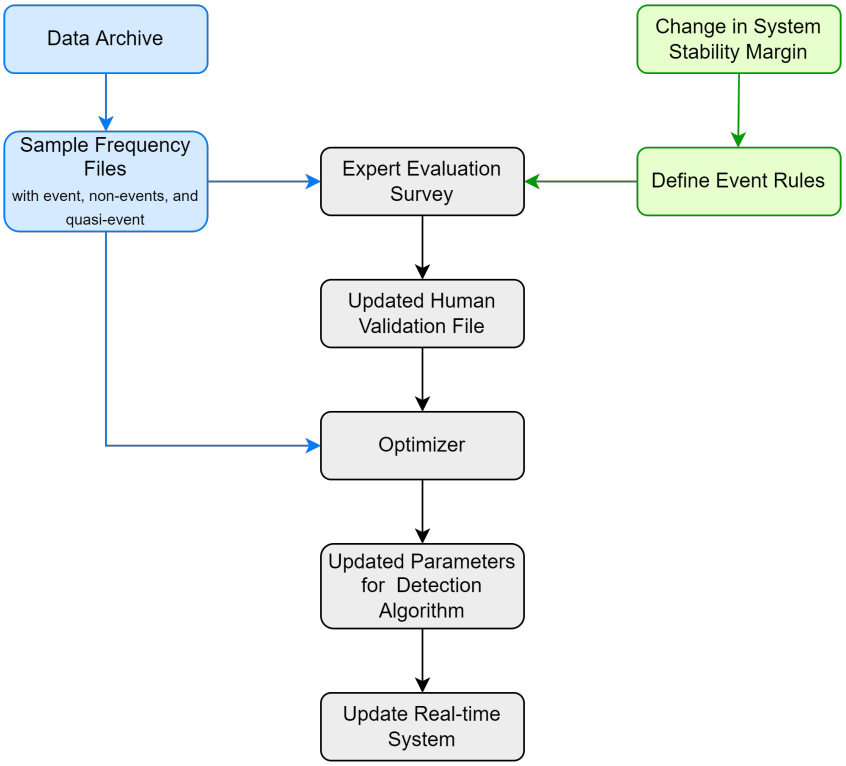

Figure 1.2: Process flow diagram for event detection and optimization framework

2 Literature Review

2.1 Event Detection

Frequency is an essential aspect of system reliability. The inability to arrest any frequency disturbance may lead to catastrophic events in power systems. The effects of these events are amplified in modern power systems characterized by large-scale renewable and distributed generation. As discussed earlier, NERC has put special emphasis on BAs to respond to frequency events but has not provided a defining criteria for frequency events [20]. In fact, frequency event is a relative term that varies for every power system. During a disturbance, a power deficit is corrected instantaneously by the inertial response of system generators, which comes from the kinetic energy stored in rotating mass. Governor response has an inherent time delay resulting from the adjustment of equipment and conversion of energy at turbine blades. Thus, the availability of large amounts of balancing inertia in a power system makes it less susceptible to be adversely affected by frequency deviations, as compared to a system with lower rotational inertia. As a real life example from the perspective of a control room engineer, the author has witnessed a situation when the system voltages increased to an unsafe level due to a sudden drop in load owing to bad weather condition. After energizing available shunt reactors for voltage control, the last resort was to disconnect parallel circuits and force the load to flow on a single circuit. This reduced the system contingency level. In

such a situation when system stability is already compromised, the system operator may want to capture even minor deviations to avoid a catastrophic event. Therefore, a detection algorithm should be able to be tuned for each BA to perform as per their specific system requirements.

Automatic event detection in power systems has engaged researchers lately and an extensive work is reported in literature about this topic. Online event detection systems take phasor measurements and provide decision support for protection and control schemes. The contemporary work on power system event detection reported in literature is divided into three main categories: Signal Processing methods, Statistical Analysis methods, and Machine Learning methods.

2.1.1 Signal Processing based methods

Event detection techniques based on signal processing often use Discrete Wavelet Transform (DWT) to decompose frequency and voltage signals for detection of active and reactive disturbances in a network. DWT involves the correlation of a measured signal with a mother wavelet at a discrete set of scales and translations. Anshuman et al. [23] used DWT to differentiate between transient events followed by oscillatory events. The method derives detail coefficients by applying DWT to the voltage and frequency signals. A novel indicator is proposed to calculate energy of the normalized signals and record the peak at all PMUs affected by an event. Peak values of the signals are compared to classify the event as an active or reactive power event. For the analysis of oscillatory events following a transient event, the proposed method uses approximate coefficient obtained after applying DWT to

PMU data. The method can identify the oscillatory frequencies that have the ability to lead to an instability. The method requires two seconds of PMU data at 60 Hz sampling rate to properly detect a transient event.

Kim et al. presented a wavelet-based algorithm to detect power events using frequency and voltage measurements from a PMU [24]. Wavelet analysis is a tool used to investigate transient behavior in a signal by transforming a one-dimensional function into a two-dimensional series of coefficients. Wavelet-based detection methods monitor coefficient energy over a moving window and identify a disturbance when the energy exceeds a certain threshold. To reduce the effect of variability in PMU data and nonevent disturbance, the proposed method uses a normalized wavelet energy function to calculate RMS of detail coefficients. The result of normalized wavelet energy function applied to PMU data is shown to be one in normal condition, and greater than one during an event.

An online event detection method is presented by Konakalla and Callafon using synchrophasor data [25]. This signal processing-based technique applies discrete time filtering to real-time phasors from a PMU for estimation of optimal filtered rate of change (FRoC) signals and detects an event using dynamic response of the output signal. The filtering of phasor data leads to a signal with a minimum variance. The parameters of the linear discrete time filter are estimated by formulating least square optimization. The proposed event detection algorithm monitors the number of times for which consecutive samples of filtered signals outstripped the variance limits.

Data obtained in low voltage networks can be used to study system dynamics since

major events that occur at high voltage levels have their effects propagated downwards to the distribution network. Vaz et al. proposed an event detection scheme using measured data at low voltage [26]. The DWT-based approach monitors the extracted energy from the wavelet detail coefficients of a PMU signal to detect an event. Certain parameters such as levels of signal decomposition, threshold and decision variables dictate the performance of the algorithm. PSO is used to optimize and select the parameters.

The above schemes [23, 24, 26] need measurement data from every bus in the network for reliable operation. Moreover, since the range of coefficient energy depends on window size, the results are highly susceptible to noisy PMU data and variations in window size. The performance of many signal processing based methods [23, 24, 25, 26] is also affected by the requirement of a user-defined threshold for proper event detection. The threshold values depend on the quality and nature of PMU data and need to be configured for each PMU in the network, which is a challenging issue. Furthermore, filtering or discrete Fourier transform based approaches require a sufficient sampling data for proper operation.

2.1.2 Statistics based methods

Principal Component Analysis (PCA)-based approaches have been widely used for event detection recently. Xie et al. proposed a dimensionality reduction method using PCA to handle large-scale data from multiple PMU deployed in a power system [27]. The method is based on dimensionality analysis of PMU data to enable power system anomaly detection using the concept of a change in core subspaces of measured data during an event. Initially in offline mode, the model gets a linear basis from PCA. In online implementation, the

event detection algorithm uses data from a few pilot PMUs to estimate system conditions for all other PMUs. The algorithm uses prediction error between actual and projected data to identify events and update models using an adaptive training mechanism. A similar work was carried out by Rafferty et al. using a moving window PCA for detection and classification of multiple events [28]. PCA-based frequency data analysis is used to compute two statistical variables representing disparity in recorded data, and a dissimilarity between measurements and its lower dimensional PCA representation. The adaptive learning mechanism allows the proposed algorithm to update the statistical variables with every new sample of normal data over a window. Event detection is carried out when the variables overshoot pre-defined confidence limits. As opposed to [27], this method uses a sliding window to adapt to the varying behaviors of a power system. Similarly, Xu and Overbye presented an event detection scheme using measurements from multiple buses [29]. The techniques uses PCA to analyze dynamic behavior of the system and highlight dominating buses after a disturbance. Derived system information is efficiently presented with the adoption of a visualization technique. However, the proposed technique does not specify a method for selection of the dominating buses. Detection methods based on PCA are heavily dependent on training data. For PCA-based supervised learning techniques, if an improper and insufficient sample space is selected, the performance of the resulting ill-trained model will be negatively affected.

An ensemble technique for event detection was presented by Pandey et al. using statistics and clustering techniques [30]. The proposed work uses a Density-based spatial clustering of applications with noise tool for detection of events. The problem with this scheme is that

it detects events based on multifarious phasor data such as frequency, current, voltage, active and reactive power from all PMUs, which adds to the complexity of the algorithm.

Recently, the challenge of event detection in weakly damped power systems was addressed by Zhu and Hill using spatial-temporal data analysis based method [31]. From the perspective of spatial-temporal correlations between multiple bus PMU data engendered by an event, the proposed data-driven approach features the similarities in regional PMU measurements by profiling spatial temporal nearest neighbor of time series data from multiple buses to detect occurrence of events. This method requires manual threshold tuning to be applied to different systems and thus needs an intelligent scheme for adaptive threshold adjustment. The online performance is also adversely affected by missing PMU data. Furthermore, this method relies on measurements from multiple buses in the system.

Statistical indices such as mean, variance, minimum, maximum, and correlation are used over a window to reduce computational complexity, but these values may vary even during the same event and hence it is very challenging to define a threshold.

2.1.3 Machine Learning/Deep Learning based methods

With the recent development of advanced data processing resources such as Graphics Processing Units (GPU) and machine learning techniques, Deep Learning has been widely employed for solving various power system problems. Due to the remarkable feature extraction capability of Convolution Neural Networks (CNN), Miranda et al. used this sophisticated image recognition tool for classification of various power system events such as generation loss, load loss, line tripping, and inter-area oscillations [32]. The work builds

upon the idea of transforming time-series frequency data from PMUs into two-dimensional images to be subject to a CNN for extracting knowledge of the type of event. A CNN has the capability to identify features in images by exploiting correlation among adjacent pixels. Motor loads have an inherent frequency response capability in a power system. Due to insufficient frequency response from motor loads in a lightly-loaded system, the frequency may continue to ramp down even in the absence of an event. The proposed method is incapable of distinguishing such a condition from an actual event due to reliance on only frequency data for image recognition. A frequency ramping event in a power system is the sudden change in frequency caused by a scheduled increase or decrease in generation in an interconnected grid. Therefore, frequency data cannot be used as the only input to a CNN for event detection in a power system since it will limit the capability of the model to differentiate a ramping event from an actual event. To overcome this difficulty, Wang et al. introduced a method that uses a Relative Phase Angle (RAS) signal in addition to Rate of Change of Frequency (ROCOF) to be transformed to images [33]. Any event in a power system generates an electromechanical wave that travels through the grid and creates angle shifts in the network. The authors used this idea and combined it with the transformation of frequency data into ROCOF to improve efficiency and accuracy. The work used a classifier fusion to combine the results from the ROCOF model and the RAS model to give a final decision about event detection. Results of proposed model were compared with the ROCOF-only model and frequency-only CNN model for generator trips and load loss. A significant improvement in accuracy was observed. FDR data from the U.S. Eastern

Interconnection was used for model validation.

Yufie et al. presented a deep learning technique, Stack Auto-Encoders (SAE), for event monitoring using PMU data [34]. The method is based on capturing the features of a dynamic event from PMU data. In the first step of the event detection framework, an energy function is formulated with construction of each of the components such as generation, transmission line, and load. Each energy function component is sensitive to a specific type of event that involves that component. For example, transmission line energy is sensitive to a fault. Similarly, generator energy is sensitive to an event that involves rotor speed deviation. From the system point of view, these components are invoked simultaneously upon the occurrence of an event due to high inherent correlation among them. In the second step, SAE is employed as a feature learning tool to learn dynamic signatures from the components. An SAE is comprised of multiple autoencoders working together to find highly complex and nonlinear data patterns. In the final step of supervised classifier training, a simple neural network is used to train in offline mode and then detect events in real-time by calculating energy function components from PMU data. Like other machine learning based techniques, the performance of this methods depends on the amount and quality of labeled historical data available.

Wang et al. integrated a quickest-change detection framework with a time series prediction model for event detection [35]. A recurrent neural network LSTM-model is adopted to capture the trend in power system measurements. Temporal evolution in PMU measurements can be captured by training the LSTM model with historical data and using the trained

model to predict system states in real-time. The model uses training data to learn about the normal distribution of prediction error under normal conditions. In online implementation mode, the trained LSTM model cannot predict the system parameters during occurrence of an event and hence the prediction error will change its distribution. The dramatic change in the statistics of the prediction error is tracked by using a cumulative sum (CUSUM) approach. The performance of this method depends on historical data.

Kesici et al. presented an online power system monitoring scheme by using a sliding window-based CNN model for identification of various phases of a power system: pre-fault, fault inception, fault duration, and post-fault [36]. Although CNN is an image recognition tool, the motive behind using CNN in this work is its capability of processing high dimensional data. Voltage magnitude measurements from a PMU represented as time series data are directly used as input to a CNN without transforming them to images to avoid loss of data. Positive sequence voltage magnitude data from the simulation of a three phase fault are used to generate a dataset for training a CNN. The model was validated by considering two scenarios for PMU placement. The first scenario takes data from all buses, whereas the second scenario takes data from optimally place PMUs.

The problem with these supervised learning methods is that they mostly suffer from inadequate amount of labeled training data. Some rare and uncommon events are not reflected in recorded PMU data. Secondly, utility event logs also miss many events. The irregular labelling of learning data may result in a biased model. The inappropriate and inadequate selection of training data adversely affect the efficiency and accuracy of these

models. Moreover, the number of recorded events is also restricted by the limited installation of PMUs. Finally, these models cannot be run without the aid of software tools and computers in control centers.

2.2 Optimization

Optimization involves modeling a problem in terms of an evaluation function called the objective function, and then using a search algorithm to minimize or maximize that objective function. Over the last two decades, meta-heuristic optimization algorithms have gained popularity. Some of these techniques such as Particle Swarm Optimization (PSO) [37], Genetic Algorithms (GA) [38], and Ant Colony Optimization (ACO) [39] are widely known in many fields. Most of the meta-heuristics are inspired by some aspect of nature. Nature has acquired solutions for almost every problem through the process of evolution over billion of years. Every meta-heuristic algorithm involves randomization at initial stage, and choosing the best solution. Randomization allows avoidance of local optima, and choosing the best enables convergence towards an optimal solution.

The problem solving mechanism of meta-heuristics is divided into two phases: exploration and exploitation. Exploration refers to the maximally broad and global search for promising regions to find the best solution in solution space. Exploitation refers to a local search to further explore those promising regions for a better solution. The vast applications of meta-heuristic, especially for this project, is due to four reasons: simplicity, derivation-free solutions, global optimality, and flexibility [40].

Meta-heuristic techniques are simple because they have been inspired by simple phenomena. Typically, they are inspired by animal behaviors, physical phenomena, or evolutionary concepts. Due to their simplicity, meta-heuristics can be easily understood and applied to different problems. Moreover, the simplicity enables scientists and researchers to imitate different natural processes and improve current methods, hybridize different methods, or proposed new ones.

Meta-heuristics adopt stochastic approaches to optimize problems as opposed to gradient-based optimization, which characterize them with a derivation-free mechanism. Finding an optimal solution does not require calculation of the derivative of search spaces as the process is initiated with random solutions. This gives meta-heuristics superiority over other methods for problems where derivatives of search space are unknown or difficult to get.

Meta-heuristics have better performance as compared to classical optimization methods for avoiding local optima. The ability of avoiding stagnation and extensively searching the search space come from the stochastic behavior of these methods. Meta-heuristics are highly suitable for optimization of real problems, which usually have unknown and complex search spaces with numerous local optima.

Owing to their flexibility, meta-heuristics do not need significant changes in the algorithm before application to different problems. Due to their superior ability of handling problems as black boxes, they can be conveniently used for optimization of different problems. Since inputs and outputs are the only parameters that signify, researchers just need to model their problems to apply meta-heuristics.

Meta-heuristics can be divided into three main classes: evolutionary algorithms, physics-based algorithms, and Swarm Intelligence algorithms.

2.2.1 Evolutionary Algorithms

The first main branch of meta-heuristics are Evolutionary Algorithms (EA), which mimic the idea of biological evolution. GAs are a widely-known technique in this branch. Holland first proposed GAs in 1992 by emulating Darwin's evolutionary theory [41]. Based on the concepts of survival of the fittest, GAs use the principles of reproduction, crossover, and mutation. The applications of GAs to engineering problems were substantially analyzed by Goldberg [42].

Generally speaking, EAs solve optimization problems by commencing with a random initial population, which evolves with time. The population at each stage is used to create a new generation of population by mutation. A fitness value is calculated for each individual in a population to evaluate its probability to participate in creating new generation. This process enables the enhancement of population at every iteration and guarantees convergence of the process.

The biogeography-based optimizer is an example of EA, which was pioneered by Simon in 2008 [43]. The development of this algorithm has been motivated by biogeography, which is the study of the geographical distribution of biological organisms. The motivation was to combine biogeography with engineering to see the benefits of the its applications to optimization problems. Biogeographic studies involve geographical distribution over space and time - different geographic locations over different time periods. It investigates different

ecosystems and spatial patterns of biodiversity to understand the patterns of variation concerning migration and mutation. The mathematical representation of biogeography explains the migration, creation, and extinction of species. The main inspiration of this algorithm was the evolution of biological habitats and territories over migration and mutation to achieve stability.

2.2.2 Physics-based Algorithms

Physics-based algorithms are the second subclass of meta-heuristics, which are inspired by physics rules. This branch of meta-heuristics came into existence with Richard Feynman's work on quantum computing in 1982, which was motivated by quantum mechanics [44]. This work paved the way for developing the concept of quantum computing. With this, the evolution of physics-based optimization commenced with the proposal of Quantum-inspired Genetic Algorithms (QGA) by Narayanan and Moore in 1996 [45]. As opposed to EAs inspired by Darwin's theory of survival of the fittest, these methods follow physics rules to solve optimization problem with a random set of search agents that communicate and explore the search space. The movement of these search agents throughout the search space is dictated by a physics concept such as gravity, electromagnetism, shifting of asteroids in space, nuclear collision reactions, gravitational radiation, inertia, and weight.

Some of the famous physics-based algorithm are Gravitational Search Algorithm (GSA) [46], Central Force Optimization (CFO) [47], Galaxy-based Search Algorithm (GbSA) [48], and Big-Bang Big-Crunch (BBBC) [49]. GSA is based on universal gravitational law and mass interactions. Every search agent is considered an object with a mass in an imaginary

universe represented by a solution space. Performance of objects, interacting with each other according to Newton's law of gravity and law of motion, is measured by their masses with the heaviest mass representing the best solution. Convergence is achieved with the concept of a gravitational constant, the value of which increases in every iteration. CFO is inspired by the theory of gravitational kinematics. Applying law of gravitation, which states that bigger masses exert a larger force of attraction as compared to smaller masses, a global optimum can be represented by the biggest mass in the context of optimization. A set of solutions, modeled as probes, moves in the search space according to the equations of particle motion in a gravitation field. Each probe gradually moves towards the probe that has gained the highest mass and thus arrives at a global optimum solution. GbSA mimics the spiral arm nature of the galaxies to search its surrounding. The algorithm avoids local optima by introducing chaos into the spiral movement-based search space exploration. Along the exploration process, if a better solution is obtained, a local search mechanism is initiated to search for a better solution near the newly obtained solution, which ensures exploitation. BBBC, inspired by the expanding and shrinking phenomenon of big bang and big crunch respectively, solve the optimization problem in two phases. During big bang phase, a new population of search agents is generated and scattered randomly within a search space. After exploring the search space, these agents gather at a point of best solution during big crunch phase.

2.2.3 Swarm Intelligence-based Algorithms

Swarm Intelligence (SI)-based techniques, the third branch of meta-heuristics, are fashioned on the collective intelligent behavior of entities in nature, especially insects and animals known for self-organization and forming of clusters. The social intelligence of these creatures in nature has fascinated scientists and researchers to simulate these behaviors by formulating advance algorithms. SI algorithms use a similar search process as physics-based methods, but the movement of search agents in SI methods is characterized by simulated knowledge swarming behavior of these entities. SI methods are characterized by two basic natural concepts: self-organization and division of labor. Self-organization allows the search agents to converge to an optimum without any external interference. Division of labor, defined as the simultaneous execution of different tasks by sub-populations, ensures tackling of complex problems.

Some of the well-known SI optimization techniques are Particle Swarm Optimization (PSO) [37], Ant Colony Optimization (ACO) [39], and Grey Wolf Optimization (GWO) [40]. PSO was introduced by Kennedy and Eberhart in 1995 to simulate the swarm behavior of birds flocking. Multiple particles move through the search space and evolve towards an optimum by taking into consideration the position of the best particle and the best position of the swarm obtained so far. ACO was proposed by Dorigo et al. and inspired by the foraging behavior of ants. Ants use indirect communication by depositing pheromone on the trails to mark a favorable path between a food source and their nest. The intensity of this chemical changes with time. ACO obtained its main inspiration from the social behavior of ants to

mark the shortest path to be followed by the other ants. In ACO, pheromones represent the intensity of a path in the search space explored by the search agents. A pheromone matrix evolves with each iteration and search agents follow the path with the highest intensity. GWO is another SI method proposed by Mirjalili et al. in 2014. It is inspired by the social behavior and leadership hierarchy of grey wolves. Among the SI-based algorithms, GWO and PSO have been widely used for various applications [50, 51, 52, 53] due to their simplicity and superior results. This project used GWO and PSO for parameter adjust of the detection algorithm. The inspiration and mathematical modeling of GWO and PSO are discussed in Chapter 3 and 4, respectively.

3 Grey Wolf Optimization Algorithm

3.1 Inspiration

Grey Wolf Optimization is an SI method introduced by Mirjalili et al. in 2014 [40]. The main inspiration of this algorithm was the hunting approach and social hierarchy of grey wolves. The social behavior of grey wolves is characterized by formation of packs with a strict hierarchy in descending order of dominance as shown in Figure 3.1. The social hierarchy has four types of wolves: alpha, beta, delta, and omega. This social hierarchy is simulated in the GWO algorithm.

Alphas are the leaders, with a male and a female, who make decisions for the pack such as hunting, sleeping, and waking. Mostly, alphas are responsible for making decisions, which are followed by the entire pack, but in some cases democracy is followed. Although the dominant wolves, alphas are not necessarily the strongest in the pack but have best managerial skills, which shows that discipline of pack is more important than strength.

The beta wolf, either a male or female, lies on the second level in the hierarchy and is the best replacement for an alpha in case of emergency. It acts as an advisor to the alpha to help with decision making, and as a commander to the pack to enforce the decisions.

Omegas have the lowest dominance in the hierarchy and obey all other dominant wolves. They seem to be the least important wolves in the pack with the role of a scapegoat but their

absence cause internal fighting in the pack because they are used for venting frustration of the pack. Thus, they have a role in maintain the dominance structure of the pack.

A wolf that does not belong to any of the mentioned hierarchies is called a delta wolf. They are subordinate to alphas and betas, but dominate omegas. They include sentinels, hunters, scouts, and elders.

The search space modeling in GWO is inspired by the interesting group hunting behavior of grey wolves, which involves the following phases [40]:

- Searching and approaching the prey
- Encircling and harassing
- Attacking

Exploration of the search space is inspired by the first two phases, whereas, the last phase is simulated for exploitation.

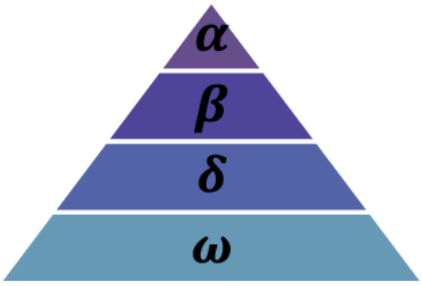

Figure 3.1: Social hierarcy of grey wolves [40]

3.2 Mathematical Modeling

This section presents the mathematical modeling and representation of social hierarchy and hunting behavior of grey wolves. In the context of GWO algorithm, prey represents the optimum (maximum or minimum depending upon the problem). Wolves are the the search agents, tracking and encircling represents exploration of the solution space, and attacking the prey represents exploitation process.

In the mathematical representation of social hierarchy of grey wolves, the best solution in the solution space is considered as alpha (α). The second and third best solutions are called beta (β) and delta (δ) respectively. α, β, and δ explore the search space during the optimization process and the rest of the candidate solutions, represented as omega (ω), follow these search agents.

The hunting process of a pack is carried out by alpha, beta, and delta, with alpha being the leader who guides the pack. They have the ability to locate and hunt the prey in real life. To simulate this behavior in an abstract search space, we designate alpha, beta, and delta as the best candidate solutions obtained so far with the assumption that they are close to the prey (optimum). The positions of these wolves (search agents), representing the three best solutions, are saved in each iteration and rest of the search agents update their positions according to them. The following mathematical equations simulate the encircling behavior of grey wolves in an abstract search space [40]:

$$\vec{D} = |\vec{C}\vec{X}_p^{\,t} - \vec{X}^t| \tag{3.1}$$

$$\vec{X}^{t+1} = \vec{X}_p^{\,t} - \vec{A}\vec{D} \tag{3.2}$$

$$\vec{A} = 2\vec{a}\vec{r_1} - \vec{a} \tag{3.3}$$

$$\vec{C} = 2\vec{r_2} \tag{3.4}$$

Where $\vec{X_p}$ indicates position vector of the prey (optimum), $\vec{X}$ represents the position vector of a wolf (search agent), $\vec{A}$ and $\vec{C}$ represents coefficient vectors for adjusting position of search agents, t indicates current iteration, $\vec{r_1}$ and $\vec{r_2}$ are random vectors between 0 and 1, and $\vec{a}$ is an important parameter whose values is linearly decreased from 2 to 0 over the optimization process.

Values of the coefficient vectors $\vec{A}$ and $\vec{C}$ can be adjusted to move the search agent in the search space with respect to the current position. The random vectors $\vec{r_1}$ and $\vec{r_2}$ help simulate the natural encircling behavior of a grey wolf so that it may take any random position within a boundary around the prey. As depicted by the 2D illustration in Figure 3.2, the position of a grey wolf given by (X, Y) can be updated with respect to the position of the prey (X^*, Y^*). This concept can be extended to an n-dimensional solution space. Equations 3.1 and 3.2 can be translated to simulate the hunting behavior wherein the rest of the wolves update their positions according to the three best positions. Following equations illustrate that the a wolf will move randomly within a circle around the prey whose position is estimated by alpha, beta, and delta.

$$\vec{D_\alpha} = |\vec{C_1}\vec{X_\alpha} - \vec{X}|, \vec{D_\beta} = |\vec{C_2}\vec{X_\beta} - \vec{X}|, \vec{D_\delta} = |\vec{C_3}\vec{X_\delta} - \vec{X}| \tag{3.5}$$

$$\vec{X_1} = \vec{X_\alpha} - \vec{A_1}\vec{D_\alpha}, \vec{X_2} = \vec{X_\beta} - \vec{A_2}\vec{D_\beta}, \vec{X_3} = \vec{X_\delta} - \vec{A_3}\vec{D_\delta} \tag{3.6}$$

$$\vec{X}^{t+1} = \frac{\vec{X_1} + \vec{X_2} + \vec{X_3}}{3} \tag{3.7}$$

Where:

- $\vec{X_\alpha}$, $\vec{X_\beta}$, and $\vec{X_\delta}$ are the position vectors of alpha, beta, and delta, which represent three best solutions during the search.

- $\vec{X}$ is the position vector of all other wolves, which represents other candidate solutions.

- $\vec{D_\alpha}$, $\vec{D_\beta}$, and $\vec{D_\delta}$ calculates the distance of alpha, beta, and delta from the rest of the wolves respectively.

- $\vec{X_1}$, $\vec{X_2}$, and $\vec{X_3}$ are the position vectors calculated with respect to the positions of alpha, beta, and delta respectively. Each of these vectors reflects a closer position for omega in reference to alpha, beta, and delta respectively. Mean of the three calculated positions gives an optimal new position for an omega wolf.

Figure 3.3 shows position update of rest of the wolves with respect to alpha, beta, and delta.

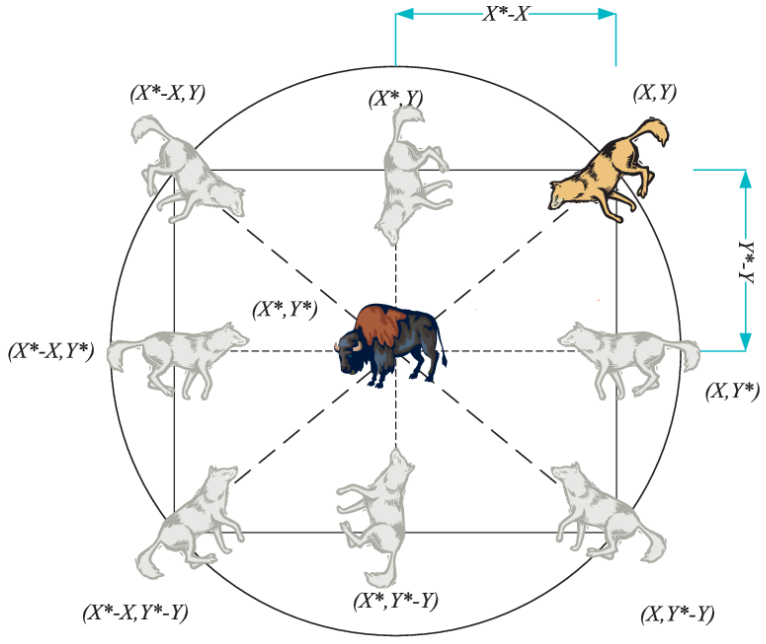

Figure 3.2: 2D illustration of encircling the prey and position vector of grey wolf [40]

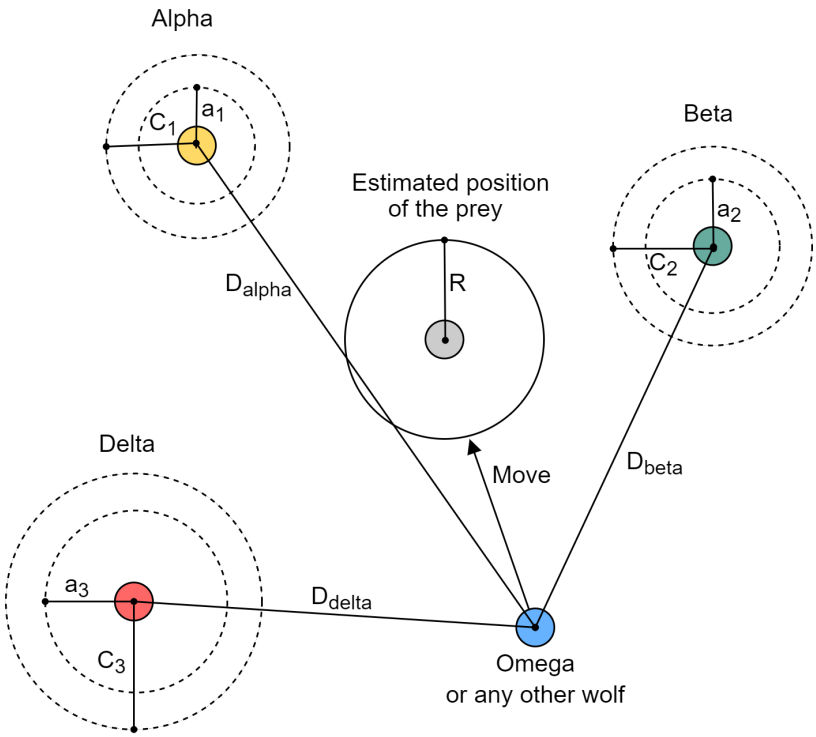

Figure 3.3: Position update of omega or any other wolf with respect to alpha, beta, and delta [40]

3.3 Exploration

GWO algorithm uses two coefficient vectors, $\vec{A}$ and $\vec{C}$, to mimic the divergence of grey wolves and realize a global search in the solution space. The values of $\vec{A}$ fluctuate randomly within a range that depends on $\vec{a}$. The fluctuation range of $\vec{A}$ decreases when a is linearly decreased from 2 to 0 during the optimization process. Divergence of search agents in the solution space is simulated when $\vec{A}$ is used with random values outside the range [-1,1]. In other words, search agents will diverge to find a better prey when $|A| > 1$ as shown in Figure 3.4(b).

$\vec{C}$ is used to achieve exploration and avoid local optima stagnation. The random values in vector $\vec{C}$ are used to assign weights to the prey for calculating distance in Eq. 3.1. This also

simulates the effect of obstacles in nature, which prevent a quick and convenient attack. As opposed to A, the value of C is not linearly decreased in order to emphasize randomization during exploration and avoid local optima. This randomization ensures exploration even in the final iterations.

3.4 Exploitation

The convergence of grey wolves to attack the prey is mathematically modeled when $\vec{A}$ has random values in the range [-1,1]. As discussed earlier, the linear decrease in the value of $\vec{a}$ over the course of iteration simulates approaching the prey and allow both exploration and exploitation. When $|A| < 1$, a search agent will move close to the prey as shown in Figure 3.4(a). In other words, $|A| < 1$ will ensure that a search agent approaches the prey in the next iteration by taking a closer position to the prey with respect to its current position.

To summarize the whole process, GWO begins with a random population of search agents. Alpha, beta, and delta, being the best solutions, estimate the position of the prey. Other search agents update their positions with respect to these best solutions to get closer to the prey. $\vec{A}$ and $\vec{a}$ guarantee exploration and exploitation when $\vec{a}$ is decreased linearly from 2 to 0. The search agents diverge in the search space for half of the iterations when $|A| > 1$ and then converge towards the prey for the other half when $|A| < 1$. Satisfaction of an end criteria terminates the algorithm. The reduced number of parameter adjustments in GWO (only a and C) make it favorable for various applications, including maximum power point tracking of wind turbine [50], optimal reactive power dispatch [54], and blackout prevention

in smart grids [51].

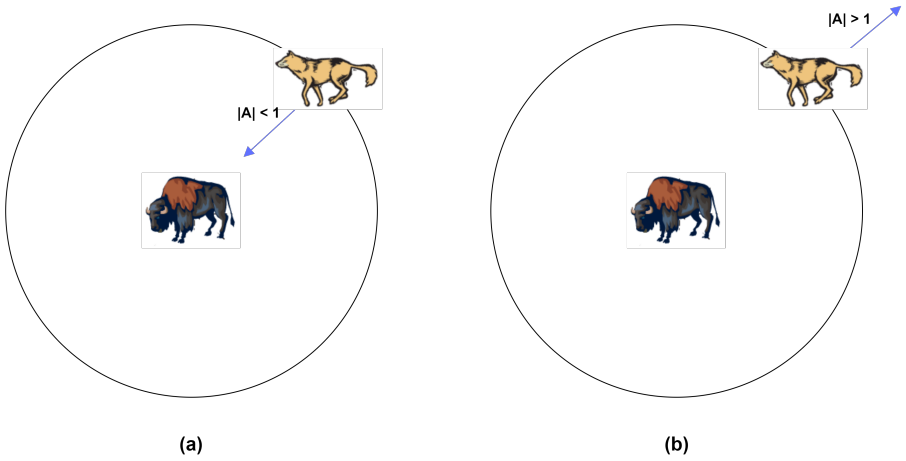

Figure 3.4: (a) Exploitation, (b) Exploration [40]

3.5 Pseudo Code

Previous sections discussed the mathematical modeling of encircling, hunting, and attacking mechanism of grey wolves along with some important components such as a, A, and C. This section presents the pseudo code for GWO algorithm [40].

——*Start*——

1: Initialize a random population of search agents in D-dimensional search space.

2: Initialize a, A and C.

3: **main loop**

4: **for** every search agent

5: Define limits of the search space in each dimension.

6: Evaluate fitness function in D dimensions.

7: Compare fitness score with α, β, and δ score. Update X_α, X_β, and X_δ based on

the current fitness value.

8: Update a, A, and C.

9: Update position of every search agent according to Eq. 3.7.

10: end **for**

11: If end criteria is met (acceptable fitness score or maximum iterations), terminate loop.

12: end **main loop**

13: return X_α

——***End***——

4 Particle Swarm Optimization Algorithm

4.1 Inspiration

Particle Swarm Optimization was proposed by Kennedy (a social psychologist) and Eberhart (an electrical engineer) in 1995 [37]. The idea was to produce computational intelligence by simulating collective and social intelligence of creatures, as inspired by the social interactions in birds flocks and fish schools. These swarms are characterized by complex synchronous dynamic movements such as dispersing, suddenly changing direction, and reassembling, despite having only local knowledge of each individual. The underlying reason for the synchrony was considered to be the birds' tendency to keep a distance between each other. The displacement of each individual is determined by exploiting their local knowledge and memory. Swarm behaviors of avoiding crowded flock regions, moving in the same direction, and staying close to the flock are used by PSO to simulate separation, alignment and cohesion respectively. PSO has been employed for various applications, including tuning power system stabilizers [52], improving battery autonomy in photovoltaic systems [53], and increasing electric vehicle autonomy [55], due to its simplicity and superior performance.

In PSO, a number of search agents, called particles, are randomly placed in the search space of a problem. Each particle, representing a solution to the problem, is characterized by three vectors: position $\vec{x_i}$, velocity $\vec{v_i}$, and previous best position $p\vec{Best}_i$. Each of

the vectors has a dimension equal to the dimension D of the search space. The current position of each particle is evaluated as a solution to the problem in each iteration. If current position produces better results than any previous solution, $p\vec{B}est_i$ is updated with the current position and fitness is stored in a vector called $pBestScore_i$. The movement of each particle in the search space is then determined by using its current and previous best position with the best position obtained in the swarm so far, along with a random deviation. The movement is realized by adding $\vec{v_i}$ to the current position of each particle, and the algorithm moves to the next iteration after movement of all particles. $\vec{v_i}$, effectively seen as a step size, is adjusted in each iteration to allow exploration and exploitation of the search space. The swarm moves eventually as a flock of birds in search of an optimum.

An individual particle cannot solve a problem by itself, therefore interaction is necessary to make progress. Similar to birds flocking, some sort of communication mechanism is required to enable social interaction that lies at the core of problem solving. As discussed earlier, each particle interacts with the neighboring particles and its velocity is influenced by the best position achieved in its neighborhood denoted by $g\vec{B}est$. The individual best and swarm best are stored by each particle in its memory. Over the course of iterations, the velocity of each particle is adjusted to enable its movement around $p\vec{B}est_i$ and $g\vec{B}est$. In short, the movement of each particle is governed by the following three components.

1. Inertia - influence the particle to move along its current direction.

2. Cognitive - influence the particle to move towards the region where previous best is achieved.

3. Social - influence the particle to move towards the region where best is achieved by its neighbors.

The above three forces dictating the movement strategy of a particle are illustrated in Figure 4.1.

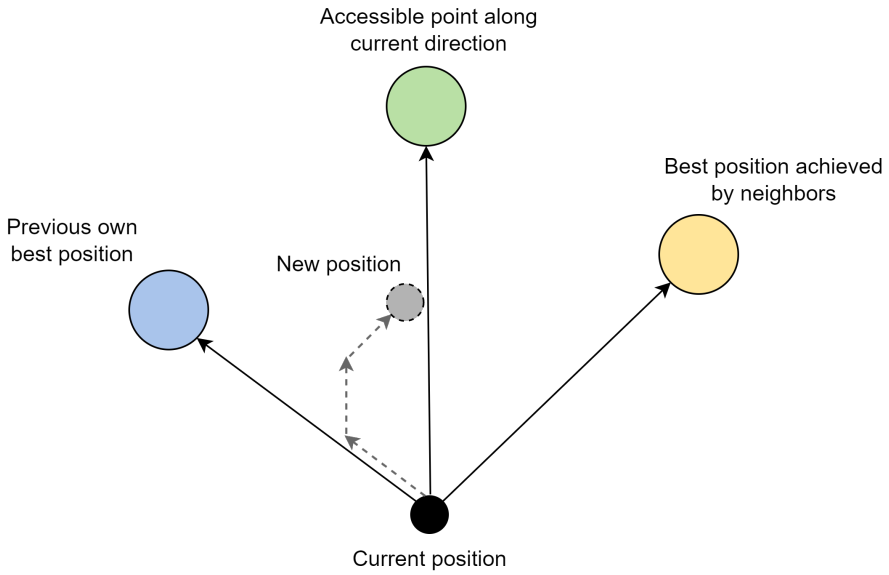

Figure 4.1: Movement of a particle in the search space [53]

4.2 Mathematical Modeling

Each particle of the swarm in a D-dimensional search space has a position vector $\vec{x_i} = (x_{i1}, x_{i2}, ..., x_{iD})$ and a velocity vector $\vec{v_i} = (v_{i1}, v_{i2}, ..., v_{iD})$. The quality of particle position is determined by the fitness value at that position. The best position achieved by a particle i is stored as $p\vec{Best}_i = (pBest_{i1}, pBest_{i2}, ..., pBest_{iD})$ and the best position of any particle in the swarm is denoted by $g\vec{Best} = (gBest_1, gBest_2, ..., gBest_D)$.

The algorithm begins with random initialization of a population of particles in the search

space. At each iteration, the objective function is evaluated for each particle and the best positions are updated. $pBest_i$ and $gBest$ at iteration $t+1$ are given by Equation 4.1 and 4.2.

$$\vec{pBest_i}^{t+1} = \begin{cases} \vec{x_i}^{t+1}, \; if \; Fitness \; > \; \vec{pBest_i}^{t} \\ \\ \vec{pBest_i}^{t}, \; if \; Fitness \; \leq \; \vec{pBest_i}^{t} \end{cases} \tag{4.1}$$

$$\vec{gBest}^{t+1} = \vec{pBest_i}(Max \; Fitness), \; for \; 1 \; \leq \; i \; \leq \; N \tag{4.2}$$

Where N is the swarm size.

After updating the best positions, particles are moved in the search space by adding a velocity term to its current position. A velocity vector is calculated for each particle at iteration $t+1$ according to Equation 4.3. The particles are then moved as per Equation 4.4.

$$\vec{v_i}^{t+1} = w\vec{v_i}^{t} + c_1.r_1^t \otimes (\vec{pBest_i}^{t} - \vec{x_i}^{t}) + c_2.r_2^t \otimes (\vec{gBest}^{t} - \vec{x_i}^{t}) \tag{4.3}$$

$$\vec{x_i}^{t+1} = \vec{x_i}^{t} + \vec{v_i}^{t+1} \tag{4.4}$$

Where w is termed as inertia weight, c_1 and c_2 are two constants known as acceleration coefficients, r_1 and r_2 are random numbers between 0 and 1, and $\otimes$ is element-wise multiplication since the velocity is calculated for every particle in each dimension.

Equation 4.3 is a linear sum of three components, which represent each of the above mentioned forces (inertia, cognitive, and social) that influence the movement of a particle.

"$w\vec{v_i}^{t}$" represents the inertial component that influence the particle to continue movement in the current direction. w controls the amount of the inertial influence.

"$c_1.r_1^t \otimes (p\vec{B}est_i{}^t - \vec{x_i}^t)$" represents the cognitive component, whose magnitude is controlled by c_1.

"$c_2.r_2^t \otimes (g\vec{B}est^t - \vec{x_i}^t)$" represents the social component, whose magnitude is controlled by c_2.

In Eq. 4.3, the value of coefficient of inertia w does not remain constant over the course of iterations. For better performance, researchers have found that w should be set to a higher value initially (e.g. 0.9) to enable exploration of the search space. Its value should be gradually reduced to a lower value to allow exploitation of the search space, where particles converge to a local optima. Starting the algorithm with $w > 1$ will induce instability in the system, which will then require sufficiently minimized value of w to stabilize the swarm.

4.3 Pseudo Code

Previous sections discussed the inspiration and mathematical modeling of PSO. The process for implementing PSO is as follows.

——***Start***——

1: Initialize a population of particles in D-dimensional search space with random positions and velocities.

2: **main loop**

3: **for** every particle

4: Define limits of the search space in each dimension.

5: Evaluate fitness function in D dimensions.

6: Compare current fitness score with $pBestScore_i$. Update $p\vec{B}est_i$ and $pBestScore_i$ if current fitness is better than the previous best.

7: Update $g\vec{B}est$ with the best position in the neighborhood.

8: Update w, calculate velocities, and move the particles according to Eq. 4.3 and 4.4.

9: end **for**

10: If end criteria is met (acceptable fitness score or maximum iterations), terminate loop.

11: **end main loop**

12: return $g\vec{B}est$

——***End***——

5 Design Methodology

This work aims at developing a configurable event detection framework for power systems, which can be tuned to perform in accordance with system requirements. The detection algorithm can be implemented in an automation equipment such as SEL-3555 Real-Time Automation Controller (RTAC) for real-time monitoring of power system events. As discussed in chapter 2, most of the existing work on event detection either requires data from multiple buses, which suffer from communication latency, or have computational complexities. A linear regression-based event detection model was developed by a previous study [22]. The event detection algorithm presented here builds on that model and modifies it to improve its performance. Two swarm intelligence-based optimization algorithms - GWO and PSO - are then applied to optimize its parameters in accordance with a human validation file, and enable it to perform as desired by the system operator. The performance of both optimization algorithms are compared.

5.1 Event Detection Algorithm

The event detection algorithm uses a least-squares linear regression method. It is a statistical procedure widely used for many application including data forecasting, time series data analysis, and determining causal effect dependencies between variables. This method

expresses the relationship between dependent and independent variables by generating a line of best fit. This regression line is placed by minimizing the square of vertical distance from the data points, also known as variance.

$$Y = a + bX + u \tag{5.1}$$

Where, Y represents dependent variable, X represents independent variable, a is intercept, b is slope, and u is regression residual.

In our case, the dependent variable is frequency and the independent variable is time. Since we are interested in determining a smoothed ROCOF, referred to as the slew rate, we just take the slope of the regression line for our calculation, given by:

$$Slope(Slewrate) = \frac{N\sum(xy) - \sum x \sum y}{N\sum(x^2) - (\sum x)^2} \tag{5.2}$$

Where, x is independent variable, y is dependent variable, and N is the number of data points.

Slew rate is preferred over ROCOF for event detection. ROCOF data calculated by PMUs undergo continuous variation and are prone to noise. Fluctuations in frequency increase with the increase in sampling rate of PMUs, which amplifies the noise level. Derivative of frequency data, (ROCOF), does not provide a smooth estimation of frequency trend and suffers from fluctuations and uncertain values. It requires a filter to smooth out the waveform, which increases computational complexity and introduces additional time delay. Linear regression provides a better estimation of frequency deviation and has superior performance over ROCOF in case of noisy data. Deviation in slew as an indication of frequency instability is shown in Figure 5.1, lower plot.

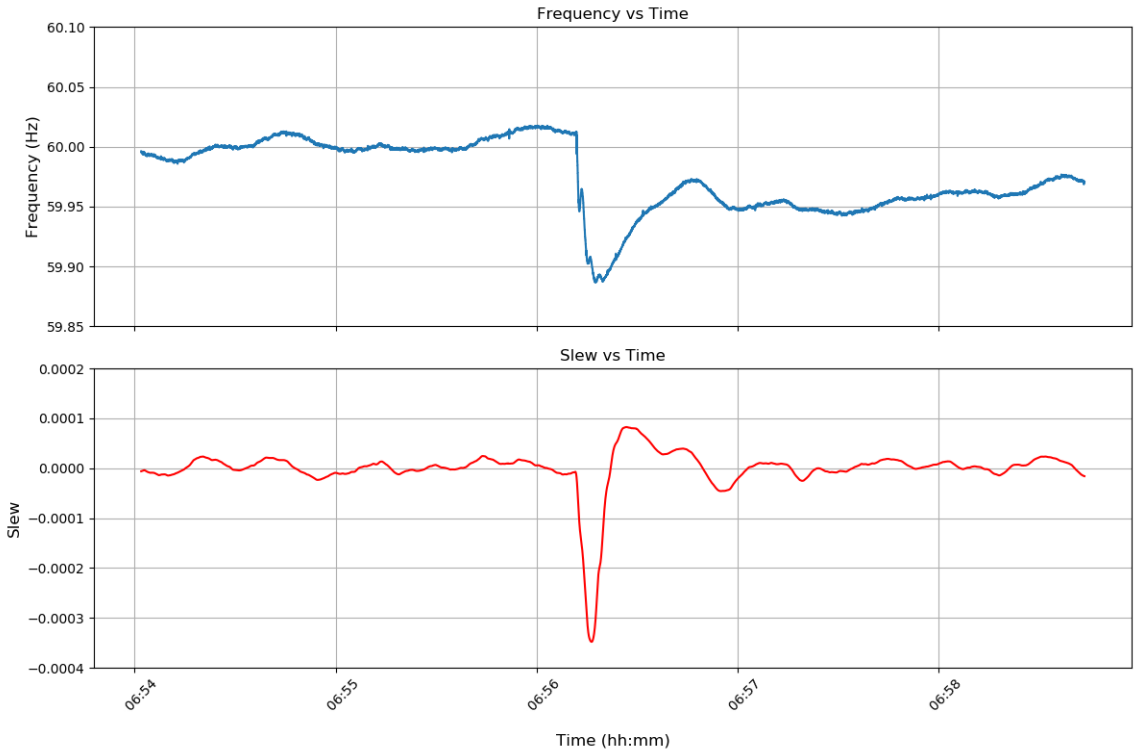

Figure 5.1: Frequency instability (blue) and the corresponding drop in slew rate (red)

5.2 Detection Algorithm Parameters

The detection algorithm calculates slew rate of frequency data obtained from PMUs over a moving window using Eq 5.2. Under normal system conditions, frequency is within a permissible band around a nominal value of 50 or 60 Hz and slew rate is almost constant. In case of an event, frequency changes abruptly and thus slew rate experiences a sudden rise or fall. Based on the this sudden change in slew rate, the detection algorithm identifies an event. The algorithm has five tunable parameters, which dictate its performance and are adjusted using the swarm intelligence optimization algorithms.

Parameter 1: Window Size

Slew rate of the frequency is calculated over a moving window using Eq 5.2. If N is the total number of data samples in the sliding window, then the sampling time over which slew rate is calculated is given by dividing N over the sampling rate of PMU (30 samples/sec in our case). The sliding window moves forward in steps of one sample. Selection of an appropriate windows size is important as it affects the detection speed and immunity to noise. A large window is immune to noise and can produce a smooth trend of frequency but has a slower detection speed.

Let x_1, x_2, x_3...x_N represent the timestamps of PMU frequency measurements and y_1, y_2, y_3...y_N represent frequency measurements, then the slew rate λ calculated over a window of length N is given by:

$$\lambda_k = \frac{\sum_{i=1}^{N} x_i y_i - \sum_{i=1}^{N} x_i \sum_{i=1}^{N} y_i}{\sum_{i=1}^{N} x_i^2 - (\sum_{i=1}^{N} x_i)^2} \tag{5.3}$$

Parameter 2: Point Separation

The slew rate calculated over a window represents a slope value. With every new frequency data point recorded by the PMU, the window moves forward in one sample step and calculates slew rate (λ_k) for the new window. To detect a rise or fall, two slew points are compared with each other. These compared values are separated at a distance defined by this parameter. Although adjacent slew points can be compared, testing showed that a gap of more than one improves the detection speed.

Let λ_1, λ_2, λ_3...λ_n represent slew calculated over each sliding window, then slew vector is given by:

$$\vec{\lambda} = [\lambda_1, \lambda_2, \lambda_3 ... \lambda_n] \tag{5.4}$$

The difference of slew is calculated between two values that are separated by a number of points defined by n_{ps}.

$$D_i = |\lambda_{i+n_{ps}} - \lambda_i| \tag{5.5}$$

Calculation of λ and D are performed in run time. In offline testing, frequency measurements are read from archived files, whereas in online testing, streaming PMU data is used.

Parameter 3: Slew Difference Threshold

The magnitude difference between any two slew points is compared with a threshold (Th_{sd}) to detect sudden rise or fall in a frequency trend. This threshold will only be exceeded in case of an event.

$$if\ D_{slew,i}\ >\ Th_{sd}$$

$$frequency\ deviation\ has\ started$$

Parameter 4: Series Over

Slew threshold can not be used as the only parameter for event detection because every frequency instability does not lead to an event. Therefore, series over value is incremented with every consecutive slew difference that exceeds slew threshold.

$$if\ D_{slew,i}\ >\ Th_{sd}$$

$$Series\ Over = Series\ Over + 1$$

The algorithm declares an event when series over value exceeds series over threshold (Th_{so}), which implies that there is a constant rise or fall in frequency trend. This variable is reset before it exceeds its threshold if frequency goes back to normal. Thus, it also guards against false detection signals.

Parameter 5: Event Threshold

Unlike minor frequency instabilities, frequency changes rapidly during an event, which is reflected by the steep slew rate curve. The slew deviation from the normal value achieved in a given time in case of an event is greater than in case of a non-event frequency deviation. Event threshold (Th_{ev}) is another parameter which checks if, after series over threshold is exceeded, slew deviation has reached a certain level indicative of an event. This parameter provides an additional level of scrutiny for event detection. Testing showed that adding this parameter in the algorithm improves its performance.

$$if\ Series\ Over\ >\ Th_{so}$$

$$check\ for\ Th_{ev}$$

5.3 Expert Evaluation

This section discusses the human validation file that is created by presenting a set of frequency plots to industry and academic experts, and recording their assessment. This forms the basis for optimization of algorithm parameters in order to enable it to perform according to the system requirements. The optimization algorithm takes this human validation file

as input along with the original frequency data files to optimize the detection algorithm parameters accordingly.

5.3.1 Synchrophasor Data and Frequency Archive

The Power Engineering Lab at Portland State University maintains a data archive with almost three years of PMU data. The data archive has voltage magnitude and phase, current magnitude and phase, and frequency measurements against timestamps saved in comma separated values (csv) format. It has a variety of events and normal frequency data files. A set of frequency data files from the archive was compiled, containing events, non-events, and quasi-events to be used as a reference for testing and development of the detection algorithm. For this purpose, event date and time information provided by Portland General Electric (PGE) was used as reference to extract some of the frequency files from the archive that corresponded to the events as declared by Salem Smart Power Center and NERC authorities.

5.3.2 Expert Evaluation Decision Rules

The set of test files extracted from the archive was presented to a group of experts for evaluation. For improved performance of the algorithm, the group of experts can be chosen as industry or academic professionals with a relevant background. Moreover, a set of decision rules can be defined for declaring the file as either an event or not. This ensures a consistency in evaluation pattern and avoids ambiguous cases. The evaluation was carried out using an online survey where each frequency and its slew rate plot from the set of test files was presented to the experts. Their evaluation for each of the candidate plots

was recorded. The survey considers the expertise level of every participant to weigh their assessments. The experts' opinions for each candidate plot were recorded as either under-frequency event, over-frequency event, or non-event. The survey then produces a human validation file in csv format containing file name, experts names, and their assessment. A final evaluation of the frequency file was then provided using weighted assessment of the experts. A sample human validation file is tabulated in Table 5.1.

Table 5.1: A sample human validation file containing file names, experts' assessments, and final declarations

Name	Expert 1	Expert 2	Expert 3	Is_event
2019-08-02-15-34_11.csv	Over frequency event	Over frequency event	Over frequency event	True
2019-10-03-18-20_5183.csv	Under frequency event	Under frequency event	Under frequency event	True
2019-10-27-03-22_12196.csv	Under frequency event	Under frequency event	Under frequency event	True
2019-12-17-11-48_12529.csv	Under frequency event	Under frequency event	Under frequency event	True
2019-09-05-08-51_7722.csv	Not an event	Not an event	Not an event	False
2019-09-18-10-15_582.csv	Not an event	Not an event	Not an event	False
2019-09-18-22-15_732.csv	Not an event	Not an event	Not an event	False
2019-09-29-01-12_3769.csv	Not an event	Not an event	Not an event	False
2019-10-24-01-51_11277.csv	Under frequency event	Under frequency event	Under frequency event	True
2019-10-04-08-10_5356.csv	Not an event	Not an event	Not an event	False
2021-01-16-01-20_538.csv	Not an event	Not an event	Not an event	False
2021-01-16-04-02_554.csv	Not an event	Not an event	Not an event	False
2021-02-08-08-33_40.csv	Not an event	Not an event	Not an event	False
2021-01-13-02-03_116.csv	Under frequency event	Under frequency event	Under frequency event	True
2021-02-08-13-50_3.csv	Not an event	Over frequency event	Over frequency event	True
2021-02-10-16-21_302.csv	Not an event	Not an event	Not an event	False
2021-01-12-09-29_18.csv	Not an event	Under frequency event	Under frequency event	True
2019-09-01-02-27_5418.csv	Not an event	Not an event	Not an event	False
2019-09-01-02-30_5419.csv	Not an event	Not an event	Not an event	False
2019-09-01-02-32_5420.csv	Not an event	Not an event	Not an event	False

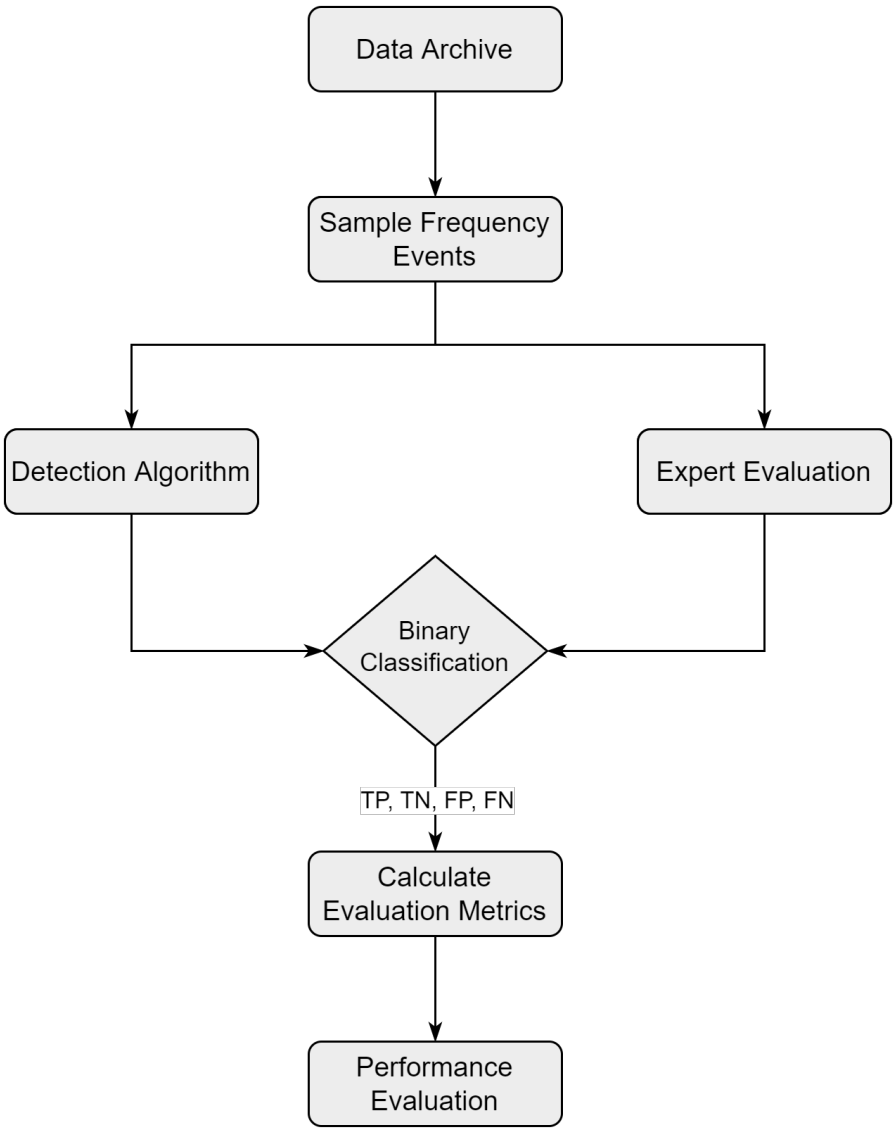

Figure 5.2: Performance evaluation of detection algorithm based on binary classification metrics

5.4 Detection Algorithm Performance Evaluation

Performance of the detection algorithm is evaluated against the assessment of industry experts using a set of performance evaluation metrics [56]. Evaluations metrics are calculated using binary classification metrics. The binary classification metrics and performance evaluations metrics are discussed below. Figure 5.2 shows a flowchart that depicts the

performance evaluation of detection algorithm based on binary classification metrics by comparing algorithm output with expert assessment.

5.4.1 Binary Classification

The event detection algorithm is tested on the same set of frequency files used for the expert evaluation. Results of the algorithm are compared with human validation assessments to form binary classification metrics as explained below.

True Positive (TP): the algorithm correctly identifies an event, as declared by the expert.

True Negative (TN): the algorithm correctly identifies a non-event frequency deviation, as declared by the expert.

False Positive (FP): the algorithm incorrectly identifies an event that is declared as non-event by the expert.

False Negative (FN): the algorithm does not identify an event that is declared an event by the expert.

5.4.2 Evaluation Metrics

Binary classification is used to calculate evaluation metrics to assess the performance of event detection algorithm against experts' evaluation.

- **Accuracy** measures the algorithm's performance in terms of classifying events and

non-events correctly. Value ranges from 0% to 100%, higher value is better.

$$Accuracy = \frac{TP + TN}{SampleSize} \times 100\% \tag{5.6}$$

- **Sensitivity** measures the capability to correctly detect events. Value ranges from 0% to 100%, higher value is better.

$$Sensitivity = \frac{TP}{TP + FN} \times 100\% \tag{5.7}$$

- **Precision** measures how many of the positively identified events are actual events. Value ranges from 0% to 100%, higher value is better.

$$Precision = \frac{TP}{TP + FP} \times 100\% \tag{5.8}$$

- **Specificity** measures the ability of the algorithm to differentiate between event and non-event and correctly identify normal frequency deviation that does not indicate an event. Value ranges from 0% to 100%, higher value is better.

$$Specificity = \frac{TN}{TN + FP} \times 100\% \tag{5.9}$$

- **False Discovery Rate** measures the tendency to falsely identify an event. It is equivalent to $1 - Precision$. Unlike $Accuracy, Sensitivity, Precision, and\ Specificity$, a lower value of FDR is desirable.

$$FDR = \frac{FP}{TP + FP} \times 100\% \tag{5.10}$$

5.5 Frequency Response Test Station

Once a detection algorithm is developed and tested offline using archived PMU data, the next step is to verify its performance for real-time event detection. The Power Engineering Lab at PSU has a Frequency Response Test Station that provides real-time event detection and testing capabilities. Figure 5.3 shows a representation of the Frequency Response Test Station. The test station comprises two SEL-351 PMUs, an SEL-3555 Real-time Automation Controller (RTAC), an NHR-9410 Grid Simulator, and an SEL-2407 GPS clock with antenna. PMUs are the most sophisticated time-synchronized tool used in power systems for advance situational awareness applications. The development in computer technology and deployment of GPS have played a major role in the advancement of PMU technology. A thorough knowledge of the system dynamics has been made possible by the growing deployment of PMUs. In recent years, PMUs have been widely used for detection of power system events due to their capability of recording GPS-synchronized synchrophasors at high sampling rates.

PMU-1 is connected to 120/208 V power supply to monitor and record electrical quantities and save it in the Power Engineering Lab data archive. PMU-2 is connected to the Grid Simulator to take frequency measurements during testing of the detection algorithm. The Grid Simulator has four-quadrant power transfer capabilities and can be used to replicate any frequency event. This provides a sophisticated tool to simulate different power system conditions for detection algorithm testing. The RTAC can be programmed with an event detection algorithm to evaluate its performance in real-time. The Frequency Response Test

Station provides an advanced system to test and evaluate event detection algorithms for historical events without the need to wait for actual events to occur on the grid.

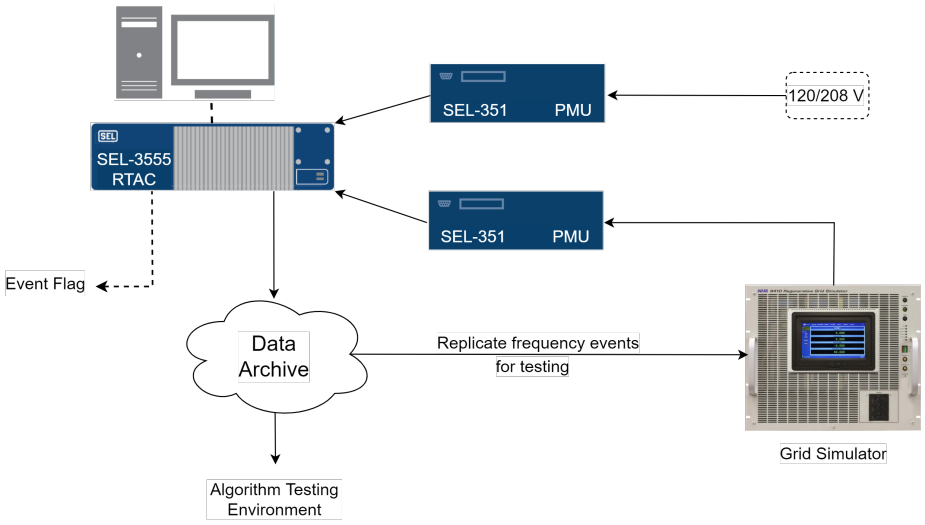

Figure 5.3: PSU real-time Frequency Response Test Station

5.6 Implementation of Optimization Algorithms

5.6.1 Objective Function

For any optimization algorithm, the problem needs to be expressed mathematically, which is referred to as an objective function. The optimization process can then be cast into a maximization or minimization problem, wherein the objective function needs to be maximized or minimized respectively. For this work, the optimization is applied to the detection algorithm to improve its performance by adjusting its five tunable parameters, which can be evaluated with the performance evaluation metrics discussed in Subsection 5.4.2. Thus the objective function for this work can be formulated using a sum of evaluation metrics, which makes

the optimization process a maximization problem to maximize the performance evaluation metrics. Only four evaluation metrics are included in the objective function i.e. Accuracy, Sensitivity, Precision, and Specificity. For the sake of simplicity, False Discovery Rate is not included in the objective function. Apart from simplicity, another reason is that the value of FDR will automatically decline if Precision increases. Objective function is given by Eq. 5.11.

$$Max(Accuracy + Sensitivity + Precision + Specificity) \tag{5.11}$$

Where the evaluation metrics are defined by Eq. 5.6-5.9. The evaluation metrics are affected by the parameters of the detection algorithm. Variation in any parameter is reflected in the performance. By maximizing the objective function (evaluation metrics), the optimization algorithm basically optimizes the parameters of the detection algorithm that will improve its performance. Since each of the evaluation metrics measures the algorithm's ability in terms of TP, FP, TN, and FN, different BAs might focus on different evaluation metric as per their requirements. For instance, a BA may need to maximize $Precision$ and $Specificity$ to reduce the number of FP and prevent unwarranted triggering of frequency response assets. Therefore, objective function can be formulated using a weighted sum of the four evaluation metrics with the highest weight assigned to the evaluation metric of interest for a specific BA.

5.6.2 GWO Algorithm Development

1. The optimization algorithm starts by initializing a random population of search agents. The number of search agents and iterations are set. Dimension of the optimization

problem is set to the number of variables on which the objective function depends. Since there are five parameters to optimize as discussed in Section 5.2, the dimension is set to 5. A boundary of the search space is defined. A position matrix of dimension $N_w \times N_d$ is generated, where N_w represents the number of search agents (wolves) and N_d represents dimension of the problem (5 in this case). The position of each search agent in the position matrix is defined by five parameters to be optimized. The search agent with the best position, i.e. set of parameters, will produce a high fitness score as described by the objective function, and will be designated as the α search agent. The second best and the third best will be designated as β and δ respectively.

2. The main loop starts with checking the position of each search agent against the boundary defined for each parameter in step one. If any of the optimized parameters violates the set boundary limits, it is rounded off to the corresponding upper or lower limit. This is an important step especially for optimizing window size since the event detection speed depends on the window size. By changing the search space boundary for window size, an optimal trade-off can be achieved between detection speed and accuracy.

3. The fitness value for each search agent with a position vector containing five parameters is calculated using the objective function. Based on the fitness value, the three best search agents are identified.

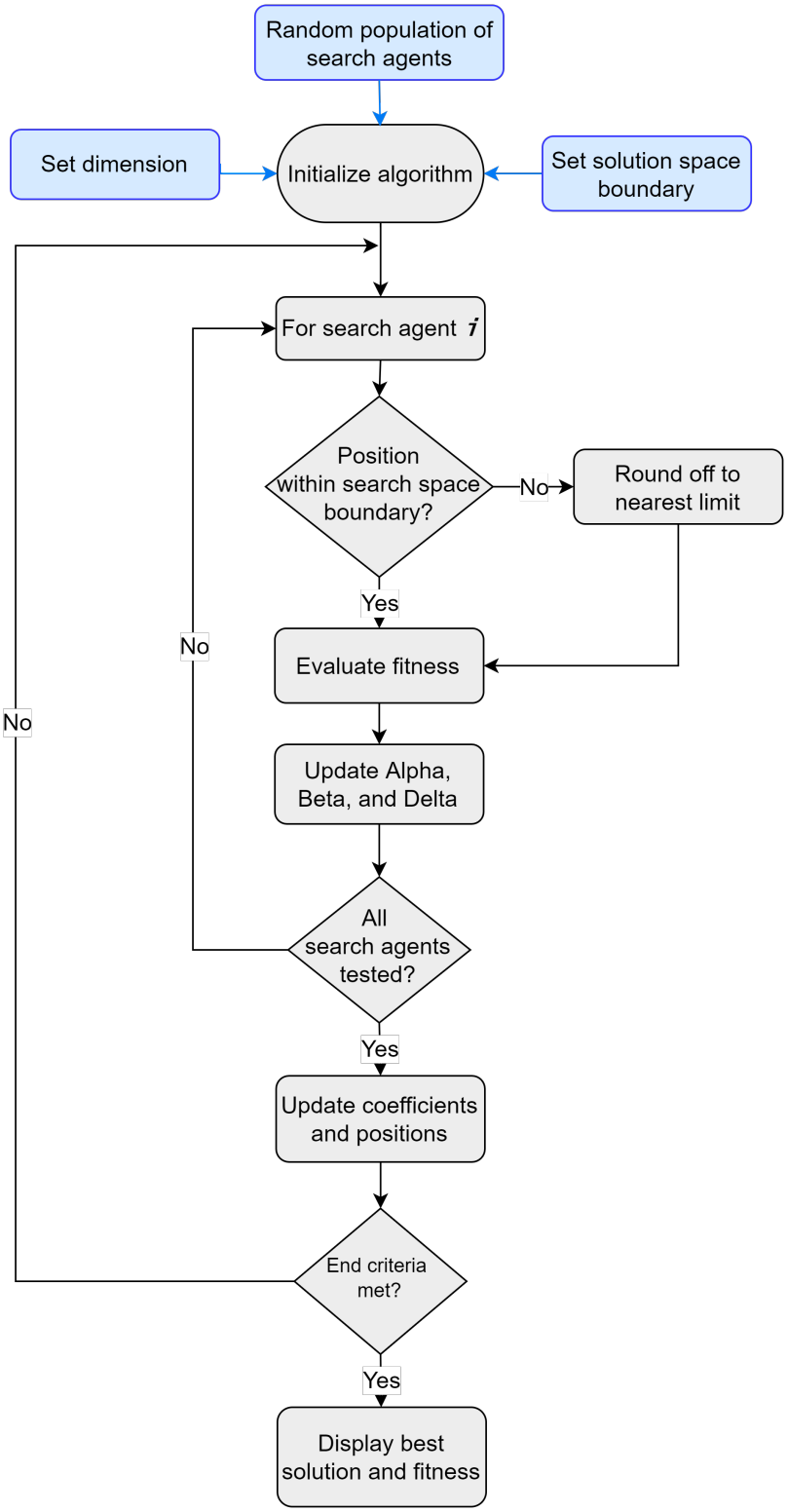

Figure 5.4: GWO algorithm flow diagram

4. After labeling the three best solutions, the value of GWO parameter a is calculated in every iteration. This is an important step. During the course of iterations, a decreases linearly from 2 to 0. This parameter dictates the value of GWO coefficient vector $\vec{A}$ that simulates the divergence of grey wolves in the search space and convergence to find a better prey.

5. In the next step, the value of GWO coefficients A and C are calculated for every search agent in each dimension (for each optimized parameter) using Eq. 3.3 and 3.4. The position of each search agent is updated using Eq. 3.5, 3.6, and 3.7. In every iteration, α, β, and δ update their positions according to the prey and the remaining search agents update their position according to α, β, and δ.

6. With the updated position matrices, the next iteration begins and the loop continues until an end criteria is met or the loop ends. At the end of the optimization process, the position matrix of the search agent α represents the optimized parameters of the detection algorithm. Figure 5.4 describes the GWO process with a flow diagram.

5.6.3 PSO Algorithm Development

The flow of PSO algorithm is similar to that of GWO algorithm. Both the algorithms start with a random population of search agents and evaluate objective function for each search agent. Omitting the details already discussed for GWO, the process of PSO algorithm is described below.

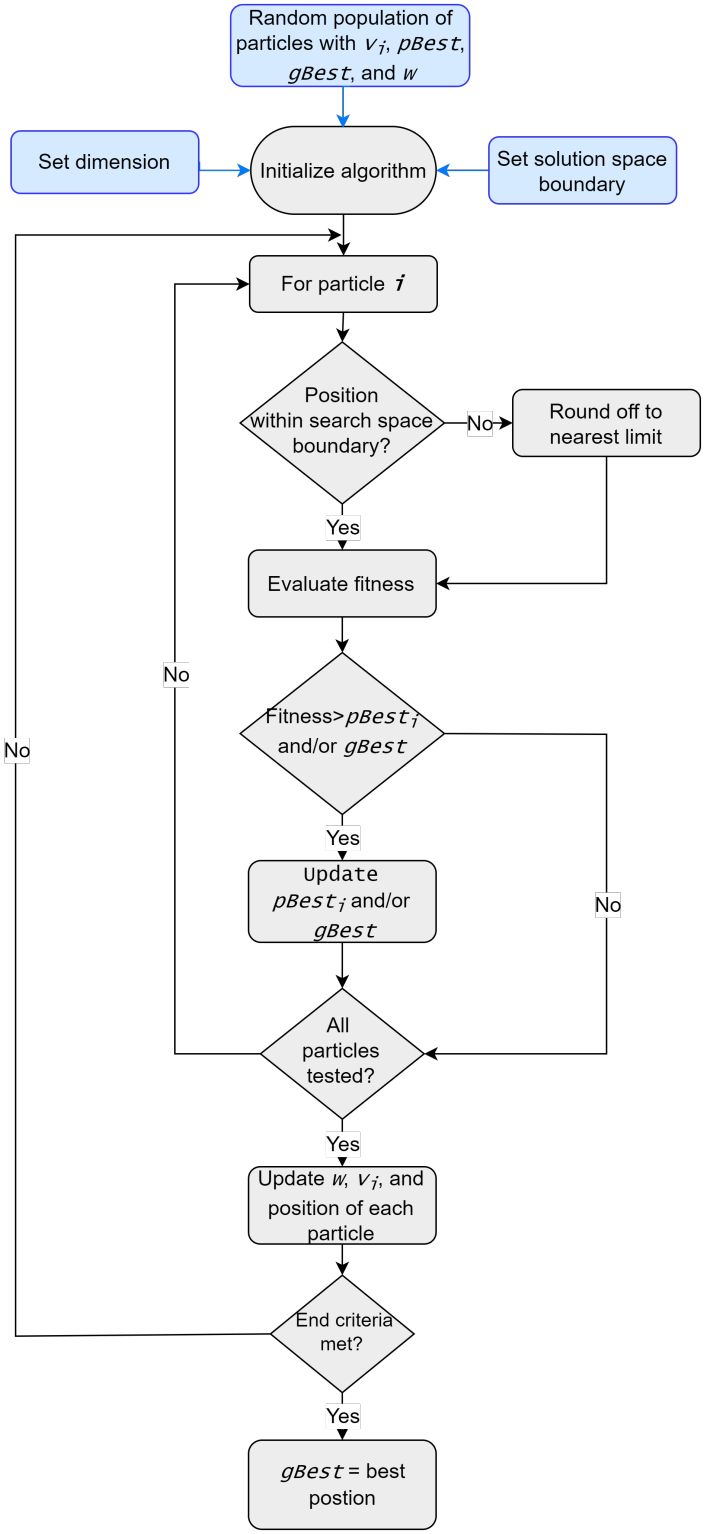

Figure 5.5: PSO algorithm flow diagram

1. The algorithm begins with a population of particles in a D-dimensional (D = 5 in this case) search space with random positions and velocities. Both the position matrix and velocity matrix has dimensions $N_p \times N_d$, where N_p represents number of particles and N_d represents dimension of the problem.

2. The main loop starts with defining a boundary for each dimension of the search space and rounding off each parameter that exceeds the boundary to the nearest limit.

3. The objective function is evaluated for each particle having a 5-dimensional position vector. If the current fitness of a particle i is greater than its previous best, update the previous best fitness $pBestScore_i$ to the current fitness and previous best position $\vec{pBest}_i$ to the current position. Similarly, if the current fitness is greater than the previous best achieved by any particle in the swarm, update the swarm best fitness $gBestScore$ to the current fitness and swarm best position $\vec{gBest}$ to the current position.

4. Update coefficient of inertia w, calculate velocity for each particle according to Eq. 4.3, and move each particle in the search space according to Eq. 4.4. The setting of the PSO simulation parameters i.e. w, c_1, and c_2, is well-discussed in the literature [57, 58]. High values of c_1 and c_2, and a low value of w will discourage exploration and cause premature convergence of the particles. To realize a balance between exploration and exploitation, both c_1 and c_2 are set to 2, and w is linearly decreased from a maximum set value (0.9) to a minimum set value (0.2) over the course of iterations [59].

5. With the updated position of the particles, the algorithm moves to the next iteration and the loop continues to evaluate the new positions until an end criteria is met. At the end of the optimization process, $gBestScore$ represents the best fitness and $\vec{gBest}$ represents the best position (5 optimized parameters). Figure 5.5 describes the PSO process with a flow diagram.

5.6.4 Challenge: Processing Time

Each frequency file in the data archive contains 18,000 measurements. For better performance of the detection algorithm, the sample space chosen for parameter optimization should be large containing events, non-events, and quasi-events. As an example, a sample size with 20 frequency files having 18,000 measurements each will result in a data set of 360,000 measurements. Processing this much data over many iterations for optimization takes an enormous amount of time. The processing time increases dramatically with the number of iterations and sample size. Several approaches were adopted to address this challenge. Since the frequency archive files are in csv format, instead of reading frequency measurements from those files in each iteration, all the frequency measurements were stored in arrays to be readily available and promptly accessed for processing. Similarly, in each iteration, the optimization algorithm processes all data in each frequency file to look for events and compare them with the human validation file in order to calculate fitness. The algorithm was forced to move to the next file when an event is detected in a specific file, by calculating slew in run time and checking for events. This helped reduce the processing time associated with calculating slew over a complete file having 18,000 measurements when an

event can be in the beginning or middle of the file. The main two approaches that considerably reduced the optimization processing time are discussed below in Subsection 5.6.5 and 5.6.6.

5.6.5 Data Curtailment

Events do no happen frequently in power systems and thus most of the frequency measurements in the frequency archive files represent normal frequency. One way to reduce the processing time was to remove the redundant frequency data representing normal frequency and ultimately diminish the data size. This techniques was implemented using the idea that a sudden rise or drop of slew indicates a frequency disturbance. In the beginning of the optimization algorithm, slew rate was calculated for all the files, and all the data not representing any considerable slew change were removed from the files. A step-by-step procedure of data curtailment is explained below.

1. Read data from all frequency files.
2. Calculate slew rate for all files.
3. Identify maximum and minimum slew value in a file along with the index.
4. Compare magnitude of the maximum value with the minimum value and identify the greater one. This shows if the file represent an under-frequency disturbance or an over-frequency disturbance.
5. Label index of the greater value as x.

6. Keep l_1 number of measurements before x and disregard the data before $x - l_1$. If $x - l_1 < 0$, set $x - l_1 = 0$. Selection of l_1 is critical to avoid degraded performance.

7. Keep l_2 number of measurements before x and disregard the data before $x + l_2$. If $x + l_2 > file\ size$, set $x + l_2 = file\ size$. Selection of l_2 is critical to avoid degraded performance.

8. Repeat the process for each file.

Adopting this approach caused a significant reduction in the processing time, almost 85%. There is a minimum limit for l_1 and l_2, below which performance will deteriorate.

5.6.6 Memoization

Memoization is a programming optimization technique used in dynamic programming to accelerate performance and reduce computation time. It caches the results of recursive functions and returns the stored value when needed later. It reduces the time complexities from exponential to polynomial by avoiding re-computation of results. Memoization is mostly applied when a costly function is executed continually, possibly with the same inputs.

Since both GWO and PSO algorithms involve random numbers with a possibility that the same position matrix (detection parameters) may be passed to the fitness function repeatedly, memoization is implemented to avoid repeated calculations. Every time the fitness function is called, it checks the memory for the same arguments. If a fitness value already exists for the same inputs, it returns that value. If not, it computes the fitness and stores it against the position matrix to be used later.

6 Results & Discussion

A Python development environment was used to implement and test the event detection and optimization algorithms. A sample set of 50 frequency files consisting of events, non-events, and quasi-events, recorded at 30 samples per second, was used from the PMU data archive to run simulations and validate the performance of the GWO and PSO-based tuning algorithms. The human validation file and the corresponding frequency files were provided to both optimization algorithms. The obtained optimized coefficients were validated by evaluating the performance of the detection algorithm using those coefficients. Two versions of the human validation file were used for the sample set - one with evaluation of both experts and non-experts, and one with evaluation of only experts. The effect of the expert evaluation process and consistency in event definition was highlighted by comparing the results obtained from both versions of the human validation. The detection speed achieved using optimized parameters is fast enough to trigger frequency control assets within a short time after the onset of an event.

6.1 Performance Evaluation

The sample set consists of 50 files with 21 events and 29 non-events. The upper and lower boundary of the solution space for each of the five parameters is given in Table 6.1. Selection of an appropriate window size is critical to achieve a trade-off between performance and

detection speed. A large window produces a smooth slew rate and improves performance but it reduces detection speed. Therefore, the upper bound for the slew window is restricted to 8 seconds to achieve improved performance with reasonable detection speed. The corresponding human validation file for the sample set was created from the evaluation of three experts. Table 6.2 shows the adjusted parameters using GWO and PSO.

Table 6.1: Boundary of the search space for the five dimensional optimization problem

	Window (samples)	**Point Separation**	**Slew Diff**	**Series Over**	**Event Threshold**
Upper Bound	250	30	0.0002	30	0.0001
Lower Bound	100	3	1×10^{-7}	3	1×10^{-6}

Table 6.2: Adjusted parameters using GWO and PSO

	Window (samples)	**Point Separation**	**Slew Diff**	**Series Over**	**Event Threshold**
GWO	216	3	0.000003	13	0.0000378
PSO	150	25	0.00005966	20	0.0000001

A comparison of the best solutions obtained using GWO and PSO along with the simulation parameters is presented in Table 6.3. The maximum possible fitness value is 400, as given by Eq 5.11.

Table 6.3: Comparison of best solutions obtained with GWO and PSO

Algorithm	**Iterations**	**Search Agents**	**Fitness**	**Sample Set Size**	**No. of Events**	**No. of non-events**
GWO	50	10	383	50	21	29
PSO	50	10	359	50	21	29

GWO outperformed PSO. The maximum fitness achieved by PSO was less than that achieved by GWO. The convergence curves for both GWO and PSO are shown in Figure 6.1.

The optimized parameters were used in the detection algorithm and tested for the 50 files. GWO-optimized parameters produced superior results. Out of 21 events, 20 were detected correctly. Since occurrence of events in power systems is a rare phenomenon, one of the most important features of a detection algorithm is to identify normal frequency deviations and not issue false detection signals. *Precision* and *Specificity*, which account for false positives, are the best metrics for considering issuance of false detections. The proposed algorithm was capable of differentiating between events and minor frequency deviations. Performance evaluation metrics and binary classification are given in Table 6.4.

Table 6.4: Performance evaluation metrics obtained using GWO and PSO-based parameter adjustment

Algorithm	**Accuracy (%)**	**Sensitivity (%)**	**Precision (%)**	**Specificity (%)**	**FDR (%)**	**TP**	**FP**	**FN**	**TN**
GWO	96	95	95	97	5	20	1	1	28
PSO	92	85	95	97	5	18	1	3	28

An *Accuracy* of more than 95% was achieved with GWO-optimized parameters, as presented in Table 6.4. Considering *Precision*, a value of 95% implies that one out of 29 non-events was falsely given as an event. This can happen if the frequency deviation is more severe like an event. The frequency file that was falsely detected as an event is shown in Fig 6.2. The frequency increases from 59.965 Hz to 60.01 Hz (0.045 Hz) in 9 seconds. The abrupt deviation in frequency gives rise to a spike in the slew rate, although the frequency is oscillating within permissible band. Some utilities might be interested in detecting such abrupt deviations, depending on the system conditions.

For optimization, initially a large solution space was chosen with increased number of iterations. The upper and lower bounds for all parameters were found to be conservative

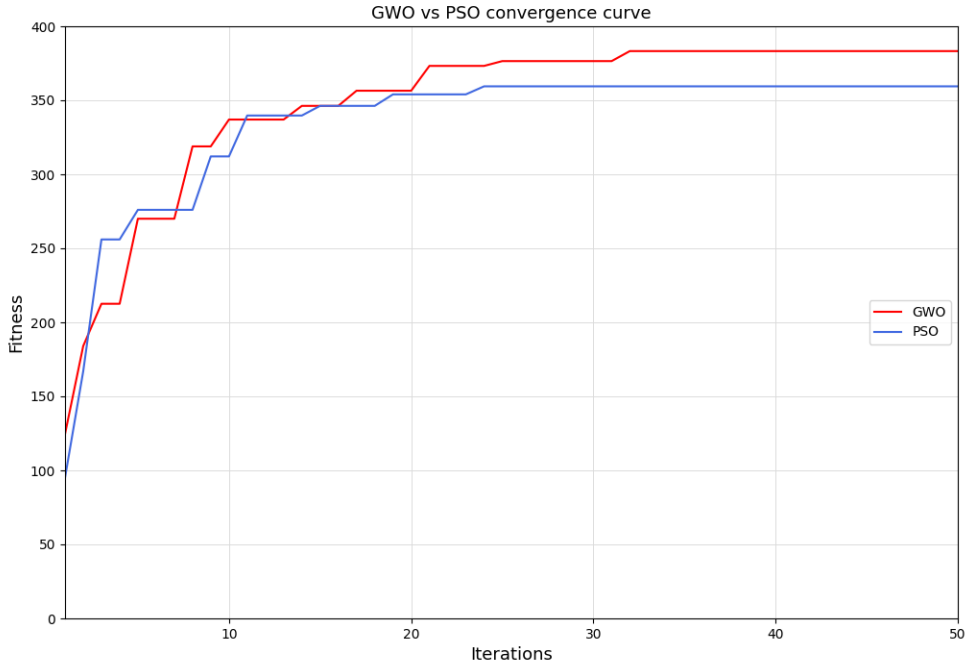

Figure 6.1: Convergence curve for GWO and PSO for the sample set with 50 files. GWO achieved higher fitness value.

for different types of events. Therefore, the solution space limits were reduced to improve convergence speed with fewer iterations without compromising performance. With a relatively smaller solution space, the optimization algorithms were able to produce the same results in a considerably shorter time.

GWO performed better than PSO for this problem as it was able to produce better results for a sample set where PSO failed to achieve the highest possible fitness. This fact is supported by the No Free Lunch Theorem [60], which proved that no single meta-heuristic can perform effectively for all optimization problems. This particularly means that an algorithm may have superior performance for some set of optimization problems but poor performance for another set of problems.

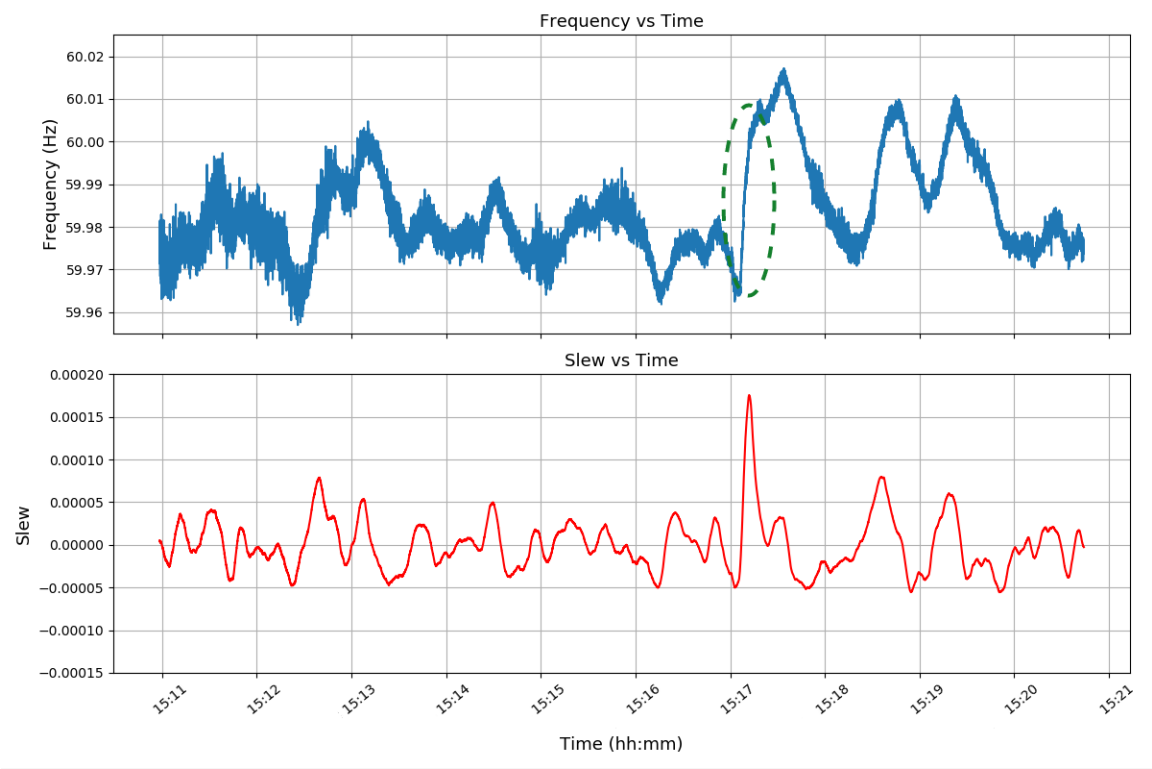

Figure 6.2: An abrupt deviation in frequency giving rise to a spike in the slew rate and false detection signal.

6.2 Event Detection Speed

For determining the speed of the detection algorithm, six random event files were chosen from the sample set. Using the GWO-optimized parameters, the frequency measurement point at which the algorithm identifies the event in an under-frequency event is highlighted in Figure 6.3.

After the fault inception, the number of frequency measurements it takes for the detection algorithm to issue event detection signal for each sample file are presented in Table 6.7. The sample set imported from the PMU data archive was recorded at 30 samples per second. For real-time implementation of detection algorithm in the RTAC, the samples will be taken from streaming PMU data instead of pre-recorded files. Table 6.7 also presents the detection time

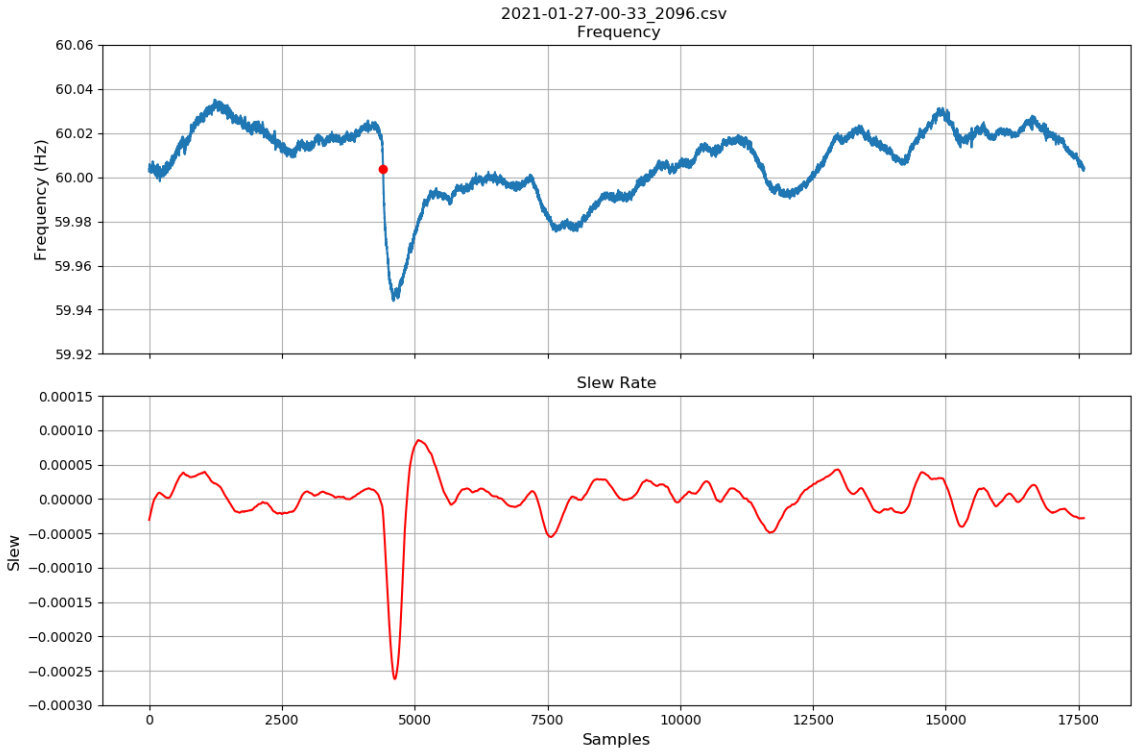

Figure 6.3: The frequency sample (red dot) at which the algorithm detected an event.

Table 6.5: Event detection speed in terms of frequency samples and seconds for files recorded at 30 samples per second.

File	Fault Inception Point	Event Detection Point	Detection Time (no. of. samples)	Detection Time (sec)
1	10502	10525	23	0.76
2	15653	15677	24	0.8
3	6165	6198	33	1.1
4	7060	7097	37	1.23
5	15692	15744	52	1.73
6	6062	6096	34	1.13

in seconds using the data sampling rate. The detection speed varies for different frequency events depending on the slew rate of the frequency event curve. If the frequency curve becomes normal for some time during a decline, the detection algorithm will take longer to declare it as an event.

The sample files were recorded with a sampling rate of 30 samples per second. To determine the detection speed for a higher sampling rate, the frequency data were interpolated to make it look like they were recorded at 60 samples per second. The same six files were subject to linear interpolation in Python. Each interpolated file had twice the number of samples as the original file, i.e., 36,000. Figure 6.4 shows an interpolated file against the original file along with the sample at which event is detected for both files, respectively.

The interpolated files were given to the GWO algorithm to obtain updated parameters. The same fitness value and performance metrics were obtained but the detection parameters derived were different because of an increased sampling rate. Event detection time was then determined for these files using the updated optimized parameters. Table 6.6 presents detection time both in terms of number of samples and seconds for the new files.

Table 6.6: Event detection speed in terms of frequency samples and seconds for interpolated files.

File	**Fault Inception Point**	**Event Detection Point**	**Detection Time (no. of. samples)**	**Detection Time (sec)**	**Improvement (sec)**
1	21004	21040	36	0.6	0.16
2	31312	31346	34	0.56	0.24
3	12330	12375	45	0.75	0.35
4	14124	14165	41	0.68	0.55
5	31384	31417	33	0.55	1.18
6	12124	12174	50	0.83	0.3

The number of samples it takes for the detection algorithm to identify an event remains almost the same even if the data sampling rate is increased. It is justified by the fact that detection algorithm parameters are adjusted according to the new sampling rate and hence the slew difference threshold will be lower for higher sampling rates owing to the temporal

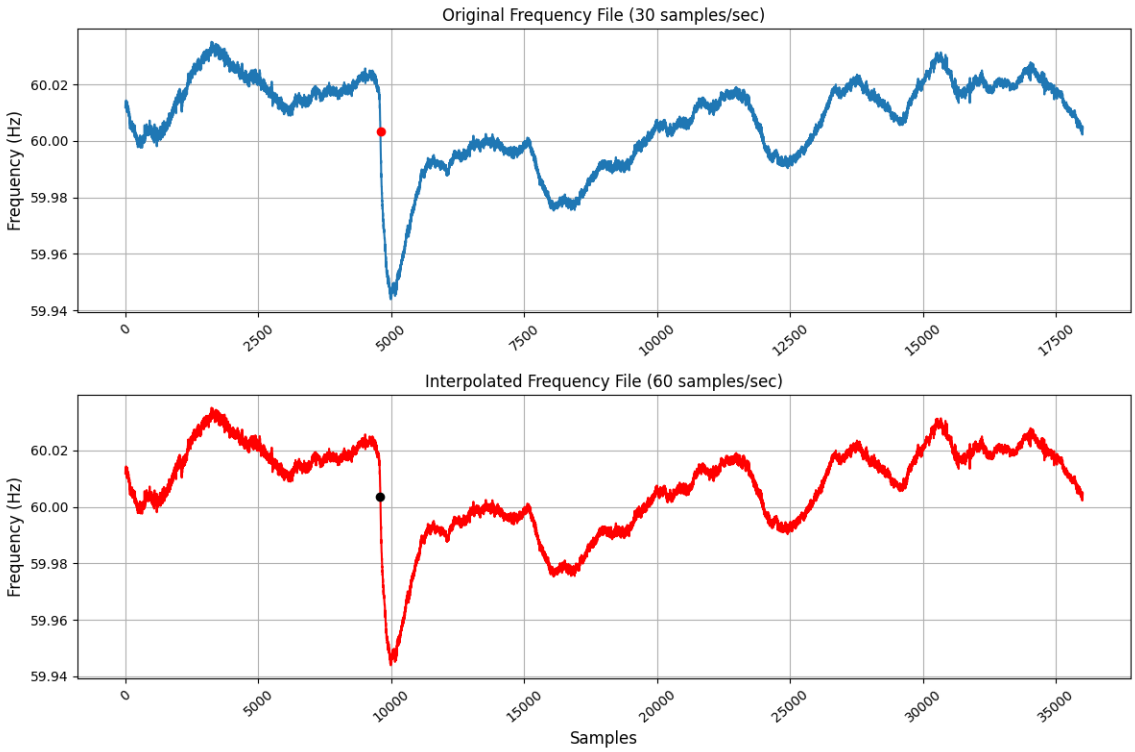

Figure 6.4: An interpolated frequency file against the original file. Red and black dots highlight the sample at which event is detected, respectively.

closeness of samples. The adjustment of parameters is carried out automatically by the GWO algorithm when it processes the highly sampled files. With the advent of synchrophasor technology, PMUs can now record data at up to 120 samples per second. Thus, detection speed can be further improved with higher PMU sampling rates. This improvement in the detection speed with the increase in PMU sampling rate can be utilized in low inertia systems. The decline in system synchronous inertia causes frequency events to exhibit higher ROCOF, thus necessitating higher detection speed.

6.3 Effect of Inconsistent Expert Evaluation

The consistency in event assessment during expert evaluation is of critical importance. Since the optimization algorithms produce optimized parameters to enable the detection algorithm to perform detection according to the human validation, there should be clear decision rules for declaring what quantifies as an event. As the detection algorithm relies on slew rate, which depends on the rate of change of frequency, consistency in event assessment will ensure uniform evaluation pattern and avoid ambiguous cases. If two similar frequency curves are assessed differently by the human experts, the detection algorithm will not be able to differentiate between them and we can expect a false positive or false negative signal. This emphasizes the fact that assessment should be carried out by experts with relevant experience who have better knowledge of power systems. As discussed earlier, the sample set had two versions of human validation created by semi-experts and experts. Table 6.3 and 6.4 present the results for expert evaluation. The performance of optimization and detection algorithms for semi-expert evaluation is discussed below. GWO has been used to identify the effect of inconsistent expert evaluation because it performed better than PSO.

The same sample set with 50 files was processed by the GWO algorithm with a new human validation file containing assessment of semi-experts. In the new human validation file, 29 files were assessed as events and 21 as non-events by the semi-experts. Due to inconsistency in the event assessment process, the maximum fitness achieved was 345 as presented in Table 6.7.

The new adjusted parameters were used in the detection algorithm and tested. Table 6.8

Table 6.7: Fitness value achieved for the same sample set with semi-expert event assessment.

Iterations	Search Agents	Fitness	Sample set size	No. of Events	No. of non-events
50	10	345	50	29	21

presents the performance evaluation metrics and binary classification. The performance of detection algorithm has been deteriorated due to false detection of two candidate frequency curves. This is reflected in the decline of *Precision* and *Specificity*. False positives need to be avoided as they lead to the unwarranted triggering of frequency response assets.

Table 6.8: Performance evaluation metrics obtained for the same sample set using optimized parameters for semi-expert event assessment.

Accuracy (%)	Sensitivity (%)	Precision (%)	Specificity (%)	FDR (%)	TP	FP	FN	TN
84	79	92	90	8	23	2	6	19

Such an ambiguity caused by the inconsistency in event assessment process is shown in Figure 6.5. Although the change in frequency as well as the rate of change of frequency is almost the same for both the plots, the frequency instability on the left is assessed as an over-frequency event whereas the one on the right is assessed as a non-event by the semi-experts. In such a situation, the optimization algorithm cannot reach a best possible solution that can clearly differentiate between the two cases and hence the fitness value drops.

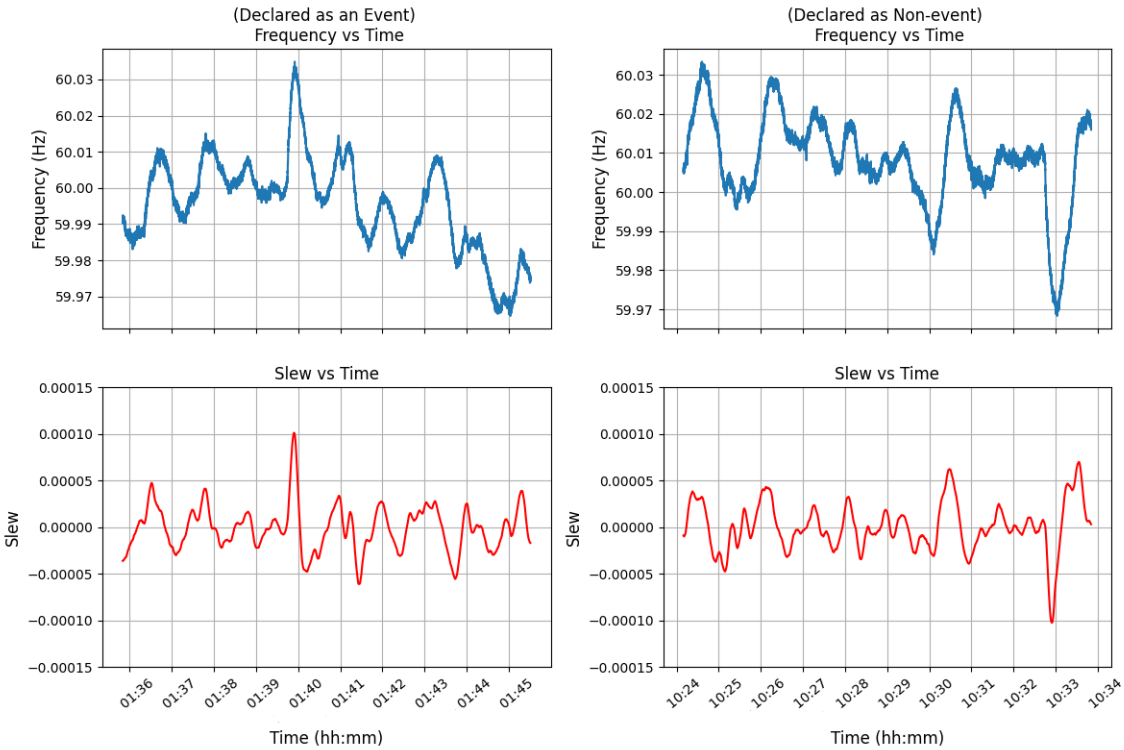

Figure 6.5: Example of an inconsistency in the event assessment process.

7 Conclusion

A constant growth in the installation of renewable energy sources and retirement of large conventional generators have posed a threat to the frequency regulation capabilities of modern electrical grid due to a decline in rotational inertia and primary frequency response. Power systems are required to maintain frequency within a permissible band around the nominal value for secure, reliable, and economic operation. To address the unprecedented challenges that have arisen as a consequence of electrical grid modernization, researchers have focused on the advance computational tools and frequency control mechanisms to improve monitoring of and response to frequency events within modern power systems.

With the development of synchrophasor technology, many existing frequency event detection techniques leverage the high sampling rate of PMUs to enable system operators to have real-time knowledge of power system networks. Automatic event detection in power systems has engaged researchers lately and an extensive work is reported in literature about this topic. However, most of the work reported in the literature need measurement data from multiple buses in the network for reliable operation and cannot differentiate between an actual event and minor deviations.

The definition of an event may vary for different balancing authorities depending upon their critical stability limits. An event detection algorithm should be able to be configured for each balancing authority to match the definition of an event as defined by their experts.

Contemporary event detection algorithms reported in the literature are not able to be configured and operated according to the system conditions. This work presented a configurable event detection method using least-sum-of-squares linear regression whose parameters can be tuned to match the definition of frequency events as specified by experts for a balancing authority. The proposed algorithm has the capability to be implemented in an automation controller to use streaming PMU data for real-time event detection and trigger dispatchable frequency response assets.

Two swarm intelligence-based optimization algorithms, GWO and PSO, were applied to automate the parameters tuning process. Each optimization algorithm uses experts' assessment of frequency events along with the original frequency files as inputs and produces optimized parameters of the detection algorithm as output which are used for detection of events as desired by the experts. Performance of both algorithms were compared and analysed for a sample set containing 50 frequency files. GWO gave better results for the problem and outperformed PSO. The detection algorithm was able to identify frequency events in less than a second after inception of a frequency event. Detection speed can be further improved with higher PMU sampling rates. Effect of inconsistency in the event assessment process by human experts was also demonstrated by using semi-expert assessment, which degraded the performance. Performance of both the detection and optimization algorithms was validated by using recorded PMU data from PSU data archive. The proposed event detection framework facilitates the configuration of the detection algorithm to enable its operation according to the detection requirement of a BA.

Future work may involve the implementation of the proposed event detection algorithm in an automation controller, such as the SEL-3555 Real-time Automation Controller. Using the optimized parameters, the detection algorithm can be validated for real-time event detection with streaming PMU data. The Frequency Response Test Station at Power Engineering Lab offers real-time event detection and testing capabilities. The Grid Simulator enables replication of archived frequency data that can be used by RTAC for real time testing.

Another related work could be using another detection technique, including a wavelet transform or filtering-based approach. Although GWO has proved its efficacy in various problems, future research may focus on a new swarm-intelligence or classical optimization algorithm for parameter adjustment. The performance of the new algorithm can then be compared with this work. Another tangential study may include triggering a dispatchable frequency response asset upon an event detection.

Bibliography

[1] Yuchen Zhang, Yan Xu, and Zhao Yang Dong. Robust ensemble data analytics for incomplete PMU measurements-based power system stability assessment. *IEEE Transactions on Power Systems*, 33(1):1124–1126, January 2018.

[2] Hassan Haes Alhelou. *An Overview of Wide Area Measurement System and Its Application in Modern Power Systems*. Handbook of Research on Smart Power System Operation and Control. IGI Global, 2019.

[3] Nicholas W Miller, Miaolei Shao, Robert D'aquila, Slobodan Pajic, and Kara Clark. Frequency response of the US Eastern Interconnection under conditions of high wind and solar generation. In *Annual IEEE Green Tech. Conf.*, pages 21–28, New Orleans, LA, USA, April 2015.

[4] P. Kundur, Neal J. Balu, and Mark G. Lauby. *Power system stability and control*. The EPRI power system engineering series. 1994.

[5] Hassan Haes Alhelou, Mohamad Esmail Hamedani-Golshan, Takawira Cuthbert Njenda, and Pierluigi Siano. A survey on power system blackout and cascading events: Research motivations and challenges. *Energies*, 12(4):682, 2019.

[6] Mohammad Dreidy, H Mokhlis, and Saad Mekhilef. Inertia response and frequency control techniques for renewable energy sources: A review. *Renewable and Sustainable Energy Rev.*, 69:144–155, 2017.

[7] Seyyed Amir Hosseini, Mohammadreza Toulabi, Alireza Ashouri-Zadeh, and Ali Mohammad Ranjbar. Battery energy storage systems and demand response applied to power system frequency control. *International Journal of Electrical Power & Energy Systems*, 136:107680, March 2022.

[8] Francesco Conte, Bruno Gabriele, Stefano Massucco, Federico Silvestro, Diego Cirio, and Lorenzo Croci. Domestic heat-pump water heater aggregates: a contribution to demand flexibility. In *AEIT International Annual Conference*, pages 1–5, Milan, Italy, October 2021.

[9] Sai Sudharshan Ravi and Muhammad Aziz. Utilization of electric vehicles for vehicle-to-grid services: Progress and perspectives. *Energies*, 15(2):589, 2022.

[10] Ognjen Stanojev, Justin Rüssli-Kueh, Uros Markovic, Petros Aristidou, and Gabriela Hug. Primary frequency control provision by distributed energy resources in active distribution networks. In *IEEE Madrid PowerTech*, pages 1–6, July 2021.

[11] Hongbiao Song and Mladen Kezunovic. A new analysis method for early detection and prevention of cascading events. *Electric Power Systems Research*, 77(8):1132–1142, June 2007.

[12] Mingjian Cui, Jianhui Wang, Jin Tan, Anthony R Florita, and Yingchen Zhang. A novel event detection method using PMU data with high precision. *IEEE Transactions on Power Systems*, 34(1):454–466, 2018.

[13] Ajeet Kumar Singh and Manoj Fozdar. A wavelet-based event detection and location framework for enhanced situational awareness in power system. In *IEEE Annual India Conference*, pages 1–6, 2016.

[14] Haoran Li, Yang Weng, Evangelos Farantatos, and Mahendra Patel. A hybrid machine learning framework for enhancing PMU-based event identification with limited labels. In *International Conference on Smart Grid Synchronized Measurements and Analytics*, pages 1–8. IEEE, 2019.

[15] Boyu Wang, Yan Li, and Jing Yang. LSTM-based quick event detection in power systems. In *IEEE Power & Energy Society General Meeting*, pages 1–5, 2020.

[16] Chao-Yuan Lai, Chih-Wen Liu, and Chia-Cheng Chao. Wide-area frequency security event detection. In *International Conference on High Voltage Engineering and Power Systems*, pages 414–417. IEEE, 2017.

[17] Yongli Zhu, Chengxi Liu, and Kai Sun. Image embedding of PMU data for deep learning towards transient disturbance classification. In *IEEE International Conference on Energy Internet*, pages 169–174, 2018.

[18] Fujia Han, Gareth Taylor, and Maozhen Li. Towards a data driven robust event detection technique for smart grids. In *2018 IEEE Power & Energy Society General Meeting*, pages 1–5, 2018.

[19] Umar Farooq and Robert B. Bass. Frequency event detection and mitigation in power systems: A systematic literature review. *IEEE Access*, 10:61494–61519, 2022.

[20] NERC. Frequency response standard background document, 2012.

[21] Sean Keene, Landon Hanks, and Robert B. Bass. A means for tuning primary frequency event detection algorithms. In *9th IEEE Conference on Technologies for Sustainability*, 2022.

[22] Abdulrahman Alamar, Ebtehal Alenezi, Sayed A. Alhashemi, and Landon Hanks. Primary frequency response detection and analysis system. *Portland State University Capstone Report*, June 2021.

[23] Arup Anshuman, Bijaya Ketan Panigrahi, and Manas Kumar Jena. A novel hybrid algorithm for event detection, localisation and classification. In *9th IEEE International Conference on Power Systems*, pages 1–6, Kharagpur, India, December 2021.

[24] Do-In Kim, Tae Yoon Chun, Sung-Hwa Yoon, Gyul Lee, and Yong-June Shin. Wavelet-based event detection method using PMU data. *IEEE Transactions on Smart Grid*, 8(3):1154–1162, 2017.

[25] Sai Akhil R. Konakalla and Raymond de Callafon. Optimal filtering for grid event detection from real-time synchrophasor data. *Procedia Computer Science*, 80:931–940, June 2016.

[26] Rodrigo Vaz, Guido R. Moraes, Eduardo H. Z. Arruda, Jyvago C. B. S. Terceiro, Antonio F. C. Aquino, Ildemar C. Decker, and Diego Issicaba. Event detection and classification through wavelet-based method in low voltage wide-area monitoring systems. *Int. J. of Elect. Power & Energy Sys.*, 130, September 2021.

[27] Le Xie, Yang Chen, and P. R. Kumar. Dimensionality reduction of synchrophasor data for early event detection: Linearized analysis. *IEEE Transactions on Power Systems*, 29(6):2784–2794, November 2014.

[28] Mark Rafferty, Xueqin Liu, David M. Laverty, and Seán McLoone. Real-time multiple event detection and classification using moving window PCA. *IEEE Transactions on Smart Grid*, 7(5):2537–2548, September 2016.

[29] Ti Xu and Thomas Overbye. Real-time event detection and feature extraction using PMU measurement data. In *IEEE International Conference on Smart Grid Communications*, pages 265–270, November 2015.

[30] Shikhar Pandey, Anurag K. Srivastava, and Brett G. Amidan. A real time event detection, classification and localization using synchrophasor data. *IEEE Transactions on Power Systems*, 35(6):4421–4431, November 2020.

[31] Lipeng Zhu and David J Hill. Spatial-temporal data analysis based event detection in weakly damped power systems. *IEEE Transactions on Smart Grid*, 12(6):5472–5474, November 2021.

[32] Vladimiro Miranda, Pedro A Cardoso, Ricardo J Bessa, and Ildemar Decker. Through the looking glass: Seeing events in power systems dynamics. *International Journal of Electrical Power & Energy Systems*, 106:411–419, 2019.

[33] Weikang Wang, He Yin, Chang Chen, Abigail Till, Wenxuan Yao, Xianda Deng, and Yilu Liu. Frequency disturbance event detection based on synchrophasors and deep learning. *IEEE Transactions on Smart Grid*, 11(4):3593–3605, July 2020.

[34] Yufei Tang and Jun Yang. Dynamic event monitoring using unsupervised feature learning towards smart grid big data. In *International Joint Conference on Neural Networks*, pages 1480–1487, Anchorage, AK, USA, May 2017.

[35] Boyu Wang, Yan Li, and Jing Yang. LSTM-based quick event detection in power systems. In *IEEE Power Energy Society General Meeting*, pages 1–5, Montreal, QC, Canada, August 2020.

[36] Mert Kesici, Can Berk Saner, Mohammed Mahdi, Yusuf Yaslan, and V. M. Istemihan Genc. Wide area measurement based online monitoring and event detection using convolutional neural networks. In *7th Int. Istanbul Smart Grids and Cities Congress and Fair*, pages 223–227, Turkey, April 2019.

[37] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proceedings - IEEE International Conference on Neural Networks*, pages 1942–1948, November 1995.

[38] Eric Bonabeau, Guy Theraulaz, Marco Dorigo, Guy Theraulaz, Directeur de Recherches Du Fnrs Marco, et al. *Swarm intelligence: from natural to artificial systems*. Oxford university press, 1999.

[39] Marco Dorigo, Mauro Birattari, and Thomas Stutzle. Ant colony optimization. *IEEE Computational Intelligence Magazine*, 1(4):28–39, November 2006.

[40] Seyedali Mirjalili, Seyed Mohammad Mirjalili, and Andrew Lewis. Grey wolf optimizer. *Advances in Engineering Software*, 69:46–61, March 2014.

[41] John H Holland. Genetic algorithms. *Scientific American*, 267(1):66–73, 1992.

[42] David E. Goldberg. *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley Pub. Co, New York, USA, 1989.

[43] Dan Simon. Biogeography-based optimization. *IEEE transactions on Evolutionary Computation*, 12(6):702–713, December 2008.

[44] Richard P Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6-7):467–488, June 1982.

[45] Ajit Narayanan and Mark Moore. Quantum-inspired genetic algorithms. In *Proceedings of IEEE international conference on evolutionary computation*, pages 61–66, May 1996.

[46] Esmat Rashedi, Hossein Nezamabadi-Pour, and Saeid Saryazdi. GSA: a gravitational search algorithm. *Information sciences*, 179(13):2232–2248, June 2009.

[47] Richard A Formato. Central force optimization: a new metaheuristic with applications in applied electromagnetics. *Progress in Electromagnetics Research*, 77(1):425–491, November 2007.

[48] Hamed Shah-Hosseini. Principal components analysis by the galaxy-based search algorithm: A novel metaheuristic for continuous optimisation. *International Journal of Computational Science and Engineering*, 6(1-2):132–140, 2011.

[49] Osman K. Erol and Ibrahim Eksin. A new optimization method: Big bang–big crunch. *Advances in Engineering Software*, 37(2):106–111, February 2006.

[50] Bo Yang, Xiaoshun Zhang, Tao Yu, Hongchun Shu, and Zihao Fang. Grouped grey wolf optimizer for maximum power point tracking of doubly-fed induction generator based wind turbine. *Energy conversion and management*, 133:427–443, 2017.

[51] Belkacem Mahdad and K. Srairi. Blackout risk prevention in a smart grid based flexible optimal strategy using grey wolf-pattern search algorithms. *Energy Conversion and Management*, 98:411–429, 2015.

[52] Humberto Verdejo, Victor Pino, Wolfgang Kliemann, Cristhian Becker, and José Delpiano. Implementation of particle swarm optimization (PSO) algorithm for tuning of power system stabilizers in multimachine electric power systems. *Energies*, 13(8):2093, 2020.

[53] Habib Kraiem, Flah Aymen, Lobna Yahya, Alicia Triviño, Mosleh Alharthi, and Sherif SM Ghoneim. A comparison between particle swarm and grey wolf optimization algorithms for improving the battery autonomy in a photovoltaic system. *Applied Sciences*, 11(16):7732, 2021.

[54] Mohd Herwan Sulaiman, Zuriani Mustaffa, Mohd Rusllim Mohamed, and Omar Aliman. Using the gray wolf optimizer for solving optimal reactive power dispatch problem. *Applied Soft Computing*, 32:286–292, 2015.

[55] Habib Kraiem, Aymen Flah, Naoui Mohamed, Majed Alowaidi, Mohit Bajaj, Shailendra Mishra, Naveen Kumar Sharma, and Sunil Kumar Sharma. Increasing electric vehicle autonomy using a photovoltaic system controlled by particle swarm optimization. *IEEE Access*, 9:72040–72054, 2021.

[56] Jun Jiang, Xinghui Zhao, Scott Wallace, Eduardo Cotilla-Sanchez, and Robert Bass. Mining PMU data streams to improve electric power system resilience. In *Proceedings of the Fourth IEEE/ACM International Conference on Big Data Computing, Applications and Technologies*, pages 95–102, December.

[57] Saptarshi Sengupta, Sanchita Basak, and Richard Alan Peters. Particle swarm optimization: A survey of historical and recent developments with hybridization perspectives. *Machine Learning and Knowledge Extraction*, 1(1):157–191, 2018.

[58] Tim M Blackwell, J Kennedy, and R Poli. Particle swarm optimization. *Swarm Intelligence*, 1(1):33–57, 2007.

[59] Mudita Juneja and SK Nagar. Particle swarm optimization algorithm and its parameters: A review. In *International Conference on Control, Computing, Communication and Materials*, pages 1–5, Allahbad, India, October 2016.

[60] David H. Wolpert and William G. Macready. No free lunch theorems for optimization. *IEEE transactions on Evolutionary Computation*, 1(1):67–82, April 1997.

Appendix : Python Code

The python functions for the proposed event detection and optimization framework are available on GitHub:

Event Detection Algorithm for Power Systems - GitHub

Information about GitHub repository

Fresp _v3.py	Event Detection Algorithm with five tunable parameters
GWO _event _detection.py	GWO algorithm for parameter adjustment
PSO _event _detection.py	PSO algorithm for parameter adjustment
Human Validation 50.csv	Human validation file for a sample set of 50 frequency files used for this work