DISSERTATION

APPLICATION OF STATISTICAL AND DEEP LEARNING METHODS TO POWER GRIDS

Submitted by

Mantautas Rimkus

Department of Statistics

In partial fulfillment of the requirements

For the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Summer 2023

Doctoral Committee:

Advisor: Piotr Kokoszka
Co-Advisor: Haonan Wang

Aaron Nielsen
Dan Cooley
Haonan Chen

ABSTRACT

APPLICATION OF STATISTICAL AND DEEP LEARNING METHODS TO POWER GRIDS

The structure of power flows in transmission grids is evolving and is likely to change significantly in the coming years due to the rapid growth of renewable energy generation that introduces randomness and bidirectional power flows. Another transformative aspect is the increasing penetration of various smart-meter technologies. Inexpensive measurement devices can be placed at practically any component of the grid. As a result, traditional fault detection methods may no longer be sufficient. Consequently, there is a growing interest in developing new methods to detect power grid faults. Using model data, we first propose a two-stage procedure for detecting a fault in a regional power grid. In the first stage, a fault is detected in real time. In the second stage, the faulted line is identified with a negligible delay. The approach uses only the voltage modulus measured at buses (nodes of the grid) as the input. Our method does not require prior knowledge of the fault type. We futher explore fault detection based on high-frequency data streams that are becoming available in modern power grids. Our approach can be treated as an online (sequential) change point monitoring methodology. However, due to the mostly unexplored and very nonstandard structure of high-frequency power grid streaming data, substantial new statistical development is required to make this methodology practically applicable. The work includes development of scalar detectors based on multichannel data streams, determination of data-driven alarm thresholds and investigation of the performance and robustness of the new tools. Due to a reasonably large database of faults, we can calculate frequencies of false and correct fault signals, and recommend implementations that optimize these empirical success rates. Next, we extend our proposed method for fault localization in a regional grid for scenarios where partial observability limits the available data. While classification methods have been proposed for fault localization, their effectiveness depends on the availability of labeled data, which is often impractical in real-

life situations. Our approach bridges the gap between partial and full observability of the power grid. We develop efficient fault localization methods that can operate effectively even when only a subset of power grid bus data is available. This work contributes to the research area of fault diagnosis in scenarios where the number of available phasor measurement unit devices is smaller than the number of buses in the grid. We propose using Graph Neural Networks in combination with statistical fault localization methods to localize faults in a regional power grid with minimal available data. Our contribution to the field of fault localization aims to enable the adoption of effective fault localization methods for future power grids.

ACKNOWLEDGEMENTS

I express my sincere gratitude to my advisor, Professor Piotr Kokoszka, for his invaluable guidance, unwavering support, and expert advice throughout my program. His mentorship has been instrumental in keeping me on track, providing encouragement, and equipping me with essential technical and soft skills. I am fortunate to have worked under his supervision.

I would also like to thank my co-advisor, Professor Haonan Wang, for his support and invaluable insights throughout my academic journey. His commitment to mentoring and his extensive expertise have greatly enhanced my understanding of complex concepts.

My gratitude extends to my doctoral committee members: Professors Dan Cooley, Aaron Nielsen, and Haonan Chen, for their support and meaningful interactions over the years.

I am grateful to my collaborators Professor Dongliang Duan, Dr. Sanjay Hosur, Dr. Kumaraguru Prabakar, and Mr. Xuao Wang for their contributions and expertise in the projects within my dissertation. Their provision of relevant data sets and deep understanding of electrical power grids have guided me through complex challenges and expanded my knowledge in the field.

I am thankful to the professors at Colorado State University for their contributions to my statistical knowledge and mentoring in teaching. I also appreciate my fellow graduate students and friends for their support and help in staying sane during challenging periods.

I express my gratitude to my undergraduate professors, Alfredas Rackauskas and Remigijus Leipus from Vilnius University, as well as my high school math teacher, Faustas Klimavicius, for equipping me with the tools to begin my graduate studies and supporting me along my journey.

Lastly, I am deeply thankful to my family, both here and in Lithuania, for their long-term and long-distance support. I am grateful to my wife Vaida, mother Asta, father Gintaras, sisters Rugile and Dominyka, brother Valentas, grandmothers Lilija, Danute, and Alma, grandfathers Pranas and Algis, and godparents Jolanta and Gintaras. Their love has made me feel empowered throughout every step of my academic pursuit.

The authors are grateful to Dr. Daniel Trudnowski of Montana Tech for his permission to use the minniWECC model. The research presented in Chapters 2 and 4 was partially supported by the United States National Science Foundation grants DMS-1923142, EECS-1828066 and OAC-1923983. The research presented in Chapter 3 was partially supported by the following grants: NSF DMS- 2123761, NSF DMS-1914882, NSF DMS-1923142. The work in Chapter 3 was authored in part by the National Renewable Energy Laboratory, operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. Funding provided by U.S. Department of Energy Office of Energy Efficiency and Renewable Energy Office. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. Government. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. Government purposes.

TABLE OF CONTENTS

LIST OF TABLES

1.55em= 0

LIST OF FIGURES

Chapter 1

Introduction

This dissertation is divided into three primary sections. Chapter 2 develops a moving-window change point statistics for the detection of faults in regional power grids. The statistics are then utilized to propose a new two-stage procedure for fault detection in regional power grids. In Chapter 3, the sequential change point monitoring methodology is employed in the context of modern power grids. The main differences between Chapter 2 and Chapter 3 are the size of the power grid and the availability of variables. Chapter 4 extends the methodology introduced in Chapter 2 to scenarios where only partial network information is available. To address this, a combination of modern deep learning methods and statistical techniques is utilized to propose a fault localization method for regional power grids without using fault location information during the training process. Each chapter provides a detailed introduction to its specific topic. Before delving into the specifics of each chapter, in the remainder of this chapter a general introduction is provided regarding the Electrical Power System (EPS), the underlying concepts of statistical fault detection, and the distinctions between statistical and deep learning approaches.

EPS represents a pinnacle of human engineering. They are characterized by their importance and remarkable complexity. In today's world, the reliability, flexibility, and accessibility of these systems are absolutely essential for sustaining modern civilization. The true magnitude of its importance becomes apparent to the public when a major blackout occurs, highlighting its profound societal, economic, and national security implications. The structure of EPS primarily consists of three interconnected components: generation, transmission, and distribution. Starting with generation, fuel or increasingly renewable energy, is converted into electricity, which is then sent into the transmission system. Through an extensive network of transmission lines, the electricity is transported to various substations. Finally, consumers receive the electricity from these substations via distribution networks. However, one of the most significant risks to this intricate network is the occurrence of electrical faults. These faults manifest themselves as abnormal conditions within

the EPS, causing deviations from the nominal values of voltages and currents. Faults within the electric power system can arise from various factors, including but not limited to adverse weather conditions, aging infrastructure, insulation failures, falling trees, structural deficiencies, and malicious attacks.

Traditional protection devices have historically relied on local current measurements to make decisions regarding circuit tripping (IEEE, 1986). However, the emergence and increasing integration of distributed energy resources (DERs) have introduced significant challenges for traditional distribution protection devices. These challenges include: i) bidirectional flow of power; ii) changes in the fault current levels; and iii) changes in the system dynamic behavior under faulty conditions. As a result, mal-trip and fail-to-trip may happen more frequently in traditional protection devices, (Häger *et al.*, 2006). Fortunately, the implementation of smart grid infrastructure offers a solution by providing access to extensive data sets for power system operation, control, and protection. Leveraging fast communication systems and advanced processing tools, these data can be utilized in real time to gain a comprehensive understanding of the smart grid's operational status systematically. Analyzing this data enables researchers to uncover the underlying mathematical and statistical structures of the grid, which can be transformative in designing improved systems.

Fault diagnosis in EPS networks encompasses three main components: fault detection, fault classification, and fault localization. Fault detection involves identifying abnormal conditions within the grid infrastructure. Fault classification entails categorizing different types of faults, such as short circuits, line-to-ground faults, phase-to-phase faults, or transient faults, to determine the specific nature of the fault. Lastly, fault localization aims to pinpoint the precise location of the fault or anomaly within the grid infrastructure, enabling prompt and targeted repair or maintenance actions. EPS researchers have devised various fault diagnosis methods tailored for distribution systems, often leveraging transmission system schemes. Consequently, both transmission and distribution networks employ similar strategies for fault diagnosis.

In the fault diagnosis and process monitoring literature, statistical fault detection methods are commonly known as data-driven methods. A key advantage of data-driven methods over model-based methods is their ease of implementation. Unlike model-based methods, data-driven methods do not rely on precise system specifications, which may not always be available for various reasons. This makes data-driven methods more accessible and practical in real-world applications.

Change point detection methods are one of the statistical approaches commonly employed for fault detection. These methods focus on identifying abrupt changes in data, specifically when a property of the time series experiences a noticeable alteration at a specific time point, denoted as t_0. A time series is a sequential arrangement of observations, represented as $(x_1, x_2, \ldots)$. One application of change point detection methods involves comparing the joint distributions $f(x_{1:t_0})$ and $f(x_{t_0+1:T})$, where $x_{i:j} = (x_i, \ldots, x_j)$. By examining whether there are significant differences in these two distributions, it is possible to infer the occurrence of a change in the data. In contrast to two-sample tests, in change point test the time t_0 is unknown, and it is not clear if a change has occurred.

Another prominent use of statistical methods in fault diagnosis is the application of statistical learning for fault classification. Methods like support vector machines or classification trees provide tools to differentiate between different types of faults in the power grid. Naturally, due to ongoing methodological research and advancements in computational resources, researchers have increasingly started applying machine learning and deep learning techniques to address the same set of problems. These methods have demonstrated good performance when dealing with complex power grid data (Li *et al.* (2019a), Zhao and Barati (2021)).

The main difference between statistical and deep learning methods lies in their underlying approaches to data analysis and modeling. Statistical methods are based on mathematical and probabilistic principles. They rely on predefined statistical models and assumptions. On the other hand, deep learning methods are a subset of machine learning techniques that focus on training artificial neural networks with multiple layers. Deep learning algorithms learn representations from the data through a process known as feature learning or representation learning. This characteristic

makes deep learning particularly effective for tasks involving large and high-dimensional datasets that are difficult to understand. However, deep learning approaches directly do not lead to better understanding of complex systems and may fail when applied to data that is different from the data in the training process.

Chapter 2 has been published as the following paper "P. Kokoszka, M. Rimkus, S. Hosur, D. Duan and H. Wang. (2023). Detection and localization of faults in a regional power grid. *Austrian Journal of Statistics*, 52, 143-162". Chapter 3 has been published as the following paper "M. Rimkus, P. Kokoszka, K. Prabakar and H. Wang. (2023). Toward statistical real-time power fault detection. *Communications in Statistics – Case Studies and Data Analysis*". Chapter 4 has been submitted as the following manuscript "M. Rimkus, P. Kokoszka, D. Duan, X. Wang and H. Wang. (2023). Graph neural networks for the localization of faults in partially observed regional transmission systems".

Chapter 2

Detection and Localization of Faults in a Regional Power Grid

2.1 Introduction

Faults in power systems cause excessive currents and can pose safety threats to personnel and property, and even cause major disasters, like widespread fires, as well as disruptions of economic and social activities. Detection of power grid faults is therefore of paramount importance. As the penetration of renewable energy sources, with pronounced stochastic components, increases, traditional fault detection methods can become insufficient. The objective of this work is to propose a statistical methodology for detecting a fault in a regional power grid, with almost no delay, and locating the faulted line. The faulted line is also determined almost immediately. The algorithm we propose takes as inputs the moduli of voltage measured at high frequency at grid buses. Such data are becoming increasing available due to the growing deployment of phasor measurement units (PMUs) able to communicate power transmission measurements in real time and from practically any location where transmission lines connect, start or terminate (generators, transformers, forks, loads etc.). Such nodes of the grid are referred to as buses. The algorithm is developed on a training dataset and evaluated on a test data set that remains inaccessible to us until the algorithm development is complete. This widely adopted approach is designed to ensure that new methodology generalizes well. The novelty of our approach relative to previous research is discussed in greater detail in Section 2.4. We merely note here that we are not aware of any study that uses a large database of faults and statistically evaluates performance of a fault detection and localization methodology in terms of success rates.

The statistical methodology used in this paper falls into the general field of change point detection and anomaly localization. The field of statistical change point detection is now well-

established, and there are consequently many monographs on change point analysis, e.g. (Brodsky and Darkhovsky, 1993; Csörgő and Horváth, 1997; Gustafsson, 2000; Chen and Gupta, 2011; Basseville *et al.*, 2012; Brodsky, 2017). The methods we consider are known as *sequential-* or *online detection* or *monitoring*. Their objective is to raise an alarm if there is some departure from the desired state. They focus on minimizing the *Expected Detection Delay* (EDD) and maximizing the *Average Run Length* (ARL), i.e., the expected time until a false detection. The last two decades have seen important advances in theoretical understanding of methods for sequential detection of change-points which may occur in one of many channels or sensors. Research in (Dragalin *et al.*, 1999) is concerned risk optimality theory for multichannel log-likelihood ratio tests. There are M channels, and a penalty is imposed if a change in density is detected in an incorrect channel; stopping occurs as soon as a fault is detected in one of the channels. The work in (Mei, 2005) incorporates a decision center into the optimality considerations, while that in (Raghavan and Veeravalli, 2010) studies a situation where the change propagates across sensors. Reference (Xie and Siegmund, 2013) is concerned with mathematical properties of sequential change point detection in an idealized multi-sensor network with a change occurring in a subset of networks. Temporal and spatial independence and normality of observations are assumed. A somewhat similar approach is taken by (Zhang *et al.*, 2010) who assume independence across i and t and Gaussian observations, and propose a method of dealing with data that have a large cross-sectional dimension p, tens of thousands, and a huge temporal size T, hundred of thousands. They study gene expression data. In our setting, the value of p will be much smaller, and T will also have to be smaller to use a reasonable moving window. There is also very extensive research on detecting anomalies in various types networks networks. To illustrate different flavors of such research, we list, as subjectively selected examples, (Huang *et al.*, 2007), (Lévy-Leduc and Roueff, 2009), (Paschalidis and Smaragdakis, 2009), (Bartos *et al.*, 2011), (Xie, 2012), (Rassam *et al.*, 2013) and (Vaughan *et al.*, 2013). The methods developed in these and many other papers are designed to study networks and anomalies different from the power grid anomalies.

Power grid faults are very different from typical data that has motivated the development of existing approaches and the theory that underlies them. They are typically based on the statistical likelihood principle that leads to procedures based on the likelihood ratio. Such procedures enjoy optimality properties, but only under specific assumptions, almost always involving independence and often normal distribution. Power grid data satisfy none of these assumptions. All lines are connected and a fault propagates almost immediately through an affected subgrid. As the figures in Section 2.2 show, the fault data are very dependent. For the detection of a fault, this is a blessing, but it makes the problem of the localization of a fault very hard (measurements at all buses look similar). Our method is specific to a power grid and has no analog in existing network anomaly detection methods.

The paper is organized as follows. Section 2.2 introduces the grid data we work with and presents exploratory data analysis that motivates our approaches. In Section 2.3, we derive our detection and localization algorithms, present them by means of mathematical formulas and pseudocode, and assess their performance. Section 2.4 is dedicated to the discussion of existing or proposed engineering approaches and a comparison with another method. We conclude with a brief summary and main conclusions in Section 2.5.

2.2 Data Description

Real power grid fault data are not publicly available mostly due to liability concerns of power companies. Most studies use simulated data. Data used for the development of the statistical methodology proposed in this paper were generated within the framework of the minniWECC , a simulation system designed for the evaluations and testing potential system-wide damping control technologies. The term minniWECC is used to reflect its utilization as a simplified model for the US grid portion managed by Western Electricity Coordinating Council (WECC). The minniWECC represents the overall inter-area modal properties and has enough complexity to reflect the relevant properties of the full western interconnection's bulk power system. This model is geographically consistent, and it matches with properties of the actual system. Results derived on minniWECC

are transferable to the real-world grid. The data generated from the minniWECC model have been widely used to validate various power system mode estimation and event detection algorithms which are now adopted for control room applications, see e.g. (Follum *et al.*, 2017; Trudnowski *et al.*, 2013; Byrne *et al.*, 2016). There are eight regional power grids covering the US and Canada. Similar regional grids exist in Europe, e.g. the British grid, the Nordic regional grid and the Baltic regional grid that are interconnected with the synchronous grid of Continental Europe.

A simplified one-line diagram of the minniWECC, following (Trudnowski *et al.*, 2013), is given in Figure 2.1. The minniWECC version we use consists of 171 lines and 122 buses. Each line $l \in \{1,\dots,171\}$ has terminal buses $(i,j) : i,j \in \{1,\dots,122\},\ i \neq j$. We consider only pairs (i,j) for which there is a line l connecting buses i and j.

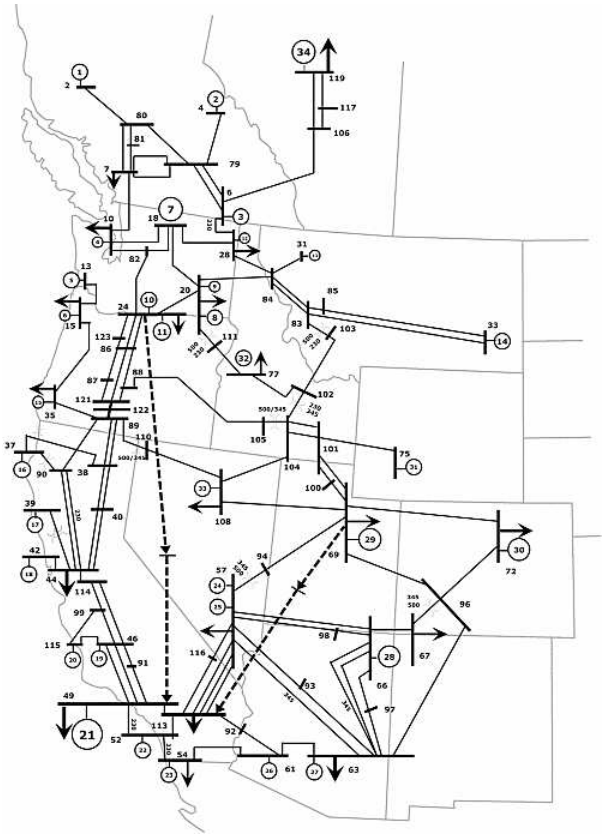

Figure 2.1: One-line diagram of minniWECC.

We simulated a large number of faults within the minniWECC. At the end of minute 10 from the beginning of each simulation, one of four fault types was applied to one of the 171 lines, and the data were recorded up to the end of minute 15. This resulted in 684 different simulations. For each simulation, the faulted line and the type of fault are known. The following fault types were generated:

line to line (LL): two phases of a three phase circuit are short circuited.

line to ground (LG): one conductor comes in contact with the ground or the neutral conductor.

line to line to ground (LLG): any two phases of the power circuit are short circuited to ground or neutral.

three-phase (TP): all three phases of the power circuit are shorted.

The minniWECC data were generated using the Power System Toolbox (PST), (Cheung *et al.*, 2009), which is based on MATLAB. The input data files required to run the simulations were created by Dr. D. Trudnowski. The details of how the simulations are carried out is well documented in the PST manual. The faults were generated using the switching condition matrix, in which the time of fault, time of clearing the fault, total duration of simulation and sampling rate were specified. The values of zero sequence ($z0$) impedance and negative sequence (zn) impedance were also specified in the switching condition matrix. For the generated data the value $z0$ and zn were chosen randomly. These values, in per unit (p.u), are $z0 = 0.17$ and $zn = 0.4$. The elements in the system model are the same for all types of faults. The sequence networks and the fault impedance vary for different types of faults. The equations used for calculating the impedance after different faults, in the simulation are:

three phase fault: $zf = 0$

line to ground fault: $zf = zn * \frac{z0}{zn+z0}$

line to line to ground: $zf = zn + z0$

line to line: $zf = zn$

In the above formulas, zf is the impedence with respect to ground after a fault.

Cases where simulation terminated due to numerical instability, are not included in further analysis. This results in a total of 546 simulations, where 117 simulations are from the TP fault cases, 139 simulations are from LLG cases, 145 simulations are from LL cases, and 145 simulations are from LG cases.

There are 116 out of 122 buses that have at least one line starting from it and can be considered as start buses. The number of possible indexes i in the (i, j) pairs is 116. For 91 out of these 116 i values, there exists only one unique j, i.e. there is only one line starting at i.

The simulated data consist of 120 measurements per second for 900 seconds (15 minutes). They include voltage readings at all buses reported as complex numbers. We consider the modulus of voltage at bus readings. The dimension of the dataset for each fault simulation is 122×108000. A fault is applied in the time interval $(600 - \frac{1}{120}, 600)$ in seconds, which is between observations 72000 and 72001 in each dataset. We set observation 72001 to be an exact fault time and so assume that second 600 is the earliest moment when the fault can be detected. To illustrate, we consider bus 105, which is the start bus of line 96. We gathered magnitude of voltage readings at bus 105 from all available simulations and highlighted simulations where line 96 was faulted. As different periods of the experiments have different variance of data, we represent the data by splitting the records into several time intervals (in seconds): $[0, 600 - \frac{1}{120})$, $[600 - \frac{1}{120}, 600 + \frac{1}{6})$, $[600 + \frac{1}{6}, 900]$. These intervals reflect the data before the fault, just after the fault including one point before the fault, and recovery after the fault. The beginning of the recovery phase is selected based on exploratory analysis. The graphs are shown in Figure 2.2. Notice that during the period $[0, 600)$ the data between simulation only differ by white noise. Bus 105 shows different responses to faults of different types. TP fault at line 96 has the largest effect on the readings at bus 105. For all fault types, most variability can be seen in the interval $[600, 600\frac{1}{6}]s$ (second panel in Figure 2.2). Approximately 0.1s after the fault, the recovery of the system begins. Figure 2.2 shows that one can expect to identify the start bus of the faulted line because it shows a special behavior. On

the other hand, this special behavior is more pronounced for some fault types than for other fault types, so the task requires careful consideration. Figure 2.2 also shows that localization of the faults should be achievable within a second after a fault.

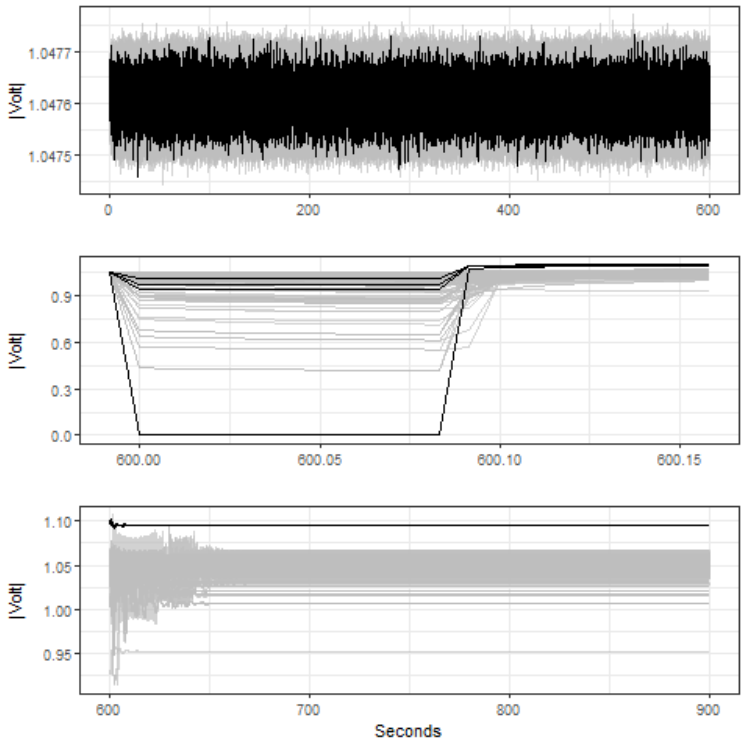

Figure 2.2: Modulus of voltage readings at bus 105 from all available (546) simulations. Faults at line 96 are highlighted (black) as bus 105 is the start bus of line 96. Readings resulting from faults at other lines (not 96) are plotted in gray. The three graphs represent different time stages of simulations. Notice the different voltage scales and different lengths of time intervals at the different stages.

Figure 2.3 presents the data in the interval $[600 - \frac{1}{120}, 600 + \frac{1}{6})$ from a different angle. The second panel of Figure 2.2 shows how a fixed bus "sees" the faults at all lines. Figure 2.3 shows how all buses see faults at a fixed line within the critical time interval just after a fault. To illustrate the differences in behavior between the buses, we consider faults at line 90. This line connects buses 88 and 121. The response to a fault at a specific bus depends on its relative location relative to the faulted line. For a given fault, the responses at all buses exhibit similar patterns; the size of the response is generally the largest at busses where the faulted line starts and ends, but the pattern is not obvious. Figure 2.3 shows that an almost instantaneous localization of a faulted line should

be possible, but the task is clearly not trivial. The next section is devoted to a systematic study of this problem.

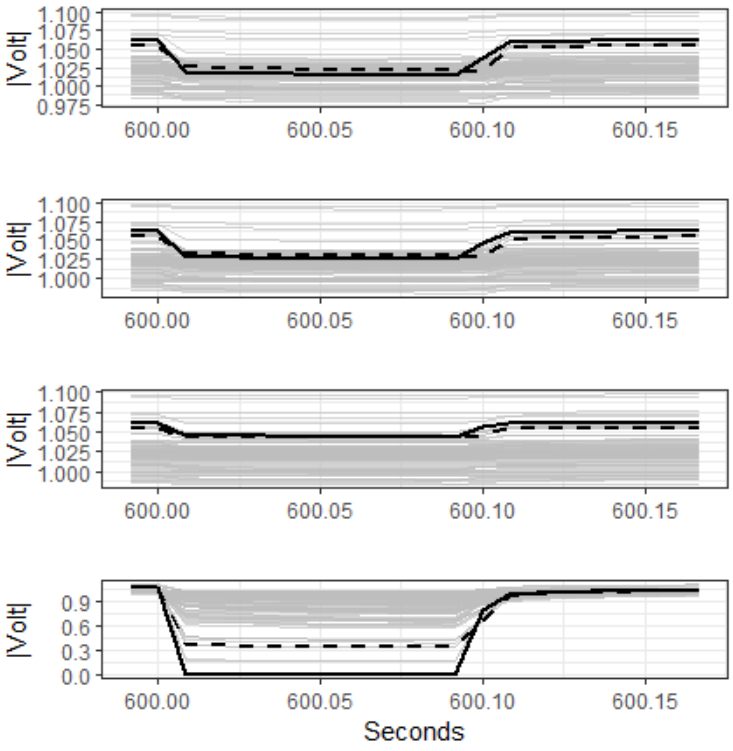

Figure 2.3: Responses at all buses to four fault types at line 90: LG, LL, LLG, and TP, from top to bottom. Responses at buses 88 and 121, which are connected by line 90, are highlighted in black; bus 88 in solid black, bus 121 in dashed black.

2.3 Derivation of Fault Detection and Localization Algorithms

The goal of the fault detection algorithms is to estimate the time M when the fault occurs and the goal of localization algorithms is to identify the faulted line. Localization is equivalent to finding the pair of buses (i_0, j_0) such that i_0 is the start bus and j_0 is the end bus of the faulted line l_0. Since we have only readings at buses, it is not possible to identify the faulty line in cases where there is more than one line that has the same start and end buses. In such cases, we can only identify the pair (i_0, j_0).

In order to describe our algorithms using mathematical formulas, it is convenient to introduce simple notation. We denote by $\mathbf{A}$ the set of all *pairs* of buses that are connected by at least one line, i.e.

$$\mathbf{A} = \{(i,j) :\ i,j \in 1,\ldots,122,\ i \neq j \text{ , there is line } l \in \{1,\ldots,171\}, \text{ such that start bus of line } l \text{ is } i \text{ and end bus is } j\}. \quad (2.3.1)$$

Define $\mathbf{B}$ the set of buses that are start buses of some line, i.e. $\mathbf{B} = \{i :\ \text{there is } j \in \{1,\ldots,122\}, (i,j) \in \mathbf{A}\}$. For each $i \in \mathbf{B}$, define set $\mathbf{B}_i$ by $\mathbf{B}_i = \{j : (i,j) \in \mathbf{A}\}$. For each $i \in \mathbf{B}$, $\mathbf{B}_i$ contains all possible end buses for lines that starts from bus i. The goal of a fault localization algorithm is to predict $(i_0, j_0) \in \mathbf{A}$, where $i_0 \in \mathbf{B}$ and $j_0 \in \mathbf{B}_{i_0}$. The prediction is denoted as $(\hat{i_0}, \hat{j_0})$.

We model the data as follows. In each simulation, we have a sample of random functions, or fine resolution time series, $x_i(t)$, where $i = 1\ldots,N$. In MinniWECC, we have $N = 122$ (the number of buses). The value $x_i(t)$ is the modulus of the voltage at bus i at time t. Define δ as the time difference, in seconds, between two consecutive time points. Then $t = 0, \delta, 2\delta, \ldots, T$ is the time domain. In this paper, $\delta = \frac{1}{120}$ and $T = 60 \cdot 15 = 900$ (seconds). Define M as the time in seconds when the fault occurs. In our simulations, a fault occurs at the beginning of second 600, thus $M = 600$s at each simulation run. This value is known to us, but not to any algorithms. The same applies to the location of the faulted line. The proposed procedure for fault detection consists of three stages: 1) detect an event of a fault in the grid in real-time; 2) identify the start bus of the faulted line; 3) identify the end bus of the faulted line.

The evaluation of algorithms on the entire data set might lead to procedures that fit the dataset well, but do not lead to a preferred algorithm that generalizes well. Therefore, in this paper, the data analysis and algorithm development are done on training data, and performance is evaluated using test data. The training dataset has been created by randomly, uniformly sampling 436 simulations

(80%) out of 546 available simulations, The remaining 110 simulations are set aside as the test dataset. This is typically called a train-test split approach to algorithm development and evaluation.

2.3.1 Detection of the time a fault event

The goal is to identify the time point M when the fault occurs. In the proposed monitoring procedure, we consider a moving window, $[t - S_0 - S_1, t]$, where t is the current moment of time, $S_0 \geq \delta$ is the length, in seconds, of the window ending at $t - S_1$, and $S_1 \geq 0$ is the length, in second, of the window ending at t. This is illustrated in Figure 2.4. The value $S_1 = 0$ corresponds to the case of taking only a single observation (at time t). A moving window ensures that a detection statistic can be calculated in real-time, as it is based on only a limited number of observations, and that it adjusts to the most recent state of the system. A slow evolution of system readings does not indicate a fault of the type we are aiming to detect.

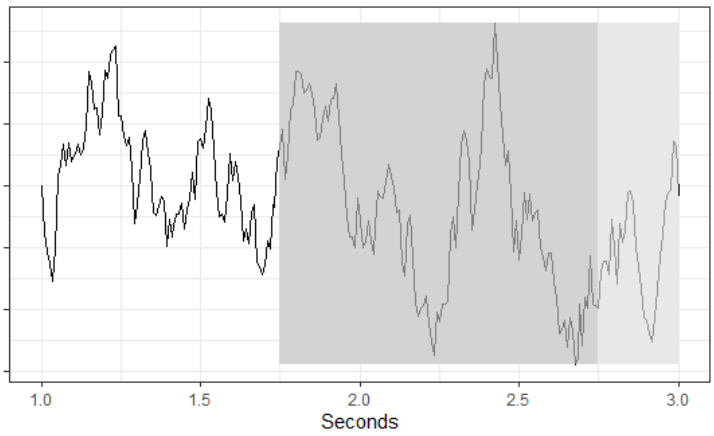

Figure 2.4: The moving window at $t = 3$ seconds with $S_0 = 1$s and $S_1 = 0.24$s. The light grey area represents the window $(t - S_1, t]$ that is used to evaluate (2.3.4). The darker gray area represents the window $[t - S_0 - S_1, t - S_1)$ that is used to evaluate (2.3.2) and (2.3.3).

To evaluate the state of the system up to time $t - S_1$, we use statistics $\bar{x}_i(t, S_0, S_1)$ and $\mathrm{SD}_i(t, S_0, S_1)$ defined for each bus i as follows:

$$\bar{x}_i(t, S_0, S_1) = \frac{1}{S_0/\delta} \sum_{k=1}^{S_0/\delta} x_i(t - S_1 - k\delta), \tag{2.3.2}$$

$$\mathrm{SD}_i(t,S_0,S_1)=\left(\frac{1}{S_0/\delta-1}\sum_{k=1}^{S_0/\delta}\left(x_i(t-S_1-k\delta)-\bar{x}_i(t,S_0,S_1)\right)^2\right)^{1/2}. \tag{2.3.3}$$

These are just the mean and the standard deviation of the magnitude of the voltage at each bus in the moving window $[t-S_0-S_1,t-S_1)$ in their native resolution. To assess the most recent state of the system, we calculate the averages in the window $[t-S_1,t]$, i.e.

$$m_i(t,S_1)=\frac{1}{1+S_1/\delta}\sum_{k=0}^{S_1/\delta}x_i(t-k\delta), \tag{2.3.4}$$

Notice that $m_i(t,S_1=0)=x_i(t)$. For each bus i, we define fault detector $D_i(t,S_0)$ as

$$D_i(t,S_0,S_1)=\frac{|m_i(t,S_1)-\bar{x}_i(t,S_0,S_1)|}{\mathrm{SD}_i(t,S_0,S_1)}. \tag{2.3.5}$$

The detector $D_i(t,S_0,S_1)$ is the absolute difference between the means in S_0 second before $t-S_1$ and S_1 seconds before t (including t) normalized by the standard deviation of the observations in the moving window $[t-S_0-S_1,t-S_1)$.

To check if a fault occurred at time t, we first compute

$$i_{\max}=\arg\max_i D_i(t,S_0,S_1). \tag{2.3.6}$$

The index $i_{\max}$ identifies the bus, which is the most "perturbed" at time point t. If the perturbation is large enough, the algorithm should detect the fault. In order to evaluate the size of the perturbation, we introduce a threshold parameter $\tau>0$. The algorithm detects a fault, if

$$D_{i_{\max}}(t,S_0,S_1)=\max_i D_i(t,S_0,S_1)>\tau. \tag{2.3.7}$$

If condition (2.3.7) fails, we move on to the next time point and perform the calculations again. The above procedure is summarized in Algorithm 1.

Algorithm 1 Detection of a fault in a grid

Input: Tuning parameters τ, S_0, and S_1, input data $\mathbf{X}$, time resolution δ, and upper time limit T;
Output: t_D The time where the fault is detected **Initialization** $Fault = 0$ and $t = S_0 + S_1$ Requires at least $S_0 + S_1$ seconds of data before the fault $Fault = 0$ & $t \leq T$ Set $t := t + \delta$ Find $D_i(t, S_0, S_1)$ for each i (2.3.5) Find $i_{\max} = \arg\max_i D_i(t, S_0, S_1)$ (2.3.6) $D_{i_{\max}}(t, S_0, S_1) > \tau$ Set $t_D := t$ and $Fault = 1$

There are several tuning parameters in Algorithm 1: S_0, S_1 and τ. We want to determine the tuning parameters that balance two criteria: 1) if there is no fault, false alarms should be rare, 2) if there is a fault, it should be detected with a large probability and the delay of detection (time of detection - time of fault) should be small. In traditional online monitoring problems, larger τ ensures that criterion 1) is met, and smaller τ ensures that criterion 2) is met. In power grids, the system is perturbed the most just after the fault happens, and the size of this perturbation is large; an occurrence of a fault is generally easy to spot.

We consider the following quantities:

F_1 - Fraction of simulation in which a fault is detected over the first 10 minutes. This quantity is similar to the size of a test or type I error.

F_2 - Fraction of simulations in which a fault is detected over the whole 15 minutes. This quantity is similar to power under alternative or 1 minus type II error.

D - The first t (in seconds) such that $\max_i D_i(t, S_0, S_1) > \tau$ less 600s, averaged over all simulation where a fault is detected. The quantity D is thus the average time of detection minus the time at each the fault has occurred (600s in our experiments).

In traditional tests, we would like to have $F_1 \approx 0.05$. Since in a power grid, a false alarm may be expensive, we may target a different value of F_1. Even $F_1 = 0$ might be reasonable. This is the case for our simulated data. Once we determined S_0, S_1, τ that give the desired F_1, we compute for them the fraction F_2 and select the values that give the largest F_2. We could then further select the tuning parameters such that D is the smallest possible. This general strategy can be applied to any regional grid, as long as software for simulating faults at the grid of interest is available. Details of its implementation are explained below, cf. the discussion around Tables 2.1 and 2.2.

The parameter S_0 determines how much prior data we need to detect a fault. In Algorithm 1, this parameter directly affects how accurate the estimates of the mean and standard deviation are. Thus S_0 should not be too small as a standard deviation estimate using just a few points is not accurate. The parameter S_1 is related to the detection delay, so it should be as small as reasonable. To determine values of S_0, S_1, and τ, we need to examine the behavior of the detector $\max_i D_i(t, S_0, S_1)$ over the first 600 seconds and during the fault. To explain, we begin with two pairs of parameters: $S_0 = 30\delta, S_1 = 0$ and $S_0 = 480\delta, S_1 = 11\delta$. Since the time step δ is equal to (1/120)s, these values correspond, respectively, to pairs of intervals of lengths (0.25s, (1/120)s) and (4s, (1/12)s). Values of the detector are larger for smaller values of S_0 and S_1, as shown in Figure 2.5. This means that useful values of τ must depend on S_0 and S_1, as illustrated in Figure 2.5.

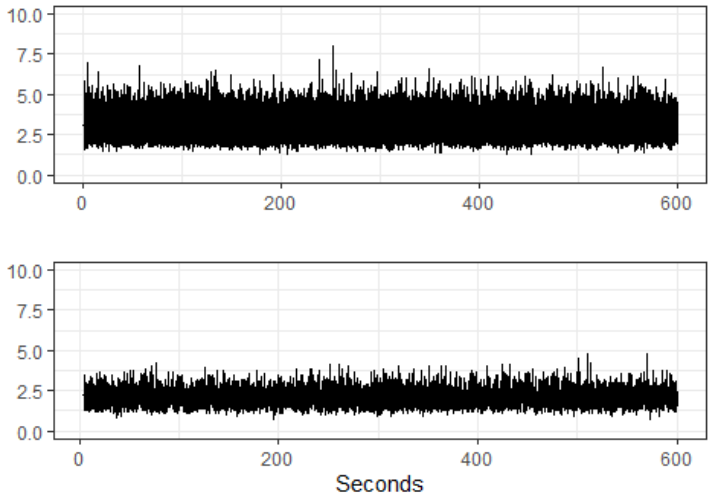

Figure 2.5: Values of the detector $\max_i D_i(t, S_0, S_1)$ over the first 600 seconds in one of the training simulations (no faults). Upper panel: $S_0 = 30\delta, S_1 = 0$, lower panel: $S_0 = 480\delta, S_1 = 11\delta$.

As every simulation is conducted identically before the fault at second 600, there is no need to examine the behavior of the detector for all simulations. However, after the fault, each simulation exhibits a different behavior as the location and type of the fault differ from one simulation to another. To understand the behavior of the detector in (2.3.7) at the time of the fault, one must therefore examine the whole training data set. A fault is detected if the detector exceed a threshold

Table 2.1: The maximum of the detector $\max_i D_i(t, S_0, S_1)$ over a single simulation in the training sample for $t \leq 600$ (before a fault) for various parameters S_0 and S_1 ($S_1 = 0$ corresponds to a single data point). For fixed S_0 and S_1, the threshold τ should be larger than the values in the table.

	$S_1 = 0$	$S_1 = 4\delta$	$S_1 = 11\delta$	$S_1 = 19\delta$
$S_0 = 15\delta$	9.84	15.16	18.26	19.79
$S_0 = 30\delta$	8.13	10.27	12.75	12.97
$S_0 = 120\delta$	5.97	5.77	6.61	7.40
$S_0 = 240\delta$	5.58	5.24	5.61	5.31
$S_0 = 480\delta$	5.31	4.85	4.78	5.31
$S_0 = 960\delta$	5.17	4.86	4.64	4.21
$S_0 = 1920\delta$	5.22	4.75	4.59	4.16
$S_0 = 7200\delta$	5.37	4.95	4.70	4.19

τ, so we must determine the minimum values of the detector at the time of the fault. These are presented in Table 2.2.

Table 2.2: The value of the detector $\max_i D_i(t, S_0, S_1)$ at $t = 600$ (time of fault) over all simulations in the training dataset.

	$S_1 = 0$	$S_1 = 4\delta$	$S_1 = 11\delta$	$S_1 = 19\delta$
$S_0 = 15\delta$	288.97	63.21	27.35	7.98
$S_0 = 30\delta$	244.72	36.22	15.18	10.58
$S_0 = 120\delta$	190.05	38.16	15.79	9.04
$S_0 = 240\delta$	133.16	26.57	11.31	6.75
$S_0 = 480\delta$	134.13	26.78	11.49	6.98
$S_0 = 960\delta$	157.24	31.64	13.66	8.29
$S_0 = 1920\delta$	175.12	35.30	15.25	9.26
$S_0 = 7200\delta$	178.11	35.91	15.51	9.44

During a fault, the values of $\max_i D_i(t, S_0, S_1)$ given in Table 2.2 are noticeably larger than the corresponding values given in Table 2.1. The quantity F_2 (power of detection) can be maximized using a threshold τ that does not exceed the values in Table 2.2. *The specific choice of S_0 and S_1 should be the one that maximizes the gap between corresponding values between S_0 and S_1 in Tables 2.1 and 2.2.* One can see that the values should be $S_0 = 30$ and $S_1 = 0$, even though all other values would be acceptable. Focusing on $S_0 = 30\delta$ and $S_1 = 0$, we see that we can take $\tau = 15$, but many other values will work too. Taking $S_0 = 30\delta, S_1 = 0, \tau = 15$ gives $F_1 = 0$ and $F_2 = 1$ and $D = 0$ on the training data set.

We summarize the general procedure for selecting the window lengths S_1, S_0 and the threshold τ.

1. Consider several potential values of S_0 and several values of S_1.

2. For each pair (S_0, S_1) in step 1, compute $\tau_{\max,0} = \max_{t\in\mathscr{T}_0} \max_i D_i(t, S_0, S_1)$, where $\mathscr{T}_0$ is the time interval with no faults.

3. For each pair (S_0, S_1) in step 1, compute $\tau_{\min,F} = \min_{t\in\mathscr{T}_F} \max_i D_i(t, S_0, S_1)$, where $\mathscr{T}_F$ is a short time interval containing the time of the faults. (The simulations must be synchronized so that all faults occur at the same time; we used $\mathscr{T}_F$ to be a single point set equal to the time of the fault.)

4. Select the pair (S_0, S_1) maximizing $\tau_{\min,F} - \tau_{\max,0}$.

5. Set $\tau = (\tau_{\min,F} + \tau_{\max,0})/2$.

As this is our final methodology for the detection of the time of the fault (derived from the training dataset), we tested it on the test dataset as well. Using $S_0 = 30, S_1 = 0, \tau = 15$ leads to $F_1 = 0$, $F_2 = 1$, and $D = 0$ on the test dataset. We conclude that there is no room for improvement in terms of performance.

The investigations reported in this section show that it is relatively easy to detect a fault in a regional grid if measurements at *all* buses are available. The parameters of the detector can be chosen in such a way that the detection is instantaneous, there are no false alarms and each fault is detected. One might be concerned that false alarms may occur due to the system evolution, e.g., load shedding or demand re-dispatch. To address this issue with confidence, one would need to simulate a large number of normal changes in the power system. Intuitively, the faults that we have in mind are sudden and large, as demonstrated in the figures shown above. The thresholds in our algorithms are trained for such large faults and they are unlikely to relatively small (in terms of the whole system), normal load changes.

2.3.2 Identification the faulted line

Once a fault has been detected, we want to determine the line at which it had occurred. This means that we must identify the pair (i_0, j_0) of terminal buses of the faulted line l_0. Our algorithm

does it in a sequential manner. We first identify a bus that exhibits the “most anomalous" behavior is a sense that will be quantified. We call such a bus the start bus of the faulted line. Then, we identify the end bus from among those that are connected to the start bus by a single line.

Denote by t_D the time of detection, i.e. the first t such that $\max_i D_i(t, S_0, S_1) > \tau$. A natural choice for the prediction of the start bus of the faulted line would be the bus i_{max} defined in (2.3.6) with t replaced by t_D. This approach indeed works reasonably well, but it can be improved. We obtained more accurate identification of the start bus by simply comparing the averages, i.e. by setting

$$\hat{i}_0 = \arg\max_{i \in \mathbf{B}} |m_i(t_D, S_1) - \bar{x}_i(t_D, S_0, S_1)| \tag{2.3.8}$$

Restricting the index i to the set $\mathbf{B}$ simply means that we consider only buses which are start buses of some line. If we use the values $S_1 = 0$ and $S_0 = 30\delta$ arising from the investigations reported in Section 2.3.1, we are looking at the difference between a single observation at the time of detection and the average of observations over the prior 0.25s. Using equation (2.3.8) with $S_0 = 30\delta$, and $S_1 = 0$, the start bus i_0 of the faulted line l_0 was identified correctly in 422 out of 436 simulations (422 cases where $\hat{i_0} = i_0$). This gives a 97% success rate.

Additional analysis shows that the 14 incorrectly identified instances ($\hat{i_0} \neq i_0$) relate to only three buses: 121, 122, and 89. The potential reason for failure in these cases could be that the magnitudes of the voltage at buses 121 and 89 correlate perfectly before the fault (correlation is equal exactly to 1). This could be due to a simulation setup. Our proposed methodology tends to favor the wrong bus. For example, in cases where bus 89 is the start bus of the faulted line, relation (2.3.8) tends to select Bus 122. More complete results are given in Table 2.3.

Table 2.3: Prediction of the start bus of the faulted line over all training simulations for buses 89, 121 and 122.

		Predicted		
		Bus 89	Bus 121	Bus 122
Actual	Bus 89	4	0	10
	Bus 121	3	0	0
	Bus 122	1	0	0

Once the start bus of the faulted line l_0 is predicted as $\hat{i}_0$, we need to predict the end bus of the faulted line. We denote this prediction by $\hat{j}_0$. Notice that once we have $\hat{i}_0$, the prediction $\hat{j}_0$ must be from $\mathbf{B}_{\hat{i}_0}$. Otherwise, the l_0 prediction would not make sense as $\hat{i}_0$ and $\hat{j}_0$ would not define a line. This greatly narrows the potential options for $\hat{j}_0$. If there is only one line originating at $\hat{i}_0$, then $\hat{j}_0$ is its end bus. However there are cases, where multiple lines originate at $\hat{i}_0$. Thus, if there are indexes j_a and j_b, such that $j_a \neq j_b$ and $j_a, j_b \in \mathbf{B}_{\hat{i}_0}$, an approach is needed to choose $\hat{j}_0$ from $\mathbf{B}_{\hat{i}_0}$. To find a prediction of the end bus of the faulted line, we propose and test two different procedures. The main difference between them is the time between the fault detection and the time when the prediction of j_0 is made. One may expect that an additional time delay in the identification of the end bus may increase the accuracy of the predictions, as the algorithm can utilize more information.

Notice that if the start bus prediction $\hat{i}_0$ is incorrect, the prediction of l_0 will be automatically incorrect. Thus for the end bus prediction, we only consider simulations, where i_0 was identified correctly. Using equation (2.3.8), we identified the start bus correctly in 422 out of 436 training simulations. Also, in 222 out of 422 simulations with correctly identified $\hat{i}_0$, $\mathbf{B}_{\hat{i}_0}$ has only one element, so the determination of $\hat{j}_0$ is trivial. The following algorithms are designed to identify the j_0 in the 200 training simulations, where the start bus was predicted correctly and there is more than one element in $\mathbf{B}_{\hat{i}_0}$.

In the first approach, j_0 is predicted at time t_D. This algorithm predicts the end bus of the faulted line similarly to equation (2.3.8), where the potential end buses must be in $\mathbf{B}_{\hat{i}_0}$, i.e.

$$\hat{j}_0 = \underset{j \in \mathbf{B}_{\hat{i}_0}}{\arg\max} |m_j(t_D, S_1) - \bar{x}_j(t_D, S_0, S_1)| \tag{2.3.9}$$

This algorithm thus predicts (i_0, j_0) at time t_D, the time of the fault detection. Using equation (2.3.9) with $S_0 = 30\delta$ and $S_1 = 0$, the end bus was correctly identified in 71 out of 200 simulations. This gives only 35% accuracy.

To describe the second approach, we need to introduce additional notation. We define the recovery process of bus i as

$$D_i^*(t,t_D,S_0,S_1) = m_i(t,S_1) - \bar{x}_i(t_D,S_0,S_1), \quad t > t_D. \tag{2.3.10}$$

Compared to (2.3.9), in (2.3.10) we use the interval ending at t_D to evaluate the state of the system before the fault, but push the current time forward after the fault. To illustrate, we show in Figure 2.6 the process (2.3.10) over the period of 0.25s after the fault (with $S_0 = 30\delta$, $S_1 = 0$). We do so for three training simulations. Notice that for each simulation, $D_i^*(t,t_D,S_0,S_1)$ undergoes a visible change approximately after 0.1s. This remains true for the remaining training simulations with the magnitude of change varying from simulation to simulation.

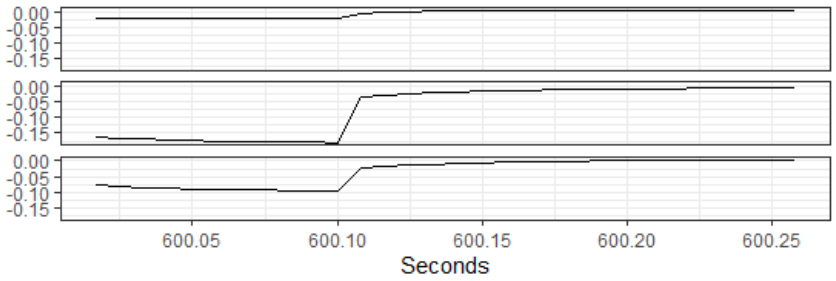

Figure 2.6: The recovery process (2.3.10) for three different training simulations ($S_0 = 30\delta$, $S_1 = 0$).

To determine the moment when recovery begins, we compute the absolute value of relative change and test the condition

$$\left| \frac{D_{\hat{i}_0}^*(t,t_D,S_0,S_1) - D_{\hat{i}_0}^*(t-\delta,t_D,S_0,S_1)}{D_{\hat{i}_0}^*(t-\delta,t_D,S_0,S_1)} \right| > \tau_1. \tag{2.3.11}$$

Examples illustrating the behavior of the left-hand side of (2.3.11) are shown in Figure 2.7. The time of the recovery, t_R, is defined as the smallest $t > t_D$ such that condition (2.3.11) is met. Its computation requires a selection of the value of τ_1. This threshold must be chosen sufficiently small to ensure that condition (2.3.11) is met for all simulations in the training dataset over a reasonably short time interval after a fault has been detected. Evaluating all potential $\tau_1 \in (0.01, 1)$

in increments of 0.01, shows that we need $\tau_1 \leq 0.17$ if we require $t_R \in (t_D + \delta, ct_D + 1)$ (the δ refers to (1/120)s and the 1 to 1s). The value $\tau_1 = 0.1$ is one of several values that maximize the performance of the algorithm we now describe on the training dataset, and this is the value we use in the following.

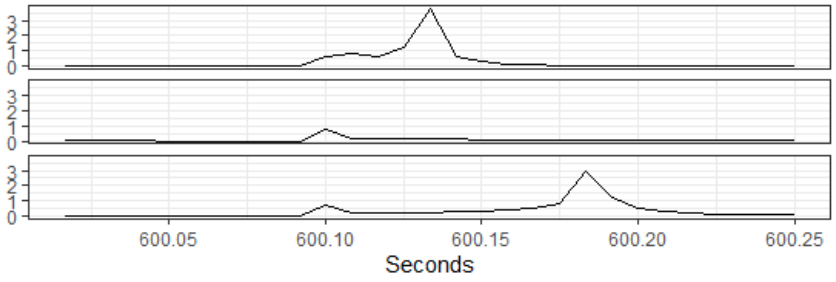

Figure 2.7: The left-hand side of (2.3.11) for three predicted start buses ($S_0 = 30\delta$, $S_1 = 0$).

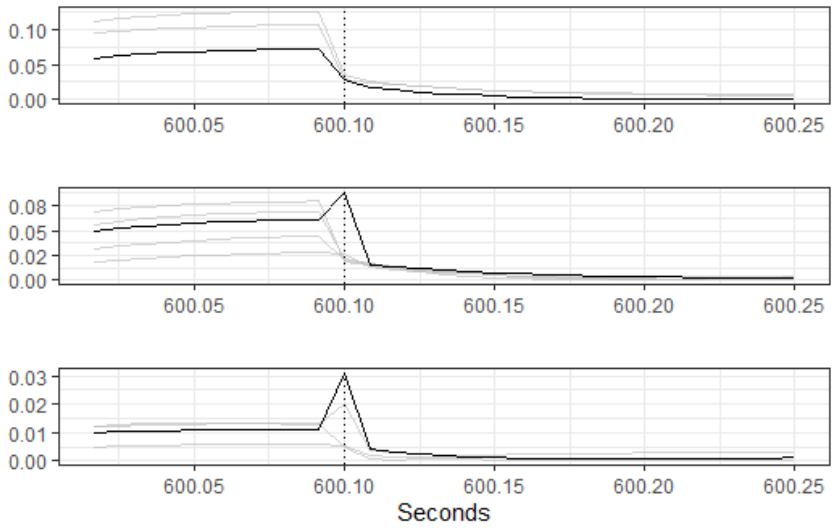

Figure 2.8: The absolute value of recovery process (2.3.10) for end buses in $\mathbf{B}_{\hat{i}_0}$ for three different training simulations ($S_0 = 30\delta$, $S_1 = 0$). In each of the three panels, the absolute value of recovery process corresponding to the correct end bus is highlighted (black). The vertical dotted line indicates the time t_R where $\tau_1 = 0.1$ is used.

We now describe an algorithm that identifies j_0 at time t_R. For each bus $j \in \mathbf{B}_{\hat{i}_0}$, we consider the recovery process $D_j^*(t, t_D, S_0, S_1)$ defined by (2.3.10). Examples illustrating its behavior are shown

in Figure 2.8. Approximately 0.1s after the fault, the buses in $\mathbf{B}_{\hat{i}_0}$ exhibit differing behaviors. The recovery process of the end bus of the faulted line shows somewhat different behavior compared to other buses in $\mathbf{B}_{\hat{i}_0}$. It generally undergoes the most pronounced change between 0.09s and 0.1s after the fault. To distinguish the end point of the faulted line from other end points, we use differences in absolute values of recovery process formula, i.e.

$$|D_j^*(t,t_D,S_0,S_1)| - |D_j^*(t-\delta,t_D,S_0,S_1)|, \quad t > t_D, \tag{2.3.12}$$

Examples illustrating the behavior of differences (2.3.12) are shown in Figure 2.9. Notice that the value of (2.3.12) for the end bus of the faulted line 0.1s after the fault is the largest compared to other buses in $\mathbf{B}_{\hat{i}_0}$. Our algorithm thus predicts j_0 as

$$\hat{j}_0 = \underset{j\in\mathbf{B}_{\hat{i}_0}}{\arg\max}(|D_j^*(t,t_D,S_0,S_1)| - |D_j^*(t-\delta,t_D,S_0,S_1)|). \tag{2.3.13}$$

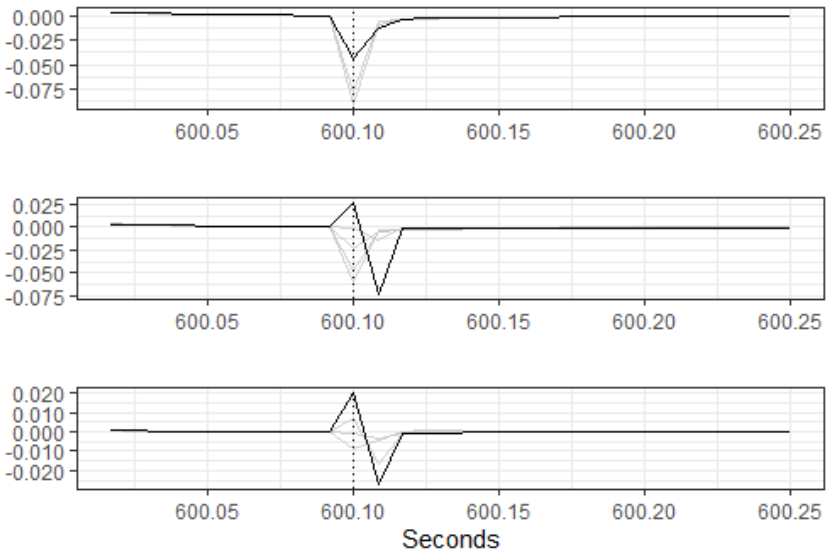

Figure 2.9: Differences in absolute value of recovery process (2.3.12) for end buses in $\mathbf{B}_{\hat{i}_0}$ for three different training simulations ($S_1 = 0$). The line for the correct end bus is highlighted (black). The vertical dotted line indicates time t_R where $\tau_1 = 0.1$ is used.

The above algorithm correctly identified the end bus in 179 out of 200 applicable simulations, 89% accuracy. Using (2.3.9), we only got 35% accuracy. We see that by delaying the identification of the end bus by 1s at most (in training simulations such value did not exceed 0.11s), we increased the performance by 54% on the training simulations. We thus recommend the algorithm based on (2.3.11). If we include the identification of the bus of the faulted line, we conclude that our methodology correctly identifies the faulted line in 401 simulations out of 436 training simulations. To summarize, our final methodology is defined by relations (2.3.8) and (2.3.13) with $S_0 = 30\delta$, $S_1 = 0$ and $\tau_1 = 0.1$. The procedure is summarized in Algorithm 2.

On the test dataset, our methodology correctly identifies the faulted line in 92% of simulations (102 simulations out of 110). The start bus of the faulted line was identified correctly in 95% validation simulations (104 cases out of 110). In simulations where the start bus i_0 was identified correctly and where was more than one eligible choice for j_0, the end bus was correctly identified in 97% of instances (56 simulations out of 58).

Algorithm 2 (i_0, j_0) prediction using time points t_D and t_R

Input: Tuning parameters τ_1, S_0, and S_1, input data $\mathbf{X}$, time resolution δ, network description data $\mathbf{Y}$, the fault moment t_D

Output: $\hat{i}_0, \hat{j}_0$ Find $\hat{i}_0$ using (2.3.8) $|\mathbf{B}_{\hat{i}_0}| = 1$ Set $\hat{j}_0 = j_0$, where $j_0 \in \mathbf{B}_{\hat{i}_0}$ Find t_R using (2.3.11)

1: Find $\hat{j}_0$ using (2.3.13)

2.4 Discussion of Other Approaches

Faults in power grids have been the subject of intense research. In general, existing fault detection schemes in power systems can be categorized into the following three approaches: quantitative model-based, qualitative model-based, and data-driven approaches (Jiang *et al.*, 2011). Quantitative model-based approaches and qualitative model-based approaches could achieve good detection performance in simulations. However, when implemented in real-world applications, they are quite sensitive to the noise in the voltage and current measurements. Moreover, they assume that the system models are accurately given and their detection performance would be greatly af-

fected by the inaccuracy in the system models. Recently, data-driven approaches begin to receive considerable attentions from the researchers due to the following reasons: 1) various intelligent electronic devices (IEDs) have been widely adopted and installed in the power grid (Moghaddass and Wang, 2017; Ghosal and Conti, 2019; Chen *et al.*, 2020), which collect large amount of different data at many nodes across the entire grid; and 2) compared to the traditional model-based approaches, the data-driven approaches are more resilient against measurement errors and system model inaccuracy. Meanwhile, they have more flexibility in their implementations and adapt better to the variations in system components and/or topology (Chen *et al.*, 2018; Chen *et al.*, 2019; Tripathi and De, 2018; Yin *et al.*, 2014). Following (Zhou *et al.*, 2019), we note that event detection was studied using moving averages, (Chow *et al.*, 2007), principal components, (Xie *et al.*, 2014), geographical visualization, (Kaci *et al.*, 2014), wavelets, (Kim *et al.*, 2017), dynamic programming, (Cui *et al.*, 2019) and energy similarities, (Yadav *et al.*, 2019). Past work has considered fault classification, (Nguyen *et al.*, 2015), cascading events, (Rafferty *et al.*, 2016), and cyber events, (Pan *et al.*, 2015), (Giani *et al.*, 2013), (Liao and Chakrabortty, 2019). The above list of citations is only meant to illustrate the scope of the research and is not exhaustive.

Several papers are closely related to our research. Before discussing them, we note that the chief and novel feature of our work is that we consider a large grid (122 buses) that mimics a real regional grid and a large number of faults. Existing research focuses on small test grids and methods are often illustrated on a single fault. Our work thus has important practical and statistical dimensions. In certain aspects, our approach is less sophisticated than many other approaches, but simplicity may be of advantage in practical applications. The complex structure of existing methods makes them difficult to implement on our grid data. Moreover, their descriptions generally omit the details of the implementation and no publicly available code is available. The papers discussed below make profound contributions, but the approaches they propose are difficult to implement using the information they provide. Another novel aspect of our approach is that it can identify the faulted line with high probability, not just an affected bus (which may have many connecting lines) or its neighborhood.

We now discuss selected papers to justify the points made above. Using the magnitude of the voltage, (Gholami *et al.*, 2019) develop advanced fault detection methods based on multiple detectors and likelihood computed from an ensemble model. The approach of (Hannon *et al.*, 2019) requires more data than our approach - current, frequency, voltages, and the application is concerned with one bus. The focus of (Ardakanian *et al.*, 2016) is the estimation of the admittance matrix. A change in this matrix and its localization provide information on the timing and the localization of a fault. The algorithm is evaluated on a 13 bus grid and one fault. In (Pandey *et al.*, 2020) the authors provide a method to find the bus (PMU) and the subgraph where the fault occurred. This is similar to finding the start bus of the faulted line in our approach. Similarly, (Li *et al.*, 2019b) develop methodology to detect the bus that is close (in a small neighborhood) to the faulted line.

To provide some idea on the advance our method makes relative to existing statistical approaches to change point detection, we implemented the standard moving window CUSUM approach which is described in many textbooks, see e.g. (Brodsky and Darkhovsky, 1993). We consider a moving window, $(t-K\delta, t]$. The lag K is similar to $(S_0+S_1)/\delta$ and corresponds to the number of points in a window. The algorithm is the same for each t, so it is easier to think about the beginning and the end of the window. We compute the averages before and after the potential change point (fault): $\hat{\mu}_i(k) = \frac{1}{k}\sum_{l=1}^{k} x_i(t-K+l\delta)$; $\tilde{\mu}_i(k) = \frac{1}{K-k}\sum_{l=k+1}^{K} x_i(t-K+l\delta)$, where $k \in \{1,\ldots,K-1\}$. Next, we compute the normalized difference of these averages,

$$P_i(t,k) = \frac{k(K-k)}{K}[\hat{\mu}_i(k) - \tilde{\mu}_i(k)]$$

and the detector for a single bus

$$D_i(t,K) = \frac{1}{K-1}\sum_{k=1}^{K-1} P_i(t,k)^2.$$

The detector for the whole grid is

$$D(t,K) = \frac{1}{N}\sum_{i=1}^{N} D_i(t,K). \tag{2.4.1}$$

We signal a fault if $D(t,K) > \tau$. The value of τ would be determined by a procedure we employed in Section 2.3, but we can illustrate more convincingly that our method works much better by the examination of Figure 2.10 which shows that the detector (2.4.1) reacts to a fault much slower than our detector.

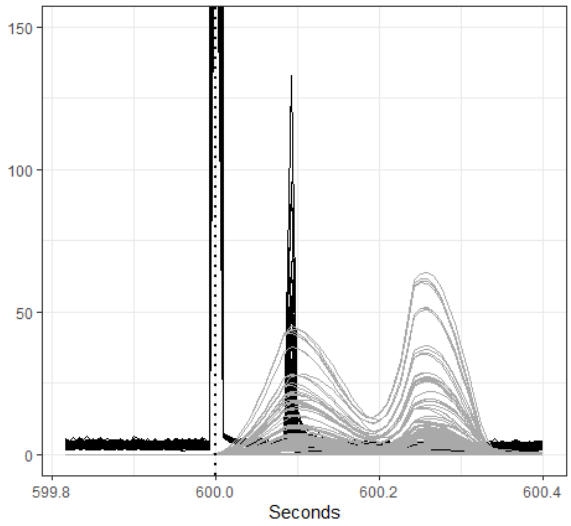

Figure 2.10: Comparison of the proposed method and the traditional moving window CUSUM approach. Curves in black show our detector calculated using (2.3.7) with $S_0 = 30\delta$ and $S_1 = 0$. Curves in gray are calculated using (2.4.1) with $K = 30$. The vertical dashed line represents the time of the fault. Both detectors have two spikes. For our proposed detector the first spike occurs almost simultaneously with the fault and is so high that it crosses the upper boundary of the display. The second, not relevant, spike of our detector coincides with the first spike of the traditional detector.

Intuitively, the detector based on equation (2.3.7) works better because instead of calculating statistics for each point of a moving window, we only calculate them at the windows $[t - S_0 - S_1, t - S_1)$ and $[t - S_1; t]$. This also make our method much faster. We introduce the normalization with the SD, as this helps to standardize the signals from each bus and tune up the τ properly. This is useful because the SDs at different buses are not necessarily the same. The identification of an

appropriate threshold τ is more difficult in case of detector (2.4.1), but we do not discuss the details because our proposed detector works much better with a clear cut threshold.

2.5 Summary

We have proposed a two-stage procedure for fault monitoring in a regional power grid. In the first stage, a fault is detected. In the second stage, the faulted line is identified. Our methods are fully data-driven and require only knowing the start and end buses of each line. Our procedure assumes that a fault occurred on a line, which corresponds to most practical scenarios. If a fault occurs on a bus, the identification of the faulted bus can follow the algorithm for finding the start bus i_0 of the faulted line. While some faults are easier to detect than others, our method does not require prior knowledge of the fault type. The approach uses only the voltage modulus measured at buses as the input. The general framework can be extended to different inputs, e.g. current or power. The chief contribution of this work is to develop a general statistical approach to anomaly detection and localization suited to regional power grids. These grids can be viewed as networks or graphs, but methods developed for such data structures in different context are not applicable due to a very special form of power transmission networks and anomalies that can occur in them. Our approach can be adapted and fine-tuned to work with somewhat different regional grids, but the general paradigm could be followed.

We emphasize that the approach we presented assumes that PMUs are placed at all buses. As pointed out by a reviewer, so far, power grids in the real world are only partially covered by PMUs. While a complete coverage may be expected in the future, it is important to extend the method to grids with incomplete PMU coverage. Partial insights in this direction are reported in (Rimkus *et al.*, 2023) who focus on fault detection (no localization) in a smaller distribution grid.

It may be of interest to compare our method to other approaches that may be developed in the future. To facilitate such comparisons, we placed our commented code at https://github.com/MantautasRimkus/DetLocRegionalGrid. The code is written in `R` (R Core Team, 2022). We are not aware of an existing method that can be readily applied to detect and localize a

fault in a regional power grid to which our method can be compared. Several sound and promising approaches exist that could potentially be generalized to a regional grid, but they are not implemented in publicly available software.

Chapter 3

Toward statistical real-time power fault detection

3.1 Introduction

Faults in power systems cause excessive currents and pose safety threats to people and property, and even cause major fires with substantial economic and social impacts. Fast and reliable detection of faults is therefore of paramount importance. Traditional methods use high fault current magnitude and power flow direction to differentiate between normal operating conditions and faulted conditions. Integration of inverter based (solar, wind) distributed energy resources (DERs) in the distribution system has created bi-directional current flows under normal operation and reduced the effectiveness of the traditional methods based on current direction. This challenge has created great interest in new methods of detecting faults in power grids. Our objective is to contribute to the recent intensive research in this area by proposing a relatively simple, but effective, way to detect power grid faults. Our approach is different from existing engineering approaches and points toward possibilities of using high-frequency measurements generated by modern measurement units. It is anchored in statistical sequential change point monitoring methodology that goes back to the work of Page (1954) and has been developed in many directions, see Lorden (1971), Lai (1998) and the book of Tartakovsky *et al.* (2015), among hundreds of significant contributions. However, due to the mostly unexplored and very nonstandard structure of high-frequency power grid data, a great deal of exploration and many adjustments need to be made to make the approach practically applicable. The objective of this paper is to propose a general statistical paradigm for power grid fault detection using the type of data that are becoming available in real time. In addition to the new statistical methodology, an advance over very extensive engineering literature is the computation of success rates, similar to type I and II errors, and incorporation of incomplete data (some measurements may be unavailable).

Existing fault detection methods in power systems fall into the following three categories: qualitative model-based, quantitative model-based, and data-driven approaches, see Jiang *et al.* (2011). Data-driven approaches have recently been receiving considerable attention for the following reasons: (1) Various intelligent electronic devices (IEDs) have been widely adopted and installed in the power grid (Moghaddass and Wang, 2017; Ghosal and Conti, 2019; Chen *et al.*, 2020). These devices collect large amounts of different data at many nodes across the grid. (2) Compared to the traditional model-based approaches, the data-driven approaches are more resilient against system model misspecification and adapt better to the variations in system components and/or topology (Chen *et al.*, 2018; Chen *et al.*, 2019; Tripathi and De, 2018; Yin *et al.*, 2014). To illustrate the scope of research on power grid fault detection, without aiming at a complete list of references, we also cite (Zhou *et al.*, 2019; Chow *et al.*, 2007; Xie *et al.*, 2014; Kaci *et al.*, 2014; Kim *et al.*, 2017; Cui *et al.*, 2019; Nguyen *et al.*, 2015; Rafferty *et al.*, 2016).

Recent new approaches generally test a proposed method to a single fault. For example, using the magnitude of the voltage, Gholami *et al.* (2019) develop advanced fault detection methods based on multiple detectors. The approach of Hannon *et al.* (2019), based on machine learning, is very promising, but the application uses only one bus. The focus of Ardakanian *et al.* (2016) is the estimation of the admittance matrix. A change in this matrix and its localization provide information on the timing and the localization of a fault. The algorithm is evaluated on a 13 bus grid, similar to the one we use, but only on a single fault. Recent methods for fault localization, e.g. Li *et al.* (2019b) and Khushwant *et al.* (2021), require the knowledge of the time the fault occurs, the issue we address. The importance of incorporating PMU measurement before, during and after a fault was emphasized by Chen *et al.* (2016) who proposed a method to detect and identify faults in near real-time by exploiting the statistical properties of voltage phase-angle measurements. We take a similar view. Our work has a statistical dimension because we apply many versions of our method to a relatively large database of 55 faults, and report success and failure rates, similar to size and power of a significance test, to identify the best methods. As we explain in Section 3.2, the 55 faults we consider are basically all possible faults in a 13 bus system we use as a test bed.

To summarize, we develop a basically complete statistical methodology to detect a fault in a small grid based on high-frequency measurements at selected notes of the grid. Since the temporal resolution of data streams is very high, the detection is almost immediate. Such an immediate detection can be expected of many other methods, if a fault in fact did occur. We therefore focus on the correct decision. It is important to avoid false alarms because cutting off power has serious consequences. On the other hand, failing to detect a fault may have even more serious consequences. We therefore focus on frequencies of correct and incorrect decisions, similarly as in the Neyman–Pearson testing paradigm. For this, reasonably many different faults are needed. We address the following items: (1) preprocessing of irregularly recorded data streams, (2) development of scalar detectors based on multichannel data streams, (3) determination of dynamic alarm thresholds, (4) investigation of the performance and robustness of our methodology. In the end, we are able to recommend specific implementations that work very well in our test-bed grid. However, our process of statistical methodology development can be even more valuable, as it can be viewed as a recipe that could be applied in similar, but different, settings.

The remainder of the paper is organized as follows. Section 3.2 is dedicated to the description of the data we study, Section 3.3 to the development of the methodology and Section 3.4 to the examination of its performance. We summarize and provide an illustrative algorithm in Section 3.5.

3.2 IEEE 13 bus feeder: structure and data generation

Real power grid fault data are generally not publicly available; power companies do not share them, chiefly for legal reasons. However, the mechanisms and types of faults that occur in real power grids are well understood. National Renewable Energy Laboratory (NREL) has hardware and software to generate faults of any type at specified times and locations within a grid that can be designed to correspond to a problem to be studied. In this paper, we use a fairly standard grid design, the IEEE 13 bus feeder, to generate streaming data and faults. The small grid we consider is the usual test bed in power grid research that can be identified with a small subgrid within which

a fault can occur. We now proceed to describe it and the data that have been generated at NREL to develop a statistical fault detections technique.

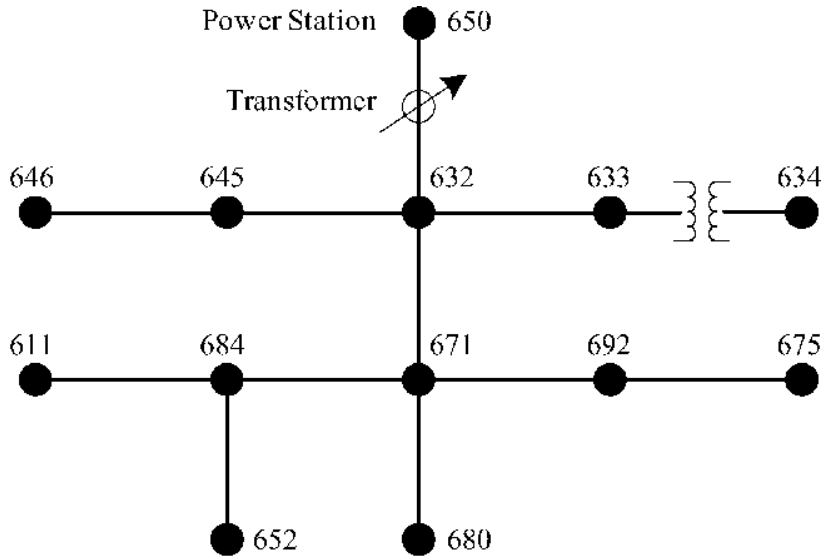

Figure 3.1: IEEE 13 bus feeder representation.

The general structure of the IEEE 13 bus feeder, PES (2020), is shown in Figure 3.1; it is a connected graph with 13 nodes. In power grid research a node is called a bus. For example, we refer to node 650 as Bus 650. Measurement devices are placed at buses. They cannot be placed at transmission lines that connect the buses. We use only data from 6 buses, similarly to Onaolapo *et al.* (2019) and Hu *et al.* (2013). We assume that each of these 6 buses represents a different part of the system:

1. Bus 650 (the substation)
2. Bus 632 (includes buses 645, 646, 633)
3. Bus 634
4. Bus 671 (includes buses 684, 611, 652, 692)
5. Bus 675
6. Bus 680

The above partition of the grid is dictated by the way the buses are connected and power transmitted. For example, there is only a circuit breaker between Bus 671 and Bus 692, so they are identical buses when the circuit breaker is closed. The data at Bus 611 and Bus 652 can be assumed to be identical to the data at Bus 671 because the lines connecting Bus 671 to Bus 611 and Bus 652 are short. Similar arguments hold for Bus 632 with the surrounding buses; voltage drops are insignificant from the perspective of the target application.

To explain what data are available at each of the six buses, we must briefly describe how power is generated and transmitted. We provide only the most essential facts that are well-known to power grid engineers and researchers, but may be less known to statisticians. Detailed background is available in dozens of textbooks, Glover *et al.* (2022) is a recent edition of a popular textbook. Alternating current (AC) is generated by a generator that has a large electromagnet spinning inside stationary coils of wire (windings). Industrial generators have three separate windings, each producing its own current. These separate currents are referred to as A, B and C phases. Measurement devices at buses can measure each phase separately and all three phases combined. If there are no faults, the voltage and current of the phases are shifted exactly by $\pi/3$ radians. This corresponds to the equi-spaced locations of the three windings inside which the electromagnet rotates at a constant speed, generating currents of constant frequency (nominally 60 Hz in the United States). *At each bus, we thus have measurements of 12 variables: voltage (in kilovolts), current (in kiloamperes) and frequency (in hertz) for phases A, B, C, and three-phase.* From these measurements, voltage and current Root Mean Square (RMS) can be computed, which is the peak voltage or current divided by $\sqrt{2}$. This convention is routinely used in sinusoidal AC power transmission systems because then the transmitted power can be calculated as the product of the RMS current and RMS voltage. *If there is a fault, the measured current, voltage and frequency will be impacted, but they also exhibit small fluctuations in the absence of any faults.*

This IEEE 13 bus systems is simulated in a digital real time simulator (DRTS) in electromagnetic transients (EMT) domain. The RMS of voltage and current are calculated in real-time. The

frequency measurements are calculated using phase locked loop (PLL) approach. The measurements are stored as csv files through UDP (User Datagram Protocol) streaming from DRTS.

For this study, 55 different simulations of faults were conducted. These include all possible bus faults in this test grid. We used a fault model from the RSCAD library which uses an impedance in the path to the ground. We used the value of 0.1 Ohms for the impedance value. In each simulation, different types of faults were applied to buses 632, 634, 646, 650, 671, 675, 680, 684. Additionally, faults were applied at the middle of the line between buses 632 and 671. In each simulation, the fault was applied at the beginning of second 10 and the simulation was running for another 5 seconds. The total length of each simulation is thus 15 seconds. It is assumed that the earliest time when a fault can be detected is the start of second 10. If a fault is detected earlier, it is treated as a false detection. Depending on the type of the faulted bus (two-phase or three-phase), up to seven different types of faults are possible. These types are (1) Phase A fault (2) Phase B fault (3) Phase C fault (4) Phases A and B fault (5) Phases A and C fault (6) Phases B and C fault (7) Three-phase fault. Also, additionally, one 35 second long simulation without a fault was conducted. It is used for training certain aspects of our algorithm, as elaborated on in Section 3.4.1.

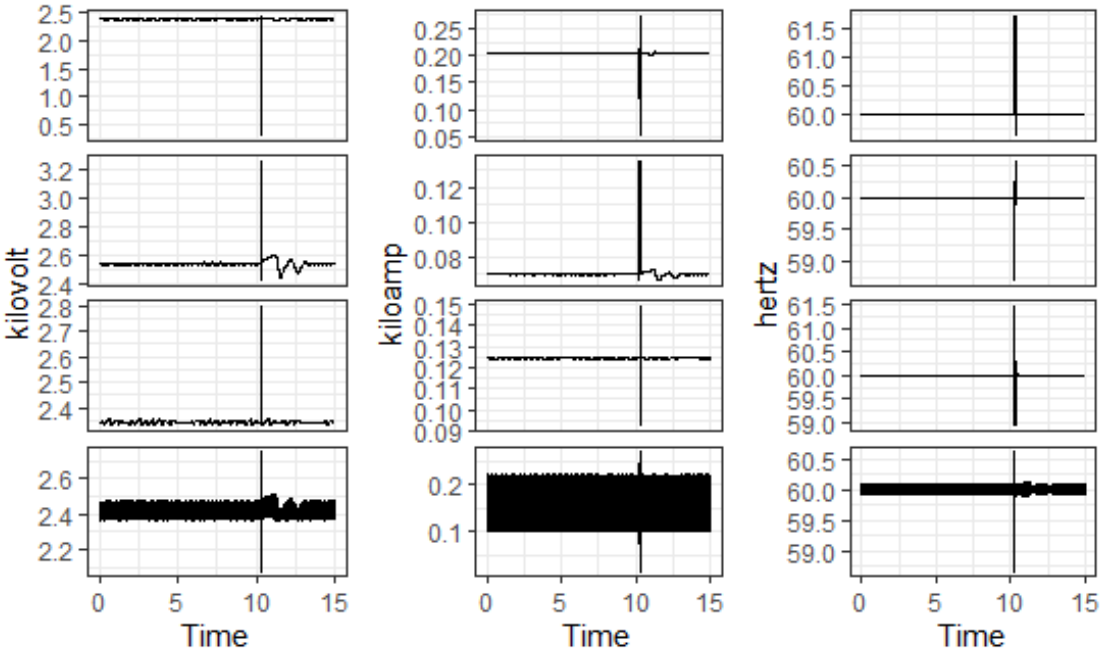

Figure 3.2: Measurements at Bus 675 with phase A fault applied at Bus 632. The fault is applied at the beginning of second 10. From Top to Bottom - phases A, B, C, and Three-phase measurements. From Right to Left - voltage, current, and frequency measurements.

Figure 3.2 presents a part of the data for Phase A fault at Bus 632. The presented data contains the measurements at Bus 675. Each time series goes through noticeable perturbations after the fault. Figure 3.3 presents the same data, but focuses on the first 0.3 seconds after the start of second 10, i.e. after the fault. Notice that the first visible reaction can be seen only around 0.25 seconds later.

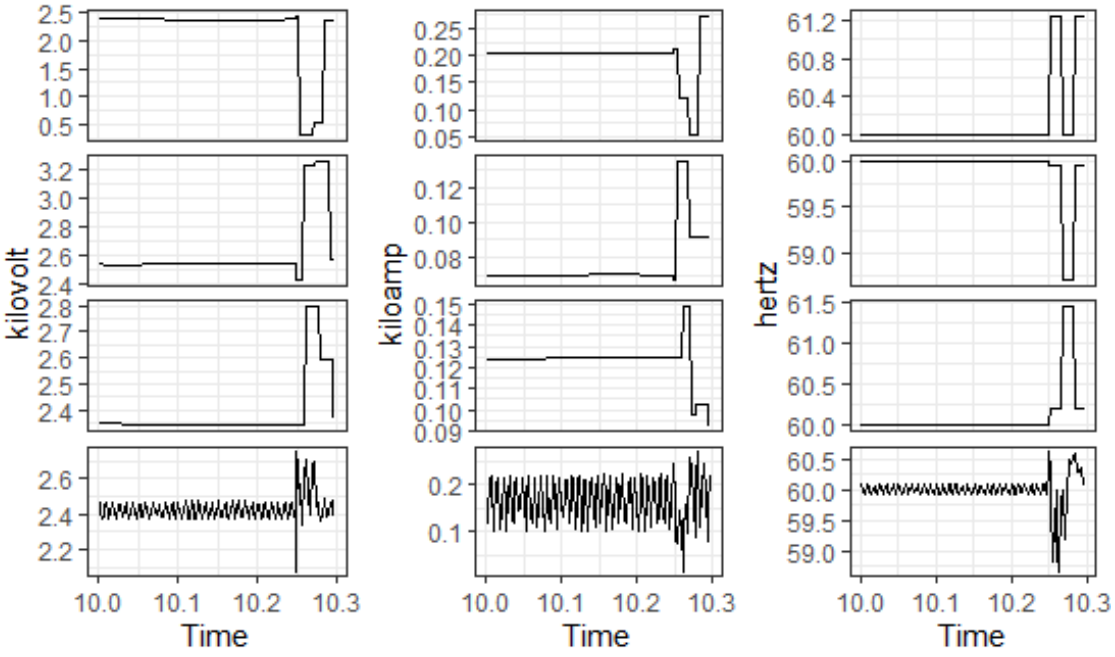

Figure 3.3: Close-ups of the data shown in Figure 3.2, the first 0.3 seconds after the fault.

If we take all simulations and all buses into account, we would end up with $6 \times 55 = 330$ Figures analogous to Figure 3.3. Visual examination of some of them reveals that while delays are similar, the shapes of responses differ. A relationship between the location of a fault and a location of the bus at which it is measured, as well as the responses of the 12 measured variables and the type of the fault, are not easy to establish. In the next section, we describe a general approach that uses all available information and work out the details of specific, most effective approaches. The effectiveness is measured by a low rate of false alarms and a high rate of correct fault detections.

3.3 Methodology

The goal of statistical change-point detection is to determine if a change in the structure of data has occurred and to estimate the time of the change. The context of the streaming power grid data we consider falls into the field of on-line, real-time or sequential change-point detection, or change point monitoring. All these terms are almost synonymous. There are now many monographs on change point detection, e.g. Brodsky and Darkhovsky (1993), Gustafsson (2000), Chen and Gupta

(2011), Basseville *et al.* (2012) and Brodsky (2017). There is a continuous stream of publications within this field that focus on specific aspects, models and applications. With the awareness of listing a subjectively selected small sample, we cite Aminikhanghahi *et al.* (2019), Horváth *et al.* (2021), Zhang *et al.* (2021), Estrada Gómez *et al.* (2022), who also provide many references specific to the directions they advance.

Our purpose is to develop an effective method of detecting a fault in power grid streaming data. We will explain the relevant statistical concepts as we progress with the methodology development. In our approach, we use a moving window change-point detection technique.

This section is organized as follows. First, in Section 3.3.1, we focus on the irregular structure of the data and present our regularization approach. In Section 3.3.2, we describe the derivation of the methodology leading to change-point detection. We explore a few different approaches. The proposed methodology requires a suitable normalization of the data streams, which is described in Section 3.3.3. In Section 3.3.4, we describe the derivation of the threshold for decision-making, the approach for determining the final form of our fault detection methodology, and criteria for parameter tuning.

3.3.1 Regularization

The data capturing mechanism used by the NREL hardware and software provides unevenly spaced time series, so a regularization approach is needed. The 55 simulations share an almost identical data capture mechanism. Within a simulation, all 12 variables, like phase A current or three-phase voltage, are recorded at the same time. Thus, at any moment, if the value of one variable is recorded, the values of for the remaining 11 variables are recorded as well. The recording times are however not identical for the 55 simulations. Moreover, if different devices are used in a monitoring of a power grid, the assumption of the identical recording times for all variables might not hold. The regularization approach proposed below would work even in this more general setting.

To illustrate the irregularity of the measurement time points, we use a simulation with A phase fault at Bus 632 and analyze differences between subsequent time points. The histogram is presented in Figure 3.4. The minimum of differences is 0.00096s, the maximum is 0.00798s (more than a 700% difference). Most of the values group around 0.001, 0.002 and 0.003 seconds. For other simulations, the broad picture is similar, but the distributions of the time points are slightly different.

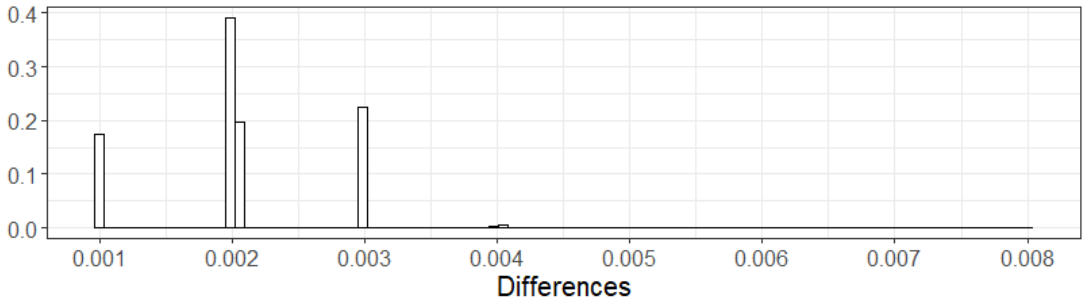

Figure 3.4: The unit area histogram of differences between subsequent time points of A phase fault at Bus 632 simulation.

The proposed regularization procedure consists of dividing the total time of a simulation into a set of fixed length subintervals. Each second is divided into D equal intervals of length $d = 1/D$ seconds. If D is too large, the intervals will be too small, some would contain no data points, eventually leading to additional noise in the data. If D is too small, intervals would contain many data points, leading to a loss of valuable information. In our analysis, we used $D = 500$, to reflect the most typical time points separations described above.

Denote by $t_1, t_2, \ldots$ the end points of the intervals, where $t_{i+1} - t_i = d$. Denote by X_t a variable, e.g phase A voltage, observed at time t. The value X_{t_j} is calculated as the average of the X_t with $t_j - d < t \leq t_j$. If such X_t do not exist (there are no observations in the interval $(t_j - d; t_j]$), we assign $X_{t_j} := X_{t_j - d}$. Recall that each simulation with a fault is 15 seconds long and starts at second 0, thus this procedure leads to a maximum of 7,500 (with $D = 500$) different equally spaced time points, and our time series are defined at times $\{0.002, 0.0004, \ldots\}$ seconds. If a fault is detected, we no longer need to continue to calculate the regularized X_{t_j}.

3.3.2 Moving window detection algorithm

We begin by introducing a convenient notation. Fix a simulation and consider all-time series $X_{(b',k',f')}$. We use the primers to distinguish the indexes identifying the available data steams from parameters of a statistical fault detection procedure. The index b' refers to a bus number, so $b' \in \{650, 632, 634, 671, 675, 680\}$. The index k' refers to a type of variable and k' can be either Voltage, Current, or Frequency. The index f' refers to a type of phase and can be either A, B, C, or three-phase. For example, $X_{(650,\text{Voltage,A})}$ refers to A-phase voltage measurements at Bus 650. and $X_{(650,\text{Voltage,A})}(t)$ is to A-phase voltage measurement at Bus 650 at time t. As described before, we have six buses, three types of variables and four types of phases. Our purpose is to determine the time of a fault using a suitable change-point detection technique based on a moving window. A moving window ensures that a detection statistic can be calculated in real-time, as it is based on only a limited number of observations, and that it adjusts to the most recent state of the system. The slow evolution of system readings does not indicate a fault of the type we are aiming to detect.

Denote by l the length of the moving window in seconds. We split the moving window into two intervals of lengths pl and $(1-p)l$ seconds. The first interval is used to evaluate the prior state. The second interval is used to evaluate the current state. It is natural to choose $p > 0.5$, as it provides a more stable evaluation of the prior state. This leads to $(1-p) < 0.5$. The shorter interval helps better capture the current state and respond to a fault quickly. However, the interval that is too short might put too much emphasis on noisy observations, thus one needs to find optimal values. We explain in the following how this is done. Assume that t_i is the current time point. Notice that if $t_i < l$, the moving window intervals cannot be built. Without loss of generality, we assume that $t_i \geq l$, because we do not expect a fault within a fraction of a second after the monitoring has begun. The moving window with the two subintervals is shown in Figure 3.5.

We now turn to the description of a general class of tests statistics, often referred to as detectors, that are based on the two subintervals of the moving window. Set

$$V_{(b',k',f')}(t_i) = \left[f(X_{(b',k',f')})(t_i) - \tilde{f}(X_{(b',k',f')})(t_i) \right], \tag{3.3.1}$$

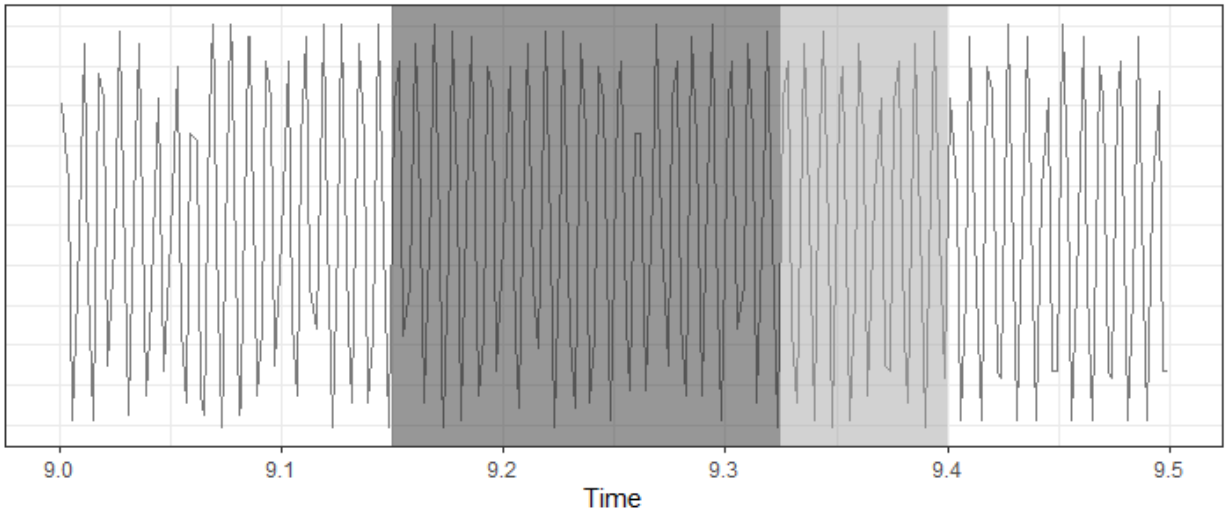

Figure 3.5: An example of the moving window using $l = 0.25$ seconds and $p = 0.7$. Here, the current time is $t_i = 9.4$ seconds. Dark grey area represents the interval of length pl and light grey - the interval of length $(1-p)l$.

where $f(X_{(b',k',f')})(t_i)$ is some function f computed from the trajectory $X_{(b',k',f')}$ using the points t_j, such that $t_i - l \leq t_j < t_i - (1-p)l$ (the first interval) and $\tilde{f}(X_{(b',k',f')})(t_i)$ is the same function on $t_i - (1-p)l \leq t_j \leq t_i$ (the second interval). Thus, for each combination of indexes, (b',k',f'), we acquire a statistics $V_{(b',k',f')}(t_i)$ at the current time point t_i.

The next step is to specify the functions f that can capture differences between the two intervals well and propose a method to combine the $V_{(b',k',f')}$ over all indexes. One should notice that the pool of choices is wide, and considering all of them is not practical. We considered several options for f and several ways to combine the $V_{(b',k',f')}(t_i)$ across the indexes. The performance of these different approaches is evaluated in Section 3.4. The choices of f we considered are listed in Table 3.1.

Table 3.1: The functions f considered in this study. The time t_i is the end point of the moving window. The specified functions are computed over the two intervals shown in Figure 3.5.

Name	$f(X_{(b',k',f')})(t_i)$	Definition
Mean	$\bar{X}_{(b',k',f')}$	The average of observations
SQMean	$\overline{X^2}_{(b',k',f')}$	The average of the squared observations
Range	$\max(X_{(b',k',f')}) - \min(X_{(b',k',f')})$	The range of observations
Median	$\text{median}(X_{(b',k',f')})$	The median of observations

The potential functions f in (3.3.2) can be viewed as features computed from the data on each of the two subintervals of the moving window. One can clearly come up with many functions, which could also be combined in various ways, e.g. by addition or multiplication. The functions listed in Table 3.1 are just the most commonly used statistics and, as we shall see, already lead to reliable detection. As illustrated in Figure 3.2, a fault is “seen" by different measurements at different buses differently, and it is not clear what differences between the two intervals one should look for. Generally, it is some sort of level shift, but the direction of the shift is unclear, and it may consist of several “sub-shifts". However, if a moving window starts to cover the fault, the differences between the two intervals should become visible for suitable functions f that must be determined empirically. For the streaming data, we think of the window as a filter through which that data stream passes. This filter must be tuned to detect a fault most effectively.

To combine the differences $V_{(b',k',f')}(t_i)$ into suitable real-valued statistics, we considered three methods, which result in statistics we denote as V_{mean}, V_{median}, and V_{trunc}. They are defined as follows. The detector V_{mean} is the average of the squared differences:

$$V_{\text{mean}}(t_i) = \frac{\sum_{(b',k',f')} \left\{V_{(b',k',f')}(t_i)\right\}^2}{\#(b',k',f')}. \tag{3.3.2}$$

Formula (3.3.2) shows that if one or more of the differences $V_{(b',k',f')}(t_i)$ between the two subintervals of the moving window is large, then $V_{\text{mean}}(t_i)$ will be large. On the other hand, one unusually large value of $V_{(b',k',f')}(t_i)$, that can be due, for example, to the instrument failure, can lead to a false detection. It can be argued that in smaller systems like the IEEE 13, if a fault happens, at least half of the measurements (collected variables) should react to it. Thus, alternatively, we considered the median:

$$V_{\text{median}}(t_i) = \text{median}_{(b',k',f')} \left\{V_{(b',k',f')}(t_i)\right\}^2. \tag{3.3.3}$$

A statistics that combines the characteristics of the mean and median is the truncated mean:

$$V_{\text{trunc}}(t_i) = \frac{\sum_{(b',k',f')\in Q} \left\{V_{(b',k',f')}(t_i)\right\}^2}{\#(b',k',f') \in Q}. \tag{3.3.4}$$

The set Q in (3.3.4) consists of the indexes (b',k',f') such that the values $(V_{(b',k',f')}(t_i))^2$ are between the $q/2$th and the $(1-q/2)$th quantiles. Here, q can be considered as a tuning parameter. Figure 3.6 illustrates the concept of truncated mean. In the following, all three methods V_{trunc},V_{mean} and V_{median} are generically denoted as V_{stats}, if the same considerations apply to all of them.

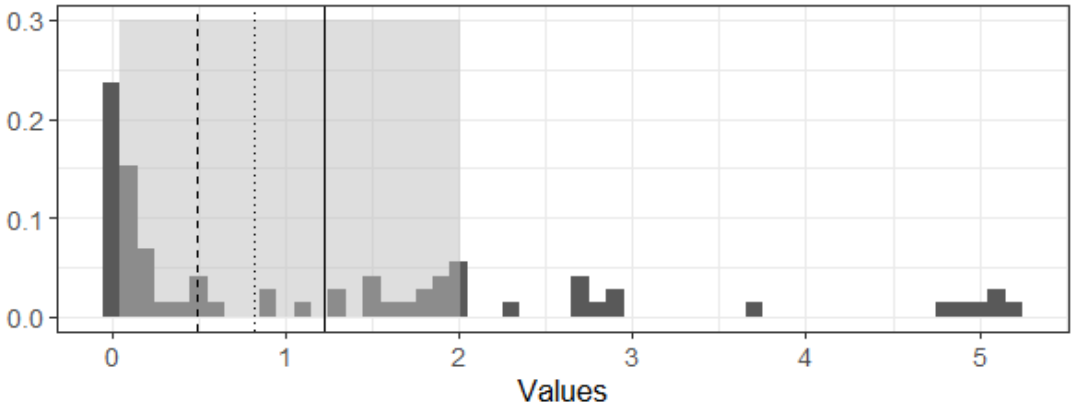

Figure 3.6: Histogram of $(V_{(b',k',f')}(t_i))^2$ at $t_i = 0.5$ seconds using Mean for f with $l = 0.25$ and $p = 0.95$. The data is from the simulation without a fault and were normalized as described Section 3.3.3 before computing the $V_{(b',k',f')}(t_i)$ values. The shaded area shows the central 60% of the $(V_{(b',k',f')}(t_i))^2$ values, corresponding to $q = 0.4$. The truncated mean is the average of observations in the shaded area. The solid line represents $V_{\text{mean}}(t_i)$, the dashed line $V_{\text{median}}(t_i)$, and the dotted line $V_{\text{trunc}}(t_i)$.

It is natural to set the value t_i at which $V(t_i)$ exceeds a suitable threshold τ as the estimated time of the fault. In principle, a slightly earlier time could be used, e.g. $(1-p)l$ seconds earlier, but this is a minor point and a delay of a small fraction of a second may be inconsequential. One could shift the detection time even more because there is a delay of about 0.25 s in the reaction of the test grid to a fault. Such a delay may however be different in a different grid and would generally be unknown. Adjustments of this type are easy to add to our procedure and do not affect the correct and false detection rates. Thus, we define the time of the fault as

$$t_f = \min\left\{t_j : V_{\text{stats}}(t_j) > \tau(t_j)\right\}, \tag{3.3.5}$$

where $\tau(t_j)$ is a threshold that is adjusted in real time. The methodology for the derivation of such a threshold is described in Section 3.3.4. First, we explain the normalization of the data streams in Section 3.3.3.

The performance of detection procedures based on different choices of $V_{\text{stats}}(t_i)$ as well as different options for $f(X)_{(b',k',f')}(t_i)$ is discussed in Section 3.4.

3.3.3 Normalization

The method described in Subsection 3.2 compares the two parts of the moving window (Figure 3.5) by evaluating V_{stats}. Thus for each variable included in the data, e.g. phase A voltage RMS, three-phase current RMS, etc., we compute separate $V_{(b',k',f')}(t_i)$ at time point t_i. Notice that each $V_{(b',k',f')}(t_i)$ has an equal weight in the formula for computing $V_{\text{stats}}(t_i)$. Figure 3.2 shows that different variables have different ranges, averages and variances. Thus, without further adjustments, our proposed methodology would favor some variables merely based on units of measurement and other irrelevant scale factors. Additionally, if units of measurements or other aspects of data collection are changed, this can lead to completely different different performance of our method. For example, instead of measuring a current using the RMS, one could decide using the peak to peak range, but leave voltage and frequency measures unadjusted, possibly getting very different results in V_{stats}. Using amperes instead of kiloamperes would basically eliminate all variables except the current. At this point, we do not know which variables are important for fault detection, and we see that they all react to a fault, so it is judicious to give them equal weight in some suitable sense.

To ensure that each bus, type of variable (voltage, current, and frequency), and phase has an equal weight in the equations of Section 3.3.2, we use normalization. Traditionally, normalization uses an estimate of standard deviation and of the mean to calculate a Z score. In the case of streaming data, one cannot take the mean and standard deviation estimates of the whole data. Moreover, the potential for a fault implies that measurements over a period considered fault-free should be used. There is no perfect and widely accepted solution. After initial exploration, we propose using the following moving window technique.

We have observations $X_{(b',k',f')}(t_i)$ at time point t_i. We compute the sample mean and the sample standard deviation over the time period $t_i - l \leq t_j < t_i - (1-p)l$ and denote them as $m(X_{(b',k',f')}(t_i))$ and $\mathrm{SD}(X_{(b',k',f')}(t_i))$. The interval over which these statistics are computed is the same as the first interval in (3.3.2). The normalized value $Z_{(b',k',f')}(t_i)$ at the point t_i is calculated as:

$$Z_{(b',k',f')}(t_i) = \frac{X_{(b',k',f')}(t_i) - m(X_{(b',k',f')}(t_i))}{\mathrm{SD}(X_{(b',k',f')}(t_i))}. \tag{3.3.6}$$

The normalized values $Z_{(b',k',f')}(t_i)$ depend on the choice of p and l. Figures 3.7 contain examples of normalization applied to Bus 632 voltage trajectories with $l = 0.25$ s and $p = 0.9$. *In the methodology of fault detection, we use the $Z_{b',k',f'}$ as input (instead of the $X_{b',k',f'}$).*

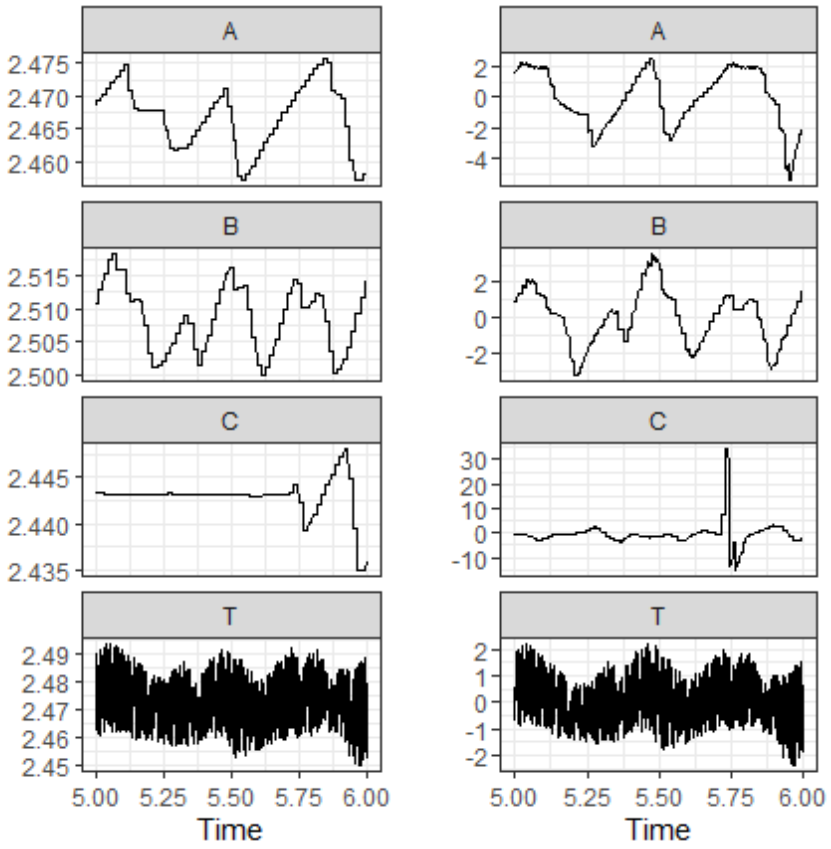

Figure 3.7: Illustration of the normalization procedure applied to Bus 632 voltage data (no fault). The right four panels contain the regularized data (original data with irregularity removed) and the left four panels show the corresponding normalized values using $l = 0.25$ second and $p = 0.9$. Note the different vertical scales between the left and right columns.

3.3.4 Threshold determination methodology

The performance of the method described in Section 3.3.2 depends on how well we can acquire the dynamic threshold τ. We want to determine the τ that balances two criteria: 1) if there is no fault, false alarms should be rare, 2) if there is a fault, it should be detected with a large probability and the delay of detection (time of detection - time of fault) should be small. A larger τ ensures that criterion 1) is met, and a smaller τ ensures that criterion 2) is met. There is no guidance on how to balance these two criteria for power grid streaming data. In this section, we explain our approach to the determination of τ. Our method relies on several tuning parameters, and it is a priori not clear which of them will work best for our data. The winners are determined after the application of our approach in Section 3.4.1.

First, we explored how different tuning parameters affect the $V_{\text{stats}}(t_i)$ values. The properties we are looking for are small variance and moderate percentage change between $V_{\text{stats}}(t_i - 1)$ and $V_{\text{stats}}(t_i)$ if there is no fault. Such properties would lead to a robust threshold calculation based on a moving window. The tuning parameters are:

- the length of the moving window l,
- the proportion parameter p,
- in case of V_{trunc} - cut off parameter q.

Also, we separately discuss the differences between the V_{trunc},V_{mean}, V_{median} values. (The median appears in two different contexts: in Equation (3.3.3) and in the definition of the statistic $f(X)_{(b',k',f')}(t_i) = \text{median}(X_{(b',k',f')}(t_i))$.) For each combination of tuning parameters, we explore for different statistics - Mean, SQMean, Range, Median, cf. Table 3.1. The properties we are looking for are small variance and moderate percentage change difference between $V_{\text{stats}}(t_{i-1})$ and $V_{\text{stats}}(t_i)$ if there is no fault at time point t_i. On the other hand, some variability in V_{stats} should be expected, as otherwise, the detector would not respond to changes and the detection of a fault would be practically impossible.

While many threshold formulas are reasonable, to narrow down our choices, we considered 3 different ways to calculate τ. To display the formulas, we first introduce the interval

$$I_1(t_i) = \{t_j : t_i - l \leq t_j \leq t_{i-1}\}.$$

The above interval resembles the interval considered in the $V_{\text{stats}}(t_i)$ calculations, but it does not contain the final time point t_i; if a fault happens, its effects should be noticed comparing current $V_{\text{stats}}(t_i)$ value to a threshold based on previous values. We considered:

1. $\tau_1(t_i) = 3\text{max}_{t_j \in I_1(t_i)} V_{\text{stats}}(t_j)$;
2. $\tau_2(t_i) = \text{max}_{t_j \in I_1(t_i)} V_{\text{stats}}(t_j) + 3\text{range}_{t_j \in I_1(t_i)} V_{\text{stats}}(t_j)$;
3. $\tau_3(t_i) = \text{max}_{t_j \in I_1(t_i)} V_{\text{stats}}(t_j) + 3\text{SD}_{t_j \in I_1(t_i)} V_{\text{stats}}(t_j)$.

Using the coefficient 3 is motivated heuristically by the 99.7% quantile of the normal distribution. In the following, all three thresholds τ_1, τ_2, τ_3 are referred to as τ. After choosing tuning parameters, V_{stats}, and a procedure for computing τ, the fault is signaled at the first time point t_j such that $V_{\text{stats}}(t_j) > \tau(t_j)$, as expressed in equation (3.3.5).

3.4 Results

In Section 3.4.1, we explore the performance of the methodology derived in Section 3.3 assuming all data streams are available. Based on the observed performance, we select the final form of the method. In Section 3.4.2, we investigate to what extend our method is robust to the elimination of certain data streams or buses at which they are measured. The first consideration may be viewed as a further tuning of our method because there may be sufficient information only in one of two data stream, and it may not be necessary to use all streams. This is however not clear a priori. The second consideration is very practical because if all buses react to a fault, it may be enough to place measurement devices only at some of them. Again, the answer is not clear a priori and the question must be investigated.

We use the following quantities for the assessment:

F_1 - Fraction of simulations in which a fault is detected over the first 10 seconds. This quantity is similar to the size of a statistical significance test or type I error.
F_2 - Fraction of simulations in which a fault is detected over the whole 15 seconds. This quantity is similar to power under an alternative or 1 minus type II error.

In statistical significance tests, F_1 equal to 5 or 1 percent would generally be acceptable. Since in power grids a false alarm may be expensive to investigate, we may target a different value of F_1. Even $F_1 = 0$ might be reasonable. We proceed analogously as in the traditional Neyman–Pearson paradigm. We first determine method parameters that give satisfactory F_1. From those, we select those that give the best F_2.

3.4.1 Detection based on complete data

In accordance with the plan outlined above, we first applied our method to a simulation that does not contain a fault. We tested different combinations of tuning parameters and different choices of V_{stats} and f. We noticed that different values l do not have a large impact on F_1. One should not choose the value that is too small, as it increases variance in V_{stats} and leads to false detections. The value we propose to use is $l = 0.25$ s, but one can choose relatively close values and get basically identical results. Regarding the selection of p, our statistical experiments showed that p should be close to 1. We have observed better properties of V_{stats} when the first interval of the moving window is longer, and the second interval is relatively short. The value we propose to use is $p = 0.95$, but one can choose relatively close values, like 0.9, and similar results. Notice that with $l = 0.25$ and $p = 0.95$, the effective interval used to detect the faults has the length of $(1-p)l = 0.0125$ s. The explorations regarding V_{stats} revealed that using V_{mean} leads to many false-detection cases. Random variability in the data combined with the sensitivity of the standard error to large and small values leads to high volatility in V_{mean} values, thus resulting high percentage change from $V_{\text{stats}}(t_{i-1})$ to $V_{\text{stats}}(t_i)$. On the other hand, V_{median} solves the aforementioned problems, but introduces some other issues; it is too conservative, as it is calculated using the middle of input values. It is basically not sensitive to a change in 50% of the largest input values. This method

had little variation and values proved to be insensitive even if there are some changes in the data. This leads to undetected faults. The best results were obtained using V_{trunc}. This statistic proved to be less volatile than V_{mean}, but more flexible than V_{median}. The value we propose to use is $q = 0.1$ (90% of the "middle" data). *To summarize, we recommend*

$$l = 0.25 \text{ s}, \ p = 0.95, \text{ and } V_{\text{stats}} = V_{\text{trunc}} \text{ with } q = 0.1. \tag{3.4.1}$$

All tables that follow are produced using these settings.

The above settings produced the best overall results for the faults we considered. With additional knowledge of the type of fault, it is possible that slightly different settings might be optimal.

Next, for each proposed threshold τ, and each statistic f in the Table 3.1, we evaluated how many times the detector was triggered over the course of the simulation without a fault (Table 3.2). The best results (0 false detections) were achieved by using Mean and Range for $f(X_{(b',k',f')})$ and either τ_1 or τ_2. We observed no false detections in either of these four cases. However, even in the worst cases we observed only 21 false alarms out of potential 17,500 time points where they could be signaled. As note above, in power grids false alarms are expensive, and we have found methods that basically do not produce them.

Table 3.2: Counts of false detections over the simulation with no fault using settings (3.4.1). The last column shows the count of false detections over the whole length of the interval (35s) which contains $35 \cdot 500 = 17,500$ regularized time points.

f	τ	Count	f	τ	Count
Mean	τ_1	0	Range	τ_1	0
Mean	τ_2	0	Range	τ_2	0
Mean	τ_3	4	Range	τ_3	5
Median	τ_1	3	SQMean	τ_1	3
Median	τ_2	3	SQMean	τ_2	2
Median	τ_3	21	SQMean	τ_3	21

We now show how the methods with the best parameters perform over the simulations with faults (55 simulations). The results are presented in Table 3.3. Notice that three of four combinations showed perfect results in terms of fault detection ($F_1 = 0$ and $F_2 = 1$). The combination of τ_2 and Mean gave results of $F_2 = 0.98$ (in one of the simulations, the fault were missed). The fractions F_1 of false detections were computed using the first 10 seconds because we know that there were no faults in the initial 10 s.

Table 3.3: Fault detection evaluation over 55 simulations with a fault using setting (3.4.1). Results are presented for the combinations of f and τ that showed the best results in Table 3.2.

f	τ	F_1	F_2
Mean	τ_1	0	1.00
Mean	τ_2	0	0.98
Range	τ_1	0	1.00
Range	τ_2	0	1.00

Allowing for uncertainty associated with using a database of 55 faults, our results show that any of the four combinations in Table 3.3 can be used. For additional certainty, One can choose to use these four combinations simultaneously. Additionally, we tested the timing of the fault detection to see if there exist any differences between these four combinations. The results show that τ_1 and τ_2 have no effect on the detection timing and are identical for both f choices. With additional information about the type of fault, e.g. its impedance or path, it might be possible to recommend an optimal τ most suitable for the specific faults. In 29% of the simulations, Range led to a faster fault detection than Mean, but only by 0.002s. In the other 71% simulations, both statistics gave identical time of the fault.

3.4.2 Detection with partial data streams

In this Section, we explore how robust our methodology is to the reduction of the available data streams. We consider only the best parameters settings, those defined by (3.4.1) and Table 3.3. Recall that the data streams are indexed by the triples (b', k', f'), where b' is the bus number, k' is the variable (voltage, current, frequency) and f' is the phase (A, B, C, 3 phase). There are a large number of ways in which data streams could be restricted. To provide information in some systematic way, we restrict each of the three coordinates separately and investigate what the impact of such a data reduction is. In Section 3.4.2, we consider the practically most important setting of measurement devices placed only at some buses. In Sections 3.4.2 and 3.4.2 we investigate, respectively, what happens if only some variables or some phases are used. Information of this type is also relevant because it might, for example, be the case that it is enough to use only voltage. Since the data streams we use have not be explored from such angles, the answers are not a priori clear. As a byproduct of the investigations in this section, we may be able to give preference to some of the methods in Table 3.3.

Limited buses

In Table 3.3 data from all buses were used. In this section, we investigate what happens if data from subsets of cardinality K of the set $\{650, 632, 634, 671, 675, 680\}$ are used. If $K = 6$, there is only one subset, if $K = 5$, there are 6 subsets, if $K = 4$, there are 15 subsets, etc. (number of combinations). For example if $K = 2$, a possible combination is $\{650, 632\}$, and we have 15 combinations in total for $K = 2$. Thus for $K = 2$, we can compute F_1 and F_2 using $15 \cdot 55 = 825$ cases. The results are displayed in Table 3.4. Notice that for smaller K, the F_1 values are higher and F_2 values are smaller. With $K \geq 3$, the F_1 stays at 0, while F_2 varies from 0.97 to 0.99. For low K values Mean gives better results in terms of F_1 than Range. For $K \geq 3$, Range gives better results than Mean with $F_1 = 0$ and larger values of F_2. *The main conclusion is that to ensure no false alarms, it is enough to use any three of the six buses, but to ensure that each fault is detected all six buses must be used.*

Table 3.4: Faults detection evaluation using complete data streams from K buses.

f	τ	K	F_1	F_2	K	F_1	F_2
Mean	τ_1	1	0.07	0.93	4	0.00	0.97
Mean	τ_2	1	0.00	0.88	4	0.00	0.96
Range	τ_1	1	0.08	0.95	4	0.00	0.98
Range	τ_2	1	0.12	0.95	4	0.00	0.98
Mean	τ_1	2	0.00	0.94	5	0.00	0.98
Mean	τ_2	2	0.00	0.92	5	0.00	0.98
Range	τ_1	2	0.12	0.96	5	0.00	0.99
Range	τ_2	2	0.22	0.96	5	0.00	0.99
Mean	τ_1	3	0.00	0.95	6	0.00	1.00
Mean	τ_2	3	0.00	0.94	6	0.00	0.98
Range	τ_1	3	0.00	0.97	6	0.00	1.00
Range	τ_2	3	0.00	0.97	6	0.00	1.00

Limited variables

We now explore if it is necessary to consider all three variables. Table 3.5 shows the results of our procedure based on only one of the three available variables. We see that *using only current gives perfect results*, even slightly better than using all three variables, cf. Table 3.3.

Table 3.5: Performance of procedures based on single variables

f	τ	Variable	F_1	F_2
Mean	τ_1	Frequency	0.00	0.96
Mean	τ_2	Frequency	0.00	0.93
Range	τ_1	Frequency	0.00	0.96
Range	τ_2	Frequency	0.00	0.96
Mean	τ_1	Current	0.00	1.00
Mean	τ_2	Current	0.00	1.00
Range	τ_1	Current	0.00	1.00
Range	τ_2	Current	0.00	1.00
Mean	τ_1	Voltage	0.00	0.91
Mean	τ_2	Voltage	0.00	0.87
Range	τ_1	Voltage	0.02	1.00
Range	τ_2	Voltage	0.04	1.00

Limited phases

We finally explore how our procedure performs if only one phase measurement is used. The phases are A, B, C and 3 (three-phase measurement). The last case revers to a single value derived from all three phases, for example the sum of currents flowing in phases A, B, C at a given time. The results displayed in Table 3.6 are not clear-cut. The performance is perfect for f =Mean and phase=3, but replacing f =Mean with f =Range gives a lot of false alarms. Using a truly single phase, A, B or C gives practically no false alarms but can miss up to 10% of faults.

Table 3.6: Performance of procedures based on single phase measurements. Phase 3 refers to the simultaneous measurement of all three phases reported as a single number.

f	τ	Phase	F_1	F_2	Phase	F_1	F_2
Mean	τ_1	A	0.00	0.91	C	0.00	0.91
Mean	τ_2	A	0.00	0.91	C	0.00	0.91
Range	τ_1	A	0.00	0.93	C	0.04	0.95
Range	τ_2	A	0.00	0.93	C	0.04	0.95
Mean	τ_1	B	0.00	0.93	3	0.00	1.00
Mean	τ_2	B	0.00	0.91	3	0.00	1.00
Range	τ_1	B	0.00	0.93	3	0.38	1.00
Range	τ_2	B	0.00	0.93	3	0.24	1.00

3.5 Summary and an algorithm

Algorithm 3 Detection procedure using f =Mean, threshold τ_1 and 3 phase current

Input: Trajectories $X_{(b',k',f')}$ with k' =current and f' =3 phase

Output: Fault = 1 or 0 (1 if fault detected); t_f (time of fault) **Initialization**

1: Set $l = 0.25$ s, $p = 0.95$, and $V_{\text{stats}} = V_{\text{trunc}}$ (3.3.4) with $q = 0.1, d = 1/500$. Fault= 0 Set $t_i := t_i + d$ Apply regularization to each $X_{(b',k',f')}(t)$, $t \in [t-2l, t_i]$ (Section 3.3.1) Find $Z_{(b',k',f')}(t_j)$ (3.3.6) for $t_j \in [t_i - l, t_i]$ using regularized $X_{(b',k',f')}(t_j)$ Find $V_{\text{trunc}}(t_j)$ (3.3.4) for $t_j \in [t_i - l, t_i]$ using $Z_{(b',k',f')}(t_j)$ and f =Mean Calculate $\tau_1(t_i) = 3\max_{t_j \in I_1(t_i)} V_{\text{trunc}}(t_j)$ (Section 3.3.4) $V_{\text{trunc}}(t_i) > \tau_1(t_i)$ Set Fault= 1 and $t_f := t_i$

We have proposed a methodology for fault detection in a small power distribution system based on a suitably developed moving window change-point analysis technique. We investigated the properties of our technique and determined optimal parameter and other settings, including data stream regularization and normalization as well as dynamic threshold selection. Investigations reported in Section 3.4, indicate that three phase current is the most important variable. In fact, a simple version of our procedure that uses only three phase current at all six buses produces perfect results; there are no false alarms and all faults are detected. This is true for both f =Mean and f =Range and for thresholds τ_1 and τ_2. Other options also produce perfect or nearly perfect results

for our data set of 55 faults, showing that the general approach is sound. Algorithm 3 summarizes the whole procedure with f =Mean, threshold τ_1 and three phase current. It is possible that some false alarms and missed faults might occur if a grid with a different topology is used, but our methodology shows how to find a nearly perfect algorithm for any grid.

Our research proposes a general, data-driven, statistical approach to instantaneous and correct detection of faults in a subgrid of a distribution system that is based on high-frequency measurements, like those generated by PMUs. This approach looks ahead to the increasing penetration of renewable energy sources that generate more random variability and bidirectional power flows. Our methodology is obviously not a definite engineering solution, but it shows a potential of data-driven, statistical approaches to fault detection and develops a scalable paradigm.

Extensions of this work might provide further useful insights. More complex grid topologies could be explored. Fault behavior can vary based on the fault path impedance and the source impedance. For this work, we did not vary the source impedance and we did not vary the fault impedance either. Events such as a direct on line (DOL) motor start can cause under voltage for more than a second. Learning on a more extensive data base of faults and various transient but normal events can refine our approach.

Chapter 4

Graph neural networks for the localization of faults in a partially observed regional transmission system

4.1 Introduction

Electric power systems are one of the most crucial infrastructures of any nation or economic area. The functioning of any modern economy and society depends on an efficient power transmission system. Cyberattacks that make information about parts of the power system unavailable or corrupted are a distinct possibility even if these parts are not physically damaged. Natural disasters, like floods or hurricanes, military action or terrorist acts can damage parts of the power grid and cut off information from affected sensors.

Our objective is to show that Graph Neural Networks (GNNs) can be employed to recover enough information to ensure that fault detection diagnostics can be performed even if no information is streamed from parts of a regional transmission system. We show how these deep learning networks must be constructed and explain which forms of GNNs work well and which do not. Our conclusions are based on statistical analysis of an exceptionally large database of faults simulated in a realistic model of a regional transmission system. Before presenting the details in the following sections, we review related statistical and engineering work and place our contribution within this large body of very active research.

Faults in power systems cause excessive currents and pose safety threats to people and property, and may cause major fires with substantial economic and social impacts. Therefore, there is an increasing interest in fault detection and localization to reduce the damage to the system and inconvenience to customers. This motivates the development of new methods to complement, or potentially replace, traditional methods. Data-driven methods can now be easily implemented due to the installation of phasor measurement units (PMUs), (Zhang *et al.*, 2012; Xu *et al.*, 2014;

Kokoszka *et al.*, 2023), among many others. In principle, the entire power system can be measured and monitored by using PMUs on each electric grid bus. However, such placement could be costly and prone to increased device failure rates as shown by Shafiullah *et al.* (2022). Thus, the research on fault detection and localization without the assumption of full network observability has principal importance. This paper concentrates on the issue of locating a faulted line in a transmission network under partial network observability.

Fault detection and localization under partial observability of the power grid using machine learning have been the subject of study in recent years. Li *et al.* (2019a) proposed a methodology to detect a faulted line employing classification based on Convolution Neural Networks (CNN) using bus voltages and discussed the optimal PMUs placement. The study by Afonin and Chertkov (2021) primarily focuses on using deep learning classification methods to locate faults in a network under partial observability. They adopt a similar setting as (Li *et al.*, 2019a), but investigate a wider range of models: linear regression (one-layer Neural Network), feed-forward Neural Networks, AlexNet, and Graph Convolutional Neural Networks. All of these models are tuned in to predict the location of the faulted line. The performance of the models depends upon the location of PMU devices. Zhao and Barati (2021) proposed a methodology for fault localization in distribution networks based on a novel feature vector design and CNN classification. The feature vector is built using the the differences in voltage and current before and after the fault, the phase angle's difference of each phase before and after the fault, and zero, negative and positive sequences for phase voltages and currents. Results were tested on a real power distribution feeder with 25 possible locations for the fault. Han *et al.* (2022) proposed to locate the faults by converting multichannel electrical signals to similarity images. Using CNNs, which are traditionally used for image classification, the authors apply algebraic operations to power signals and convert them to images. The model was tested on a fully observed 24-bus system. The authors argued that the model they proposed was robust to changes in the topology of the power grid. (Devi *et al.*, 2018) and Zainab *et al.* (2019) proposed a similar method as (Li *et al.*, 2019a) but with additional utilization of electrical area information. As the above review shows, the work done so

Figure 4.1: The principal diagram illustrates the proposed approach to combine reconstruction and some fault localization that effectively works under full grid observability in order to achieve fault localization under partial observability.

far considers the limited availability of measuring devices (PMUs) to build classification models for fault localization. Some work considers the approach where one trains the model on limited PMUs and examines its performance (Devi *et al.*, 2018; Li *et al.*, 2019a; Afonin and Chertkov, 2021). There are several limitations to the above classification based approaches. First, if a particular measuring device fails, the model needs to be re-trained. Second, classification models provide only 0-1 answers without additional information about the unobserved readings. Third, these models do not incorporate the fact that with full PMU data availability, the current fault detection/localization methods work with high accuracy (Salehi-Dobakhshari and Ranjbar, 2014; Furse *et al.*, 2021). We emphasize that classification based methods perform very well when trained on rich and appropriately selected data. However, during training, they must know the location of the fault, even if the fault occurs in an area of the grid that is not covered by measuring devices.

Rather than proposing a new fault localization method applicable under partial network observability, this study aims to bridge the gap between limited PMU availability and fault localization methods that are highly accurate under the assumption of the full PMU coverage of the power grid. We propose deep learning networks that can reconstruct data at locations where data are missing. With such an approach, any method that works well under full grid observability can be applied. Our approach is thus akin to data imputation, but with a specific focus on fault localization. It is summarized by the following flow diagram:

The topic of missing data imputation in power grids has gained attention in recent years. Zhu and Lin (2021) propose a method that utilizes spatio-temporal correlations for imputing missing PMU data. This method is applied in situations where device malfunctions and communication

failures lead to poor PMU measurements and data loss. The approach utilizes not only past information but also a global and local spatial perspective in order to achieve a higher accuracy of data imputation. Foggo and Yu (2022) fill in missing values through the use of two components: a non-dynamic component that is predicted using past data and a dynamic component that is inferred from all other available PMUs. The proposed model corrects past data and incorporates events data that can be inferred from all available PMUs data for the modulus of voltage. However, these methods only address the missing data scenarios after the failure of the PMUs and assumes that the past data is available. Dynamic state estimation is another area where significant progress has been made (Jakir and Rahnamay–Naeini, 2021; Park *et al.*, 2023). These authors used deep learning methods and a large amount of diverse data to learn the relationship between power grid nodes under normal operating conditions. Our work has a different focus. We use as little data as possible, namely the modulus of voltage, to reconstruct trajectories *during a fault* with a focus of using them in statistical fault localization methods. Our data are described in Section 2.2.

We are aware of only one study in the field of power grids that concentrates on employing reconstructed trajectories in certain fault detection techniques, namely Li and Deka (2021) who developed a physics-informed learning approach for detecting high impedance faults. They employed auto-encoders for feature learning and utilized unlabeled data for training, which is similar to our methodology. Our research focuses on utilizing Graph Neural Networks for reconstructing missing trajectories, with specific applications in fault localization. We offer techniques for enhancing the accuracy of the reconstruction process and propose models that can be utilized in various data availability scenarios. Most research in this area has relied on labeled data (fault locations are known). Our work addresses the challenge of localizing faults in situations where obtaining labeled data is difficult or expensive. We propose employing GNNs to impute data trajectories at buses without PMU devices during a fault in a power grid, and using this data in an existing fault detection technique proposed in Kokoszka *et al.* (2023). Different techniques could be used, but they are difficult to employ in our experimental test bed of the Western Regional grid due to the lack of deployable code. We explore the capabilities and limitations of GNNs by us-

ing two different scenarios, one where we assume access to full PMU data during training of the GNN model and another where only limited PMU data is available during training. We consider good results under the first scenario as our baseline, and aim to achieve even better results under the second scenario, where data at buses without PMU devices is not available at all, even during training. We explore different feature transformations, loss functions, strategies for optimal PMU placement, and data availability scenarios using only unlabeled data. We present our results and discuss the means to implement our methods. Moreover, our PMU data reconstruction methodology contributes to the initiatives to mitigate cyberattacks on power grids, as we show that corrupted PMU readings can be replaced by imputed readings from our model. Broadly speaking, our work falls under the category of artificial intelligence techniques in power grids.

The paper is organized as follows. Section 4.2 introduces the the transmission system and the grid data we work with. In Section 4.3, we formulate the problem and focus of the paper in greater detail. In Section 4.4, we describe the methods we propose for data reconstruction using GNNs. We examine various of loss functions, optimal PMUs placement, and success rates for fault localization. Section 4.5 reports the performance of various versions of our method, while Section 4.6 summarizes our contribution and main conclusions.

4.2 Data Description

As in Chapter 2, we work with data generated using the miniWECC system that is a reduced-order dynamic model of the Western Electricity Coordinating Council system. The minniWECC has enough complexity to reflect the relevant properties of the full Western Interconnection's bulk power system, which serves over 80 million customers in 14 U.S. states and two Canadian provinces. The simulations conducted with the minniWECC model have been used to work on various power system mode estimation and event detection algorithms which are now adopted for control room applications, see e.g. (Follum *et al.*, 2017; Trudnowski *et al.*, 2013; Byrne *et al.*, 2016). A more detailed description of the miniWECC can be found Trudnowski *et al.* (2013). A simplified one-line diagram of the minniWECC is given in Figure 4.4 in Section 4.5. Only high

voltage lines are shown to make the graph readable. We use a version of the minniWECC that consists of 158 lines between 122 buses (nodes). Some buses can be physically connected by more than one line, but we treat all these lines as a single connecting line because measurements at end buses cannot distinguish between them. Thus, each connection $l \in \{1,\ldots,158\}$ has two buses $(i,j) : i,j \in \{1,\ldots,122\},\ i \neq j$. We consider only pairs (i,j) for which there is a line between buses i and j.

We simulated a large number of faults in the minniWECC using the Power System Toolbox (PST), Cheung *et al.* (2009), which is based on MATLAB. The description of the fault generating mechanism in the next paragraph is technical and requires some background in power systems. The essence is that we generated a very large number of faults that can realistically occur in a regional transmission system. Our statistical approach does not require the knowledge of the fault type or its characteristics. *If the fault occurs closer to bus i, we call i the near end bus and j the far end bus.*

The faults were generated using the switching condition matrix. The values of the zero sequence impedance, $z0$, and the negative sequence impedance, zn, were generated randomly for each simulation. System characteristics imply that they can be generated as random variables uniformly distributed on the interval [0.0004, 0.189] ohms. Using the random values of $z0$ and zn, four types of line faults can be generated: line-to-line (LL), line-to-ground (LG), line-to-line-to-ground (LLG), and three-phase (TP) faults. Furthermore, the location of the fault on a line between buses i and j can also be randomly simulated. Ignoring the randomness of $z0$ and zn, 1,264 unique faults (158 lines $\times$ 4 types $\times$ 2 ends). For each random pair $(z0, zn)$, each of the 1,264 options is chosen randomly. We added ambient noise to each simulation using MinniWECC parameters described in Trudnowski *et al.* (2013), which also increases the randomness, but not crucially, as will become apparent in the following. The total number of distinct faults that can be simulated is practically unlimited.

The simulated data consists of 120 measurements per second for 10 seconds. Define the time resolution as $\delta = 1/120$. For this paper, we only consider $\frac{32}{120}$ s before the fault and $\frac{32}{120}$ after

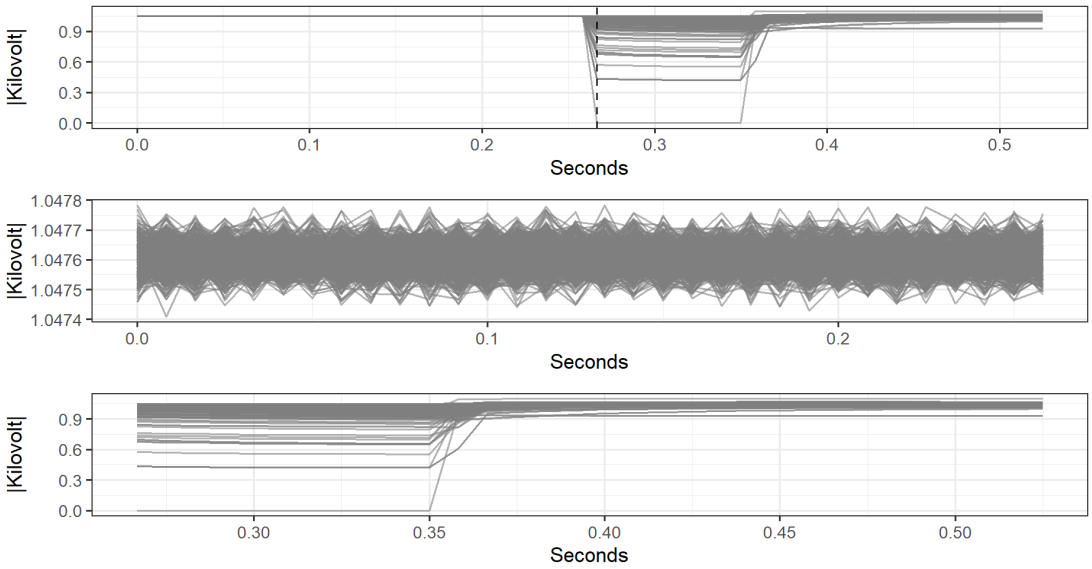

Figure 4.2: Randomly selected modulus of voltage readings at bus 105. The three graphs represent different time stages of simulations. The upper panel represents the whole trajectories, while the lower two represent trajectories before the fault, and after the fault. Notice the different voltage scales at the different stages.

(in total, 64 time stamps). We assume that each simulation starts at $t = 0$, the fault is applied at $t = \frac{32}{120}$s, and data is recorded up to $t = \frac{64}{120}s$. We set the time of the fault as $t_f = \frac{32}{120}$. The considered length of simulation is equal approximately equal to 0.5 seconds. We use only the modulus of voltage, which is sufficient to localize faults.

Figure 4.2 shows 500 randomly chosen readings at bus 105. Figure 4.3 shows responses at all buses to three faults applied between bus 97 and bus 66, with bus 97 being a near end bus. It shows how all buses see the faults. The responses at all buses exhibit similar patterns, but there is a possibility to identify the faulted line, at least the near end bus. More detailed visual analysis reveals that faults of certain types are more difficult to localize. The three phase faults show much less variability between the lines that would allow to identify the faulted line. Our methods are applied without assuming any knowledge of the fault type.

4.3 Problem Formulation

In this section, we introduce suitable notation, define the problem, describe the tasks, and explain the challenges that arise. We model the system as a graph $\mathbf{G} = (\mathbf{A}, \mathbf{L})$, where $\mathbf{A} =$

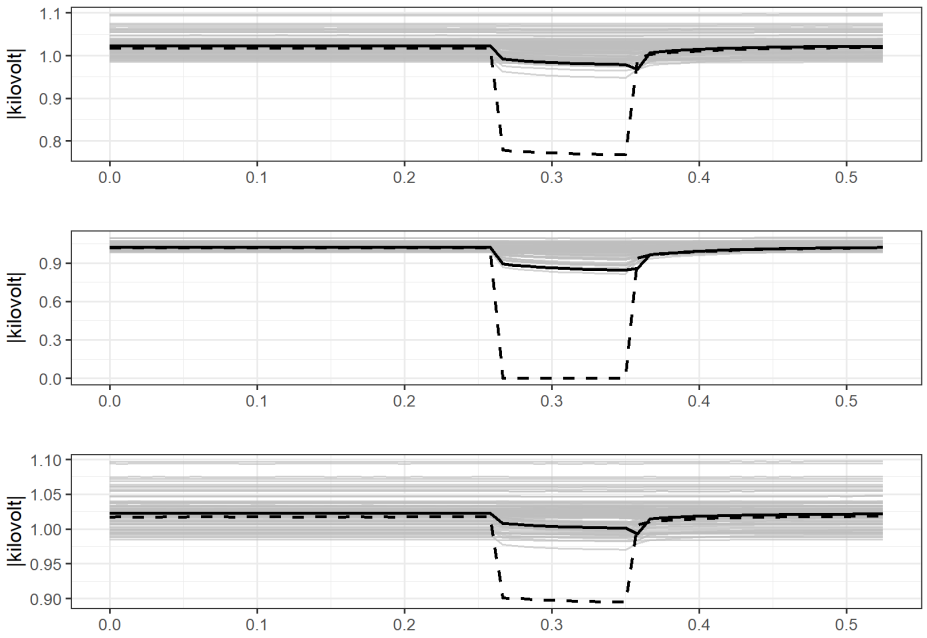

Figure 4.3: Responses at all buses to three faults between bus 97 (near end) and bus 66. Responses at bus 66 are highlighted in solid black and those at bus 97 in dashed black. The three panels represent different types of faults with different parameters.

$\{1,2,\ldots,122\}$ is the set of buses, including both those with PMU devices and those without them, and $\mathbf{L}$ is the set of lines connecting the buses. Notice that if $(i,j) \in \mathbf{L}$, then $(j,i) \in \mathbf{L}$. The neighborhood of bus i is defined as the set $\mathbf{B}_i = \{j : (i,j) \in \mathbf{L}\}$ of buses that are connected to i by a line. We assume that the power grid $\mathbf{G} = (\mathbf{A}, \mathbf{L})$ is fixed.

We assume that a fault occurs at an unknown line $l_0 \in \mathbf{L}$. We consider a prediction $\hat{l}_0$ of the faulted line to be correct if $\hat{l}_0 = l_0$. Additionally, we consider a localization to be partly correct if $\hat{i}_0 = i_0$, where $\hat{i}_0$ is the predicted near end bus of the faulted line $\hat{l}_0$. The localization failure ratio is defined by the ratio of incorrect predictions and the total number of simulations.

Our framework assumes that PMUs are available only at a certain subset of buses, which we denote by $\mathbf{K} \subset \mathbf{A}$. The count of buses in $\mathbf{K}$ is denoted by $K := |\mathbf{K}|$. At buses in $\mathbf{K}$, we have records of modulus of voltage during a fault, with m data points before the known fault time t_f and m points after. This leads to a feature matrix $\mathbf{X}_K \in \mathbb{R}^{2m \times K}$. The set of unobserved buses is denoted as the $\mathbf{U} = \mathbf{A} \setminus \mathbf{K}$ with $|\mathbf{U}| = U$. The trajectories at these buses are denoted as $\mathbf{X}_U \in \mathbb{R}^{2m \times U}$. Denote by $\mathbf{X} \in \mathbb{R}^{2m \times (U+K)}$ the dataset under full availability of measuring devices.

The objective is to find a function $f_\theta(\mathbf{X}_K) = \widehat{\mathbf{X}}_U$ that satisfies two conditions:

1. minimizes a suitable loss function $\ell(\mathbf{X}_U, \widehat{\mathbf{X}}_U)$;
2. minimizes the difference in fault localization accuracy based on $\widehat{\mathbf{X}} = \mathbf{X}_K \cup \widehat{\mathbf{X}}_U$ (the available and reconstructed values) relative to the same method based on $\mathbf{X}$ (the full dataset).

Objective 2 can be seen as applying a discrete, problem focused loss function. The optimal reconstruction should produce trajectories that lead to fault localization that is a close as possible to the localization based on the completely observed system. To our knowledge, no previous work has addressed the task of training the model for imputing trajectories during a fault without using information of the fault location with the intention to employ the trained model for fault localization. We use the method of Kokoszka *et al.* (2023), but any other method for which code applicable to minniWECC is available could be used. Our statistical evaluation approach could also be used in different testbeds.

We consider two scenarios to obtain f_θ: (1) The training process of f_θ uses $\mathbf{X}_U$, (2) The training process of f_θ does not use $\mathbf{X}_U$. These two scenarios differ in the amount of information used during the training process, and can be applied to different real-world situations. As we emphasized before, in either scenario, during the training process, the information about which line is faulted is not included. We refer to it as working with unlabeled data.

We also explore the following options to potentially improve trajectories reconstruction and fault detection:

1. investigate if different choices of the loss function ℓ increase the accuracy of fault localization;
2. examine feature transformation techniques that could help increase the accuracy;
3. investigate selecting an optimal set of $\mathbf{K}$ (optimal PMU placement).

In the next section, we only describe the methods that are used to obtain the final, tabulated or displayed, results. However, in Section 4.5, we also discuss some other techniques and options that we experimented with.

4.4 Methods

In this section, we describe the proposed methodology. We begin with a brief account in Section 4.4.1 of Graph Neural Networks (GNNs) focusing on aspects relevant to the methods we propose. In Section 4.4.2, we explain the details of the proposed development of GNNs to make them the applicable to PMU data reconstruction with the ultimate objective of fault localization. The remaining subsections discuss the loss functions, selection of an optimal set $\mathbf{K}$, a fault localization scheme used after trajectories in $\mathbf{U}$ have been reconstructed and a benchmark model against which the performance of our method is judged.

The development of the methodology in this section is analogous to the development of traditional statistical methodology. We work with data objects that have the form of structured $K + U \times (2m)$ matrices, and we basically want to solve a missing data problem with a specific objective in mind. For this, we need to propose a model for the data and show how it can be estimated and used for purpose focused data imputation. The model is formulated in terms of the layers of a deep learning network. There are no explicit formulas for estimating the parameters of the layers, they must be estimated through a training process that must be specified. Details of such approaches are presented e.g. in Goodfellow *et al.* (2016) and Zhang *et al.* (2023), where terminology used in the remainder of this paper is also explained

4.4.1 Graph Neural networks

Graph Neural Networks (GNNs) utilize graph data structures as inputs to learn and make predictions. They have seen a rise in popularity due to their ability to generalize the convolution operator to graph structures, see Kipf and Welling (2017). These networks have proven to be effective in many applications, such as drug discovery, recommendation systems, social networks, molecular structures, and electrical grids, see (Li *et al.*, 2018; Yu *et al.*, 2018; Wu *et al.*, 2019), among many others.

One of the key components of GNNs is the use of graph convolutional layers designed to capture the local and global structure of the input graph. The computation is driven by the input

graph topology, which is described by its adjacency matrix. The adjacency matrix $\mathbf{L}_a$ of a graph $\mathbf{G}$ with n nodes is an $n \times n$ matrix such that $\mathbf{L}_a(i,j) = 1$ if there is an edge between nodes i and j, and $\mathbf{L}_a(i,j) = 0$ otherwise. In our setting, nodes are buses and edges the lines connecting them. The first layer of a GNN typically encodes the graph structure and the initial node features, while subsequent layers aggregate information from the node's neighbors and update the node's representation. The final layer of a GNN produces the output, which can be used for tasks such as node classification, link prediction, or recommendation. These layers use the adjacency matrix of the graph and a trainable weight matrix to compute the convolution according to the equation

$$H^{(l+1)} = \sigma(\mathbf{L}_a, H^{(l)}, \Theta^{(l)}),$$

where $H^{(l)}$ and $H^{(l+1)}$ represent the input and output of the convolutional layer, respectively, $\mathbf{L}_a$ is the adjacency matrix, $\Theta^{(l)}$ is the weight matrix for the layer l, and σ is an activation function.

GNNs have been extensively studied in recent years, with significant progress in the development of various architectures and techniques for improving their performance, e.g. (Zhou *et al.*, 2020; Hamilton, 2020). Under some assumptions, a GNN model can be approximate in probability all functions on graphs with any required precision Scarselli *et al.* (2009). A review of the capabilities of GNN is given by Xu *et al.* (2018). Liao *et al.* (2022) discuss their applications in power systems. Recently, models that combine kriging convolution networks and GNNs have been proposed (Hamilton *et al.*, 2017; Wu *et al.*, 2020; Appleby *et al.*, 2020). Our approach is based on networks of this type.

In the original framework, GNNs are inductively trained, adapting their parameters based on a supervised learning environment. Inductive GNNs, in particular, are designed to generalize to new, unseen nodes in a graph. In this paper, we assume that the graph $\mathbf{G} = (\mathbf{A}, \mathbf{L})$ is fixed, which does not fully align with the definition of inductive learning. However, our model needs to generalize to buses that are in $\mathbf{G}$, but their trajectories are unseen. To incorporate different data availability scenarios listed in Section 4.3, we consider two approaches:

(1) the training process of f_θ can use both known and unknown trajectories $\mathbf{X}_K$ and $\mathbf{X}_U$, respectively. In this approach, the whole information about $\mathbf{L}$ is used, and the training set patterns and their targets are used for training. This approach is similar to traditional regression models. We call it *Regression Graph Neural Network* (RGGN);
(2) the training process of f_θ cannot use unknown trajectories $\mathbf{X}_U$ and thus needs to learn from the known trajectories $\mathbf{X}_K$ with a knowledge of fixed edges $\mathbf{L}$. We call this approach *Semi-Inductive Graph Neural Network* (SIGGN).

We need to predict the trajectories at unseen nodes, but the training can use the information about these buses' existence. Such information is crucial for letting the network properly learn the message passing mechanism. The work of Hamilton *et al.* (2017) showed that our considered approaches are a reasonable attempt to tackle the challenges listed in Section 4.3. Although there are more methods to tackle the problem under scenario (1) than under scenario (2), we chose to use the same model concept for both to better understand the challenges that need to be addressed when moving from (1) to (2). Moreover, efficient results under scenario (1) can demonstrate the proof of concept and provide a foundation for addressing the challenges of scenario (2). By using the same model for both scenarios, we can also investigate how the model's performance changes with respect to the availability of data.

4.4.2 GNNs for PMU data reconstruction

In this section, we describe how we propose to use the RGNN and the SIGNN for the purpose of PMU data reconstruction. We use the GNN architecture proposed by Wu *et al.* (2020) as the basis of our work and develop it to fit our needs. The initial architecture uses an M-layer Diffusion Graph Convolution Network (DGCN), Li *et al.* (2018). The DGCN performs the following operations:

$$\mathbf{H}_{l+1} = \sum_{k=0}^{\mathscr{K}} T_k(\overline{\mathbf{L}}_a)\mathbf{H}_l\Theta_l^k,$$

where $\mathbf{H}_l$ is the input feature matrix at layer l and $\mathbf{H}_{l+1}$ is the output feature matrix, $\overline{\mathbf{L}}_a$ is the transition matrix calculated from the adjacency matrix $\mathbf{L}_a$, $\mathscr{K}$ is the maximal order of diffusion

convolution, $T_k(x) = x^k$ (pointwise). We denote by Θ the set of all weight matrizes Θ_l^k. In the following, we use the term *learning* to describe finding the set Θ such that some loss function ℓ is minimized. The dimension of matrices Θ_l^k can be determined by the user by specifying the *width of the network*, z. Specifically,

$$\Theta_1^k \in \mathbb{R}^{2m\times z},\ \ \Theta_l^k \in \mathbb{R}^{z\times z},\ 2 \le l \le M-1,\quad \Theta_M^k \in \mathbb{R}^{z\times 2m}. \tag{4.4.1}$$

Based on many numerical experiments, we determined that, in our context, better results are obtained by using the network

$$\mathbf{H}_l = \sigma\left(\sum_{k=0}^{\mathscr{K}} T_k(\overline{\mathbf{L}}_a)\mathbf{H}_{l-1}\Theta_l^k\right) + \mathbf{H}_{l-1},\quad 1 \le l \le M, \tag{4.4.2}$$

where σ the scaled exponential linear unit (SELU) activation function, Klambauer *et al.* (2017), with standard parameters. The initial input is the matrix $\mathbf{H}_0 \in \mathbb{R}^{n\times 2m}$, where $n = 122$ is the number of buses and $m = 32$ is the count of time points before and after the fault used in our algorithms. We thus use only 64 measurements corresponding to the interval of about 0.53 s centered at the time of the fault. the input $\mathbf{H}_0$ has the unobserved trajectories masked, as described in Algorithms 1 and 2 below. The masked buses only pass 0 to their neighbors in the first layer, while the remaining $M-1$ layers are responsible for more generalized representations. The Mth layer is used to output the reconstructed trajectories for each bus. Thus, $\mathbf{H}_M \in \mathbb{R}^{n\times 2m}$ is the reconstruction results that we use to calculate the loss and update weights Θ. In our experiments, we explored different structures and building pieces (GAT, Chebynet, different activation functions) that would bring the gain in fault localization accuracy without loss in efficiency. Through the series of experiments we found that the proposed structure is the most efficient for the specific problem we aim to solve. In many applications $M = 3$ is sufficient, however, for power grid data, deeper networks may be more effective, as demonstrated in a different context by Ringsquandl *et al.* (2021). The parameters $\mathscr{K}$, the order of convolution, and z, the number of hidden channels of the weight matrix Θ_l^k, may also play an important role. Using small $\mathscr{K}$ values allows us to keep the diffusion process more

localized in a power grid, while the parameter z controls the number of parameters for each filter $T_k(\overline{\mathbf{L}}_a)\mathbf{H}_{l-1}$. It is not practical to find the optimal values of M, $\mathscr{K}$, and z for each model and setup. We performed a limited grid search to find the optimal settings for M, $\mathscr{K}$ and z. We discuss the results in Section 4.5. Other issues that must be addressed are: implementing a validation scheme in order to choose the best model and getting reconstruction results from $\mathbf{H}_M$. We proceed with detailed descriptions of all steps.

Detailed description of the SIGNN implementation. For each training iteration, we randomly divide buses from $\mathbf{K}$ into $\mathbf{K}_0$ (prediction buses) and $\mathbf{K}_1$ (target buses). The size of these sets is K_0 and K_1, respectively. The training process utilizes information from $\mathbf{K}_0$ to learn the trajectories of modulus of voltage in $\mathbf{K}_1$ and records the results of the loss function. The loss function is used to update parameters Θ. We utilize a mask matrix, denoted M_{sample}, to keep buses in $\mathbf{K}_0$ as observed and buses in $\mathbf{K}_1$ as unobserved, see Algorithm 4. The size K_1 is randomly selected as a number between 2 to 7 for each iteration. During this process, the adjacency matrix $\mathbf{L}_a$ remains fixed. The information about the existence of $\mathbf{U}$ is contained only in $\mathbf{L}_a$, no measurements in $\mathbf{U}$ are used. The resampling of $\mathbf{K}_0$ and $\mathbf{K}_1$, in principle, allows the model to learn the relationships between different trajectories, and generalize results over the nodes in $\mathbf{U}$.

To ensure that the validation error is measured uniformly for each epoch, it is necessary to determine sets $\mathbf{K}_0^{\text{Val}}$ and $\mathbf{K}_1^{\text{Val}}$ before the training procedure. To better understand the validation error over nodes that are not seen during training, we randomly select 3 nodes from $\mathbf{K}$ to create $\mathbf{K}_1^{\text{Val}}$. During training, we effectively have $\mathbf{U}$ to include $\mathbf{K}_1^{\text{Val}}$ and subtract it from $\mathbf{K}$ (i.e., $\mathbf{U} := \mathbf{U} \cup \mathbf{K}_1^{\text{Val}}$ and $\mathbf{K} := \mathbf{K} \setminus \mathbf{K}_1^{\text{Val}}$). As a result, the model cannot access trajectories at $\mathbf{K}_1^{\text{Val}}$ during training, but uses them to compute the validation error. This helps us to understand the generalization power of the model, and choose settings that give optimal results for the test error.

The learning procedure is summarized in Algorithm 4. Recall that we denote by $\mathbf{X}$ the $n \times (2m)$ matrix obtained from a single fault simulation. In Algorithm 4 we denote by $\mathbf{X}^{\text{Tr}}$ the set of simulations used in training (we use 10,000 of them) and by $\mathbf{X}^{\text{Val}}$ the simulations used for validation (we use 2,000). We emphasize that Step 8 in Algorithm 4 guarantees that we preserve

fixed graph structure $\mathbf{G}$, but ensure that the training process has no access to trajectories of buses without PMU devices. Due to the structure of the model, it is suggested to set high number of iterations.

Algorithm 4 Random mask generation and model learning for SIGNN

Input: Training data $\mathbf{X}^{\text{Tr}}$, validation data $\mathbf{X}^{\text{Val}}$, graph structure $\mathbf{G}$, length of each simulation $2m$, observed buses $\mathbf{K}$ and unobserved buses $\mathbf{U}$; Parameters: size of each batch S, maximum iteration I_{max}, maximum epoch E_{max}, loss function ℓ, fixed $\mathbf{K}_1^{\text{Val}}$ set for validation.

for epoch = $1 : E_{max}$ **do**
  for iteration=$1 : I_{max}$ **do**
    Generate random integer $k_1 \in \{2, \ldots, 7\}$.
    Randomly sample k_1 buses from $\mathbf{K} \setminus \mathbf{K}_1^{\text{Val}}$ to form $\mathbf{K}_1$.
    Set $\mathbf{K}_0 = \mathbf{K} \setminus (\mathbf{K}_1 \cup \mathbf{K}_1^{\text{Val}})$.
    for sample = $1 : S$ **do**
      Randomly choose a simulation $\mathbf{X}_{\text{sample}}$ from $\mathbf{X}^{\text{Tr}}$.
      Set rows of $\mathbf{X}_{\text{sample}}$ corresponding to buses in $\mathbf{U}$ and $\mathbf{K}_1^{\text{Val}}$ to 0.
    end for
    Generate a mask matrix M_{sample} of the same size as $\mathbf{X}_{\text{sample}}$,

$$M_{sample}[i,:] = \begin{cases} 1 & \text{if } i \in \mathbf{K}_0 \\ 0 & \text{otherwise} \end{cases}$$

    Use sets $\{\mathbf{X}_{1:S}\} \otimes M_{1:S}$ and ℓ to train GNNs (update weights matrix Θ)
  end for
  Calculate validation error Err_{epoch} using ℓ, $\mathbf{X}^{\text{Val}}$ with a predictor/response split $\mathbf{K}_0^{\text{Val}}$ and $\mathbf{K}_1^{\text{Val}}$, and obtained weight matrix Θ. Corresponding buses in $\mathbf{U}$ and $\mathbf{K}_1^{\text{Val}}$ of $\mathbf{X}^{\text{Val}}$ are equal 0.
end for
Final learned model is GNN with weight matrix $\Theta = \Theta_{\arg\min_{\text{epoch} \in \{1,\ldots,E_{max}\}} Err_{\text{epoch}}}$

Detailed description of the RGNN implementation We simplify Algorithm 4 to employ the data in $\mathbf{X}_U$. The procedure is summarized in Algorithm 5.

Reconstruction We explain the process of reconstructing the trajectories in the unseen nodes $\mathbf{U}$ for each simulation (sample) using both SIGNN and RGNN. Assume we have trained the model and obtained the weight set Θ. First, we form masked signals $\mathbf{X}_{sample}^M = [\mathbf{X}_K, \mathbf{X}_U]$ with $\mathbf{X}_U = 0$ and set $\mathbf{H}_0 := \mathbf{X}_{sample}^M$. We input obtained Θ and $\mathbf{H}_0$ to the GNN described architecture and obtain $\mathbf{H}_M$. Set $\widehat{\mathbf{X}}_{sample}^M := \mathbf{H}_M$. The $\widehat{\mathbf{X}}_{sample}^M$ can be further split into $\widehat{\mathbf{X}}_{sample}^M = [\widehat{\mathbf{X}}_K, \widehat{\mathbf{X}}_U]$. The

Algorithm 5 Random mask generation and model learning for RGNN

Input: Training data $\mathbf{X}^{\text{Tr}}$, validation data $\mathbf{X}^{\text{Val}}$, graph structure $\mathbf{G}$, length of each simulation $2m$, observed buses $\mathbf{K}$ and unobserved buses $\mathbf{U}$; Parameters: size of each iteration S, maximum iteration I_{max}, maximum epoch E_{max}, loss function ℓ.

for epoch = $1 : E_{max}$ **do**
  for iteration=$1 : I_{max}$ **do**
    for sample = $1 : S$ **do**
      Randomly choose a simulation $\mathbf{X}_{\text{sample}}$ from $\mathbf{X}^{\text{Tr}}$.
      Generate a mask matrix M_{sample} of the same size as $\mathbf{X}_{\text{sample}}$,

$$M_{sample}[i,:] = \begin{cases} 1 & \text{if } i \in \mathbf{U} \\ 0 & \text{otherwise} \end{cases}$$

    end for
    Use sets $\{\mathbf{X}_{1:S}\} \otimes M_{1:S}$ and ℓ to train GNNs (update weights matrix Θ)
  end for
  Calculate validation error Err_{epoch} using ℓ and $\mathbf{X}^{\text{Val}}$ with a predictor/response split $\mathbf{K}$ and $\mathbf{U}$.
end for
Final learned model is GNN with weight matrix $\Theta = \Theta_{\arg\min_{\text{epoch}\in\{1,\ldots,Emax\}} Err_{\text{epoch}}}$

GNN architecture outputs the fit of the trajectories at all buses, but the final product of reconstruction is only $\widehat{\mathbf{X}}_U$. During the training process of RGNN, the intermediate results have the same structure as the final reconstruction, but that is not the case for SIGNN. In SIGNN, a similar structure is used to obtain reconstruction results within each iteration, but $\mathbf{X}^M_{sample}$ is replaced with $\mathbf{X}^M_{sample,K_0,K_1} = [\mathbf{X}_{K_0}, \mathbf{X}_{K_1}, \mathbf{X}_U]$, where $\mathbf{X}_{K_1} = 0$ and $\mathbf{X}_U = 0$.

4.4.3 Loss functions

In addressing the problem formulated in Section 4.3, a typical approach would be to use the mean squared error (MSE) loss function. However, this may not always be the best approach. Most deep learning methods rely on the MSE or its variants (such as MAE and quantile loss). In our case, the primary goal is to accurately predict the fault location, which may not be best achieved through the MSE alone. To address this, we have explored alternative loss functions, including bias-weighted MSE, normalized soft time warping, and others (Cuturi and Blondel, 2017; Guen

and Thome, 2022). We only discuss the MSE and the Bias-weighted MSE, as they are used to report results. The methods are additionally evaluated by their ability to minimize fault detection failure rate, see Section 4.4.5, which can also be seen as a loss function, even though it is not used in training, but is used to arrive at final recommendations.

To simplify the explanation, let us consider the case of evaluating the loss function for a single simulation. This can be easily extended to multiple simulations by taking the average. In this section, we define the matrix of reconstructed trajectories as $\widehat{\mathbf{Y}} = (\hat{\mathbf{y}}_1, \ldots, \hat{\mathbf{y}}_r)$, and the true matrix is defined as $\mathbf{Y} = (\mathbf{y}_1, \ldots, \mathbf{y}_r)$, where r is the number of reconstructed trajectories. Further, define each reconstructed trajectory $\hat{\mathbf{y}}_i = (\hat{y}_{i,1}, \ldots, \hat{y}_{i,2m})$ in terms of $2m$ data points. Similarly, set $\mathbf{y}_i = (y_{i,1}, \ldots, y_{i,2m})$.

The MSE loss function is defined as:

$$\ell_{\text{MSE}}(\hat{\mathbf{Y}}, \mathbf{Y}) = \frac{1}{r} \sum_{i=1}^{r} ||\hat{\mathbf{y}}_i - \mathbf{y}_i||^2 = \frac{1}{r} \sum_{i=1}^{r} \sum_{j=1}^{2m} (\hat{y}_{i,j} - y_{i,j})^2.$$

For the bias-weighted MSE, we propose to decompose the MSE into two components: the difference in means (bias) and the difference in variance. This is motivated by the statistical fault localization technique employed in this paper, where a significant bias in the reconstruction has a lesser impact on the fault localization accuracy compared to a large difference in shape, approximated by the variance. The bias of prediction for each $i \in \{1, \ldots, r\}$ is defined as $b_i = \bar{y}_i - \bar{\hat{y}}_i$, where the averages are taken of the $2m$ data points. We then define the bias adjusted reconstruction matrix as $\widehat{\mathbf{Y}}_{ba} = (\hat{y}_1 - b_1, \ldots, \hat{y}_r - b_r)$. With a few simple steps, it can be shown that

$$\ell_{\text{MSE}}(\widehat{\mathbf{Y}}, \mathbf{Y}) = \ell_{\text{MSE}}(\hat{\mathbf{Y}}_{ba}, \mathbf{Y}) + \frac{2m}{r} \sum_{i=1}^{r} b_i^2.$$

We propose weighting these two parts of $\ell_{\text{MSE}}(\widehat{\mathbf{Y}}, \mathbf{Y})$, putting more emphasis on the first part. Using this notation and including a tuning parameter $\alpha \in (0, 1)$, we define the bias-weighted MSE

as

$$\ell_{\text{BMSE}}(\hat{\mathbf{Y}}, \mathbf{Y}) = \alpha \ell_{\text{MSE}}(\hat{\mathbf{Y}}_{ba}, \mathbf{Y}) + (1-\alpha)\frac{2m}{r}\sum_{i=1}^{r} b_i^2.$$

The hyperparameter α is determined during the training process. Notice that if $\alpha = 0.5$, $\ell_{\text{BMSE}} = \frac{1}{2}\ell_{\text{MSE}}$.

4.4.4 Optimal PMU placement

Our framework assumes that PMUs are available only at a subset of buses $\mathbf{K} \subset \mathbf{A}$. A question of interest is what the locations $\mathbf{K}$ should be to minimize aggregated loss functions and the fault localization failure rate, given that $K = |\mathbf{K}|$ is fixed. A brute-force search for the optimal $\mathbf{K}$ by testing every possible configuration given K is infeasible due to the astronomical number of possibilities. With $K = 70$ and 122 possible locations for PMUs, there are approximately 1.02E+35 possible combinations for different placement of PMU's.

PMU placement algorithms discussed in the literature, e.g. (Aminifar *et al.*, 2010; Enshaee *et al.*, 2012; Abiri *et al.*, 2014; Abiri *et al.*, 2015; Shafiullah *et al.*, 2022) do not necessarily ensure optimal results for reconstruction and fault localization accuracy, which are the criteria we use in our work. To address this issue, we propose optimizing the PMU placement set $\mathbf{K}$ using one of the proposed loss functions ℓ. We propose a version a regression technique called the Backward Stepwise procedure, specified in Algorithm 6. Starting with $\mathbf{K}$ equal to the set of all buses ($K = 122$, the algorithm progressively removes buses from the set $\mathbf{K}_{opp}$ of optional buses and adds them to the set $\mathbf{U}_{opp}$ of used buses until $|\mathbf{K}_{opp}| = K$. The decision of which bus to move is based on the loss function ℓ. We consider moving each possible bus from the set $\mathbf{K}_{opp}$ and evaluate the cost of the decision using ℓ. For simplifications, we employed this algorithm with simpler model (RGNN) using the MSE loss function and without features transformation (Algorithm 5). As Algorithm 6 requires running Algorithm 5 many times, we reduced computational burden by lowering settings (reducing dimension for graph convolution, using fewer iterations, and fewer epochs).

Algorithm 6 Optimal PMU placement strategy

Input: Training data $\mathbf{X}^{\text{Tr}}$, validation data $\mathbf{X}^{\text{Val}}$, graph structure $\mathbf{G}$, length of each simulation $2m$; Parameters: sample size each iteration S, maximum iteration I_{max}, maximum epoch E_{max}, desired $\mathbf{K}$ size K.

Output: Optimal set $\mathbf{K}_{opp}$ of size K

Set $\mathbf{K}_{opp} = \{1, \ldots, 122\}$ and $\mathbf{U}_{opp} = \{\}$
while $|\mathbf{K}_{opp}| > K$ **do**
    for b in $\mathbf{K}_{opp}$ **do**
        Run Algorithm 5 with $\mathbf{U} := \{\mathbf{U}_{opp}, b\}$ and $\mathbf{K} := \mathbf{K}_{opp} \setminus \{b\}$
        Record validation error Err_b
    end for
    Set $\hat{b} = \arg\min_{b \in \mathbf{K}_{opp}} Err_b$
    Set $\mathbf{K}_{opp} := \mathbf{K}_{opp} \setminus \{\hat{b}\}$ and $\mathbf{U}_{opp} := \{\mathbf{U}_{opp}, \hat{b}\}$
end while

4.4.5 Fault Localization Scheme

The overall structure of the proposed fault localization method under partial observability involves two modules: 1) reconstructing the modulus of voltages at unobserved buses, and 2) using existing methods, which require full grid observability, to predict the buses (i_0, j_0) connected by the faulted line l_0. Module 1) has already been described. In this subsection, we focus on module 2), assuming possession of the full data set $\mathbf{X}$. In practice, this requires replacing $\mathbf{X}$ by $\widehat{\mathbf{X}}$.

In principle, any fault localization method can be used, but we use a slightly revised method of Kokoszka *et al.* (2023) because we have code for it that works in the minniWECC setting, and we have been unable to gain access to code for other methods. We assume that the time of the fault, t_f, is estimated using one of the available methods, e.g. (Shafiullah and Abido, 2018; Li and Deka, 2021; Kokoszka *et al.*, 2023; Rimkus *et al.*, 2023). In the following, t_f is treated as known.

We have $\mathbf{X} = \mathbf{x}_1, \ldots, \mathbf{x}_{122}$, where $\mathbf{x}_k = x_k(0), x_k(\delta), \ldots, x_k(2m \cdot \delta)$ is a trajectory of the modulus of voltage at bus k, and δ is the time separation of the consecutive data points ($\delta = \frac{1}{120}$ in this paper). We denote by S_0 and S_1 the time windows (in seconds) used to extract information from trajectories around fault time t_f. For each bus k, we set

$$\bar{x}_k(t, S_0) = \frac{1}{S_0/\delta} \sum_{l=1}^{S_0/\delta} x_k(t - l\delta), \tag{4.4.3}$$

$$m_k(t, S_1) = \frac{1}{1 + S_1/\delta} \sum_{l=0}^{S_1/\delta} x_k(t + l\delta). \tag{4.4.4}$$

We obtain the prediction of the near end bus by setting:

$$\hat{i}_0 = \underset{k \in \{1,\ldots,122\}}{\arg\max} |m_k(t_f, S_1) - \bar{x}_k(t_f, S_0)|. \tag{4.4.5}$$

Once we have predicted the near end bus $\hat{i}_0$ of the faulted line l_0, we predict the far end bus by employing the properties of fault clearance. Notice that the prediction $\hat{j}_0$ must be one of the buses to which $\hat{i}_0$ is connected by a direct line, so there are only a few (or no) choices. For each bus k, we set

$$D_i^*(t, t_f, S_0) = m_k(t, 0) - \bar{x}_k(t_f, S_0), \quad t > t_f. \tag{4.4.6}$$

The time of the recovery, t_R, is defined as the smallest $t > t_f$ such that

$$\left| \frac{D_{\hat{i}_0}^*(t, t_f, S_0) - D_{\hat{i}_0}^*(t - \delta, t_f, S_0)}{D_{\hat{i}_0}^*(t - \delta, t_f, S_0)} \right| > \tau_1. \tag{4.4.7}$$

The parameter τ_1 was determined Kokoszka *et al.* (2023) who also explained that its choice is not critical, as long as it remains in a reasonable range. We obtain the prediction of the remote end bus by setting

$$\hat{j}_0 = \arg\max_k (|D_k^*(t, t_f, S_0)| - |D_k^*(t - \delta, t_f, S_0)|), \tag{4.4.8}$$

where the maximum is taken over busses k that connect directly to $\hat{i}_0$.

The predicted faulted line $\hat{l}_0$ is the line between buses $\hat{i}_0$ and $\hat{j}_0$. Under the assumption of PMUs at every possible bus, this methodology produces results with around 4% failure rate. To report fault localization results under partial observability, we also use the failure rate - the percentage of simulations where the faulted line/near-end bus was predicted incorrectly.

4.4.6 Benchmark method

To assess the results of the proposed deep learning methodology, we use a relatively simple and intuitive benchmark method (BM). To the best of our knowledge, there is no other work that focuses on fault localization in a regional power grid without using labeled data, so we cannot use an existing benchmark. Recall that $\mathbf{U}$ is the set of buses where trajectories need to be predicted and $\mathbf{K}$ is the set of the available buses. The BM is implemented for each trajectory $\mathbf{y}_k$ with $k \in \mathbf{U}$. It takes the trajectories of directly connected buses and averages them to estimate the missing trajectory. If some of the buses connecting directly to bus $k \in \mathbf{U}$ are themselves in $\mathbf{U}$, their reconstructed trajectories are used, which are obtained through an iterative process. The iterations continue until each bus in $\mathbf{U}$ has all neighbors with available trajectories The process can be formalized by setting

$$\hat{\mathbf{y}}_k = \frac{1}{|\mathbf{B}_k|} \sum_{j \in \mathbf{B}_k} \mathbf{y}_j^*,$$

where $\mathbf{B}_k$ is set of buses connecting directly to bus k. Some of the $\mathbf{y}_j^*$ can be real trajectories, if $j \in \mathbf{K}$, some can be reconstructed trajectories, if $j \in \mathbf{U}$.

4.5 Application to the Western Interconnection

In this section, we apply the methods of Section 4.4 to the data described in Section 4.2. We implement them with 10,000 training simulations, 2,500 validation simulations, and 9,068 testing simulations. As in all deep learning approaches, the performance of the methods is evaluated on the testing simulations that were not used during the training and validation process. The selection of the simulations to the three groups was random. The models were implemented using the

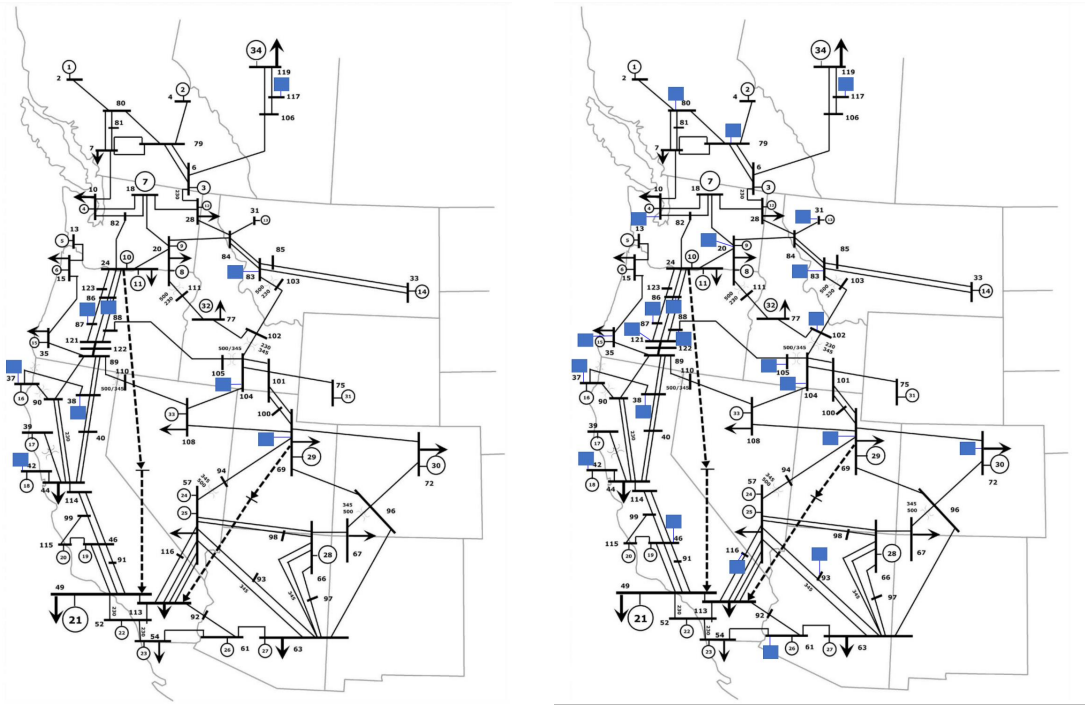

Figure 4.4: The squares indicate recommended locations, according to Algorithm 6, of monitoring devices. Only buses connecting high voltage lines are shown. Left: results for $K = 27$, Right: results for $K = 57$.

Torch framework Collobert *et al.* (2011). We utilized the Adam optimization algorithm Kingma and Ba (2014), which includes the learning rate parameter determining the size of the step for updating the GNN parameters. We conducted experiments with various learning rates (ranging from 0.001 to 0.00005) to update the weight matrix represented as Θ for different-sized models with various hyperparameters M, z, and $\mathscr{K}$ described in Section 4.4.2, cf. (4.4.1) and (4.4.2). We implemented our methods on the CUDA platform with an NVIDIA GeForce GTX 1080 Ti with 11 GB GDDR5X.

We first implemented the optimal PMU placement algorithm, Algorithm 6, using a small RGNN ($M = 3$, $z = 150$, $\mathscr{K} = 2$), and obtained a sequence of the $\mathbf{U}$ sets (the buses without data) used to present the results. Starting with $U = 12$ buses in $\mathbf{U}$, we obtained the sets $\mathbf{U}$ up to $U = 77$. Recall that the total number of buses is 122. By construction, a set $\mathbf{U}$ with a smaller cardinality is always a subset of a set $\mathbf{U}$ with a larger cardinality. Algorithm 6 is a useful tool in its own right. It might allow grid operators to determine where monitoring devices should be placed if only a limited number of them can be monitored. This is illustrated in Figure 4.4.

Second, we implemented a grid search to find optimal hyperparameters. Choosing the appropriate hyperparameters for neural networks is crucial, as poorly chosen ones can result in suboptimal performance such as slow convergence, overfitting, or underfitting Bengio *et al.* (2013). There are several hyperparameters that can affect the results, including batch size, learning rate, network depth, the size of inner layers, diffusion localization level, and number of epochs. In our grid search, we focused on network depth M, the size of inner layers z, and diffusion localization level $\mathscr{K}$. Due to the resources required to train each model, it is not feasible to conduct such a search for every $\mathbf{U}$, so we only performed the grid search for RGNN and SIGNN with $U = 17$. The grid search was implemented solely using validation results, leaving the test data untouched to avoid data leakage. We recommend a similar search to determine the settings that works best for any specific application. This issue is elaborated on in Section 4.5.3.

We evaluated the accuracy of trajectory reconstruction using the loss functions and localization failure rate explained in Section 4.4.3. We compared these results to the Benchmark method of Section 4.4.6. Our results are reported for two testing data splits:

(1) all testing simulations;

(2) testing simulations where the near-end bus of a faulted line is in the set $\mathbf{U}$.

Finding the bus closest to the fault, the near-end bus, is the most important aspect of the localization algorithm.

Finally, to better understand the mechanics of the models, we used two sets of data: the unmodified data and the rescaled data. The rescaled data was generated by dividing each bus's trajectory by its average value prior to the fault, which was determined through the system parameters given in Trudnowski *et al.* (2013). Such a normalization is feasible in a real system because the average voltages of the lines under normal operations are either known or can be easily measured. The assessment of the models on both the unmodified data and the rescaled data facilitates the identification of their strengths and weaknesses and reveals important distinctions between them. Using the unmodified data, the BM and the SIGNN give results influenced by the differences between the trajectories at different buses before the fault happens, the differences in average voltages of

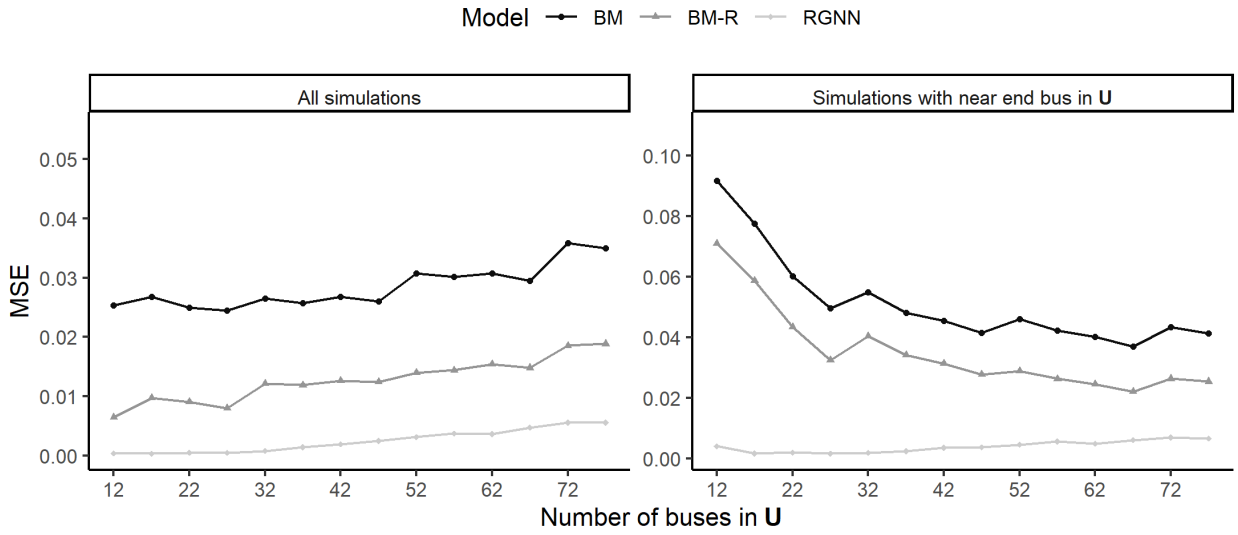

Figure 4.5: Comparison of reconstruction results in terms of the MSE for the RGNN and the benchmark models BM and BM-R, with varying sizes of **U**. The expected trend of decreasing MSE as U increases is observed in the second panel due to more trajectories being included in **U**.

the lines. As the focus of the paper is the reconstruction of the trajectories during a fault, using rescaled data that eliminate this bias is justified. In the following, we use the abbreviations BM or SIGNN to refer to these models trained on the unmodified data, by BM-R and SIGNN-R to these models trained on the rescaled data. More precisely, the application of the BM-R and the SIGNN-R involves the following steps: 1. use rescaled data to train the SIGNN (BM does not need training), 2. use rescaled data to reconstruct missing trajectories, 3. rescale reconstructed trajectories back to the unmodified data space, 4. calculate the MSE and fault localization results as for other methods.

4.5.1 RGNN results

During a grid search for optimal hyperparameters, we observed that for RGNN we do not need to use large z, M, and $\mathscr{K}$ values. This suggests, that the learning is more local and the trajectories' reconstruction is performed mostly using trajectories that are close in the grid. The results below are presented using the model with $z = 200$, $M = 4$, and $\mathscr{K} = 2$ for $U \leq 42$. For $U > 42$, we used $M = 6$.

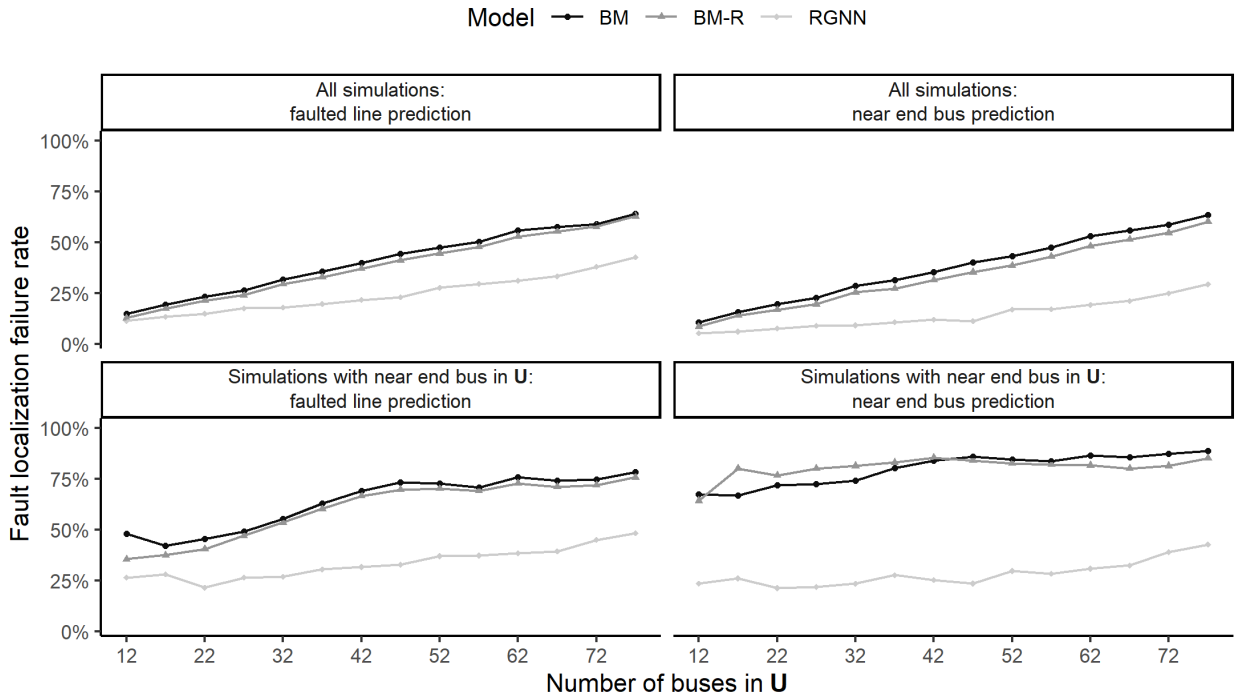

Figure 4.6: Comparison of fault localization results in terms of failure rate for RGNN and the benchmark model (BM and BM-R), with varying sizes of **U**.

Figures 4.5 and 4.6 demonstrate that the RGNN exhibits strong performance in terms of trajectory reconstruction and fault localization. We compared the performance of the RGNN to that of BM and BM-R, it outperforms both of them. With only 36% of the power grid observed ($U = 77$), the RGNN correctly detects the faulted line 3 times out of 4. Looking at the bottom two panels of Figure 4.6, one can see that the RGNN localizes a fault with 60% success rate, even if the fault occurs at a bus not measured by a PMU. That suggests that it can successfully infer where the fault is and reconstruct the trajectories with high success. In Figure 4.7 we illustrate the accuracy of the fit. We present results for $U = 17$ with a testing simulation in which the line between buses 30 and 31 is faulted, with bus 30 being the near-end bus of the faulted line. Bus 30 is included in **U** with $U = 17$. The relative performance of the RGNN remains better than that of BM and BM-R if a different loss functions are used, and there is not much change in the localization success rates.

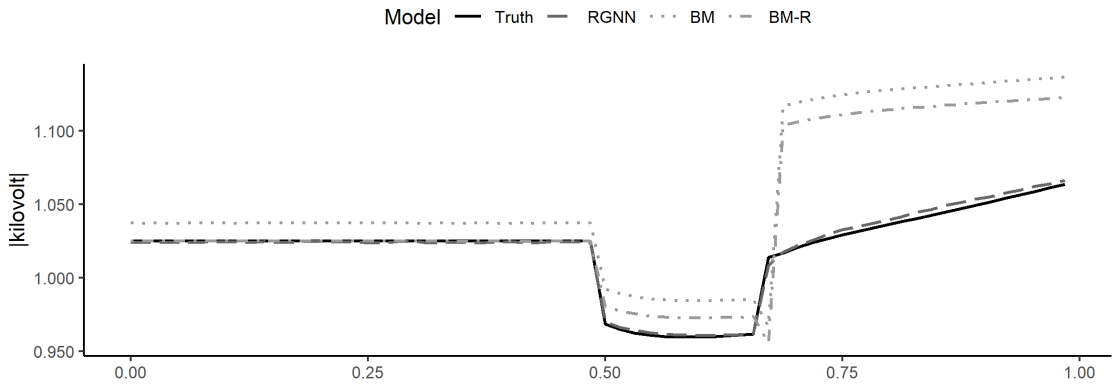

Figure 4.7: Trajectory reconstruction at bus 30 in **U** for $U = 17$. Bus 30 is the near end bus of the fault. The RGGN reconstruction visually almost coincides with the true.

4.5.2 SIGNN results

We now present the results for the second scenario described in Section 4.3 (the SIGNN), in which the training process cannot use trajectories in **U**, and thus needs to learn only from the trajectories in **K**, with the knowledge of all edges **L**. The RGNN can use the trajectories in **U** during training. The SIGNN must thus solve a much more challenging task. This corresponds to a situation where no PMUs have ever been placed in **U**. The RGNN is trained on the actual data in **U** that may become unavailable due to, say, a natural disaster or a cyberattack.

We first discuss the performance of the SIGNN using ℓ_{MSE} and highlight the need for improvements. We later show the results for SIGNN with ℓ_{BMSE} and SIGNN-R. During a grid search for the optimal hyperparameters for SIGNN, we found that the hyperparameters did not significantly change the validation error. However, we found that the best validation MSE results were achieved using $M = 4$, $\mathscr{K} = 2$, and $z = 200$, which we used to present the results below. The values of M and $\mathscr{K}$ are the same as for the RGNN, the network with z is smaller. We observed that the default version of SIGNN failed to generalize over the original data, as shown in Figures 4.8 and 4.9, and did not outperform the benchmark models.

We observed a bias in the SIGNN predictions for all trajectories, indicating that it failed to generalize the overall levels of the unknown trajectories, resulting in a shift in the predicted values that impeded effective learning. This explains the large MSE and reasonable fault localization results

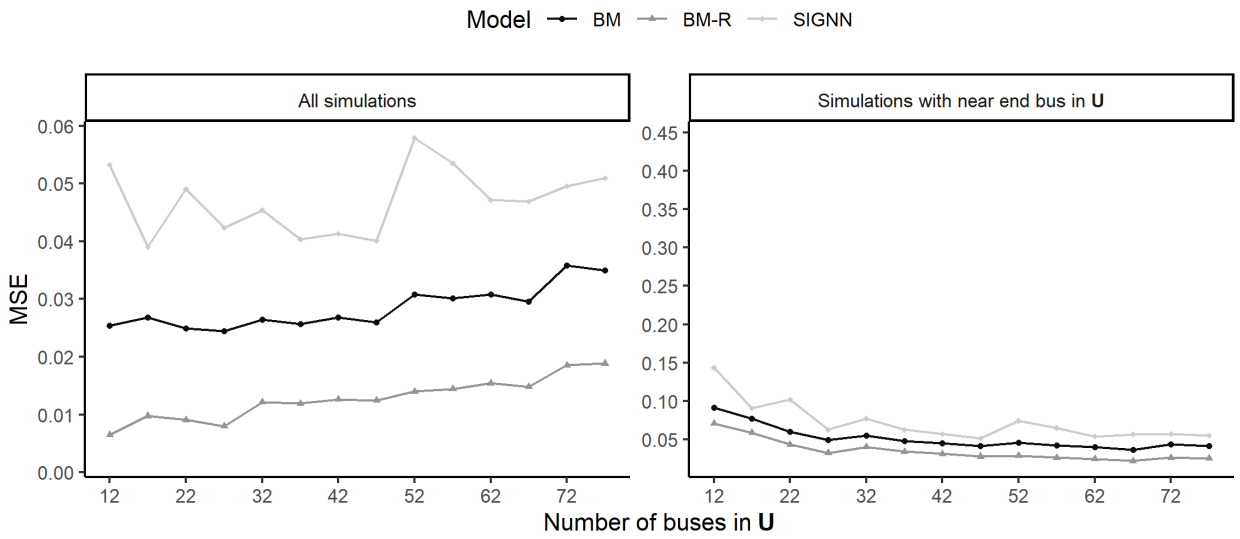

Figure 4.8: Comparison of reconstruction results for SIGNN and the benchmark models BM and BM-R, with varying sizes of **U**.

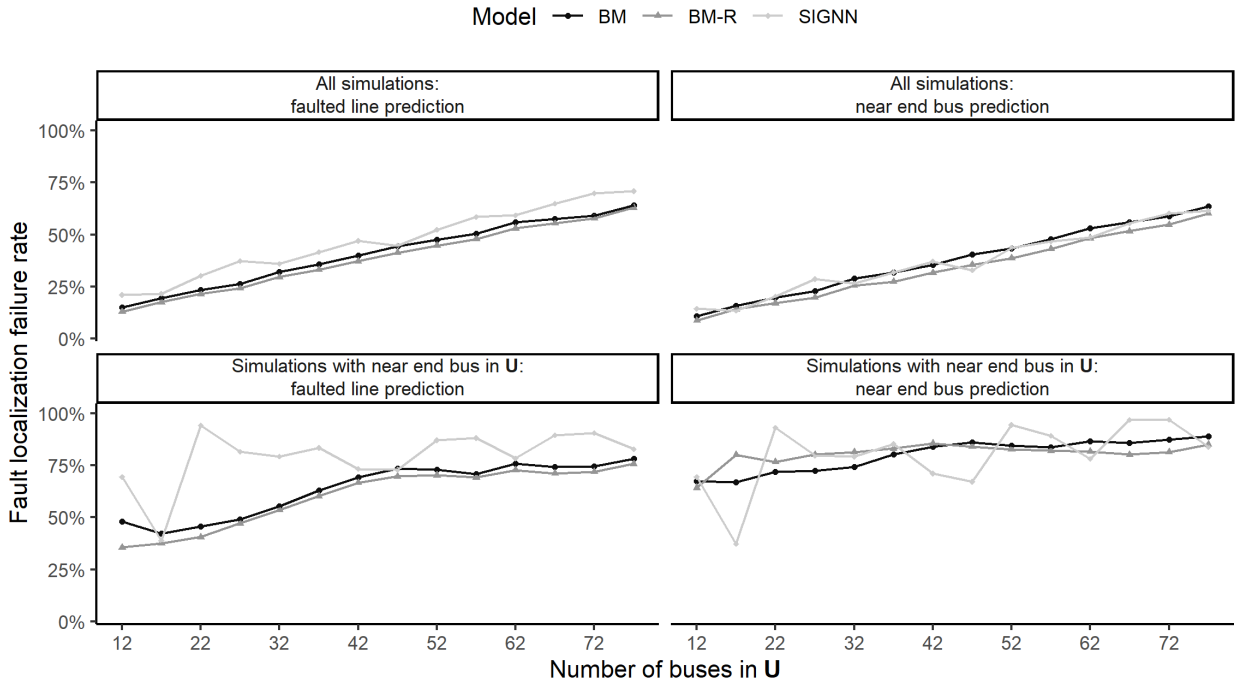

Figure 4.9: Comparison of fault localization results in terms of failure rate for SIGNN and the benchmark models BM and BM-R.

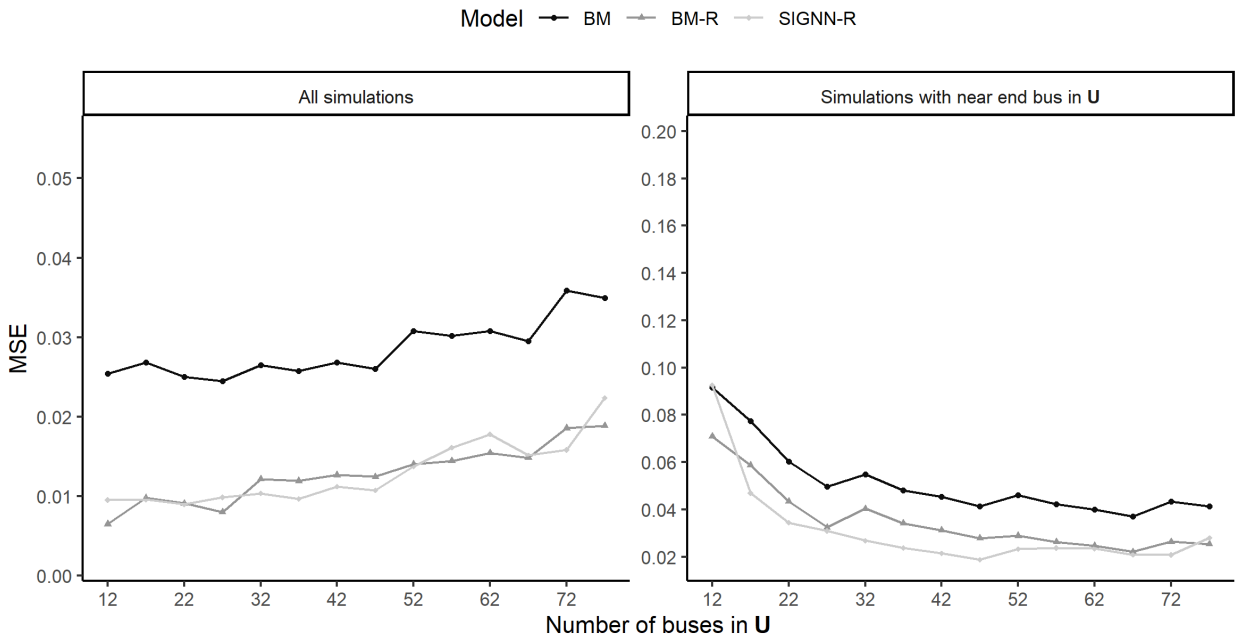

Figure 4.10: Comparison of reconstruction results for SIGNN-R with varying sizes of **U**.

that use mostly the shape of the trajectories during a fault. Because of this bias, the SIGNN trained using the ℓ_{MSE} does not effectively learn the level of trajectories. To gain a better understanding of the capabilities of the SIGNN, we have experimented with various approaches. We first examined the SIGNN-R. We repeated the grid search for the SIGNN-R, and obtained very different behavior from the SIGNN. We discovered that larger depth of SIGNN-R lowers the validation error. We found that the best validation MSE results were achieved using the model with $z = 200$, $M = 6$, and $\mathscr{K} = 2$ for $U \leq 42$. For $U > 42$, we used $M = 8$ and $\mathscr{K} = 1$.

The reconstruction and fault localization results are shown in Figures 4.10 and 4.11, respectively. The SIGNN-R does not bring much improvement when all simulations are used for evaluation. However, it does perform better than the benchmark methods if the near end bus is in **U**. This means that the SIGNN-R can locate a bus near a fault better than the benchmarks without ever seeing a fault at that bus. Using ℓ_{BMSE} with $\alpha = 0.99$ on the unmodified data produces similar, only slightly worse, results than SIGNN-R. Recall that ℓ_{BMSE} was designed to reduce the impact of the level of the trajectories. A broad conclusion from these experiments is that the typical voltage, in the absence of faults, at buses in **U** must either be made available to a GNN during a learning

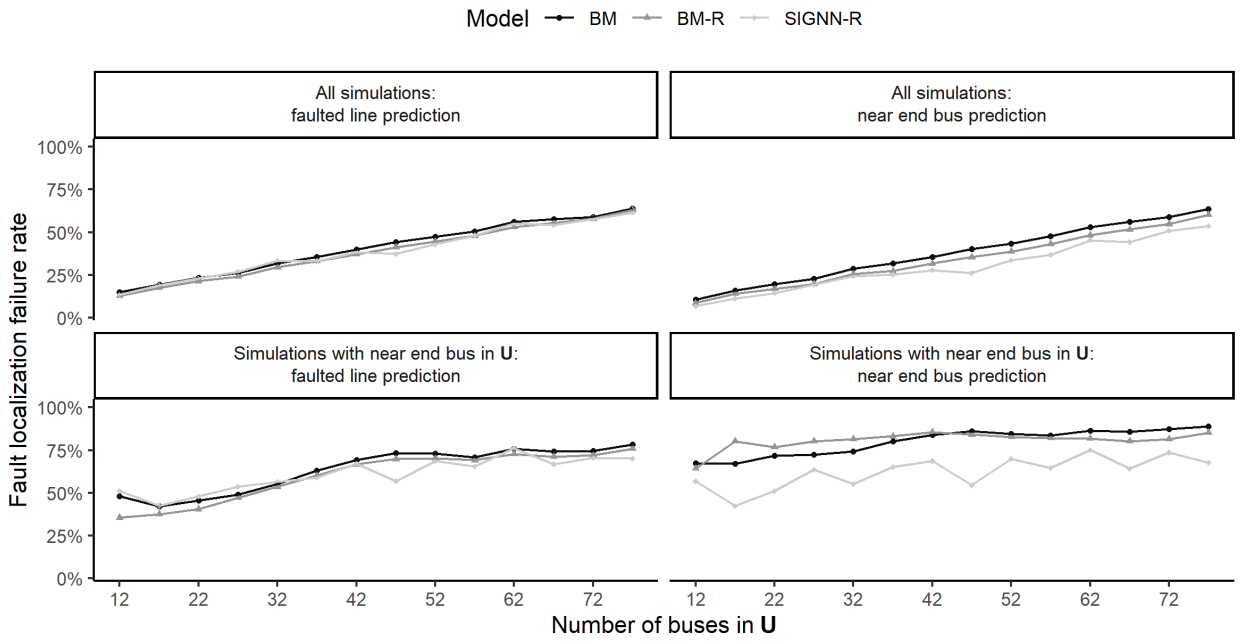

Figure 4.11: Comparison of fault localization results in terms of localization failure rate for SIGNN-R with different sizes of **U**.

process, or all the features must be adjusted for these levels (rescaling), or the loss function must be made basically independent of them.

4.5.3 Additional information on hyperparameters and feature selection

We conducted a grid search for SIGNN, SIGNN-R and RGNN with $U = 17$ and $U = 52$ using the validation error. We observed that the parameter $\mathscr{K}$ had minimal impact on the results, and we found that setting $\mathscr{K} = 1;2$ produced reasonable results. We reported all results with $\mathscr{K} = 2$, except for models trained with $M = 8$. In these cases, due to GPU limitations, we used $\mathscr{K} = 1$ instead. Our findings also showed that there was no significant improvement in increasing z beyond 200. Therefore, we reported all results using $z = 200$. Additionally, we observed that the parameter M (the depth of the network) had an impact on the results, with larger M working better for more complex problems that the network needed to solve. For smaller values of U (e.g.,$U = 17$), the SIGNN-R performed best with $M = 6$, while the RGNN with $M = 4$. However, for higher values of U, the SIGNN-R performed the best with $M = 8$, while the RGNN with $M = 6$. This difference

can be attributed to the amount of time required for the network to distill the masked trajectories in H_0.

Examination of Figures 4.2 and 4.3 suggests that using the Haar (rectangular) wavelet transform might lead to better reconstruction results, as this transform is effective in decomposing signals with “rectangular blocks" into sparse representations. Interested readers are referred to Percival and Walden (2000). We conducted experiments involving the application of the Haar wavelet transform. The inputs and outputs of the GNNs are represented as the wavelet coefficients. From the outputs the trajectories in time domain can be reconstructed exactly and diagnostics can be applied to the reconstructed trajectories. Although the final results were comparable to those for the raw trajectories, we observed that the application of the Haar wavelet transform facilitated convergence to the solution for RGNN and SIGNN-R, necessitating fewer iterations for model training. In many cases, the time needed to properly train these models decreases by half when Haar wavelets were used.

4.6 Summary and discussion

We proposed Graph Neural Networks that can be effectively applied to reconstruct PMU trajectories with the aim of power grid fault localization. Compare to previous methods employed for fault localization, our learning networks do not use labeled data, they do not be given the exact fault location to learn to find it in the future. We studied two scenarios for data availability and implemented the RGNN and SIGNN methods. Firstly, we showed that assuming full data availability during the learning process (RGNN) results in strong trajectory reconstruction and fault localization. With this in mind, we relaxed this assumption and showed that the SIGNN fails to generalize to unobserved trajectories, primarily due to the difference between the levels of the trajectories before the fault. We then tested the SIGNN with a different loss function and with rescaled data. Both approaches showed similar improvements in the fault localization results, mainly the localization of the near-end bus of the faulted line. As noted abobe, to the best of our knowledge, other deep learning methods proposed for fault localization rely on labeled data - information on where the

fault has occurred - during the training process. However, in our SIGNN implementation, we did not even use data from the unobserved part of the power grid. We consider this a significant step toward fault localization when labeled data is not possible or practical to obtain. We showed the potential that one could use the partially observed network and map it a fully observed network, and then use fault localization methods that are proven to work if measurements over the whole network are available. We also note that our benchmark methods, BM and BM-R, originally devised as simple standards, are actually quite effective in the second scenario when faults in a large part of the grid have never been observed.

For future research, it could be beneficial to explore modifying the neural network architecture to more accurately reflect the underlying physical laws governing the system. As observed from the reconstructed trajectories of the RGNN, there is a higher variability of the predicted trajectories than what actually exists in the power system. A possible solution to improve results could be to incorporate a better understanding of the system's trajectories or introducing additional penalization for any violations of the physical rules. It could be useful to investigate the use of physics-informed neural networks to address these issues. It is important to note that our study only utilized purely data-based approaches. Another potential direction for future research is to explore the use of additional variables from the PMU data. In this study, we focused solely on the modulus of voltage, but including other types of data from the power grid could potentially improve the performance of the SIGNN model. For example, incorporating data on phase angles, line currents, or other relevant variables could provide additional insights and help us to train more accurate models.

Chapter 5

Summary and Future Research

Statistical fault diagnosis provides methods to detect, classify, and localize faults in electrical power grids. It is effective and simple enough to be implemented in real systems.

In Chapter 2, we proposed a data-driven, two-stage procedure for fault monitoring in regional power grids. In the first stage, a fault is detected, and in the second stage, the faulted line is identified. Our method only requires knowledge of the start and end buses of each line. It assumes that faults occur on lines, which is a typical situation. If a fault occurs on a bus, the faulted bus can be identified using an algorithm for finding the start bus of the faulted line. Our approach is not dependent on prior knowledge of the fault type and utilizes voltage modulus measurements at buses as input. It offers a general statistical approach for anomaly detection and localization in regional power grids, which have unique characteristics that require specialized statistical methods.

In Chapter 3, we proposed a methodology for fault detection in a small power distribution system based on a suitably developed moving window change-point analysis technique. Our investigations showed that three-phase current is the most important variable, and a simplified version of our procedure using only three-phase current at all buses achieved perfect fault detection without false alarms. We also explored other options that yielded near-perfect results for our dataset of 55 faults, demonstrating the soundness of the general approach. Our algorithm summarizes the entire procedure and can be adapted to different grid topologies with some potential for false alarms or missed faults. While not an engineering solution, the methodology demonstrates the potential of data-driven approaches and establishes a scalable paradigm.

In Chapter 4, we extended the work done in Chapter 2. The approach described in Chapter 2 assumes that PMUs are placed at all buses. So far, power grids are only partially covered by PMUs. To address it, we proposed the use of Graph Neural Networks (GNNs) to reconstruct PMU trajectories for fault localization in power grids. Unlike approaches suggested by other researchers, our network do not require labeled data or knowledge of the exact fault location during training. We

examined two scenarios for data availability and demonstrated that assuming full data availability during training leads to accurate trajectory reconstruction and fault localization. However, when we relaxed this assumption, the second scenario failed to generalize to unobserved trajectories due to differences in trajectory levels before the fault. To address this, we experimented with a different loss function and rescaled data, and showed the improvements in fault localization, particularly in identifying the near-end bus of the faulted line.

Possible extensions and future work on the topics covered by this dissertation include the following directions. Firstly, it is important to note that the work presented in this dissertation is based on theoretical work and simulations and has not been implemented in real-world electrical power grids. The implementation could allow to better understand the proposed methods, and further develop them. Also, the methodology developed in Chapter 3 could be evaluated and potentially improved on more complex grid topologies. Additionally, considering the current technological limitations, the study was limited to smaller grids. To broaden the scope, it would be valuable to explore fault behavior variations resulting from factors such as fault path impedance and source impedance, which were not varied in this particular study. Therefore, future analysis could encompass a wider range of these parameters. Moving on to Chapter 4, it was observed that the reconstructed trajectories in both scenarios exhibited a higher variability than what actually exists in the power system. One potential approach to address this issue could involve introducing different network architectures and signal transformations that enable better control of such discrepancies. Furthermore, the current work focused solely on the modulus of voltage. In the future, it would be beneficial to incorporate other data provided by PMUs to assess whether increased accuracy can be achieved.

References

Abiri, E., Rashidi, F. and Niknam, T. (2015). An optimal PMU placement method for power system observability under various contingencies. *International Transactions on Electrical Energy Systems*, **25,** number 4, 589–606.

Abiri, E., Rashidi, F., Niknam, T. and Salehi, M. (2014). Optimal PMU placement method for complete topological observability of power system under various contingencies. *International Journal of Electrical Power & Energy Systems*, **61,** 585–593.

Afonin, A. and Chertkov, M. (2021). Which neural network to choose for post-fault localization, dynamic state estimation, and optimal measurement placement in power systems? *Frontiers in Big Data*, **4**.

Aminifar, F., Khodaei, A., Fotuhi-Firuzabad, M. and Shahidehpour, M. (2010). Contingency-constrained PMU placement in power networks. *IEEE Transactions on Power Systems*, **25,** number 1, 516–523.

Aminikhanghahi, S., Wang, T. and Cook, D. (2019). Real-time change point detection with application to smart home time series data. *IEEE Transactions on Knowledge and Data Engineering*, **31,** 1010–1023.

Appleby, G., Liu, L. and Liu, L. (2020). Kriging convolutional networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, **34,** 3187–3194.

Ardakanian, O., Yuan, Y., Dobbe, R., von Meier, A., Low, S. H. and Tomlin, C. J. (2016). Event detection and localization in distribution grids with phasor measurement units. *CoRR*.

Bartos, K., Rehak, M. and Krmicek, V. (2011). Optimizing flow sampling for network anomaly detection. In *Proc. 7th Int. Conf. Wireless Communications and Mobile Computing (IWCMC)*, pp. 1304–1309.

Basseville, M., Nikifirov, I. V. and Tartakovsky, A. (2012). *Sequential Analysis: Hypothesis Testing and Change–Point detection*. Chapman & Hall/CRC.

Bengio, Y., Courville, A. and Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **35,** number 8, 1798–1828.

Brodsky, B. E. (2017). *Change-Point Analysis in Nonstationary Stochastic Models*. CRC Press.

Brodsky, B. E. and Darkhovsky, B. S. (1993). *Nonparametric Methods in Change–Point Problems*. Kluwer.

Byrne, R. H., Concepcion, R. J., Neely, J., Wilches-Bernal, F., Elliott, R. T., Lavrova, O. and Quiroz, J. E. (2016). Small signal stability of the western north american power grid with high penetrations of renewable generation. In *Proc. of 2016 IEEE 43rd Photovoltaic Specialists Conference (PVSC)*, *Portland, OR*, pp. 1784–1789.

Chen, Hongtian, Jiang, Bin, Chen, Wen and Yi, Hui (2018). Data-driven detection and diagnosis of incipient faults in electrical drives of high-speed trains. *IEEE Trans. Ind. Electron.*, **66,** number 6, 4716–4725.

Chen, Hongtian, Yi, Hui, Jiang, Bin, Zhang, Kai and Chen, Zhiwen (2019). Data-driven detection of hot spots in photovoltaic energy systems. *IEEE Trans. Syst. Man Cybern. Syst.*, **49,** number 8, 1731–1738.

Chen, J. and Gupta, A. K. (2011). *Parametric Statistical Change Point Analysis: With Applications to Genetics, Medicine, and Finance*. Birkhäuser.

Chen, Tianrui, Hill, David John and Wang, Cong (2020). Distributed fast fault diagnosis for multimachine power systems via deterministic learning. *IEEE Trans. Ind. Electron.*, **67,** number 5, 4152–4162.

Chen, Y., Banerjee, T., Domínguez-García, A. and Veeravalli, V. (2016). Quickest line outage detection and identification. *IEEE Transactions on Power Systems*, **31,** number 1, 749–758.

Cheung, K. W., Chow, J. and Rogers, G. (2009). Power system toolbox, v 3.0. Technical Report. Rensselaer Polytechnic Institute and Cherry Tree Scientific Software,.

Chow, Joe H., Chakrabortty, Aranya, Arcak, Murat, Bhargava, Bharat and Salazar, Armando (2007). Synchronized phasor data based energy function analysis of dominant power transfer paths in large power systems. *IEEE Transactions on Power Systems*, **22,** number 2, 727–734.

Collobert, R., Kavukcuoglu, K. and Farabet, C. (2011). Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*.

Csörgő, M. and Horváth, L. (1997). *Limit Theorems in Change-Point Analysis*. Wiley.

Cui, M., Wang, J., Tan, J., Florita, A. R. and Zhang, Y. (2019). A novel event detection method using PMU data with high precision. *IEEE Transactions on Power Systems*, **34,** number 1, 454–466.

Cuturi, M. and Blondel, M. (2017). Soft-dtw: a differentiable loss function for time-series. In *Proceedings of the 34 th International Conference on Machine Learning*, volume 70, pp. 894–903.

Devi, M. Meenakshi, Geethanjali, M. and Devi, A. Rama (2018). Fault localization for transmission lines with optimal phasor measurement units. *Computers and Electrical Engineering*, **70,** 163–178.

Dragalin, V. P., Tartakovsky, A. G. and Veeravalli, V. V. (1999). Multihypothesis sequential probability ratio test–part i: Asymptotic optimality. *IEEE Transactions on Information Theory*, **45,** 2448–2461.

Enshaee, A., Hooshmand, R. and Fesharaki, F. (2012). A new method for optimal placement of phasor measurement units to maintain full network observability under various contingencies. *Electric Power Systems Research*, **89,** 1–10.

Estrada Gómez, A. M., Li, D. and Paynabar, K. (2022). An adaptive sampling strategy for online monitoring and diagnosis of high-dimensional streaming data. *Technometrics*, **64,** 253–269.

Foggo, B. and Yu, N. (2022). Online PMU missing value replacement via event-participation decomposition. *IEEE Transactions on Power Systems*, **37,** number 1, 488–496.

Follum, J., Pierre, J. W. and Martin, R. (2017). Simultaneous estimation of electromechanical modes and forced oscillations. *IEEE Transactions on Power Systems*, **32,** number 5, 3958–3967.

Furse, M., Kafal, M., Razzaghi, R. and Shin, Y. (2021). Fault diagnosis for electrical systems and power networks: A review. *IEEE Sensors Journal*, **21,** number 2, 888–906.

Gholami, A., Srivastava, A. and Panday, S. (2019). Data-driven failure diagnosis in transmission protection system with multiple events and data anomalies. *Journal of Modern Power Systems and Clean Energy*, **7,** 767–778.

Ghosal, Amrita and Conti, Mauro (2019). Key management systems for smart grid advanced metering infrastructure: A survey. *IEEE Commun. Surv. Tutor.*, **21,** number 3, 2831–2848.

Giani, A., Bitar, E., Garcia, M., McQueen, M., Khargonekar, P. and Poolla, K. (2013). Smart grid data integrity attacks. *IEEE Transactions on Smart Grid*, **4,** number 3, 1244–1253.

Glover, J. D., Sarma, M. S., Overbye, T. and Birchfield, A. (2022). *Power System Analysis and Design*, 7th edn. Cengage.

Goodfellow, I., Bengio, Y. and Courville, A. (2016). *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

Guen, V. and Thome, N. (2022). Deep time series forecasting with shape and temporal criteria. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Gustafsson, F. (2000). *Adaptive filtering and change detection*. Wiley.

Häger, M., Sollerkvist, F. and Bollen, M. (2006). The impact of distributed energy resources on distribution-system protection. In *Proceedings of Nordic Distribution and Asset Management Conference (Nordac)*.

Hamilton, W. (2020). Graph representation learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, **14,** number 3, 1–159.

Hamilton, W., Ying, R. and Leskovec, J. (2017). Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Han, J., Miao, S., Li, Y., Yang, W. and Yin, H. (2022). Fault diagnosis of power systems using visualized similarity images and improved convolution neural networks. *IEEE Systems Journal*, **16,** number 1, 185–196.

Hannon, C., Deka, D., Jin, D., Vuffray, M. and Lokhov, A. Y. (2019). Real-time anomaly detection and classification in streaming PMU data.

Horváth, L., Kokoszka, P. and Wang, S. (2021). Monitoring for a change point in a sequence of distributions. *Annals of Statistics*, **49,** 2271–2291.

Hu, B., She, J. and Yokoyama, R. (2013). Hierarchical fault diagnosis for power systems based on equivalent-input-disturbance approach. *IEEE Transactions on Industrial Electronics*, **60,** 3529–3538.

Huang, L., Nguyen, X., Garofalakis, M., Jordan, M., Joseph, A. and Taft, N. (2007). In–network PCA and anomaly detection. In *Advances in Neural Information Processing Systems*, pp. 617–624. MIT Press.

IEEE (1986). IEEE recommended practice for protection and coordination of industrial and commercial power systems. *ANSI/IEEE Std 242-1986*.

Jakir, M. and Rahnamay–Naeini, M. (2021). State estimation in smart grids using temporal graph convolution networks. In *2021 North American Power Symposium (NAPS)*, pp. 01–05.

Jiang, Joe-Air, Chuang, Cheng-Long, Wang, Yung-Chung, Hung, Chih-Hung, Wang, Jiing-Yi, Lee, Chien-Hsing and Hsiao, Ying-Tung (2011). A hybrid framework for fault detection, classification, and location, Part I: Concept, structure, and methodology. *IEEE Trans. Power Delivery*, **26,** number 3, 1988–1998.

Kaci, A., Kamwa, I., Dessaint, L. and Guillon, S. (2014). Synchrophasor data baselining and mining for online monitoring of dynamic security limits. *IEEE Transactions on Power Systems*, **29,** number 6, 2681–2695.

Khushwant, R., Hojatpanahand, F., Ajaei, F. B. and Grolinger, K. (2021). Deep learning for high-impedance fault detection: Convolutional autoencoders. *Energies*, **14,** 3623.

Kim, D., Chun, T. Y., Yoon, S., Lee, G. and Shin, Y. (2017). Wavelet-based event detection method using pmu data. *IEEE Transactions on Smart Grid*, **8,** number 3, 1154–1162.

Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12.

Kipf, T. and Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*.

Klambauer, G., Unterthiner, T., Mayr, A. and Hochreiter, S. (2017). Self-normalizing neural networks. In *Advances in Neural Information Processing Systems*, volume 30, pp. 971–980. Curran Associates, Inc.

Kokoszka, P., Rimkus, M., Hosur, S., Duan, D. and Wang, H. (2023). Detection and localization of faults in a regional power grid. *Austrian Journal of Statistics*, **52,** 143–162.

Lai, T. L. (1998). Information bounds and quick detection of parameter changes in stochastic systems. *IEEE Transactions on Information theory*, **44,** 2917–2929.

Lévy-Leduc, C. and Roueff, F. (2009). Detection and localization of change-points in high-dimensional network traffic data. *The Annals of Applied Statistics*, **3,** 637–662.

Li, W. and Deka, D. (2021). Physics-informed learning for high impedance faults detection. In *2021 IEEE Madrid PowerTech*, pp. 1–6.

Li, W., Deka, D., Chertkov, M. and Wang, M. (2019a). Real-time faulted line localization and PMU placement in power systems through convolutional neural networks. *IEEE Transactions on Power Systems*, **34,** number 6, 4640–4651.

Li, W., Deka, D., Chertkov, M. and Wang, M. (2019b). Real-time faulted line localization and PMU placement in power systems through convolutional neural networks.

Li, Y., Yu, R., Shahabi, C. and Liu, Y. (2018). Diffusion graph convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations*.

Liao, M. and Chakrabortty, A. (2019). Optimization algorithms for catching data manipulators in power system estimation loops. *IEEE Transactions on Control Systems Technology*, **27,** number 3, 1203–1218.

Liao, W., Bak-Jensen, B., Pillai, J., Wang, Y. and Wang, Y. (2022). A review of graph neural networks and their applications in power systems. *Journal of Modern Power Systems and Clean Energy*, **10,** number 2, 345–360.

Lorden, G. (1971). Procedures for reacting to a change in distribution. *The Annals of Mathematical Statistics*, **42,** 1897–1908.

Mei, Y. (2005). Information bounds and quickest change detection in decentralized decision systems. *IEEE Transactions on Information Theory*, **51,** 2669–2681.

Moghaddass, Ramin and Wang, Jianhui (2017). A hierarchical framework for Smart Grid anomaly detection using large-scale smart meter data. *IEEE Trans. Smart Grid*, **9,** number 6, 5820–5830.

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

Onaolapo, A. K., Akindeji, K. and Adetiba, E. (2019). Simulation experiments for faults location in smart distribution networks using ieee 13 node test feeder and artificial neural network. In *Journal of Physics Conference Series 1378:032021*.

Page, E. S. (1954). Continuous inspection schemes. *Biometrika*, **41,** 100–105.

Pan, S., Morris, T. and Adhikari, U. (2015). Classification of disturbances and cyber-attacks in power systems using heterogeneous time-synchronized data. *IEEE Transactions on Industrial Informatics*, **11,** number 3, 650–662.

Pandey, S., Srivastava, A. and Amidan, B. (2020). A real time event detection, classification and localization using synchrophasor data. *IEEE Transactions on Power Systems*, **35,** 4421–4431.

Park, S., Gama, F., Lavaei, J. and Sojoudi, S. (2023). Distributed power system state estimation using graph convolutional neural networks. *Proceedings of the Hawaii International Conference on System Sciences*.

Paschalidis, I. C. and Smaragdakis, G. (2009). Spatio–temporal network anomaly detection by assessing deviations of empirical measures. *IEEE/ACM Trans. Networking*, **17,** 685–697.

Percival, D. B. and Walden, A. T. (2000). *Wavelet Methods for Time Series Analysis*. Cambridge University Press, Cambridge.

PES, IEEE (2020). Ieee 13 node test feeder document. http://sites.ieee.org/pes-testfeeders/resources/. Accessed March 30, 2021.

R Core Team (2022). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Rafferty, M., Liu, X., Laverty, D. M. and McLoone, S. (2016). Real-time multiple event detection and classification using moving window pca. *IEEE Transactions on Smart Grid*, **7,** number 5, 2537–2548.

Raghavan, V. and Veeravalli, V. (2010). Quickest change detection of a markov process across a sensor array. *IEEE transactions on information theory*, **56,** 1961–1981.

Rassam, M. A., Zainala, A. and Maarof, M. (2013). An efficient distributed anomaly detection model for wireless sensor networks. In *Proc. Conf. Parallel and Distributed Computing and Systems (AASRI Procedia)*, pp. 9–14.

Rimkus, M., Kokoszka, P., Prabakar, K. and Wang, H. (2023). Real-time power grid fault detection. *Communications in Statistics: Case Studies, Data Analysis and Applications*; Forthcoming.

Ringsquandl, M., Sellami, H., Hildebrandt, M., Beyer, D., Henselmeyer, S., Weber, S. and Joblin, M. (2021). Power to the relational inductive bias: Graph neural networks in electrical power grids. *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*.

Salehi-Dobakhshari, A. and Ranjbar, A. (2014). Application of synchronised phasor measurement to wide-area fault diagnoses and location. *Generation, Transmission and Distribution, IET*, **8,** 716–729.

Scarselli, F., Gori, M., Tsoi, A., Hagenbuchner, M. and Monfardini, G. (2009). Computational capabilities of graph neural networks. *IEEE Transactions on Neural Networks*, **20,** number 1, 81–102.

Shafiullah, M. and Abido, M. (2018). S-transform based FFNN approach for distribution grids fault detection and classification. *IEEE Access*, **6,** 8080–8088.

Shafiullah, M., Abido, M. and Al-Mohammed, A. (2022). *Power System Fault Diagnosis*. Elsevier Inc.

Tartakovsky, A., Nikifirov, I. V. and Basseville, M. (2015). *Sequential Analysis: Hypothesis Testing and Change–Point detection*. CRC Press.

Tripathi, Sharda and De, Swades (2018). Dynamic prediction of powerline frequency for wide area monitoring and control. *IEEE Trans. Ind. Inf.*, **14,** number 7, 2837–2846.

Trudnowski, D., Kosterev, D. and Undrill, J. (2013). Pdci damping control analysis for the western north american power system. In *Proc. of 2013 IEEE Power & Energy Society General Meeting*, *Vancouver, BC*, pp. 1–5.

Vaughan, J., Stoev, S. and Michailidis, G. (2013). Network–wide statistical modeling, prediction and monitoring of computer traffic. *Technometrics*, **55,** 79–93.

Wu, Y., Zhuang, D., Labbe, A. and Sun, L. (2020). Inductive graph neural networks for spatiotemporal kriging. In *AAAI Conference on Artificial Intelligence*.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C. and Yu, P. (2019). A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, **32,** 4–24.

Xie, L., Chen, Y. and Kumar, P.R. (2014). Dimensionality reduction of synchrophasor data for early event detection: Linearized analysis. *IEEE Transactions on Power Systems*, **29,** number 6, 2784–2794.

Xie, Y. (2012). Statistical signal detection with multi-sensor and sparsity. Ph.D. Thesis. Stanford University.

Xie, Y. and Siegmund, D. (2013). Sequential multisensor change-point detection. *Annals of Statistics*, **41,** 670–692.

Xu, K., Hu, W., Leskovec, J. and Jegelka, S. (2018). How powerful are graph neural networks? In *International Conference on Learning Representations*.

Xu, Z., Xue, Y. and Wong, K. (2014). Recent advancements on smart grids in china. *Electric Power Components and Systems*, **42**.

Yadav, R., Pradhan, A. K. and Kamwa, I. (2019). Real-time multiple event detection and classification in power system using signal energy transformations. *IEEE Transactions on Industrial Informatics*, **15,** number 3, 1521–1531.

Yin, Shen, Li, Xianwei, Gao, Huijun and Kaynak, Okyay (2014). Data-based techniques focused on modern industry: An overview. *IEEE Trans. Ind. Electron.*, **62,** number 1, 657–667.

Yu, B., Yin, H. and Zhu, Z. (2018). Spatio-temporal graph convolutional neural. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, IJCAI'18, p. 3634–3640. AAAI Press.

Zainab, A., Refaat, S., Syed, D., Ghrayeb, A. and Abu-Rub, H. (2019). Faulted line identification and localization in power system using machine learning techniques. In *2019 IEEE International Conference on Big Data (Big Data)*, pp. 2975–2981.

Zhang, A., Lipton, Z., Li, M. and Smola, A. (2023). Dive into Deep Learning.

Zhang, N., Siegmund, D., Ji, H. and Li, J. (2010). Detecting simultaneous changepoints in multiple sequences. *Biometrika*, **97,** 631–645.

Zhang, P., Qian, K., Zhou, C., Stewart, B. and Hepburn, D. (2012). A methodology for optimization of power systems demand due to electric vehicle charging load. *Power Systems, IEEE Transactions on*, **27,** 1628–1636.

Zhang, Y., Wang, R. and Shao, X. (2021). Adaptive inference for change points in high-dimensional data. *Journal of the American Statistical Association*, **000,**; Published online: 27 Apr 2021.

Zhao, M. and Barati, M. (2021). A real-time fault localization in power distribution grid for wildfire detection through deep convolutional neural networks. *IEEE Transactions on Industry Applications*, **57,** number 4, 4316–4326.

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C. and Sun, M. (2020). Graph neural networks: A review of methods and applications. *AI Open*, **1,** 57–81.

Zhou, M., Wang, Y., Srivastava, A. K., Wu, Y. and Banerjee, P. (2019). Ensemble-based algorithm for synchrophasor data anomaly detection. *IEEE Transactions on Smart Grid*, **10,** number 3, 2979–2988.

Zhu, L. and Lin, J. (2021). Learning spatiotemporal correlations for missing noisy PMU data correction in smart grid. *IEEE Internet of Things Journal*, **8,** number 9, 7589–7599.