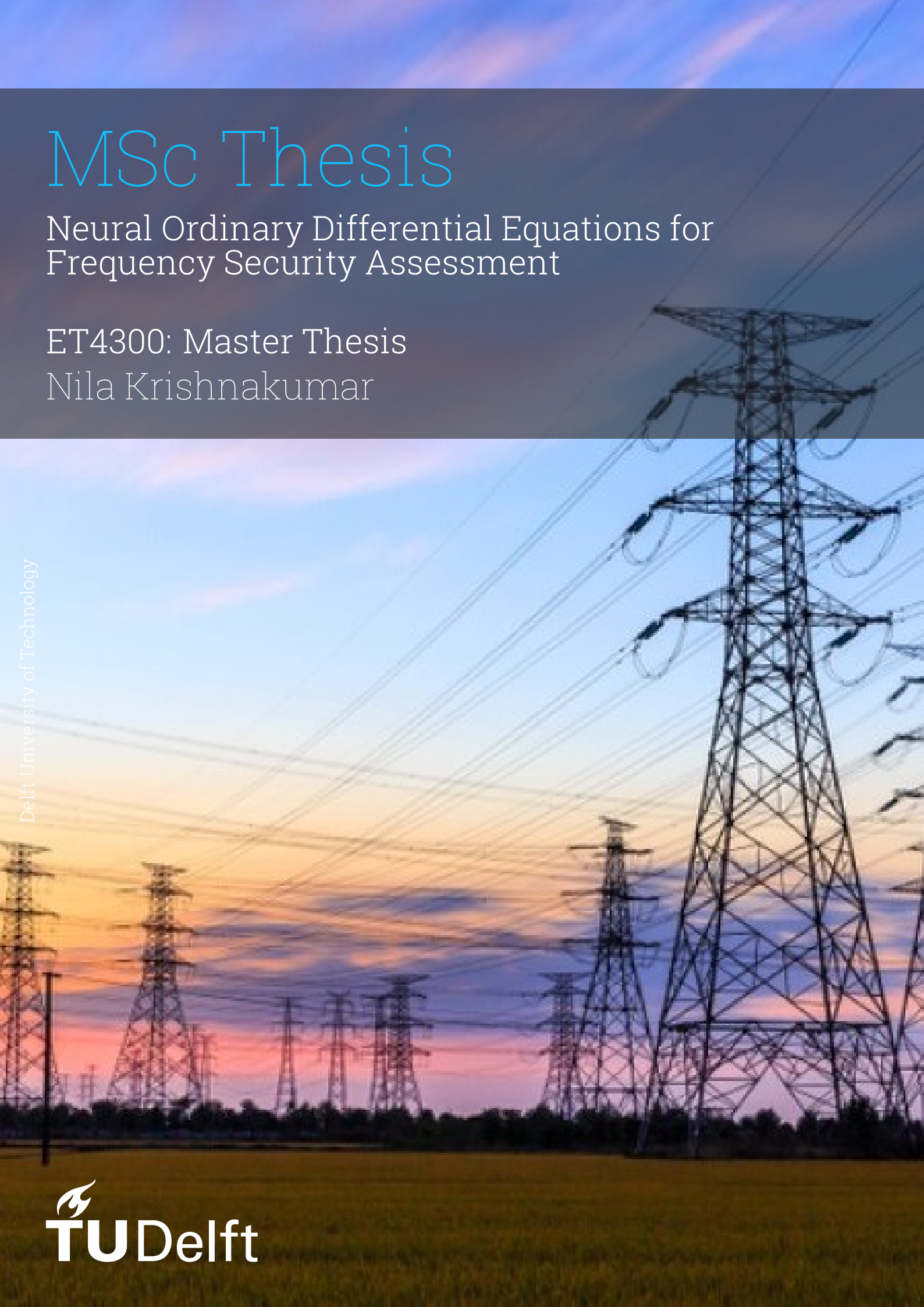
MSc Thesis

Neural Ordinary Differential Equations for Frequency Security Assessment

ET4300: Master Thesis

Nila Krishnakumar



Delft University of Technology

TU Delft

MSc Thesis

Neural Ordinary Differential Equations for Frequency Security Assessment

by

Nila Krishnakumar - 5505364

to obtain the degree of Master of Science at the Delft University of Technology, to be defended publicly on Friday the 22nd of September, 2023 at 10:00.

Student number: 5505364
Thesis Duration: January, 2023 - September, 2023

This thesis was carried out in collaboration with Reddyn B. V.

Thesis Supervisor:	Dr. Jochen Cremer, Assistant Professor
Company Supervisor:	Mr. Martijn Janssen, Alliander N. V.
Thesis Committee Chair:	Dr. Ir. José Luis Rueda Torres, Associate Professor
Thesis Committee Member:	Dr. Aditya Shekhar, Assistant Professor
Faculty:	Faculty of Electrical Engineering, Mathematics and Computer Sciences

Cover:	Transmission lines, image retrieved from `https://usea.org/regional-partnerships/south-asia-regional-energy-hub-sareh`
Style:	TU Delft Report Style, with modifications by Daan Zwaneveld

TU Delft

Abstract

To keep pace with increasing renewable energy penetration and consequent increase in inverter-based resources in the power grid, it is pertinent for present-day research to address the resulting drop in system inertia levels and its impact on frequency stability. With decreasing levels of inherent rotational inertia present in the system, any sudden disturbance causing an energy imbalance in the grid could lead to more drastic excursions of system frequency than those experienced hitherto. To ensure the resilience of the grid in such scenarios, advanced and competent frequency stability assessment and control methods are required. This thesis presents Neural Ordinary Differential Equations (NODE), a recently introduced family of neural networks, as an effective tool to achieve fast, real time estimates of the expected frequency response trajectory during an energy imbalance event.

Since high-impact frequency instability events are sparse in reality, both real-world grid data and synthetically generated data corresponding to different inertial conditions are used to train predictive NODE models. Firstly, NODE is adapted to frequency prediction applications through relevant data processing steps, and modification of network parameters and algorithmic aspects pertaining to the predictive model definition. Secondly, patterns corresponding to specific sections of the frequency response curve are used to selectively train NODE models. Pattern-specific training methods exhibit better prediction performance when the NODE model encounters frequency behaviour similar to the one it initially trained on. Thirdly, a pre-training approach to cut short on the real-time training time required by NODE models to achieve desired levels of prediction performance is presented. Fast estimates of critical frequency stability parameters like nadir could act as potential triggers for early stability control actions to achieve a more controlled frequency response.

Application of predictive NODE models for different frequency scenarios are presented using three test-cases: normal operating scenario, restoration post-system split scenario and synthetically generated high-impact frequency disturbance scenarios. Model tuning and training methods specific to each test-case are described, and prediction results are evaluated with relevant performance metrics. Finally, a comparison is made between the implementation of NODE among different test-cases and real-world implications of the frequency prediction outcomes from the test-cases are further discussed.

Acknowledgements

I would like to take this space to show my gratitude to everyone that made the long journey of my MSc thesis at TU Delft a very enriching and fulfilling experience.

Firstly, I would like to thank my supervisor at TU Delft, Dr. Jochen Cremer, and my daily PhD supervisor, Mr. Mert Karaçelebi for their consistent and unwavering support throughout the thesis. I have always looked forward to each of our discussions for Jochen's insightful comments and a string of new, exciting ideas to work on. Furthermore, I am grateful for Mert's patience and solid guidance on all aspects of my thesis, right from our initial interaction regarding the thesis topic an year ago.

Next, I would like to thank my company supervisor, Mr. Martijn Janssen, for providing me with all the technical and logistical support required to carry out this thesis, and for always looking out for any kind of assistance I might need from the company. I would also like to thank all the concerned members at Reddyn B. V. for their critical support in providing me with necessary grid data and for every meeting arranged with various power system experts in the industry that helped bring a more holistic approach to my thesis.

I take this opportunity to extend my thanks to my professor whose very intriguing power system courses I have had the fortune of taking up over the last two years, Dr. Ir. José Luis Rueda Torres, and Dr. Aditya Shekhar for agreeing to be part of the thesis committee and evaluating my work. I would also like to thank everyone from Jochen's research team, Delft AI Energy Lab and my MSc colleagues for our regular thesis update meetings, practical tips and exchange of constructive feedback.

Last, but importantly, I would like to thank my most dear family and friends, from near and afar, for always standing by me and showing only their immense love and firm faith in everything I set out to do. They sure do make the world go round for me.

Nila Krishnakumar,
Delft, September 2023

Contents

List of Figures

List of Tables

Nomenclature

Abbreviations

Abbreviation	Definition
RES	Renewable Energy Sources
EV	Electric Vehicles
PE	Power Eelctronics
RoCoF	Rate-of-Change-of-Frequency
PMU	Phasor Measurement Unit
RMS	Root Mean Squared
ANN	Artificial Neural Networks
LoG	Loss-of-Generation
DSA	Dynamic Security Assessment
DSC	Dynamic Security Control
ML	Machine Learning
DT	Decision Trees
SVM	Support Vector Machines
UFLS	Under-frequency Load Shedding
ESS	Energy Storage Systems
MLP	Multi-layer Perceptron
RNN	Recurrent Neural Network
CNN	COnvolutional Neural Network
LSTM	Long Short-term Memory
NODE	Neural Ordinary Differential Equations
SciML	Scientific Machine Learning
PDE	Partial Differential Equation
AI	Artificial Intelligence
GD	Gradient Descent
ReLU	Rectified Linear Unit
ELU	Exponential Linear Unit
RL	Reinforcement Learning
MSE	Mean Square Error
ResNet	Residual Network
ODE	Ordinary Differential Equation
IVP	Initial Value Problem
ACA	Adaptive Checkpoint Adjoint
RMSE	Root Mean Square Error
AUC - ROC	Area Under the Curve - Receiver Operator Characteristic

Abbreviation	Definition
ENTSO-E	European Network of Transmission System Operators for Electricity
AGC	Automatic Generation Control
LFC	Load Frequency Control
ACE	Area Control Error
FFR	Fast Frequency Response
TSO	Transmission System Operator
LFDD	Low-frequency Demand Disconnect
WECC	Western Electricity Coordinating Council
BFGS	Broyden-Fletcher-Goldfarb-Shanno
MAE	Mean Absolute Error

1

Introduction & Literature Review

1.1. Background and Motivation

As electrical power systems continuously evolve and adapt to keep up with the increasing demands placed upon electricity grids, state-of-art dynamic security assessment systems become indispensable. Higher penetration of renewable energy sources (RES) and developing futuristic grids that can support Electric Vehicles (EVs) or flexible energy consumption lead to significant variations in the dynamic behaviour and stability of power systems. This consequently calls for advancements in real-time monitoring and control of power system dynamics [1], both in terms of accuracy in event predictions and speed of control actions.

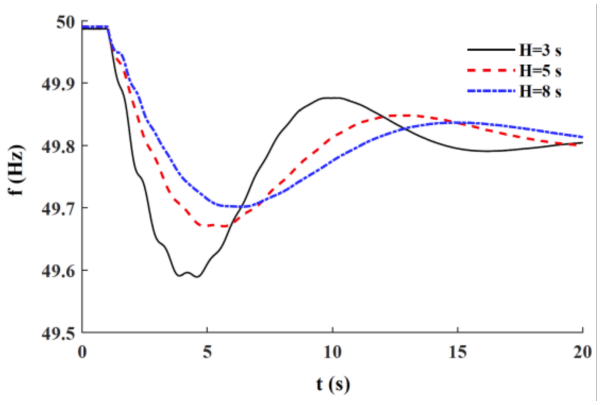


Figure 1.1: Impact of inertia levels on system frequency response, figure retrieved from [2]

Amongst the classification of types of power system stability [3], increasing integration of renewable energy with conventional power grids would have a profound impact on frequency stability. The rising levels of RES penetration and the consequent presence of non-traditional power electronics (PE) - interfaces for distributed energy sources would have an impact on the system frequency response that follows a power system disturbance. Unlike the innate rotational inertia present in traditional synchronous generators, PE-interfaced machines cannot provide sufficient inertia or damping response. The impact of different inertia levels on post-disturbance frequency response is represented in Figure 1.1. Hence, the frequency fluctuations might have a tendency to shoot to relatively higher/lower values than before; insufficient damping of the instability-induced frequency oscillations could lead to larger frequency nadirs and higher rate-of-change-of-frequency (RoCoF) values.

With the higher inter-dependencies and interconnections of power system networks across countries, a large-scale power system disturbance like a sudden loss of a major generation unit or disconnection of a critical area that supplies power to the network could have high impact consequences. These could potentially have a cascading effect across networks, leading to a large scale power outage or a blackout. Some of these low frequency, high impact situations have led to major economic and social losses for the affected areas and its consumers in different parts of the world [4]. There is also the added threat of cybersecurity-related attacks on power grids in the recent times. In order to improve the power systems to be more resilient to such disturbances, a superior dynamic security assessment and control system is required [5]. Faster assessment of such events, combined with timely control actions can help prevent high impact losses; it would pave way for implementation of data-informed and improved stability restoration methods.

1.2. Literature Review

1.2.1. Data-driven Methods in Power System Security Assessment

With respect to high impact disturbances, the instability in the system dynamics cascades to a larger area of the network in a very short period of time. In such situations, the conventional real-time Root Mean Squared (RMS) simulation and processing of Phasor Measurement Unit (PMU) grid measurements might not be fast enough to trigger/implement the required control actions in time. Integrating data-driven methods into these assessment methods can enhance the performance of stability control by faster prediction of the resulting dynamic response or the extent of expected overshoot/drop/oscillations in the suitable power system quantities (for instance, voltage or frequency). If these predictions are to be used to trigger major control actions across the grid, it is critical that the data-driven method provides an adequately accurate estimation.

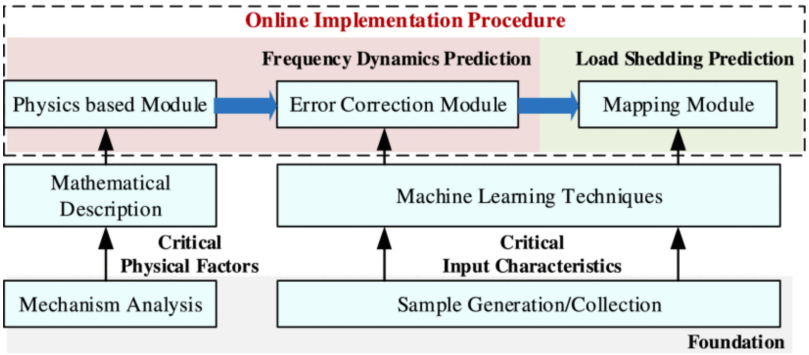


Figure 1.2: An example of an integrated approach for power system frequency stability and control, figure retrieved from [6]

Artificial Neural Networks (ANN) is a machine learning model that is capable of estimating various power system quantities (for instance, dynamic state estimation [7], Loss-of-Generation (LoG) size estimation [8]) using time-series measurement data, and by employing suitable input data preprocessing techniques. Neural networks offer the flexibility of changing their network definition or aspects like length of the input training data to achieve an improvement in the performance of the prediction model. With several good examples of predictive mathematical and ANN models in the literature [9, 10, 11], it could be observed that these

models have an ability to work very fast with low computation times in real-time. However, at the same time, it is important that the model is capable of providing estimates/information capable of triggering suitable control actions to mitigate system instability events [12].

In some instances of power system assessment and control, integrating data-driven approaches to pre-existing model-based approaches could prove to improve the speed and efficiency of the model without adversely affecting its accuracy [6]. The approach in [6], for example, achieves this improvement by enabling real-time error correction of system state estimates using a machine learning model that continuously trains in real time (see Figure 1.2). Using error-corrected state estimates to trigger control actions early in time could result in a more controlled dynamic response and faster restoration of the system (see Figure 1.5).

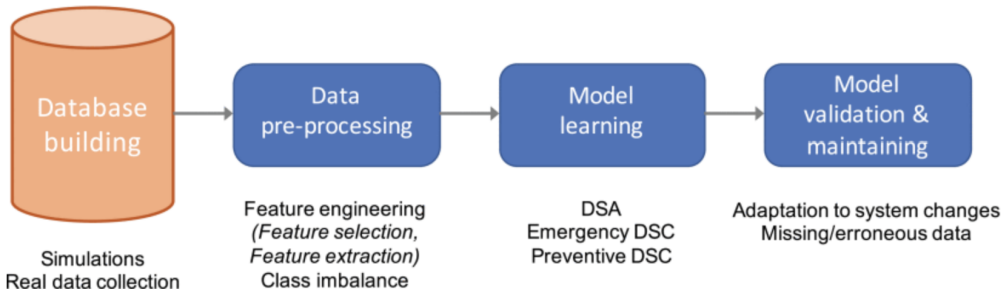


Figure 1.3: Basic steps in using ML for Dynamic Security Assessment (DSA) and Dynamic Security Control (DSC), figure retrieved from [13]

Consequently, an increasing application of data analytics and machine learning (ML) methods in ensuring the real-time dynamic security of power systems can be observed in the literature [13, 14]. The usual steps taken to implement DSA or DSC with ML are shown in Figure 1.3. The initial step of data collection is often a critical step, for the data must be representative of the security situation in concern. A common source for power system-related ML databases are PMU measurements which could provide real-time, synchronised voltage and current data in the system. ML methods could aid in tackling power system security problems through a range of approaches: from predicting possible security breaches expected in the future to decision making for triggering early control and restoration processes. [13] addresses elaborately the exploitation of a variety of ML models in the literature (ranging from deep learning algorithms to Decision Trees (DT)/Support Vector Machines (SVM) to ensemble methods) in three areas of dynamic security, namely security assessment, emergency control and preventive control.

Given the high stakes involved in power system security and operations, the challenges that exist in data-driven approaches must also be taken into consideration. In order to be accepted by the various stakeholders involved in electrical power systems (including system operators and their planning teams), data-driven approaches should show the required level of reliability and adaptability to keep up with the developments happening in power systems. These approaches should be easy to integrate with existing conventional approaches and support possibilities for continuous assessment and improvement of their efficacy in practical power system applications. The rising relevance of cybersecurity in electrical systems is another significant aspect to consider while assessing the vulnerability of ML approaches [15]. The possibility of false data injection can have far-reaching, adverse consequences on the power system [16]; un-

derstanding the vulnerabilities of the system is key in designing a robust and secure data-driven approach for power systems.

1.2.2. Frequency Security Assessment and Control

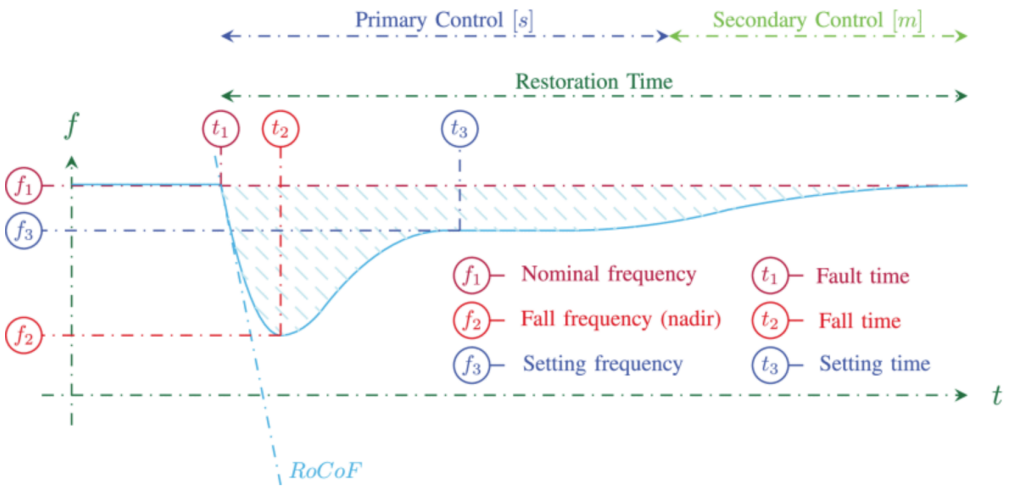


Figure 1.4: Typical frequency response curve, figure retrieved from [11]

Conventional frequency stability assessment methods use real-time measurements of frequency and RoCoF to detect frequency instabilities in the system [17, 18, 19]. As is the norm, the different frequency control schemes, namely primary control, secondary or load-frequency control, and tertiary control are implemented sequentially to bring the system back from an abnormal/unstable state to its stable operating state (see a typical post-disturbance frequency response curve and key curve parameters in Figure 1.4). Since frequency instability is often a direct consequence of active power (demand or supply) imbalance, the presence of generation reserves, flexible generator set points and ancillary services that enable control/regulation of system frequency play a crucial role [20]. In case of large scale load-generation imbalance, emergency control and protection schemes like Under-frequency Load Shedding (UFLS) scheme [21] are in place to prevent possible cascading situations.

With the influx of RES, the power supply becomes more variable and intermittent. This, in turn, requires more flexible reserves and ancillary services to mitigate possible instabilities in the system. The new assessment and control methods have to additionally compensate for the low levels of inertia available from the non-synchronous generating units. One possible approach for integrating capable flexibility services is to integrate power-electronics interfaced technologies, for instance say, Energy Storage Systems (ESS) that could induce virtual inertia in the system [22]. There also exist other suggested solutions such as having an inertia floor or compensating generators to supply the required inertia.

To further facilitate the integration of large shares of variable RES with their limited inertia capacity, demand-side flexibility and response [23] can prove to be helpful in providing ancillary services for effective frequency stability control. If there is higher centralisation in the control/switching of demand-side elements, it could aid in faster regulation of frequency in unstable situations involving a large or interconnected power network. A difference to be noted among the mentioned frequency response approaches is that UFLS is often employed in critical

situations whereas demand-side response could be employed on a continuous basis for normal operating conditions to improve the grid stability and resilience to potential disturbances.

1.2.3. Neural Networks in Predictive Analysis of Frequency

Exploitation of Neural Networks for time-series prediction in energy systems has been prevalent in the literature over the last few decades [24]. The predictive applications of ANN range from load/energy forecasting to stability analysis, security assessment and a few more areas in power system studies. The ability of ANN to fast process data, deal with non-linear relationships in the system and the available variability in terms of types (for instance, multi-layer perceptron (MLP), recurrent-neural network (RNN), convolutional-neural network (CNN) and long-short-term memory (LSTM) models), make them applicable for predictive analysis in power system problems.

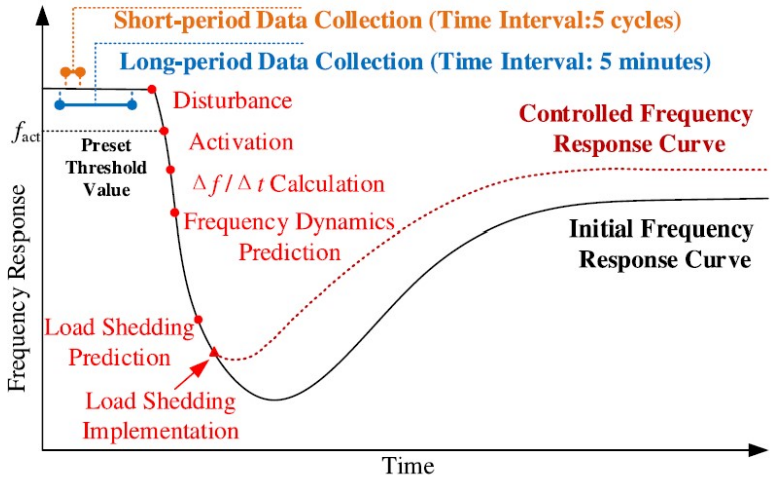


Figure 1.5: An example of achieving a controlled frequency response using a hybrid-model based frequency dynamics prediction approach, figure retrieved from [6]

As far as frequency prediction and analysis is concerned, there are a few examples of integrating machine learning methods to enhance the performance of conventional methods in the literature. In Figure 1.5, results from a hybrid approach (more information on the implementation procedure is shown in Figure 1.2) wherein conventional frequency dynamics prediction is supported by an extreme learning-based real-time error correction to achieve an improved frequency dynamics prediction is shown. It has been described in [6] about how this approach could lead to a more controlled frequency response. Similarly, utilising ANNs trained with time-series data in a purely data-informed model with high predictive capabilities (in terms of speed and accuracy) to predict future dynamic frequency response could help in implementing an early and controlled frequency restoration in the system.

Training ANNs to learn the dynamic behaviour of a system using representative input data could enable the system to predict future dynamic behaviour of the system for data from an unobserved system. [25] discusses two categories in frequency dynamics prediction, namely estimation of frequency characteristics (like RoCoF, nadir, breach of stability margins) and frequency curve prediction. In frequency analysis and security assessment, the focus of many ANN-based works is on the prediction of frequency nadir after a generation loss or disconnec-

tion in the system [26, 27]. As an example for short-term frequency forecasting, [28] proposes an LSTM model to predict the power system frequency trajectory for the subsequent minute.

1.3. Research Direction and Contribution

Developing advanced frequency security assessment methods that can keep up with the changes introduced to electrical power systems require new methods that are highly reliable, fast and accurate. In this thesis, Neural Ordinary Differential Equations (NODE) are introduced as a suitable family of neural networks that could be applied to predictive analysis of power system frequency to achieve an improved security assessment procedure. The suitability and the rationale behind choosing this specific type of ANN are discussed elaborately in the next chapter.

NODE can learn non-linear dynamic phenomenon by approximating differential equations governing the relationship between different parameters in a physical system. With careful processing and modelling of input data, tuning of model parameters and choosing relevant performance metrics, it is possible to achieve a frequency prediction algorithm that is computationally fast with an acceptable level of accuracy for monitoring and assessing the status of a system. Another aspect to consider is the ability of the algorithm to learn in real-time. Since frequency response trajectories are highly event-dependent and are quite unique among different events, the prediction algorithm should be able to learn in real-time post a disturbance/event to continuously update and improve its performance.

This thesis approaches the application of NODE in frequency security assessment on the basis of four aspects. The aspects and the contribution in each aspect are concisely stated below.

1. **Developing and tuning a NODE algorithm for different frequency prediction applications**
   A NODE model for frequency prediction must be able to take different power system quantities (while including frequency) as input features, and churn out expected frequency values in the near future. This requires an ANN that is capable of approximating the dynamics of frequency and related available system quantities. It is possible to achieve the desired level of frequency prediction performance by developing a NODE algorithm with proper model tuning (for example, the optimal number of hidden layers in the ANN or a well-performing sequential combination of optimizers).
2. **Approximation method of definite patterns observed in frequency behaviour using NODE model**
   Using the ability of NODE to learn non-linear patterns, it is possible to train a model to learn specific sections of a frequency response curve for a given test system. With prior knowledge of how a system reacts to a similar kind of disturbance, it is possible for the NODE model to enhance its prediction performance when an expected type of disturbance occurs. This thesis attempts to use a real-time frequency restoration curve after a major system-split event to train a model with the observed restoration pattern.
3. **Pre-training approach to achieve fast real-time predictions using short-term online retraining**
   It is required to cut short on the real-time training time if the NODE models are to be used for forecasting fast characteristics of frequency dynamics. By pre-training a model to learn a typical frequency response curve after a system disturbance, it is feasible to quickly estimate key frequency instability parameters like frequency nadir in real-time

after the detection of a frequency disturbance. This could help in relatively early triggering of frequency control actions to achieve a more controlled stability response in the system.

4. **Case studies to demonstrate differences in NODE performance when working with real system data and synthetically generated data**
   With real system data corresponding to major frequency disturbance events being sparse, it is necessary to work with both real system data and synthetically generated data. Working around the constraints posed by both the types of available data, NODE models are implemented for a few test cases concerning different frequency situations observed in power systems. With changing prediction requirements, these test cases illustrate the differences in defining NODE models and obtaining relevant prediction results using the two types of power system data.

1.4. Research Objective and Thesis Outline

The research objective of the thesis could be stated as:

"To use Neural Ordinary Differential Equations (NODE) for real-time frequency security assessment and subsequently enable timely frequency stability control."

In an attempt to reach the research objective, the following questions are sequentially addressed and discussed with relevant results over the course of this document.

1. How can Neural Ordinary Differential Equations be adapted to frequency dynamics predictions?
2. What are the challenges in obtaining relevant input data for training and testing NODE models?
3. Which aspects of the NODE algorithms need to be tuned to address different frequency security situations?
4. What are some possible real-world implications of the frequency prediction outcomes from NODE models?

The outline of the thesis report and the structure of its chapters are as follows. An introduction to the base theoretical knowledge required to carry out this thesis is discussed in the second Chapter - "Theoretical Background". Applying machine learning to physical systems, an introduction to ANNs and NODE, relevant data processing principles and improving model performance in ML algorithms are all encompassed in this chapter. The next chapter - "Predictive NODE Algorithm - Methodology" focuses on the work flow observed, starting from the data collection step to the code framework to the setting up of different case studies to obtain a proof of concept and working results. The results, their analysis and discussion is carried out with relevant plots and tables in the chapter - "Results & Discussion". To conclude the report, future scope, possible improvements and concluding remarks regarding the thesis work are stated in the final chapter - "Future Scope & Conclusion".

2

Theoretical Background

2.1. Machine Learning in Scientific Computing

Vast amounts of scientific computations in diverse fields ranging from aerospace to molecular sciences to macroeconomics have been dependent on mechanistic models in the past. While machine learning has managed to achieve great feats in areas like image recognition or natural language processing using "big data", many areas of computational science suffer from the lack of the right quantity or right set of data to build an accurate machine learning model [29]. As an attempt to deal with complex problems in different domains with insufficient scientific data, scientific machine learning (SciML) happens to be an emerging field in data science that acts as a bridge between machine learning and computational science in areas that require domain-specific knowledge.

As the name suggests, SciML is an interdisciplinary field that combines two hitherto independently evolving research areas - namely, machine learning and scientific or physics-based modelling. Some obstacles in integrating these two research areas are lack of sufficient scientific training data for ML, high requirements for reliability of ML based solutions for scientific systems and efficient utilisation of available theoretical knowledge to support ML-based scientific models. For instance, a common challenge for ML-based classification methods in scientific computing is to process an imbalanced data set. When the available data for different classes of classification are unequal in terms of quantity, it is difficult to train the ML model well. Taking the field of power system security assessment as an example, a major challenge is to obtain data that pertain to high-risk or emergency situations in the power system. Since such situations are quite sparse when compared to normal operational situations, the ML method should account for the data imbalance in a logical way.

There are many ways in which SciML proves useful in introducing novel methods to assess and model complicated science and engineering problems. [30] lists some application-centric objectives of employing SciML methods in any arbitrary scientific problem. These objectives include obtaining reduced-order models that are more computationally efficient, discovering underlying governing equations between different parameters, data generation to obtain realistic synthetic data and physics-informed forward solving partial differential equations (PDEs). A few approaches to achieve these objectives are also mentioned in [30] - physics-guided loss function, physics-guided initialization, physics-guided design of architecture and hybrid modeling. While choosing a relevant SciML approach, it would help to pay attention to the computational objectives and requirements of the scientific problem in hand. Some prevalent

computational objectives while using SciML in the literature are achieving better prediction performance in terms of speed, accuracy, sampling efficiency etc., and ensuring a good interpretability of the proposed SciML approach.

In frequency security assessment, the computational objectives of using a SciML approach in predictive analysis of frequency could range from improving prediction performance in terms of accuracy and speed, to easy interpretation of the proposed assessment method. In [6], hybrid modelling was used to achieve better prediction performance through better error correction methods. In this thesis, the proposed methodology takes subtle inspiration from some SciML approaches such as physics-guided loss function and initialisation to achieve better prediction performance of expected frequency response. Neural Networks, with a wide range of types to choose from based on the computational requirements of a scientific problem, could be a suitable ML method for integrating SciML approaches into power system security assessment methods. With all the new developments expected in our power systems (like large-scale RES integration, increasing penetration of EVs), achieving high speed and accuracy in security assessment is critical in ensuring the security of a more volatile and flexible electrical network. These requirements, combined with the need for high non-linear modelling capabilities to predict power system dynamics, make ANNs a relevant ML tool for future frequency security assessment methods.

2.2. Neural Networks - Overview

With a sweeping number of applications in a wide range of fields, ANNs are one of the most predominant ML tools in use today. As the name suggests, ANNs are touted to be inspired from the functioning of biological neurons in animal systems. Similar to how neurons in a human brain form extensive interconnections with one another to process complex patterns and information, an ANN is expected to process information for artificial intelligence (AI) based real-world applications. Each biological neuron receives input information from other cells or neurons, which is then passed on to other interconnected cells or neurons. While the complexity of and unsolved mysteries around biological neural networks could possibly lead to advancements in future ANNs, a typical artificial neuron as used in current applications can be depicted as shown in Figure 2.1.

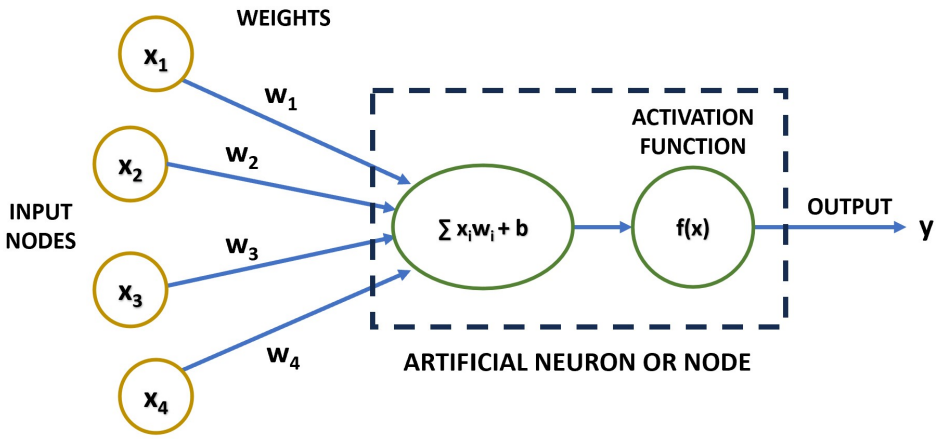


Figure 2.1: A typical artificial neuron or node

An artificial neuron, commonly referred to as a node, has a set of inputs, corresponding

weights, a bias, an activation function and an output. In Figure 2.1, x_i is any arbitrary input value, w_i is its associated weight and b is a bias value for a given node. For a set of input values coming from different nodes, a single neuron or node takes the weighted sum of inputs and processes it through an activation function f(x). The weights help decide how much significance is to be given to each input node in determining the output. Bias is a constant offset value added to the weighted sum of inputs before it is processed by the activation function. The bias value can shift the activation function across the horizontal axes, towards left or right. Hence, a change in the combination of weights and bias used can give rise to different values or outcomes. The output from the activation function determines to what extent the information from the given node is passed further on to the next set of nodes. As a simple example, a binary activation function that gives 0 or 1 as its output can be considered. If the output is 1, the node is activated/energised and the information it received is passed on to the next node. If the output is 0, the node is not activated and the information is not passed on further. Often, many activation functions output continuous values that correspond to the extent to which information is passed on from the given node as its output to the consequent set of nodes.

An ANN is, hence, an interconnection of multiple layers of artificial neurons capable of processing complex information to achieve desired computational results. Once trained, ANNs are capable of recognising complex patterns or approximating almost any non-linear system of equations. The various elements that form an ANN, the working and learning of an ANN and some prominent types of ANNs are discussed elaborately in the following sections.

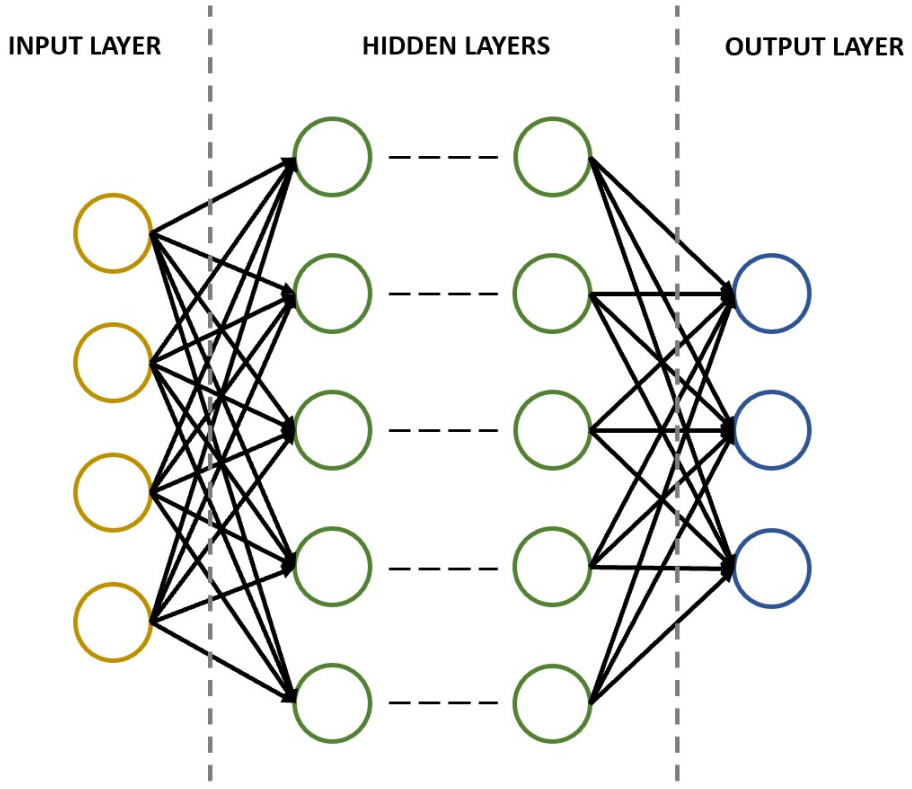


Figure 2.2: A simple multi-layer feed-forward ANN

2.2.1. Structure of Neural Networks

The structure of a simple feed-forward ANN (shown in Figure 2.2) is a good starting point for understanding the different elements that form an ANN. By feed-forward, it could be understood that information is propagated through the network in only one direction. An ANN generally has three types of layers: the input layer, one or more hidden layers and the output layer. The simplest form of a neural network could be a two layer network with an input layer and an output layer. The input layer is always a passive layer wherein all features or available input variables are passed on to all the nodes in the next layer. No changes are made to the data in this layer. The forward arrows or lines connecting these nodes to the next layer are representative of a specific value of weight. So, in case of a two layer network, the weighted sum of all inputs is passed through a transfer function (called as an "activation function") in the nodes of the output layer to give a final output variable. This output layer could consist of a single node in case of classification or multiple nodes when two or more output variables are required.

In order to achieve higher predicting capabilities, one or more hidden layers are added in an ANN. Deep neural networks generally have multiple hidden layers between the input and the output layer. The depth (number or hidden layers) and width (number of nodes in a hidden layer) of the hidden layers can be tuned optimally according to a given problem. In each node of the hidden layer, the weighted sum of data from the previous hidden layer or the input layer is passed through a specified activation function. The outputs from all the nodes of the hidden layer are once again passed through a different set of weights to each of the nodes in the next hidden layer or the output layer. The weights between nodes are initialised randomly at the beginning and subsequently tuned as the ANN model starts learning. While the predicting power of the ANN model increases with the number of layers, it is also important to limit the number of layers so as to avoid the problem of over-fitting. Having high complexity in the ANN structure and a limited amount of training data could lead to over-fitting of the ANN model with respect to the training data.

2.2.2. Activation Functions

Every node in a hidden layer or a output layer of an ANN has an associated activation function that transfers the weighted sum of inputs entering the node to an output value. More often than not, these activation functions are non-linear in nature and enable the ANN to possess non-linear predicting capabilities. In fact, without a non-linear activation function, an ANN would be equivalent to a linear regression model that cannot process complex information. The simplest activation function is a linear activation function or the identity function $f(x) = x$, typically used only in the output layers of ANNs. The other non-linear activation functions allow for non-linear combination of inputs over multiple layers to model complex problems successfully. Some prevalent activation functions in ANN applications, their characteristics and deciding how to choose an activation function for a given prediction problem are discussed next.

1. **Binary step function:** As shown in Figure 2.3, a binary step function has only two outputs: 0 and 1. When 0, the neuron remains inactive and when 1, the neuron is activated. With the bias value in a neuron, the output is decided based on whether a threshold limit has been crossed by the weighted sum of inputs or not. This function is applicable only in elementary binary classifier models as they are not capable of multi-class classifica-

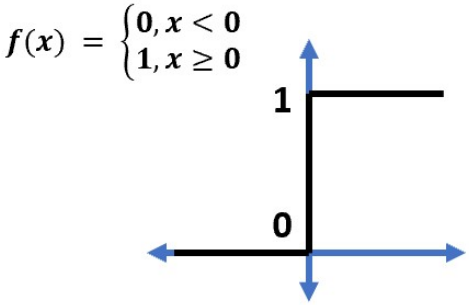


Figure 2.3: Binary step activation function

tion. Moreover, the function has a gradient of zero and is hence, not applicable for ANNs with gradient descent (GD) algorithms (discussed in subsection 2.2.3).

2. **Sigmoid or Logistic activation function:** This is a bounded activation function that gives an output only between 0 and 1. The sigmoid function is, hence, suitable for logistic regression or binary classification problems. Being continuously differentiable, the function has a non-zero derivative centred between -3 and 3 as shown in Figure 2.4. The derivative is not monotonic and that could lead to a vanishing gradient problem during training.

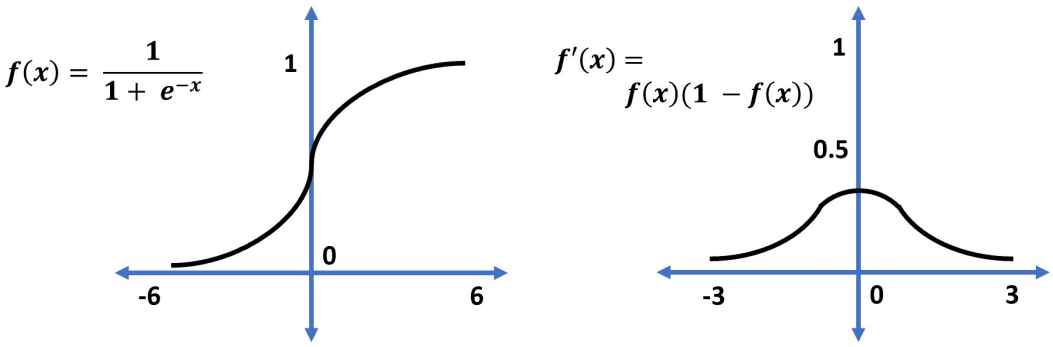


Figure 2.4: Sigmoid activation function and its derivative

3. **Hyperbolic tangent function - tanh:** The tanh function has a similar shape as the sigmoid function and bounds its output between -1 and 1. Having a zero-centred nature makes the function good at mapping data as negative, close to zero or positive to the next layer, making learning easier. As shown in Figure 2.5, the derivative of the tanh function has a steeper gradient compared to sigmoid. This could imply larger gradients or higher learning steps during learning. While vanishing gradients is still a problem for tanh activation, having a zero-centred nature allows for a less restricted gradient motion (in both positive and negative directions) as compared to the sigmoid function. Both sigmoid and tanh activation functions find many applications in RNN models.
4. **Rectified linear unit function - ReLU:** ReLU seems to be one of the most widely used activation functions in various types of ANN like CNN and other deep learning models. A major advantage of this function is its higher computational speed and faster conver-

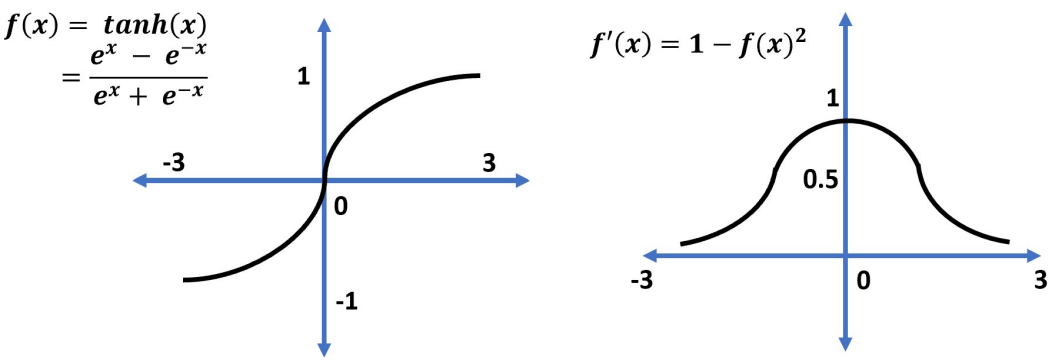


Figure 2.5: tanh activation function and its derivative

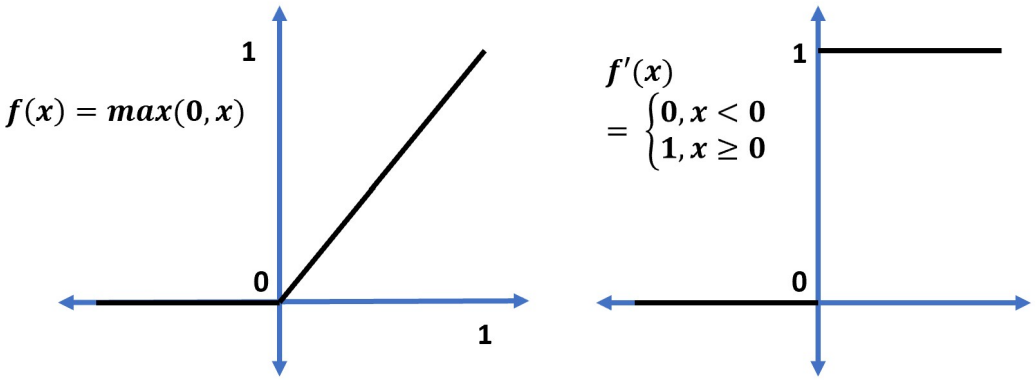


Figure 2.6: ReLU activation function and its derivative

gence rate. As shown in Figure 2.6, the function is unbounded in the positive side and consists of two linear parts with corresponding linear, constant derivative parts. Since the neurons are only activated if they receive a positive input, it is computationally simpler and more efficient. This function does not have the problem of vanishing gradients as well. However, the deactivated neurons corresponding to negative input values are dead neurons that hamper a proper mapping of negative values during training. This is referred to as the dying ReLU problem as the function outputs a constant zero value for inputs in the negative range. Also, since the function is not continuously differentiable, it works well with lower learning rates and without large negative bias values.

To deal with the dying ReLU problem, there are many improved variations of ReLU used commonly in many ANN applications. Figure 2.7 shows a few activation functions that have a non-zero curve defined for the negative range of values. Each of these functions also have their own limitations. For instance, the Leaky ReLU function has a smaller gradient in the negative side and would hence require more computation time. The Parametric ReLU function requires a good tuning and selection of the α parameter value. The Exponential Linear Unit (ELU) function, on the other hand, could lead to a exploding gradient problem. Hence, each activation function comes with its own shortcomings and can be chosen based on the performance requirements of a given ANN.

There are many other activation functions that show good performance in hidden layers like

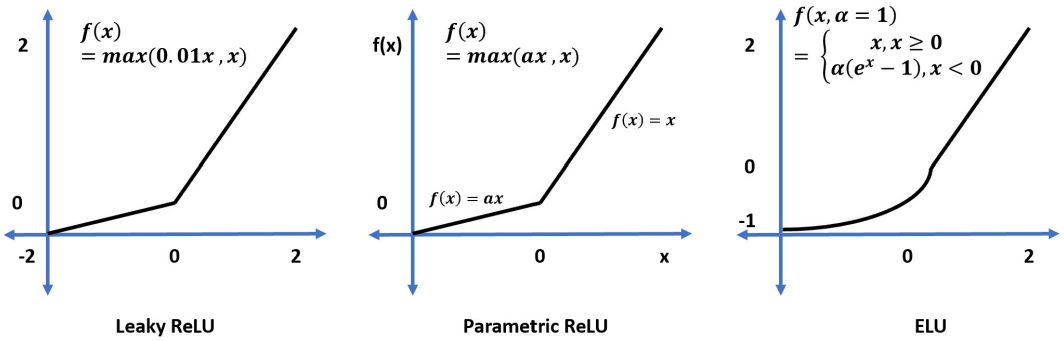


Figure 2.7: Improvements on ReLU to handle negative input values

the Gaussian Error Linear Unit, Scaled Exponential Linear Unit, the Swish function which are out of scope for this thesis. However, it is important to make an informed decision in choosing an activation function for the different layers of the ANN (see Table 2.1). Hidden layers make use of non-linear activation functions that can be chosen based on their prediction performance or convergence rates for a given problem. For instance, RNNs typically use sigmoid or tanh functions in their hidden layers. CNNs and feedforward ANNs like MLPs, on the other had, use ReLU as their activation function for the hidden layers. The output layers have an activation function depending on the nature of the prediction problem. Regression problems could make use of the linear activation function (or the softplus function for positive real-valued outputs) whereas the sigmoid function would be suitable for binary classification. Softmax is another prominent activation function used for multiclass classification problems.

Table 2.1: Choosing an activation function. Note: Typical hidden layers depend on ANN types. For instance, CNNs and MLPs use ReLU and its variations whereas tanh and sigmoid are common among RNNs

Prediction problem	Output layer	Hidden layer
Regression	Linear, Softplus	Depends on ANN type
Binary classification	Sigmoid	Depends on ANN type
Multiclass classification	Softmax	Depends on ANN type
Multilabel classification	Sigmoid	Depends on ANN type

With the right set of activation functions, weights and biases, the universal approximation theorem suggests that there exists a neural network to approximate any arbitrarily complex, non-linear, continuous function. This applies for any neural network having one hidden layer of any arbitrary size. Hence, ANNs are really good universal function approximators.

2.2.3. Learning and Optimization

An ANN with a defined structure and set of activation functions becomes capable of processing unseen information and making reasonable predictions only after it has undergone learning with relevant training data. In the beginning, after the structure and elements of an ANN have been defined, the weights are assigned randomly generated values. Similar to how the human brain learns based on the inputs it receives and changes its outputs accordingly, an ANN learns from the input sent through it and the corresponding processed output. This learning is reflected in the ANN as an updating of its weights between different nodes. While the input, hidden and output layers, and their activation functions remain fixed, an ANN learns and adapts to any

complex problem through modification of its weights.

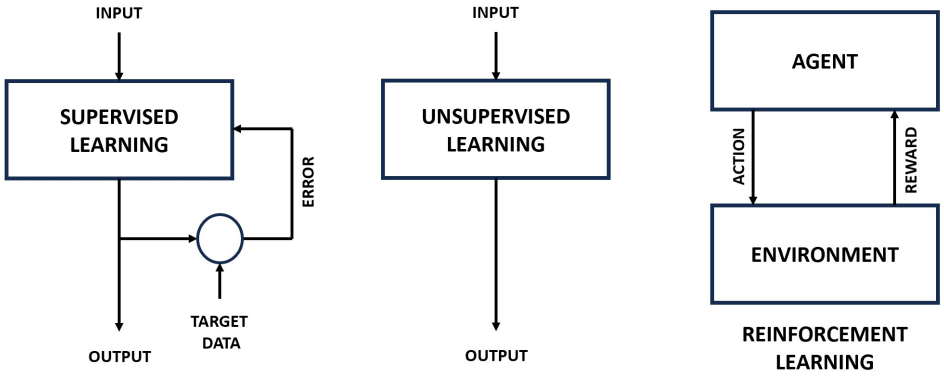


Figure 2.8: The three basic learning models

Learning in ANN can be broadly classified as supervised learning, unsupervised learning or reinforcement learning (RL) - the three primary learning paradigms for an ML tool. Supervised learning is dependent on a set of target data that is used for comparison with the output of the ANN to determine further updates in the network. In other words, it could be said that the learning is guided/supervised by this available set of target data. Unsupervised learning, on the other hand, is independent of any kind of target data that could act as a feedback for the ANN. Instead, the model tries to make sense of input patterns and the hidden information present in unlabeled data, and learn on its own. [31] reviews learning and training methods for unsupervised learning in ANNs. Reinforcement learning, on the other hand, learns from experience or interaction with a given environment. RL tries to learn the optimal behaviour of agents in an environment by performing actions sequentially and acting based on the resulting outcomes/rewards. The goal of an RL problem is to maximise a numerical reward. Since data is obtained through interaction, this type of learning involves data-sets that change dynamically with time. Figure 2.9 summarises the differences between these three learning methods in a simplified manner.

The thesis, henceforth, focuses only on supervised learning using ANNs. Security assessment in power systems have certain quantities whose futuristic prediction adds value to conventional assessment methods. These quantities are, hence, the target values for supervised learning-based prediction algorithms. The updating of weights based on the error between the output of an ANN and the target value could be done using different training algorithms. The most common way to process this error is to propagate it backwards through the ANN towards the input layer. This is referred to as back-propagation. The optimization of parameters (or weights and bias values) based on the back-propagated error is usually carried out using Gradient Descent - an optimization algorithm.

Considering a regression prediction problem, the objective of an optimization algorithm is to obtain an optimal fit of the prediction with respect to the target values using an optimal set of parameters. For example, say the error is computed using a loss function L defined as the mean square error (MSE) between the output and target values:

$$L(\theta) = \frac{1}{N} * \sum_{n=1}^{N} (y_n - y_{pred,n})^2 \tag{2.1}$$

N is the total number of training samples (target values available for training), y is the target value and y_{pred} is the predicted output from the ANN. Obtaining an optimal fit, hence, depends on minimising the loss function L through changes in parameters (referred by θ). Thus, the optimal parameters (θ^*) are:

$$\theta^* = \arg\min_{\theta} L(\theta) \tag{2.2}$$

The iterative updating of parameters could be written as:

$$\theta_{k+1} = \theta_k - \alpha.\frac{\partial L(\theta)}{\partial \theta} \tag{2.3}$$

k refers to a learning step, α is the learning rate and the partial derivative of L with respect to θ is the gradient. The parameters are, hence, adjusted in a direction opposite to the gradient which explains why it is called gradient descent. If the ANN can be expressed as a function $f_\theta(x)$, then the output y_{pred} for the n^{th} training input sample (x_n) can be written as:

$$y_{pred,n} = f_{\theta_k}(x_n) \tag{2.4}$$

The gradient term in Equation 2.3 can be expanded as:

$$\frac{\partial L(\theta)}{\partial \theta} = -\frac{2}{N} * \sum_{n=1}^{N} (y_n - f_{\theta_k}(x_n))^2 . \frac{\mathrm{d} f_{\theta_k}(x_n)}{\mathrm{d}\theta_k} \tag{2.5}$$

The derivative term in Equation 2.5 represents the sensitivity of the output prediction with respect to changes in the parameter values. The difference between the output prediction and the target value is the error computed for a given input training sample. If W_l is the set of weights between layer (l-1) and layer l, a_l is the activation function in layer l, then the function $f_\theta(x)$ can be written as:

$$f_\theta(x) = a_L(W_L.a_{L-1}(W_{L-1}.a_{L-2}(..(W_1.x)..) \tag{2.6}$$

In Equation 2.3, computing the loss gradient for each weight across the ANN is inefficient. Instead, computing gradient for the weighted inputs in each layer from the last layer towards the first helps avoid unnecessary and repetitive calculations. If z_l is the weighted sum at layer l, A_{l-1} is the activation value at layer l, then:

$$z_l = W_l.A_{l-1} \tag{2.7}$$

$$A_l = a_l(z_l) \tag{2.8}$$

Then the gradient term of loss for each individual weight ($w_{i,j}$ is the weight between j^{th} node in previous layer to i_{th} node in the next layer) can be computed using chain rule through back-propagation:

$$\frac{\partial L(\theta)}{\partial {w_{i,j}}^l} = \frac{\partial L(\theta)}{\partial {z_i}^l} . \frac{\partial {z_i}^l}{\partial {w_{i,j}}^l} \tag{2.9}$$

Through back-propagation, error is propagated efficiently backwards through layers where at each layer, gradients of weights are computed using corresponding derivatives in activation functions and matrix multiplication of weight values. In computing software, back-propagation is carried out through reverse mode automatic differentiation, wherein the chain rule is used to

compute partial derivatives backwards across layers. Vanishing gradients and exploding gradients are two problems that could occur during back-propagation. The former happens when the gradients diminish a lot across layers during back-propagation and the latter happens when the gradients become too large. This leads to too small gradients to effectively update weights in the front layers or divergence to extreme values due to very large gradients, respectively.

In real-world applications, several derivations of GD are used for optimization and learning. These include mini-batch gradient descent, stochastic gradient descent, momentum-based gradient descent, Nesterov-accelerated gradient, Adagrad and Adam. Each optimizer comes with different advantages and could be chosen based on requirements like convergence rates or better generalising capabilities.

2.2.4. Types of Neural Networks

Without elaborate explanations, some common types of neural networks and their applications are listed below. This is not an exhaustive list.

- **Feed-forward ANNs**: One of the oldest and simplest ANNs, these have a single hidden layer and there are no backward loops. Some applications include simple classification or image processing algorithms. ANNs with a similar structure but multiple hidden layers are the Multi-layer Perceptrons.
- **Radial Basis Function ANNs:** These networks are feed-forward ANNs that use radial basis functions as their activation functions, and can be used for universal function approximation applications with advantages of ease in designing and high learning speed.
- **Convolutional Neural Networks:** Widely used in image recognition and processing, CNNs use convolution layers in one or more hidden layers that can identify complex patterns or images by progressively learning from small portions of an image (like the brightness or the colour of a set of pixels) to more complex parts of the image.
- **Recurrent Neural Networks:** These networks have cyclic flow among nodes which allows for past outputs to be used to influence current inputs or decisions. These ANNs are prevalent in text-processing and speech recognition applications.
- **Long/Short Term Memory (LSTM):** These networks are a type of RNNs that come with a capability to learn long-term dependencies among sequential data. They find applications in various time-series data processing applications, for instance, in natural language processing applications.
- **Residual Networks (ResNet):** A relatively new type of ANN, these networks allow for building of deeper neural networks using skip connections that connect outputs from earlier layers to the outputs of other stacked layers. They help in solving vanishing gradients (for instance, a problem in case of large CNNs) and achieve higher accuracy in deep ANNs.

2.3. Neural Ordinary Differential Equations

Neural Ordinary Differential Equations (NODE) are a rather recently introduced type of neural networks, discussed extensively in [32]. To understand the working of NODE, it helps to explain about the residual blocks present in ResNets. Certain ANN models like ResNets use a discrete set of transformations for their hidden states (see Equation 2.11). For instance, the skip connections of a ResNet mentioned in subsection 2.2.4 could be represented as shown in

Figure 2.9. If the transformation applied to data x_t across the layers is represented as F(x) and θ_t is the parameters of the layers, then x_{t+1} can be expressed as:

$$x_{t+1} = x_t + F(x_t, \theta_t) \tag{2.10}$$

This can be seen as a discrete transformation of hidden states from h_t to h_{t+1}:

$$h_{t+1} = h_t + F(h_t, \theta_t) \tag{2.11}$$

Ordinary Differential Equations (ODEs) refer to one or more functions dependent on a single

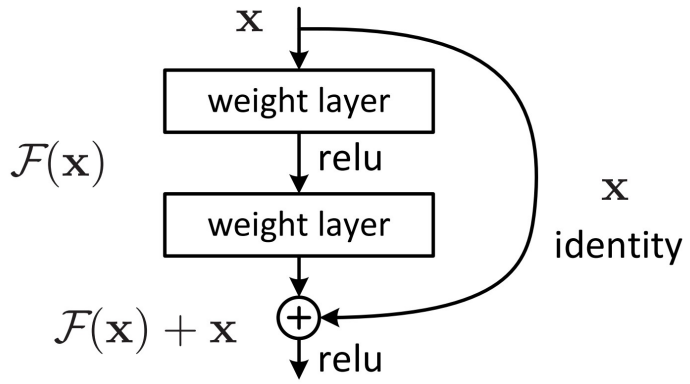


Figure 2.9: A building block in a ResNet, figure retrieved from [33]

independent variable and its derivatives. A simple, first order ODE can be defined as:

$$\frac{\partial x(t)}{\partial t} = f(x(t)) \tag{2.12}$$

Using Euler discretization, it is possible to approximate an unknown continuous curve by solving ODEs for an initial value problem (IVP). So, given an initial value $x_0 = x(t_0)$ at time t_0, it is possible to take small tangential steps starting from the initial value to approximate the actual curve. If one step size is h and time t_n after n steps is $(t_0 + nh)$, then one step in the Euler method can be written as:

$$x(t_{n+1}) = x(t_n) + h.f(x(t_n), t_n) \tag{2.13}$$

When the time steps are taken to be infinitesimally small, it is possible to obtain a continuous curve. Similarly, when the number of hidden layers are increased and smaller steps are taken, neural networks can be used to represent ODEs for the continuous dynamics of the hidden states (see Equation 2.11) as:

$$\frac{\mathrm{d}h(t)}{\mathrm{d}t} = f(h(t), t, \theta) \tag{2.14}$$

Hence, if h(0) is the input layer of the ANN, then h(T) can be defined as the output layer at a desired time T and a black-box ODE solver can be used to compute the hidden state dynamics defined by function f in Equation 2.14. Differences in hidden state dynamics between ResNets and ODE networks are depicted in Figure 2.10. However, in order to train an ODE network, back-propagating through reverse-mode differentiation in the ODE solver is difficult. Hence, the adjoint sensitivity method is used in gradient computations for ODE solvers.

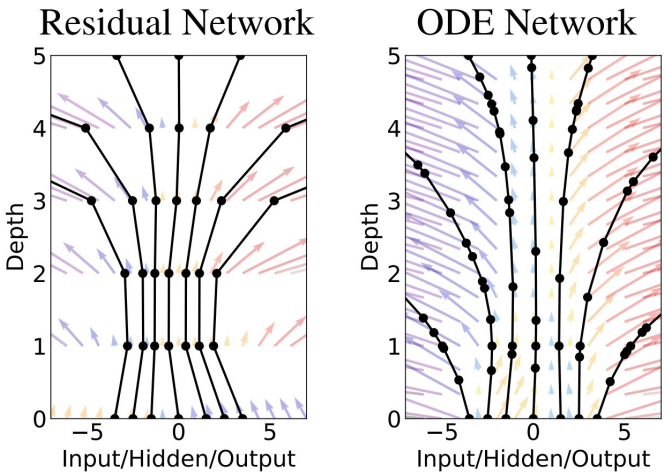


Figure 2.10: Comparison of hidden state dynamics in ResNets and ODE networks, figure retrieved from [32]

2.3.1. Adjoint Sensitivity Method

For optimization of the loss function and computing gradients across the black-box ODE solver, ODE networks use the adjoint sensitivity method. The optimization method explained below has been presented as a detailed algorithm with derivations in [32]. Say, the loss function for the ODE network is a function of its ODE solver's output $z(t_1)$, defined as:

$$L(z(t_1)) = L(z(t_0) + \int_{t_0}^{t_1} f(z(t), t, \theta)\, dt) \tag{2.15}$$

Hence, the inputs to the ODE solver are $z(t_0)$, t_0, t_1, θ and f. Similar to optimisation of the loss function using GD in subsection 2.2.3, the gradient of L with respect to θ needs to be computed. The gradient of L is dependent on the hidden state z(t) at each instant t, and this dependence is denoted by a value called adjoint a(t):

$$a(t) = \frac{\partial L}{\partial z(t)} \tag{2.16}$$

The adjoint state is, thus, indicative of the sensitivity of the loss function with respect to z(t). The dynamics of the adjoint state is defined in [34] as an ODE:

$$\frac{\mathrm{d}a(t)}{\mathrm{d}t} = -a(t)^T(\frac{\partial f(z(t), t, \theta)}{\partial z}) \tag{2.17}$$

Starting from $a(t_1)$, the ODE solver moves backwards towards to compute $a(t_0)$, as shown in Figure 2.11. Since this requires the knowledge of z(t) over the period t_0 to t_1, z(t) is also computed backwards from $z(t_1)$ to $z(t_0)$. Then the gradient of L with respect to θ is now dependent on both a(t) and z(t):

$$\frac{\mathrm{d}L}{\mathrm{d}\theta} = \int_{t_1}^{t_0} a(t)^T(\frac{\partial f(z(t), t, \theta)}{\partial \theta})\, dt \tag{2.18}$$

Thus, in reverse-mode differentiation for the ODE solution, both the hidden state z(t) and the sensitivity of L with respect to that state are considered. Whenever there is a direct dependence of L on the hidden state at an observation point (indicated by the dots in Figure 2.11), the adjoint state is modified in the direction of the gradient of loss at that observation point. Hence, the inputs and outputs of the reverse-mode differentiation algorithm [32] using the adjoint sensitivity method are given in Table 2.3.

Table 2.2: Inputs and outputs for the reverse-mode differentiation algorithm [32] using adjoint states

	Parameters in reverse-mode differentiation algorithm
Inputs	t_0, t_1, $z(t_1)$, θ, Gradient of L with respect to $z(t_1)$
Outputs	Gradient of L with respect to $z(t_0)$, Gradient of L with respect to θ

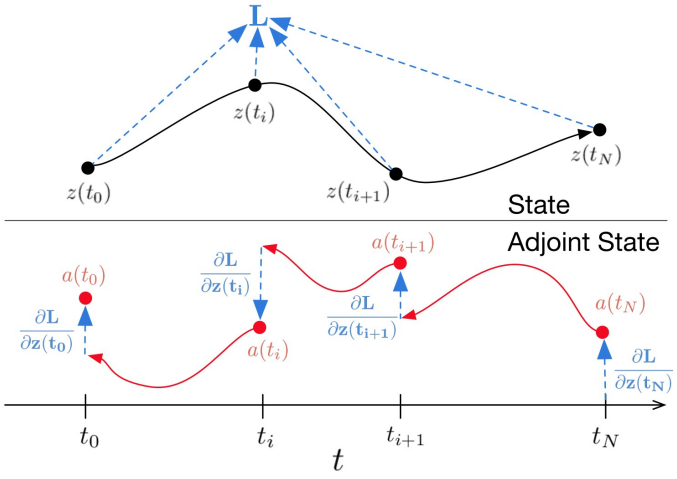


Figure 2.11: Reverse-mode differentiation using adjoint states, figure retrieved from [32]

2.3.2. DiffEqFlux.jl - A Julia Library

The DiffEqFlux.jl package in Julia programming language allows for easy integration of differential equation methods in neural network models to support various SciML applications. For instance, this package facilitates the usage of DifferentialEquations.jl package to implement suitable differential equation solvers for applications in ML libraries with neural network frameworks like Flux.jl and Lux.jl. With respect to NODEs, this package allows neural networks to make use of differential methods for optimization and adjoint sensitivity methods. A good example of using DiffEqFlux.lj to approximate differential equations using NODEs has been presented in [34].

An ODE problem can be solved (using *solve* function) in Julia using the DifferentialEquations.jl package when the corresponding ODE function (f), initial condition (u_0), time-span (tspan) and the function parameters (p) are specified:

$$prob = ODEProblem(f, u_0, tspan, p)$$
$$sol = solve(prob, ode\_algorithm)$$

In case of NODEs, a neural network is used to approximate the ODE function (indicated as f in Equation 2.14). Using the Lux.jl library, a neural network with say, a hidden layer of 20 neurons with ReLU activation to predict 3 states in a non-linear system, could be defined as:

$$dudt = Lux.Chain(x- > x, Lux.Dense(3, 20, Lux.relu), Lux.Dense(20, 3))$$

Given a time-span (tspan), an ODE solver algorithm (for example, Tsit5()) and a saveat value (for example, 0.1 time unit) to specify the time points when the solver has to save the solutions, then the NODE layer can be defined using the *NeuralODE* function:

$$node = NeuralODE(dudt, tspan, Tsit5(), saveat = 0.1)$$

In order to back-propagate using the adjoint sensitivity method, the ODE integrators need to be reversible to allow reverse-mode differentiation. DiffEqFlux.jl makes it easy for the user to switch between these different gradient methods using its three functions: diffeq_fd, diffeq_rd and diffeq_adjoint. Hence, the package allows switching between different modes of differentiation by a change of one character.

2.3.3. Advantages of NODEs

Some benefits of using NODEs for regression prediction problems are listed below.

- NODEs have a constant memory cost. Unlike other prevalent ANNs used for regression like RNNs, no intermediate quantities are stored in the network during a forward pass. Instead all inputs are accounted for through the gradient computation method discussed in subsection 2.3.1. This ensures high memory efficiency, especially in building deep neural network models.
- The ODE solvers used for learning and predicting any unknown curve vary greatly in terms of their ability to accurately approximate the true curve. With the range of ODE solvers that are available today, it is possible to trade-off between achieving high levels of accuracy and high costs of computing based on the complexity or speed requirements of a problem. NODEs, thus, allow for adaptability in terms of computation.
- It is suggested in [32] that the number of network parameters required could be reduced because the hidden state dynamics of NODE layers allow closer layers to be tied together automatically. This results in higher parameter efficiency.
- NODEs can support continuous time-series data which arrive at arbitrary time intervals. With irregularly sampled data, conventional RNNs face difficulty in handling arbitrary time gaps between observations. With NODEs, it is possible to define a unique latent trajectory given any initial latent state. A generative, latent variable-based approach for modelling time-series has also been presented in [32].

The benefits of using NODE are, thus, multi-fold. There are good examples of successful implementation of NODEs in dynamic system modelling and parameter estimation problems [35, 36] in the literature. However, there are also many papers addressing some shortcomings of the NODE models and mathematical solutions to mitigate them using augmenting methodologies. Some good examples include [37] that proposes more expressive models which can augment the ODE solving space to provide added dimensions to learn more complex problems, [38] which proposes a more accurate gradient estimation method for NODEs called the Adaptive Checkpoint Adjoint (ACA) method and [39] that proposes NODEs with time-varying weights to achieve enhanced expressiveness in image processing applications.

2.4. Data Handling and Performance Evaluation

The key elements in an ML algorithm include the training input data in the form of features (or) multiple independent quantities corresponding to a range of observation points, the output quantities at these observation points which are expected to be predicted, a loss function that is representative of the accuracy of the model prediction, an optimization method to minimise the loss function and a set of unseen data reserved for validation and testing of the trained model. All these elements are introduced into an ML workflow at suitable points. A simple ML workflow with all the basic blocks of a prediction algorithm is shown in Figure 2.12.

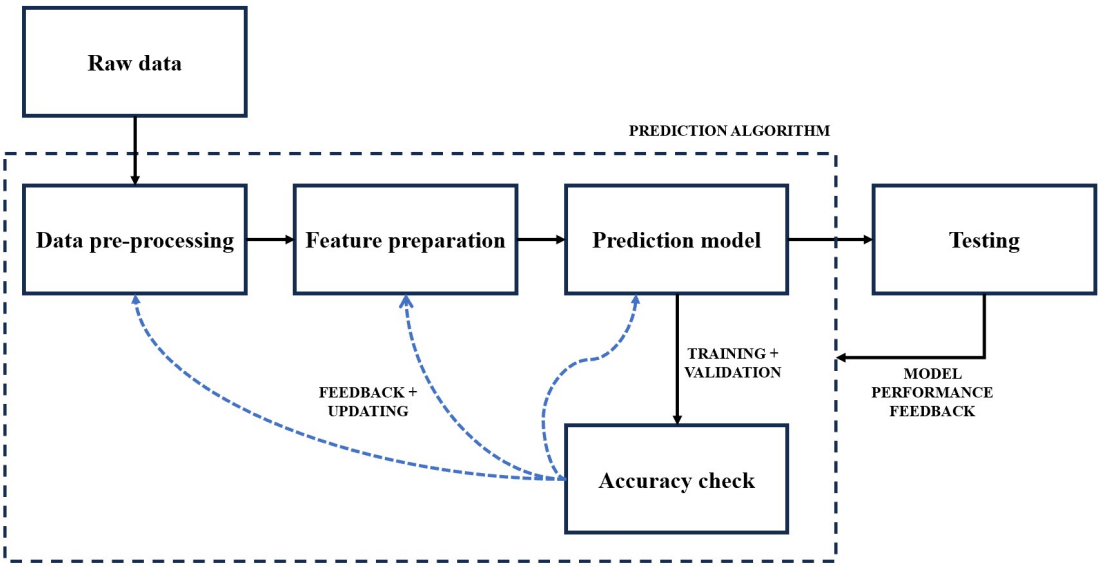


Figure 2.12: A sample workflow in an ML-based prediction algorithm

Raw available data is generally pre-processed before being used in a prediction model. In case of security assessment in power systems, a relevant example of raw data are PMU measurements from power grids. For a given time-span, these measurements could have missing values or noisy readings or inconsistent outlier values. Processing this data to have a continuous, realistic time-series input data is often a necessity for a prediction model. Once processed, data transformations could be done to obtain more representative derived quantities. A simple example is deriving the power angle and active power values from the available voltage and current PMU measurements. In prediction problems with large number of available features, feature reduction techniques are carried out to address possible redundancy in data and improve computational efficiency of the model. In security assessment, it is often possible to logically decide if certain features are not important. Once selected, these set of features could be reduced further (in terms of dimensions, if required) using various feature extraction or reduction techniques. Another important aspect in data-preprocessing is scaling of features. Having differently scaled features (for example, voltage measured in the range of a few thousand volts and frequency measured in the range of 49 to 51 Hz) would be misleading to prediction models that are largely dependent on the numeric values of the features and related inter-dependencies. In order to effectively capture the variations in a feature and understand its correlation with other features or the target output values, scaling transformations like min-max normalization or standard normalization are carried out for numerical data.

The prediction performance after training a model depends on its complexity and the corresponding ability to be applied to unseen data accurately. The two extremes of model complexity are when the trained model is under-fitting or over-fitting for the prediction problem (see Figure 2.13). Under-fitting models give rise to bias errors due to high simplifications in approximating the target function. Over-fitting gives rise to variance errors when the model is presented with new input data to predict from. Hence, a reasonable trade-off between bias and variance errors results in a prediction model that is more representative of the underlying dynamic phenomena and capable of predicting from unseen data. In order to evaluate the performance error with respect to model complexity, a small fraction of available, unseen data

is reserved for validation immediately after training. The resulting validation error from the prediction model helps in finding a viable level of model complexity, as shown in Figure 2.14. So, after a few prediction models have been optimized through training, validation could help in choosing the more suitable model.

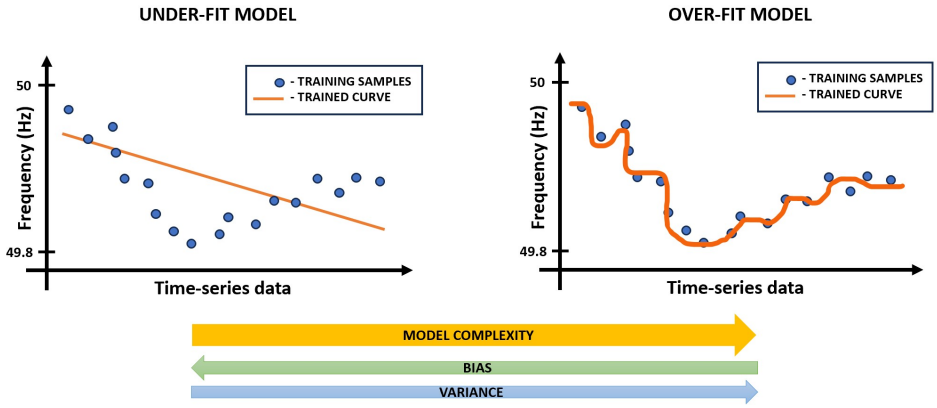


Figure 2.13: Impact of model complexity on curve-fitting

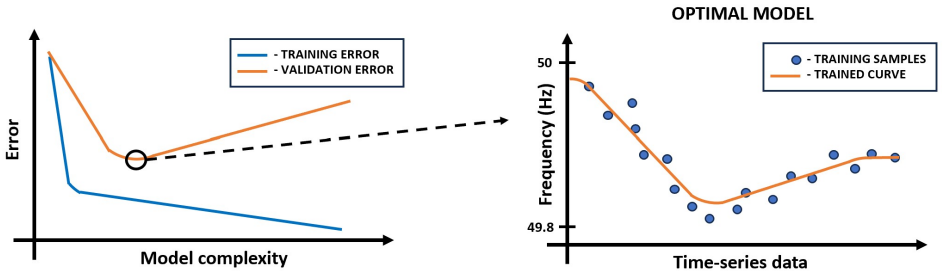


Figure 2.14: A simple validation approach to choose an optimal level of model complexity

While the training data is used to optimize a set of prediction models, validation data could be used to choose an optimal model and test data could be used to finally evaluate the performance of the model. Some common ways to optimize models include modifying the loss function to enhance the training speed and be more representative of the required model performance, tuning parameters of the ANN model (like the number of layers, activation functions etc.), trying different optimizers and tuning corresponding parameters like learning rates, and choosing a suitable set of evaluation metrics for validation and testing. MSE, R^2 score and root mean square error (RMSE) are some prevalent evaluation metrics for regression problems. Accuracy, precision, recall, F_1 score, confusion matrix and the area under the curve in a receiver operator characteristic (AUC - ROC) curve are some common evaluation metrics for classification problems. The relevant evaluation metrics used in this thesis are elaborated upon in chapter 3 and chapter 4.

2.5. Frequency Stability - Continental Europe

When there is an imbalance between generation and demand, the ability of a power system to maintain a steady frequency value could be referred to as frequency stability [3]. After any

severe disturbance, if the system is unable to maintain or restore its frequency to nominal values, tripping of generating units or loads could take place. European Network of Transmission System Operators for Electricity (ENTSO-E) suggests the range of 49.8 Hz to 50.2 Hz to fall under the ordinary operation range. The system becomes prone to severe outages and also, possible blackout situations when the frequency deviations go beyond the acceptable range for stable operation.

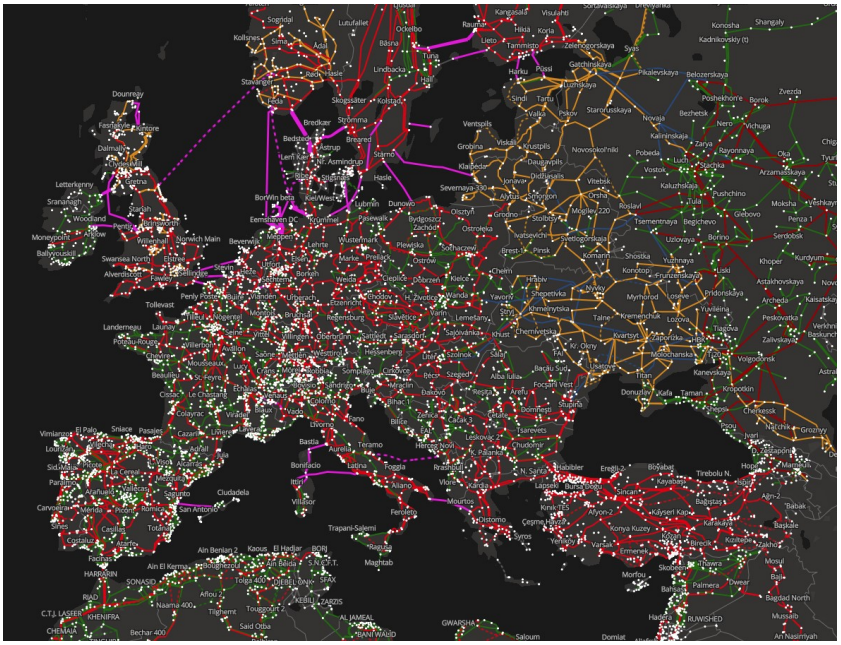


Figure 2.15: A snippet of transmission lines (220 kV or higher) across the synchronous grid of Continental Europe, part of the ENTSO-E. More information about the map is available at the official ENTSO-E website.

There are various wide-area synchronous grids across the world that operate at a specified utility frequency and have interconnections spanning across large regions, or also across many countries. There are about 26 countries synchronously interconnected in the case of the synchronous grid of Continental Europe (see Figure 2.15). With a fixed nominal frequency setpoint of 50 Hz, it is possible for any major disturbance in the Continental Europe gird to have repercussions at any other location across the entire synchronous grid. The impact of instability due to frequency disturbances and the subsequent control or stability restoration measures taken in a power system (with an emphasis on the Continental Europe grid, currently a part of the ENTSO-E) are discussed below.

2.5.1. Power Imbalance and Impact on System Stability

In any synchronous grid, the frequency is directly representative of the balance between generation and demand, and needs to be within operational security limits at all points. In case of Continental Europe, when the frequency breaches the 47.5 Hz (under-frequency) or 51.5 Hz (over-frequency) limit, all generating units and connected devices are expected to automatically disconnect. Apart from the risk of losing synchronism across the grid, the impact of frequency deviations beyond acceptable thresholds could range from poor load performance and overloaded transmission lines to protection failures, large scale load-shedding and loss of generating units.

The dynamics in synchronous machines during power imbalances can be described by the swing equation:

$$\frac{2H}{w_s}\frac{\mathrm{d}^2\delta}{\mathrm{d}t^2} = P_a = P_m - P_e \tag{2.19}$$

H is the inertial constant, w_s is the synchronous speed of the rotor, δ is the load angle (also referred to as the power angle), P_a is the accelerating power, P_m is the mechanical power and P_e is the electrical power. In steady state, the accelerating power is zero as the the mechanical torque and the electromagnetic torque are in balance, and the machine runs at synchronous speed. Since the frequency and the speed of the synchronous generators are directly proportional, any change in the speed of the machine reflects in the frequency response of the system. During a power imbalance, when P_m is not equal to P_e, an accelerating or decelerating torque exists and leads to an increase or decrease in speed (indicated by the derivative term in Equation 2.19) and frequency, respectively. The RoCoF term could then be expressed as:

$$RoCoF = \frac{\mathrm{d}f}{\mathrm{d}t} = \frac{\Delta P f_s}{H} \tag{2.20}$$

f_s is the nominal frequency and the change in P refers to the difference between P_m and P_e.

The causes of power imbalance could range from excessive load demand or loss of generating units (leading to a drop in frequency) to decreasing demand levels (leading to a rise in frequency) or disconnection of interconnected areas in a power network. Most of the severe imbalances are generally due to unforeseen changes in large-scale generation or consumption. Over the last few years, ENTSO-E has also noted significant frequency deviations in the Continental Europe grid that are deterministic in nature. Occurring at similar times in an expected fashion, these deviations could be attributed to the influence of market rules on generating units. The step-wise changes in generation and a continuous demand curve, hence, lead to short duration of imbalances at fixed times. [3] classifies frequency instability further as short-term and long-term phenomenon. Some examples of frequency stability-related problems and their possible impact on the power system and its components are given in Table 2.3.

Table 2.3: Examples of frequency instability problems and their possible impact on power systems

Frequency instability problems	Possible impact	Nature
Significant changes in voltage	Tripping of protection relays in generator, for instance, volts/Hertz relays	Short-term
Under-frequency load shedding (UFLS)	Significant loss of load and associated socio-economic costs	Short-term
Poor equipment response and control - Boilers, reactors, voltage regulators etc.	Inefficient stability restoration	Long-term
System splitting in interconnected networks	Insufficient generation or UFLS schemes in islanded systems; blackout in extreme cases	Short-term

Irrespective of the cause or nature of instability, frequency deviations are expected to have increasing amplitudes or nadirs due to low inertia levels in RES-penetrated power systems. With automatic generation control (AGC) in conventional power systems with synchronous

generation, resistance against speed changes from inertia of large rotating masses in generators is critical in providing valuable extra time for other stability controls to start acting. Since variable energy sources like wind power lack rotational inertia, new frequency control measures are required to address the reduction in inertia levels.

2.5.2. Control and Restoration Measures

Automatic Generation Control is employed in power systems with synchronous generators to maintain the power balance, the system frequency and also, net interchange in power when there are multiple control areas that are connected by tie-lines. Three steps of control, namely, primary, secondary and tertiary control are used to restore a system back to its nominal state. The impact of each of the three controls could be seen in a typical frequency response curve after a disturbance, as shown in Figure 1.4 and Figure 2.16. Frequency control responses and their corresponding time-scales, however, may vary among different nations [40]. Figure 2.16 shows typical time-scales for each tier of frequency control.

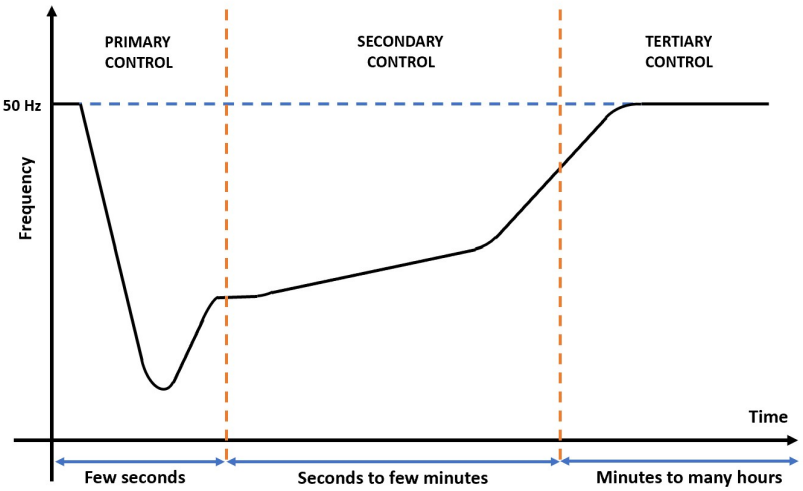


Figure 2.16: Tiers of frequency control and corresponding time-scales

Primary control aims to restore the generation-demand balance and bring the frequency to a stationary, stable value. This is implemented through droop control using a speed governor in the generator turbines. The speed-governing characteristic of a generator is a plot between its frequency (representative of speed) f and power output P, where the slope -R is the droop constant. It represents the sensitivity of system frequency to changes in output power or vice versa.

$$K_T = -\frac{\Delta P}{\Delta f} = \frac{1}{R} = \frac{1}{R_u}\frac{P_R}{f_R} \tag{2.21}$$

K_T is the turbine constant (in MW/Hz) defined for each turbine, R_u is the per-unit droop used in speed regulation, P_R is the rated power output and f_R is the rated frequency. When the power balance is restored by say, changing the amount of fuel flowing in to the generator, the system is brought to a new steady state value of frequency defined by the droop value. When there are N generating units operating in synchronism, then:

$$\Delta f = -\frac{\Delta P}{\sum_{n=1}^{N} K_{T_n}} \tag{2.22}$$

Hence, load could be shared among multiple generating units based on their turbine constants. When the rated frequency and droop are common for different generating units operating in parallel, load sharing depends on individual power ratings of the generators.

In secondary control, available secondary control reserves are activated and active power set-points of the generators or controllable loads are modified to restore the frequency back to its nominal levels. In case of multiple control areas (as typical in large synchronous grids), the tie-line power flows also have to be restored to ensure a stable system frequency. This tier of control is also referred to as load-frequency control (LFC). AGC in LFC is dependent on Area Control Error (ACE) defined for each control area.

$$ACE_{area} = \Delta P_{tie-line,area} + K_{area}\Delta f \tag{2.23}$$

K_{area} is the frequency bias setting for a given control area defined by:

$$K_{area} = -\frac{\Delta P_{area}}{\Delta f} \tag{2.24}$$

Change in power flow inside a control area is taken to be the sum of both internal and tie-line power flow changes. So, ACE provides information about the power flow changes required in each control area to achieve LFC.

Tertiary control is implemented generally over a longer period of time where set-points of generators and controllable loads could be modified to ensure that the primary and secondary control reserves are restored, while also accounting for economic considerations. This type of control could be manual or automatic.

For the Continental Europe grid, ENTSO-E has a separate policy on “Load Frequency Control and Performance” that clearly defines the technical and operational requirements and framework for each level of frequency control.

To address reducing inertia levels in RES-penetrated systems, new additions to instantaneous frequency control like fast frequency response (FFR) and synthetic inertia have been introduced. Based on frequency deviations or the RoCoF measured, FFR schemes are activated almost instantly (in the time frame of 1-2 seconds) to provide rapid increase or decrease in active power for compensating the reduced inertia levels. Synthetic inertia, on the other hand, tries to mimic the kinetic energy released from a rotating mass by providing a resisting electrical torque proportional to the detected RoCoF [41]. With increasing RES penetration and inadequate methods to compensate for low inertia, sudden events like large-scale LoG or system split in large networks could lead to drastic frequency deviations more frequently. In situations where frequency thresholds are breached, defensive control methods like UFLS, over-frequency in-feed reduction are employed. When all available control methods fail to stabilise the frequency, cascading outages and/or a blackout could occur.

3

Predictive NODE Algorithm - Methodology

3.1. Introduction

Defining a NODE algorithm for frequency prediction is largely dependent on available input data, the selected power system model, desired prediction horizon, nature of frequency events, desired output quantities and relevant performance metrics. Figure 3.1 summarises the general workflow in a frequency predictive NODE algorithm. This chapter discusses adapting NODEs for frequency security assessment to achieve fast and reliable futuristic predictions that could provide timely, critical details regarding the (impending) status of a power system.

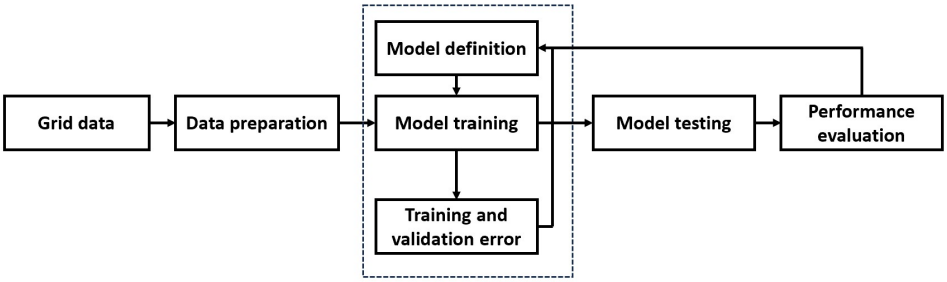


Figure 3.1: General workflow in predictive NODE algorithm

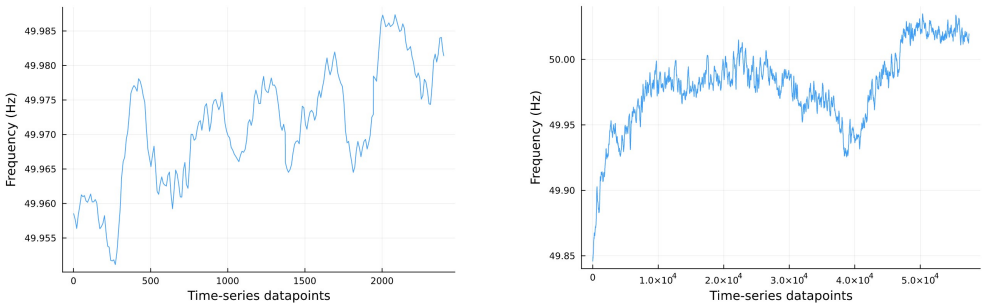
3.2. Data Preparation

Since drastic frequency events are sparse in reality, both real grid data and synthetically simulated data are crucial for training the prediction model. While the real grid data provides an insight into the real-time PMU measurements that present security assessment algorithms work with, the number of recorded frequency events corresponding to large LoG and major system split events are quite small in number. On the other hand, it is possible to simulate a range of frequency events for different system conditions with the help of modern power system simulation software like PowerFactory. This thesis attempts to define a prediction model for frequency events while acknowledging the impact of training on synthetic data and the expected differences in performance when dealing with real-time PMU data. To do so, both types

of data are processed, trained upon and the resulting differences in prediction performance are noted.

3.2.1. PMU Measurements from the Netherlands Grid

PMU measurements of real-time grid data are available at three substations across the Dutch High Voltage grid. At each substation, quantities are measured at two locations. For a given timestamp, frequency, RoCoF, magnitudes and angles of each phase of voltage and current are measured. A few data-sets corresponding to small frequency deviations and two system split events in Continental Europe are available for processing on request from a Dutch Transmission System Operator (TSO). Results of processing two of these data-sets, one corresponding to normal operational scenario (say, scenario 1) and one corresponding to frequency restoration scenario (say, scenario 2) post-system split are presented in this thesis. These data-sets were chosen by checking if important data quantities were not missing for large intervals at all three substations across the entire available time-span. The interpolated frequency data from one of the six locations (say, Location 1, Substation or SS 1) for the two selected data-sets looks as shown in Figure 3.2.



(a) Normal operation scenario **(b)** Restoration post-system split scenario

Figure 3.2: Interpolated frequency data from Location 1, SS1 of the selected data-sets

The data processing of the PMU data to attain the final set of features is summarised in Figure 3.3.

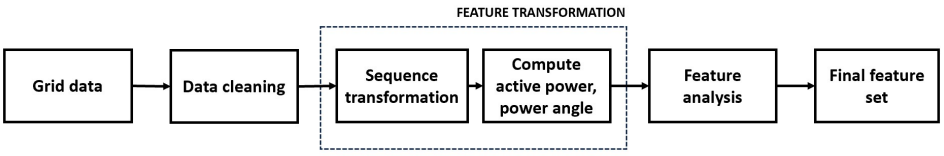


Figure 3.3: Data preparation with PMU data from the Netherlands grid

The missing entry statistics for both the data-sets is shown in Table 3.1. Since continuous intervals of missing data are restricted to a span of 0.3 seconds, linear interpolation was considered sufficient for data imputation. However, the effects of interpolation on the angle data are noted in the form of abnormal spikes during the interpolated intervals in voltage, current and other derived quantities during further processing. This is due to sharp turns arising from the measuring range used in PMU data i.e., -180 to 180 degrees (see Figure 3.4). To prevent errors in measurement during the sharp turns, missing data are removed instead. This does not lead to significant changes or abrupt shifts in the measured quantities as the missing intervals are small.

Table 3.1: Missing entry statistics for PMU data

	Scenario 1	Scenario 2
Number of locations with available data	6	4
Total number of missing entries (with respect to full time-span of data)	183 (out of 6300) i.e., 2.9%	1259 (out of 57319) i.e., 2.2%
Maximum duration of continuous missing entries	0.3s Total duration: 10.5 mins	0.3s Total duration: 95.5 mins

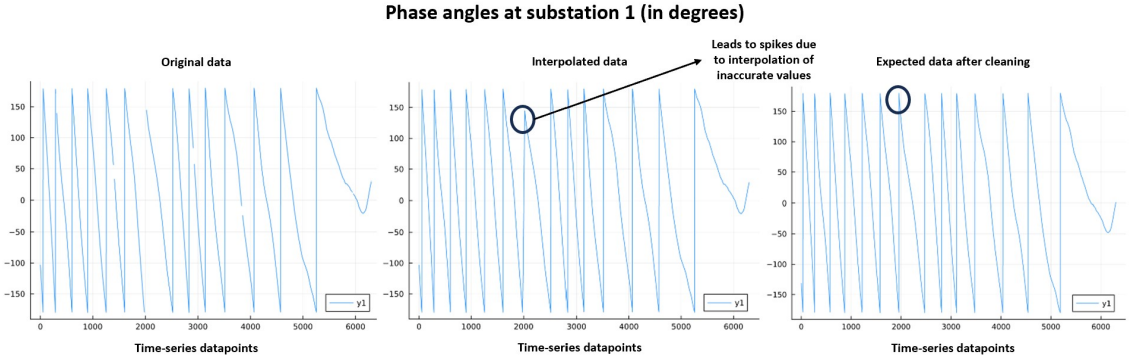


Figure 3.4: Interpolation of angle values from PMU data

Cleaned data consists of 14 different quantities with possibly redundant information for frequency prediction. To reduce the number of features without losing any valuable information, feature transformation is carried out to process three-phase quantities and compute useful derived quantities from the original data. After sequence transformation, the positive sequence current and voltage show a similar waveform as compared to all the individual phase quantities (see Figure 3.5), as expected. Hence, the magnitudes and angles data are transformed without any loss of information.

Since the three substations collecting PMU data are spread across the Netherlands, power flow at these locations could be a good indicator of changing generation or demand across the grid. Hence, the three phase active power and power angle values are computed at the available locations.

$$P_{3-phase} = V_1.I_1.cos(\theta_1) + V_2.I_2.cos(\theta_2) + V_3.I_3.cos(\theta_3) \tag{3.1}$$

V_1, V_2, V_3, I_1, I_2, I_3 are the phase voltages and currents. θ_1, θ_2 and θ_3 are the power angles. Power angles are found to be in the range of either 40 - 50 degrees or -320 to 320 degrees. They are adjusted to lie in similar numerical ranges as shown in Figure 3.6.

There are 30 features in total (frequency, positive-sequence voltage, positive-sequence current, active power and power angle from six PMU data collection locations) for scenario 1 and 20 features (same quantities from four PMU data collection locations) for scenario 2. With lower number of features and lower redundancy in data, it is more computationally efficient for the NODE algorithm to learn the inter-dependencies among features. To eliminate redundancy, features are visually checked for high correlation among each other. Firstly, the frequency measurements at all six locations almost fully coincide. This is expected as the Netherlands grid is but a small part of the synchronous grid of Continental Europe. Secondly, the two loca-

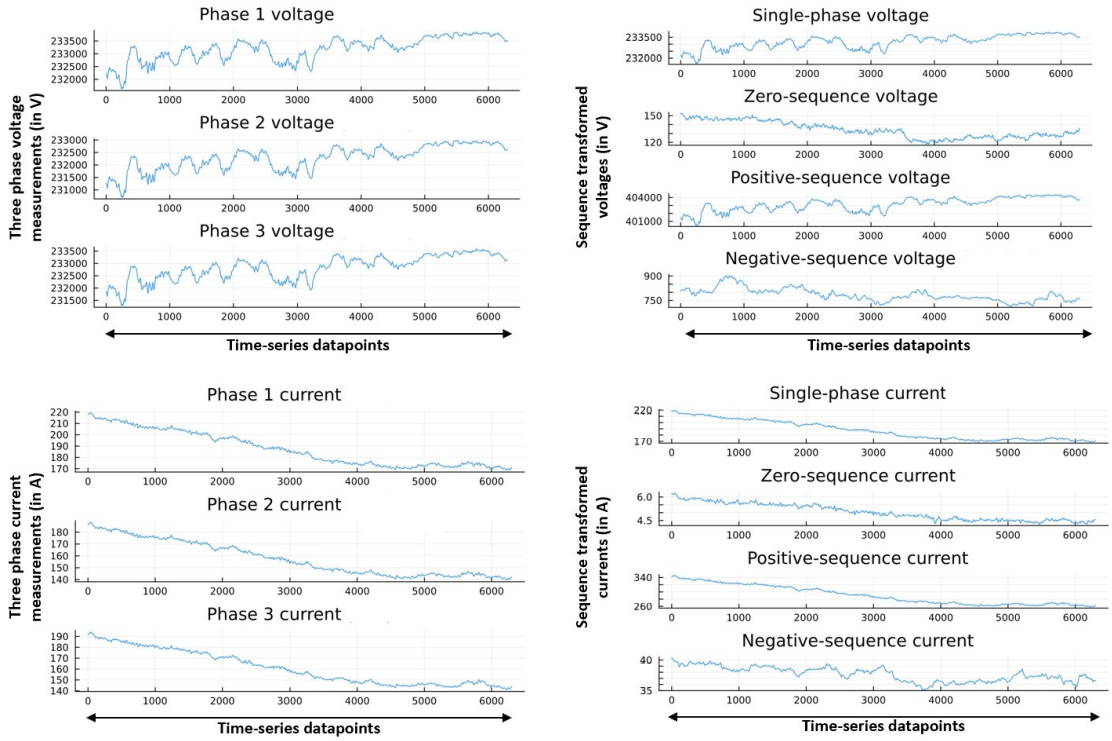


Figure 3.5: Sequence transformed quantities from original PMU data

tions in each of the 3 substations are at high proximity to each other. The measurements, thus, mostly coincide and reinforce the accuracy of PMU blocks in the same substation. Hence, all the features could be grouped (by taking their mean) based on their substations to effectively represent the measurements across the grid. With 3 substations, there are still 13 features (mean frequency from all 6 locations and other quantities from each substation) available after grouping. These quantities are shown in Figure 3.7.

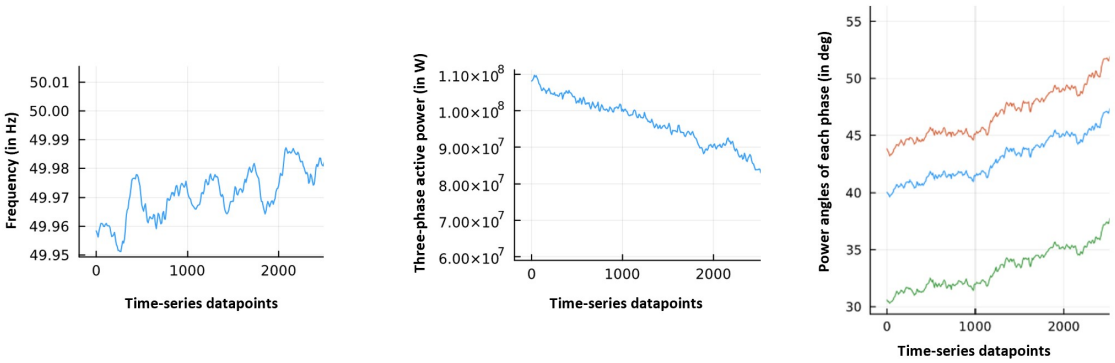


Figure 3.6: Active power and power angle values at Location 1, SS1 from scenario 1

Among the 13 features, 5 features are chosen for the final feature set: Mean frequency, mean voltage, active power at SS1, active power at SS2 and active power at SS3. Since current and active power values at a given substation show high correlation, active power values at 3 substations are chosen. Since the voltages at all three substations have a similar waveform, the mean of these values are taken. This is because NODEs work effectively with scaled data

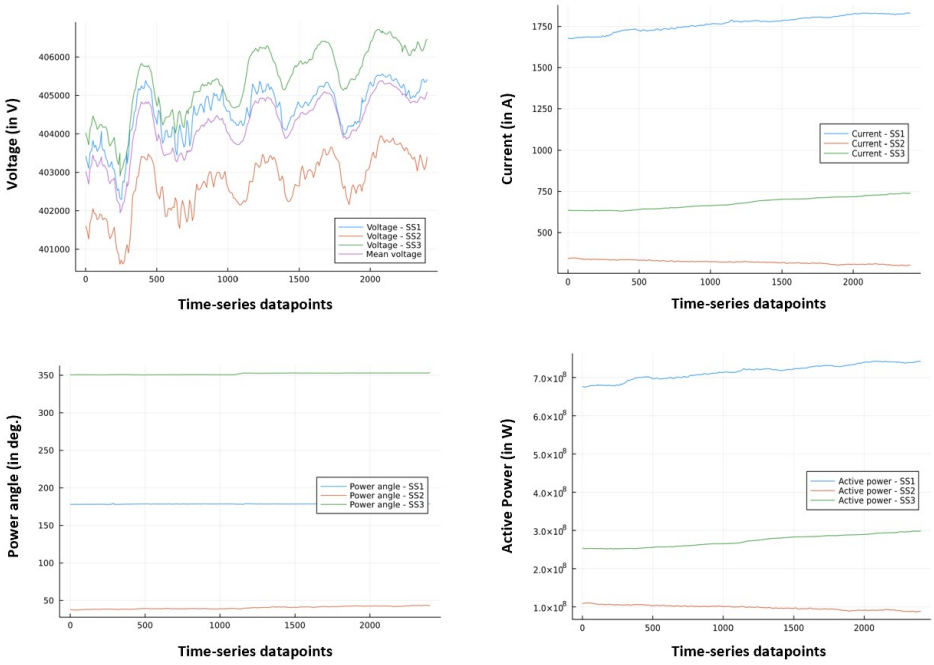


Figure 3.7: Time-series plots of grouped features

(often around the range of 0-1) and hence, the dynamics captured by a waveform are given higher precedence than difference observed in magnitudes. Since the power angle values are taken into consideration while computing active power, this information is not lost in the final feature set.

3.2.2. Synthetic Measurements from a Modified IEEE39 Bus System

For synthetic simulations of frequency events, simulation results from the other thesis project as part of the research project with Reddyn B. V. on low-frequency demand disconnect (LFDD) schemes are used. The events were introduced on a modified IEEE39 bus system (see Figure 3.8) simulated on PowerFactory software. In Figure 3.8, the orange circles indicate the synchronous generators with available current, active power and rotor angle data, and the blue markers indicate buses with available voltage and frequency data. Bus 39 is the slack bus and generator 01 is representative of the interconnection of the system to the larger grid. The red marker indicates the excessive loads simulated to create frequency events in the system by causing supply-demand imbalances of large magnitude. The green markers in Figure 3.8 indicate the RES generation points that are added or removed sequentially to change the inertia levels in the system. RES generation has been added to the IEEE39 system using the Western Electricity Coordinating Council (WECC) type 4B wind turbine connected at 7 different points in the system. To create different scenarios, each wind turbine sequentially replaces a synchronous generator in each scenario and causes a drop in the total system inertia. The power ratings of the added RES generation is matched to the ratings of the replaced synchronous generator.

To analyse the available data, a low RES penetrated system (with 9% RES generation out of the total supply) with a 300MW imbalance event due to increase in load is considered. The frequency, voltage, current, active power and rotor angles at the different measurement points

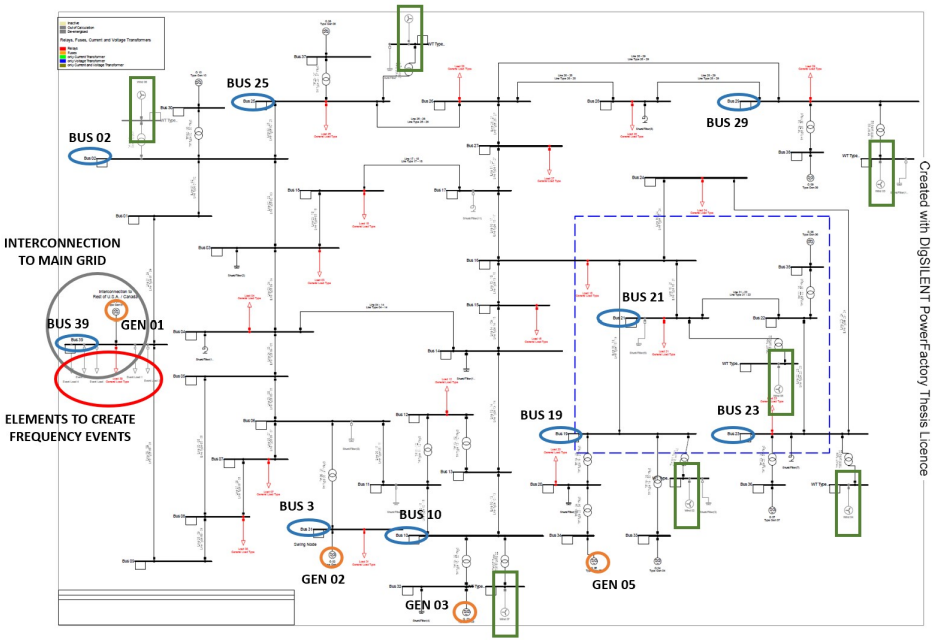


Figure 3.8: Modified IEEE39 grid diagram

on the system are plotted in Figure 3.9. As expected, the frequency at all the points coincide to ensure that the system is synchronous. Since Bus 39 and Generator 01 act as the point of interconnection with the main grid, the effect of internal changes in the IEEE39 system are reflected evidently at this location. While all internal bus voltages show a similar response during the frequency event, the voltage response at Bus 39 shows a higher correlation to the frequency response in the system. Similar to the PMU data from the Dutch High Voltage grid, current and active power responses are quite identical to each other for a given location. The rotor angle at Generator 02 is set as the reference angle and is, hence, always zero in all scenarios. While the rotor angles at all the synchronous generators recover quickly after the frequency event, the rotor angle measured at Generator 01 is again indicative of the frequency disturbance experienced by the entire IEEE39 system.

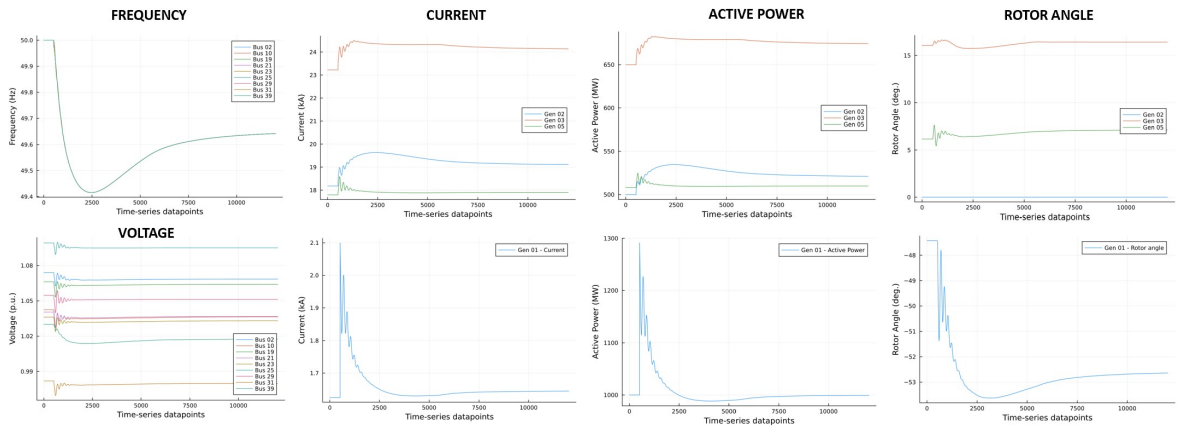


Figure 3.9: Available system data for 9% RES generation scenario, 300MW event simulation

Given all the available measurements from simulations, data preparation is relatively easier for synthetic data when compared to real-time PMU data. The observed dynamic response in

all quantities are noise-free and do not have any missing data or unexpected outliers. The more important aspect in this case is to reduce the feature size by grouping or transforming data without losing key information. The final set of features used for training the NODE model is mentioned in chapter 4. Since the number of scenarios and volume of available data are large in number (see Table 3.2), it is possible to implement offline training with synthetic data to show improvement in performance of the prediction model.

Table 3.2: Available simulated frequency disturbance scenarios

	RES generation levels	System inertia levels (Hsync)	Energy imbalance events
Available scenarios	0%, 9%, 20%, 33%, 43%, 53%, 66%	12.23s, 11.85s, 11.40s, 10.86s, 10.45s, 9.91s, 9.25s	300MW, 500MW, 700MW, 800MW, 1000MW

3.3. Predictive Model Definition

3.3.1. Data Initialisation

The initial set of data taken by a NODE algorithm are the input features, initial set of points in the system of ODEs that is to be solved, prediction time span, time steps for the ODE solver to save at and a random number generator to initialise the initial set of parameters in the neural network. Some important points to note while initialising data are:

- **Sampling rate of input features:** Sampling rate provides the ODE solver with the set of points using which the prediction model is trained. While having a high sampling rate might lead to higher computational times with no considerable change in model performance, having a very low sampling rate might not be enough to capture the dynamics in the non-linear system. Hence, a good selection of the sampling rate is required to ensure optimal model performance with minimal computational time (see Figure 3.10.

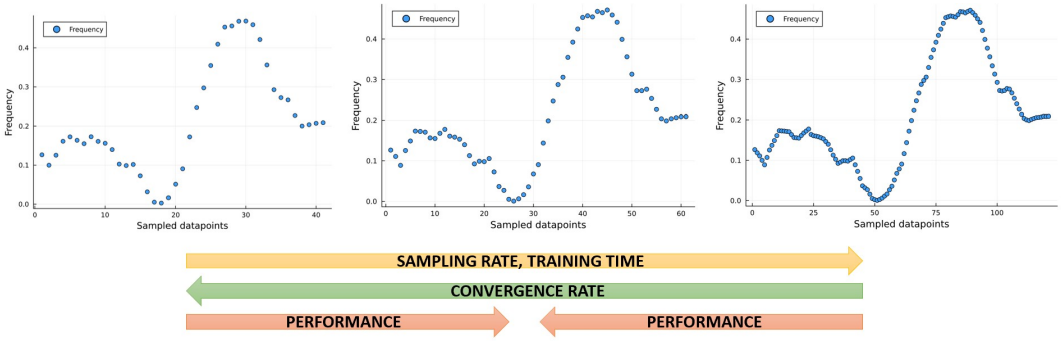


Figure 3.10: Impact of sampling rate on training

- **Seeding the random generator:** It is important to be able to reproduce model performance for a given set of parameters and model definition. Seeding ensures a good base workflow for comparing performance results between different model settings.
- **Prediction time span:** Choosing a time span for training and prediction depends on our desired end outputs. For instance, estimating nadir after a frequency event and predicting

the settling time post-nadir require different time scales of training depending on the typical duration of the respective phenomenon.

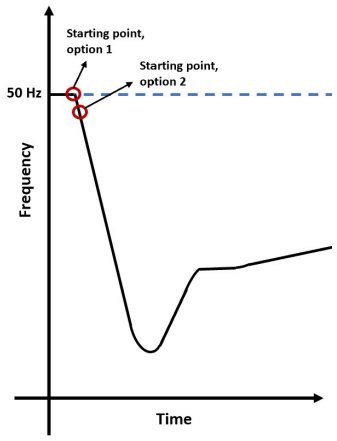


Figure 3.11: Choosing initial set of points for the ODE solver

- **Choosing the initial conditions (u0) for the ODE solver:** As mentioned in subsection 2.3.2, u0 is the initial condition of that states in a system given as an input to the ODE solver. Since this is one of the key information passed on to NODE block, it is effective to introduce differences in real-time status of the system frequency by passing the state values at a few points in time after the frequency event occurs (shown as option 2 in Figure 3.11). With option 1, an opportunity is lost to send in real-time information post-event in a system which is more useful in predicting the ensuing dynamics.

3.3.2. Neural Network Definition

A neural network definition includes a good choice of width and depth of hidden layers, and the activation function. The impact of changing activation functions on a simple, single hidden layer neural network that is training on 1 minute of grid data is shown in Figure 3.12. All data have been scaled to lie in the 0 -1 numerical range. While the model with Relu activation seems to have achieved a decent fit, its test loss with unseen data was recorded at the scale of 10^{27}, compared to test loss in the range of 20 - 30 for sigmoid and tanh activated models (see Table 3.3). Among sigmoid and tanh activation functions, sigmoid activation displays a higher level of adaptibility to non-linear curves on tuning other network parameters. Henceforth, sigmoid activation is used in other NODE models presented in this thesis.

Table 3.3: Loss scores after training with different activation functions

Activation function	Train loss value (RMSE)	Test loss value (RMSE)
Sigmoid	2.37905	21.38202
tanh	2.43350	23.98665
Relu	1.90882	1.09465e27

With sigmoid activation, the impact of changing the width and depth of the neural network is considered. Figure 3.13 shows the change in learning capabilities of NODEs by adding more neurons in the hidden layer. While there is no significant difference seen on changing the number of neurons up to 25, increasing the number up to 50 shows improved learning of

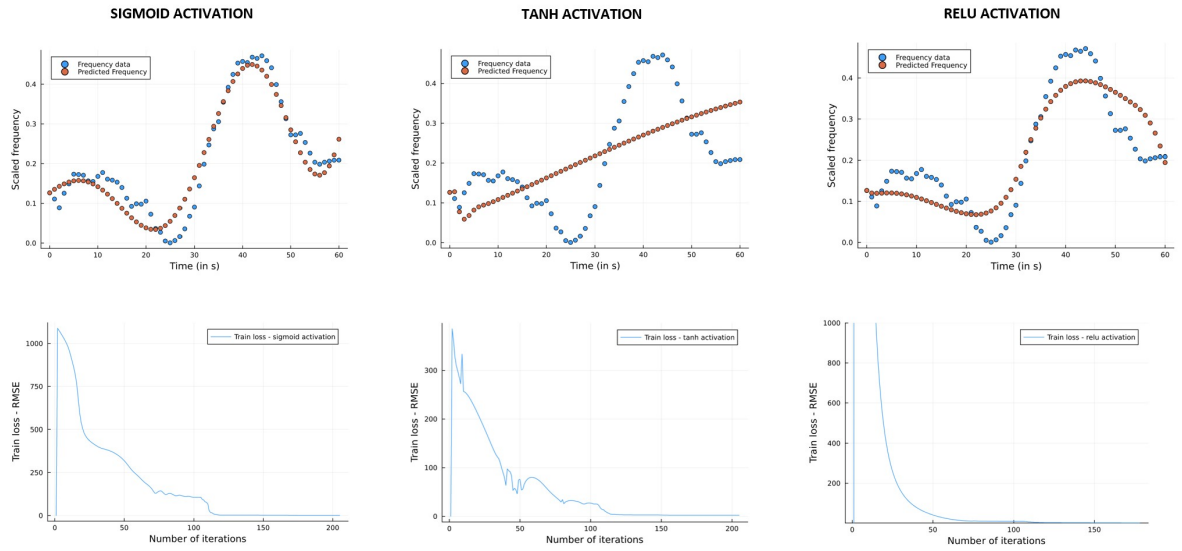


Figure 3.12: Impact of activation functions on training NODEs on an identical network structure for a 5-state system

Table 3.4: Test loss scores for different widths of hidden layer

	NODE models with different hidden layers		
Number of neurons in hidden layer	10	25	50
Test loss value (RMSE)	17.13825	16.93585	20.77384

the non-linear curve in the same number of iterations. However, as shown in Table 3.4, the test loss for the model with 50 neurons in hidden layer is higher and indicates a case of relatively higher over-fitting. Hence, the choice of the width depends on the output requirements from the model. If it is important to capture the oscillatory behaviour of frequency, higher width in hidden layers is preferable. If predicting the overall mean trajectory of the frequency is more important, shorter widths of hidden layer could be preferred.

Figure 3.14 shows the impact of adding an extra hidden layer on the learning capabilities of the NODE model. Visually, adding one or more hidden layers do not lead to much differences in the prediction performance of the NODE model. However, the difference in learning could be observed in the loss function plots. While more hidden layers seem to cause more erratic learning patterns, it is possible to tune these patterns by changing other parameters like the learning rate and the number of iterations in the optimizer. Hence, depending on the nature of the frequency data being processed (based on factors like real-time or synthetic data, normal or abnormal frequency data etc.), more hidden layers could be added to check for possible improvement in model performance.

It is important to note that there is not necessarily a best combination of activation, width and depth of hidden layers for the frequency prediction problem. It is possible to achieve high performance with different combinations of network parameters by tuning the other model parameters (like learning rates, loss function definition, number of input features, training time etc.). However, since sigmoid activation with a large width in the hidden layer provides reasonably good results, it is chosen to be a good starting point to train the prediction model

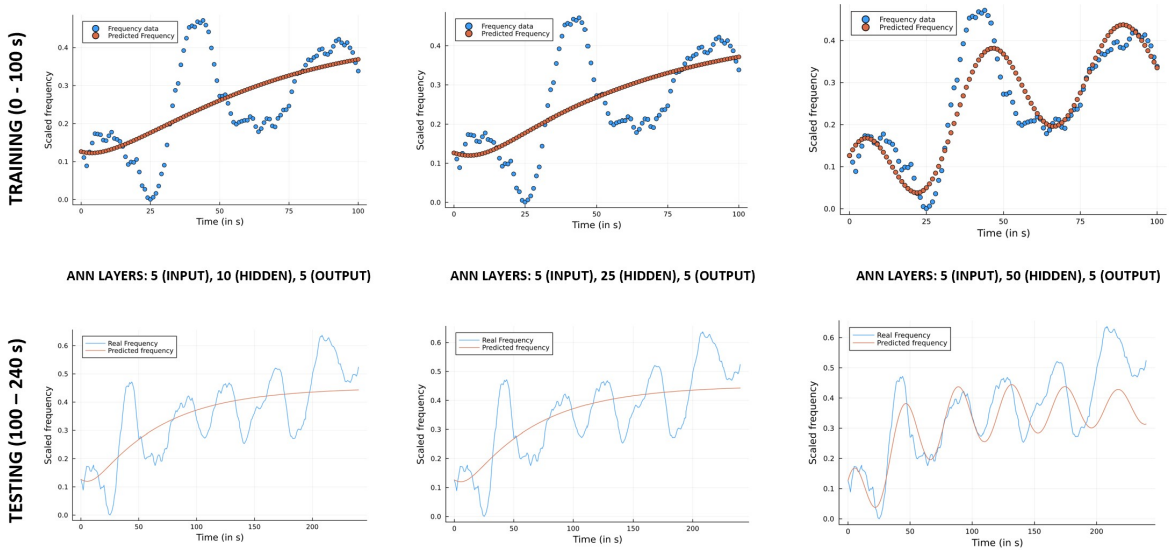


Figure 3.13: Impact of changing the width of a hidden layer on the learning capabilities of NODEs

on different frequency data and tune model parameters subsequently to further improve the prediction performance.

3.3.3. Loss Function Definition

Since the output prediction is expected to approximate a non-linear curve representing the system frequency, choosing a function that measures the fitting of the curve to target data would be required for the loss function. Using NODEs, it is possible to approximate multiple states in a non-linear system and learn the inter-dependencies among these states through the training process. With the help of PMU data corresponding to scenario 1, Figure 3.15 shows the expected training result on passing the given input features into the NODE model. So, to help the model learn the behaviour of all the passed states, the loss function must consider the deviations of all the state values from their respective target values.

An example loss function giving equal importance to all the states in the system could be written in Julia as:

$$loss = sum(abs2, [1, 1, 1, 1, 1]. * (real\_data. - predicted\_data))$$

real_data and predicted_data are matrices consisting of the time-series data points available for training, corresponding to all the states in the system. The above function finds the error among all elements in the feature matrix with respect to its target values, squares the error for each element and returns the sum of all the errors as the loss value. The weight matrix can be adjusted to give different weights to different features. Since frequency is the principle quantity requiring estimation and the other features are supporting quantities, assigning more weight to frequency helps the model to prioritize learning the frequency response better. This provides the model with more direction, and possibly explains the increased speed in learning as shown in Figure 3.16 and Figure 3.17. Assigning equal weights could still show a good prediction performance with an increase in the number of iterations and longer learning times.

Modifying the loss function in sightly different ways gives rise to a considerable differences in prediction performance as shown in Figure 3.18. The four different loss functions used to

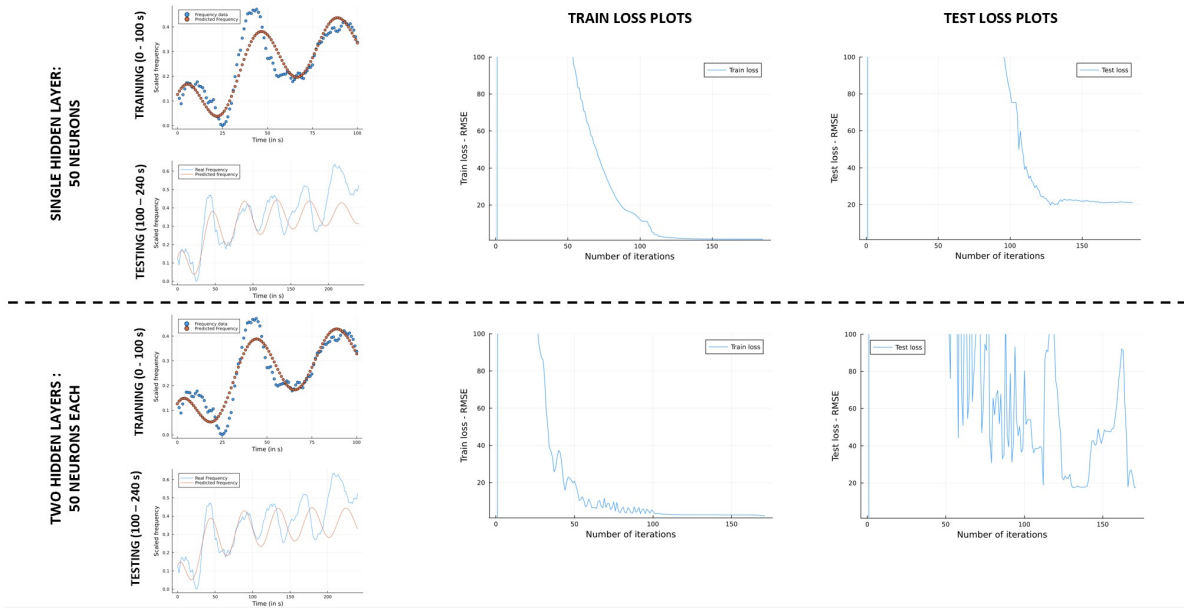


Figure 3.14: Impact of adding a hidden layer on the learning capabilities of NODEs

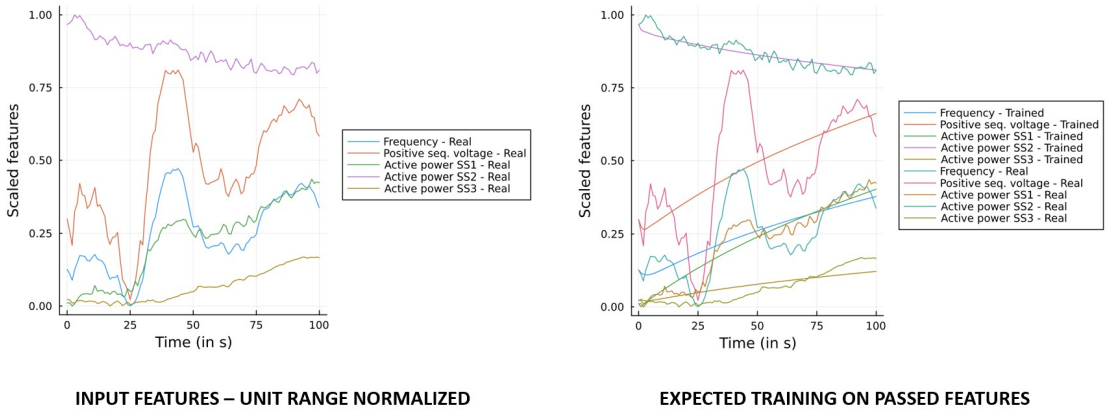


Figure 3.15: Expected learning of passed features by the NODE model

obtain the predictions in Figure 3.18 are:

$$loss\_function\_1 = sum(abs2, [2, 1, 1, 1, 1]. * (real\_data. - predicted\_data))/5$$

$$loss\_function\_2 = sqrt(sum(abs2, [2, 1, 1, 1, 1]. * (real\_data. - predicted\_data))/5)$$

$$loss\_function\_3 = sum(abs2, [2, 1, 1, 1, 1]. * (real\_data. - predicted\_data))$$

$$loss\_function\_4 = sqrt(sum(abs2, [2, 1, 1, 1, 1]. * (real\_data. - predicted\_data)))$$

While loss functions 1 and 2 seem to require more iterations to achieve better prediction performance, loss functions 3 and 4 provide relatively better approximating power in lesser number of iterations to the NODE model. Since computed training loss values often range anywhere between 1 to 10 in the final few iterations of training for any of the loss functions used, achieving a loss of, say 1, has different meanings in each case. The ability to bring down

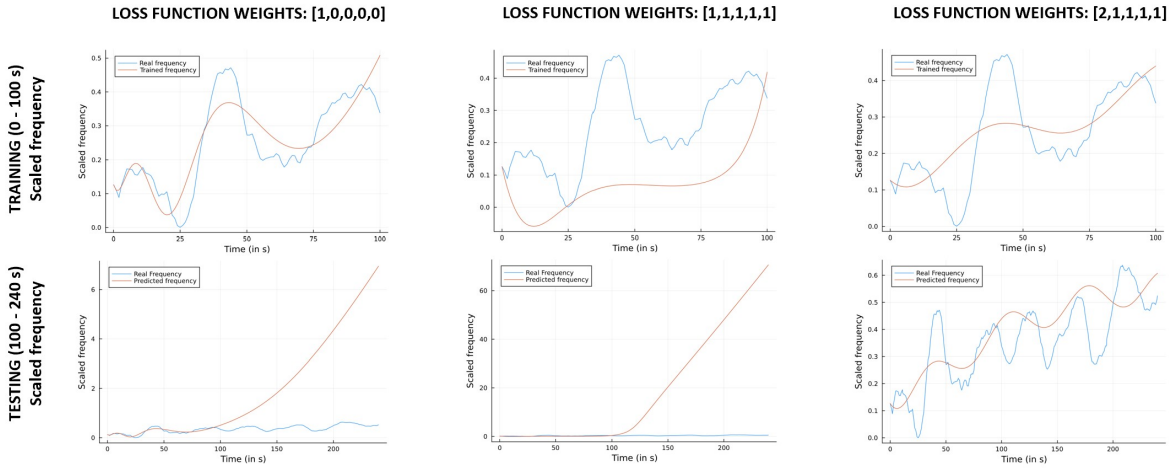


Figure 3.16: Impact of different weights in loss function on frequency prediction

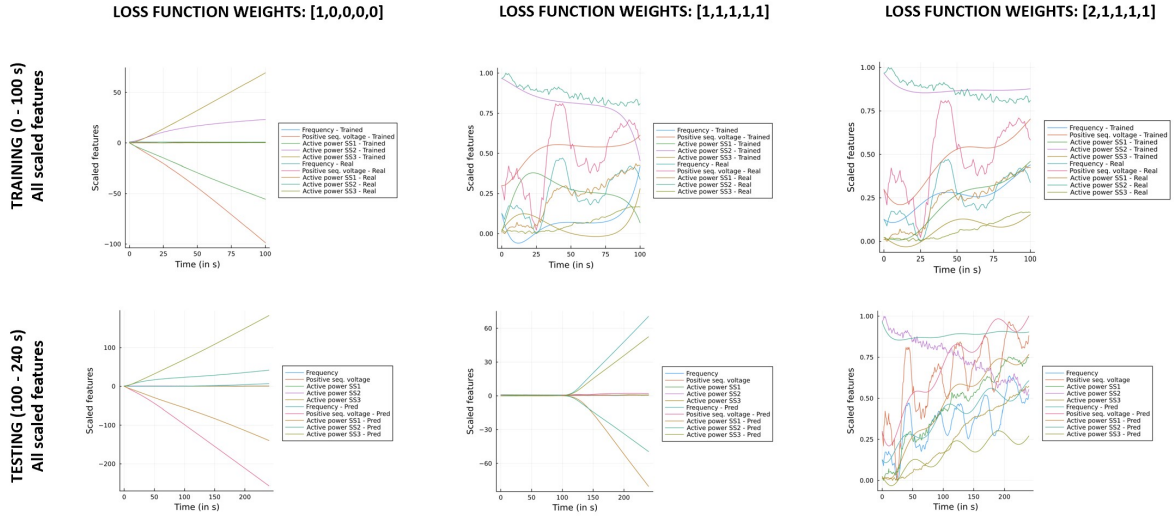


Figure 3.17: Impact of different weights in loss function on prediction of all features

the numeric value of any loss function to around 1 could be attributed to the optimizer used and its parameters. However, for a given combination of neural network structure and optimizers, choosing loss functions 3 and 4 ensures overall lower error magnitudes in all the predicted data. Among loss functions 3 and 4, loss function 3 penalises larger deviations more than loss function 4. This difference could explain the ability of loss function 4 to enable the model to learn short-term oscillatory information better than loss function 3, which, on the other hand, captures the mean trajectory of the frequency response better.

3.3.4. Optimizers

The Adam optimizer and the Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithm-based optimizer are used sequentially in all the prediction models used in this thesis. An example of training on frequency data using both the optimizers is shown in Figure 3.19. While Adam is one of the most commonly used optimizers for training neural networks, the BFGS optimizer is a gradient-based optimizer that shows higher convergence rate compared to gradient descent algorithms. Figure 3.20 shows the jump in performance of the model shown in Figure 3.19,

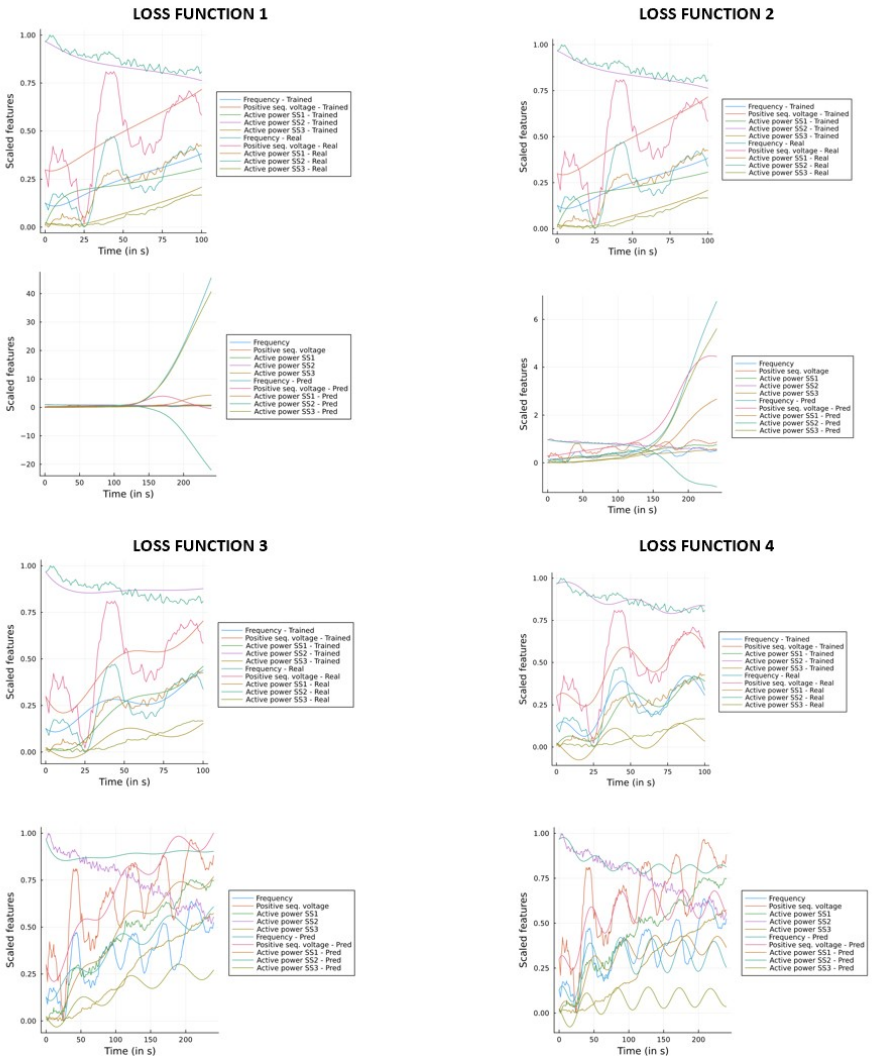


Figure 3.18: Training (0-100s) and prediction (100-240s) results from using different loss functions

on switching optimizers after the 100^{th} iteration. Since both the optimizers showed promising performance in the preliminary set of prediction models, other optimizers have not been considered. Some advantages of Adam include its simplicity in implementation, faster convergence rates and good adaptability of its learning rate. Both Adam and BFGS have very few parameters whose tuning can make the optimization highly adaptable to a range of learning requirements of the prediction models.

The learning rate α and the momentum values β_1 and β_2 in the Adam optimizer, and the initial_stepnorm value in the BFGS optimizer are a few parameters that could be tuned to improve learning performance. Stopping limits of the optimizer could be assigned based on different factors like maximum number of iterations, maximum time for optimization to run or tolerance values in changes in the objective value of the optimization problem. Table 3.5 and Figure 3.21 show the results of training with different learning rates in the Adam optimizer. It is evident that small changes in the learning rate have a considerable impact on the learning outcomes of the model. Similarly, changing initial_stepnorm values lead to changes in the loss values after training that are not quite significant for this example (see Table 3.6). Hence, to

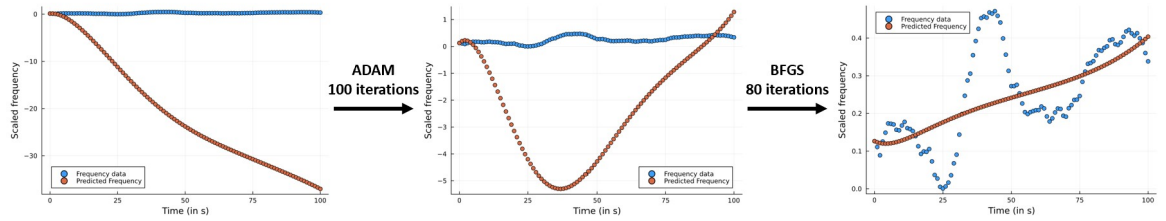


Figure 3.19: Sequential improvement in training results with Adam and BFGS optimizers

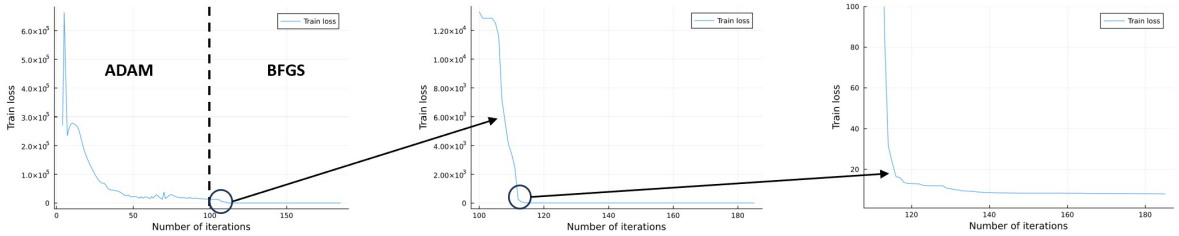


Figure 3.20: Converging loss function over 100 iterations of Adam and 80 iterations of BFGS optimization

achieve a better fit for such cases, other neural network parameters could be tuned.

Table 3.5: Training loss values for different learning rates in Adam optimizer

	100 iterations		200 iterations	
Alpha	Train loss value	Test loss value	Train loss value	Test loss value
0.001	12839.07965	2.40216e7	5288.92266	1.27512e7
0.005	615.48583	15232.90941	165.09250	4804.60210
0.01	333.75176	8853.94240	77.13131	2799.40234
0.05	123.70518	4992.40481	47.96406	2585.68578
0.1	675.58215	18578.57905	416.70319	12535.83436

3.4. Training Methods

Depending on the availability of data, presence of noise in data and the nature of the frequency event/response, different training methodologies are used on the predictive NODE models. While low availability of data leads to constraints for offline learning, having multiple data-sets for similar frequency events could allow for offline learning to enhance the real-time performance of a model. Also, any unexpected addition of a disturbance or a frequency control scheme at any point in the prediction horizon would require the NODE model to detect the change and re-calibrate accordingly. The following training methods consider these aspects depending on the nature of their respective test cases and corresponding data-sets.

3.4.1. Adjusting Starting Parameters for Online Training

With random initialisation of starting network parameters, it is possible to ensure there is no inherent biasing in the initial prediction of the NODE model. However, in some cases, it is possible that random initialisation could lead to very high initial loss values which require a larger number of iterations during online training for the predictions to converge to reasonable values. Trying to train the model with a fixed set of starting parameters that are more close

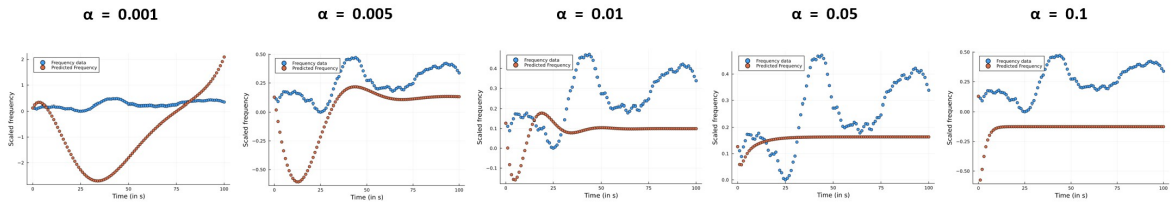


Figure 3.21: Training results from different parameter settings in Adam optimizer

Table 3.6: Training loss values for different initial_stepnorm values in BFGS optimizer

initial_stepnorm	Train loss value	Test loss value
0.001	7.44423	467.37722
0.01	6.81161	370.79517
0.05	6.81941	322.20040
0.1	6.85345	576.07417

in magnitude to the scaled features (in the range of 0 to 1) implies introducing a bias into the model that is not backed with a scientific reason (as shown in Figure 3.22). While introducing biased parameters helps scale down the initial prediction values to reasonable ranges, it could also affect the performance of the NODE model adversely due to the initial biasing.

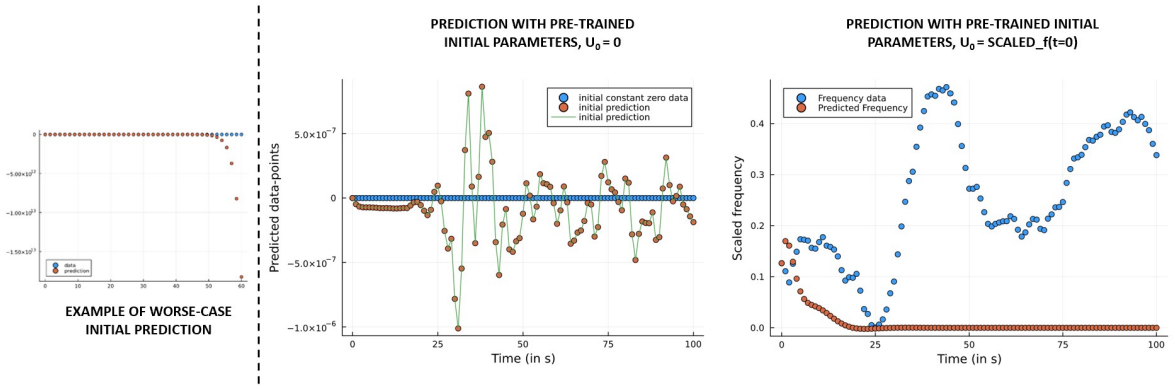


Figure 3.22: Possibility of using biased pre-trained starting parameters to replace worse case randomly generated initial parameters

One workaround is to use the Adam optimizer with pre-tuned parameters on a randomly initialised network during online training to bring the predictions closer to the scaled magnitude of features. If the prediction results after a fixed number of iterations using the Adam optimizer are still very farther from the target values (say, the predictions are in the range of 10^5 or more compared to target values in the range of 0 to 1), pre-trained starting parameters could be used. The latter option of using biased initial starting parameters is expected to be infrequent given the optimization capabilities of the Adam optimizer.

Another option to ensure better prediction performance is to choose from consecutively generated random parameter sets based on their initial loss scores. Table 3.7 shows the train loss values for predictions from 8 consecutively generated random starting parameter values. The maximum deviations of predicted frequency values (with respect to scaled frequency values) are all less than 100, which are good starting predictions for the optimizers to work with.

Table 3.7: Starting predictions using 8 consecutive randomly initialised set of parameters

Iteration number:	1	2	3	4	5	6	7	8
Initial loss	9.2e05	2.8e06	4.0e06	1.6e06	4.8e06	3.8e06	2.8e06	1.5e06
Max. deviation (predicted frequency)	21.7	84.8	3.8	51.3	44.3	78.7	17.3	93.4

Hence, for almost all cases, this selection method is expected to work well. However, it is also safe to have an alternate set of starting parameters for worst-case situations during online training, as the estimates are required to provide fast, real-time estimates about the system.

3.4.2. Detection of Change in Frequency Restoration Response

The predicting capabilities of a NODE model depends largely on the nature of the event it is trained on. For instance, if a model is trained with normal operating range frequency data, it is capable of learning the minor fluctuations that might occur in a system that is in its stable state. This model would perform badly in situations where new control actions or major disturbances or any topological changes are introduced in the system. Hence, training with the right set of event data corresponding to the relevant system and suitable influencing features is required.

To demonstrate the ability of NODE to learn specific response patterns, a PMU data-set corresponding to a system-split event in Continental Europe grid originating from disturbances in the Balkan Peninsula was considered. The data received corresponded to the restoration phase of the frequency response as shown in Figure 3.2. It could be observed that as the system recovers to 50Hz, there is a minor drop in frequency again about an hour after the start of the available data. The system is seen to start restoring again towards 50Hz. This provides an opportunity to study the ability of a NODE model trained on initial restoration response to recreate a new restoration response arising from a disturbance much smaller in magnitude.

It is important to note that by assuming the system will show a similar recovery pattern, a bias is introduced into the NODE model. While it is possible and also likely for the system to show different types of frequency response depending on a multitude of factors governing frequency control over the large-area synchronous grid, this assumption biases the model into learning only a single response pattern that may or may not occur again. Though the model cannot be relied on to provide accurate real-world estimates of the frequency response (largely because of lack of information regarding all the control schemes that are acting on the system), it is still a good test case to explore the pattern recreation and real-time retraining capabilities of the NODE model to provide predictions for similar nature of frequency events.

To detect a second restoration pattern, the deviation between the predicted values from the initial online-trained model and the real-time measured values (referred to as target values in Figure 3.23) is observed. When the deviation crosses a pre-defined limit, the model starts observing and waits until another restoration pattern is detected (by checking for zero-crossing, in this case). Once the pattern in detected, the model uses the point of maximum deviation from the past data as the new initial point for the NODE solver. First, an initial prediction is made with the same NODE model, but with a different starting point u_0. Second, a short time-window of data starting from u_0 is used for real-time retraining of the model to provide an improved frequency response estimate for the next few minutes. The impact of retraining the model after detecting a new pattern could be seen in terms of improvement in the evaluation metric values

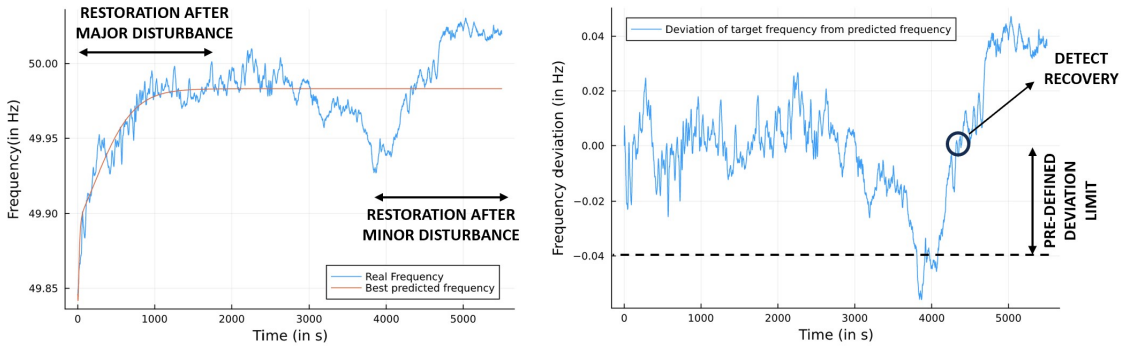


Figure 3.23: Possibility of NODE to recreate frequency response for similar type of events

of the predictions.

3.4.3. Offline Training for Improved Starting Parameters

With multiple data-sets available for training from different simulations on the modified IEEE39 grid, it is possible to incorporate offline training to introduce prior information about frequency dynamics into the NODE model.

While Table 3.2 indicates 35 different data-sets in total, not all of them are practically relevant for training and testing a NODE model. This is because, at high RES penetration levels and during large power imbalance situations, the system is neither equipped with sufficient frequency control nor has well-defined operational constraints and limits to mimic real-world operation of similar power systems. Hence, frequency could drop to unrealistic values (for instance, less than 47 - 48 Hz) and the simulations still show a frequency response wherein the frequency slowly recovers to a stable yet practically infeasible value. To ensure an acceptable degree of practical relevance, 9 scenarios are chosen to obtain training and testing data-sets from (as shown in Figure 3.24).

Since the IEEE39 system was modified to introduce LFDD schemes and study their impact on frequency response, the frequency limits breached for a given event size act as triggers for implementing demand disconnection. If real-time frequency prediction could be used for aiding similar demand-disconnect schemes, estimating the frequency thresholds that will be breached until the nadir, the time taken until a threshold is breached and the settling frequency values a few seconds or minutes after the nadir could act as useful information. To indicate frequency thresholds, every 0.2Hz interval below 49Hz is represented as a trap as seen in Figure 3.24.

The least-impact event corresponding to all synchronous generation (i.e., 0% RES penetration) and a power imbalance of 300MW is chosen for offline training the NODE model. This fulfills the main purpose of offline training - to introduce information about how different system quantities (like frequency, voltage, rotor angle and active power) respond during a power imbalance event to the prediction model. The entire 120 seconds of data is used to obtain a trained NODE model shown in Figure 3.25.

By offline-training, valuable computational time is saved and faster convergence can be achieved for online-trained models. The decrease in loss value over 350 iterations (200 iterations of Adam + 150 iterations of BFGS) is shown in Figure 3.26. For this test case, the loss function is similar to the loss_function_3 mentioned in subsection 3.3.3, only differing by the fact that all features are given equal weights in this case. It takes the model 64.828 seconds to

Data-set (RES%_EventSize)	Duration (s)	Nadir - Value (Hz)	Nadir - Time (s)	Settling value (Hz)
0%_300MW	120	49.491199	22.501666	49.665722
0%_500MW	120	48.963955	28.11	49.43575
9%_300MW	120	49.415513	24.435	49.642382
9%_500MW	120	48.759124	29.858333	49.401231
20%_300MW	120	49.277489	28.394999	49.613933
20%_500MW	120	48.558483	30.168333	49.334291
33%_300MW	120	49.146053	28.351666	49.575166
43%_300MW	120	48.978162	29.001666	49.544446
53%_300MW	120	48.812573	28.731666	49.470357

	Nadir Values
	> 49Hz
	< 49Hz, > 48.8Hz - TRAP 1
	< 48.8Hz, > 48.6Hz - TRAP 2
	< 48.6Hz, > 48.4Hz - TRAP 3

Figure 3.24: Set of simulated scenarios used for training and testing NODE prediction model

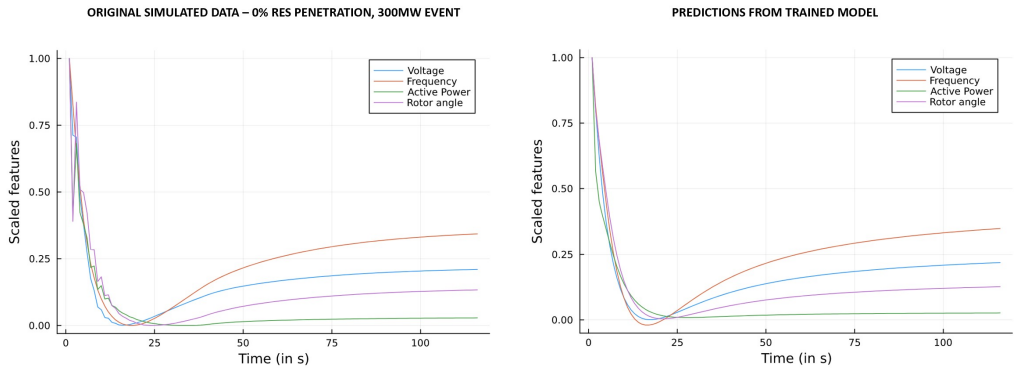


Figure 3.25: Predictions from offline-trained NODE model

train and reach a final loss value of 0.3325 starting from a initial loss value of 781865.3805. Since the unseen test data-sets will consist of scaled values with their respective frequency nadirs and settling values in similar ranges as that of the offline-trained values, it is easier for the online models to use the pre-trained starting parameters for real-time training. This is a necessity as the nadirs are expected to occur in about 20 to 30 seconds after an event is detected. Hence, every second saved helps in improving the prediction capability of the model.

3.5. Performance Evaluation

Performance metrics for all the test cases presented in this thesis focus solely on the frequency predictions, unlike the loss function formulation. In case of PMU data-sets, regression metrics measuring accuracy and those that are indicative of practical relevance of the predictions are considered important. The ability of the model to capture the future mean trajectory or possible oscillatory instabilities in the next few minutes are considered significant when a system operates in normal frequency ranges. On the other hand, in case of specific frequency events generated by simulations, it is considered more important to capture the instability limits breached by a system. For instance, the frequency thresholds breached or the expected settling time during restoration are key aspects for analysing the stability of a system. Hence,

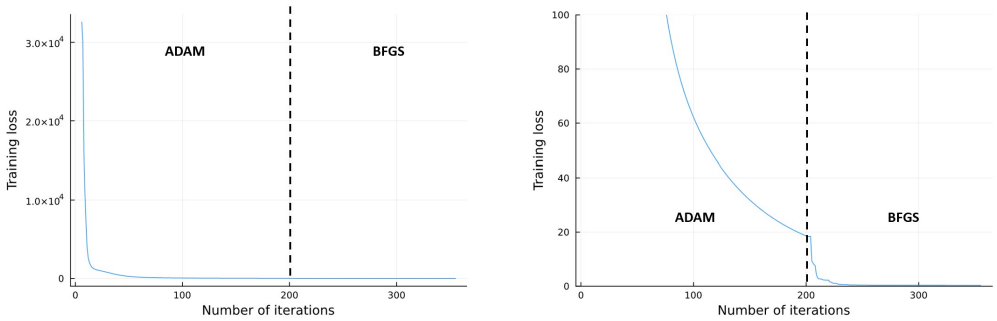


Figure 3.26: Training loss curve from offline training

regression metrics to predict the severity of a frequency event are preferred for the test cases with synthetic data.

In case of both the PMU data and the simulated data, it is important for the regression prediction to fit the original frequency curve well. Mean absolute error (MAE) of the predicted frequency values from the target values would be a suitable performance metric for regression using PMU data. Since capturing noise or minor oscillations of frequency in the normal operating range is not very useful, penalising higher deviations is not quite necessary. Hence, RMSE or MSE is not preferred for performance evaluation of the test cases with PMU data. The two regression metrics used for evaluating performance on PMU data in chapter 4 are:

- **Mean absolute error:** It directly represents how close in magnitude the predicted values are to the target values on an average basis.
- **Maximum absolute error:** The maximum deviation in the predicted value from the target value in Hz shows the worst performing prediction for a given data-set.

To evaluate prediction results on simulated data, specific aspects of the frequency curve like the nadir and settling value are important. Hence, deviation in the estimates of such critical values from the real values are used as regression metrics. The three regression metrics used for evaluating performance on simulated data in chapter 4 are:

- **Deviation of estimated frequency nadir value:** Since the test cases represent large frequency disturbance events, the frequency nadirs are expected to reach lower magnitudes than the normal operating range during the simulations. The difference between the estimated nadir and the actual nadir in the simulation is, thus, used as a performance metric.
- **Deviation in estimated time (T_{nadir}) of when the frequency nadir occurs:** Estimation of the time point at which the nadir occurs could provide key information about the dynamics of a system and the expected recovery period post an event. Deviation of the estimated T_{nadir} from the the actual T_{nadir} in the simulation is, thus, observed in the prediction results.
- **Deviation in estimated frequency value 2 minutes after the onset of a event:** Since all the data-sets available for testing having a maximum duration of 2 minutes, the final frequency value of the predicted curve and the original curved are compared. For low RES penetration and low power imbalance scenarios, this final value represents a settled frequency value as secondary and tertiary controls are not activated in the simulated system. For larger power imbalance scenarios, it is possible that the frequency has not settled within the 2-minute window.

3.6. Conclusion

The training methods and tuneable aspects present in NODE algorithms that enable the application of NODE to frequency security assessment have been addressed elaborately in this section. Depending on the nature of the frequency assessment situation, the presented methods could be modified and adapted to achieve relevant prediction results. The final prediction results, observations during the implementation of NODE models for different test cases and the impact of using the training methods introduced in this section on enhancing the predictive powers of NODE models are presented thereupon in chapter 4.

4

Results & Discussion

4.1. Applications of the Predictive Model - Case Studies

Predictive NODE models applied to a set of different test cases are presented in this section. The models have been tuned with respect to the test case and appropriate training methods from section 3.4 have been implemented. For all the test cases, the simple validation approach shown in Figure 2.14 has been implemented to find a well-performing set of prediction parameters. In each of the prediction result plots, the best predicted frequency curve corresponds to the parameters chosen after the validation approach and the final predicted frequency curve corresponds to the parameters obtained after the maximum number of iterations specified in the optimizers. All the models use Tsit5 as their ODE solver, and work with the Adam and BFGS optimizers.

4.1.1. PMU Data: Normal Operating Frequency

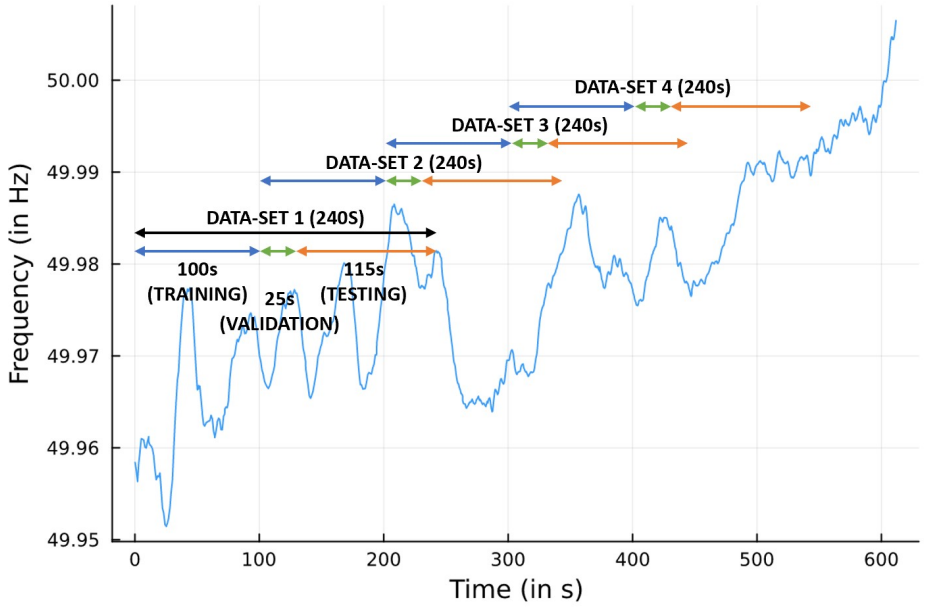


Figure 4.1: Data-sets used for producing results from the available 10 minute window of data

For frequency prediction in the normal operation scenario, a generic prediction model that is capable of learning in a fixed time with past data to predict for the the next fixed interval of time is required. To show the generalising ability of the model, a large time window of 10 minutes of frequency data is considered. The same pre-tuned model is trained with four data-sets (each with a total time-span of 4 minutes) taken from the 10-minute data, as shown in Figure 4.1. With each data-set, the model trains on 100 seconds of past data, validates with the subsequent 25 seconds of data and predicts for the next 115 seconds. The network parameters and relevant model information are summarised in (add table in Comparison section).

Table 4.1: Training and performance metrics for prediction results from PMU data - normal operation scenario

Data-sets	Training time (s)	MAE (Hz)	Maximum absolute error (Hz)	Minimum validation loss (unit-less)
Data-set 1	75.3	0.0047	0.0118	76.75
Data-set 2	48.2	0.0059	0.0102	84.81
Data-set 3	43.5	0.0099	0.0156	23.24
Data-set 4	42.6	0.0061	0.0121	22.76

The training and prediction results on using a fixed prediction model on four different data-sets are shown in Table 4.1 and Figure 4.2. The oscillations shown in the frequency data are very small in magnitude and do not represent any significant change in the system. Hence, a prediction model that can fit any given frequency data to provide an estimate of the direction in which the frequency trajectory would evolve in the next many seconds is used in this test case. While it is possible to capture the oscillatory behaviour (though farther in terms of the magnitude predicted) by increasing the number of iterations or tuning the neural network differently, it could lead to over-fitting of the model.

It might seem ideal for the model to be able to predict the oscillatory behaviour while ensuring it is in the right direction in which frequency evolves. This is not feasible in this test case due to a few reasons. Firstly, the model tries to learn the evolution of other quantities like active power and voltage simultaneously, while trying to find the inter-dependencies of these quantities with the frequency. The chosen set of features might be completely unrelated to the unexpected minor oscillations seen in the frequency response during normal operational conditions. Secondly, the erratic behaviour in frequency could be attributed to subtle changes in the electrical system that are insignificant, and are inherently present in any system with electro-mechanical components. It is, thus, not possible for the NODE model to learn small-scale erratic behaviour arising due to non-traceable physical reasons from the available PMU data. However, this test case helps in demonstrating the fundamental predicting capabilities of NODE models for frequency data.

4.1.2. PMU Data: Frequency Restoration Post-System Split Event

The PMU data-set corresponding to the restoration of frequency after a system-split in Continental Europe grid has a total duration of about 90 minutes (5500 seconds, precisely). The data-split for training, validation and testing, and the available features after data preparation are shown in Figure 4.3.

While the NODE model trains online from the start of the available data, a training data corresponding to about 1000 seconds is required to approximate the mean value and direction in which the response evolves over the next hour. Though this duration of training data might be

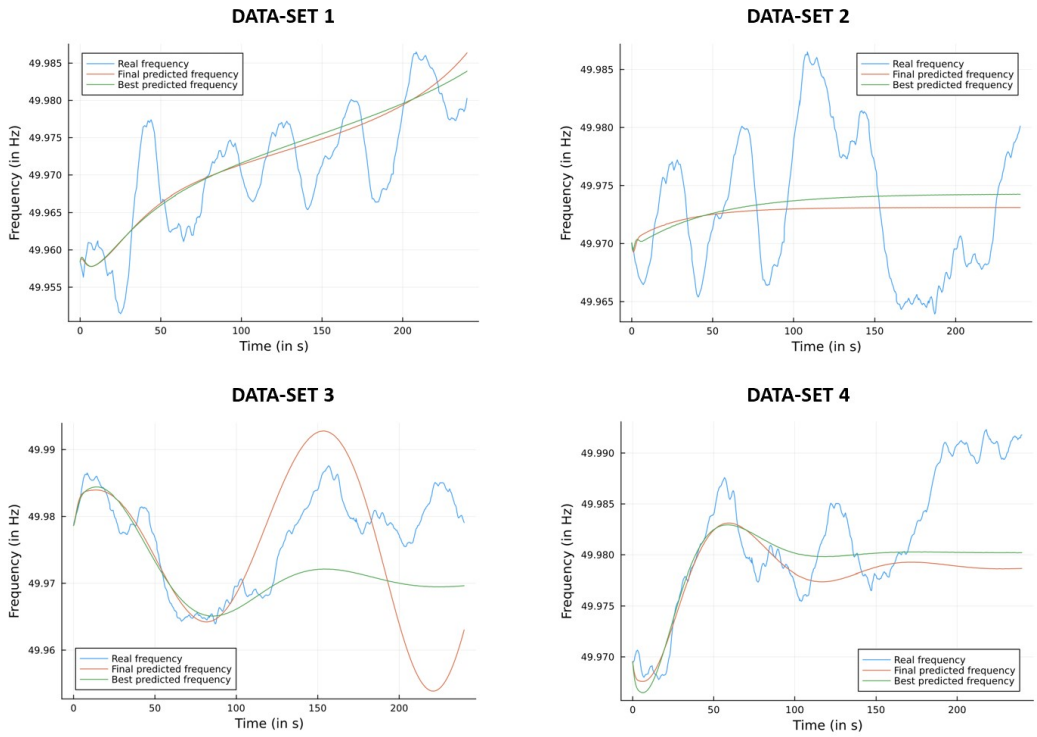


Figure 4.2: Prediction results from PMU data - normal operation scenario

quite long for producing fast real-time estimates about the system, it is still useful in learning the typical restoration response of the system in order to detect and predict for any similar restoration scenarios in the future.

Table 4.2: Training and performance metrics for preliminary prediction results from PMU data (only frequency) - restoration scenario

Training time (s)	MAE (Hz)	Maximum absolute error (Hz)	Minimum validation loss (unit-less)
120.8	0.0155	0.0560	53.06

The preliminary prediction results and corresponding performance metrics for the entire duration of available data are shown in Table 4.2 and Figure 4.4. The NODE model is trained initially with all the four features, and then trained with only the frequency data. It is observed that training with only frequency data is more computationally efficient and easier for the model to learn from. The spikes in the initial seconds shown in the prediction results from training on all features could be due to the influence of initial oscillations present in other features. Since the frequency event and eventual response are majorly triggered by actions occurring in different part of the synchronous grid of Continental Europe, the local voltage and active power measurements in the Netherlands grid might not significantly add any new information regarding frequency to the NODE model. In fact, since voltage and frequency show high correlation in all the available PMU data-sets, it could be said that the trajectory of local voltages are but a result of the frequency disturbance due to the system-split event. Hence, using only frequency data was nominal for training the frequency prediction NODE model. Owing to the large duration of training data and high noise content, the model took

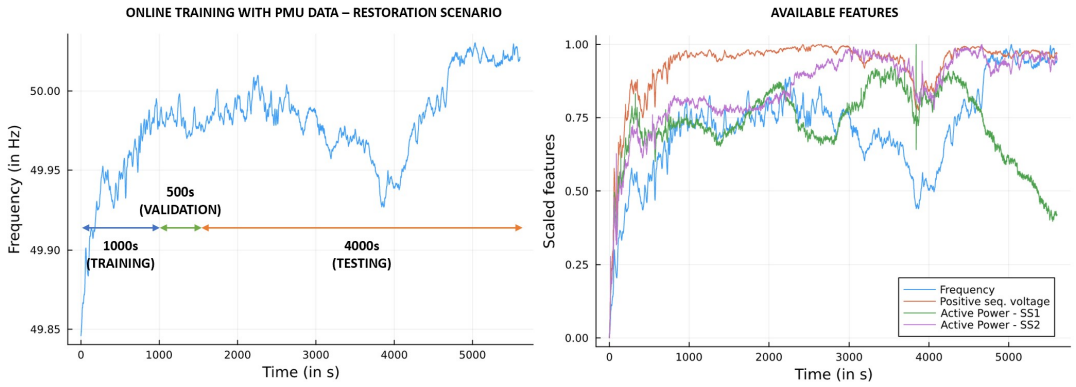


Figure 4.3: Available features and data-split for preliminary online training with PMU data - restoration scenario

about 120 seconds to train and produce the preliminary prediction results. The best prediction corresponds to the result with minimum observed validation loss. The computed loss is the sum of squared deviations of predicted values from target values, and is unit-less as the model operates with scaled frequency data during training.

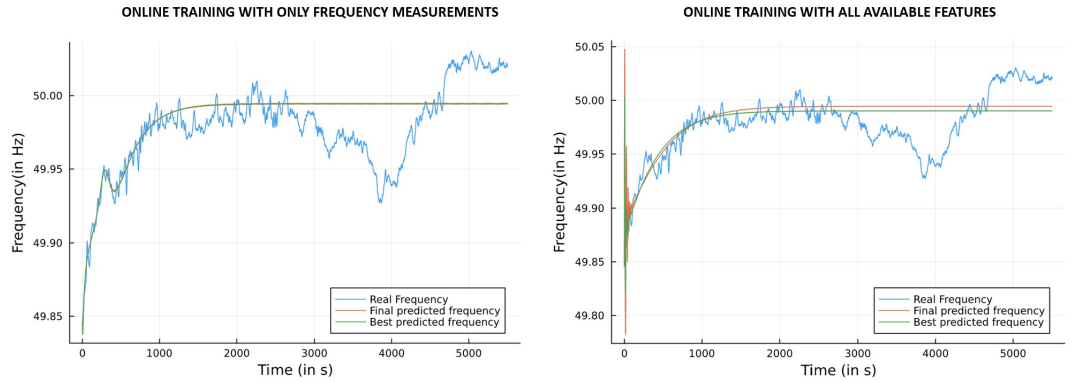


Figure 4.4: Preliminary prediction results from PMU data - restoration scenario. Left: Trained with only frequency data. Right: Trained with all available features

After the detection of a minor restoration at around 4000s and finding a new starting point u_0 to make better predictions as described in subsection 3.4.2, prediction results are obtained with and without retraining the NODE model as shown in Figure 4.5. In the first case, the model predicts with the same parameters obtained after training on the preliminary frequency data but uses the local minimum in the curve as the new starting point. With a changed input to the ODE solver, an improved prediction is obtained and the corresponding performance metrics are given in Table 4.3. In the second case, with the new starting point and pre-trained parameters, the NODE model is further retrained using about 500 seconds of data to adjust itself to restoration pattern being observed. The retraining time taken by the model (see Table 4.3) is short as the model starts by already having prior information about a similar restoration pattern. The retraining also uses lesser number of iterations in the optimizers to ensure that the model does not over-fit to the newly observed data and consequently lose previously learnt information from initial training. Among the multiple iterations, the best predicting set of parameters are chosen based on the observed validation loss and the results are plotted as shown in Figure 4.5. The evident improvement in the fit of the predicted curve after retraining

is reflected in the performance metrics presented in Table 4.3.

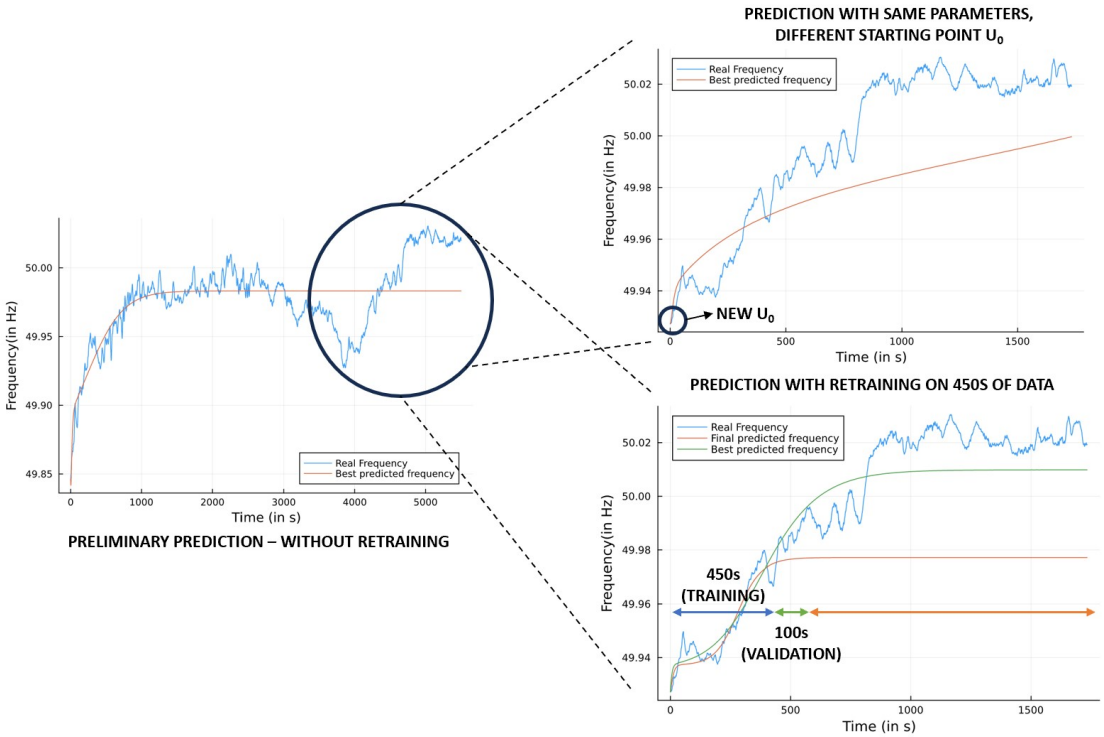


Figure 4.5: Prediction results from retraining the preliminary model after the minor disturbance

Table 4.3: Training and performance metrics for prediction results before and after retraining with PMU data - restoration scenario

	Training time (s)	MAE (Hz)	Maximum absolute error (Hz)	Minimum validation loss (unit-less)
Without retraining	0	0.0216	0.0419	N/A
With retraining	20.53	0.0091	0.0208	31.02

Starting retraining with prior information introduces a bias into the NODE model that expects any restoration detected to follow a similar pattern. However, such a pattern might not prevail in reality when there are other frequency control schemes in place. The assumption of expecting similar responses for any detected restoration holds true only when the control factors acting on the system are the same as the ones that were active during initial training of the model. While retraining of the model can account for any minor factors that are newly active in the system, the impact of any change in or new activation of important power system components cannot be predicted by the existing model. Hence, the status of the system and the active power system components/parameters during initial training are key indicators of the information processed and withheld by the predictive NODE model before it starts retraining.

4.1.3. Synthetic data: High-impact Frequency Events

All prediction models for the synthetic data start with parameters obtained from offline training on the 0% RES 300 MW event, wherein the predicted frequency curve is made to exactly fit

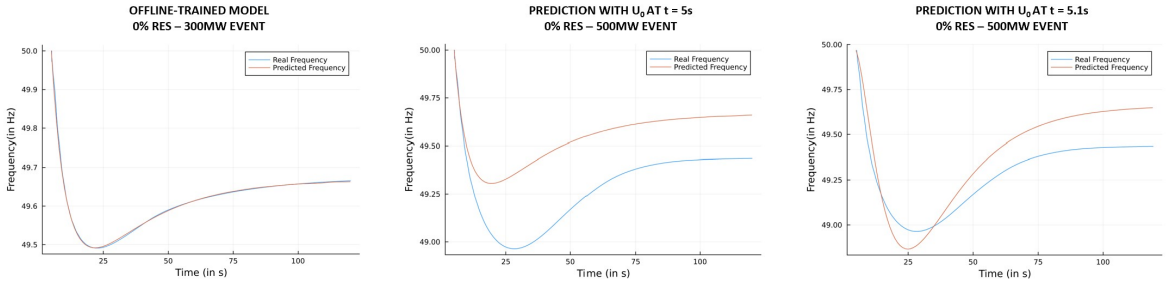


Figure 4.6: Impact of choosing a different starting point u_0 on prediction results for unseen events

the original curve. Due to absence of noise, it is possible for NODE models to almost exactly replicate the original curve during offline training by using the entire duration of available data for learning. In all the simulated events, the frequency event occurs exactly at t = 5s. As an initial result, the offline-trained NODE model is used to predict for one unseen test case (0% RES 500 MW event) only by changing the starting point u_0 that is fed to the NODE model. To observe the impact of using a starting point from a few milliseconds after the onset of the event, two different offline models are trained: one starting from t = 5s and the other starting from t = 5.1s. The prediction curve from the initial offline-trained model starting from t = 5s and the two prediction curves for unseen test data obtained using different starting point inputs are shown in Figure 4.6. In the first few milliseconds after the onset of the event, all the measured quantities (including voltage, rotor angle and active power) are in very similar magnitudes in different data-sets. Since the difference in response can be observed at about t = 5.1s, the consequence of using this information as input for the starting point for the ODE solver reflects as an improvement in the prediction of the NODE model.

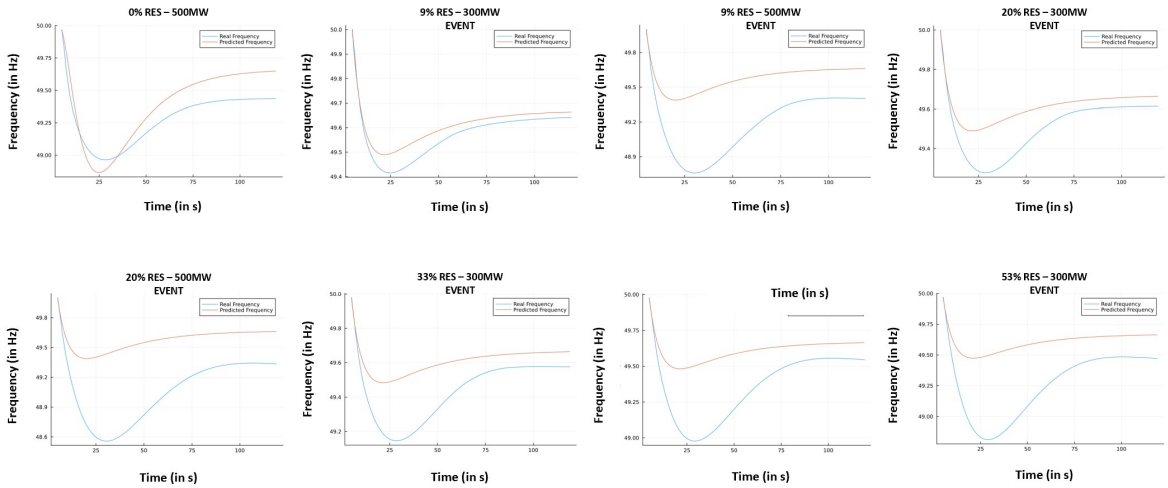


Figure 4.7: Prediction results for unseen test cases using their respective starting points at t = 5.1s and offline-trained parameters

Using a similar approach of taking the starting point at t = 5.1s, predictions are made for all the chosen scenarios with the offline-trained model (see Figure 4.7). The prediction performance deteriorates quite fast for increasing RES penetration levels and larger event sizes. This drop in performance could be attributed to too little information being available to the offline-trained model to predict for varying system scenarios.

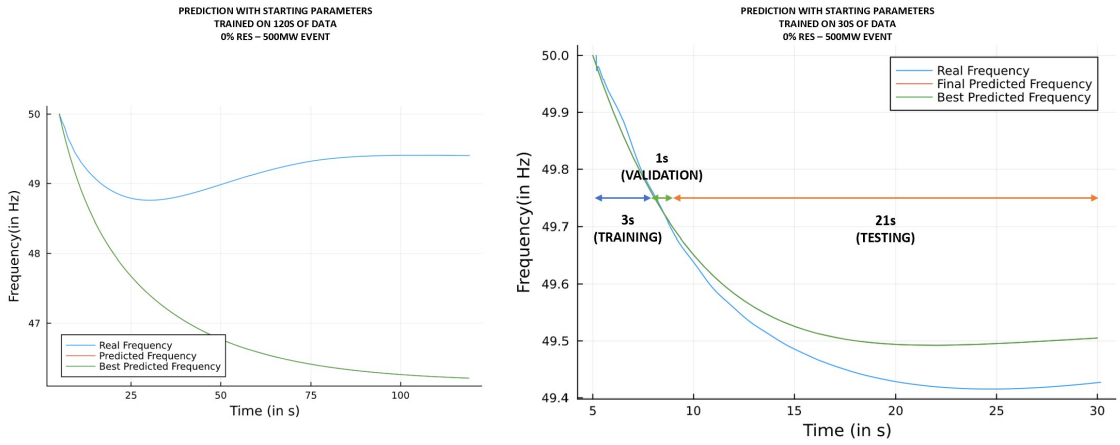


Figure 4.8: Data-split for training, validating and testing the NODE model to predict frequency nadir

In order to provide more information about the real-time status of the system, a short time-window of a few seconds after the onset of an event is chosen for retraining the offline-trained NODE model. Two changes are made to the model at this stage. Firstly, offline training is carried out separately on two sections of the frequency curve - the section from the onset of an event until the nadir and the section after the nadir up to t = 120s. As replicating the entire frequency response based on training on a few seconds of steeply dropping frequency data makes the model over-fit to the local trends in the curve during retraining (see Figure 4.8), separation of the curve into two sections corresponding to nadir occurrence and post-nadir recovery is preferred. Secondly, since re-training has to happen very fast to provide immediate short-term predictions, the features being used are reduced to include only the frequency data (similar to the test case with PMU data corresponding to restoration scenario). The model is still able to predict frequency with reasonable accuracy (as seen in the right hand test plot in Figure 4.8) when only frequency is used for training the model.

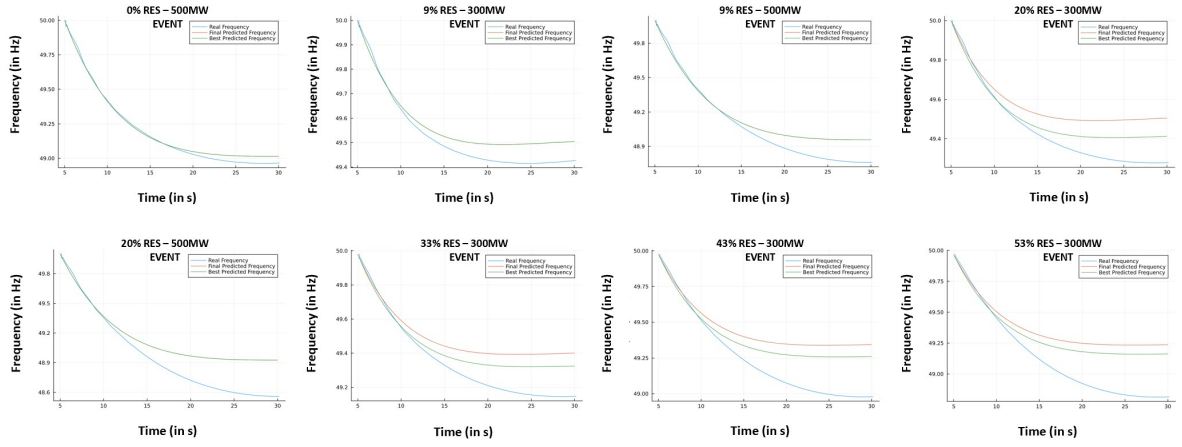


Figure 4.9: Frequency nadir prediction results for unseen test cases using real-time retraining of the NODE model for 3 seconds after the onset of an event

For the nadir prediction case, 3 seconds of data are used for quick online training followed by 1 second of validation data. Since the number of iterations in the optimizer are as low as 10, the time taken by the NODE model for retraining from the offline-trained parameters

is almost instant and always less than 1 second. Having a lower number of iterations also ensures that model does not over-fit to the local patterns of the observed frequency curve. The prediction plots and the corresponding performance metrics for the obtained results are shown in Figure 4.9 and Table 4.4, respectively.

Table 4.4: Frequency nadir prediction model - Results

Scenario (RES%_EventSize)	Real Nadir (Hz)	Predicted Nadir (Hz)	Nadir Deviation (Hz)	Real T_{nadir} (s)	Predicted T_{nadir} (s)
0%_500MW	48.9640	49.0143	0.0503	28.11	28.5
9%_300MW	49.4155	49.4924	0.0768	24.44	22
9%_500MW	48.7591	48.9579	0.1988	29.86	29
20%_300MW	49.2775	49.4058	0.1283	28.39	23.5
20%_500MW	48.5585	48.9258	0.3673	30.17	29.5
33%_300MW	49.1461	49.3203	0.1742	28.35	25
43%_300MW	48.9782	49.2586	0.2804	29.00	26
53%_300MW	48.8126	49.1606	0.3480	28.73	26.5

It can be observed that the deviation of the predicted nadir values from the real values range from a closest of 0.05Hz to farthest of 0.37 Hz across different scenarios. Among all the 300 MW events, the deviation shows a progressive increase with decrease in system inertia levels and increase in RES penetration. The deviation in the predicted time at which nadir (T_{nadir}) occurs from the real values, on the other hand, shows a more erratic behaviour across the scenarios. Nevertheless, the absolute deviation ranges from a lowest of 0.39 seconds to a highest of 4.89 seconds. In fact, all scenarios except the 20% RES 300 MW scenario have an absolute deviation in T_{nadir} of less than 3 seconds.

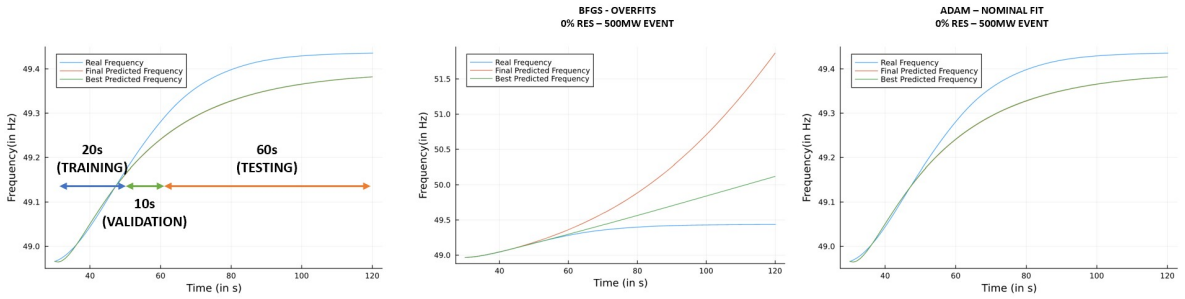


Figure 4.10: Retraining the NODE model to predict the post-nadir restoration curve and the impact of using different optimizers for real-time retraining

The post-nadir recovery section is longer in duration compared to the first section of the frequency curve. Accordingly, the data-split for training, validation and testing are shown in Figure 4.10. Unlike the previous section of the curve, a higher number of iterations in the optimizer are required to learn the higher number of data-points and fit reasonably well for longer duration. In an attempt to reduce the number of iterations to low numbers, the BFGS optimizer is tried for retraining the offline-trained model. In spite of changing the different optimizer parameters (namely, the initial_stepnorm and maxiters values), the model tends to over-fit to the training part of the curve in as low as about 5 to 10 iterations. To achieve a slower learning rate and a more gradual modification of the offline-trained curve to fit on the

newly encountered training data, the Adam optimizer is used. Though the model takes about 200 iterations to show a good fit, the training time taken by the model to learn is still as low as a maximum of 6 seconds. This could be attributed to prior training of the model on the data-set of the base test case - 0% RES 300 MW scenario. The difference in prediction results on using BFGS and Adam optimizers for retraining the NODE model is shown in Figure 4.10.

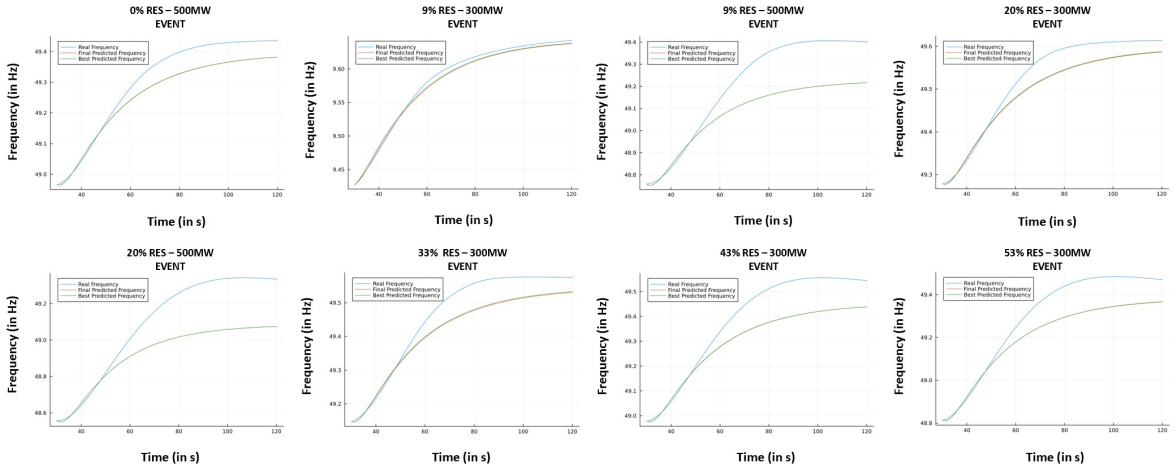


Figure 4.11: Post-nadir frequency curve prediction results for unseen test cases using real-time retraining of the NODE model for 20 seconds after the occurrence of the nadir

Table 4.5: Post-nadir frequency curve prediction model - Results

Scenario (RES%_EventSize)	Real Frequency at t = 120s (Hz)	Predicted Frequency at t = 120s (Hz)	Frequency Deviation (Hz)
0%_500MW	49.4357	49.3822	-0.0536
9%_300MW	49.6424	49.6382	-0.0042
9%_500MW	49.4012	49.2163	-0.1848
20%_300MW	49.6139	49.5876	-0.0263
20%_500MW	49.3343	49.0748	-0.2595
33%_300MW	49.5752	49.5328	-0.0424
43%_300MW	49.5444	49.4387	-0.1057
53%_300MW	49.4704	49.3659	-0.1045

The prediction results of post-nadir recovery from training on 20 seconds of data after the nadir and validating on the next 10 seconds of data for different system scenarios are shown in Figure 4.11. The corresponding performance metrics are given in Table 4.5. The deviation in predicted frequency values are relatively smaller compared to the results from nadir prediction. Similar to the deviation in predicted nadir values, a progressive increase in the deviation of predicted frequency at t = 120s could be observed for all 300 MW events with decreasing system inertia levels and increasing RES penetration. For larger imbalance events corresponding to 500 MW, there is still a scope for improvement in the prediction results. A possible reason for worse results in the 500 MW scenarios could be change in power flows experienced by the system for a larger imbalance event. Since these changes could lead to slightly different behaviour in frequency response, a different offline-trained model that learns from a system with similar power flows could be expected to provide better prediction results.

Table 4.6: Predictive model definition - Summary

	Neurons in hidden layer	Activation function	Training-Validation -Test data split (in s)	Learning rates (Adam; BFGS)	Maximum iteration	Performance metrics	Is retraining required?
Test Case 1: Normal operation	30	Sigmoid	100 - 25 - 115	0.05; 0.1	200+100	MAE, Max. absolute error	No
Test Case 2: Restoration - Initial model	40	Sigmoid	1000 - 500 - 4000	0.01; 0.05	200+100	MAE, Max. absolute error	No
Test Case 2: Restoration - Retrained model	40	Sigmoid	450 - 100 - 1200	0.1; 0.05	10+100	MAE, Max. absolute error	Yes
Test Case 3: Nadir prediction	30	Sigmoid	3 - 1 - 21	0.001; N/A	10	Deviation of estimates: Nadir, T_{Nadir}	Yes
Test Case 3: Post-nadir restoration	30	Sigmoid	20 - 10 - 60	0.001; N/A	200	Deviation of estimates: Frequency at t = 120s	Yes

4.2. Comparison and Discussion

The prediction models for each test case differ from each other in their respective model definition in aspects like number of network layers, loss function definition, train-validation-test data split, data pre-processing and performance metrics. These differences are required to account for the changes in the nature of available data (noise levels, measurement locations, test system etc.) and changes in prediction requirements (required outputs, computational speed, performance indicators etc.) among different test cases. Table 4.6 summarises key model definition aspects for all the test cases. While the initial choice of having a single hidden layer and sigmoid activation continued to competently support the model performances, further tuning was achieved by changing the learning rates and max_iters values of the optimizers. It could be observed that even a small change in the optimizer settings from the values shown in Table 4.6 leads to a drop in model performance. On the other hand, this makes it easier to obtain the optimal set of tuned optimizer parameters. In test case 3, the very low learning rates in the optimizer enable the model to learn the new, local behaviour in a short number of iterations while avoiding over-fitting. The major differences in training-validation-test data split is a direct consequence of the changing test cases and the respective performance metrics have also been introduced accordingly.

The test cases also illustrate working with two types of input data that are quite different from each other. The PMU data is seldom smooth and indicates how all the power system quantities fluctuate and evolve in real-time. Consequently, NODE models require a higher number of iterations or more data to process out local, insignificant fluctuations in measured quantities. The PMU data has also been collected from three different substations, allowing for possibilities of a centralised security assessment. However, discussions with the industry during the course of the thesis suggest that it in the foreseeable future, it is more practically feasible to work with local grid measurements and control actions, as compared to a large centralised system of security assessment and control. With the synthetic data, features obtained correspond to measurements at a single location - Bus 39. Since all the critical dynamic power system behaviour are reflected at this point, localised measurements suffice for test case 3. Since frequency dynamics in a system are decided by factors spread across a large synchronous grid, local control actions might not help in mitigating large scale disturbances. In such situations, a centralised assessment and control system would be required to effectively address large scale frequency disturbances. However, in case of smaller area grids that could be quickly disconnected from the main synchronous grid, it is convenient to implement localised frequency assessment and control approaches.

Other changes among the models include small modifications in scaling, weights used in loss functions and data-preprocessing steps. The noise-free synthetic data requires almost no data-preprocessing. Scaling of data for models depends on the retraining method used. With models requiring no retraining involving a new starting point, the entire available training data can be scaled together close to the ranges of 0 to 1 and processed further. However, when the starting point is of significance, a standard scaling rate that could accommodate all possible drop in frequency values across different scenarios is required. Only then could the model make use of the starting point information to its advantage. As for the loss function, some test cases require an increase in the weights assigned to frequency data for the model to start converging over iterations. These weights often depend on the converging capabilities of the optimizer with respect to the available training data. Overall, tuning a few key aspects in predictive NODE models aids in the applicability of the model to a wide range of test cases and frequency disturbance scenarios.

5

Future Scope & Conclusion

Drawing on the results and learning from this thesis, this chapter provides answers to the questions raised in section 1.4 and elaborates on the steady progress towards the research objective stated as: "To use Neural Ordinary Differential Equations (NODE) for real-time frequency security assessment and subsequently enable timely frequency stability control." Additionally, areas for improvement and possible avenues for further research in using NODE for frequency dynamics studies are also discussed thereupon.

5.1. Research Questions

1. How can Neural Ordinary Differential Equations be adapted to frequency dynamics predictions?

Neural Ordinary Differential Equations are applied in scientific areas for capturing complex dynamic behaviour among different variables in any non-linear system. Since frequency response is a similar phenomenon governed by various power system quantities, frequency stability studies is a relevant area for applying NODE in. Frequency instabilities are often controlled after the detection of large RoCoFs and a nadir in the frequency response curve. Attempts have been made in the literature to predict post-disturbance frequency behaviour in advance so as to trigger stability control actions earlier than in conventional methods. Similarly, NODE provides an opportunity to learn frequency dynamics from historic patterns and power system simulations which could be further used to make real-time frequency predictions after a disturbance has been detected. With a good selection of parameters to predict like the frequency nadir or the settling time of the post-nadir frequency curve, it is possible to provide key information in advance to pre-existing frequency stability control mechanisms that are currently active in our power systems.

2. What are the challenges in obtaining relevant input data for training and testing NODE models?

Major frequency disturbance scenarios are sparsely available from real system data and hence, synthetically generated frequency events are required to provide NODE models with the necessary information about expected post-disturbance frequency response in power systems. However, the characteristics of real system data available from PMU and synthetically simulated data are very different from each other. There is no noise observed in synthetic data, which reduces the processing and curve-approximating time in NODE models by a large margin, while

simultaneously increasing the prediction capabilities of the model. Hence, to obtain realistic estimates of how well the model could perform in real world applications, it would be necessary to induce realistic noise characteristics in synthetic data. Working with real system data is also constrained by the limited number of locations where PMU data are available. These locations also have an impact in deciding the final set of features that best represent the changing dynamics in the observed area of the power system. Another significant aspect to considering while collecting relevant input data is that it might be practically infeasible to carry out security assessment for a large-scale synchronous grid in a centralised manner when there are indefinite number of factors influencing the system dynamics. In such cases, smaller areas of grid that could be disconnected from the main grid during major system disturbances seem to be more practically feasible test systems to work on.

3. Which aspects of the NODE algorithms need to be tuned to address different frequency security situations?
Certain aspects from preliminary predictive NODE models could be retained while approaching different test cases and corresponding frequency scenarios, whereas some key aspects addressed in section 4.2 need to be tuned to allow effective application of NODE for different frequency security situations. Factors like depth and width of neural networks, activation functions and choice of optimizers need not be changed across test cases. The initial set of parameters chosen for these factors provide enough flexibility to model frequency-related dynamic behaviour in power system quantities across different test cases. However, achieving the desired level of prediction performance in each test case is only possible when parameters like the learning rates and the number of iterations used in the optimizers and/or the duration of training data and the length of the prediction horizon are changed. With changing prediction requirements depending on the security situation detected (for instance, a large RoCoF suggesting an impending nadir or a nadir in the detected frequency curve suggesting an impending restoration), it is also required to change the performance metrics with respect to the set of quantities that need to be predicted. Since these parameters are easy to tune, it is possible to develop a generic NODE algorithm with tuneable parameters for frequency security assessment applications.

4. What are some possible real-world implications of the frequency prediction outcomes from NODE models?
A major motivation behind this thesis is that real-time predictive analysis of frequency during disturbances in the power system could aid in earlier onset of stability control actions to achieve a more controlled frequency response. While possible control actions after estimation of relevant frequency instability parameters have not been considered in this thesis, it has been shown that is is possible to obtain fast predictions of the expected frequency trajectory using NODE as compared to what is possible using conventional frequency security assessment methods. Any relevant information about the power system with respect to frequency dynamics could be used as an input for NODE models using appropriate data-processing steps. Hence, with more available PMU locations and higher access to abnormal frequency data, it is possible to recreate expected frequency response patterns when similar system disturbances occur in the future. This information could be useful in analysing the impact of possible disturbances due to, say, new developments in the power system. With rising RES levels and the corresponding expected drop in system inertia levels, the prospects of higher number of large scale frequency

disturbances also increases. Accordingly, the frequency stability and control methods present in the power system have to remain updated and competent in the future. Combining advanced frequency monitoring and assessment methods that could perform reliable predictive analysis on the status of a system would, hence, contribute to achieving state-of-art frequency stability and control for future electrical power systems.

5.2. Avenues for Further Research

Some limitations of NODE-based predictive analysis of frequency that could be observed from the results of this thesis are stated below:

- The frequency scenarios and respective power system data created synthetically to train NODE models are not good approximations of real-world dynamic phenomenon that occur in power system quantities (which consist of multiple local oscillations in data and/or noise) during frequency disturbances.
- The test system used for synthetic generation of frequency data is less complex, and thus easy to learn for the NODE model. However, more sophisticated power system models are required to close the gap between the differences in input received from simulations and real-world systems.
- The NODE models in this thesis do not consider any information concerning frequency control reserves that act as important frequency stability controlling elements in the power system during frequency recovery.
- The extent to which early estimation of frequency instability parameters like nadir and RoCoF can improve the frequency response curve is not considered in this thesis. The scope of using NODE-based prediction results to improve system stability can be evaluated by using them as inputs for standard frequency control methods and observing the change in frequency response.

Based on the limitations discussed, it is possible to suggest a few avenues (as listed below) wherein NODE-based frequency analysis could be further extended and applied to:

- Develop a synthetically generated data-set for training frequency predictive NODE models that is more realistic and close in noise characteristics with respect to real power system data.
- Model and simulate frequency events using a more sophisticated power system model that is representative of other stability-influencing technologies (like offshore connections, large-scale onshore RES generation, EV charging hubs etc.) being connected to the European grid in the recent times.
- Study further about prescribed stability and control actions taken in the synchronous grid of Continental Europe during major frequency disturbances, and incorporate information about significant frequency control reserves or available load shedding schemes into NODE-based prediction models.
- Check the feasibility of using frequency predictions for achieving improved stability control by combining the results from predictive NODE models with relevant frequency control schemes in appropriate test systems.
- Extend NODE to monitor and assess other dynamic phenomena in power systems to combine multiple stability studies for a more holistic power system security assessment.

References

[1] Jianzhong Tong and Lei Wang. “Design of a DSA Tool for Real Time System Operations”. In: *2006 International Conference on Power System Technology*. 2006, pp. 1–5. DOI: 10.1109/ICPST.2006.321419.

[2] Thi Ha Nguyen, Guangya Yang, Arne Nielsen, and Peter Jensen. “Challenges and Research Opportunities of Frequency Control in Low Inertia Systems”. In: *E3S Web of Conferences* 115 (Jan. 2019), p. 02001. DOI: 10.1051/e3sconf/201911502001.

[3] P. Kundur, J. Paserba, V. Ajjarapu, G. Andersson, A. Bose, C. Canizares, N. Hatziargyriou, D. Hill, A. Stankovic, C. Taylor, T. Van Cutsem, and V. Vittal. “Definition and classification of power system stability IEEE/CIGRE joint task force on stability terms and definitions”. In: *IEEE Transactions on Power Systems* 19.3 (2004), pp. 1387–1401. DOI: 10.1109/TPWRS.2004.825981.

[4] Hassan Haes Alhelou, Mohamad Esmail Hamedani-Golshan, Takawira Cuthbert Njenda, and Pierluigi Siano. “A Survey on Power System Blackout and Cascading Events: Research Motivations and Challenges”. In: *Energies* 12.4 (2019). ISSN: 1996-1073. DOI: 10.3390/en12040682. URL: https://www.mdpi.com/1996-1073/12/4/682.

[5] Lei Tang. “Dynamic security assessment processing system”. PhD thesis. Iowa State University, 2014. URL: https://dr.lib.iastate.edu/handle/20.500.12876/28101.

[6] Qi Wang, Feng Li, Yi Tang, and Yan Xu. “Integrating Model-Driven and Data-Driven Methods for Power System Frequency Stability Assessment and Control”. In: *IEEE Transactions on Power Systems* 34.6 (2019), pp. 4557–4568. DOI: 10.1109/TPWRS.2019.2919522.

[7] Junbo Zhao, Marcos Netto, Zhenyu Huang, Samson Shenglong Yu, Antonio Gómez-Expósito, Shaobu Wang, Innocent Kamwa, Shahrokh Akhlaghi, Lamine Mili, Vladimir Terzija, A. P. Sakis Meliopoulos, Bikash Pal, Abhinav Kumar Singh, Ali Abur, Tianshu Bi, and Alireza Rouhani. “Roles of Dynamic State Estimation in Power System Modeling, Monitoring and Operation”. In: *IEEE Transactions on Power Systems* 36.3 (2021), pp. 2462–2472. DOI: 10.1109/TPWRS.2020.3028047.

[8] Rasoul Azizipanah-Abarghooee, Mostafa Malekpour, Mario Paolone, and Vladimir Terzija. “A New Approach to the Online Estimation of the Loss of Generation Size in Power Systems”. In: *IEEE Transactions on Power Systems* 34.3 (2019), pp. 2103–2113. DOI: 10.1109/TPWRS.2018.2879542.

[9] Kaur Tuttelberg and Jako Kilter. “Predicting Frequency Disturbances from Wide Area Monitoring of Ambient Power System Dynamics”. In: *2018 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*. 2018, pp. 1–6. DOI: 10.1109/ISGTEurope.2018.8571711.

[10] Moein Abedini, Seyed-Alireza Ahmadi, and Majid Sanaye-Pasand. “A Straightforward and Robust Algorithm for Accurate Estimation of Power System Frequency”. In: *IEEE Transactions on Industrial Electronics* 68.12 (2021), pp. 12830–12839. DOI: 10.1109/TIE.2020.3044793.

[11] Harold R. Chamorro, Alvaro D. Orjuela-Cañón, David Ganger, Mattias Persson, Francisco Gonzalez-Longatt, Vijay K. Sood, and Wilmar Martinez. “Nadir Frequency Estimation in Low-Inertia Power Systems”. In: *2020 IEEE 29th International Symposium on Industrial Electronics (ISIE)*. 2020, pp. 918–922. DOI: 10.1109/ISIE45063.2020.9152296.

[12] Shuli Wen, Yu Wang, Yi Tang, Yan Xu, Pengfei Li, and Tianyang Zhao. “Real-Time Identification of Power Fluctuations Based on LSTM Recurrent Neural Network: A Case Study on Singapore Power System”. In: *IEEE Transactions on Industrial Informatics* 15.9 (2019), pp. 5266–5275. DOI: 10.1109/TII.2019.2910416.

[13] Laurine Duchesne, Efthymios Karangelos, and Louis Wehenkel. “Recent Developments in Machine Learning for Energy Systems Reliability Management”. In: *Proceedings of the IEEE* 108.9 (2020), pp. 1656–1676. DOI: 10.1109/JPROC.2020.2988715.

[14] Jian Xie, Inalvis Alvarez-Fernandez, and Wei Sun. “A Review of Machine Learning Applications in Power System Resilience”. In: *2020 IEEE Power Energy Society General Meeting (PESGM)*. 2020, pp. 1–5. DOI: 10.1109/PESGM41954.2020.9282137.

[15] Yize Chen, Yushi Tan, and Deepjyoti Deka. “Is Machine Learning in Power Systems Vulnerable?” In: *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. 2018, pp. 1–6. DOI: 10.1109/SmartGridComm.2018.8587547.

[16] Yan Xu. “A review of cyber security risks of power systems: from static to dynamic false data attacks”. In: *Protection and Control of Modern Power Systems* 5 (Sept. 2020), p. 19. DOI: 10.1186/s41601-020-00164-w.

[17] Silvia Canevese, Emanuele Ciapessoni, Antonio Gatti, and Marco Rossi. “Monitoring of frequency disturbances in the European continental power system”. In: *2016 AEIT International Annual Conference (AEIT)*. 2016, pp. 1–6. DOI: 10.23919/AEIT.2016.7892763.

[18] Ling Wu. “Power System Frequency Measurement Based Data Analytics and Situational Awareness”. PhD thesis. University of Tennessee, 2018. URL: https://trace.tennessee.edu/utk_graddiss/4897.

[19] Luis Badesa, Fei Teng, and Goran Strbac. “Conditions for Regional Frequency Stability in Power System Scheduling—Part I: Theory”. In: *IEEE Transactions on Power Systems* 36.6 (2021), pp. 5558–5566. DOI: 10.1109/TPWRS.2021.3073083.

[20] Luis Badesa, Fei Teng, and G. Strbac. “Conditions for Regional Frequency Stability in Power System Scheduling—Part II: Application to Unit Commitment”. In: *IEEE Transactions on Power Systems* PP (Apr. 2021). DOI: 10.1109/TPWRS.2021.3073077.

[21] Hassan Haes Alhelou, M.E.H. Golshan, R. Zamani, M. P. Moghaddam, Takawira C. Njenda, Pierluigi Siano, and Mousa Marzband. “An Improved UFLS Scheme based on Estimated Minimum Frequency and Power Deficit”. In: *2019 IEEE Milan PowerTech*. 2019, pp. 1–6. DOI: 10.1109/PTC.2019.8810497.

[22] Istvan Vokony. “Effect of inertia deficit on power system stability - synthetic inertia concepts analysis”. In: *2017 6th International Youth Conference on Energy (IYCE)*. 2017, pp. 1–6. DOI: 10.1109/IYCE.2017.8003725.

[23] Fei Teng, Marko Aunedi, Danny Pudjianto, and Goran Strbac. “Benefits of Demand-Side Response in Providing Frequency Response Service in the Future GB Power System”. In: *Frontiers in Energy Research* 3 (2015). ISSN: 2296-598X. DOI: 10.3389/fenrg.2015.00036. URL: https://www.frontiersin.org/articles/10.3389/fenrg.2015.00036.

[24] Mehrdad Tarafdar Hagh and Atabak Mashhadi Kashtiban. “Application of Neural Networks in Power Systems; A Review”. In: *Proceedings of World Academy of Science, Engineering and Technology* 6 (Jan. 2005).

[25] Yi Zhang, Xiaohan Shi, Hengxu Zhang, Yongji Cao, and Vladimir Terzija. “Review on deep learning applications in frequency analysis and control of modern power system”. In: *International Journal of Electrical Power Energy Systems* 136 (2022), p. 107744. ISSN: 0142-0615. DOI: https://doi.org/10.1016/j.ijepes.2021.107744. URL: https://www.sciencedirect.com/science/article/pii/S0142061521009686.

[26] Hsin-Wei Chiu and Le-Ren Chang-Chien. “A Supervised Learning Scheme for Evaluating Frequency Nadir and Fast Response Reserve in Ancillary Service Market”. In: *IEEE Access* 9 (2021), pp. 100934–100943. DOI: 10.1109/ACCESS.2021.3096962.

[27] Qingyue Chen, Xiaoru Wang, Jintian Lin, and Longyu Chen. “Convolutional LSTM-based Frequency Nadir Prediction”. In: *2021 4th International Conference on Energy, Electrical and Power Engineering (CEEPE)*. 2021, pp. 667–672. DOI: 10.1109/CEEPE51765.2021.9475832.

[28] Ogun Yurdakul, Fatih Eser, Fikret Sivrikaya, and Sahin Albayrak. “Very Short-Term Power System Frequency Forecasting”. In: *IEEE Access* 8 (2020), pp. 141234–141245. DOI: 10.1109/ACCESS.2020.3013165.

[29] Christopher Rackauckas, Yingbo Ma, Julius Martensen, Collin Warner, Kirill Zubov, Rohit Supekar, Dominic Skinner, and Ali Jasim Ramadhan. “Universal Differential Equations for Scientific Machine Learning”. In: *CoRR* abs/2001.04385 (2020). arXiv: 2001.04385. URL: https://arxiv.org/abs/2001.04385.

[30] Jared Willard, Xiaowei Jia, Shaoming Xu, Michael Steinbach, and Vipin Kumar. *Integrating Scientific Knowledge with Machine Learning for Engineering and Environmental Systems*. 2022. arXiv: 2003.04919 [physics.comp-ph].

[31] Happiness Ugochi Dike, Yimin Zhou, Kranthi Kumar Deveerasetty, and Qingtian Wu. “Unsupervised Learning Based On Artificial Neural Network: A Review”. In: *2018 IEEE International Conference on Cyborg and Bionic Systems (CBS)*. 2018, pp. 322–327. DOI: 10.1109/CBS.2018.8612259.

[32] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. “Neural Ordinary Differential Equations”. In: *CoRR* abs/1806.07366 (2018). arXiv: 1806.07366. URL: http://arxiv.org/abs/1806.07366.

[33] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV].

[34] L.R. ten Klooster. *Approximating differential equations using neural ODEs*. July 2021. URL: http://essay.utwente.nl/87568/.

[35] Xianghao Kong, Koji Yamashita, Brandon Foggo, and Nanpeng Yu. "Dynamic Parameter Estimation with Physics-based Neural Ordinary Differential Equations". In: *2022 IEEE Power & Energy Society General Meeting (PESGM)*. 2022, pp. 1–5. DOI: 10.1109/PESGM48719.2022.9916840.

[36] Jennifer Brucker, René Behmann, Wolfgang G. Bessler, and Rainer Gasper. "Neural Ordinary Differential Equations for Grey-Box Modelling of Lithium-Ion Batteries on the Basis of an Equivalent Circuit Model". In: *Energies* 15.7 (2022). ISSN: 1996-1073. DOI: 10.3390/en15072661. URL: https://www.mdpi.com/1996-1073/15/7/2661.

[37] Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. *Augmented Neural ODEs*. 2019. arXiv: 1904.01681 [stat.ML].

[38] Juntang Zhuang, Nicha Dvornek, Xiaoxiao Li, Sekhar Tatikonda, Xenophon Papademetris, and James Duncan. *Adaptive Checkpoint Adjoint Method for Gradient Estimation in Neural ODE*. 2020. arXiv: 2006.02493 [stat.ML].

[39] Jared Quincy Davis, Krzysztof Choromanski, Jake Varley, Honglak Lee, Jean-Jacques Slotine, Valerii Likhosterov, Adrian Weller, Ameesh Makadia, and Vikas Sindhwani. *Time Dependence in Non-Autonomous Neural ODEs*. 2020. arXiv: 2005.01906 [cs.LG].

[40] Ashish Shrestha and Francisco Gonzalez-Longatt. "Frequency Stability Issues and Research Opportunities in Converter Dominated Power System". In: *Energies* 14.14 (2021). ISSN: 1996-1073. DOI: 10.3390/en14144184. URL: https://www.mdpi.com/1996-1073/14/14/4184.

[41] Robert Eriksson, Niklas Modig, and Katherine Elkington. "Synthetic inertia versus fast frequency response: a definition". In: *IET Renewable Power Generation* 12.5 (2018), pp. 507–514. DOI: https://doi.org/10.1049/iet-rpg.2017.0370. eprint: https://ietresearch.onlinelibrary.wiley.com/doi/pdf/10.1049/iet-rpg.2017.0370. URL: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/iet-rpg.2017.0370.