PHYSICS INFORMED NEURAL NET MODEL FOR THE PARAMETER ESTIMATION OF EV CHARGER

by

Vinay Kiran

to obtain the degree of Master of Science in Electrical Engineering
at the Delft University of Technology
to be defended publicly on Thursday, September 22, 2022 at 12:30 PM.

Student number:	5052149		
Thesis committee:	Prof. dr. ir. Pavol Bauer	Chair	DCE&S, TU Delft
	Dr. Zian Qin	Supervisor	DCE&S, TU Delft
	Dr. Pedro P. Vergara	Committee Member	IEPG, TU Delft

An electronic version of this thesis is available at
http://repository.tudelft.nl/

It does not matter how slowly you go,
as long as you do not stop.

Confucius

CONTENTS

List of Figures vii

List of Tables ix

Abstract xi

Acknowledgements xiii

List of Abbreviations xv

1 Theoretical Framework 1

1.1 Overview of FCS . . . 1

1.1.1 Impact of FCS on Grid . . . 1

1.1.2 Power Quality Issues from FCS . . . 2

1.1.3 Architecture of FCS . . . 2

1.2 Modelling of FCS . . . 4

1.2.1 Current Harmonic Analysis . . . 4

1.2.2 Impedance Modelling of AFE . . . 5

1.3 Existing Methodologies for Parameter Estimation. . . . 7

1.3.1 Invasive Methodology . . . 7

1.3.2 Non-Invasive Methodology . . . 7

1.4 Introduction to Physics Informed Neural Net . . . 8

1.5 Research Objective . . . 9

1.6 Thesis Structure. . . . 10

2 Methodology 11

2.1 Modelling of Physics Informed Neural Net . . . 13

2.1.1 General Equations of PINN Model . . . 13

2.1.2 Formulation of Latent State . . . 13

2.2 Discretization of the Model 14
2.3 Coupling of Neural Net and Physics Model 14
2.4 Programming Algorithm of PINN 16
2.4.1 Data Generation 16
2.4.2 Data Preprocessing 17
2.4.3 Initializing the PINN 17
2.4.4 Loss Function and Hyper-parameter of the Optimiser 19
2.4.5 Guidelines to Train a PINN 19

3 Parameter Estimation of Boost Converter 21

3.1 Dynamic Model of Boost Converter 21
3.2 PINN Modelling of Boost Converter 22
3.3 Discretization of Boost Converter and Loss Function 23
3.4 Data Acquisition and Configuration of PINN for the Boost Converter 23
3.4.1 Data Acquisition 23
3.4.2 Defining the Structure of PINN. 24
3.4.3 Validation and Optimisation of the PINN for Boost Converter 25
3.5 Results of PINN for Boost Converter 31

4 Parameter Estimation of Active Front End Converter 35

4.1 Dynamic Modelling of the AFE converter 35
4.2 PINN Modelling of AFE Converter 38
4.3 Discretization of AFE Converter and Loss Function 39
4.4 Data Acquisition and Configuration of PINN for the AFE Converter. 41
4.4.1 Data Acquisition 41
4.4.2 Defining the Structure of PINN. 41
4.5 Validation and Optimisation of the PINN for the AFE Converter 42
4.6 Results of PINN Model for the AFE Converter 50

5 Conclusion **53**

5.1 Revisiting the research question 53

5.2 Discussion 54

5.3 Future Scope of Work 55

List of Figures

1.1 Outlook for annual global passenger-car and light-duty vehicle sales [4] . 2

1.2 Structure of FCS connected to AC grid [6] 3

1.3 Impedance model of FCS connected to the grid 4

1.4 Calculation of input impedance of AFE in dq or sequence domain. [5] . . . 5

1.5 Overall outline of the thesis 10

2.1 Comparison of conventional Neural Network architecture and Physics Informed Neural Net 12

2.2 Discretization of the PINN model 14

2.3 Coupling of observable points and latent state points 15

2.4 Flow chart of the program flow to implement PINN 16

3.1 Circuit topology of Boost Converter 22

3.2 Architecture of PINN applied for the Boost Converter 24

3.3 Validation of PINN model for the Boost Converter by setting the physics parameters L, C, R_{load} and V_{in} to their true value and it is set as non-trainable in the PINN model 25

3.4 Variation of error percentage of estimated parameters showing non-convergence of the optimisation process for Boost Converter when learning rate is set as (a) $8 \cdot 10^{-6}$, (b) $1.6 \cdot 10^{-5}$ and (c) $3.1 \cdot 10^{-5}$ for the Adams optimiser 27

3.5 Variation of error percentage of estimated parameters showing convergence of the optimisation process for Boost Converter when learning rate is set as (a) $6.3 \cdot 10^{-5}$, (b) $1.25 \cdot 10^{-4}$, (c) $2.5 \cdot 10^{-4}$ and (d) $1 \cdot 10^{-3}$ for the Adams optimiser 28

3.6 PINN parameter estimation of the boost converter at steady state 32

3.7 Current and Voltage waveform prediction for backward and forward dataset at steady state 32

3.8 PINN parameter estimation of the boost converter at transient state 33

3.9 Current and Voltage waveform prediction for backward and forward dataset at transient state 33

4.1 Circuit topology of the 2-level VSR 36

4.2 PLECS Simulation of AFE 36

4.3 3-phase Sine PWM for 1 cycle of the sine wave 37

4.4 Sub-interval diagram of 3-phase Sine PWM used for AFE 37

4.5 Architecture of PINN applied for the AFE Converter 41

4.6 Case:1a Performance of AFE PINN model with parameters set as non-trainable and initialised with true values | Adams Learning Rate = 0.001 42

4.7 Case:1b Performance of AFE PINN model with parameters set as non-trainable and initialised with true values | Adams Learning Rate = 0.002 43

4.8 Case:1C Performance of AFE PINN model with parameters set as non-trainable and initialised with true values | Adams Learning Rate = 0.004 44

4.9 Case:1C Performance of AFE PINN model with parameters set as non-trainable and initialised with true values | Adams Learning Rate = 0.005 44

4.10 Case:1d Convergence of the loss value with physics parameters set as non-trainable and initialised with true values | Adams Learning Rate = 0.003 . . 45

4.11 Case:1d Convergence of the loss value with parameters set as non-trainable and initialised with true values | Adams Learning Rate = 0.003 46

4.12 Case:1d Performance of AFE PINN model with parameters set as non-trainable and initialised with true values | Adams Learning Rate = 0.003 46

4.13 PINN parameter estimation error for the AFE converter 47

4.14 Test Case 1: AFE PINN model to estimate only La 48

4.15 Test Case 2: AFE PINN model to estimate only La and Lb 49

4.16 Current and Voltage waveform prediction for backward and forward dataset of AFE converter with 1 Hidden Layer and 50 Neurons 50

4.17 PINN parameter estimation of the AFE converter with 1 Hidden Layer and 50 Neurons 51

4.18 PINN parameter estimation of the AFE converter with 5 Hidden Layer and 200 Neurons 52

LIST OF TABLES

1.1 Comparison of AFE topologies 4
1.2 Procedure to estimate input impedance of FCS in dq and sequence domain 6

2.1 Generalized form of data collection and preprocessing 17
2.2 Parameters defining basic format floating-point numbers [28] 18

3.1 Specification of the Boost Converter 21
3.2 Pseudo Normalisation of Physics Parameters for Boost Converter 26
3.3 Percentage error of estimated parameters of the boost converter 29
3.4 Number of iterations required for convergence 29
3.5 Performance evaluation for Hidden Layer = 1 30
3.6 Performance evaluation for Hidden Layer = 2 30
3.7 PINN results of boost converter for steady state and transient state 31

4.1 Specification of the 2-level VSC Converter as the AFE converter 36
4.2 Pseudo Normalisation of Physics Parameters for AFE Converter 47
4.3 Parameter Estimation of AFE Converter 50

ABSTRACT

The impedance-based approach is promising to analyse the harmonic emission and stability of a converter-based system, e.g., an EV fast charging station. When extracting the accurate impedance model of an EV charger, the knowledge of the charger's circuit and controller parameters is indispensable. However, EV chargers' manufacturers do not disclose the design parameters of chargers since they are confidential designs. Therefore, developing an approach to estimate the charger's design parameters is practically beneficial for establishing the charger's impedance model and thereby analysing the harmonic and stability of the charger-grid system.

Recent works on physics-informed machine learning provide a promising data-efficient approach to estimating unknown parameters of a system with a known physical model. However, to the author's knowledge, such a technique has not been explored extensively in the power electronics domain. Thus, exploring whether PINN is also suitable for the parameter estimation of power electronic converters and how to implement this technique is worthwhile. In this thesis, to begin with, a Physics Informed Neural Net model is developed as a proof of concept to estimate the parameters of the boost converter. And subsequently applied to the AFE converter of the Fast-Charging Station to explore the scalability and generalisation of the model developed.

Implementing the physics model within the neural net is still an open problem; hence the physics parameters are estimated similarly to the weights and biases during the training process using the existing optimiser. This approach requires diving deeper into the development of the neural net, and hence instead of eager execution, TensorFlow's computational graph method is used to build the neural net and the physics model. This allows complete flexibility to build the network from scratch with added complexity.

In the PINN model developed for the boost converter, the mean estimation error was 0.89%, and the parameters were estimated in 25.76 seconds. Because of the complexity of the physics model, optimizing the PINN model was challenging for the AFE converter. With a mean estimation error of 30.25%, the PINN model for the AFE converter took 361 seconds to execute. For both models, less than 150 data points collected over 25 switching cycles peak-to-peak values are used. The proposed PINN model developed in this thesis thus proves to be highly data-efficient with quick convergence. Despite the success of the developed PINN model, several drawbacks of PINN-based parameter estimation are also noticed and summarised.

ACKNOWLEDGEMENTS

The journey of my thesis has been a roller coaster ride. I got to see the toughest point of my life, and it also showed me how few wonderful people could shower positivity, inspire you and help you keep growing. There is no way I would have reached this far without the support of these people.

First of all, I would like to express my deepest gratitude to my daily supervisor Lu Wang. Right from the time of the extra project to the whole journey of my thesis, he has always been my mentor and supported me. Every single time I hit a dead end, Lu was always there as a guru to guide me on the way out.

I would like to extend my heartfelt gratitude to my supervisor, Dr. Zian Qin, who inspired me to take on this challenging topic in the first place. I still remember the day when I was unsure about the journey of TU Delf; a mail from Dr. Zian changed the course of my life. Through him, I have learned to get the most out of myself and been motivated to take on challenging tasks and never give up.

Also, I wish to thank Prof. Dr. Pavol Bauer and Dr. Pedro P. Vergara for being on my graduation committee and helping evaluate my thesis.

My master's journey would not have been possible without the support of my parents, my fiancée, and my friends back in India and here in Delft. I am extremely grateful for my uncle and aunt, who have been my backbone throughout my life.

This journey cannot end without mentioning the life in Cojo and the wonderful people I met there.

Vinay Kiran
Delft, September 2022

List of Abbreviations

ICE Internal Combustion Engine

EV Electric Vehicle

FCS Fast Charging Station

PQ Power Quality

PCC Point of Common Coupling

LV Low Voltage

MV Medium Voltage

AFE Active Front End

VSC Voltage Source Converter

AI Artificial Intelligence

ML Machine Learning

DL Deep Learning

NN Neural Net

PINN Physics Informed Neural Net

VSR Voltage Source Rectifier

ODE Ordinary Differential Equation

PDE Partial Differential Equation

1

THEORETICAL FRAMEWORK

In the distribution grid, Power Quality (PQ) issues have been rising due to the increasing number of power electronics-based systems connected to the grid. These issues can be and are not limited to voltage fluctuation and harmonic emissions. With the rise in Electric Vehicle (EV) and Fast Charging Station (FCS), it is critical to address these issues to have a reliable distribution grid. The literature review during this thesis is divided into four parts as follows.

FCS Overview: Study on FCS architecture and its scale of impact on the grid.

FCS Modelling: Explore existing methodologies which are applied in analysis of PQ issues caused by FCS.

Parameter estimation methodologies: Study of available methodologies and identification of the short comes.

Research Gap: Identification of the research gap and formulate research objective.

1.1. OVERVIEW OF FCS

1.1.1. IMPACT OF FCS ON GRID

Internal Combustion Engine (ICE) still remains a major contributor to greenhouse gas emissions, which makes the transportation sector a major threat to climate change[1]. The Announced Pledges Scenario report from a survey done by International Energy Association shows that the commitments made by various governments across the world to reach net-zero emissions will be possible by 2050[2].

To achieve this, stringent policies are implemented worldwide to reduce emissions and incentives for the development and purchase of EV[3]. Efforts from governments, coupled with the evolution of consumer attitude towards EV, have made it possible to see the trend to shift from ICE to EV. Forecast from Deloitte analysis in Figure 1.1, it is observed that by 2030 EV share concerning total vehicles on the road will be close to 25%[4].

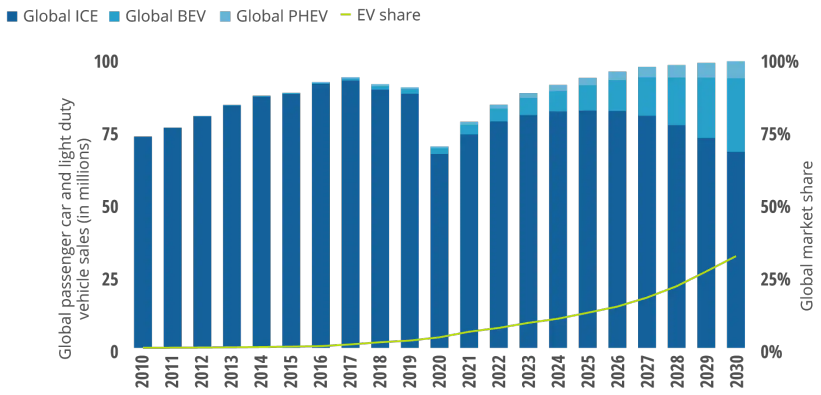


Figure 1.1: Outlook for annual global passenger-car and light-duty vehicle sales [4]

1.1.2. POWER QUALITY ISSUES FROM FCS

Efforts are made to make EV adoption more practical and comparable to that of ICE. One is to increase the number of FCS installations across the road network to ease long-distance travel. And the other is to increase the power rating of FCS to reduce charging time. The charging at FCS generally happens in the daytime and for a shorter duration. Since the load is centralized and requires higher power to deliver in a short period of time, FCS acts as a highly pulsating load to the distribution system[5].

Due to their higher power demands, FCS generally connects to the Medium Voltage (MV) grid. Lu et al. [5] describes the PQ issue of FCS in terms of voltage fluctuations, harmonic stability, and harmonic emission. This analysis of the PQ impact is limited to inside FCS and at the Point of Common Coupling (PCC) of the grid & FCS.

1.1.3. ARCHITECTURE OF FCS

FCS can be constructed in various ways depending on the manufacturer. A single unit or multiple parallel modules can be connected to the distribution grid. An AC distribution grid or a DC distribution grid can be used as a distribution grid. Currently, most FCS are modular parallel units connected to the MV grid via MV/Low Voltage (LV) low-frequency transformers, as shown in Figure 1.2. As described below, each FCS unit is composed of two power conversion stages:

First Stage: An AC-DC converter is used to step up the voltage and for active power factor correction. It is generally called as Active Front End (AFE) converter.

Second Stage: A DC-DC converter connected to the EV.

DC link capacitor is used to maintain stable DC voltage for the DC-DC conversion stage. Additionally, the DC-link capacitor decouples the DC-DC stage from the AC-DC stage, resulting in a scenario where harmonic current emission is largely determined by the AC-DC stage(AFE).

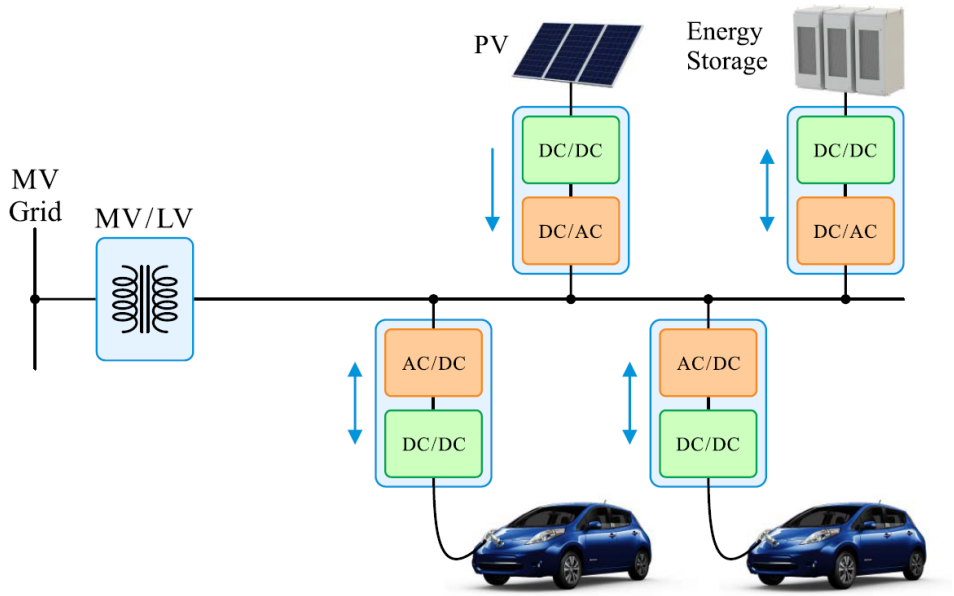


Figure 1.2: Structure of FCS connected to AC grid [6]

It is therefore sufficient to model the FCS's AFE to analyze harmonic emission. To generalize the AFE model, it is necessary to understand the various topologies of converters used. For AFE, the author in [5] explores three topologies: Vienna rectifier, 2-Level Voltage Source Rectifier, and Multi-Pulse Rectifier. Table 1.1 compares the topologies. Multi-pulse rectifiers are not preferred due to their high harmonic emission, which necessitates a larger filter inductor. The Vienna rectifier and 2-Level Voltage Source Rectifier (VSR) are currently used as mainstream AFE. Unlike a 2-Level VSR, the Vienna rectifier requires mid-point voltage balance control. This difference may be ignored if zero-sequence impedance analysis is not of interest, and a general Voltage Source Converter (VSC) modelling approach can be employed for AFE as well.

1

Table 1.1: Comparison of AFE topologies

AFE Topology	Advantage	Disadvantage
Vienna Rectifier	• 3 Voltage levels • Lesser input filter inductance • Reduced voltage stress on switches • Higher efficiency and power density	• Uni-directional power flow
2-level Voltage Source Rectifier	• Bi-directional power flow • Promising application for V2G functionality	• Lower power density due to 2-level switching
Multi-Pulse Rectifier	• Simple to design and implement	• Higher harmonic emission

1.2. MODELLING OF FCS

1.2.1. CURRENT HARMONIC ANALYSIS

Small signal impedance model is generally employed for harmonic analysis and stability studies [7]- [8]. In this approach, AFE is modelled as a harmonic current source(I_e) in parallel with converter impedanceZ_c. And, the grid is modelled as a background voltage source (V_g) in series with grid impedance (Z_g). Figure 1.3 shows the impedance model.

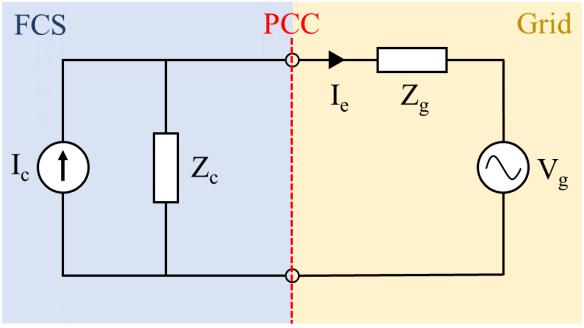


Figure 1.3: Impedance model of FCS connected to the grid

The current harmonic emission from FCS can be calculated via equation 1.1 [5]. The first term in the right-hand side of the equation 1.1 provides the harmonic current emission contribution from AFE. And, the second term provides the contribution from the

distorted background grid voltage.

$$I_e(s) = \frac{Z_c(s)I_c(s)}{Z_c(s)+Z_g(s)} - \frac{V_g(s)}{Z_c(s)+Z_g(s)} \tag{1.1}$$

1.2.2. IMPEDANCE MODELLING OF AFE

To estimate current harmonic emission from equation 1.1, it is required to model the input impedance of the AFE converter. It can be achieved via a numerical or analytical approach. In the numerical method approach, the switching model of the AFE converter is developed using a simulation platform. Then voltage perturbation at varied frequencies (f_h) is applied at the input terminal of FCS and its impact on the respective currents are observed. Since AFE is connected to a 3-phase grid, this can be done either in the dq-domain or in the sequence domain as depicted in figure 1.4.

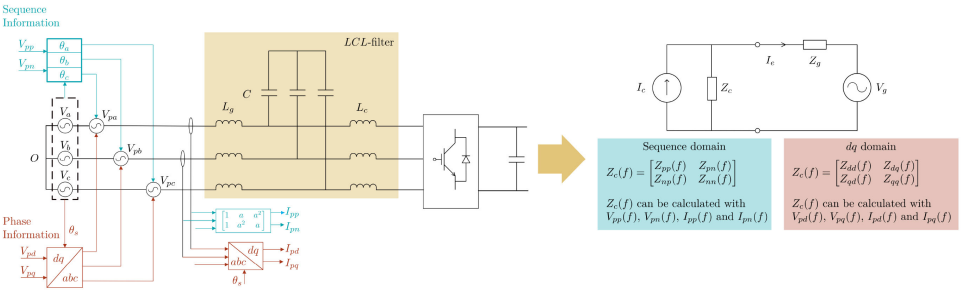


Figure 1.4: Calculation of input impedance of AFE in dq or sequence domain. [5]

Estimation of input impedance of AFE based on dq domain is shown in the table 1.2. Where $V_{pd}(f_h)$ and $V_{pq}(f_h)$ are direct and quadrature axis injected voltage perturbation. $I_{pd}(f_h)$ and $I_{pq}(f_h)$ are measured direct axis and quadrature axis current at respective perturbation frequency f_h.

$$\begin{aligned} z_c(f_h) &= \begin{bmatrix} z_{dd}(f_h) & z_{dq}(f_h) \\ z_{qd}(f_h) & z_{qq}(f_h) \end{bmatrix} \\ &= \begin{bmatrix} V_{pd}(f_h) & 0 \\ 0 & V_{pq}(f_h) \end{bmatrix} \begin{bmatrix} I_{pd}(f_h) & I_{pd}(f_h)^* \\ I_{pq}(f_h) & I_{pq}(f_h)^* \end{bmatrix}^{-1} \end{aligned} \tag{1.2}$$

Similarly, estimation of input impedance of FCS based on sequence domain is shown in the table 1.2. Where $V_{pp}(f_h)$ and $V_{pn}(f_h)$ are positive sequence and negative injected voltage perturbation. $I_{pp}(f_h)$ and $I_{pn}(f_h)$ are measured positive and negative sequence current at respective perturbation frequency f_h. $I_{pc}(2f_1 - f_h)$ and $I_{nc}(2f_1 + f_h)$ are frequency coupling component of positive and negative sequence currents respectively at

fundamental frequency f_1 and perturbation frequency f_h.

$$\begin{aligned} z_c(f_h) &= \begin{bmatrix} z_{pp}(f_h) & z_{pc}(2f_1 - f_h) \\ z_{nc}(2f_1 + f_h) & z_{nn}(f_h) \end{bmatrix} \\ &= \begin{bmatrix} V_{pp}(f_h) & 0 \\ 0 & V_{np}(f_h) \end{bmatrix} \begin{bmatrix} I_{pp}(f_h) & I_{nc}(2f_1 + f_h) \\ I_{pc}(2f_1 - f_h) & I_{pn}(f_h) \end{bmatrix}^{-1} \end{aligned} \tag{1.3}$$

Table 1.2: Procedure to estimate input impedance of FCS in dq and sequence domain

	Input Injected	**Output Measured**
Step: 1	Inject d-axis perturbation voltage $V_{pd}(f_h)$ with $V_{pq}(f_h) = 0$	$I_{pd}(f_h)$ and $I_{pq}(f_h)$
Step: 2	Inject q-axis perturbation voltage $V_{pq}(f_h)$ with $V_{pd}(f_h) = 0$	$I_{pd}(f_h)^*$ and $I_{pq}(f_h)^*$
Step: 3	Estimate Z_c at f_h based on equation 1.3	

(a) dq domain

	Input Injected	**Output Measured**
Step: 1	Inject positive sequence voltage $V_{pp}(f_h)$	$I_{pp}(f_h)$ and $I_{pc}(2f_1 - f_n)$
Step: 2	Inject positive sequence voltage $V_{np}(f_h)$	$I_{np}(f_h)$ and $I_{nc}(2f_1 + f_n)$
Step: 3	Estimate Z_c at f_h based on equation 1.4	

(b) Sequence domain

Even to begin with this approach, simulation of the switching model of AFE the converter is required. Which assumes that the converter parameters are available. Similarly, converter parameters are needed for any given analytical approach to solving the equation, irrespective of the method. Due to the design's confidentiality, manufacturers of FCS generally do not disclose the design parameters. Referring to datasheets of FCS solutions from various manufacturers such as Tesla, Phihong, ABB, Siemens, ChargePoint, and Hyundai, it is observed that FCS parameters of filter inductor and DC link capacitor are not available. Motivations for parameter estimation are as follows:

- Scheduled parameter estimation can provide an indication of the health condition of the FCS and can be used for preventive maintenance [9].
- Parameters of chargers are needed to develop simulation or impedance models of chargers for the stability or power quality study. However, they are generally confidential information hidden by the manufacturers.

1.3. EXISTING METHODOLOGIES FOR PARAMETER ESTIMATION

1.3.1. INVASIVE METHODOLOGY

Parameter estimation can be broadly classified as invasive and non-invasive methods. Invasive methods require intervention to the existing model and might require additional external hardware. For instance, in [10], a small signal model is used to estimate the impedance of the circuit. This approach requires external line-to-line current injection at a particular frequency, and the impedance is estimated at that frequency. Similarly, the approach proposed in [11], [12], [13] and [14] requires external signal addition.

1.3.2. NON-INVASIVE METHODOLOGY

Non-invasive methods are more practical, and the data can be acquired with an available measurement interface. Data-driven solution such as machine learning and deep learning methods are generally employed for this approach [15], [16], [17] and [17].

Shuai et al. [18] provides an overview of data-driven models based on Artificial Intelligence (AI), and these methodologies have a common approach. Firstly, data is generated via simulation software which renders multiple example data set. The model is trained to approximate the relationship between the example dataset and the desired output by utilizing available Machine Learning (ML) or Deep Learning (DL) learning tools. Once the model is trained, experimental data is used to verify the model.

Here is the problem with the pure data-driven approach for the power electronics domain. Generally, power electronics converters are mission-specific designs, which means there would be an extremely high number of possible combinations of parameters depending on application profiles and the topology of the converter. A given converter can operate at different operating conditions, and measurements must contain most of this combination in the training data set. In a data-driven solution, data sets are distributed as training, validation and test data sets with identical data distribution. Considering this for the power electronics domain, with limited data set for training, results for generalization of the solution for a complete unseen data will be poor[19]. The data required to achieve a generalized solution for a given topology would be high in a purely data-driven approach.

Unlike image classification or natural language processing with existing database repositories like MNIST, EMNIST, and ImageNet containing billions of example datasets [20], the power electronics domain doesn't have such an existing data set. Say, for instance, if one has to develop a model for image classification using the available vast data repository, the development can be done. All the efforts are translated into the development of the model.

Whereas in the power electronics domain, one must generate the data first. Apart from time, it costs money and infrastructure to run practical tests to collect data if the simulation is impossible. It makes more sense to develop a model which is light on data requirements.

1

1.4. INTRODUCTION TO PHYSICS INFORMED NEURAL NET

In 2017, Raissi et al. [21] - [22] introduced the concept of Physics Informed Neural Net (PINN). The basic concept of PINN is to combine any know domain knowledge or physical laws in conjunction with a feedforward neural net. The known physical laws are generally expressed as Ordinary Differential Equation (ODE) or Partial Differential Equation (PDE). These equations are used as a hard constraint to reduce the space of possible solutions during the training process, drastically reducing the required number of data points. The hard constraint can be applied in two ways:

- Directly in NN architecture - which still remains an open problem [23].
- Indirectly during optimization of the neural net.

For parameter estimation, the data-driven discovery approach proposed by Raissi et al. [21] uses an indirect method wherein the parameters are estimated by the optimizer. The PINN approach is data-efficient and is specifically designed to integrate equations in differential form. This makes it a promising approach for parameter estimation in the power electronics domain.

Extensive literature survey during the thesis showed that not much research was carried out in the application of PINN in the power electronics domain. The first work to be published was by Shuai et al.[24] on 20 May 2022, which gave a major breakthrough for this thesis. They propose a non-invasive PINN approach to estimate the parameters of a buck converter, based on the work of Raissi et al. [21]. They focus on implementing PINN to a buck converter and assess the robustness of the model by comparing it with data from experiments. Being a pilot work in the domain, it is an impressive work yet has a few areas which can be addressed further.

As said before, the implementation of PINN is majorly an optimization task, as the optimizer of the model estimates the parameters. Observing and tuning the optimiser parameters is critical for quicker solution convergence. Below are the few observations which were observed during the replication of the work of [24] :

- The hyperparameters of the optimizers used in [24] are set to default values, and no study was performed to observe their behaviour.
- Two different optimizers (Adams and L-BFGS) are used, and the reason for such a framework is not established in the work.
- Since the circuit parameters are unknown, it is required to initialize it randomly in the algorithm. However, constant value initialization is used in work.
- Authors claim that there is no limit on the number of parameters to be estimated, and hence the method is completely scalable. However, Observing the optimiser's behaviour is required before establishing the scalability. In addition, scalability can be questioned in a scenario where multiple equations with cross-coupling parameters are used.

1.5. RESEARCH OBJECTIVE

This thesis aims to develop an approach to estimating the EV charger's design parameters for power quality study with highly data-efficient machine learning techniques.

Based on a preliminary literature study, physics-informed machine learning is a promising candidate to be applied to the parameter estimation task. Compared with the other ML-based approaches, it requires less data for the training.

On top of the preliminary study, the thesis aims to answer the following questions:

1. How to apply the PINN-based approach to the parameter estimation of a power electronic converter?
2. How to implement the PINN-based estimation approach?
3. What are the challenges when applying PINN and how to address them?

1

1.6. THESIS STRUCTURE

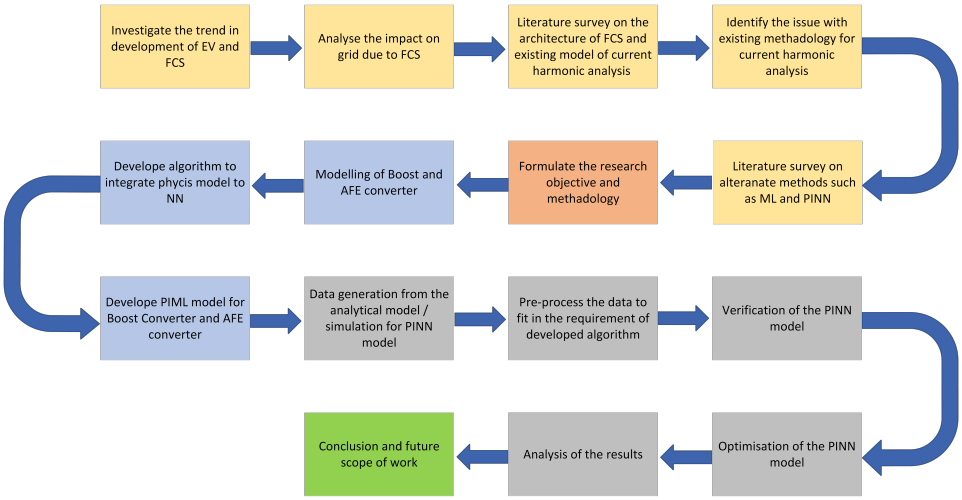


Figure 1.5: Overall outline of the thesis

The chapters of this thesis are summarised here:

- **Chapter 1:** provides the introduction for the thesis topic, motivation for the research questions formed, and lays a theoretical foundation for the thesis.
- **Chapter 2:** gives the detailed general mathematical derivation of PINN and outlines the guidelines to develop and implement PINN for power electronics applications.
- **Chapter 3:** PINN model is developed and implemented for a boost converter and the results are presented.
- **Chapter 4:** PINN model is developed and implemented for an AFE converter. The challenges faced during the implementation and possible solutions of PINN for a more complex converter are presented.
- **Chapter 5:** Results of PINN for boost and AFE converters are discussed. Research questions are revisited, and the thesis is summarised. Limitations and possible future work are also discussed.

2

METHODOLOGY

In a supervised deep learning task, a set of example data sets (X) are used to train a Neural Net (NN) by comparing the predictions (Y') to a targeted set of required true values(Y). During the training iterations, the prediction and required outputs are compared by a loss function, which generates a loss value. This loss value is used by the optimizer to change the weights and biases of the neural network to reduce the loss value to the minimum.

One of the approaches for PINN is to utilize the NN to predict a latent stage of values, which is used as an input to the physics model to correlate to the required true values. Now, the loss value is generated by the physics model, and the mathematical equations within the physics model act as hard constraint or regularization agent which binds the prediction values to a smaller space of the required solution. The major advantage is that PINN solutions have lower prediction errors even for limited data set points [25].

It is required to understand the basic structure of the PINN applied before the general equations are formulated. Figure 2.1 shows the basic structure of the both conventional NN and PINN. The major difference between conventional NN and PINN is that optimizer has to estimate the hyperparameter (weights and biases) and the physics model's parameters. Adding to it, the loss value is fed to the optimiser from the physics. In the next section, the approach to seamlessly integrating this is discussed.

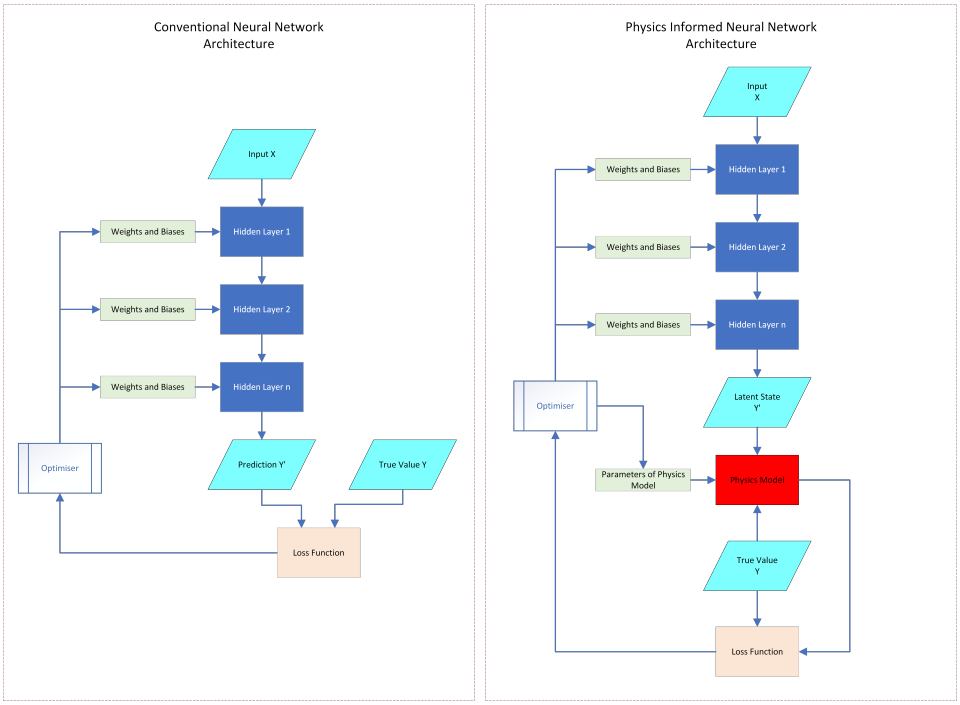


Figure 2.1: Comparison of conventional Neural Network architecture and Physics Informed Neural Net

2.1. MODELLING OF PHYSICS INFORMED NEURAL NET

2.1.1. GENERAL EQUATIONS OF PINN MODEL

At first, based on [21], the general form of equations will be derived and later will be extrapolated to the boost and AFE converters. Differential equations of any given dynamic system can be expressed in the general form of non-linear PDE equations as below:

$$u_t + \mathcal{N}[u;\lambda] = 0, x \in \Omega, t \in [0,T] \tag{2.1}$$

where:

$u_t = u(x,t)$ is the solution of the differential equation.

λ is the system parameter set.

$\mathcal{N}[\cdot]$ is the nonlinear differential operator.

x defines the space coordinates bounded by Ω.

Ω is the subset of the euclidean space $\mathbb{R}^D$.

t is the time co-ordinate bounded between 0 to T.

The equations which describe the dynamics of boost and AFE are generally ODE, henceforth, to simplify the mathematics, space coordinates x can be dropped in the following derivations. Once the space coordinates are dropped, $u(x,t)$ is defined as either current or voltage function with respect to time. λ is defined as the parameters of interest to be estimated, such as inductance, resistance or capacitance. $\mathcal{N}[u;\lambda]$ has been framed as the ordinary differential equation of current and voltages with respect to time containing the parameters of interest.

Here is the advantage while formulating the ODE, the equations need not encapsulate all the physics of the system. The equations are not used to provide accurate solutions; rather, it is used as hard constraint while function approximation of neural net. This has to offer, for example, in a boost converter, if the forward voltage drop of the diode or equivalent resistance of the capacitor is not considered, the performance of PINN is not compromised.

To elaborate, the dynamics of the solution are preserved in the data collected via simulation or experiments. A neural net as a universal functional approximator will correlate the data's inputs and outputs, while the physics model helps constrain the approximation.

2.1.2. FORMULATION OF LATENT STATE

Next steps is to formulate a latent state which can combine the NN with the physics model. To form a PINN, a function $f = f(t)$ called as latent state predicted by the NN is approximated to $u(x,t)$. And, $f(t)$ is defined as below:

$$f := u_t + \mathcal{N}[u;\lambda] \tag{2.2}$$

To understand the concept of latent state, let us consider an example of current measurement in a converter. During the sampling of a current waveform, data is collected

based on sampling frequency. Lets us assume two points of currents measured are i(t1) and i(t2). Only these data are measurable and observed. And points between i(t1) and i(t2) are unobservable, hence termed latent points or latent states.

For parameter estimation, it is expected that the solution $u(t)$ is known, equations defining the system i.e $\mathcal{N}[u;\lambda]$ is also defined and λ is learnt during the process of approximating the latent state $f = f(t)$ by NN to the known solution $u(t)$.

2.2. DISCRETIZATION OF THE MODEL

u(t) is a continuous time function of either current or voltage. It is required to define a set discretisation interval for data collection and later to couple the NN to the physics model. Let us assume the available solution to $u(t)$ are either available, measurable or observable at the time t_n and t_{n+1}, where $t_{n+1} = t_n + \Delta t$. Then the unobservable latent state f(t), can be discretised as q number of points between t_n and t_{n+1} with an evenly distributed time spread of $t_{n+c_i} = t_n + c_i \Delta t$, where $c_i \in [0,1]$ and $i = 1,2,3....q$. Figure 2.2, visualises the discretisation of the function u(t) and f(t).

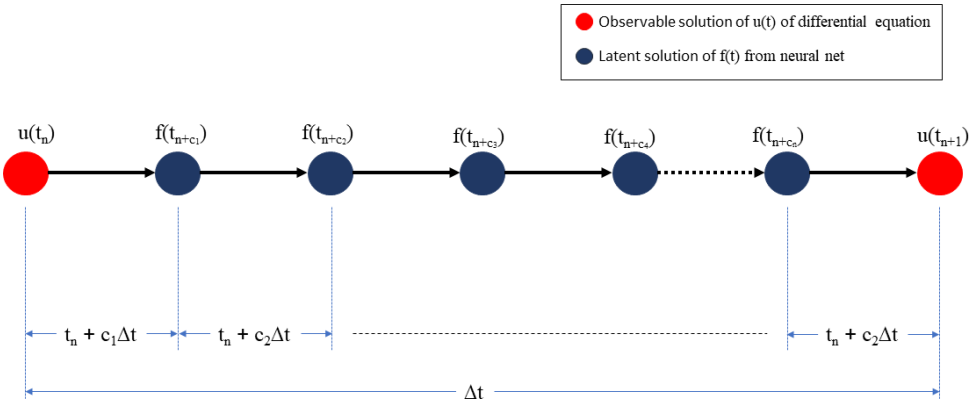


Figure 2.2: Discretization of the PINN model

2.3. COUPLING OF NEURAL NET AND PHYSICS MODEL

Observable solution points $u(t_n)$ and $u(t_{n+1})$ and the latent solution points of NN, $f(t_{n+c_i})$ needs to be coupled. Raissi et al. [21], proposes general form of Runge-Kutta method. Considering there are q intermittent latent points between $u(t_n)$ and $u(t_{n+1})$, general Runge-Kutta equation can be derived as below:

$$f(t_{n+c_i}) = u(t_n) - \Delta t \sum_{j=1}^{q} a_{ij} \mathcal{N}[f(t_{n+c_j});\lambda], \quad i = 1,2...q \tag{2.3}$$

$$u(t_{n+1}) = u(t_n) - \Delta t \sum_{j=1}^{q} b_j \mathcal{N}[f(t_{n+c_j});\lambda] \tag{2.4}$$

Incorporating equation 2.4 in 2.3:

$$f(t_{n+c_i}) = u(t_{n+1}) - \Delta t \sum_{j=1}^{q} (a_{ij} - b_j) \mathcal{N}[f(t_{n+c_j}); \lambda], \quad i = 1,2...q \tag{2.5}$$

Equation 2.3 which couples latent points with $u(t_n)$ is called as backward equation and equation 2.5 coupling latent points with $u(t_{n+1})$ is called as forward equation. Figure 2.3 visualizes the concept of coupling latent state points with the observable points. Equation 2.3 and 2.5 contains constant parameters (a_{ij}, b_j, c_j) of Runge-Kutta methods. John C. Butcher, classified the Runge-Kutta methods and proposed a generalization form by using a matrix (a_{ij}) and two vectors (b_j, c_j), where i,j = 1,2,...q. [26].

This generalization is called as Butcher tableau. Depending upon the choice of either implicit or explicit method and the number of intermittent points q, the general form can be depicted as in equation 2.6. For the implicit Runge-Kutta method, Raissi et al. [21] have calculated the Butcher tableau for various values of q as open source, which will be used further.

$$\begin{array}{c|cccc} c_1 & a_{11} & a_{12} & \dots & a_{1q} \\ c_2 & a_{21} & a_{22} & \dots & a_{2q} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_q & a_{q1} & a_{q2} & \dots & a_{qq} \\ \hline & b_1 & b_2 & \dots & b_q \end{array} \tag{2.6}$$

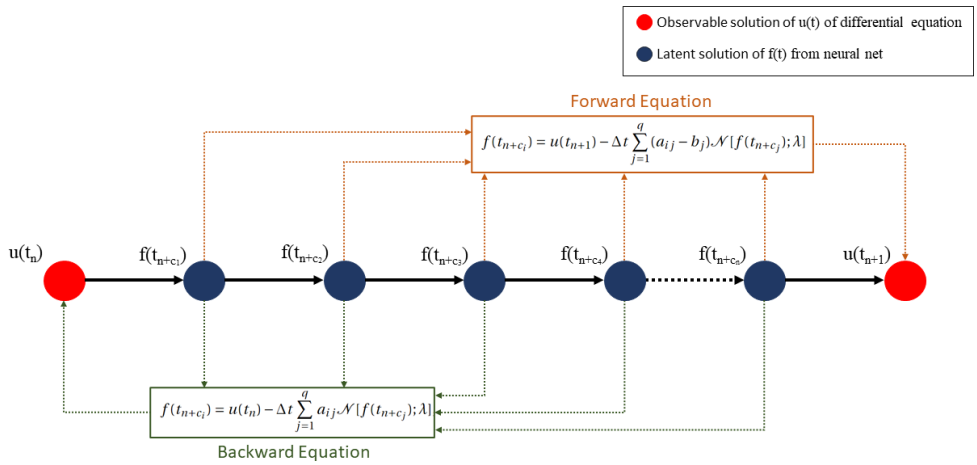


Figure 2.3: Coupling of observable points and latent state points

2.4. PROGRAMMING ALGORITHM OF PINN

To develop PINN, Python(v3.6) with TensorFlow(v1.15) backend is used. Tensorflow has two approaches; one is eager execution, and the other is the computational graph method. In this thesis, the computational graph method is applied as it provides complete flexibility and control to define each computation to be performed, which allows the integration of the physics model into the neural net. However, using the computational graph approach comes with the cost of its complexity, high learning curve and literally every computational flow have to be defined manually. The flow of the program is depicted in figure 2.4 and described below:

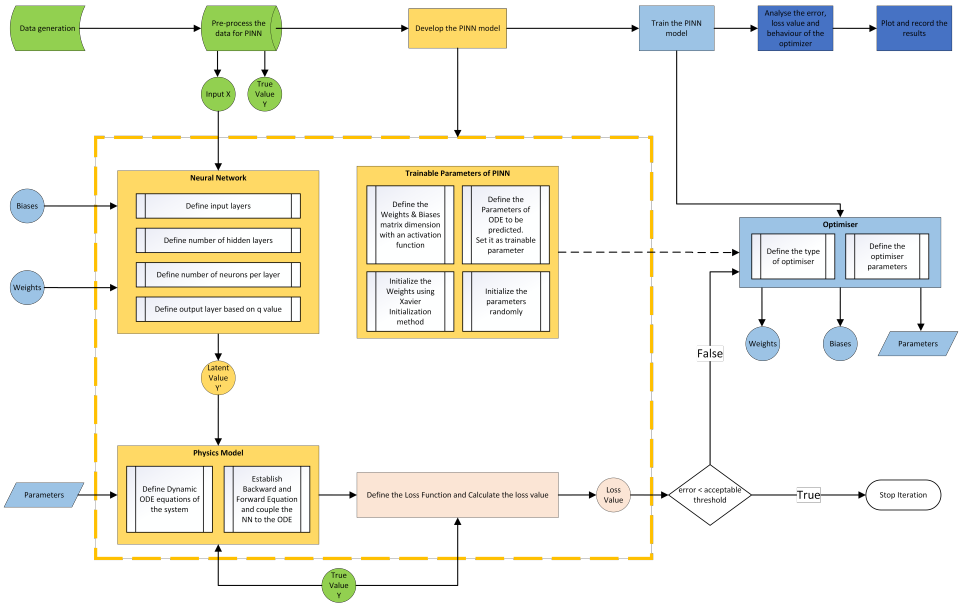


Figure 2.4: Flow chart of the program flow to implement PINN

2.4.1. DATA GENERATION

Considering various application profiles of the converter, it is important to formulate a generalized approach to sample and collect data. Data can be collected in multiple ways, such as simulations, analytical solutions or even directly from field application measurements. Irrespective of the data collection method, a common sampling time can be established to collect data at the time t_n and t_{n+1}. It was observed during discretisation that Δt is the major parameter defining the structure of the coupling mechanism of PINN. For any given power electronics converter, the dynamics are mostly observed between the change of state of the switches, which either causes a higher peak or a lower peak of current or voltage waveform. Currents and voltages can be sampled and collected exactly at the change of switch states. Hence, peak-to-peak data collection will be used in this thesis. This peak to peak values will correspond to $u(t_n)$ and $u(t_{n+1})$ in figure 2.2.

The point of interest in this thesis is to estimate the parameters with minimal execution and least data points. Adding to that, the motto is not to understand or predict the dynamic behaviour of the converter. This allows for collecting the data in a steady state operation. Hence, for the boost converter application, 23 switching cycles of data are captured via simulation in PLECS. And, for AFE, 10 switching cycles of data are used. This is quite interesting as these correspond to a low amount of data that a conventional NN can never converge for a generalised solution.

2.4.2. DATA PREPROCESSING

For NN data is segregated as set of Inputs(X) and outputs, also called as true values(Y) as in Figure 2.1. It is required to process the data in a way accepted by the design philosophy of PINN. In this thesis, forward equation 2.5 and backward equation2.3 are used to couple the physics model to NN. Each sampled point will translate to two sets of data. One set is for the forward equation and the other for the backward equation. The backward equation data set will have data at only time t_n. The forward equation will require the next sampling point t_{n+1}. A value of -2 is used to indicate the forward data set and +2 for the backward data set, which makes it easy for the PINN algorithm to sort the data.

Table 2.1: Generalized form of data collection and preprocessing

Input Current X_i	Input Voltage X_v	Switch state	Output Current Y_i	Output Voltage Y_v	Forward Backward Indicator	Δt
$i(t_n)$	$v(t_n)$	0	$i(t_n)$	$v(t_n)$	+2	$(t_{n+1})-(t_n)$
$i(t_n)$	$v(t_n)$	0	$i(t_{n+1})$	$v(t_{n+1})$	-2	$(t_{n+1})-(t_n)$
$i(t_{n+1})$	$v(t_{n+1})$	1	$i(t_{n+1})$	$v(t_{n+1})$	+2	$(t_{n+2})-(t_{n+1})$
$i(t_{n+1})$	$v(t_{n+1})$	1	$i(t_{n+2})$	$v(t_{n+2})$	-2	$(t_{n+2})-(t_{n+1})$
$i(t_{n+2})$	$v(t_{n+2})$	0	$i(t_{n+2})$	$v(t_{n+2})$	+2	$(t_{n+3})-(t_{n+2})$
$i(t_{n+2})$	$v(t_{n+2})$	0	$i(t_{n+3})$	$v(t_{n+3})$	-2	$(t_{n+3})-(t_{n+2})$
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.
.	.	.	.	.	.	.

2.4.3. INITIALIZING THE PINN

The initialisation of PINN can be broadly classified as follows:

STRUCTURE OF NEURAL NET

There are no available guidelines or deterministic ways to formulate the number of layers and neurons for PINN. These numbers are decided based on the trial and error method. In this thesis, several iterations of a different combination of several layers and neuron combinations are performed to arrive at an optimal value. Optimal value here means size of the neural net which will have least estimation error and quick convergence time.

NUMBER OF LATENT POINTS

Since the Runge-Kutta method is a numerical method, it causes truncation error when approximating a given function's solution. Truncation error depends on the step size (Δt) and the number of intermittent points (q). Δt in our application depends on the sampling rate, which in turn depends on the converter's PWM frequency. For the boost converter switching frequency is 10kHz and for AFE is 40kHZ. Local truncation error for each function approximation from the Runge-Kutta method can be calculated as:

$$Truncation\ Error \approx \Delta t^{(2*q)}$$

The truncation error should be less than the machine epsilon(ϵ). Machine epsilon depends on the data type used by the program and the type of operating system. Machine epsilon can be defined as the difference between 1, and the next larger floating point number[27] is given by the below equation:

$$Machine\ Epsilon = b^{-(p-1)}$$

where:
b is the base of the floating point system
p is the precision

All numeric values in the program are used as datatype float, corresponding to binary64. Referring to the table IEEE Std 754™-2008 [28] as in table 2.2, machine epsilon for a 64-bit operating system can be calculated as:

$$Machine\ Epsilon = 2^{-(53-1)} = 2.22 * 10^{-16}$$

Table 2.2: Parameters defining basic format floating-point numbers [28]

	Binary format (b=2)			**Decimal format (b=10)**	
Parameter	**binary32**	**binary64**	**binary128**	**decimal64**	**decimal128**
p, digits	24	53	113	16	34
emax	+127	+1023	+16383	+384	+6144

Number of intermittent points q for the boost converter:

$$q > \frac{1}{2} \cdot \frac{log(\epsilon)}{log(\Delta t)}$$

$$q > \frac{1}{2} \cdot \frac{log(2.22 * 10^{-16})}{log(10^{-5})}$$

$$q > 1.56$$

$$\boxed{q = 2}$$

Number of intermittent points q for the AFE converter:

$$q > \frac{1}{2} \cdot \frac{log(\epsilon)}{log(\Delta t)}$$
$$q > \frac{1}{2} \cdot \frac{log(2.22 * 10^{-16})}{log(25^{-6})}$$
$$q > 0.93$$
$$\boxed{q = 1}$$

2.4.4. LOSS FUNCTION AND HYPER-PARAMETER OF THE OPTIMISER

As described in the figure 2.1, loss value is obtained via the physics model. Based on the loss value, an optimiser will optimise the weights, biases and parameters of the physics model through backpropagation. Weights and biases are initialised via Xavier initialisation with a tanh activation function. Parameters of the physics model are initialised randomly. Loss function (ϕ) of the PINN can be expressed as equation 2.7. Adam optimiser is chosen as the optimiser. Parameters of Adams, especially the learning rate, must be set with a trial and error methodology. One such approach is to set a value for the learning rate, reduce it by half for each iteration trial, and observe if there are any issues with convergence.

$$\begin{aligned}\phi = &\sum_n \left[\left(X_{True_1}(t_n) - X_{Predict_1}(t_n) \right)^2 + \left(X_{True_1}(t_{n+1}) - X_{Predict_1}(t_{n+1}) \right)^2 \right] \\ &+ \sum_n \left[\left(X_{True_2}(t_n) - X_{Predict_2}(t_n) \right)^2 + \left(X_{True_2}(t_{n+1}) - X_{Predict_2}(t_{n+1}) \right)^2 \right]\end{aligned} \tag{2.7}$$

2.4.5. GUIDELINES TO TRAIN A PINN

Majorly there are two parts in the model, one is the Neural Net which requires training of hyper-parameters (weights and biases) and the other is physics model with circuit parameters (L,C,R). It is definitely not straight forward and requires several iterations of trial and error to optimise the model. Based on the experience in this thesis, following guidelines to train a PINN is proposed:

Step 1: Once the model parameters are decided as per the initialisation proposed, initialise the parameters of the physics model to the true values(instead of random initialisation) and set the variable in the TensorFlow computational graph as non-trainable. It only trains the weights and biases of the Neural Net to match the input and true values, allowing us to observe if the formulation of the physics equation is right and observe the convergence of the NN.

Step 2: If the model is not converging, there can be an issue with formulated ODEs in the physics model or the parameters of the neural net or optimiser. For trouble shooting, try the below cases

(a) Re-verify the formulated ODEs in the physics model

(b) Increase or decrease the size of the NN (layers and neurons)

(c) Change the learning rate of the Adam optimiser

Step 2: If the model converges, set back the parameters of the physics model to trainable and initialise it randomly.

(a) If the model doesn't converge now, observe the behaviour of:

i. The trend of error value of each parameter.

ii. The trend of evolution of loss value.

(b) Increase or decrease the size of the NN (layers and neurons)

(c) Change the learning rate of the Adam optimiser

Step 3: Once the model converges for both the cases, it is now required to optimise the size of a neural net.

(a) Set the hidden layer to 1, and increase the number of neurons in a certain step size. Tabulate the observations such as error, loss value, convergence iteration and execution time.

(b) Increase the number of hidden layers and repeat the above step.

(c) Analyse the tabulated results and, based on the predetermined trade-off criteria, select the size of the neural net.

Step 4: Since the size is decided, it is now required to check how the PINN perform at different settings of the learning rate of Adams optimiser. Increase or decrease the learning rate and observe at what point the model performs the best.

3

Parameter Estimation of Boost Converter

To validate and verify the approach derived in section 2, the PINN model for a boost converter is implemented as a proof of concept. This is done as the boost converter has fewer parameters to estimate with simpler equations. Based on the results from the boost converter, PINN will be modelled for an AFE converter helping us to assess the optimisation and scalability problem in depth.

3.1. Dynamic Model of Boost Converter

PINN approach uses the dynamic system equations in the form of ODE as a hard constraint and bounds the domain of acceptable solutions[29]. Hence, ODE equations of boost converter based on the state of the switch based on figure 3.1 can be derived to approximate the space of solutions. The boost converter considered in this thesis is ideal in nature, with no voltage drop across the switches and diodes. Ideal inductor and capacitor are considered with no parasitic resistances. Below is the specification of the boost converter:

Table 3.1: Specification of the Boost Converter

Parameter	**Magnitude**
Input Voltage (V_{in})	24V
Output Voltage (V_o)	40V
Inductor Value (L)	250μH
Capacitor Value (C)	200μF
Load Resistor (R_{load})	2Ω
Switching Frequency (fsw)	10kHz

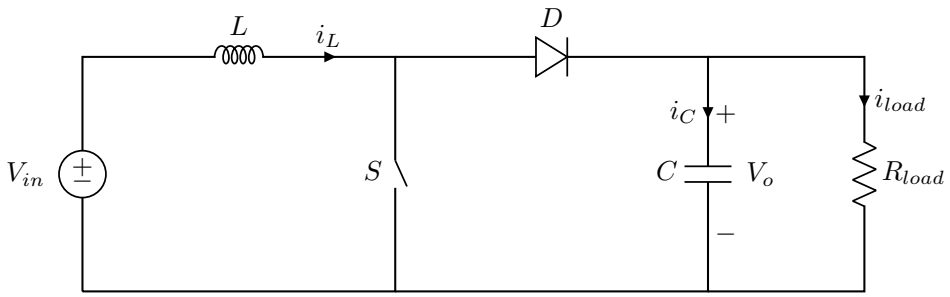


Figure 3.1: Circuit topology of Boost Converter

The equations for ON state (S=1):

$$\begin{aligned} \frac{dv_{out}}{dt} &= \frac{-v_{out}}{R_{load} \cdot C} \\ \frac{di_L}{dt} &= \frac{v_{in}}{L} \end{aligned} \tag{3.1}$$

The equations for OFF state (S=0):

$$\begin{aligned} \frac{dv_{out}}{dt} &= \frac{i_L}{C} - \frac{v_{out}}{R_{load} \cdot C} \\ \frac{di_L}{dt} &= \frac{v_{in} - v_{out}}{L} \end{aligned} \tag{3.2}$$

3.2. PINN MODELLING OF BOOST CONVERTER

The derivation of the PINN model follows exactly the procedure in chapter 2. For the boost converter, it is assumed that the output voltage(V_{out}) and inductor current(i_L) are measurable with peak-to-peak sampling. Parameters to be estimated are the Inductance(L), Capacitance(C), Load Resistance (R_{load}), and input voltage(V_{in}). Based on the derived equations 3.1 and 3.2, and using the general form defined in equation 2.1, PINN model equations of the boost converter can be derives as below:

$$\begin{aligned} \frac{dv_{out}}{dt} + \mathcal{N}[v_{out}; \lambda] = 0 \\ \frac{di_L}{dt} + \mathcal{N}[i_L; \lambda] = 0 \end{aligned} \tag{3.3}$$

Where:

$$\lambda = \{L, C, R_{load}, V_{in}\}$$

$$\begin{aligned}\mathcal{N}[v_{out};\lambda] &= \left[S\frac{v_{out}}{R_{load}\cdot C}\right] - \left[(1-S)\left(\frac{i_L}{C} - \frac{v_{out}}{R_{load}\cdot C}\right)\right] \\ \mathcal{N}[i_L;\lambda] &= \left[S\frac{-v_{in}}{L}\right] - \left[(1-S)\frac{v_{in}-v_{out}}{L}\right]\end{aligned} \tag{3.4}$$

3.3. DISCRETIZATION OF BOOST CONVERTER AND LOSS FUNCTION

Using the general equation 2.3, and the equations 3.6 and 3.5 forward equation for backward equation for boost converter can be formulated as below:

Backward equations for boost converter:

$$\begin{aligned}v_{out}(t_{n+c_i}) &= v_{out}(t_n) - \Delta t\sum_{j=1}^{q} a_{ij}\mathcal{N}[v_{out}(t_{n+c_j});\lambda], \quad i = 1,2...q \\ i_L(t_{n+c_i}) &= i_L(t_n) - \Delta t\sum_{j=1}^{q} a_{ij}\mathcal{N}[i_L(t_{n+c_j});\lambda], \quad i = 1,2...q\end{aligned} \tag{3.5}$$

Forward equations for boost converter:

$$\begin{aligned}v_{out}(t_{n+c_i}) &= v_{out}(t_{n+1}) - \Delta t\sum_{j=1}^{q} (a_{ij}-b_j)\mathcal{N}[v_{out}(t_{n+c_j});\lambda], \quad i = 1,2...q \\ i_L(t_{n+c_i}) &= i_L(t_{n+1}) - \Delta t\sum_{j=1}^{q} (a_{ij}-b_j)\mathcal{N}[i_L(t_{n+c_j});\lambda], \quad i = 1,2...q\end{aligned} \tag{3.6}$$

Loss function for boost converter depends on the prediction of inductor current and output voltage, and it is defined based on general loss function expressed in equation 2.7:

$$\begin{aligned}\phi_{Boost} = &\sum_n \left[\left(v_{out}(t_n) - v_{out_{c_i}}(t_n)\right)^2 + \left(v_{out}(t_{n+1}) - v_{out_{c_i}}(t_{n+1})\right)^2\right] \\ &+ \sum_n \left[\left(i_L(t_n) - i_{L_{c_i}}(t_n)\right)^2 + \left(i_L(t_{n+1}) - i_{L_{c_i}}(t_{n+1})\right)^2\right], \quad i = 1,2...q\end{aligned} \tag{3.7}$$

3.4. DATA ACQUISITION AND CONFIGURATION OF PINN FOR THE BOOST CONVERTER

3.4.1. DATA ACQUISITION

To collect data for boost converter, simulation and numerical solution for carried out. Simulation was performed in PLECS and for numerical solution MATLAB was used based on ODE45 solver to solve equation 3.1 and 3.2. Both these methods gave similar results; hence any of these can be used to generate raw data. Matlab was used to preprocess the data based on section 2.4.2.

3.4.2. DEFINING THE STRUCTURE OF PINN

Inputs for the NN are defined as inductor current and output voltage at time t_n. The size of the neural net is decided based on trial and error methodology as described in section 2.4.5. The number of outputs of the latent state is decided by the q value. The value of q is selected as 2 based on derivation in section 2.4.3. True values of the boost PINN model are inductor current and output voltage at time t_n and t_{n+1}

Physics model of the boost converter consists of ODE equation set 3.4. To couple the latent state and the true value via the physics model, forward equations 3.6 and back equations 3.5 are applied. Figure 3.2 shows the visualisation of the PINN model for the boost converter.

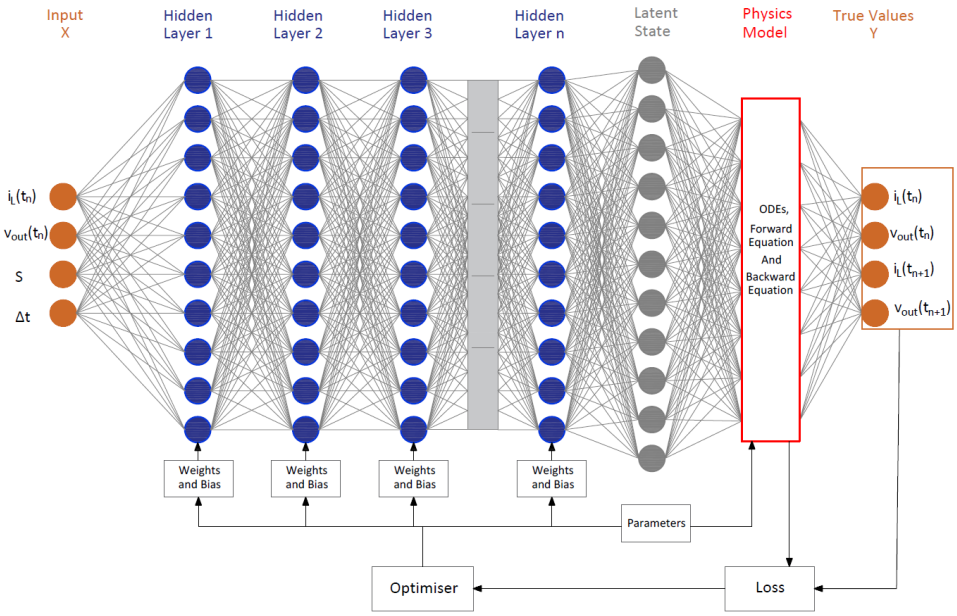


Figure 3.2: Architecture of PINN applied for the Boost Converter

3.4.3. VALIDATION AND OPTIMISATION OF THE PINN FOR BOOST CONVERTER

Stage 1: Verification of Physics Model

To begin with, based on the guidelines formulated in 2.4.5, parameters of the boost converter L, C, R_{load}, V_{in} were initialised to their true values and were set as non-trainable in the computational graph of TensorFlow. Only the neural net parameters (weights and biases) will be trained in this stage. For the initial trial, the neural net size was 5 hidden layers with 50 neurons in each stage. This selection is just a starting point, and the actual size will be optimised in the later stage.

It is observed in figure 3.3 that the model did converge, confirming the rightness of the defined physics model. What this means is that the PINN model was able to correlate the latent states predicted by the NN to the values of currents and voltages at t_n and t_{n+1}.

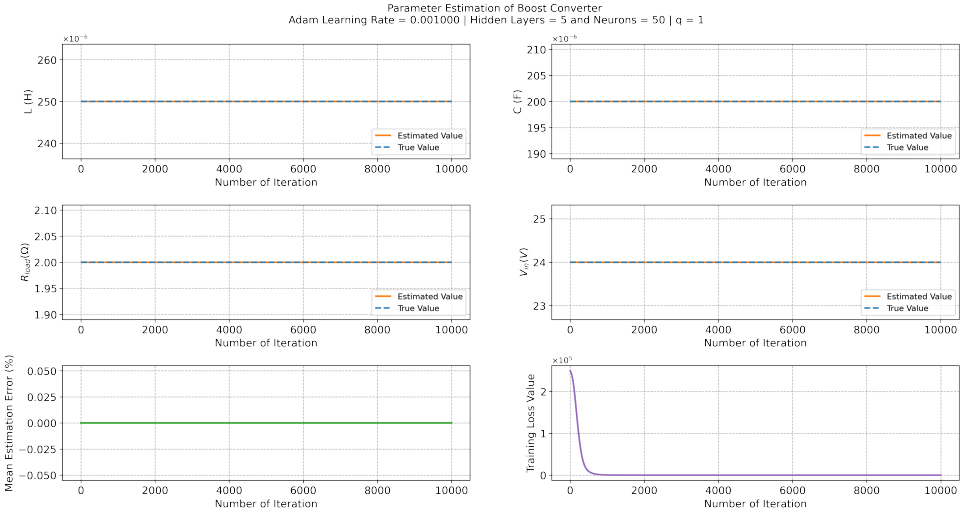


Figure 3.3: Validation of PINN model for the Boost Converter by setting the physics parameters L, C, R_{load} and V_{in} to their true value and it is set as non-trainable in the PINN model

Stage 2: Verification of convergence of PINN

In the second stage, the parameters of the physics model (L, C, R_{load}, V_{in}) were randomly initialised and were set as trainable parameters. Now, the optimizer will try to estimate these parameters during the training process. Evolution and convergence of the percentage error value of the parameters (L, C, R_{load}, V_{in}) were observed for various learning rate settings of the Adams optimiser. In figure 3.4, it can be observed that the parameters do not converge to a least stable error value for learning rates of $8x10^{-6}$, $1.6x10^{-5}$ and $3.1x10^{-5}$ of the optimiser. In figure 3.5, it can be observed that even though the percentage error is higher, the error value did reach a steady state value for the learning rates set as $6.3x10^{-5}$, $1.25x10^{-4}$, $2.5x10^{-4}$ and $1x10^{-3}$.

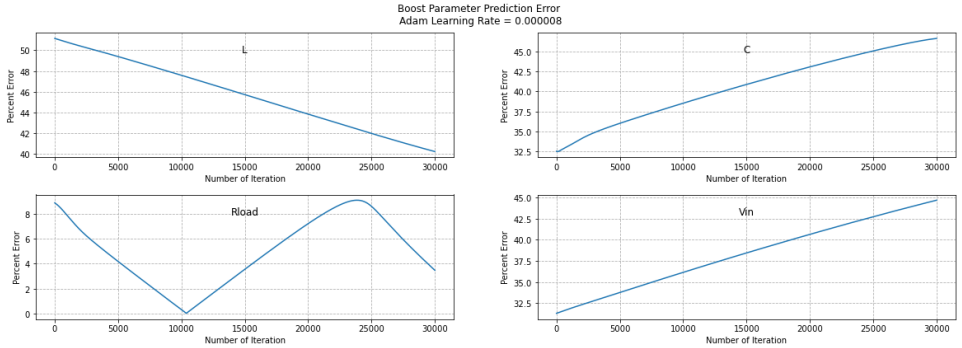
Observation and Analysis of optimisation

1. During the first few iterations of backpropagation, the loss value will be extremely high. The aim of the optimiser is to reduce this loss by training the weights, biases and physics parameters. Weights and biases are unit-less normalised values with a range bound based on the activation function. In our case, tanh activation function would give the range for weights between -1 to 1 centered around 0.

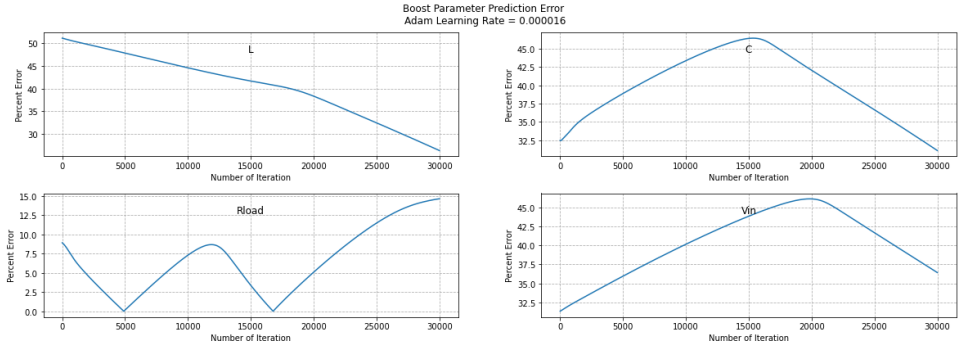
   The issue with physics model is that, the parameters(L, C, R_{load}, V_{in}) are in different units. For instance, inductance in μH and in resistance in Ω. To normalise them, the following strategy is used as per table 3.2:

Table 3.2: Pseudo Normalisation of Physics Parameters for Boost Converter

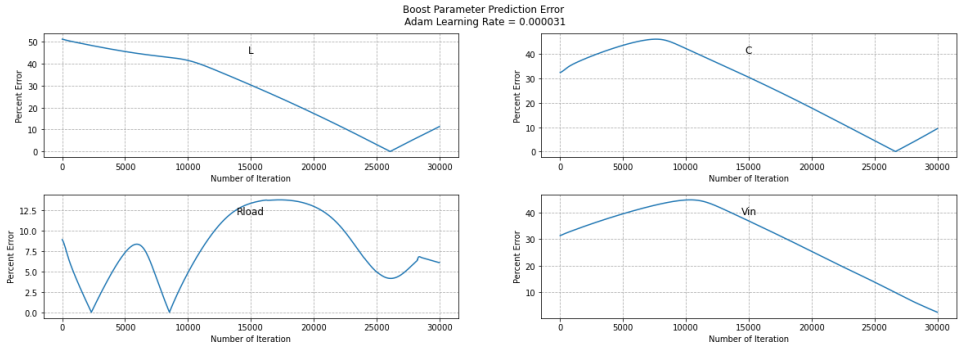
Parameter	Actual Value		Scaling for TensorFlow Variable		Re-scaling for Physics Equation
L (H)	250μ	2.5×10^{-4}	$ln(2.5)$	0.916290732	$e^{0.92} \times 10^{-4}$
C (F)	200μ	2×10^{-4}	$ln(2)$	0.693147181	$e^{0.69} \times 10^{-4}$
R_{load}(Ω)	2	2	$ln(2.4)$	0.693147181	$e^{0.69}$
V_{in} (V)	24	2.4×10^{-1}	$ln(2)$	0.875468737	$e^{0.88} \times 10^{1}$



(a)

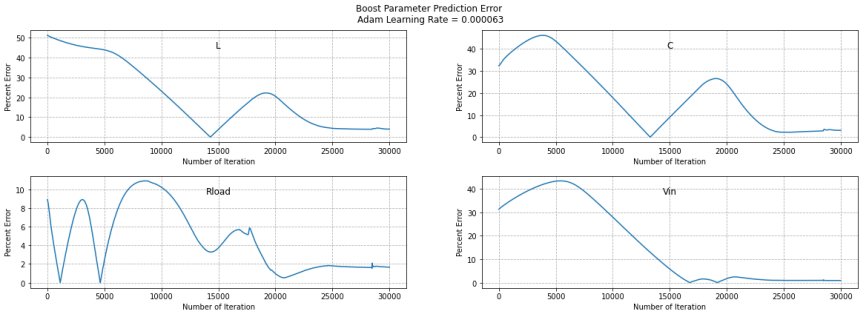


(b)

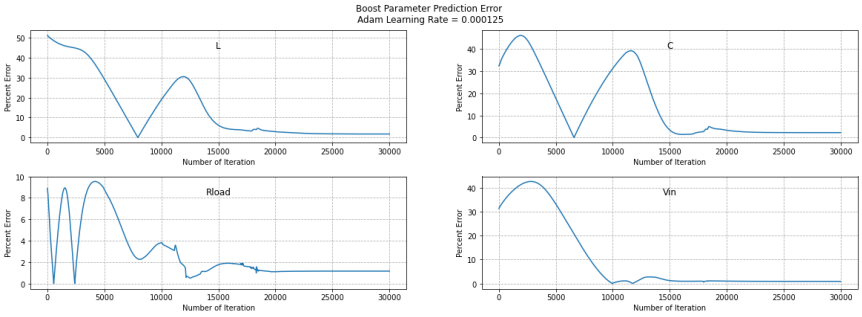


(c)

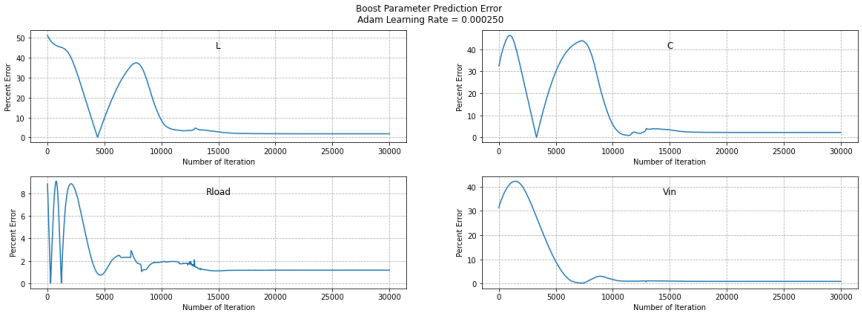
Figure 3.4: Variation of error percentage of estimated parameters showing non-convergence of the optimisation process for Boost Converter when learning rate is set as (a) $8 \cdot 10^{-6}$, (b) $1.6 \cdot 10^{-5}$ and (c) $3.1 \cdot 10^{-5}$ for the Adams optimiser



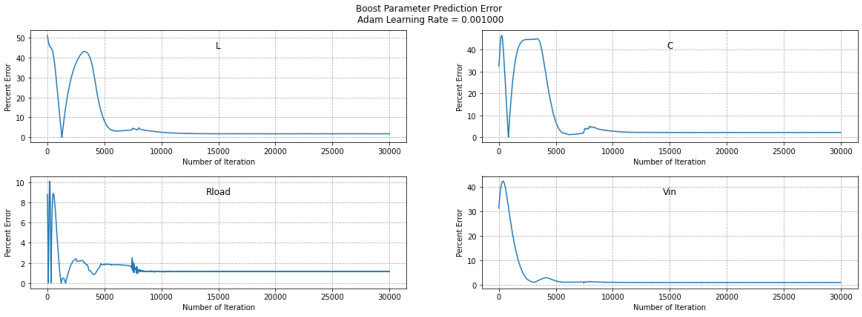
(a)



(b)



(c)



(d)

Figure 3.5: Variation of error percentage of estimated parameters showing convergence of the optimisation process for Boost Converter when learning rate is set as (a) $6.3 \cdot 10^{-5}$, (b) $1.25 \cdot 10^{-4}$, (c) $2.5 \cdot 10^{-4}$ and (d) $1 \cdot 10^{-3}$ for the Adams optimiser

Stage 3: Optimisation of NN size

Based on guidelines in section 2.4.5, various combinations of hidden layers and the number of neurons were tested. Starting with a lower number, the number of hidden layers and the number of neurons per hidden layer increased gradually. Table 3.3 presents results of estimated parameters in terms of percentage error. And, table 3.4 presents the number of iteration of convergence. The number of iterations for convergence is a better evaluation parameter as it doesn't depend on the type of computer used.

It is observed that the NN with a smaller size performs better in the estimation of the parameters. For an optimiser, the physics parameter is just another variable to optimise. Hence the probability of having a lower error for physics parameter estimation is higher in a smaller neural net.

Table 3.3: Percentage error of estimated parameters of the boost converter

Neurons	Error Percentage																								
	Hidden Layer = 1					Hidden Layer = 2					Hidden Layer = 4					Hidden Layer = 6					Hidden Layer = 8				
	L	C	Rload	Vin	mean	L	C	Rload	Vin	mean	L	C	Rload	Vin	mean	L	C	Rload	Vin	mean	L	C	Rload	Vin	mean
5	0.6	0.3	1.4	1.4	0.9	33.5	32.8	1.5	2.0	17.5	33.5	32.8	1.5	2.0	17.5	33.5	32.8	1.5	2.0	17.5	33.5	32.8	1.5	2.0	17.5
10	0.6	0.3	1.3	1.4	0.9	0.6	0.3	1.3	1.4	0.9	0.6	0.3	1.3	1.4	0.9	0.6	0.3	1.3	1.4	0.9	17.5	15.9	2.1	1.0	9.1
20	0.6	0.3	1.3	1.4	0.9	0.6	0.3	1.3	1.4	0.9	0.6	0.3	1.3	1.4	0.9	0.6	0.3	1.3	1.4	0.9	0.6	0.3	1.3	1.4	0.9
40	0.6	0.3	1.3	1.4	0.9	0.6	0.3	1.3	1.4	0.9	0.6	0.3	1.3	1.4	0.9	0.6	0.3	1.3	1.4	0.9	0.6	0.3	1.3	1.4	0.9
60	0.6	0.3	1.3	1.4	0.9	0.6	0.3	1.3	1.4	0.9	0.6	0.3	1.3	1.4	0.9	0.6	0.3	1.3	1.4	0.9	0.6	0.3	1.3	1.4	0.9

In the previous case, the neural net size was compared based on the percentage error of the estimated parameter. It is observed that similar results are obtained for various neural net sizes. Smaller neural nets have lower execution times. Hence it makes sense to optimise the network size based on the iteration the PINN model requires to converge. From table 3.4, a Neural net with one and two hidden layers performed the best. It is also observed that increasing the number of neurons has a higher impact on convergence than increasing the number of layers.

Table 3.4: Number of iterations required for convergence

Neurons	Number of Iterations at Convergence				
	Hidden Layer = 1	Hidden Layer = 2	Hidden Layer = 4	Hidden Layer = 6	Hidden Layer = 8
5	1,00,000	60,000	60,000	60,000	60,000
10	50,000	52,000	50,000	58,000	1,00,000
20	30,000	30,000	32,000	26,000	40,000
40	18,000	17,000	19,000	20,000	25,000
60	16,000	13,000	15,000	19,000	20,000

To investigate further, only the number of neurons for the neural net with one and two hidden layers will be increased to a point where no improvement in performance is observed. Table 3.5 and 3.6 shows the results for hidden layer one and two respectively. It is observed that, from table 3.3, 3.5 and 3.6, PINN model with larger size tend to have higher error of estimation. It is found that PINN developed for the boost converter performed the quickest with 1 hidden layer and 100 neurons with a mean error of 0.89%, and the model converged at 4000^{th} iteration. It is impressive to note that the developed PINN model just took 25.76 seconds with only 25 switching frequency cycle data (100 data points) to predict the parameters of the boost converter.

Table 3.5: Performance evaluation for Hidden Layer = 1

Neurons	Hidden layer = 1, q = 1, adam learning rate = 0.001						
	L Error %	C error %	Rload error %	Vin error %	mean error %	Iterations at Convergence	Execution Time (seconds)
100	0.2	0.3	1.3	1.8	0.89	4000	25.76
200	0.2	0.3	1.3	1.8	0.89	4000	25.77
300	0.2	0.3	1.3	1.8	0.89	4000	25.74
400	0.2	0.3	1.3	1.8	0.89	4000	25.52
1000	0.2	0.3	1.3	1.8	0.89	4000	31.64

Table 3.6: Performance evaluation for Hidden Layer = 2

Neurons	Hidden layer = 2, q = 1, adam learning rate = 0.001						
	L Error %	C error %	Rload error %	Vin error %	mean error %	Iterations at Convergence	Execution Time (seconds)
100	0.1	0.3	1.4	1.8	0.88	4000	25.30
200	0.1	0.3	1.3	1.7	0.84	3500	25.60
300	0.1	0.3	1.3	1.7	0.85	3700	26.07
400	0.2	0.3	1.3	1.8	0.89	4000	31.07
1000	0.2	0.3	1.3	1.8	0.89	5000	123.20

Observation and Analysis of NN size

- PINN performance has a higher impact with an increasing number of neurons than the number of the hidden layer.
- NN with 2 hidden layers and 60 neurons performed the best.
- Larger NN size doesn't guarantee better performance. The PINN model with a smaller neural net size had a smaller estimation error.

3.5. RESULTS OF PINN FOR BOOST CONVERTER

Developed PINN model for boost converter, in steady state, can estimate the parameters with a mean error of 0.89% in 25.76s. After optimisation based on the guidelines proposed, stable and quicker convergence is observed in the figure 3.6. NN of PINN predicts the latent state, which in turn is used by the physics model to relate to the true output values. The results of these prediction is presented in the figure 3.7.

Even though the point of interest is the estimation of parameters which can be achieved by training the PINN on steady-state data, verifying the performance of the developed model for a transient state is also carried out. It is impressive to observe an acceptable performance of the PINN model for the transient phase with a mean error of 7.75% and execution time of 34.74s. Figure 3.8 and 3.9 presents the results for the transient state. In table 3.7, results of PINN for steady state and transient state is compared.

Table 3.7: PINN results of boost converter for steady state and transient state

Test Type	Hidden layer = 1, Neurons = 100, q = 1, adam learning rate = 0.001						
	L Error %	C error %	Rload error %	Vin error %	mean error %	Iterations at Convergence	Execution Time (seconds)
Steady State	0.2	0.3	1.3	1.8	0.89	4000	25.76
Transient State	9.2	10.5	6.2	5.1	7.75	6000	34.74

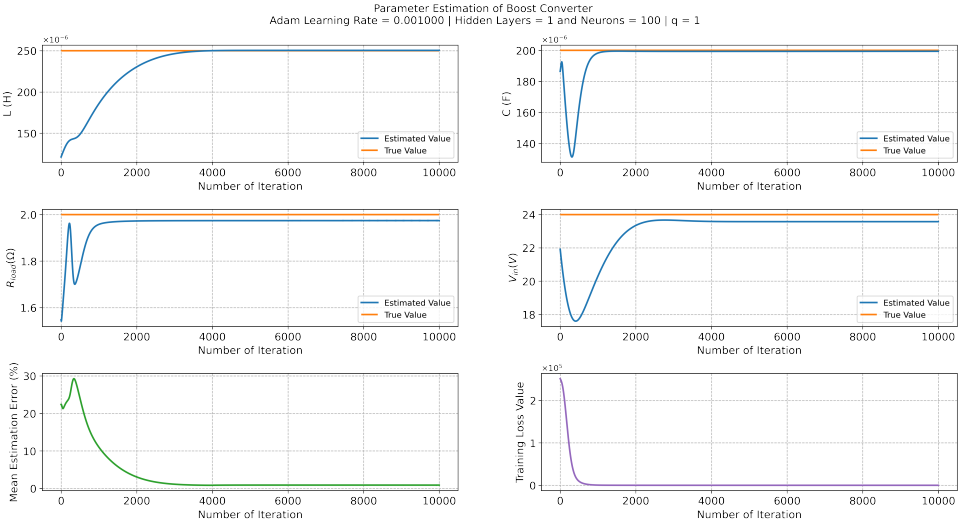


Figure 3.6: PINN parameter estimation of the boost converter at steady state

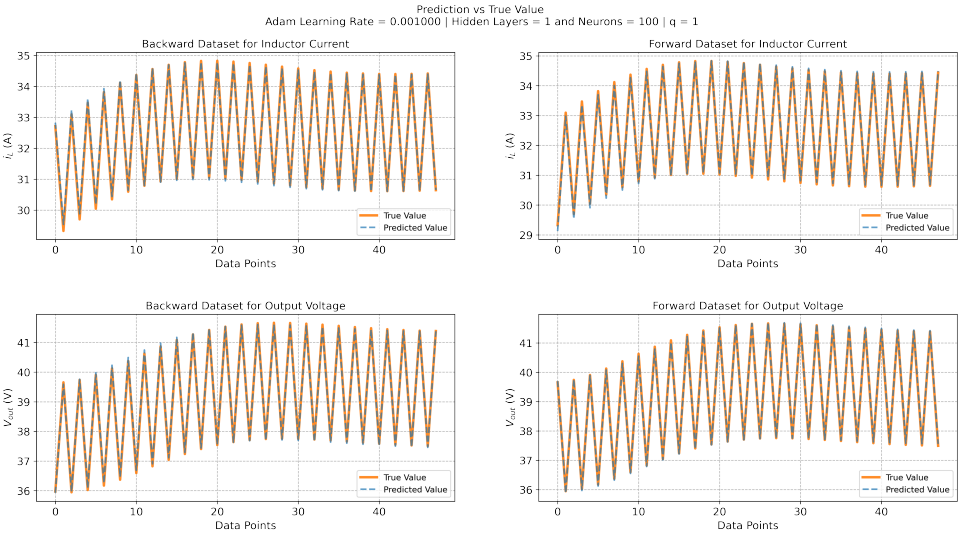


Figure 3.7: Current and Voltage waveform prediction for backward and forward data-set at steady state

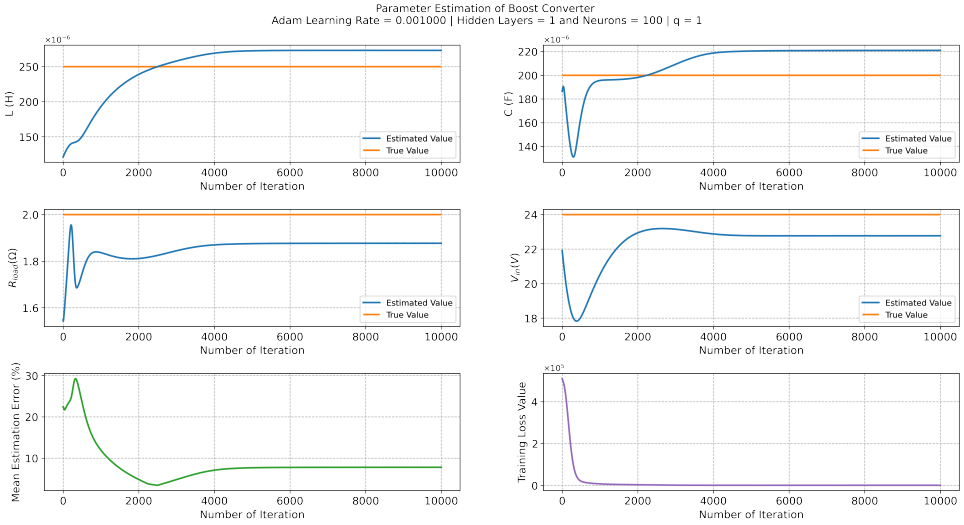


Figure 3.8: PINN parameter estimation of the boost converter at transient state

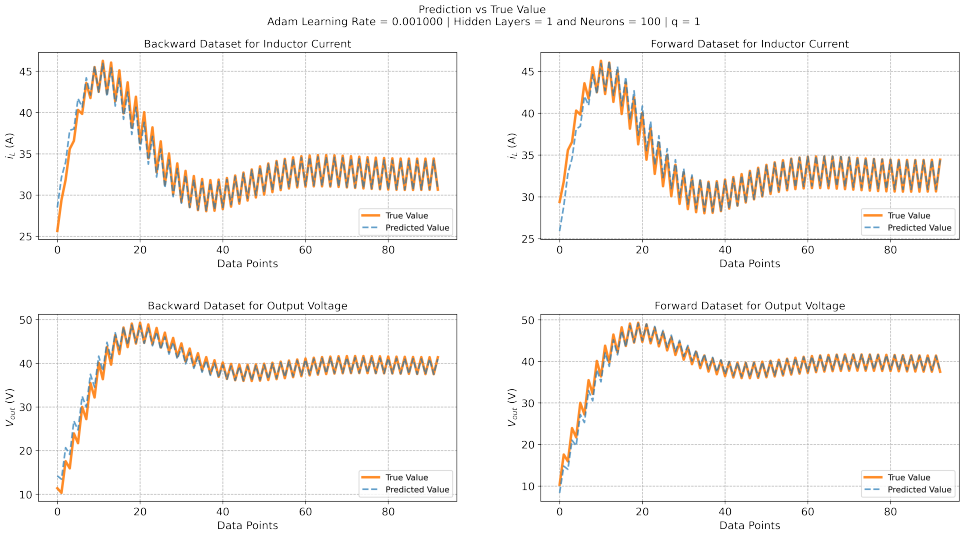


Figure 3.9: Current and Voltage waveform prediction for backward and forward data-set at transient state

4

PARAMETER ESTIMATION OF ACTIVE FRONT END CONVERTER

Guidelines developed in section 2.4.5 are applied to develop a PINN model to estimate the parameters of the boost converter. A similar approach is used in developing the PINN model of the AFE converter. As the first step, a simulation model is developed in PLECS to generate the data required, and Matlab is used to preprocess the data. In the next step, equations derived in general form in the section 2 will be used to formulate the PINN model for the AFE. Finally, the results are analysed.

4.1. DYNAMIC MODELLING OF THE AFE CONVERTER

The point of interest in the thesis is parameter estimation of the circuit parameters such as filter inductance, DC-link capacitor and load resistance. To validate the feasibility of the PINN model, 2-level VSR as an AFE is simulated in open loop control configuration with a Sinusoidal PWM. Table 4.1 shows the circuit specifications of the 2-level VSR as AFE converter and figure 4.1 visualises the circuit topology of the AFE.

Each phase of the AFE converter has two switches operated in complementary. Which means when the top switch of the leg (S_{a1}, S_{b1}, S_{c1}) are in ON state, then the bottom switches(S_{a2}, S_{b2}, S_{c2}) are in OFF state and vice versa. The control pulses for these switches are provided via bipolar sine PWM where S_a, S_b and S_c are for top switches and S'_a, S'_b and S'_c for the complimentary bottom switches. Figure 4.4 shows the sub-interval control pulses, and figure 4.3 shows the control pulses for 1 cycle of fundamental frequency generated by the Sine PWM.

Table 4.1: Specification of the 2-level VSC Converter as the AFE converter

Parameter	Magnitude
Grid Line-Neutral Voltage (u_{abc})	230V
Grid fundamental frequency (f_1)	50 Hz
Input Filter Inductance (L_{abc})	250 μH
Input Filter Resistance (R_{abc})	20$m\Omega$
DC-Link Capacitor (C)	3000 μF
DC-Link Voltage (V_{dc})	800V
Output Power (P_{out})	30kW
Switching Frequency (fsw)	40kHz

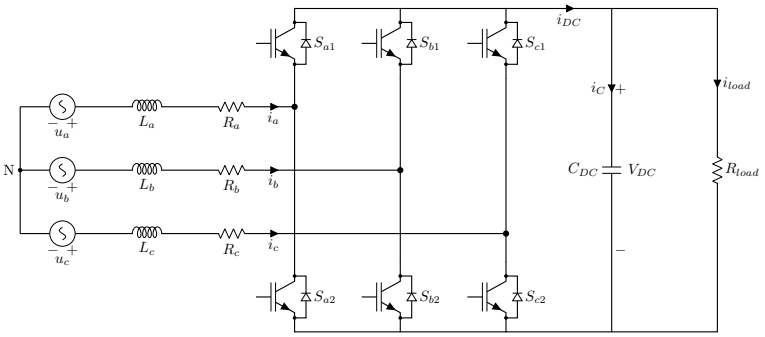


Figure 4.1: Circuit topology of the 2-level VSR

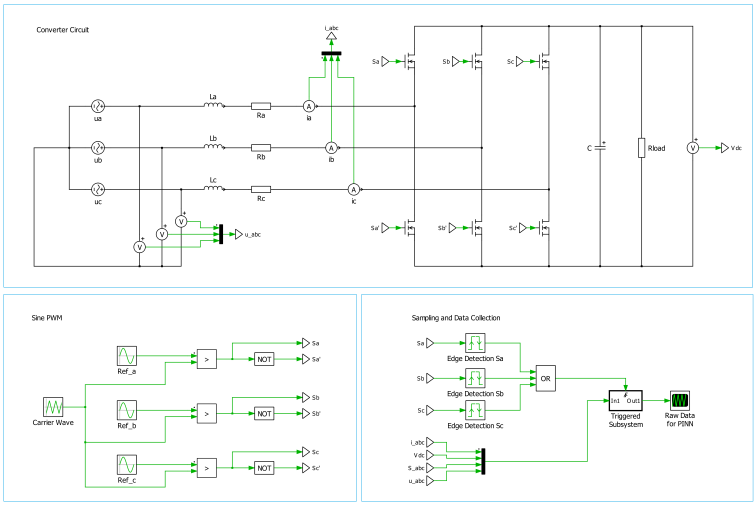


Figure 4.2: PLECS Simulation of AFE

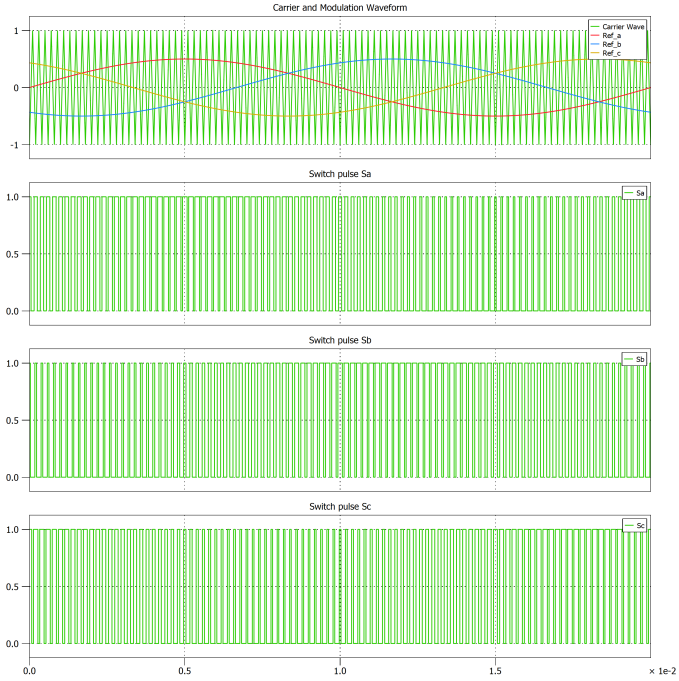


Figure 4.3: 3-phase Sine PWM for 1 cycle of the sine wave

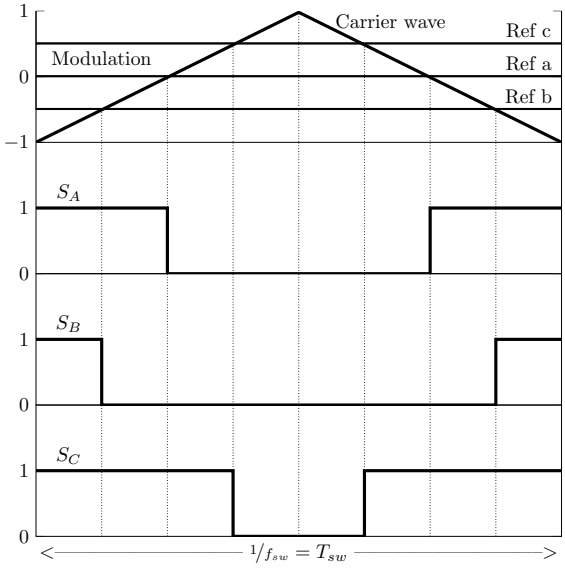


Figure 4.4: Sub-interval diagram of 3-phase Sine PWM used for AFE

The switching combination for 2-level VSR will render 8 different possibilities of switching states, which give 8 different sets of equations depending on the switch state. The advantage of PINN is that the ODE equations are used as a constraint to reduce the solution space. Adding to it, the dynamics of the converter are preserved in the measurement. Hence it is sufficient to just derive the average switching equation for the PINN model. The 3-phase grid voltage is defined by the equation 4.1. It is assumed during the training that the AFE is in a steady state and the 3-phase grid voltage and currents are balanced. The equations modelled for the AFE in figure 4.1 are given by 4.2 and 4.5.

$$\begin{aligned} u_a &= V_m Cos(\omega_0 t) \\ u_b &= V_m Cos(\omega_0 t - 120^\circ) \\ u_c &= V_m Cos(\omega_0 t + 120^\circ) \end{aligned} \tag{4.1}$$

$$\begin{aligned} \frac{di_a}{dt} &= \frac{1}{L_a}\left(u_a - R_{L_a} i_a - S_a V_{dc}\right) \\ \frac{di_b}{dt} &= \frac{1}{L_b}\left(u_b - R_{L_b} i_b - S_b V_{dc}\right) \\ \frac{di_c}{dt} &= \frac{1}{L_c}\left(u_c - R_{L_c} i_c - S_c V_{dc}\right) \end{aligned} \tag{4.2}$$

$$i_{dc} = S_a i_a + S_b i_b + S_c i_c \tag{4.3}$$

$$i_{load} = \frac{V_{dc}}{R_{load}} \tag{4.4}$$

$$\begin{aligned} \frac{dV_{dc}}{dt} &= \frac{1}{C}\left(i_{dc} - i_{load}\right) \\ &= \frac{1}{C}\left(S_a i_a + S_b i_b + S_c i_c - \frac{V_{dc}}{R_{load}}\right) \end{aligned} \tag{4.5}$$

4.2. PINN MODELLING OF AFE CONVERTER

Parameters to be estimated for the AFE converter are the filter inductance (L_a, L_b and L_c) & its parasitic resistance (R_a, R_b and R_c), DC-link capacitance (C) and the load resistance (R_{load}). It is assumed that the grid voltage(u_{abc}), input currents(i_{abc}) and the output DC voltage(V_{dc}) are measurable. Based on the derived equations 4.1, 4.2 and 4.5, and using the general form defined in equation 2.1, PINN model equations of the AFE converter

can be derives as below:

$$\begin{aligned}
\frac{dv_{dc}}{dt} + \mathcal{N}[v_{dc};\lambda] = 0 \\
\frac{di_a}{dt} + \mathcal{N}[i_a;\lambda] = 0 \\
\frac{di_b}{dt} + \mathcal{N}[i_b;\lambda] = 0 \\
\frac{di_c}{dt} + \mathcal{N}[i_c;\lambda] = 0
\end{aligned} \tag{4.6}$$

Where:

$$\lambda = \{L_a, L_b, L_c, R_{L_a}, R_{L_b}, R_{L_c}, C, R_{load}\}$$

$$\begin{aligned}
\mathcal{N}[v_{dc};\lambda] &= \frac{1}{C}\left(\frac{V_{dc}}{R_{load}} - S_a i_a - S_b i_b - S_c i_c\right) \\
\mathcal{N}[i_a;\lambda] &= \frac{1}{L_a}\left(R_{L_a} i_a + S_a V_{dc} - u_a\right) \\
\mathcal{N}[i_b;\lambda] &= \frac{1}{L_b}\left(R_{L_b} i_a + S_b V_{dc} - u_b\right) \\
\mathcal{N}[i_c;\lambda] &= \frac{1}{L_c}\left(R_{L_c} i_a + S_c V_{dc} - u_c\right)
\end{aligned} \tag{4.7}$$

4.3. DISCRETIZATION OF AFE CONVERTER AND LOSS FUNCTION

Using the general equation 2.3, and the equations 4.9 and 4.8 forward equation for backward equation for AFE converter can be formulated as below:

Backward equations for AFE converter:

$$\begin{aligned}
v_{dc}(t_{n+c_i}) &= v_{dc}(t_n) - \Delta t \sum_{j=1}^{q} a_{ij} \mathcal{N}[v_{dc}(t_{n+c_j}); \lambda], \quad i = 1,2...q \\
i_a(t_{n+c_i}) &= i_a(t_n) - \Delta t \sum_{j=1}^{q} a_{ij} \mathcal{N}[i_a(t_{n+c_j}); \lambda], \quad i = 1,2...q \\
i_b(t_{n+c_i}) &= i_b(t_n) - \Delta t \sum_{j=1}^{q} a_{ij} \mathcal{N}[i_b(t_{n+c_j}); \lambda], \quad i = 1,2...q \\
i_c(t_{n+c_i}) &= i_c(t_n) - \Delta t \sum_{j=1}^{q} a_{ij} \mathcal{N}[i_c(t_{n+c_j}); \lambda], \quad i = 1,2...q
\end{aligned} \tag{4.8}$$

Forward equations for AFE converter:

$$\begin{aligned}
v_{dc}(t_{n+c_i}) &= v_{dc}(t_{n+1}) - \Delta t \sum_{j=1}^{q} (a_{ij} - b_j) \mathcal{N}[v_{dc}(t_{n+c_j}); \lambda], \quad i = 1,2...q \\
i_a(t_{n+c_i}) &= i_a(t_{n+1}) - \Delta t \sum_{j=1}^{q} (a_{ij} - b_j) \mathcal{N}[i_a(t_{n+c_j}); \lambda], \quad i = 1,2...q \\
i_b(t_{n+c_i}) &= i_b(t_{n+1}) - \Delta t \sum_{j=1}^{q} (a_{ij} - b_j) \mathcal{N}[i_b(t_{n+c_j}); \lambda], \quad i = 1,2...q \\
i_c(t_{n+c_i}) &= i_c(t_{n+1}) - \Delta t \sum_{j=1}^{q} (a_{ij} - b_j) \mathcal{N}[i_c(t_{n+c_j}); \lambda], \quad i = 1,2...q
\end{aligned} \tag{4.9}$$

Loss function for AFE converter depends on the prediction of currents i_a, i_b, i_c and output voltage v_{out}, and it is defined based on general loss function expressed in equation 4.10.

$$\begin{aligned}
\phi_{AFE} = &\sum_n \left[\left(v_{dc}(t_n) - v_{dc_{c_i}}(t_n) \right)^2 + \left(v_{dc}(t_{n+1}) - v_{dc_{c_i}}(t_{n+1}) \right)^2 \right] \\
&+ \sum_n \left[\left(i_{L_a}(t_n) - i_{L_{a_{c_i}}}(t_n) \right)^2 + \left(i_{L_a}(t_{n+1}) - i_{L_{a_{c_i}}}(t_{n+1}) \right)^2 \right] \\
&+ \sum_n \left[\left(i_{L_b}(t_n) - i_{L_{b_{c_i}}}(t_n) \right)^2 + \left(i_{L_b}(t_{n+1}) - i_{L_{b_{c_i}}}(t_{n+1}) \right)^2 \right] \\
&+ \sum_n \left[\left(i_{L_c}(t_n) - i_{L_{c_{c_i}}}(t_n) \right)^2 + \left(i_{L_c}(t_{n+1}) - i_{L_{c_{c_i}}}(t_{n+1}) \right)^2 \right], \quad i = 1,2...q
\end{aligned} \tag{4.10}$$

4.4. DATA ACQUISITION AND CONFIGURATION OF PINN FOR THE AFE CONVERTER

4.4.1. DATA ACQUISITION

To collect data for the AFE converter, simulation of 2-Level VSR was performed in PLECS as represented in figure 4.2. Based on section 2.4.2, data is pre-processed in MATLAB to have forward data-set and backward data-set. Motive of this thesis is to develope a model which is data-efficient hence, data of 25 cycles of switching frequency is collected at the steady state.

4.4.2. DEFINING THE STRUCTURE OF PINN

Inputs for the NN are defined as input current(i_{abc}) and output voltage(V_{dc}) at time t_n. The size of the neural net is decided based on the methodology as described in section 2.4.5. The number of outputs of the latent state is decided by the q value. The value of q is selected as $q = 1$ based on derivation in section 2.4.3. True values of the AFE PINN model are input currents and output voltage at time t_n and t_{n+1}

Physics model of the AFE converter consists of ODE equation set 4.7. To couple the latent state and the true value via the physics model, forward equations 4.9 and back equations 4.8 are applied. Figure 4.5 shows the visualisation of the PINN model for the AFE converter.

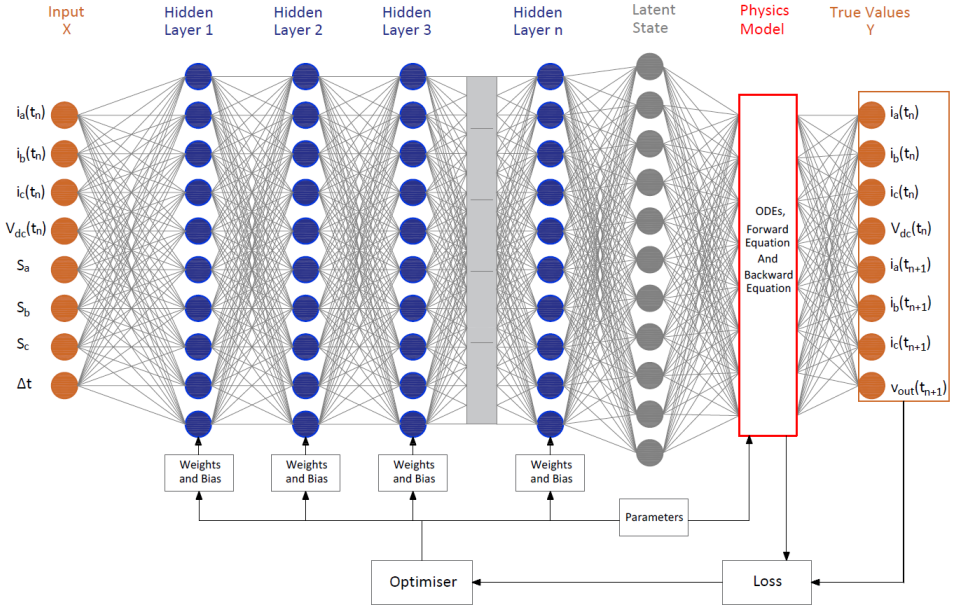


Figure 4.5: Architecture of PINN applied for the AFE Converter

4.5. Validation and Optimisation of the PINN for the AFE Converter

Stage 1: Verification of Physics Model

Case 1a: Adams Learning Rate = 0.001

To verify the physics model, based on the guidelines formulated in section 2.4.5, parameters L_a, R_a, L_b, R_b, L_c, R_c, C, R_{load} were initialised with their true values and in the TensorFlow computational graph, these parameters are set as non trainable. Only the neural net parameters (weights and biases) will be trained in this stage. Based on the boost PINN's experience, the neural net's initial size was set to 2 hidden layers with 200 neurons each. The learning rate of the Adams optimiser was set at 0.001 initially

Figure 4.6, shows the prediction of the PINN model for the backward and forward data set. It can be observed that the set learning rate of 0.001, which performed well for the boost converter, did not perform well in this case. There was a sufficient error being observed for the output voltage V_{dc}.

In the next cases, the learning rate will be gradually increased, and the error will be observed. This is done till the prediction curve matches the true curve with considerably lesser error.

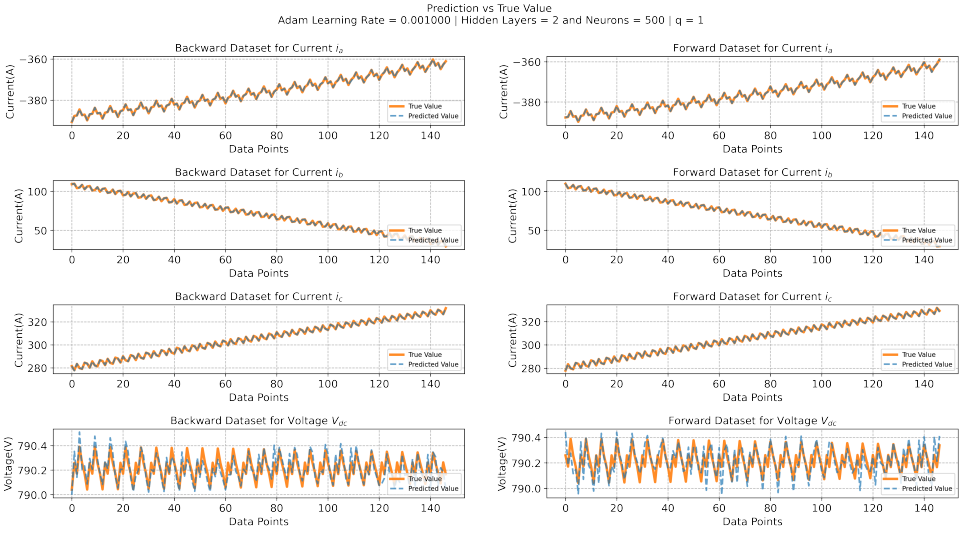


Figure 4.6: Case:1a Performance of AFE PINN model with parameters set as non-trainable and initialised with true values | Adams Learning Rate = 0.001

Case 1b: Adams Learning Rate = 0.002

As the prediction had a higher error, the learning rate of the Adams optimiser is increased to 0.002 in this case. From figure 4.7, it can be observed that the prediction error has reduced, yet there is a sufficient error.

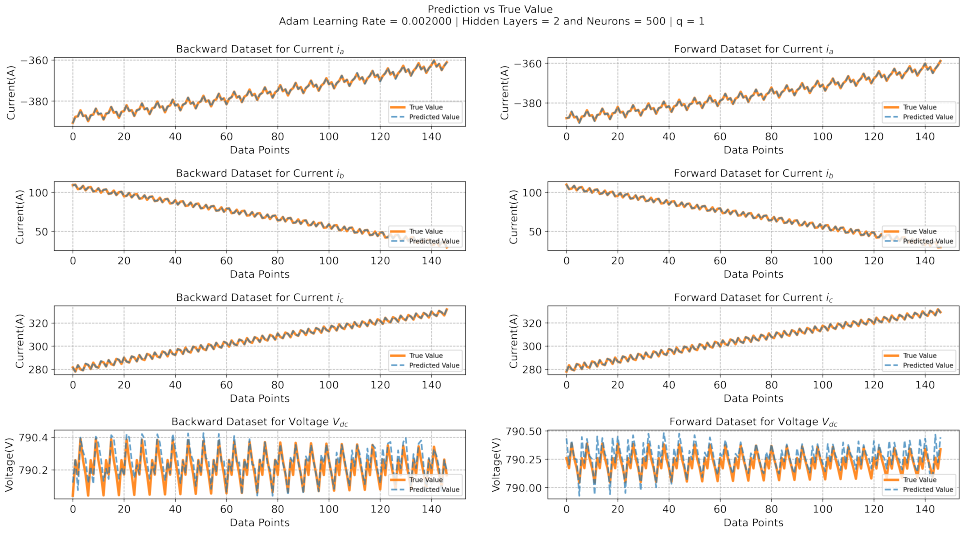


Figure 4.7: Case:1b Performance of AFE PINN model with parameters set as non-trainable and initialised with true values | Adams Learning Rate = 0.002

Case 1c: Adams Learning Rate = 0.004 and 0.005

Progressively, the error did reduce with an increase in the learning rate, but after 0.003, the model performance reduced. Figure 4.8 and 4.9 shows the output for learning rate of 0.004 and 0.005 respectively. It can be observed that the error is higher in prediction for forward and backward data-sets.

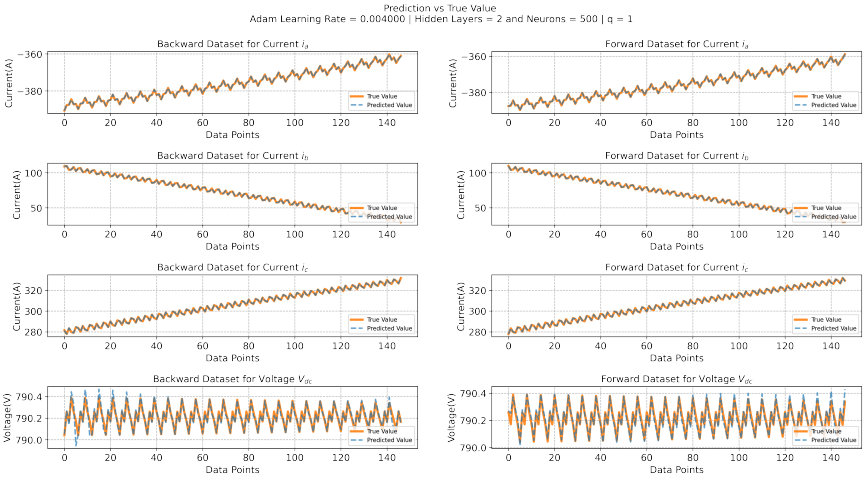


Figure 4.8: Case:1C Performance of AFE PINN model with parameters set as non-trainable and initialised with true values | Adams Learning Rate = 0.004

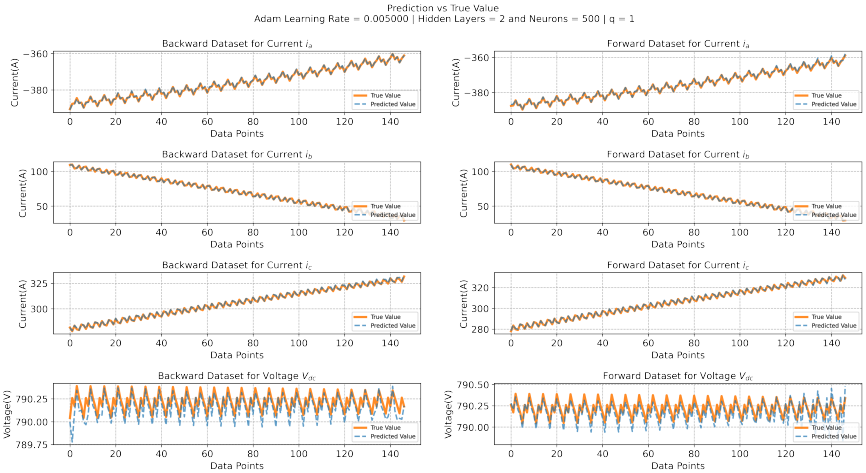


Figure 4.9: Case:1C Performance of AFE PINN model with parameters set as non-trainable and initialised with true values | Adams Learning Rate = 0.005

Case 1d: Adams Learning Rate = 0.003

Optimal performance is observed when the learning rate is set to 0.003. The prediction error of the PINN is the least as shown in figure 4.11 and can predict the current and voltage waveform perfectly, as shown in figure 4.12. From figure 4.10, it can be observed that the model did converge to a satisfactory loss value. This validates the physics model developed for the AFE. This means that the PINN model could correlate the latent states predicted by the NN to the values of currents and voltages at t_n and t_{n+1} via the forward and backward equations modelled. In the next step, the convergence of the PINN model will be verified.

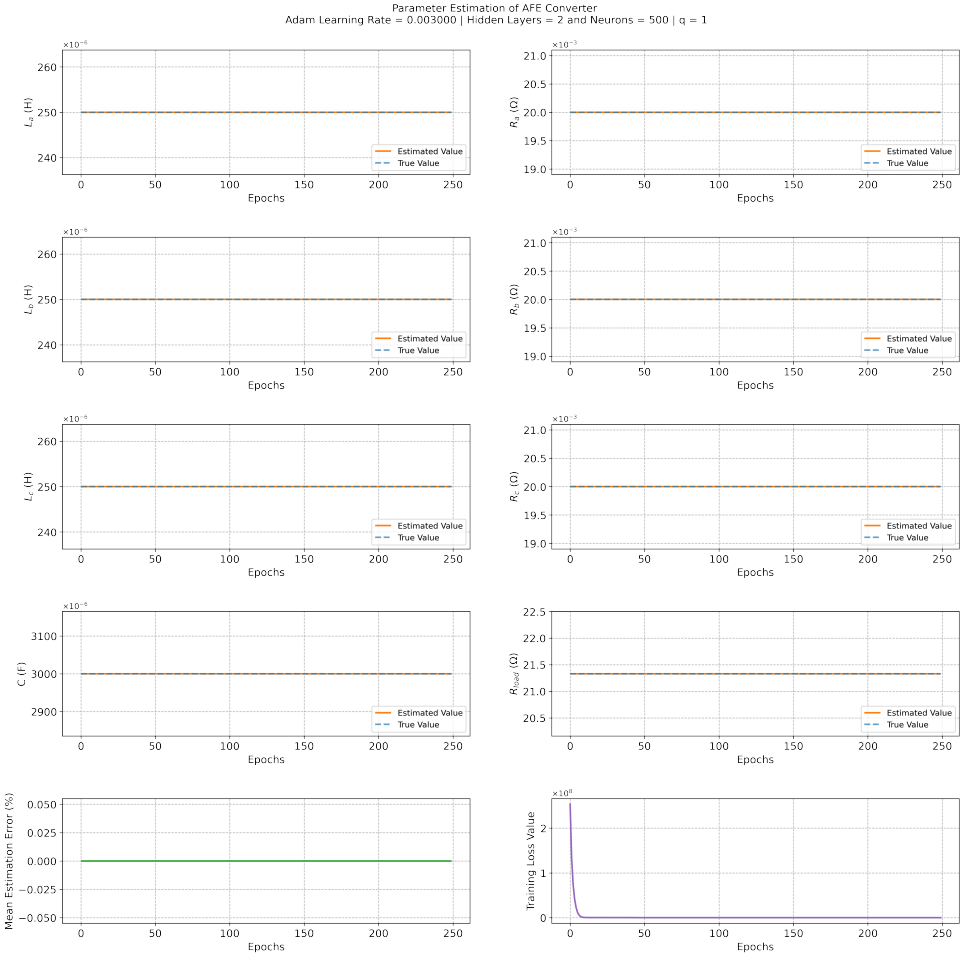


Figure 4.10: Case:1d Convergence of the loss value with physics parameters set as non-trainable and initialised with true values | Adams Learning Rate = 0.003

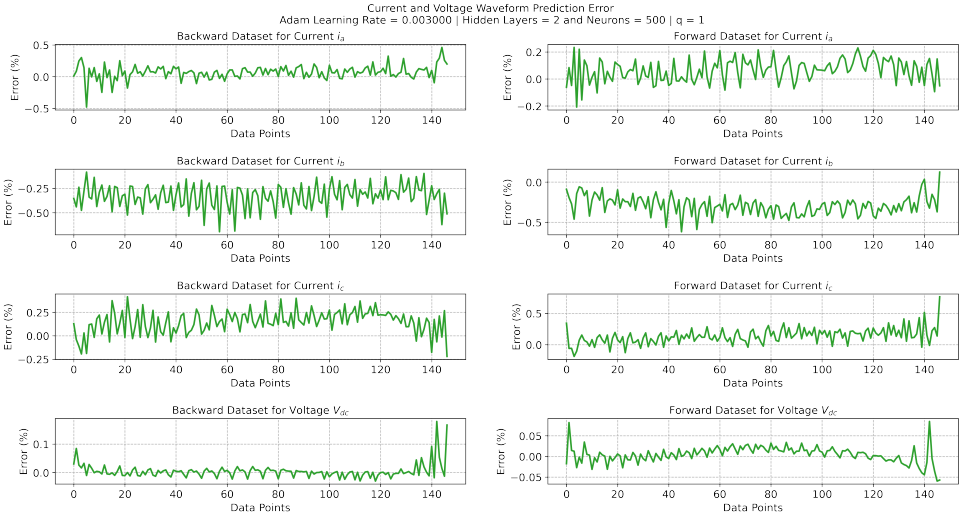


Figure 4.11: Case:1d Convergence of the loss value with parameters set as non-trainable and initialised with true values | Adams Learning Rate = 0.003

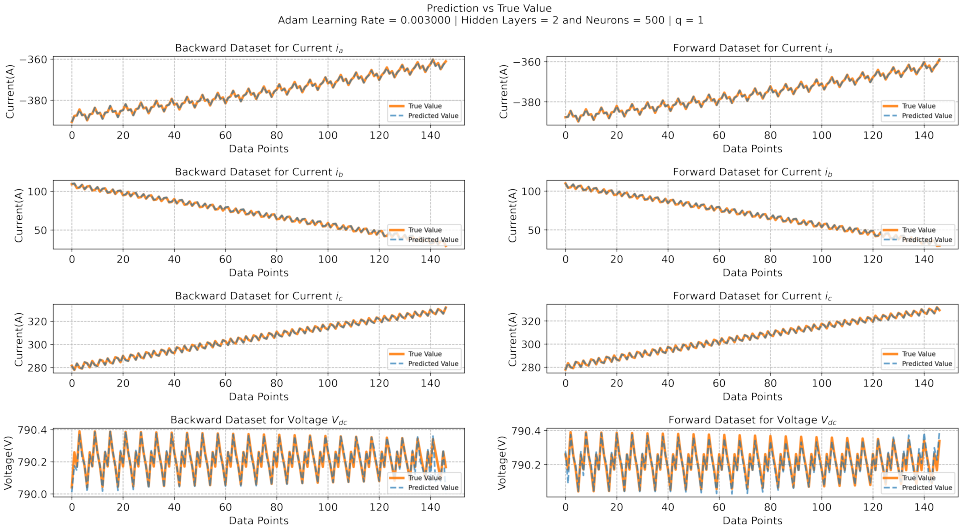


Figure 4.12: Case:1d Performance of AFE PINN model with parameters set as non-trainable and initialised with true values | Adams Learning Rate = 0.003

STAGE 2: VERIFICATION OF CONVERGENCE OF PINN

In the second stage, the parameters of the physics model for AFE were randomly initialised and set as trainable parameters. Now, the optimizer will try to estimate these parameters during the training process. Evolution and convergence of the percentage error value of the parameters (L_a, R_a, L_b, R_b, L_c, R_c, C, R_{load}) is observed. The first step is to perform the pseudo normalisation of the Physics parameters so that they lie within a similar range of bounds as show in table 4.2

Table 4.2: Pseudo Normalisation of Physics Parameters for AFE Converter

Parameter	Actual Value		Scaling for TensorFlow Variable		Re-scaling for Physics Equation
L_{abc} (H)	250μ	2.5×10^{-4}	$ln(2.5)$	0.916290732	$e^{0.92} \times 10^{-4}$
C (F)	3000μ	3×10^{-3}	$ln(3)$	1.098612289	$e^{1.09} \times 10^{-3}$
$R_{load}(\Omega)$	21.33	2.133×10	$ln(2.133)$	0.757529439	$e^{0.76} \times 10$
$R_{abc}(\Omega)$	$20m$	2×10^{-2}	$ln(2)$	0.693147181	$e^{0.69} \times 10^{-2}$

In the intial trial run of parameter estimation, it was observed that the PINN model had issue with convergence. From figure 4.13, it can be inferred that the PINN model is not able to optimise the parameters of the physics model. The initial hypothesis is that the complexity of the AFE converter equations might be creating the issue. Because, unlike boost converter, a different situation has arisen here. Now there are four equations and eight parameters. Interdependence of parameters might create an optimisation issue. For example, a change in L_a, changes the current value i_a, which in turn affects the value of V_{dc}.

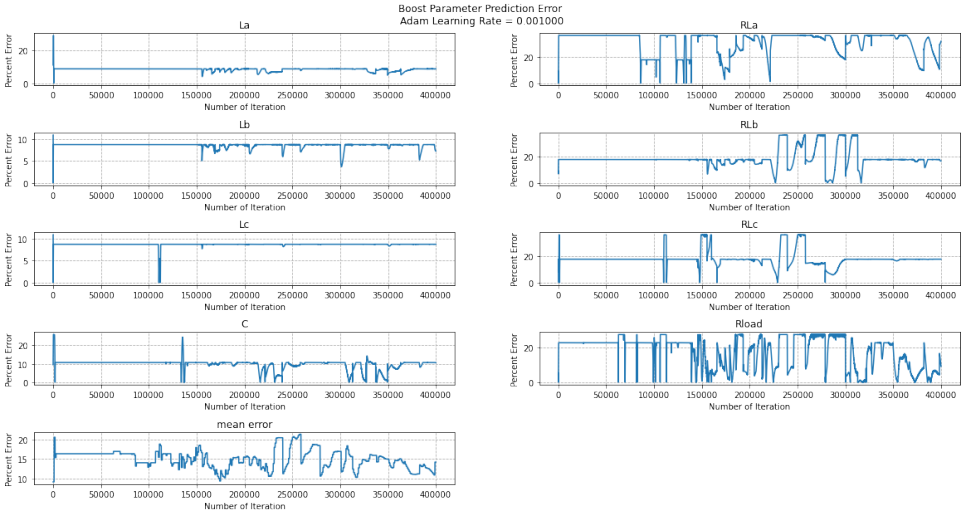


Figure 4.13: PINN parameter estimation error for the AFE converter

To investigate the hypothesis two test cases are proposed.

Test Case 1: Analysis of the PINN model by estimating just one parameter, while the rest of the parameters are being initialised with true value and set as non-trainable.

In this test case, only the phase-a inductance value is estimated (L_a) during the training process, while the rest of the parameters are initialised to the true value and set as non-trainable. Figure 4.14 shows the parameter estimation error for (L_a) and it is found to converge to a stable value of close to 10 percent error.

Test Case 2: Analysis of the PINN model by estimating two parameters, while the rest of the parameters are being initialised with true value and set as non-trainable.

In this test case, two of the parameters of the AFE converter are estimated. The phase-a and phase-b inductance values (L_a and L_a) are estimated during the training process, while the rest of the parameters are initialised to the true value and set as non-trainable. Figure 4.15 shows the parameter estimation error for L_a and L_b, and it is observed to show non-convergence behaviour and failed to converge to a stable error value.

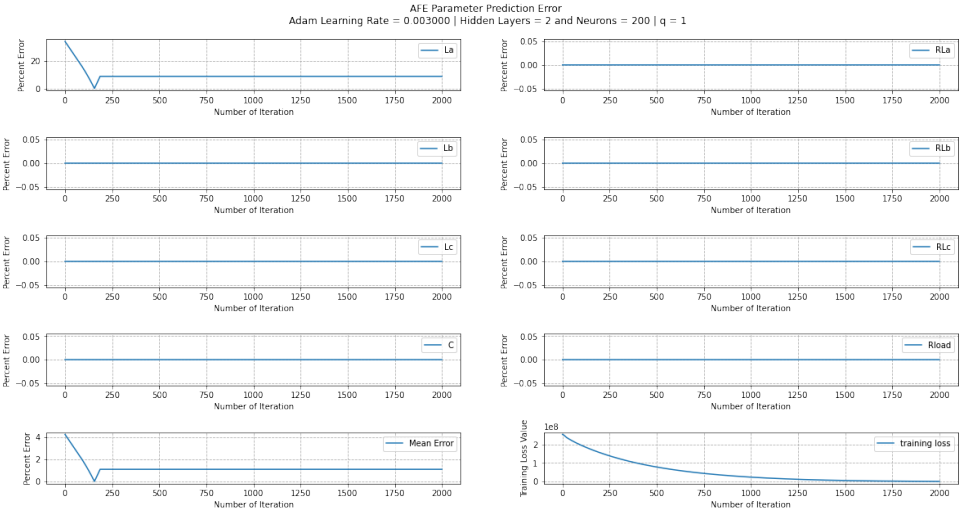


Figure 4.14: Test Case 1: AFE PINN model to estimate only La

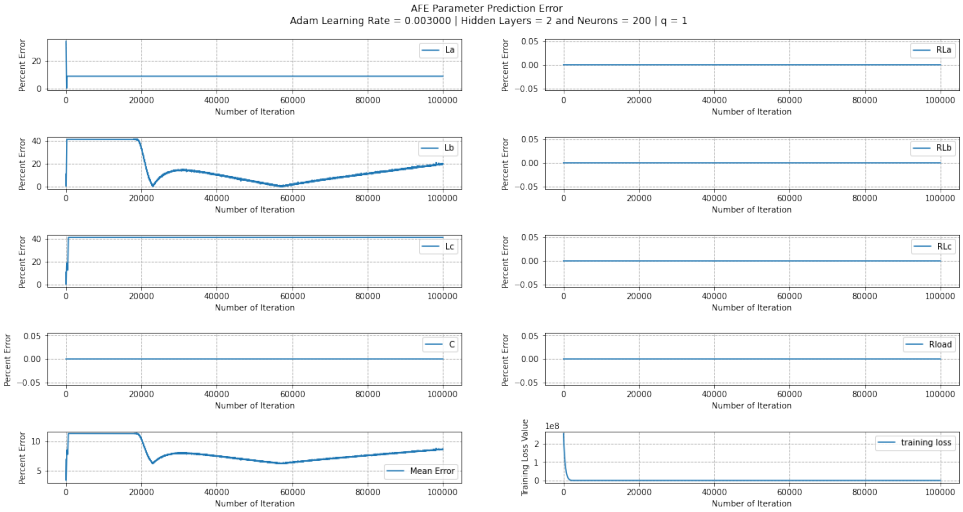


Figure 4.15: Test Case 2: AFE PINN model to estimate only La and Lb

This opens up the question on how else can the optimiser be tuned to tackle this scenario? After few trial and error approaches, following observations are made:

- Based on the work of Aditi et. al [30], full batch optimisation is used in this thesis instead of mini-batch. Which means the weights, biases and physics parameters are updated at every iteration pass. In full batch, the possibility to hit scaling ceiling is higher as observed in figure 4.13. Using mini-batch can help overcome this issue. Choosing batch size is experimental and after few trial and error a batch size of 100 is chosen.
- From TensorFlow documentation[31], it is suggested that the default value of epsilon set as e^{-7} may not be the optimal value. Smaller epsilon corresponds to larger weight updates and higher epsilon corresponds to smaller weight updates with slower training. Again, there is no deterministic way to set the epsilon value and hence with trial and error approach, epsilon value of e^{-4} is chosen.
- Presently, the physics parameters are defined as trainable tensorflow variable with a range bound of -1 to +1. It is possible to use activation function such as tanh or sigmoid to add non-linearity for the physics parameters during optimisation.
- From results of PINN for boost, it is observed that the neural net with smaller size had better performance. It makes sense as the optimiser will see the physics parameter as just another parameter in addition to weights. Having fewer weights in the neural net will provide higher possibility of convergence for physics parameter estimation.

4.6. RESULTS OF PINN MODEL FOR THE AFE CONVERTER

Implimenting the previously discussed changes, the PINN model for the AFE converter is set at adams learning rate = 0.003, epsilon = e^{-4}, batch size of 100, and tanh activation function for the physics parameters. The neural net with 1 hidden layer and 50 neurons is selected. Figure 4.17 shows the result for PINN model of the AFE converter with the following results as tabulated in table 4.3. The execution time for the PINN model is 361 seconds for $2x10^5$ iterations with batch size of 100 resulting in 2000 Epochs.

Table 4.3: Parameter Estimation of AFE Converter

	L_a	R_a	L_b	R_b	L_c	R_c	C	R_{load}	mean
Error (%)	7.4	38.2	0.3	24	7.4	61.8	29.7	73.2	30.25

Figure 4.16 shows the predicted waveform for forward and backward data-set. With higher error in estimation of C and R_{load} it is expected to have higher prediction error for the V_{dc} voltage waveform. Similar to boost converter, PINN model with larger neural net size have higher error for AFE converter as well. Figure 4.18 shows the result of estimation with 5 hidden layer and 200 neurons per hidden layer.

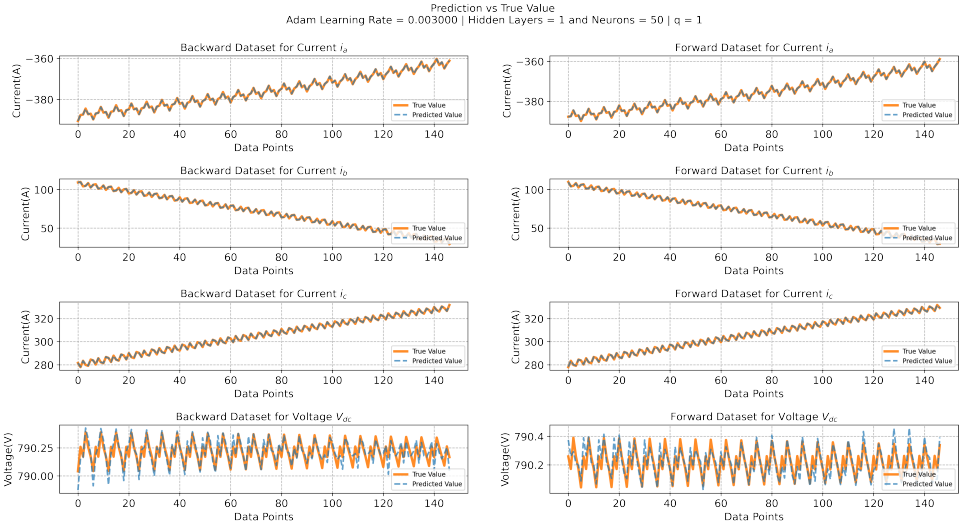


Figure 4.16: Current and Voltage waveform prediction for backward and forward data-set of AFE converter with 1 Hidden Layer and 50 Neurons

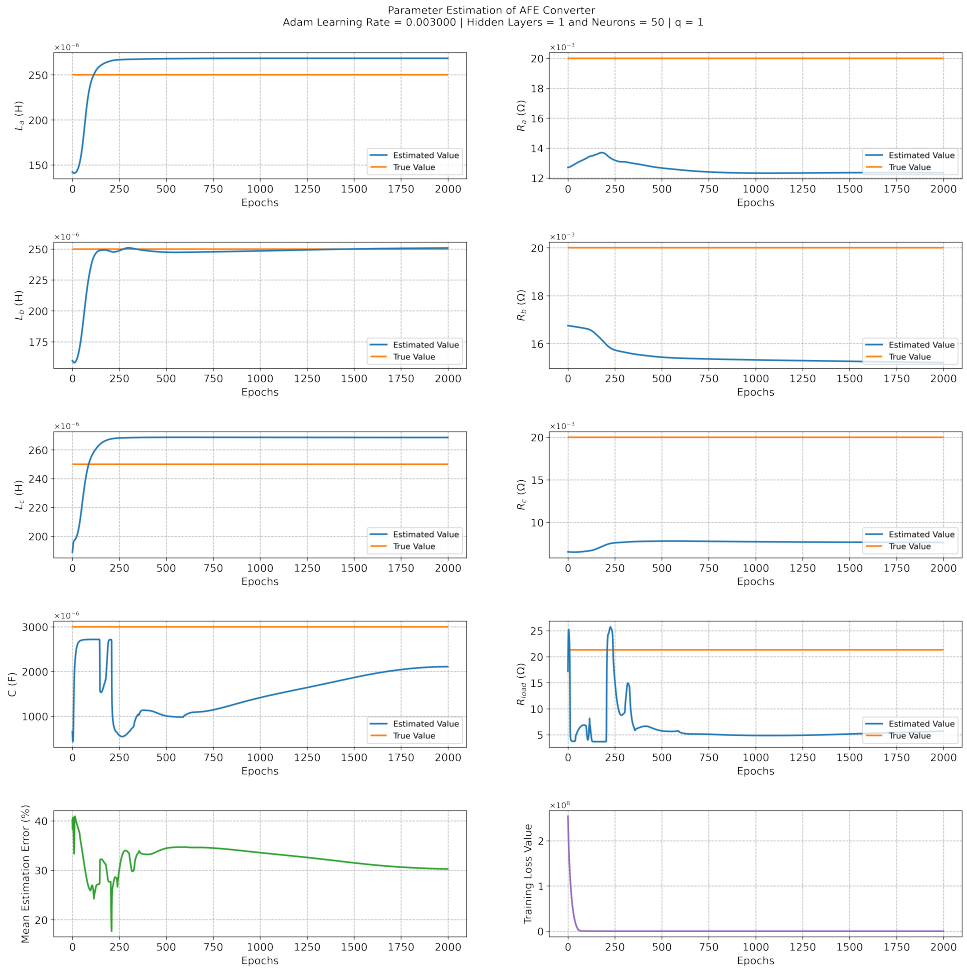


Figure 4.17: PINN parameter estimation of the AFE converter with 1 Hidden Layer and 50 Neurons

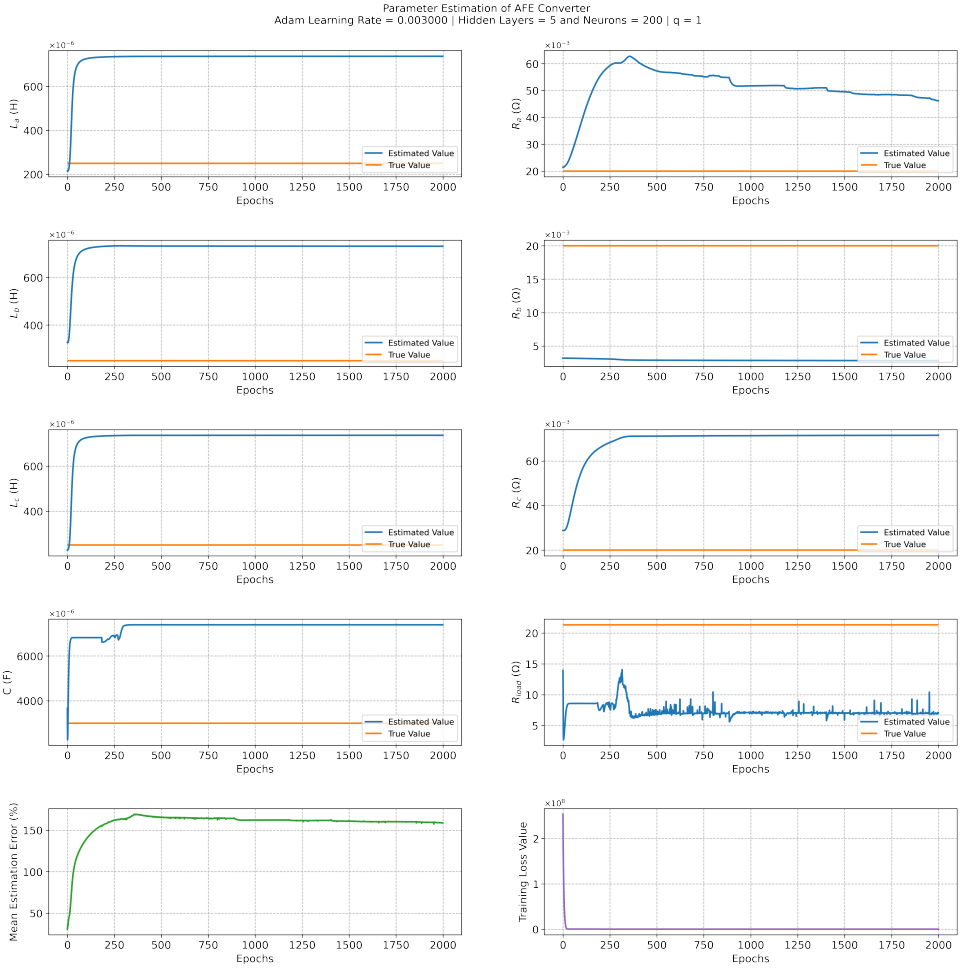


Figure 4.18: PINN parameter estimation of the AFE converter with 5 Hidden Layer and 200 Neurons

5

CONCLUSION

Impedance-based current harmonic analysis requires knowing the circuit parameters of the FCS. Being a confidential design, manufacturers of the FCS don't reveal the convert's design parameters. Developing a non-invasive model that can estimate the converter's parameters with available measurement data is necessary.

Data-driven machine learning approaches require a tremendous amount of data to perform satisfactorily. The power electronics domain has no such existing database repositories which can be used to develop machine learning models to estimate the parameters of a given power converter. The scarcity of available databases and the non-availability of the FCS circuit parameters dictate the development of a highly data-efficient machine learning model to estimate the converter's parameters.

This is the key point of focus in this thesis, and a physics-informed neural net model was developed and implemented to estimate the parameters of the boost converter and the AFE converter of the FCS. It was a challenging journey as the concept of PINN was introduced in 2019, and the application of PINN in the power electronics domain had just one research paper published during this thesis.

The PINN model developed for the boost converter had a mean estimation error of 0.89%, and the PINN model estimated the parameters in 25.76s. For the AFE converter, optimising the PINN model was challenging due to the physics model's complexity. The PINN model for the AFE converter had a mean estimation error of 30.25%, with an execution time of 361s.

5.1. REVISITING THE RESEARCH QUESTION

This thesis aimed to answer the following research questions:

How to apply the PINN-based approach to the parameter estimation of a power electronic converter?

First, the converter's ODEs containing the estimated parameters are derived. In the ODE equation, the current and voltage values must be measurable from peak to peak. In the second step, a neural network predicts the latent intermittent state values between the peak-to-peak measured values. After discretizing the measured data, the latent state of the neural net could be coupled to the measured values via the physics model using the Runge-Kutta method. The final step involves using the neural net's optimiser to predict the physics parameters.

How to implement the PINN-based estimation approach?
To collect the data, simulations are performed in PLECS and MATLAB is used to pre-process the data. Python(v3.6) with TensorFlow(v1.15) backend is used to develop the neural net and physics model. The TensorFlow computation graph method provides the flexibility to define the neural net from scratch. Using this, the physics parameters are defined as a trainable TensorFlow variable, which is called by the optimiser to estimate the parameters during the training of the PINN model.

What are the challenges when applying PINN and how to address them?
Implementation of the physics model within the neural net is an open problem. Hence the parameters are estimated indirectly via the optimisation process where the parameters are treated just like weights and biases. Weights are biases are normalised values in the range of -1 to 1 for the tanh activation function, unlike the physics parameters. Pseudo-normalisation is used to solve this problem. Additionally, hyperparameters of the optimiser have no deterministic way to set the values. This requires a trial and error method to achieve optimal performance.

5.2. DISCUSSION

1. It was recognized during the literature review that impedance modelling is essential for power quality analysis and that circuit parameters are not readily available. It was also discovered that physics-informed neural networks can be used to estimate parameters. One of the major gaps is the lack of research on PINN modeling implementation in the power electronics domain. The PINN model was successfully applied to the boost converter and the AFE converter to estimate circuit parameters. The PINN model developed in this thesis is highly data-efficient, quick and non-invasive.

2. The concept of PINN is quite recent. During the initial phase of this thesis, the modelling direction was to integrate the physics model within the neural net. Nevertheless, it remains an open question. Hence, the neural net is used to predict the latent state for the current and voltage waveform. Using this latent state, the physics model is coupled to the neural net using the forward and backward equations derived. During the coupling process, the optimiser of the neural net is employed to estimate the physics parameter.

3. The existing works on PINN, such as [24], use two optimisers with default hyperparameter settings. The Adams optimiser is used to update the weights, biases, and physics parameters for the first few iterations. After fixing the weights and biases, only the physics parameters are estimated via the L-BFGS optimiser in the next

training iterations. This approach is resource-intensive and time-consuming.

In this thesis, only the Adams optimiser is used, and hyper-parameters of the optimiser are tuned to estimate the physics parameters. Consequently, the model converged in 25.76 seconds for the boost converter, and in 6 minutes for the more complex AFE converter.

4. It is crucial to initialize the physics parameters randomly. The PINN model in this thesis assumes that the circuit parameters are unavailable, and the loss function is defined using only the measured current and voltage data. Hence the error of the parameters estimated is unsupervised.

   Unlike conventional data-driven neural net models, the PINN model developed should go through the training process every time to estimate the physics parameters of the circuit. And, during the training process the optimiser estimates the physics parameters. It is not possible to re-use the trained model for a different circuit.

5.3. FUTURE SCOPE OF WORK

This thesis introduces the concept of Physics-Informed Neural Networks, which can be used to estimate parameters for two power converters. This concept of PINN is at the beginning of its development, and the work in this thesis provides a solid foundation for future work. Below are some recommendations for future research based on this thesis.

1. In this thesis, only the circuit parameters of the converter are estimated. Extending this method to estimate the control parameters would be beneficial.
2. Parameter estimation in this was performed via the physics model using the optimiser outside of the neural network. Integrating the physics model within the neural net would be an interesting concept to research. This approach would solve an open problem in the domain.
3. Current work is developed completely based on simulation. Observing how the model will perform in hardware implementation facing a more realistic environment will be interesting.
4. Parameter estimation can be used for health monitoring the converter; it would be beneficial if the PINN model is implemented to work on an online platform providing real-time estimation.
5. Currently, the optimiser used for the estimation of parameters is developed very specific to solve neural network problems. For instance, the physics parameters will be treated similarly to the weights and biases of the neural network. The value of weights and biases are invaluable; hence adding a priority to the physics model parameters will be an interesting approach.

BIBLIOGRAPHY

[1] *Carbon pollution from transportation.* [Online]. Available: https://www.epa.gov/transportation-air-pollution-and-climate-change/carbon-pollution-transportation.

[2] *World energy investment 2022 – analysis.* [Online]. Available: https://www.iea.org/reports/world-energy-investment-2022.

[3] *Policies to promote electric vehicle deployment.* [Online]. Available: https://www.iea.org/reports/global-ev-outlook-2021/policies-to-promote-electric-vehicle-deployment.

[4] B. Walton, J. Hamilton, G. Alberts, S. F. Smith, E. Day, and J. Ringrow, *Electric vehicles*, Jul. 2020. [Online]. Available: https://www2.deloitte.com/us/en/insights/focus/future-of-mobility/electric-vehicle-trends-2030.html.

[5] L. Wang, Z. Qin, T. Slangen, P. Bauer, and T. van Wijk, "Grid impact of electric vehicle fast charging stations: Trends, standards, issues and mitigation measures - an overview," *IEEE Open Journal of Power Electronics*, vol. 2, pp. 56–74, 2021. DOI: 10.1109/OJPEL.2021.3054601.

[6] H. Tu, H. Feng, S. Srdic, and S. Lukic, "Extreme fast charging of electric vehicles: A technology overview," *IEEE Transactions on Transportation Electrification*, vol. 5, no. 4, pp. 861–878, 2019. DOI: 10.1109/TTE.2019.2958709.

[7] J. Sun, "Impedance-based stability criterion for grid-connected inverters," *IEEE Transactions on Power Electronics*, vol. 26, no. 11, pp. 3075–3078, 2011. DOI: 10.1109/TPEL.2011.2136439.

[8] B. Wen, D. Boroyevich, R. Burgos, P. Mattavelli, and Z. Shen, "Analysis of d-q small-signal impedance of grid-tied inverters," *IEEE Transactions on Power Electronics*, vol. 31, no. 1, pp. 675–687, 2016. DOI: 10.1109/TPEL.2015.2398192.

[9] S. Yang, D. Xiang, A. Bryant, P. Mawby, L. Ran, and P. Tavner, "Condition monitoring for device reliability in power electronic converters: A review," *IEEE Transactions on Power Electronics*, vol. 25, no. 11, pp. 2734–2752, 2010. DOI: 10.1109/TPEL.2010.2049377.

[10] J. Huang, K. A. Corzine, and M. Belkhayat, "Small-signal impedance measurement of power-electronics-based ac power systems using line-to-line current injection," *IEEE Transactions on Power Electronics*, vol. 24, no. 2, pp. 445–455, 2009. DOI: 10.1109/TPEL.2008.2007212.

[11] S. Neshvad, S. Chatzinotas, and J. Sachau, "Wideband identification of power network parameters using pseudo-random binary sequences on power inverters," *IEEE Transactions on Smart Grid*, vol. 6, no. 5, pp. 2293–2301, 2015. DOI: 10.1109/TSG.2015.2397552.

[12] M. Esparza, J. Segundo, C. Gurrola-Corral, N. Visairo-Cruz, E. Bárcenas, and E. Barocio, "Parameter estimation of a grid-connected vsc using the extended harmonic domain," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 8, pp. 6044–6054, 2019. DOI: 10.1109/TIE.2018.2870404.

[13] Z.-H. Liu, H.-L. Wei, Q.-C. Zhong, K. Liu, X.-S. Xiao, and L.-H. Wu, "Parameter estimation for vsi-fed pmsm based on a dynamic pso with learning strategies," *IEEE Transactions on Power Electronics*, vol. 32, no. 4, pp. 3154–3165, 2017. DOI: 10.1109/TPEL.2016.2572186.

[14] L. Chang, X. Jiang, M. Mao, and H. Zhang, "Parameter identification of controller for photovoltaic inverter based on l-m method," in *2018 IEEE International Power Electronics and Application Conference and Exposition (PEAC)*, 2018, pp. 1–6. DOI: 10.1109/PEAC.2018.8590362.

[15] H. Soliman, I. Abdelsalam, H. Wang, and F. Blaabjerg, "Artificial neural network based dc-link capacitance estimation in a diode-bridge front-end inverter system," in *2017 IEEE 3rd International Future Energy Electronics Conference and ECCE Asia (IFEEC 2017 - ECCE Asia)*, 2017, pp. 196–201. DOI: 10.1109/IFEEC.2017.7992442.

[16] T. Kamel, Y. Biletskiy, and L. Chang, "Capacitor aging detection for the dc filters in the power electronic converters using anfis algorithm," in *2015 IEEE 28th Canadian Conference on Electrical and Computer Engineering (CCECE)*, 2015, pp. 663–668. DOI: 10.1109/CCECE.2015.7129353.

[17] H. Soliman, P. Davari, H. Wang, and F. Blaabjerg, "Capacitance estimation algorithm based on dc-link voltage harmonics using artificial neural network in three-phase motor drive systems," in *2017 IEEE Energy Conversion Congress and Exposition (ECCE)*, 2017, pp. 5795–5802. DOI: 10.1109/ECCE.2017.8096961.

[18] S. Zhao, F. Blaabjerg, and H. Wang, "An overview of artificial intelligence applications for power electronics," *IEEE Transactions on Power Electronics*, vol. 36, no. 4, pp. 4633–4658, 2021. DOI: 10.1109/TPEL.2020.3024914.

[19] L. von Rueden, S. Mayer, K. Beckh, *et al.*, "Informed machine learning - a taxonomy and survey of integrating prior knowledge into learning systems," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2021. DOI: 10.1109/TKDE.2021.3079836.

[20] F. Chollet, *Deep Learning with Python*, 2nd ed. Shelter Island: Manning, 2021.

[21] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations," *arXiv preprint arXiv:1711.10561*, 2017.

[22] ——, "Physics informed deep learning (part ii): Data-driven discovery of nonlinear partial differential equations," *arXiv preprint arXiv:1711.10566*, 2017.

[23] A. Daw, J. Bu, S. Wang, P. Perdikaris, and A. Karpatne, *Rethinking the importance of sampling in physics-informed neural networks*, Jul. 2022. [Online]. Available: https://arxiv.org/abs/2207.02338.

[24] S. Zhao, Y. Peng, Y. Zhang, and H. Wang, "Parameter estimation of power electronic converters with physics-informed machine learning," *IEEE Transactions on Power Electronics*, vol. 37, no. 10, pp. 11 567–11 578, 2022. DOI: 10.1109/TPEL.2022.3176468.

[25] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, *Physics-informed machine learning*, May 2021. [Online]. Available: https://www.nature.com/articles/s42254-021-00314-5.

[26] J. C. Butcher, *Numerical Methods for Ordinary Differential Equations*, 3rd ed. England: John Wiley and Sons, Ltd, 2016.

[27] *Basic issues in floating point arithmetic and error analysis*. [Online]. Available: https://people.eecs.berkeley.edu/~demmel/cs267/lecture21/lecture21.html.

[28] "Ieee standard for floating-point arithmetic," *IEEE Std 754-2008*, pp. 1–70, 2008. DOI: 10.1109/IEEESTD.2008.4610935.

[29] S. Cuomo, V. S. di Cola, F. Giampaolo, G. Rozza, M. Raissi, and F. Piccialli, *Scientific machine learning through physics-informed neural networks: Where we are and what's next*, Jun. 2022. [Online]. Available: https://arxiv.org/abs/2201.05624.

[30] A. S. Krishnapriyan, A. Gholami, S. Zhe, R. M. Kirby, and M. W. Mahoney, *Characterizing possible failure modes in physics-informed neural networks*, Nov. 2021. [Online]. Available: https://arxiv.org/abs/2109.01050.

[31] *Tf.keras.optimizers.adam :   tensorflow v2.10.0*. [Online]. Available: https://www.tensorflow.org/api_docs/python/tf/keras/optimizers/Adam.