Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Reliability & Risk Engineering

Tom Lausberg

Multi-period Optimal Power Flow with Physics-informed Graph Neural Networks

Master Thesis

RRE – Reliability and Risk Engineering Lab
Swiss Federal Institute of Technology (ETH) Zurich

Expert: Prof. Dr. Giovanni Sansavini
Supervisor: Anna Varbella

Zürich, September 9, 2024

Abstract

The AC Optimal Power Flow Problem with Unit Commitment (UC-ACOPF) presents a significant challenge in power systems management due to its non-linear, non-convex, and mixed-integer optimization nature. Transmission System Operators (TSOs) require the ability to solve the UC-ACOPF quickly and repeatedly to ensure a safe and stable grid. However, current methods are often too computationally expensive to meet these demands effectively. To address this issue, we propose a novel approach that leverages the capabilities of Physics-Informed Neural Networks (PINNs) and Graph Neural Networks (GNNs). Our research aims to develop a more efficient and scalable solution to the UC-ACOPF problem. To validate the effectiveness of our proposed method, we will conduct a comprehensive comparison between our model and a traditional solver utilizing the Alternating Direction Method of Multipliers (ADMM).

Acknowledgements

I would like to express my sincere gratitude to my supervisors, Anna Varbella and Laya Das, for their guidance and support. Their expertise and insights have been invaluable to the development of this thesis. Additionally, I extend my gratitude to Prof. Dr. Giovanni Sansavini for providing me with the opportunity to conduct my thesis research within the Reliability and Risk Engineering laboratory. Finally, I would like to express my profound appreciation to my family for their constant support, understanding, and encouragement throughout my studies. Their unwavering belief in me has been a source of strength and motivation.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The growing complexity of modern power systems, driven by the integration of renewable energy sources, evolving demand patterns, and the need for enhanced grid reliability, necessitates advanced methods for optimal power flow (OPF) management. Among these, Multiperiod Alternating Current Optimal Power Flow (MP-ACOPF) with Unit Commitment (UC-ACOPF) stands at the heart of Transmission System Operator (TSO) power markets, playing a crucial role in both operational and planning stages. First formulated in 1962 [2], ACOPF has since been a fundamental tool for ensuring the optimal operation of power systems. It is solved annually for system planning, daily for day-ahead markets, and even on an hourly or minute-by-minute basis for real-time stability calculations.

Recent advances in computing power and algorithmic development have enabled more accurate modeling of constraints, reducing the need for simplifying approximations that have historically limited the precision of ACOPF solutions [3]. Despite these advances, a method for efficiently solving the full ACOPF problem across practical time scales remains elusive. Achieving such a breakthrough would not only allow for the optimal usage of grid resources but could also result in significant economic benefits, potentially saving billions of dollars annually while enhancing the reliability of power systems [4].

This thesis explores the theoretical foundations and proposes a novel approach to solving the Unit Commitment problem. We begin by providing a precise definition of the problem and discussing various traditional solution approaches. Our research then introduces an innovative method that combines Physics-Informed Neural Networks (PINNs) and Graph Neural Networks (GNNs). PINNs are neural networks designed to solve supervised learning tasks while respecting any given laws of physics described by general nonlinear partial differential equations. GNNs, on the other hand, are a class of deep learning methods designed to perform inference on data described by graphs. By leveraging these advanced techniques, we aim to enhance the efficiency and accuracy of Unit Commitment problem-solving. To validate our approach, we will compare our results to those obtained using a traditional solver employing the Alternating Direction Method of Multipliers (ADMM).

Chapter 2

Modelling

To solve the unit commitment problem with AC optimal power flow constraints we use a graph neural network with a physics informed loss function to explore the solution space. In the following section we describe the precise formulation of the problem, discuss the intricacies of the alternating current and unit commitment constraints and evaluate alternative approaches for solving this class of problems. We will then introduce to the concept of physics informed neural networks (PINNs) and graph neural networks (GNNs).

2.1 Optimal Power Flow

The optimal power flow problem is a fundamental concept in power systems engineering which involves determining the best operating levels for power plants on a specific grid to meet electricity demand while minimizing costs and ensuring system reliability.

We describe the grid as a graph consisting of a set $\mathcal{B}$ of buses which are connected by a set $\mathcal{L} \in \mathcal{B} \times \mathcal{B}$ transmission lines. $\mathcal{G}$ is a group of generators which can inject active and reactive power into the system. Each generator is associated to a bus in $\mathcal{B}$.

π-model for transmission lines

To accurately compute the optimal power flow problem we need to be able to compute both the flow and loss of active and reactive power along a transmission line. We model the transmission lines using the π-model. This is an algebraic lumped-circuit line model, which is the solution to the telegraph equations at the end of the lines. The π-model is a suitable model for this problem as it accurately computes necessary quantities without modelling switching transients or needing to solve the corresponding PDEs. Using the π-model we can describe a transmission line from bus k to bus m, and optionally associated transformer, with the following parameters:

- Voltage magnitudes at bus k,m: V_k, V_m [p.u.]
- Voltage angles at bus k,m: θ_k, θ_m [radians]

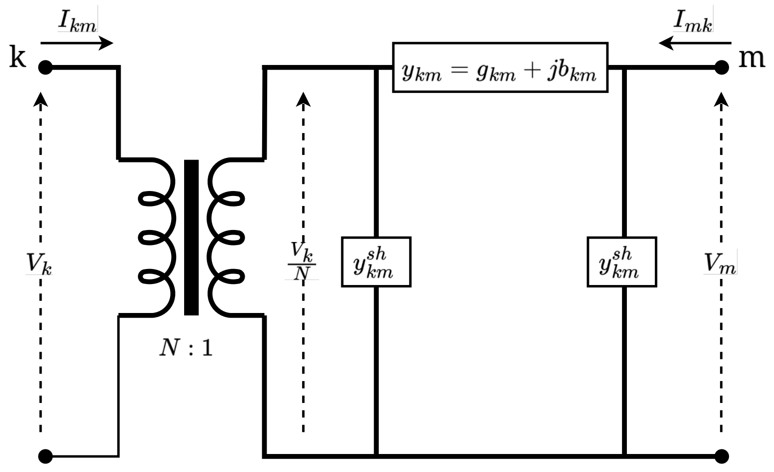


Figure 2.1: π-model

- Series impedance: $z_{km} = r_{km} + jx_{km}$ [Ω]
- Series admittance: $y_{km} = \frac{1}{z_{km}} = g_{km} + jb_{km}$ [S] consists of the conductance g_{km} [Ω] and susceptance b_{km} [S]
- Shunt admittance at each end of the transmission line: y^{sh}_{km}, y^{sh}_{mk} [S]
- Tap Ratio: $N = \tau e^{j\theta_{shift}}$ comprises of the magnitude τ and transformer phase shift θ_{shift}

These parameters can be combined into a admittance matrix for each branch.

$$Y^{br}_{km} = \begin{pmatrix} (y_{km} + y^{sh}_{km})\frac{1}{\tau^2} & -y_{km}\frac{1}{\tau e^{-j\theta_{shift}}} \\ -y_{km}\frac{1}{\tau e^{-j\theta_{shift}}} & y_{km} + y^{sh}_{km} \end{pmatrix} \tag{2.1}$$

For a full derivation of the π-model and admittance matrix see the Matpower manual 8.0 chapter 3 [5].

Power Flow Equations

Using this formulation we can calculate the current by multiplying the branch admittance matrix with the voltage phasor at each end of the branch.

$$\begin{pmatrix} I_{km} \\ I_{mk} \end{pmatrix} = Y^{br}_{km} \begin{pmatrix} V_k e^{j\theta_k} \\ V_m e^{j\theta_m} \end{pmatrix} \tag{2.2}$$

We can then compute the apparent power flow by multiplying the voltage with the complex conjugate of the current.

$$S_{km} = V_k e^{j\theta_k} I_{km}^* \tag{2.3}$$
$$S_{mk} = V_m e^{j\theta_m} I_{mk}^* \tag{2.4}$$

The complex valued apparent power is then split into it's real component $P_{km} = \Re(S_{km})$ the active power flow and imaginary component $Q_{km} = \Im(S_{km})$ which represents the reactive power flow.

Kichhoff's Current Law

We have now described a grid with transmission lines defined by the π-model, generators and buses. For the buses in the power system, Kirchhoff's Current Law (KCL) is fundamental in describing the physical constraints of the problem. The KCL states that for each bus $k \in \mathcal{B}$ the sum of the sum of the apparent power flowing through it must be zero.

$$P_{D,k} + \sum_g P_g - \sum_{m}^{\substack{\forall m \in \mathcal{B} \\ \text{connected to k}}} P_{km} = 0 \tag{2.5}$$

$$Q_{D,k} + \sum_g Q_g - \sum_{m}^{\substack{\forall m \in \mathcal{B} \\ \text{connected to k}}} Q_{km} = 0 \tag{2.6}$$

In the equations above, $P_{D,k}.Q_{D,k}$ describe the active and reactive power loads demanded at bus k. The second term describes the sum of the power from the generators at bus k and the third term describes the power flow to each bus connected to k.

System Constraints

There are a number of power systems engineering constraints that we now must integrate into the formulation of the OPF problem. Voltage magnitude constraints are enforced on the buses. Active and reactive power limits are included for each generator and maximum power flow constraints, governed by thermal limits, are put on transmission lines.

$$V_{b,min} \leq V_b \leq V_{b,max} \qquad \forall b \in \mathcal{B} \tag{2.7}$$
$$P_{g,min} \leq P_g \leq P_{g,max} \qquad \forall g \in \mathcal{G} \tag{2.8}$$
$$Q_{g,min} \leq Q_g \leq Q_{g,max} \qquad \forall g \in \mathcal{G} \tag{2.9}$$
$$S_l \leq S_{l,max} \qquad \forall l \in \mathcal{L} \tag{2.10}$$

Cost Function

The cost function for which we try to optimize defined by

$$c\left(P_g, Q_g\right) = C_2 P_g^2 + C_1 P_g + C_0 + C_2 Q_g^2 + C_1 Q_g + C_0 \quad \forall g \in \mathcal{G} \tag{2.11}$$

2.2 Multiperiod optimal power flow & unit commitment

Multiperiod optimal power flow (MPOPF) with unit commitment (UC) is an extension of the the traditional full OPF problem described in the previous section. The MPOPF strives to model the temporal dynamics of a power grid and integrate discrete decision making into the optimization framework. While the standard OPF focuses on optimizing power system operation for a single time point, multi-period OPF considers a sequence of time intervals, typically spanning hours or days. Expanding the scope of the model allows for the inclusion of time-dependent constraints and objectives, such as ramping limits of generators, energy storage dynamics, and time-varying load profiles. The integration of unit commitment decisions further enhances the model by determining the on/off status of generating units over the planning horizon, accounting for startup and shutdown costs, minimum up and down times, and other operational constraints.

The majority of the system used to describe the OPF remains the same for the multiperiod unit commitment problem. The grid topology, number of generators, buses and lines remains constant over time. Also the engineering constraints described in Eq (2.7-2.10) are seen are time invariant. The major difference is that the state variables used to describe the system now have a time component. Whilst previously P, Q, V, θ were vector in of length $\mathbb{R}^{|\mathcal{G}|}$ and $\mathbb{R}^{|\mathcal{B}|}$ respectively, they are now matrices representing elements of $\mathbb{R}^{|\mathcal{G}|\times T}$ and $\mathbb{R}^{|\mathcal{B}|\times T}$, where T denotes the number of time steps considered our problem. In addition to P, Q, V and θ include a new binary state variable $U \in \{0,1\}^{|G|\times T}$ which describes the on or off state of each generator.

Ramping Constraints

To couple the individual time steps our problem we include a ramping constraint which ensures that the increase in power injected by a specific generator from one time step to another does not exceed a specified limit.

$$\left|P_g^t - P_g^{t-1}\right| - P_{g,rmax} \quad \forall g \in \mathcal{G} \quad \forall t \in \{1, .., T-1\} \tag{2.12}$$

2.2.1 Formulation of the MPOPF as an Optimization Problem

The goal now is to establish a complete formulation of the problem, encompassing all the governing equations and constraints outlined above. This comprehensive formulation will serve as the foundation for accurately solving the unit commitment problem with ACOPF constraints, ensuring that all relevant physical laws and system limitations are fully integrated into the optimization process.

For a grid of $|\mathcal{B}|$ buses and $|\mathcal{G}|$ generators, we are given a load profile $P_D, Q_D \in \mathbb{R}^{|\mathcal{B}|\times T}$ over a time horizon of T timesteps. Our goal is to find the optimal dispatch of generators $P_G, Q_G \in \mathbb{R}^{|\mathcal{G}|\times T}, U \in 0, 1^{|\mathcal{G}|\times T}$, and corresponding bus voltage $V, \theta \in \mathbb{R}^{|\mathcal{B}|\times T}$, which minimize the cost function without violating a set of equality and inequality constraints. Our optimization problem is described as

$$\begin{aligned}\min_{P_G,Q_G,U} \quad & c\left(P_G, Q_G, U\right) \\ s.t \quad & f_i(P_G, Q_G, U, V, \theta, P_D, Q_D) = 0 \quad \forall f_i \in F \\ & h_j(P_G, Q_G, U, V, \theta, P_D, Q_D) \leq 0 \quad \forall h_j \in H\end{aligned} \tag{2.13}$$

where F is the set of equality constraints derived from the KCL (eq 2.5) and H the set of inequality constraints derived from the engineering limits of the system. We use a number of intermediate variables to define F and H.

$$P_{G}{}_b^t = \sum_{g}^{\mathcal{G}_b} P_g^t U_g^t, \tag{2.14}$$

$$Q_{G}{}_b^t = \sum_{g}^{\mathcal{G}_b} Q_g^t U_g^t \tag{2.15}$$

$P_{G}{}_b^t, Q_{G}{}_b^t$ denote the sum of power injected by generators into bus b at time t, where $\mathcal{G}_b$ is the generators connected to b.

$$P_{f}{}_b^t = \sum_m P_{b,m} \tag{2.16}$$

The term $P_{f}{}_b^t$ describes the power flowing to/from b to all connected buses m. $P_{b,m}$ is defined in eq 2.3.

The equality constraints are defined as

$$F = \bigcup_{\forall b \in \mathcal{B}, \forall t \in T} \left\{P_{G}{}_b^t - P_{D}{}_b^t - P_{f}{}_b^t\right\} \cup \left\{Q_{G}{}_b^t - Q_{D}{}_b^t - Q_{f}{}_b^t\right\}$$

The inequality constraints on the generator limits are defined as

$$H_P = \bigcup_{\forall g \in \mathcal{G}, \forall t \in T} \left\{U_g^t(P_{g,min} - P_g^t),\ U_g^t(P_g^t - P_{g,max})\right\} \tag{2.17}$$

$$H_Q = \bigcup_{\forall g \in \mathcal{G}, \forall t \in T} \left\{U_g^t(Q_{g,min} - Q_g^t),\ U_g^t(Q_g^t - Q_{g,max})\right\} \tag{2.18}$$

$$H_{ramp} = \bigcup_{\forall g \in \mathcal{G}, \forall t \in T} \left\{U_g^t\left(\left|P_g^t - P_g^{t-1}\right| - P_{g,rmax}\right)\right\}. \tag{2.19}$$

$$\tag{2.20}$$

The inequality constraints on the voltage magnitude limits are defined as

$$H_V = \bigcup_{\forall b \in \mathcal{B}, \forall t \in T} \left\{V_{b,min} - V_b^t, V_b^t - V_{b,max}\right\} \tag{2.21}$$

The inequality constraints on the line flow limits are defined as

$$H_S = \bigcup_{\forall l \in \mathcal{L}, \forall t \in T} \left\{ \left(P{f}_l^t\right)^2 + \left(Q{f}_l^t\right)^2 - S_{l,max} \right\} \tag{2.22}$$

Thus H is the union of the inequality constraints defined above.

$$H = H_P \cup H_Q \cup H_{ramp} \cup H_V \cup H_S \tag{2.23}$$

The objective function c is composed of the polynomial generator costs and ramping costs for each generator in each timestep.

$$\begin{aligned} c\left(P_G, Q_G, U\right) = & \sum_{g}^{\mathcal{G}} \sum_{t=1}^{T} U_g^t \left(C_{g,p2} {P_g^t}^2 + C_{g,p1} P_g^t C_{g,p0} + C_{g,ramp} \left| P_g^t - P_g^{t-1} \right| \right) + \\ & \sum_{g}^{\mathcal{G}} \sum_{t=1}^{T} U_g^t \left(C_{g,q2} {Q_g^t}^2 + C_{g,q1} Q_g^t C_{g,q0} + C_{g,ramp} \left| Q_g^t - Q_g^{t-1} \right| \right) \end{aligned} \tag{2.24}$$

$C_{g,p0} - C_{g,p2}$ describe the generation cost coefficients at bus g and $C_{g,ramp}$ the ramping cost.

2.3 Traditional Methods for solving the UC-ACOPF Problem

The unit commitment (UC) problem with ACOPF constraints is a non-convex, quadratically constrained, mixed-integer optimization problem that poses significant challenges for traditional solvers. Although tools like CPLEX [6] and SCIP [7] are adept at handling non-convexity, non-linearity, and mixed-integer constraints individually, finding efficient methods to solve the combined UC-ACOPF problem remains an ongoing research area. The UC-ACOPF problem can be formulated as a non-convex quadratically constrained optimization problem, but while CPLEX can solve non-convex quadratic programming (QP) problems, it often struggles when quadratic constraints are combined with mixed-integer variables.

The complexity of the problem also varies depending on the formulation of the ACOPF. In the angular formulation, the constraints involve sine and cosine non-linearities, whereas in the rectangular formulation, these constraints are expressed as bilinear terms, each adding different layers of difficulty. To make the problem more tractable for commercial solvers, approximations like second-order conic programming (SOCP) [8] or semidefinite programming (SDP) [9] are frequently used. These methods can offer better results than the DC approximation, but they are not guaranteed to find AC-feasible solutions [10]. Additionally, while some methods introduce tighter constraints to improve solution feasibility, they may inadvertently exclude parts of the AC solution space.

Another approach to tackling the UC-ACOPF problem is through outer approximation and decomposition methods. These methods iteratively solve a relaxed mixed-integer problem, then fix the integer variables and solve the full non-linear programming (NLP) problem. However, due to the non-convex nature of the problem, these methods do not guarantee optimality. Finally, SOCP can be employed to solve a formulation of the rectangular ACOPF with linearized power flow constraints, which simplifies the non-linearities but still struggles with ensuring both AC feasibility and optimality.

To compare our results, we also test the ExaADMM solver [11], which employs the alternating direction method of multipliers (ADMM) to solve the problem. This method leverages a component-based decomposition, allowing the solver to efficiently tackle the resulting subproblems. ExaADMM is built with Julia and is one of the few openly available solvers capable of addressing the full ACOPF problem with unit commitment, making it a valuable tool for benchmarking our model's performance.

2.4 Augmented Lagrangian Formulation

To transform the optimization problem from a non-convex constrained optimization problem into an unconstrained locally convex problem we use the augmented langrangian method. Here we add the equality and inequality constraints as additional terms to our objective function. We summarize the various parameters used to describe our system into the state variable $x = \{P_G, Q_G, U, V, \theta, P_D, Q_D\}$.

$$\min_{x} \quad f\left(x\right) = c\left(x\right) + \mu_H^k \sum_{h \in H} \delta h\left(x\right) + \mu_G^k \sum_{g \in G} g\left(x\right)^2 + \sum_{h \in H} \lambda_h^k h\left(x\right) + \sum_{g \in G} \lambda_h^k g\left(x\right) \tag{2.25}$$

where k is the iteration, $\delta(h) = \max(0, h)^2$, $0\mu_H^k$ and μ_G^k the inequality and equality penalty multipliers for iteration k, λ_h^k and λ_h^k the augmented lagrangian multipliers. The multipliers are updated with the following scheme:

$$\begin{aligned} \mu_H^{k+1} &= \beta_H \mu_H^k, \quad \mu_H^0 = \alpha_H \\ \mu_G^{k+1} &= \beta_G \mu_G^k, \quad \mu_G^0 = \alpha_G \\ \lambda_h^{k+1} &= \max\left(0, \lambda_h^k + 2\mu_H^k h\left(x\right)\right) \\ \lambda_g^{k+1} &= \lambda_g^k + 2\mu_G^k g\left(x\right) \end{aligned} \tag{2.26}$$

α and β are hyperparameters.

2.5 PINNs

Physics-Informed Neural Networks (PINNs) [12] represent a cutting-edge approach in the intersection of machine learning and scientific computing, providing a powerful tool for solving complex physical problems. Unlike traditional neural networks, which are typically trained on large datasets with labeled examples, PINNs are designed to incorporate the underlying physical laws directly into the learning process. This integration is achieved by embedding these laws, often expressed as partial differential equations (PDEs) or ordinary differential equations (ODEs), into the neural network's loss function.

PINNs operate by enforcing that the network's outputs satisfy the physical laws governing the system. This is done by introducing terms in the loss function that penalize any deviation from these laws at specific points in the domain, known as colocation points. The key idea is that instead of training the neural network solely on data points (input-output pairs), the network is also trained to minimize the error in satisfying the governing equations at these colocation points.

PINNs can overcome this limitation by leveraging the physical laws as a form of prior knowledge. This allows the model to perform well even with limited or no labeled data, making it particularly useful in fields where experimental data is scarce or difficult to obtain.

Many scientific problems involve complex systems governed by PDEs or ODEs, such as fluid dynamics, heat transfer, or structural mechanics. These equations can be difficult to solve using classical numerical methods, especially in high-dimensional spaces or when the equations are nonlinear. PINNs offer a flexible and efficient alternative by approximating solutions directly through the neural network, which can handle high-dimensional inputs and outputs and adapt to complex, nonlinear relationships.

Physics-Informed Neural Networks (PINNs) are widely recognized for their effectiveness in solving partial differential equations (PDEs) on continuous domains. This approach leverages the ability of PINNs to incorporate the underlying physical laws governing such systems directly into the neural network's loss function. By enforcing these laws at specific colocation points within a continuous domain, PINNs can accurately model complex physical phenomena, such as fluid dynamics, heat transfer, and wave propagation.

However, while PINNs are traditionally employed for problems involving continuous domains, our work takes a different approach. We apply PINNs to solve an optimization problem on a discrete graph, rather than a continuous space. In this context, the graph's nodes and edges represent the discrete structure of our problem, and the neural network is trained to find optimal solutions that satisfy both the inequality and equality constraints as well as the cost function associated with the optimization problem.

To achieve this, we formulate the loss function using the augmented Lagrangian method, which allows us to effectively evaluate the constraints and objectives at every node and edge of the graph. This approach ensures that the PINN is fully integrates the problem's discrete nature, even though the underlying technique is inspired by methods typically applied to continuous domains.

Our method is entirely unsupervised, meaning that the training process does not rely on any labeled data. Instead, it depends solely on the physical and mathematical principles embedded in the loss function. This adaptation of PINNs from continuous to discrete domains highlights their versatility and showcases their potential to tackle a broad range of scientific and engineering problems beyond their conventional use cases.

2.6 Graph Neural Networks

Optimal power flow and related problems are naturally described by graphs, where nodes represent buses or generators and edges correspond to transmission lines or transformers. Representing these systems using neural networks is challenging due to the importance of both local and global graph features, as well as the diverse forms these graphs can take. Graph neural networks (GNNs) offer a powerful solution, generalizing the idea behind convolutional neural networks (CNNs). For CNNs, an image or pixel array can be seen as a specific type of graph where each pixel is a node connected to its neighboring pixels. GNNs generalize this concept by learning message passing parameters on arbitrary connected nodes instead of relying on fixed kernel stencils, making no assumptions about the number of neighbors each node has. As with CNNs, which only need to learn the kernel parameters and work with arbitrary image sizes, GNNs can by applied to arbitrary grid sizes and topologies, once a set of message passing parameters has been learnt. The concept of GNNs was first introduced by Scarselli et al. [13] and later extended to graph convolutional networks by Kipf and Welling [14]. Our neural network is based on a message-passing layer named TransformerConv defined in [15], which proposes a unified message-passing model, adopting a graph transform network [16] and incorporating novel methods to reduce overfitting from self-loop label information.

GNNs have proven effective in representing the graph structure of power grids and have also brought significant advancements in various fields, such as protein folding, recommender systems, and combinatorial optimization. Notably, GNNs have shown promise in enhancing branch-and-bound algorithms [17], a method for solving high-dimensional mixed-integer problems, although the impact on our method's performance has not been directly investigated.

Chapter 3

Implementation

3.1 Neural Network Design

In our approach, we utilize a compact network architecture that is centered around the TransformerConv neural network layer, as detailed in the previous section. The network architecture consists of two layers of TransformerConv message passing, which effectively capture and propagate information across the graph structure. These message-passing layers are then followed by two fully connected linear layers, which further process the aggregated information to generate the final predictions. An overview of the specific structure used in our results can be seen in figure 4.2.

The input features provided to the network include active power demand (P_D), reactive power demand (Q_D), node index, one-hot encoded bus types (such as generator bus or reference bus), and the edges representing transmission lines between buses. The output features that the network predicts are active power generation (P_G), reactive power generation (Q_G), voltage magnitude (V), voltage angle (θ), and a binary state variable (U). These input and output features are defined at each node within the graph and are time-dependent, meaning that each node is characterized by a feature vector with dimensions proportional to the number of timesteps. For example, if $n_{timesteps}$ is the number of timesteps, then each node will have a feature vector of size $7 \times n_{timesteps}$.

Power grids are naturally modeled as graphs where nodes correspond to buses, and edges represent transmission lines. Generators are typically associated with specific buses and are usually considered as features of those buses. However, incorporating multiple generators directly into the GNN framework can be challenging due to the complexity of representing multiple generators as features of a single node. To address this challenge, we introduce extra virtual nodes into the graph. These virtual nodes represent the generators and are connected to the bus nodes they are associated with. This approach allows the network to explicitly model the relationships between generators and buses while preserving the graph structure.

The network also processes a list of to-from pairs that describe the edges, which define the connectivity between buses. All additional information about the transmission lines, such as impedance and capacity, is not directly included in the input data but is instead incorporated into the loss function. This design choice ensures that the network focuses on learning the most relevant features for the task while allowing for the inclusion of more complex domain-specific knowledge during the optimization process.

This architecture, with its efficient message-passing mechanism and careful handling of generator features, enables the network to effectively learn and generalize over different grid configurations. It ensures that the network remains scalable and adaptable to various grid sizes and complexities while maintaining the ability to accurately predict the necessary output variables.

3.2 Training and evaluation

The grid data is formatted using the MATPOWER MPC case file, which is an industry-standard format commonly used for both testing and real-world examples [18]. This format provides easy access to a variety of grid configurations, facilitating the preparation and preprocessing of input data. Initially, we utilized MATPOWER's MOST for comparison and validation purposes [19]. However, due to issues with convergence–particularly the fact that MOST supports only DC power flow–we transitioned to using ExaADMM.jl [11].

For model input, the grid is represented as a graph, including both the original grid structure and virtual nodes corresponding to the generators. Consequently, the output dimension of the predicted values, y_{pred}, is $(n_{\text{buses}} + n_{\text{gens}}) \times n_{\text{timesteps}}$. This data is then transformed into appropriate data loaders for efficient training and inference.

The training process is implemented using the PyTorch and PyTorch-Geometric (PyG) packages [20]. PyTorch is a standard library for machine learning in Python, providing tensor data structures along with numerous auxiliary functions and classes. PyG, built on top of PyTorch, is specifically designed for graph neural networks and irregular data structures. In our model, we utilize TransConv layers [15] from PyG.

Given the absence of labeled data, a custom loss function is employed, relying solely on physics-based losses. The basic training step follows these steps:

1. Run the network.
2. Denormalize the predicted values.
3. Calculate power flows based on the denormalized output.
4. Construct the custom loss function.
5. Perform backpropagation to compute gradients.
6. Execute an optimizer step to update the network parameters.
7. Update the Lagrange multipliers.
8. Repeat the process until certain convergence conditions are met.

Throughout training, values are logged to Neptune.ai [21] for monitoring and analysis. Model snapshots are periodically evaluated locally to ensure performance and accuracy.

After training, the PyTorch model and grid characteristics are saved for future use. To compare our model's performance with a traditional solver, we generate data compatible with ExaADMM.jl. This involves running a script in Julia to obtain the reference solution, which is then imported back into our environment. Using this reference solution, we calculate the necessary intermediate variables.

Subsequently, we analyze the results by computing the differences between our model's predictions and the reference solution, focusing on constraint violations and costs.

Chapter 4

Results

In the following chapter, we will demonstrate the effectiveness of our model on three distinct grid systems, each varying in complexity and size. The first system, referred to as ”Case 9” is a highly simplified model of the Western Systems Coordinating Council (WSCC) in the United States. It consists of 9 buses, where 6 inner buses are connected in a ring configuration, and 3 outer buses, each hosting a generator, are connected to every bus in the ring.

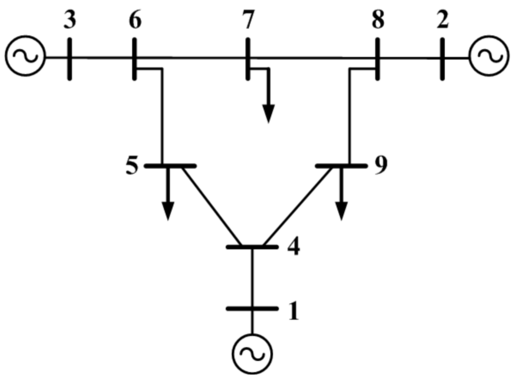


Figure 4.1: IEEE 9-Bus System [1]

The second system, ”Case 30” is an approximation of the American Electric Power system from the 1960s. This grid contains 30 buses and 6 generators, providing a more complex scenario than case 9. Finally, ”Case 39” is a more extensive grid, known as the 10-machine New England power system [22]. It includes 39 buses and 10 generators interconnected by 46 transmission lines, offering a robust testbed for evaluating the performance of our model.

Network Configuration & Hyperparameters

We use a single configuration of our network for all grids and test cases. The only parameters which changes between the different runs are T, the number of timesteps in our prediction horizon, and $N_{nodes} = |B| + |G|$, which is the total nodes in the graph. The structure of the network is visuallized in figure 4.2. The network has 24 neurons in its hidden layers and 2 heads in the second message passing layer and first dense layer.

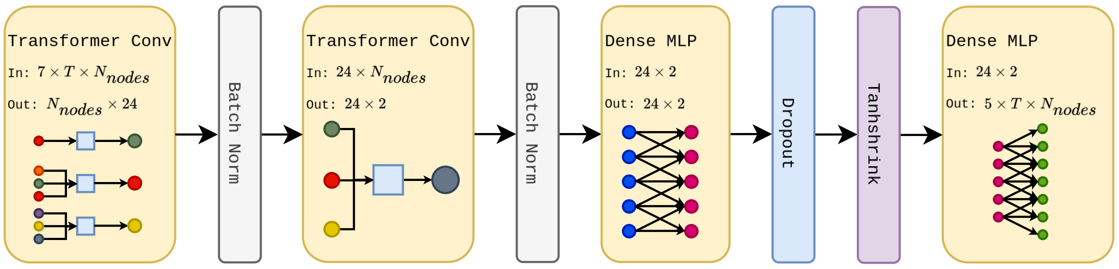


Figure 4.2: PIGNN-model

We use an ADAM optimizer [23] with a starting learning rate of 0.01 and the learning rate is decreased with an exponential scheduler with a decay factor of $\gamma = 0.996$ every 10 epochs.

Augmented Lagrangian penalty are calculated with the following parameters:

Table 4.1: Augmented Lagrangian Parameters

μ_{cost}	100
μ_H	0.01
μ_G	0.1
β_{cost}	1.0001
β_H	1.0008
β_G	1.0006

Table 4.2: Loss Function Scaling

Active Power Balance	1
Reactive Power Balance	1
Voltage Magnitude	30
Acitve Power Limits	1
Reacitve Power Limits	1
Transmission Limits	0.01
Plate Loss	1

4.1 9-Bus System: Periodic Demand Profile

In the following section, we provide a detailed analysis of the solution for case 9, utilizing a periodic demand profile modeled as a sine wave. This specific demand profile is designed to stress-test the unit commitment process by imposing varying load conditions that compel the network to turn off generators at certain times. The subsequent analysis is based on the load profile provided as input, as illustrated in 4.3.

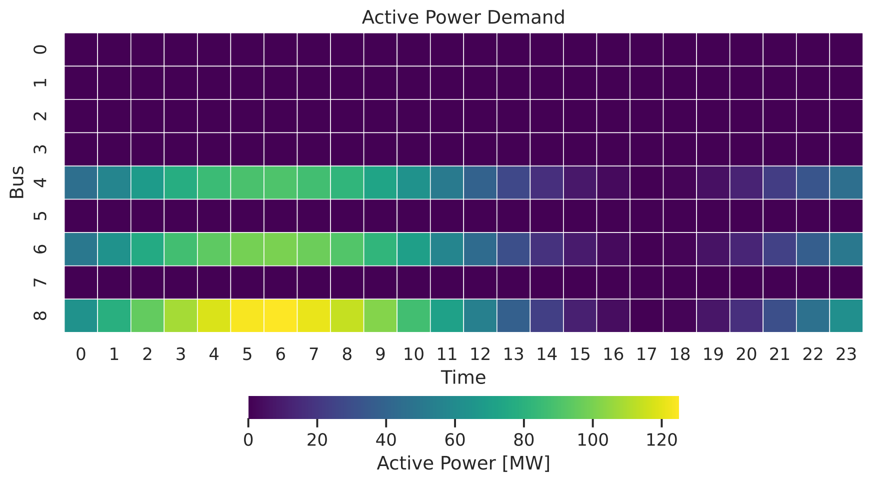
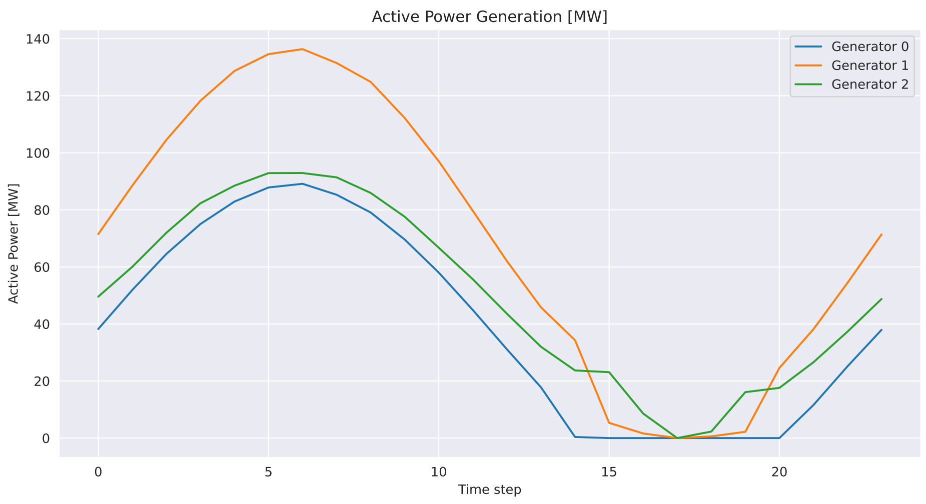


Figure 4.3: Active Power Demand, Case 9

Our model produces the following operating schedule and unit commitment decisions for the three generators in the system. Figure 4.4 shows the active power injected in [MW] from each generator. The lower plot shows the commitment keys. As the network computes a probabilistic value between 0 and 1 for each key, not all values shown are binary values. This allows us to understand how certain the network is of the optimality of an on/off state. However, in the loss computation and outputting of the solution, these variables are rounded up or down to guarantee their binary nature.



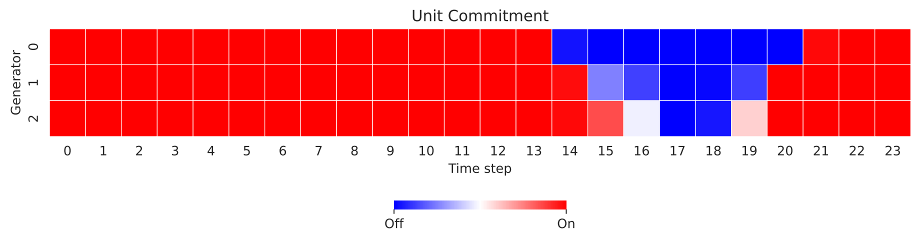


Figure 4.4: Active Power Generation & Unit Commitment, Case 9

In addition to determining the generation schedule, the model is also responsible for calculating a stable set of voltage magnitudes and angles. These values are crucial for ensuring that the power flows through the network correctly, without violating any of the engineering constraints, such as maintaining line flows within safe operational limits. The upper plot in figure 4.5 shows the computed voltages magnitude at each node. The voltage magnitudes are shown between 1 ± 0.1p.u. which corresponds to the defined engineering limits of the buses. The lower plot of figure 4.5 shows the voltage angle between $\pm 0.05\pi$. Bus 0 serves as the reference bus in the system, and all other voltage angles are measured relative to this reference point. In practice, this means that the voltage angle at bus 0 is set to zero, simplifying the system by fixing a common phase reference.

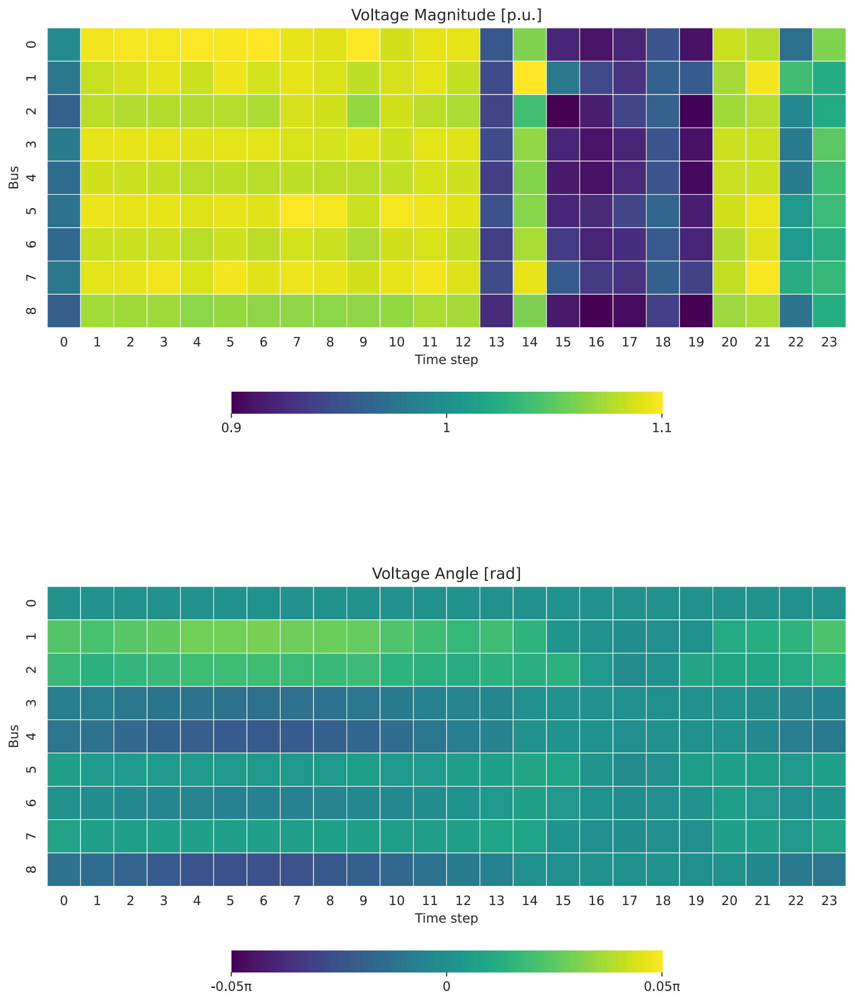


Figure 4.5: Voltage Magnitude & Angle, Case 9

4.2 Comparison with the ExaAdmm Solver

The following section will evaluate the performance of the three grid cases on two demand profiles and compare the results to reference solutions computed using the ExaADMM solver [11]. We begin with the periodic demand profile discussed in the previous section, followed by an analysis using a step function demand profile where the load experiences a rapid drop after 12 timesteps. The tables presented below compare the performance of our model and the ExaADMM solver across six different metrics. These metrics are evaluated per entity, whether it be a generator, bus, or transmission line, and are assessed at each timestep. The cost variable denotes the average per generator and timestep and includes ramping costs. All other variables denote the average constraint violation.

Table 4.3: Comparison NN/ADMM, Case 9

Load Profile	Step		Period	
Method	NN	ADMM	NN	ADMM
Cost [$]	1152.63	1118.58	908.35	903.44
Voltage [p.u.]	0.0000	0.0000	0.00001	0.0000
Active power [MW]	0.0009	0.0339	0.0022	0.5716
Reactive power [MVar]	0.0003	0.0000	0.0000	0.0000
Transmission [MVar]	0.0000	0.0000	0.0000	0.0000
Power balance [MW]	0.9191	33.4046	0.9581	26.2458

Table 4.4: Comparison NN/ADMM, Case 30

Load Profile	Step		Period	
Method	NN	ADMM	NN	ADMM
Cost [$]	58.70	54.58	43.23	43.41
Voltage [p.u.]	0.0001	0	0.00004	0
Active power [MW]	0.0024	0.0000	0.0058	0.0000
Reactive power [MVar]	0.0020	0.0000	0.0082	0.0000
Transmission [MVar]	0.1594	0.0000	0.1569	0.0000
Power balance [MW]	0.0118	6.1867	0.0821	4.9514

Table 4.5: Comparison NN/ADMM, Case 39

Load Profile	Step		Period	
Method	NN	ADMM	NN	ADMM
Cost [$]	2415.07	3164.25	1642.31	2904.63
Voltage [p.u.]	0.02127	0	0.01771	0
Active power [MW]	0.6739	28.9327	0.2243	26.4081
Reactive power [MVar]	0.5775	12.8500	0.2268	12.8500
Transmission [MVar]	1.3478	4541.7271	0.0108	831.6963
Power balance [MW]	2.6830	145.3378	1.3666	121.4072

Chapter 5

Discussion

5.1 Model performance

Section 4.1 demonstrates that our model is capable of computing accurate results for the full AC Optimal Power Flow (ACOPF) problem with unit commitment. The model effectively computes a generation dispatch that meets the expected demand profile while maintaining voltage angles and magnitudes within acceptable bounds across all time steps and buses. However, while unit commitment predictions generally perform well, challenges arise in low demand transition zones, where the model may fail to identify the optimal commitment schedule. This issue could be partly due to the absence of penalties or costs associated with the rapid switching of generators.

In Section 4.2, we compare our model's performance with that of a traditional mixed-integer solver. Although the ExaADMM solver finds marginally cheaper solutions for both case 9 and case 30, it consistently struggles to produce solutions that adequately satisfy the power balance equalities. This difficulty is even more pronounced in case 39, where the solver has significant challenges in finding valid solutions on the larger grid. Figure 5.1 illustrates the ratio of the neural network's cost, inequality, and equality losses compared to those of the ExaADMM solver. In this figure, cost values above 1 indicate better performance by our model.

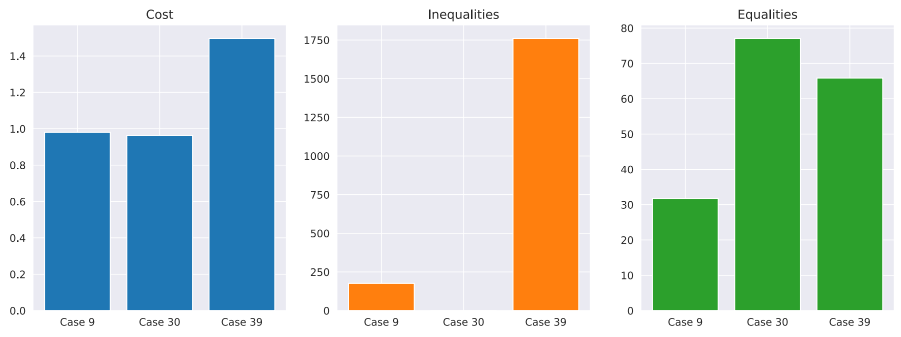


Figure 5.1: Ratio of NN cost to ADMM cost

5.2 Training and convergence

Our model exhibits a high sensitivity to hyper parameters, particularly the μ parameters, which scale the penalty and cost losses. These parameters are crucial for ensuring that no single loss component dominates the others excessively. Similarly, the β parameters play a significant role in determining how the loss components scale over the epochs of the training process. Given that we formulate the problem as a multi-objective optimization, it is essential to ensure that all aspects of the objective function are adequately represented during optimization.

To achieve this, we carefully select the μ and β parameters so that the training process emphasizes different aspects of the problem during three main stages. Initially, the objective function is dominated by the cost component and inequality constraints, which rapidly drive the model into a physically feasible solution space and set the expected generator output close to zero. In the next phase, the plate loss becomes more significant, reflecting the model's prioritization of meeting the demand over minimizing generation costs. During this phase, the model further refines its solution space, ensuring that inequality violations are minimized and that the total power generated closely matches the total demand.

Finally, the loss function becomes dominated by the equality constraints derived from Kirchhoff's current law. Since the solution space defined by these equalities has a volume of zero, these loss components cannot be reduced to zero completely. As a result, this portion of the loss function is weighted more heavily when the algorithm's learning rate is low, ensuring that the model focuses on accurately fulfilling the power balance equalities as training progresses. In this phase, the total generated power also often increases high than the total demand as to account for and minimize the transmission losses in the system. Therefore, a finely tuned set of hyper-parameters is essential for accurately training models that fully incorporate all aspects of the ACOPF problem. Given that the augmented Lagrange multipliers are adjusted based on the constraint violations from previous iterations, the success of the training process can be highly dependent on the initial conditions and the specific input data used. This sensitivity underscores the importance of careful hyper-parameter selection and initialization to achieve optimal performance and reliable convergence.

As is common with many other physics-informed machine learning methods, our model exhibits slow convergence [24]. A higher learning rate can quickly lead to instability, necessitating careful adjustment. Although the loss computation itself is relatively inexpensive, thanks to the vectorization of intermediate variable and loss component calculations, the overall process still requires a significant number of iterations to converge to a satisfactory solution. Both the forward pass and back-propagation are also relatively efficient due to the small size of the network.

Currently, our network has been designed to be trained on a single demand profile. This must be expanded in future work to allow for further vectorization and parallelization opportunities. Given that the entire training step typically takes less than 50 milliseconds, training is performed on a CPU.

Once the network is trained, inference is very inexpensive. The graph neural network (GNN) stage is independent of the grid size, resulting in a computational complexity of $\mathcal{O}(m+n)$, where m represents the number of edges and n the number of nodes. The dense readout layers of our network are relatively small, with only the final layer dependent on grid size and the number of time steps. Although this final layer has a complexity of $\mathcal{O}(m+n)$, it remains more efficient than traditional optimization methods, which model the full ACOPF problem and have a computational complexity of at least $\mathcal{O}(n^3)$ [25]. This significant reduction in computational cost highlights the efficiency of our approach compared to conventional methods.

Chapter 6

Conclusions and Outlook

Our research introduces an innovative machine learning algorithm that leverages Graph Neural Networks (GNNs) with a physics-based loss function to solve the Unit Commitment problem. By employing the Augmented Lagrangian method, we successfully transform the constrained optimization problem into an unconstrained one, enabling the effective incorporation of various complex constraints. Our model demonstrates promising results for small grids and shows potential for scaling to larger systems, albeit with some challenges. While neural networks lack guarantees for scaling to practical, real-world grid sizes, a major advantage of our approach lies in the computational efficiency of the forward pass, which exhibits low dependence on grid size. To fully utilize this model's potential, future work should focus on training with multiple demand profiles simultaneously, potentially by adjusting the architecture to avoid directly learning the demand profile. Incorporating recurrent neural network (RNN) elements could facilitate information flow through individual layers, enhancing the model's capabilities. These extensions could make our model particularly powerful for tackling complex problems like N-2 security constrained AC Optimal Power Flow, which would greatly benefit from extremely fast inference speeds. Furthermore, integrating a more refined method of loss scaling and multiplier updating could improve performance. Currently, the model relies on finely tuned hyper-parameters found through trial and error. Developing a method to compute and balance loss scaling during training could reduce sensitivity to hyper-parameters, making the system more robust and adaptable to various grid scenarios.

Bibliography

[1] Y. Song, D. Hill, and T. Liu, “Small-disturbance angle stability analysis of microgrids: A graph theory viewpoint,” 09 2015.

[2] Carpentier, “Contribution to the economic dispatch problem,” *Bulletin de la Societe Francoise des Electriciens, vol. 3, no. 8, pp. 431â447*, 1962.

[3] F. Capitanescu, “Critical review of recent advances and further developments needed in ac optimal power flow,” *Electric Power Systems Research*, vol. 136, pp. 57–68, 2016.

[4] Z. Yang, H. Zhong, Q. Xia, and C. Kang, “Fundamental review of the opf problem: Challenges, solutions, and state-of-the-art algorithms,” *Journal of Energy Engineering*, vol. 144, no. 1, p. 04017075, 2018.

[5] R. D. Zimmerman and C. E. Murillo-SÃ¡nchez, “Matpower user’s manual,” May 2024.

[6] I. I. Cplex, “V12. 1: Userâs manual for cplex,” *International Business Machines Corporation*, vol. 46, no. 53, p. 157, 2009.

[7] S. Bolusani, M. Besançon, K. Bestuzheva, A. Chmiela, J. Dionísio, T. Donkiewicz, J. van Doornmalen, L. Eifler, M. Ghannam, A. Gleixner, C. Graczyk, K. Halbig, I. Hedtke, A. Hoen, C. Hojny, R. van der Hulst, D. Kamp, T. Koch, K. Kofler, J. Lentz, J. Manns, G. Mexi, E. Mühmer, M. E. Pfetsch, F. Schlösser, F. Serrano, Y. Shinano, M. Turner, S. Vigerske, D. Weninger, and L. Xu, “The SCIP Optimization Suite 9.0,” ZIB-Report 24-02-29, Zuse Institute Berlin, February 2024.

[8] Z. Yuan and M. R. Hesamzadeh, “Second-order cone ac optimal power flow: convex relaxations and feasible solutions,” *Journal of Modern Power Systems and Clean Energy*, vol. 7, no. 2, pp. 268–280, 2019.

[9] X. Bai, H. Wei, K. Fujisawa, and Y. Wang, “Semidefinite programming for optimal power flow problems,” *International Journal of Electrical Power Energy Systems*, vol. 30, no. 6, pp. 383–392, 2008.

[10] D. A. Tejada-Arango, S. Wogrin, P. SÃ¡nchez-Martin, and A. Ramos, “Unit commitment with acopf constraints: Practical experience with solution techniques,” in *2019 IEEE Milan PowerTech*, pp. 1–6, 2019.

[11] W. Zhang, Y. Kim, and K. Kim, "On solving unit commitment with alternating current optimal power flow on gpu," 2023.

[12] M. Raissi, P. Perdikaris, and G. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational Physics*, vol. 378, pp. 686–707, 2019.

[13] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, 2009.

[14] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[15] Y. Shi, H. Zhengjie, S. Feng, H. Zhong, W. Wang, and Y. Sun, "Masked label prediction: Unified message passing model for semi-supervised classification," pp. 1548–1554, 08 2021.

[16] S. Yun, M. Jeong, R. Kim, J. Kang, and H. J. Kim, "Graph transformer networks," in *Advances in Neural Information Processing Systems* (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds.), vol. 32, Curran Associates, Inc., 2019.

[17] H. Liang, S. Wang, H. Li, L. Zhou, X. Zhang, and S. Wang, "Bignn: Bipartite graph neural network with attention mechanism for solving multiple traveling salesman problems in urban logistics," *International Journal of Applied Earth Observation and Geoinformation*, vol. 129, p. 103863, 2024.

[18] R. D. Zimmerman, C. E. Murillo-SÃ¡nchez, and R. J. Thomas, "Matpower: Steady-state operations, planning, and analysis tools for power systems research and education," *IEEE Transactions on Power Systems*, vol. 26, no. 1, pp. 12–19, 2011.

[19] C. E. Murillo-SÃ¡nchez, R. D. Zimmerman, C. L. Anderson, and R. J. Thomas, "Secure planning and operations of systems with stochastic sources, energy storage, and active demand," *IEEE Transactions on Smart Grid*, vol. 4, no. 4, pp. 2220–2229, 2013.

[20] M. Fey and J. E. Lenssen, "Fast graph representation learning with PyTorch Geometric," in *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[21] neptune.ai, "neptune.ai: experiment tracker," 2024.

[22] T. Athay, R. Podmore, and S. Virmani, "A practical method for the direct analysis of transient stability," *IEEE Transactions on Power Apparatus and Systems*, vol. PAS-98, no. 2, pp. 573–584, 1979.

[23] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *International Conference on Learning Representations*, 12 2014.

[24] D. C. V. G. F. e. a. Cuomo, S., “Scientific machine learning through physicsinformed neural networks: Where we are and whatâs next.,” *Journal of Scientific Computing*, 2022.

[25] Y. Zhu, Y. Zhou, W. Wei, and N. Wang, “Cascading failure analysis based on a physics-informed graph neural network,” *IEEE Transactions on Power Systems*, vol. 38, no. 4, pp. 3632–3641, 2023.

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every written paper or thesis authored during the course of studies. In consultation with the supervisor, one of the following three options must be selected:

○ I confirm that I authored the work in question independently and in my own words, i.e. that no one helped me to author it. Suggestions from the supervisor regarding language and content are excepted. I used no generative artificial intelligence technologies[1].

○ I confirm that I authored the work in question independently and in my own words, i.e. that no one helped me to author it. Suggestions from the supervisor regarding language and content are excepted. I used and cited generative artificial intelligence technologies[2].

[✓ I confirm that I authored the work in question independently and in my own words, i.e. that no one helped me to author it. Suggestions from the supervisor regarding language and content are excepted. I used generative artificial intelligence technologies[3]. In consultation with the supervisor, I did not cite them.]

Title of paper or thesis:

Multi-period Optimal Power Flow with Physics-informed Graph Neural Networks

Authored by:
If the work was compiled in a group, the names of all authors are required.

Last name(s):	**First name(s):**
Lausberg	Tom

With my signature I confirm the following:
- I have adhered to the rules set out in the Citation Guide.
- I have documented all methods, data and processes truthfully and fully.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for originality.

Place, date	**Signature(s)**
Zürich, 09.09.2024	[illegible]

If the work was compiled in a group, the names of all authors are required. Through their signatures they vouch jointly for the entire content of the written work.

[1] E.g. ChatGPT, DALL E 2, Google Bard
[2] E.g. ChatGPT, DALL E 2, Google Bard
[3] E.g. ChatGPT, DALL E 2, Google Bard