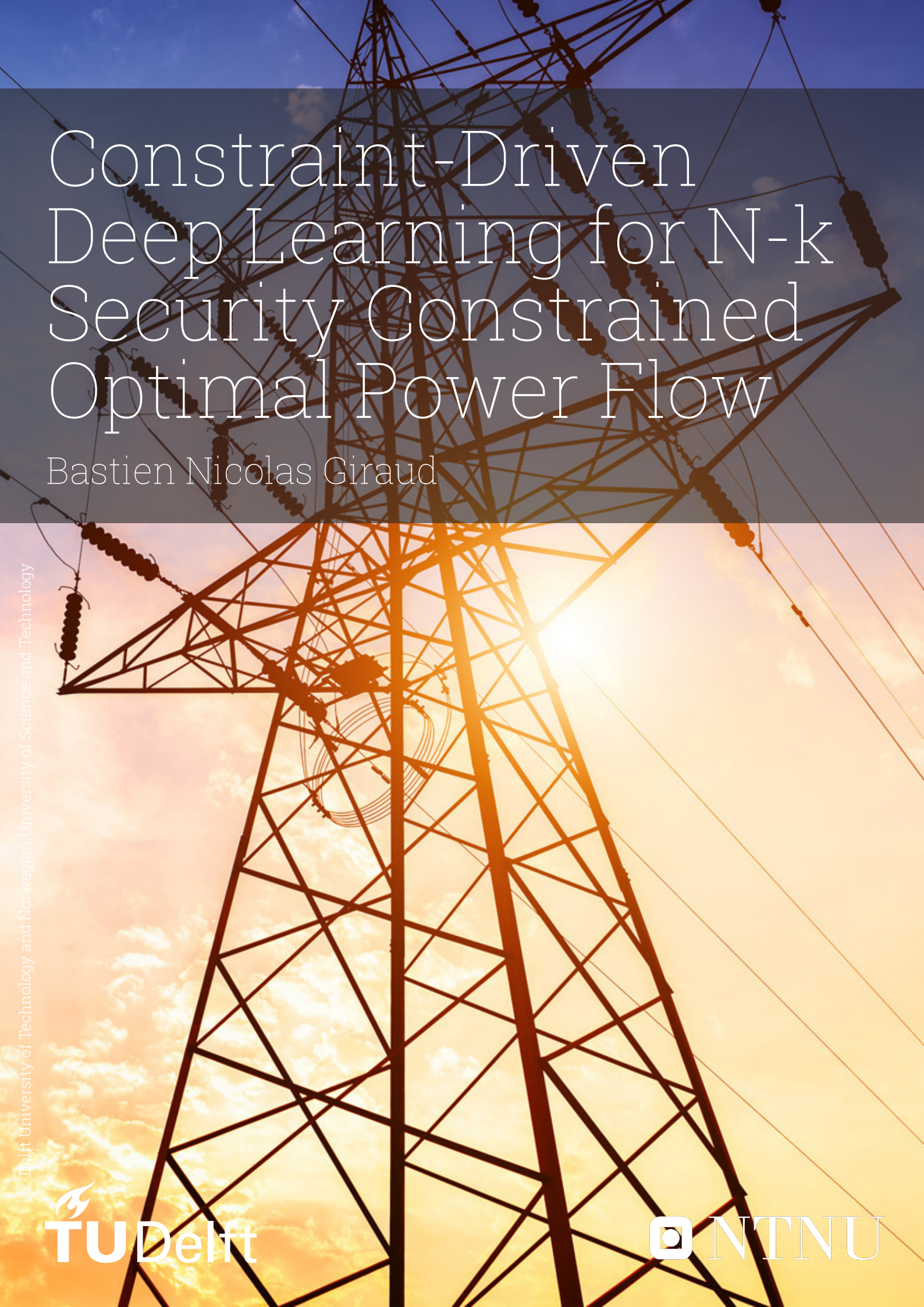


Constraint-Driven Deep Learning for N-k Security Constrained Optimal Power Flow

Bastien Nicolas Giraud

Delft University of Technology and Norwegian University of Science and Technology

TU Delft

NTNU

Constraint-Driven Deep Learning for N-k Security Constrained Optimal Power Flow

by

Bastien Nicolas Giraud

TU Delft	4606566
NTNU	582370

to obtain the degree of MSc. Electrical Engineering
at the Delft University of Technology,
and the degree of MSc. Wind Engineer,
at the Norwegian University of Science and Technology,
to be defended publicly on the 31st of October, 2023 at 09:00 AM.

Project duration:	January, 2023 – October, 2023	
Thesis committee:	Dr. ir. José Rueda Torres,	TU Delft, committee chair
	Dr. ir. Jochen Cremer,	TU Delft, supervisor
	Ir. Ali Rajaei,	TU Delft, daily supervisor
	Prof. dr. ir. Olav Bjarte Fosso,	NTNU, supervisor

Cover: Power System Analysis - EA Technology cover image from eatechnology.com/services/power-system-analysis/

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

TU Delft NTNU

Acknowledgements

This thesis marks the end of my life as a student and concludes my studies in the European Wind Energy Master. The EWEM program enabled me to study in Copenhagen, Delft and Trondheim, enriching both my academic and personal growth tremendously. Therefore, I would like to begin with a few words of gratitude.

First and foremost, I would like to express my gratitude to Jochen Cremer, Ali Rajeai and Olav Bjarte Fosso for their supervision during this thesis. I have thoroughly enjoyed our discussions and am grateful for the help and expertise you've offered along the way. Your enthusiasm for the field has sparked my interest for the intersection of AI and power systems. Secondly, I would like to thank José Rueda Torres for acting as chair and for providing me with valuable feedback in the final stages of my thesis. I would also like to extend my gratitude to Kim Batselier for your time in explaining the basic concepts of tensor train decomposition.

In addition, I would like to thank all the people involved in the EWEM program for enabling me to have this fantastic experience. All the people and close friends I met along the way made the past two years even more memorable. Finally, I'm greatly thankful to my parents for their unconditional support and encouragement throughout my student life.

Bastien Nicolas Giraud
Delft, October 2023

Abstract

The transition to green energy is reshaping the energy landscape, marked by increased integration of renewable energy sources, distributed resources, and the electrification of other energy sectors. These changes challenge grid security, particularly regarding the N-1 security criterion, a crucial factor in preventing blackouts. Furthermore, climate change is contributing to the growing frequency of extreme weather events, which constitute the second major cause of blackouts. As grid complexity keeps on increasing, the need for N-k security, where k lines fail simultaneously, and increased resilience against extreme weather events is becoming increasingly evident. This necessitates studying the security constrained optimal power flow (SCOPF) problem considering multiple line outages (N-k). Current methods exhibit poor scalability as k increases. In response to the challenge of limited scalability, this thesis proposes a constraint-driven machine learning approach to approximate N-k SCOPFs.

The proposed approach relies on the linearized direct current optimal power flow. The approach utilizes a neural network to map power system loads to generator setpoints. A feasibility restoration layer is employed to restore base case infeasible predictions. By incorporating line outage distribution factors (LODFs), all post-contingency flows are computed. The loss function utilized to train the neural network draws inspiration from the penalty function method. Lastly, a copula analysis computes joint outage probabilities for k >1 enabling a probabilistic security assessment. The first academic contribution of this thesis is the development of a constraint-driven approach to approximate N-k SCOPFs considering all contingencies using LODFs. The second academic contribution is the formulation of a N-k risk based security criterion, providing an alternative to the current deterministic N-1 security criterion.

The approach shows promise in its ability to scale effectively to N-k contingencies. Using LODFs, the approach effectively computes all post-contingency flows for up to k = 3. Moreover, case studies show the constraint-driven approach's effectiveness in identifying violating post-contingency cases, with up to 173$\times$ speedups and close to optimal dispatch costs. However, the consideration of N-k contingencies holds combinatorial complexity, and more efficient methods need to be developed for the computation and storage of all LODFs, and for the computation of all post-contingency flows. Additionally, the proposed constraint-driven approach can not enforce any post-contingency constraints, necessitating post-contingency feasibility checks when security against specific contingencies is required. Next, by incorporating probabilities, the approach shows promise in improving power systems security and resilience, but further research is necessary.

In this thesis, only line outages are considered. In the future, the approach could be modified to additionally account for other equipment outages (e.g. generator outages). Furthermore, future research could investigate the adoption of this approach in corrective control settings, where it is employed in the restorative phase of a contingency event. Another suggestion is centered around the incorporation of graph neural networks in the proposed approach, which could provide a more scalable alternative to fully connected linear neural networks. Furthermore, more scalable methodologies could be explored to construct the matrix containing all LODFs, and a more scalable methodology for computing all post-contingency flows could be developed. Finally, future work could investigate how to utilize the proposed approach under varying conditions like network topology changes or changing outage probabilities.

Contents

Nomenclature

Abbreviations

Abbreviation	Definition
ACOPF	Alternating Current Optimal Power Flow
BD	Benders Decomposition
C&CGA	Column and Constraint Generation Algorithm
COO	Coordinate
cctkmyr	Circuit-kilometer-year
CS	Contingency Screening
CSC	Compressed Sparse Column
CSR	Compressed Sparse Row
DCOPF	Direct Current Optimal Power Flow
ED	Economic Dispatch
EENS	Expected Energy Not Supplied
FLODF	Full Line Outage Distribution Factor
FPTDFc	Full Power Transfer Distribution Factor in contingency
GNN	Graph Neural Network
GCNN	Graph Convolutional Neural Network
H	Heuristic
HILP	High Impact Low Probability
KKT	Karush Kuhn Tucker
LODF	Line Outage Distribution Factor
LOLE	Loss of Load Expectation
ML	Machine Learning
MLP	Multi-Layer Perceptron
OPF	Optimal Power Flow
PTDF	Power Transfer Distribution Factor
p.u.	per unit
ReLU	Rectified Linear Unit
RO	Robust Optimization
SCOPF	Security Constrained Optimal Power Flow
SLODF	Single Line Outage Distribution Factor
SVD	Singular Value Decomposition
TSO	Transmission System Operator
TT	Tensor Train
TTD	Tensor Train Decomposition

Symbols

Symbol	Definition	Unit
$\mathbf{A}$	Branch incidence matrix	[-]
$\mathbf{B}$	Susceptance matrix	[Ω]
$\mathbf{B_{br}}$	Diagonal susceptance matrix	[Ω]
$\mathbf{c_G}$	Vector of generator cost	[$/MW]
c	Contingency index	[-]
$\mathbf{C}$	Covariance matrix	[-]
d	Distance	[km]
$\mathbf{E}$	Identity matrix	[-]
$\mathcal{E}$	Set of edges	[-]
$\mathbb{E}$	Expected value	[-]
f	Function	[-]
F	Line flow	[MW]
$\mathcal{F}$	Cumulative distribution function	[-]
$g(\cdot)$	Inequality constraints	[-]
$\mathbf{G}$	Conductance matrix	[Ω]
$\mathcal{G}$	Graph	[-]
$h(\cdot)$	Equality constraints	[-]
$\mathbf{H}_u$	Graph weight matrix	[-]
i, j	Bus indices	[-]
I	Current	[A]
k	Number of simultaneous outages	[-]
l, m	Line indices	[-]
L	Characteristic line length	[km]
L_l	Line length	[km]
M	Set of monitored lines	[-]
n	Generator index	[-]
N	Number of elements	[-]
O	Set of outaged lines	[-]
p	Independent probability	[-]
P_G	Active power	[MW]
$\widehat{P}_G$	Predicted active power	[MW]
$\mathbf{P_{N-k}}$	Vector of probabilities	[-]
P_{shed}	Shedded load	[MW]
Q	Reactive power	[MVAr]
$\mathbb{R}$	Real numbers	[-]
s	Load index	[-]
$\mathbf{T}$	Shifted graph signal	[-]
U	Hop count	[-]
V	Voltage magnitude	[V]
$\mathcal{V}$	Set of nodes	[-]
$\mathbf{W}$	Adjacency matrix	[-]
$\mathbf{X}$	Reactance matrix	[Ω]
x	Line reactance	[Ω]
y	Variable	[-]
$\mathbf{Y}$	Random variable	[-]
$\mathbf{Z}$	Graph signal	[-]

Symbol	Definition	Unit
0	Pre-contingency state	[-]
α	Scaling factor	[-]
$\gamma^{+,-}$	Slack terms	[-]
δ	Voltage phase angle	[rad]
Δ	Difference	[-]
$\lambda_{0,1,2}$	λ with a number is a penalty term	[-]
$\lambda_{h,g}$	λ with a letter is a Lagrangian multiplier	[-]
μ	Mean	[-]
ρ	Correlation	[-]
σ	Standard deviation	[-]
$\phi(\cdot)$	Non-linearity	[-]
Ω^C	Set of contingencies	[-]
Ω^B	Set of buses	[-]
Ω^D	Set of loads	[-]
Ω^F	Set of features	[-]
Ω^G	Set of generators	[-]
Ω^L	Set of lines	[-]
Ω^S	Set of samples	[-]

List of Figures

List of Tables

1 Introduction

The energy sector is swiftly evolving towards sustainability, driven by the increasing integration of renewable energy sources, expanding distributed energy assets, and electrification across diverse domains [1]. These developments pose a challenge to grid security, or operational reliability, making it harder to meet the N-1 security criterion, the failure of which has led to numerous blackouts. Furthermore, unforeseen weather events, now more frequent due to climate change, account for the second major cause of these blackouts [2, 3]. The growing grid complexity highlights the importance of N-k outages, where k lines fail simultaneously, necessitating enhanced N-k security and increased resilience against high impact low probability (HILP) weather triggered events [4]. While the N-1 criterion historically balanced security and computational efficiency, recent developments advocate for a probabilistic risk-based assessment, encompassing N-k contingencies with k >1, to enhance power system operational reliability [5]. Conventional methods encounter difficulties in effectively tackling the computationally demanding N-k security due to its inherent combinatorial complexity. This thesis focuses on the optimization problem faced by transmission system operators (TSOs), known as N-k security constrained optimal power flow (SCOPF), which seeks to find the most cost-efficient dispatch which is secure against k outages.

1.1. Conventional Approaches

In the context of solving N-k SCOPF problems, different approaches have been introduced to reduce problem size. In [6, 7], Benders decomposition (BD) [8] is used, to decompose the problem into a master problem and subproblems representing contingency cases. Furthermore, [9] employs a column-and-constraint generation algorithm (C&CGA) to iteratively add the most violating contingencies. However, these approaches face challenges like slow convergence when dealing with numerous contingencies or the risk of generating infeasible predictions when too many constraints are introduced. Alternatively, [10] employs an iterative contingency screening method to identify the most critical contingencies, allowing the SCOPF to be solved with a selected subset of contingencies. In [11, 12], this screening process leverages line outage distribution factors (LODFs) due to their computational efficiency in managing multiple contingencies. The proposed approaches in [6, 7, 9, 10, 11, 12] do not scale with the number of outages k, as demonstrated in [13]. A comparison between BD, a combination of C&CGA with robust optimization (RO), and LODFs reveals issues such as timeouts and slow convergence for N-3. This is particularly limiting for addressing the N-k SCOPF problem that requires multiple daily resolutions.

The recurrent nature of the N-k SCOPF problem has prompted exploration of machine learning (ML) techniques as a means to approximate its solution. Reference [14] predicts generator setpoints from power system loads for N-1 SCOPF using a multi-layer perceptron (MLP). Similarly, reference [15] utilizes a MLP to classify binding constraints. Both approaches in [14, 15] do not scale with increasing k. This is because they rely on training data, which becomes exceedingly difficult to generate for k >1. To address these limitations, [16] proposes a C&CGA-ML algorithm in combination with RO, iteratively adding constraints, succesfully solving N-1 SCOPFs, but unable to solve N-k SCOPFs. Moreover, in [17], an adversarially robust ML approach is introduced, using a min-max strategy where the worst-

case contingency is computed and incorporated in the SCOPF. However, the success of [17] is highly reliant on the ability to identify the worst-case contingency. To address the issue of prediction feasibility, recent attention within various engineering disciplines has focused on the integration of constrained optimization and ML techniques [18, 19]. However, these methods also struggle in finding secure and feasible predictions while maintaining scalability with increasing k. Therefore, there is a pressing need to develop novel, practical methodologies that can effectively approximate the N-k SCOPF problem.

1.2. Proposed Approach

This thesis proposes a constraint-driven ML approach to approximate the N-k SCOPF problem, considering all possible contingencies. The approach utilizes the computationally efficient linearized direct current optimal power flow which is widely adopted in transmission network operation [20]. A MLP learns the mapping from loads to generator setpoints. With the use of LODFs, all post-contingency flows are efficiently computed through a simple matrix multiplication. The LODF matrices are expressed in sparse format to significantly reduce memory cost and accelerate the computation. An algorithm is developed to reduce the computational graph which guides the backpropagation process. The proposed approach is constraint-driven and does not require labeled training data, eliminating the necessity of solving the computationally expensive N-k SCOPF [13]. Furthermore, to formulate a risk-based security criterion, contingency probabilities are included during training. These probabilities are computed considering a spatial correlation between individual line outages which is characteristic to extreme weather phenomena [21, 22]. The performance of the security assessment is evaluated using loss of load expectation and expected energy not supplied.

1.3. Research Questions

This section outlines the primary academic contributions of this thesis, along with their associated research questions:

Objective 1: Develop a deterministic constraint-driven approach to approximate N-k SCOPFs, considering all possible contingencies using LODFs.

- Q1 Can LODFs be effectively employed to approximate N-k SCOPFs, incorporating all contingencies, in a scalable and computationally efficient manner?
- Q2 Can a constraint-driven ML approach reduce post-contingency violations?

Objective 2: Perform a probabilistic security assessment to formulate an N-k risk-based security criterion, providing an alternative to the current deterministic N-1 security criterion.

- Q3 Does a probabilistic risk-based security criterion incorporating all contingencies improve power systems security?
- Q4 Does this proposed risk-based security criterion increase power systems resilience?

1.4. Thesis Outline

Chapter 2 discusses the basics on electric power system optimization, including a description of the SCOPF problem. Additionally, this chapter clarifies the foundational ideas in ML, ML for constrained optimization and mathematical fundamentals. Moving forward, Chapter 3 provides an overview of the popular conventional methods and current ML approaches for approximating the N-k SCOPF problem. To give a thorough analysis of the key issues, both their advantages and disadvantages will be outlined. The chapter wraps up with a discussion on power systems operational reliability and resilience. Chapter 4 sequentially explains the proposed approach for approximating the N-k SCOPF problem. This includes an explanation of the probabilistic framework. Chapter 5 presents the case studies for testing the proposed approach. Finally, Chapter 6 provides a discussion and conclusion including suggestions for future work. A schematic of the thesis outline is presented in Figure 1.1.

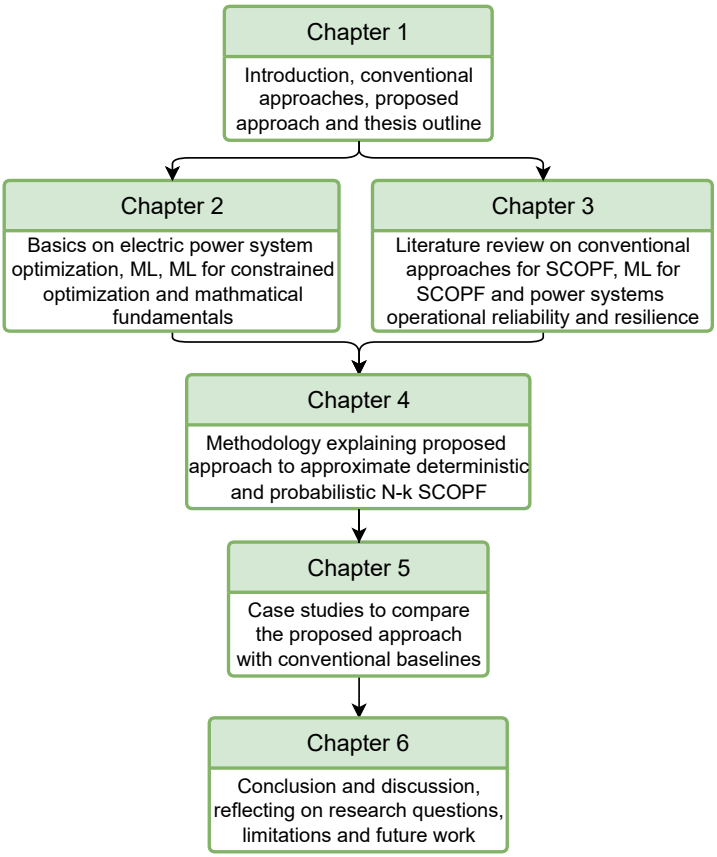


Figure 1.1: Schematic illustration of the thesis outline.

2 Background

2.1. Electric Power System Optimization

At the heart of the optimization problems to determine the optimal mode of operation of the power grid lies the optimal power flow (OPF) problem. Transmission system operators (TSOs) solve this problem multiple times an hour to balance generation with the constantly fluctuating demand. The OPF problem and its different variants will be briefly discussed.

2.1.1. Economic Dispatch

Economic dispatch (ED) is a power systems problem that aims to find the lowest-cost generation dispatch for a given system. In the end, the total generation equals the total load [23]. The only constraints that are considered are the generator limits. The network is assumed to be a copper plate, meaning that infinite transmission capacity is assumed and network constraints are neglected [24]. Optimal power flow is an optimization problem which couples this ED with the power flow equations and solves them simultaneously. Not only is the optimal generation dispatch found, the power flow equations are also solved and the physical network constraints are taken into account. These constraints include the generator limits, voltage limits and line flow limits. Optimal power flow can be done by using the full AC power flow equations or by using the simplified DC power flow equations.

2.1.2. AC Optimal Power Flow

The AC optimal power flow (ACOPF) problem provides the most comprehensive description of the OPF problem and is given in Eq. (2.1). It employs the AC power flow equations, presented in Eq. (2.1b) and Eq. (2.1b), which most accurately capture the system's physics. As these equations contain the square of the voltage and trigonometric functions, they are quadratic in nature and non-convex. This results in a non-linear and non-convex optimization, making it more complex to solve, especially when the problem increases with size. The objective of the optimization problem as described in Eq. (2.1a) is to minimize the generator dispatch cost. The dispatch cost is the summation of the individual generator power outputs multiplied by their respective cost factors. Eq. (2.1b) - Eq. (2.1i) show the set of equality and inequality constraints. These constraints are respectively the AC power flow equations, the line flow limits, the current limits, the active and reactive power limits of the generators, the bus voltages and the voltage angles. The set Ω^G represents the set of all generators, the set Ω^L represents the set of all lines and the set Ω^B represents the set of all buses. The ACOPF problem is formulated as follows.

$$\min_{n \in \Omega^G} \sum c_{G_n} P_{G_n} \qquad \forall n \in \Omega^G \tag{2.1a}$$

subject to:

$$P_i = \sum_{j \in \Omega^B} (|V_i||V_j|(G_{ij}\cos(\delta_i - \delta_j) + B_{ij}\sin(\delta_i - \delta_j))) \tag{2.1b}$$

$$Q_i = \sum_{j \in \Omega^B} (|V_i||V_j|(G_{ij}\sin(\delta_i - \delta_j) - B_{ij}\cos(\delta_i - \delta_j))) \tag{2.1c}$$

$$F_l^{min} \leq F_l \leq F_l^{max} \quad \forall l \in \Omega^L \tag{2.1d}$$

$$I_l^{min} \leq I_l \leq I_l^{max} \quad \forall l \in \Omega^L \tag{2.1e}$$

$$P_{G_n}^{min} \leq P_{G_n} \leq P_{G_n}^{max} \quad \forall n \in \Omega^G \tag{2.1f}$$

$$Q_{G_n}^{min} \leq Q_{G_n} \leq Q_{G_n}^{max} \quad \forall n \in \Omega^G \tag{2.1g}$$

$$V_i^{min} \leq V_i \leq V_i^{max} \quad \forall i \in \Omega^B \tag{2.1h}$$

$$\delta_i^{min} \leq \delta_i \leq \delta_i^{max} \quad \forall i \in \Omega^B \tag{2.1i}$$

2.1.3. DC Optimal Power Flow

Alternatively, the linearized DC optimal power flow problem (DCOPF) can be utilized to solve the OPF problem and is given in Eq. (2.2). The DCOPF employs the linearized DC power flow equations, given in matrix form in Eq. (2.2b). To convert the ACOPF equations into the DCOPF equations, certain assumptions are applied: fixed bus voltage magnitudes set to 1 p.u., absence of reactive power considerations, and the assumption of small voltage angle differences. Despite the decreased accuracy with respect to an ACOPF, a DCOPF still gives a reasonable representation of reality and is convex, enabling more straightforward computations. Consequently, the DCOPF is widely used by TSOs as a support tool to operate the grid [20]. Similarly to the ACOPF problem, the objective is to minimize the generator dispatch cost Eq. (2.2a). Eq. (2.2b) - Eq. (2.2d) are the equality and inequality constraints for this optimization problem. These constraints are respectively the DC power flow equations, the line flow limits and the generator limits. The optimization problem is formulated as follows.

$$\min_{n \in \Omega^G} \sum c_n P_{G_n} \qquad \forall n \in \Omega^G \tag{2.2a}$$

subject to:

$$\mathbf{B} \cdot \boldsymbol{\delta} = \mathbf{P_G} - \mathbf{P_D} \tag{2.2b}$$

$$F_l^{min} \leq \frac{1}{x_{ij}}(\delta_i - \delta_j) \leq F_l^{max} \quad \forall i,j \in \Omega^B, \forall l \in \Omega^L \tag{2.2c}$$

$$P_{G_n}^{min} \leq P_{G_n} \leq P_{G_n}^{max} \quad \forall n \in \Omega^G \tag{2.2d}$$

Sensitivity Factors

In the context of the DCOPF problem, linear sensitivity factors such as the power transfer distribution factor (PTDF) and line outage distribution factor (LODF) can be employed to compute approximate changes in line flows for changes in power injections or network configuration [23]. These sensitivity factors, which depend on the topology and characteristics of the network, make it possible to quickly recompute network parameters in response to shifting circumstances.

Power Transfer Distribution Factor The PTDF gives the incremental change of flow over a line due to an injection or withdrawl of power at a certain bus. In the following formulation $\mathbf{A}$ is the branch incidence matrix $\mathbf{A} \in \{-1, 1\}^{|\Omega^L| \times |\Omega^B|}$, where 1 indicates the 'from bus' and -1 indicates the 'to bus'. The matrix $\mathbf{B} \in \mathbb{R}^{|\Omega^B| \times |\Omega^B|}$ is the susceptance matrix which is computed using the line reactances x_{ij} as follows.

$$B_{ii} = \sum_{j=i} \frac{1}{x_{ij}} \quad \forall i, j \in \Omega^B \tag{2.3a}$$

$$B_{ij} = -\frac{1}{x_{ij}} \quad \forall i, j \in \Omega^B \tag{2.3b}$$

To avoid singularity, the row and the column corresponding to the slack bus are removed from the susceptance matrix $\mathbf{B}$. The matrix $\mathbf{X}$ is the inverse of the susceptance matrix $\mathbf{X} = \mathbf{B}^{-1}$. Finally, the matrix $\mathbf{B_{br}} \in \mathbb{R}^{|\Omega^B| \times |\Omega^B|}$ is a diagonal matrix with the line reactances on the diagonal entries, and zeros elsewhere. The PTDFs can be computes as

$$\mathbf{PTDF} = \mathbf{B_{br}} \times \mathbf{A} \times \mathbf{X} \tag{2.4}$$

Subsequently, the change of flow over a line can be computed by the change of power injection at a bus, multiplied by the PTDF.

$$\Delta \mathbf{F} = \mathbf{PTDF} \cdot \Delta \mathbf{P} \tag{2.5}$$

In the context of the DCOPF problem, the PTDF can also be used to reformulate the line flow constraints. This removes the need for computing the phase angels to determine the line flow.

$$\mathbf{PTDF}(\mathbf{P_G} - \mathbf{P_D}) \leq \mathbf{F^{max}} \tag{2.6}$$

DCOPF Formulation Based on PTDF By employing PTDFs to formulate the DCOPF problem, the need for nodal balance equations and variables δ is eliminated. Instead, Eq. (2.7b) is introduced to ensure the total generation equals the total demand. The DCOPF formulation based on PTDFs is provided below:

$$\min_{n \in \Omega^G} \sum c_n P_{G_n} \qquad \forall n \in \Omega^G \tag{2.7a}$$

subject to:

$$\sum_{n \in \Omega^G} P_{G_n} - \sum_{s \in \Omega^D} P_{D_s} = 0 \tag{2.7b}$$

$$\mathbf{PTDF}(\mathbf{P_G} - \mathbf{P_D}) \leq \mathbf{F^{max}} \tag{2.7c}$$

$$P_{G_n}^{min} \leq P_{G_n} \leq P_{G_n}^{max} \quad \forall n \in \Omega^G \tag{2.7d}$$

Line Outage Distribution Factor Similarily, LODFs can be employed to compute the change of flow on a line following the outage of another line in the system. Because of this characteristic, LODFs are useful for determining line overloads in contingency cases. The method of generalized line outage distribution factors for computing LODFs is given below [25]:

$$\mathbf{LODF_{M,O}} = \mathbf{PTDF_{M,O}}(\mathbf{E} - \mathbf{PTDF_{M,O}})^{-1} \tag{2.8}$$

The letter M denotes the set of monitored lines, the letter O denotes the set of outaged lines and the superscript 0 indicates the pre-contingency state. Here, $\mathbf{E} \in \mathbb{R}^{|\Omega^L| \times |\Omega^L|}$ is the identity matrix. The matrix $\mathbf{A}$ is the bus-to-monitored line incidence matrix. Additionally, matrix $\hat{\mathbf{A}}$ is the bus-to-tripped line incidence matrix where $\hat{\mathbf{A}} \in \{-1, 1\}^{|\Omega^L| \times |\Omega^B|}$. $\mathbf{PTDF^0_{M,O}}$ and $\mathbf{PTDF^0_{O,O}}$ are computed as:

$$\mathbf{PTDF^0_{M,O}} = \mathbf{B_{br}} \times \mathbf{A} \times \mathbf{X} \times \hat{\mathbf{A}}^T \tag{2.9}$$

$$\mathbf{PTDF^0_{O,O}} = \mathbf{B_{br}} \times \hat{\mathbf{A}} \times \mathbf{X} \times \hat{\mathbf{A}}^T \tag{2.10}$$

As depicted in Eq. (2.11), the change of flow on a line can be computed by multiplying the pre-contingency flow of the outaged line with the LODF. The post-contingency flow can then be determined by adding the change of flow to the pre-contingency line flow as shown in Eq. (2.12).

$$\Delta F_l = LODF \cdot F_l^0 \tag{2.11}$$

$$F_l^c = F_l^0 + \Delta F_l \tag{2.12}$$

2.1.4. Security Constrained Optimal Power Flow

The security constrained optimal power flow (SCOPF) problem seeks to find the most cost-efficient dispatch which is also secure in contingency cases, playing a critical role in maintaining a reliable power system [26]. The SCOPF adresses the OPF problem in both the base case and during one or multiple contingency cases, defined as instances where line or equipment failures lead to outages. The SCOPF can be formulated using both the AC and DC power flow equations. In the context of DC power flow equations, the only additional constraints compared to a DCOPF are the line flow limits during contingency cases. In the formulation in Eq. (2.13), the set Ω^C represents the set of all contingency cases, with $c = 0$ indicating the base case without any contingencies.

$$\min_{n \in \Omega^G} \sum c_n P_{G_n} \qquad \forall n \in \Omega^G \tag{2.13a}$$

subject to:

$$\mathbf{B_c} \cdot \boldsymbol{\delta_c} = \mathbf{P_G} - \mathbf{P_D} \qquad \forall c \in \Omega^C \tag{2.13b}$$

$$F_l^{min} \leq \frac{1}{x_{ij,c}}(\delta_{i,c} - \delta_{j,c}) \leq F_l^{max} \quad \forall i,j \in \Omega^B, \forall l \in \Omega^L, \forall c \in \Omega^C, (i,j) \neq c \tag{2.13c}$$

$$P_{G_n}^{min} \leq P_{G_n} \leq P_{G_n}^{max} \qquad \forall n \in \Omega^G \tag{2.13d}$$

The letter k denotes the number of simultaneous outages. For a N-1 SCOPF, the SCOPF must be secure against any single line outage. For a N-2 SCOPF, the SCOPF must be secure against any double line outage. As the number of contingency cases $|\Omega^C|$ holds combinatorial complexity $\binom{N}{k}$, the size of the SCOPF problem increases exponentially as k increases, presenting a big computational challenge.

The SCOPF problem can be approached in either a preventive or corrective fashion. In preventive SCOPF, the system operating conditions must fulfill both pre- and post-contingency constraints and remain constant once established. On the other hand, corrective SCOPF allows for redispatching following an outage. This indicates that going above network restrictions within a given time frame is acceptable if the violations can be mitigated through redispatching.

2.2. Machine Learning

Having established the fundamentals on power system optimization, the next section will discuss basic concepts of machine learning (ML) [27] for this thesis. Since directly solving a SCOPF is computationally challenging, the objective of this thesis is to utilize ML to develop a proxy that can approximate the SCOPF, eliminating the need to directly solve it.

2.2.1. Neural Network Architectures

Multi-Layer Perceptron

A neural network is a ML model inspired by the human brain. Figure 2.1 shows the most common type of neural network, a fully connected linear neural network, otherwise known as a multi-layer perceptron

(MLP). It is designed to learn and predict complicated functions and patterns from data and is built as a network of interconnected artificial neurons structured in layers, comprising an input layer, one or more hidden layers, and an output layer [28]. Each neuron employs an activation function to its input, transmitting the output to the subsequent layer. The inter-neuron connections, characterized by weights, are adjusted during training using an algorithm called backpropagation [28]. This iterative process minimizes a user defined loss function, fine tuning the parameters of the model after each iteration. During backpropagation, the gradients of the error are computed with respect to the weights. MLPs are very expressive, and are well suited to develop a proxy which can approximate an OPF [14, 29, 30, 31].

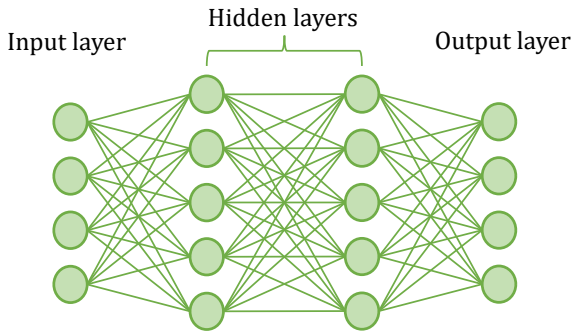


Figure 2.1: Schematic representation of a fully connected linear neural network, known as a multi-layer perceptron (MLP).

Graph Neural Networks

Delving into graph neural networks (GNNs), these are a particular category of neural networks tailored to process graph-structured data. In situations like power systems, which are intrinsically defined by graph-structured data, GNNs find practical applicability since they are particularly adept at revealing underlying connections within such data. GNNs have not only found applicability in power flow computations, but also in fault scenario applications or time series predictions [32].

Lets denote a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the set of nodes Ω^B and $\mathcal{E}$ is the set of edges or branches Ω^L. Introducing matrix $\mathbf{W}$, defined as the adjacency matrix of the network where $W_{i,j} = w_{i,j}$ if $(i, j) \in \Omega^L$. Here, a graph signal $\mathbf{Z} \in \mathbb{R}^{|\Omega^B| \times |\Omega^F|}$ where $\mathbf{Z}$ signifies the state of the nodes, with each node associated with a set of features Ω^F.

Leveraging the adjacency matrix $\mathbf{W}$, a shifted version of signal $\mathbf{Z}$ can be defined as:

$$\mathbf{T} = \mathbf{W}\mathbf{Z} \tag{2.14}$$

Resulting in a new graph signal $\mathbf{T} \in \mathbb{R}^{|\Omega^B| \times |\Omega^F|}$, which is a shifted version of $\mathbf{Z}$ across the graph. This shift operation involves updating each state in the graph based on a linear combination of states from neighboring nodes. In a convolution, adjacent values are similarly combined linearly, to use an analogy. The idea of a graph convolution, which is basically a linear combination of a graph signal that has experienced a shift, is created by integrating both the shift operation and a convolution.

The result of the shift of $\mathbf{Z}$ is weighted by $\mathbf{H}_u \in \mathbb{R}^{|\Omega^F| \times |\Omega^G|}$. The matrix $\mathbf{H}_u$ takes the input features Ω^F and returns a new set of features Ω^G which are a linear combination of the input features. Consequently, the output assumes dimensions $\mathbf{Y} \in \mathbb{R}^{|\Omega^B| \times |\Omega^G|}$.

Information from U-hop neighbors can be obtained by repeated application of $\mathbf{W}$ resulting in $\mathbf{W}^u\mathbf{Z}$. This can be achieved locally by performing u exchanges with 1-hop neighbors. To go from these graph convolutions to a layer in a GNN, a sequence of K graph convolutions are performed to which a non-linearity $\phi(\cdot)$ is applied.

$$\mathbf{T} = \phi\left(\sum_{u=0}^{U-1} \mathbf{W}^u \mathbf{Z}\mathbf{H}_u\right) \tag{2.15}$$

2.2.2. Training Neural Networks

Having established the neural network architectures, the next step involves defining training methodologies for these models. One frequently employed technique for training ML models is supervised learning [27]. In this method, an algorithm learns from labelled training data, where the dataset includes input data paired with their corresponding correct outputs. The goal is to learn the algorithm the mapping or function that can predict the correct output for new, unseen input data. The algorithm learns by minimizing the difference between its predictions and the true labels presented in the training data. When utilized to approximate a DCOPF, the input is the demand while the labels are the generator setpoints [29]. While supervised learning methods prove effective, their applicability becomes limiting in situations where labelled training data is unavailable. The applicability of supervised learning in the context of N-k SCOPFs is limited, since obtaining training data is computationally challenging.

When confronted with limited labelled training data, alternative learning methods come into play. Semi-supervised learning is an approach where a small portion of labelled data is used in combination with a large amount of unlabelled data [28]. Moreover, in scenarios where the model has to adhere to certain constraints, constraint-driven learning can be used. Optimization problems which have to adhere to hard constraints occur in many settings. The OPF problem is an example of an optimization problem with hard physical and engineering constraints. Consequently, the fusion of ML and constrained optimization has attracted significant interest [18]. The following section will delve deeper in ML for constrained optimization.

2.3. Machine Learning for Constrained Optimization

By fusing ML and mathematical optimization, results can be obtained which the two fields cannot achieve independently [18]. A big opportunity for enhancing optimization with ML is when the optimization is simply too slow. This often happens when there are either real time constraints which have to be solved, or when the simulation is too large. In the power systems domain, there is also an increasing interest in ML models which can be used as approximate solvers for complex power system tasks [14, 16, 17]. For a large problem like the N-k SCOPF problem, ML can be extremely useful to get an approximate solution.

Naive deep learning approaches can not strictly enforce the constraints which are imposed on the optimization problem. This can result in infeasible solutions. The field of integrating solvers or optimization methods into deep learning is referred to as end-to-end constrained optimization. This domain can subsequently be divided into two areas [18]. The first area approaches constrained optimization as a layer in a neural network structure. This will be referred to as predict-and-optimize. The second area tries to predict approximate solutions of constrained optimization problems using ML architectures. This will be referred to as learning with constraints.

2.3.1. Predict-and-Optimize

Predict-and-optimize methods are a combination of prediction models (ML models) and decision models (constrained optimization models). The specification of the partially defined decision models are completed by predictions from data. Based on the accuracy of the model's output, the model is trained end-to-end. In this area, the goal is to create a model that can learn to make decisions based on input data [18].

Implicit Formulation

Often the layers in a neural network are explicitly defined, where there is a direct, closed-form mapping from input to output [28]. These explicit functions are computationally cheap and their gradients can be easily computed allowing for backpropagation. Much richer behaviour can be expressed by implicitly defining the relationship between input and output. An implicit function as such cannot be solved in closed form; instead, an iterative approach, akin to Newton's technique, must be used to achieve the solution. In recent years, the use of an implicit layer in a neural network setting has been investigated

[33, 34, 35]. A category of implicit layers in neural networks are optimization layers such as [33, 34]. The output of these layers is the solution of a constrained optimization problem based on its preceding layers. In principle, such layers can be used to enforce constraints, by projecting the output of a ML model on a constraint set [19]. In the context of the OPF problem, such a layer can also be used to directly solve the optimization.

To be able to employ such a layer in a neural network, it has to support a forward pass and a backward pass. As discussed, the forward pass of an implicit layer is implemented using an iterative solution method such as Newton's. The backward pass is performed using implicit differentiation. For an optimization problem, the conditions ensuring stationarity, primal feasibility, dual feasibility and complementary slackness are the Karush Kuhn Tucker (KKT) conditions [36]. For convex optimization problems such as the DCOPF problem, these conditions are the necessary and sufficient conditions for indicating the optimal solution. Therefore, for convex problems, the gradients can be computed by implicitly differentiating these KKT conditions at its solution. For convex problems, such a layer can scale nicely. For quadratic programming layers, such layers have cubic complexity in the number of variables and constraints and therefore do not scale as nicely [33].

2.3.2. Learning with Constraints

Alternatively to a predict-and-optimize method, learning with constraints methods develop ML architectures to predict approximate solutions to predefined constrained optimization problems, without the use of solvers at the time of inference [18]. In [19], a framework is proposed applying deep learning to a wider range of optimization problems with hard constraints, showing its capacity to solve the non-convex ACOPF problem. Using differentiable *equality completion* and *inequality correction*, the constraints are incorporated in the training loop. This framework leverages insights from the previously discussed optimization layers ([33, 34]). The framework employs implicit differentiation for backpropagation through the equality constraints, providing a less computationally expensive alternative for these optimization layers.

Constrained to Unconstrained Optimization

In the context of learning with constraints, a possible avenue is to directly incorporate constraint violations in the loss function. This is enabled by transforming a constrained optimization problem into an unconstrained one. This is precisely what the penalty function method accomplishes, it transforms a constrained optimization problem into an unconstrained counterpart by augmenting the objective function with penalty terms for constraint violations, each scaled by a penalty coefficient. Consider the following optimization problem:

$$\min f(y) \tag{2.16a}$$

subject to:

$$h(y) = 0 \tag{2.16b}$$
$$g(y) < 0 \tag{2.16c}$$

The goal becomes to solve the unconstrained optimization problem in the form of:

$$\min f(y) + \lambda_1 |g(y)| + \lambda_2 \max(0, h(y)) \tag{2.17}$$

Here, $\lambda_{1,2}$ denote penalty coefficients assigned to constraint violations, and these coefficients are iteratively adjusted until the constraints are close to being satisfied. Nevertheless, the penalty function method transforms the problem into a fully unconstrained one, potentially resulting in infeasible approximations, as demonstrated for an ACOPF in [19]. An alternative strategy involves a Lagrangian relaxation approach based on constraint violations, as discussed in [37]. This formulation is based on augmented Lagrangian relaxations [38]. The violation-based Lagrangian function becomes:

$$f(y) + \lambda_g |g(y)| + \lambda_h \max(0, h(y)) \tag{2.18}$$

And the optimization problem becomes:

$$LR_\lambda = \underset{y}{\operatorname{argmin}}(f(y) + \lambda_g|g(y)| + \lambda_h \max(0, h(y))) \tag{2.19}$$

In this formulation, λ_g and λ_h serve as Lagrangian multipliers. This is similar to the penalty method function, where now the augmented Lagrangian can be seen as the unconstrained optimization problem. However, by leveraging Lagrangian duality and by solving the dual problem, the constraints can be satisfied, whereas the optimization in the penalty method function is completely unconstrained. The Lagrangian dual aims to find the best Lagrangian multipliers, and is given by:

$$LD = \underset{\lambda}{\operatorname{argmax}}(LR_\lambda) \tag{2.20}$$

A combination of Lagrangian duality and deep learning has proved to be effective in predicting ACOPFs [39].

2.4. Mathematical Fundamentals

2.4.1. Exploring Convexity

A function f is convex if the line segment connecting any two points on a graph lies above the graph f itself as shown in Figure 2.2. An optimization problem is convex if its objective function is convex, the inequality constraints are convex and the equality constraints are affine [36]. A function is affine, if it can be expressed by a linear transformation, and a linear translation. Especially in the context of optimization, optimization problems which are convex have some desirable properties. One of the properties of convex optimization problems is that the optimal solution lies on the border of the solution space. Another fundamental property in the context of convex optimization is that a local optimum in the feasible space is also immediately a global optimum [36]. Moreover, if the solution space is an intersection of a convex set, the solution space is also convex. These properties make convex optimization problems noticeably more tractable than their non-convex counterparts. The solution of convex optimization problems can be obtained with increased efficiency, reliability, and with robust guarantees about the optimal solution.

As an additional note in the context of optimization problems, if an optimization problem contains integers, the solution space becomes disjunctive and not continuous.

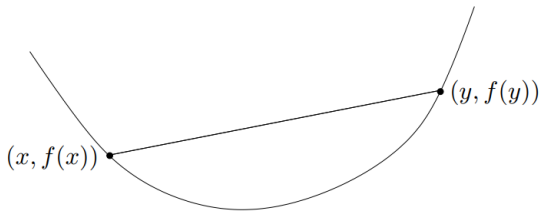


Figure 2.2: Schematic visualisation of a convex function [36].

2.4.2. Tensors

A tensor is a multidimensional representation of a vector or matrix [40]. Up to 3 dimensions, a tensor can be intuitively visualized as shown in Figure 2.3. When more dimensions are considered, this representation gets more abstract. Reducing the storage complexity of large, high-dimensional tensors is a topic of interest in academia [40, 41, 42, 43].

Tensor Decomposition One possible tensor decomposition method is called tensor train decomposition (TTD) [41, 42, 43]. TTD decomposes a large tensor into smaller tensors, referred to as cores, with lower ranks. When these cores are arranged sequentially, they resemble a *train* of tensors, hence the

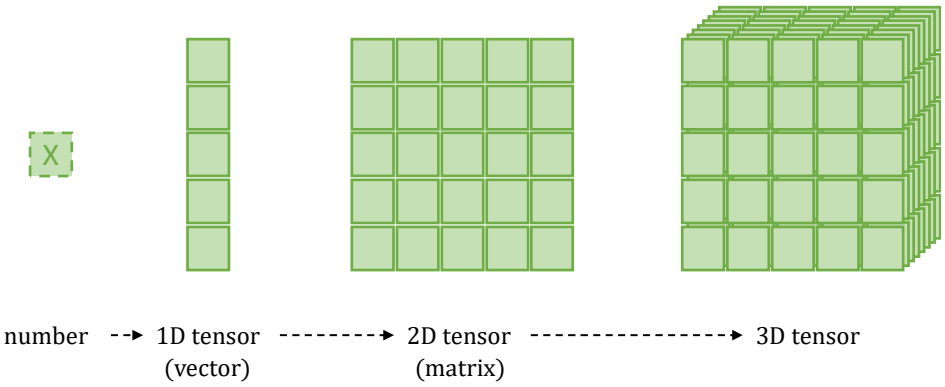


Figure 2.3: Schematic visualisation of a tensor up to three dimensions.

name tensor train. TTD reduces the exponential storage complexity of a tensor to a quadratic storage complexity. The figure below illustrates a standard graphical representation of a tensor train.

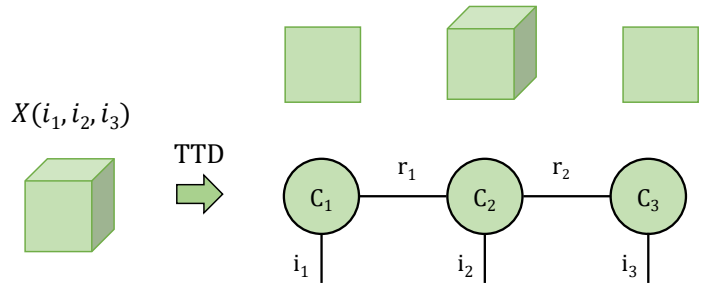


Figure 2.4: Schematic visualisation of a 3D tensor X, decomposed using tensor train decomposition into a tensor train with three cores.

Once a tensor is transformed to TT format, all operations should also be done in TT format to keep the advantage of the compression. Many basic operations can be easily performed in TT format. As a consequence of many of these operations, the storage complexity increases, and a recompression step called *rounding* has to be performed. This rounding step introduces a trade-off between accuracy and storage complexity.

When extensive tensors are involved which result in memory issues, first constructing the full tensor, and then applying TTD is not possible. In this case, the TT has to be constructed alternatively. One approach could be to construct the TT with a stochastic process called *randomization*. Without prior knowledge about the structure of the TT, this method results in very large errors. Another option is to recursively build the TT. This involves building up the full TT in small chunks and iteratively performing adding and rounding steps. The final compression achieved with TTD depends on the structure of the tensor and the similarities between dimensions. For a more extensive background on TTD, Appendix D can be consulted.

2.4.3. Sparse Matrix Handling

Sparse matrices are matrices in which most of the elements are zero. Therefore, storing a sparse matrix as a dense matrix would require significant valuable memory resources. To reduce the storage requirements for a sparse matrix, it can be expressed in sparse format. When a matrix is expressed in sparse format, only the non-zero values and their corresponding locations in the matrix are stored. Expressing sparse matrices in sparse format can reduce the number of stored elements significantly. Among others, there are three popular methods to express a matrix in sparse format. The most intuitive method is the coordinate list (COO) format, where simply the non-zero element along with its row and

column index are stored. Other popular methods are the compressed sparse row (CSR) format and the compressed sparse column (CSC) format, which store the row and column indices more efficiently, making them more efficient for row-oriented and column-oriented operations respectively.

3

Literature Review

3.1. Conventional Methods for SCOPF

Various methods exist to handle the combinatorial complexity associated with N-k SCOPFs. Relying on the fact that only a subset of contingencies account for the majority of faults, these methods have the same end goal in common; reduce the problem size to enable an off-the-shelf solver to solve it. Three conventional approaches are commonly used; Benders decomposition, column-and-constraint generation algorithm and iterative contingency screening [44].

3.1.1. Benders Decomposition

In [6, 7], Benders decomposition (BD) [8] is used to decompose the problem into a master problem and subproblems representing contingency cases. In the context of the SCOPF, the master problem represents the OPF in the pre-contingency state, while the subproblems check for infeasibility of post-contingency states. If a subproblem identifies a violation of constraints, a Benders cut is added to the master problem, restricting the feasible region of the master problem. This iterative solving of the master problem and subproblems is done until no more violations are found. A schematic of a BD workflow is shown in Figure 3.1. A Benders cut is a linear inequality constraint derived from the solution of the subproblem. The computational time required for solving BD depends on the size of the master problem and the number of subproblems that have to be solved. As hinted before, BD can only be used for convex linear optimization problems. In [6, 45], frameworks are proposed for BD schemes for power system problems. In these studies, the contingencies are checked one by one as described above, and Benders cuts are generated in case of violations.

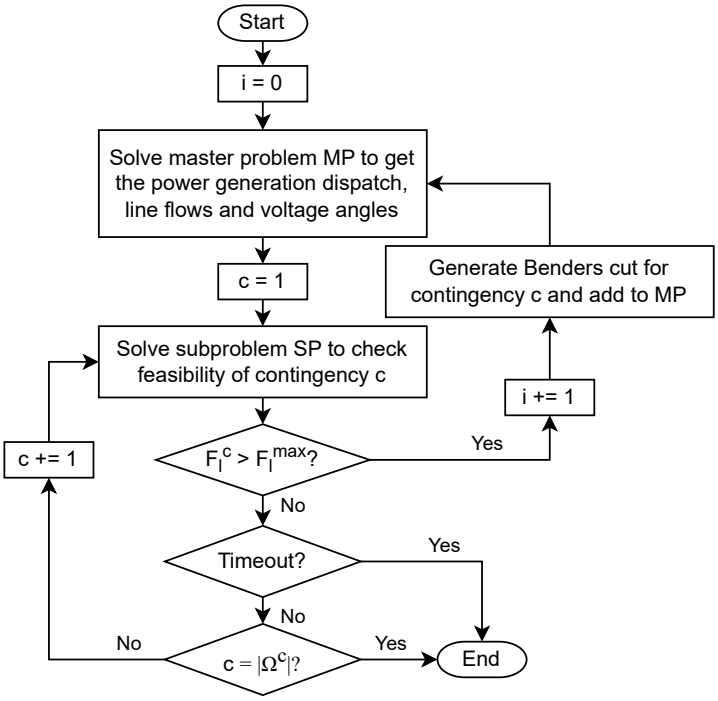


Figure 3.1: Benders decomposition workflow for SCOPF as described in [13]. The master problem is denoted with MP, the subproblems are denoted with SP.

BD has demonstrated effectiveness in solving various power system problems by decomposing them into a master problem and different subproblems, often maintaining small-sized subproblems. However, when applied to the N-k SCOPF problem which considers many contingencies, iteratively solving all subproblems is slow and can become computationally challenging for large scale optimization problems. Moreover, adding unnecessary cuts not only results in slow convergence, it can even lead to infeasible solutions. Consequently, the scalability of BD is limited when addressing N-k SCOPFs.

3.1.2. Column-and-Constraint Generation Algorithm

Furthermore, [9] employs a column-and-constraint generation algorithm (C&CGA) to iteratively add the most violating contingencies. Whereas BD is a technique specific for linear optimization problems, C&CGA can be used to incorporate constraints for optimization problems which are not necessarily linear. Robust optimization (RO) is a technique used to find solutions that are resilient to different forms of uncertainty in the problem parameters [46]. When C&CGA and RO are used in combination, the most violating contingency is identified, and only for that contingency a constraint is added to the master problem [44]. A schematic workflow of combined C&CGA with RO is shown in Figure 3.2. This constraint can be in the form of a linear Benders cut if suited, but other types of constraints can also be incorporated. This iterative process continues until no further violations occur for the worst-case contingency, ensuring a robust and feasible solution. Exemplarily, in [47] a two-stage RO method is proposed to solve N-k security constrained unit commitment.

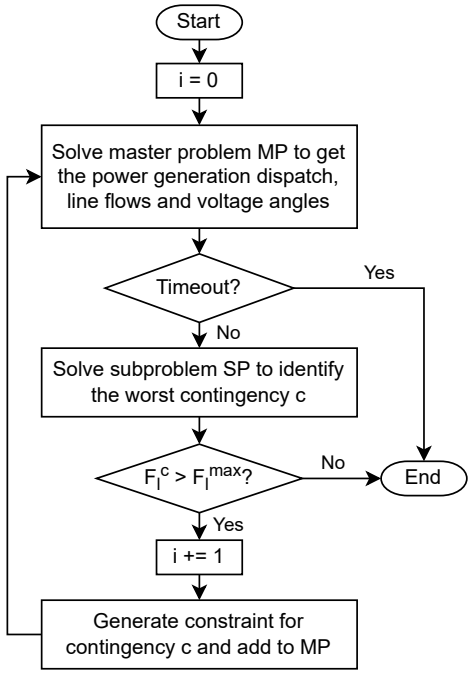


Figure 3.2: Column-and-constraint generation algorithm workflow for SCOPF as described in [13]. The master problem is denoted with MP, the subproblems are denoted with SP.

Similar to BD, a combination of C&CGA with RO can face challenges such as slow convergence when dealing with numerous contingencies, or the risk of generating infeasible predictions when too many constraints are introduced. Therefore, the C&CGA shows limited scalability for the N-k SCOPF problem.

3.1.3. Iterative Contingency Screening

Alternatively, [10] employs an iterative contingency screening method to identify the most critical contingencies, allowing the SCOPF to be solved with a selected subset of contingencies. In [11, 12], this screening process leverages LODFs due to their computational efficiency in managing multiple contingencies. A schematic workflow is shown in Figure 3.3. This is also the workflow of the baseline utilized in this thesis. In [23] a method is described using performance index sensitivities based on decoupled power flow. Reference [48] proposes another contingency screening method to effectively reduce the number of contingencies.

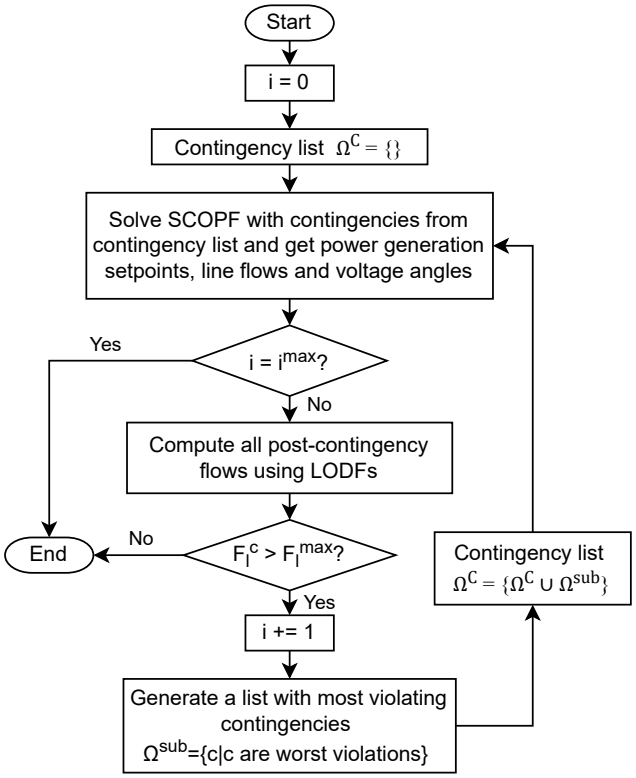


Figure 3.3: Iterative contingency screening workflow as used as baseline in this thesis. Ω^{sub} is the subset of the most violating contingencies which are added at every iteration.

The proposed approaches in [6, 7, 9, 10, 11, 12, 45, 47] struggle to scale efficiently with the number of outages k, as demonstrated in [13]. A comparison between BD, a combination of C&CGA with RO, and LODFs reveals issues such as timeouts and slow convergence for N-3. This is particularly limiting for addressing the N-k SCOPF problem that requires multiple daily resolutions.

3.2. Machine Learning for SCOPF

The recurrent nature of the N-k SCOPF problem has prompted exploration of ML techniques as a means to approximate its solution. The next section delves into proposed ML approaches for SCOPFs.

3.2.1. Determining Active Constraints

Despite the rapid growth in the total number of constraints with system size, the number of relevant active sets is considerably smaller [49]. Consequently, there is a growing interest in leveraging ML to identify these active constraint sets. The objective of the ML agent is to determine the set of active constraints. Once the agent has identified the binding constraints, a traditional solver is employed to solve the optimization problem.

In [15], an algorithm is developed to establish a mapping between input parameters and their corresponding active set of constraints. The optimal active set is identified using an 'ensemble policy' among a set of relevant active sets. Similarly, [50] introduces a MLP as a classifier to learn the mapping between input parameters and the active set of constraints at the optimal solution. Moving on to [51], an enhancement to BD is proposed. This time a support vector machine is trained to identify valuable Benders cuts. A valuable cut is a cut which is tight at the optimal solution or a cut which significantly cuts the feasible region.

These methods offer a significant advantage in terms of reduced computational time compared to conventional decomposition algorithms like BD. However, these methods do not scale with increasing k. This is because these methods rely on training data, which becomes exceedingly difficult to generate

for k >1. Additionally, once a model is trained for a specific system, its applicability is confined to systems of the same size and with the same number of constraints, further limiting scalability with k.

3.2.2. Iterative Constraint Incorporation

To address these limitations, [16] proposes a C&CGA-ML algorithm in combination with RO, iteratively adding constraints. This approach successfully approximates N-1 SCOPFs with notable reductions in computational times and minimal optimality gaps. The approach in [16] uses a Lagrangian dual method to penalize violating constraints. In instances of infeasible approximations, a feasibility restoration step is employed. Additionally, in [17], an adversarially robust ML approach is introduced, using a min-max strategy where the worst-case contingency is computed and incorporated in the SCOPF. The approach in [17] successfully reduces violating post-contingency cases up to N-3 for large test systems.

These methods demonstrate efficacy in risk reduction by selectively incorporating the most critical constraints. Nevertheless, the success of [17] is highly reliant on the ability to identify the worst-case contingency, and no guarantee is given for the strength of the attack. Futhermore, while the method in [16] successfully approximates N-1 SCOPFs, scalability becomes a challenge as k increases.

3.2.3. Alternative Approaches for OPF

References [14, 29] use MLPs to predict generator setpoints from power system loads for a DCOPF and N-1 SC-DCOPF, respectively. Similarily, [31] employs a MLP to learn the mapping between loads and generator setpoints as well as voltages for an ACOPF. Eventually, the power flow equations are solved to ensure overall ACOPF feasibility. While these methods demonstrate effectiveness, their dependency on labeled training data imposes limitations on scalability as k increases.

Other methods try to decrease training time, improve accuracy and minimise worst-case violations by including the physics of the system during training [30, 52]. In [52], a MLP is trained using a loss function which contains the discrepancies from zero of the KKT conditions. If the MLP outputs the optimal solution, these discrepancies will be zero. As the constraints are incorporated in the loss function, these methods also show limited scalability with increasing k.

To address the issue of solution feasibility, recent attention within various engineering disciplines has focused on the integration of constrained optimization and ML techniques as discussed in Section 2.3 [18, 19]. However, these methods also struggle in finding secure and feasible solutions while maintaining scalability with increasing k. Therefore, there is a pressing need to develop novel, practical methodologies that can effectively tackle the N-k SCOPF problem.

3.2.4. Graph Neural Networks for OPF

As previously discussed, power systems are models with graph structured data. Because GNNs can capture inherent structural information and dependencies from graph structured data, they are suited for various power systems related problems. Also for the OPF problem different methods have been introduced using GNNs. In [53] a graph convolutional neural network (GCNN) is proposed to approximate an OPF. In [54] a encoder-decoder architecture is proposed for GNNs to account for the high-dimensional nature of the input data. These methods show promise for the employement of GNNs to approximate OPF problems, but to the best of our knowledge, no research on N-k SCOPFs using GNNs has been performed yet.

3.3. Power Systems Operational Reliability and Resilience

3.3.1. Operational Reliability and Resiliency

The entire population must have access to electricity, which is provided by the electrical power grid, a pillar of contemporary society. As a result, it is crucial that the electricity system runs safely, ensuring reliability while minimizing costs. Security, or operational reliability, can be defined as the ability of the grid to withstand disturbances i.e. contingencies without interrupting the electricity supply [1]. Achieving 100% security in grid operations is both impractical and financially taxing. A widely accepted trade-off is the N-1 security criterion, signifying the power systems capability to withstand any single component outage. Failing to meet this criterion can result in more outages, possibly leading to blackouts. Resiliency can be defined as the power systems ability to mitigate and recover after extremely

damaging events. A distinction can be made between short-term resiliency and long-term resiliency [2]. Short-term resilience encompasses the attributes that the grid must possess before, during, and immediately after an event. Contrarily, long-term resiliency is primarily concerned with post-blackout learning. As high impact low probability (HILP) events are becoming more and more likely due to climate change [2], the need for a resilient power system is increasing. Table 3.1 presents the main differences between reliability and resilience.

Table 3.1: Main differences between reliability and resilience [1]

Criterion	Reliability	Resilience
Events considered	N-k (Mainly N-1)	N-k
Event probability	High	Very low
Event impact	Low/medium/high	High
Objective	Minimize objective cost	Minimize unserved energy

3.3.2. Power System Blackouts

A blackout is an unplanned complete interruption of power to a certain area of the grid for an unknown period of time. Two causes for blackouts can be typically distinguished [1].

1. Cascading equipment failures leading to complete blackouts represent the primary category and constitute a major cause of blackouts. If the system is not N-1 secure, a single component failure exerts substantial stress on the system, rendering equipment protection ineffective in mitigating the resultant overload. This, in turn, triggers a rapid sequence of component trips, leading to a cascade of failures.
2. The second category encompasses simultaneous failures of k components during a short period of time, the common cause often being natural disasters or extreme weather events. Such extreme weather events are HILP events.

Blackouts falling within the first category are typically associated with power systems security. The fulfillment of the N-1 security criterion, coupled with adequate contingency handling, would have averted such blackouts. The blackouts in the second category pertain to power systems resilience. The events happening in this category are low probability but have a high impact.

3.3.3. Probabilistic Reliability Indicators

Different metrics are used to assess power systems reliability and resilience. Loss of load expectation (LOLE) and expected energy not supplied (EENS) are two of those reliability indicators [2, 5, 55]. The LOLE represents the expected amount of time per period that the demand cannot be fully supplied. Moreover, EENS quantifies the amount of energy that is expected not to be supplied during a given time period, serving as a quantitative addition to LOLE [56]. To evaluate a system's reliability, different simulations are performed for different scenarios, and the values for these indices are computed. Additionally, these reliability indices can be evaluated during a HILP event. Such assessments can enhance understanding of short-term resilience to extreme weather events [2], and these indices reveal the impact of the HILP event in the absence of any recovery actions [57].

3.3.4. Correlated Component Failures

Because large cascades are rare but have a huge impact, it is important to quantify the risk of their occurrence well. A common assumption is that k component outages are statistically independent. However, different types of correlation exist between outages. According to [58] multiple line outages often occur in motifs, and contingency motifs of two lines with a common bus occur much more often. Moreover, in highly loaded systems, failures in nearby detection devices are more likely due to large transients [59]. Similarly, cascading branch failures with a common cause like extreme weather events are likely to be spatially correlated [21, 22]. In [21, 22], a copula analysis is introduced to quantify the risk while taking into account the correlation between cascading failures. A copula is a statistical concept used to describe the dependence between multiple random variables. It couples the individual (marginal) probability distributions to the joint probability distribution based on the dependent structure.

4

Methodology

A constraint-driven approach is presented considering all contingencies. The proposed approach utilizes the linearized DCOPF, ensuring a convex solution space and enabling the utilization of linear sensitivity factors. To transform it into a SCOPF, only the post-contingency flows need to be considered as additional constraints. The following will be discussed in this chapter. Section 4.1 discusses two methods for computing all post-contingency flows. Section 4.2 sequentially explains the proposed method. Finally, Section 4.3 describes the method for the risk-based N-k SCOPF.

4.1. Post-Contingency Flows

In this section, the methods of using PTDFs and LODFs are compared for computing all post-contingency flows. Key factors in this comparison encompass the computational time required for constructing all sensitivity factors, the degree of compression achieved by both methods (indicating a decrease in the number of elements stored), and their potential practical applicability within the proposed approach.

The PTDF method involves the computation of the post-contingency flows utilizing a contingency specific PTDF matrix, denoted as $\mathbf{PTDF_c}$. This matrix is obtained by removing the rows and columns corresponding to the line that is out of service from the susceptance matrix $\mathbf{B}$ to form $\mathbf{B_c}$. The inverse of this modified matrix, denoted as $\mathbf{X_c}$, is employed to calculate $\mathbf{PTDF_c}$ using Eq. (2.4). Due to the contingency-specific nature of $\mathbf{B_c}$, inverting it for each contingency poses a computational challenge. Consequently, all individual $\mathbf{PTDF_c}$ matrices are stacked to form a 3D tensor termed full $\mathbf{PTDF_c}$ ($\mathbf{FPTDF_c}$). To reduce memory requirements, TTD is applied to iteratively construct the $\mathbf{FPTDF_c}$ tensor, as constructing it entirely and then applying TTD is not feasible.

In contrast, the LODF method computes the post-contingency flows through the use of LODFs. These LODFs are structured into a matrix, facilitating efficient computation of all post-contingency flows through a matrix multiplication and a summation. The sparsity of the LODF matrix makes it well suited for sparse matrix operations, thereby enhancing computational efficiency. Additionally, the smaller size of the matrix to be inverted (k $\times$ k), results in a notably faster computation of sensitivity factors.

A comparative analysis between PTDFs and LODFs for computing all post-contingency flows reveals several key findings:

1. Both methods produce the same post-contingency flows, indicating equivalent accuracy in post-contingency flow computation.
2. While PTDFs can be efficiently represented in TT-format for smaller systems, scalability becomes a challenge for larger systems, necessitating iterative construction and posing computational bottlenecks. In contrast, LODFs exhibit faster construction times and greater compression for both small and large systems.
3. The construction of the loss function using the LODF method is a straightforward process. In contrast, no practical and manageable method to construct the loss function in TT-format has been found.

Ultimately, the LODF method is chosen for the final framework due to its computational efficiency, scalability, and practical advantages. A more extensive description of both methods, including numerical results, is given in Appendix C.

4.2. Proposed Constraint-Driven Learning Approach

The following section introduces a schematic visualization detailing the implementation workflow of the approach. Initially, a brief outline of the approach at a high level will be provided. Following the visual representation in Figure 4.1, a comprehensive, sequential explanation of the entire workflow will develop.

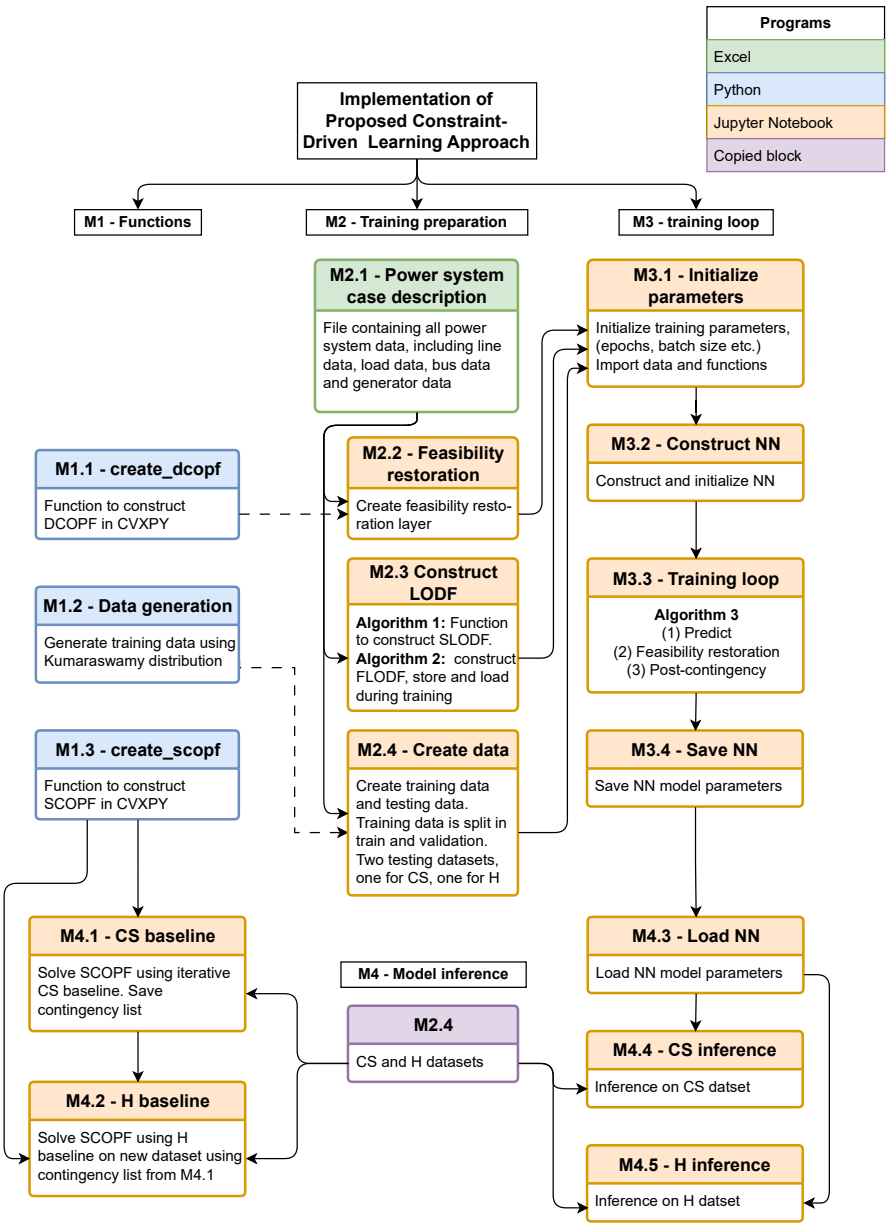


Figure 4.1: Full workflow of the implementation of the method.

The proposed approach uses ML to approximate the solution of a N-k SCOPF. As Figure 4.2 shows, a MLP maps the loads to generator setpoints ($\widehat{P}_G$). When the MLP prediction lies beyond the feasibility boundaries constrained by the DCOPF equations, a restoration layer is employed to realign it within the feasible region. The convex DCOPF enables the utilization of a CVXPYlayer [34], a convex solver

that computes the gradients through the solution of the optimization. The restoration layer minimizes the L2 norm between the predicted setpoints and the setpoints adhering to the feasible region. The restoration layer outputs the restored generator setpoints, along with other variables in the optimization problem, including phase angles ($\boldsymbol{\delta}$) and line flows ($\mathbf{F}$). The formulation of the optimization is outlined as follows:

$$\min_{n \in \Omega^G} \sum ||P_{G_n} - \widehat{P}_{G_n}||_2 \tag{4.1a}$$

subject to:

$$\mathbf{B} \cdot \boldsymbol{\delta} = \mathbf{P_G} - \mathbf{P_D} \tag{4.1b}$$

$$F_l^{min} \leq \frac{1}{x_{ij}}(\delta_i - \delta_j) \leq F_l^{max} \quad \forall i,j \in \Omega^B, \forall l \in \Omega^L \tag{4.1c}$$

$$P_{G_n}^{min} \leq P_{G_n} \leq P_{G_n}^{max} \quad \forall n \in \Omega^G \tag{4.1d}$$

To ensure the MLP adheres to the generator limits, the MLP outputs a scaling factor $\alpha_n \in [0,1]$, and $\widehat{P}_{G_n}$ is computed as $\widehat{P}_{G_n} = \alpha_n \cdot (P_{G_n}^{max} - P_{G_n}^{min}) + P_{G_n}^{min}$.

The set of all post-contingency flows for up to k outages is computed using LODFs.

$$F_l^c = F_l^0 + LODF_{N-k} \times F_l^0 \quad \forall l \in \Omega^L \tag{4.2}$$

The loss function considers the dispatch cost, the predicted base case flow violations, the predicted generation-demand imbalance and the post-contingency flow violations as below:

$$\begin{aligned} Loss = \lambda_0 \mathbf{c_G}^T \mathbf{P_G} + \lambda_1 \|ReLU(|\widehat{\mathbf{F^0}}| - \mathbf{F^{max}})\|_1 + \lambda_2 \|ReLU(|\mathbf{F^c}| - \mathbf{F^{max}})\|_1 \\ + \lambda_3 \|\Sigma \widehat{\mathbf{P}}_\mathbf{G} - \Sigma \mathbf{P_D}\|_1 \end{aligned} \tag{4.3}$$

Here, $\mathbf{c_G}$ is the vector of generator cost. The predicted base case flows are computed using power transfer distribution factors (PTDFs) as:

$$\widehat{\mathbf{F^0}} = \mathbf{PTDF}(\widehat{\mathbf{P}}_\mathbf{G} - \mathbf{P_D}) \tag{4.4}$$

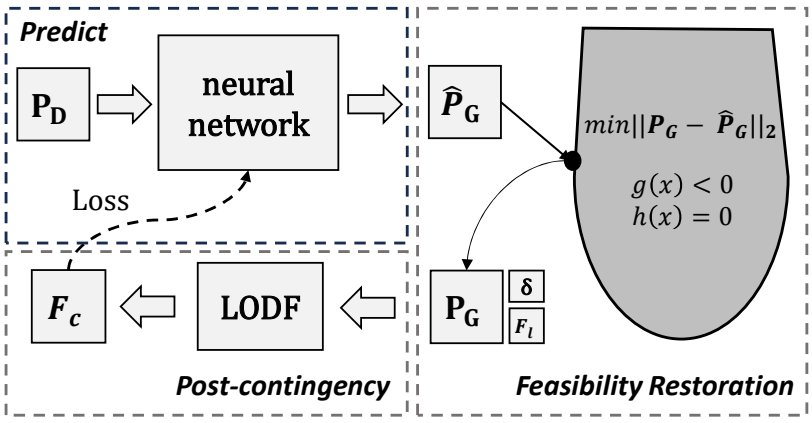


Figure 4.2: The workflow of the proposed constraint-driven approach. A MLP maps the loads to generator setpoints. A feasibility restoration layer restores infeasible setpoints. Using LODFs, all post-contingency flows are obtained.

4.2.1. Fundamental Functions

The proposed approach employs three Python-based functions. These functions are instrumental in constructing the feasibility restoration layer, generating training and testing data, and creating baselines for comparative evaluation.

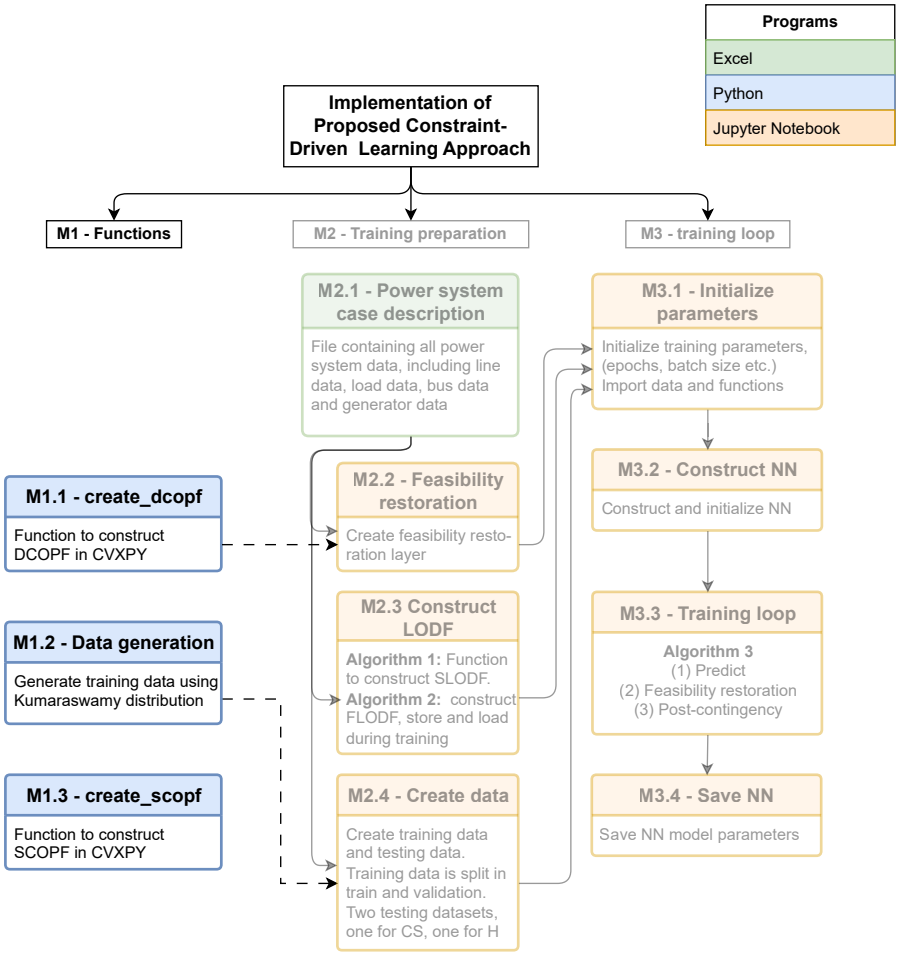


Figure 4.3: M1 - Functions; the functions created for the proposed approach.

M1.1 - Create DCOPF function Function M1.1 is implemented utilizing the CVXPY package [60], and constructs an optimization problem adhering to the DCOPF constraints, given by Eq. (4.1b), Eq. (4.1c), and Eq. (4.1d). To ensure feasibility of the optimization problem at all times, two slack terms (γ^+, γ^-) are introduced into the nodal balance equations. These slack terms are heavily penalized within the objective, driving it towards a value of zero. Non-zero values of the slack terms indicate violations of the DCOPF constraints.

Notably, its objective function deviates from the conventional DCOPF objective, focusing on the minimization of the L2 norm between the predicted setpoints and the setpoints adhering to the feasible region, as given in Eq. (4.1a). To achieve this, the predicted generator setpoints $\widehat{\mathbf{P}}_{\mathbf{G}}$ are introduced as parameters to the optimization, with the restored generator setpoints $\mathbf{P}_{\mathbf{G}}$ are treated as variables. The function takes as input the dataset constructed in M2.1, and generates essential data for the construction of the feasibility restoration layer. Function M1.1 outputs the optimization problem, a set of parameters ($\mathbf{P}_{\mathbf{D}}$, α) and a set of variables ($\mathbf{P}_{\mathbf{G}}$, $\mathbf{F}$, δ, γ^+, γ^-). Consequently, this function forms the core element for the construction of the feasibility restoration layer in M2.2.

M1.2 - Data generation function The second function, M1.2, is designed to generate data samples utilizing a Kumaraswamy distribution. It samples load data around the nominal loads, taking as input the

Pearson correlation coefficient, lower and upper bounds for load sampling, distribution shape parameters, and the desired number of samples. The function produces a matrix containing scaling factors of size $\mathbb{R}^{|\Omega^D|\times|\Omega^S|}$. By multiplying this matrix with the nominal load profile, the function generates the complete dataset of power system loads.

M1.3 - Create SCOPF function Finally, function M1.3, constructed with CVXPY, is similar to M1.1, but also incorporates the line flow limits in contingency cases. This function serves to establish baseline results for comparative analysis with the proposed approach. It shares the conventional objective of a SCOPF, to minimize dispatch cost, while also incorporating the heavily penalized slack variables.

4.2.2. Fundamental Component Construction

The following phase involves the construction of fundamental components crucial to the proposed approach. Initially, an Excel file, denoted as *M2.1 - Power system case description*, is created for describing the specifications of the test system data. This file includes required data pertaining to buses, lines, generators, and loads.

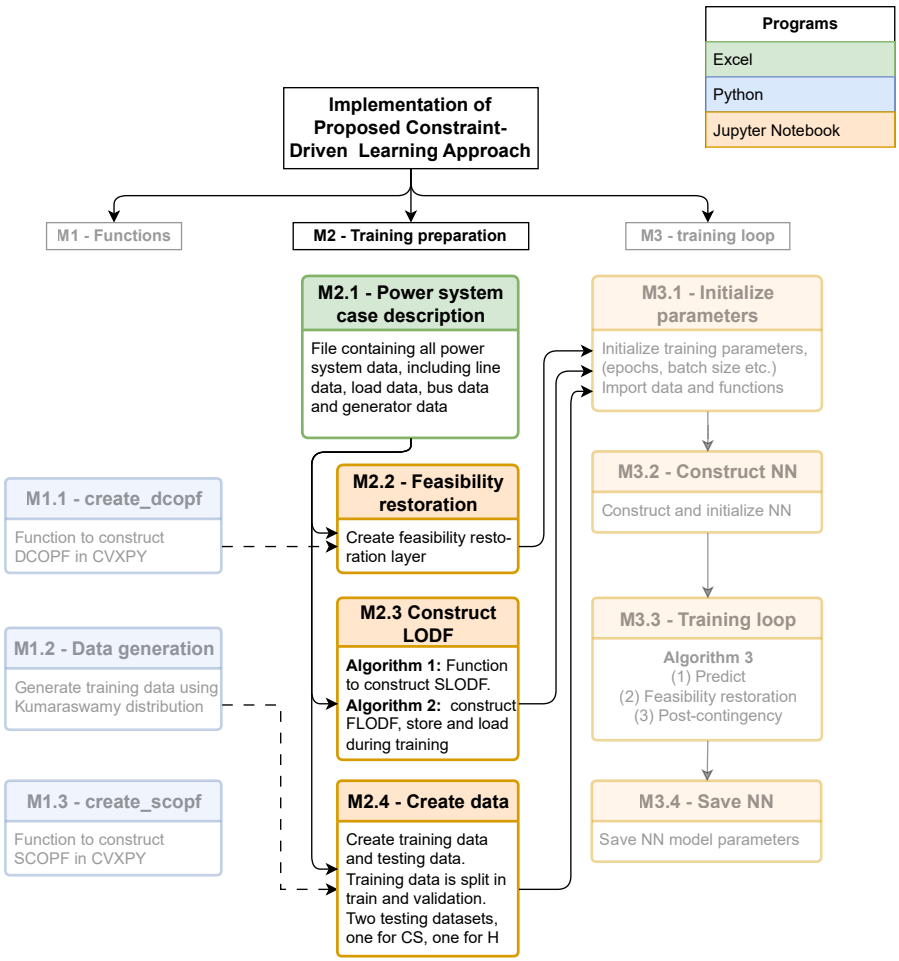


Figure 4.4: M2 - Training preparation; preparing the data and building blocks for the proposed approach.

M2.2 - Feasibility Restoration Layer

The feasibility restoration layer is a function which takes as input the predicted generator setpoints, and outputs the restored generator setpoints, along with other variables in the optimization problem, including phase angles (δ) and line flows ($\mathbf{F}$). The function is constructed using the Python package CVXPYlayer [34]. This feasibility restoration layer is initialized with the problem specification derived

from the function created in M1.1. The feasibility restoration layer is integrated in the proposed approach, solving the optimization during the forward pass, and computing the gradients through the solution of the optimization in the backward pass. This enables backpropagation of the loss.

M2.3 - Line Outage Distribution Factors
To compute all post-contingency flows, LODFs are used. The LODFs are computed using the method of generalized line outage distribution factors as explained in Section 2.1.3.

$$\mathbf{LODF}_{\mathbf{M,O}} = \mathbf{PTDF}_{\mathbf{M,O}}(\mathbf{E} - \mathbf{PTDF}_{\mathbf{M,O}})^{-1} \tag{4.5}$$

For each contingency case, a square single LODF ($\mathbf{SLODF}$) matrix of size $\mathbb{R}^{|\Omega^L|\times|\Omega^L|}$ is constructed. Algorithm 1 is the algorithm employed to construct each $\mathbf{SLODF}$ matrix. In lines 1-2, the algorithm takes as input the power system case description and the number of simultaneous outages, k. Line 3 determines the set of all possible contingencies. Line 5 determines the indices of the lines in outage for the considered contingency case and line 6 initializes a square matrix of zeros of size $\mathbb{R}^{|\Omega^L|\times|\Omega^L|}$. Lines 7-12 compute the LODFs for the lines in outage. In case of singularity of the matrix $(\mathbf{E}-\mathbf{PTDF}^{\mathbf{0}}_{\mathbf{O,O}})^{-1}$, a part of the network gets disconnected, and the LODFs are set to zero. Finally, in line 13 the LODFs are added to the $\mathbf{SLODF}$ matrix at the indices of the lines in outage. Line 14 transforms the $\mathbf{SLODF}$ matrix in sparse COO format. For this thesis, the PyTorch function `.to_sparse_coo()` is utilized.

Algorithm 1 Function to create $\mathbf{SLODF}$

1: **Import:** data ← M2.1 - power system case description
2: **Input:** data, k
3: Obtain all contingency combinations Ω^C ← data, k (Eq. Eq. (4.6))
4: **function** $createSLODF(i)$
5: Line indices outaged lines c_1,c_2...c_k ← $\Omega^C(i)$
6: $\mathbf{SLODF}$ = $zeros(l, l)$
7: $\mathbf{PTDF}^{\mathbf{0}}_{\mathbf{M,O}}$, $\mathbf{PTDF}^{\mathbf{0}}_{\mathbf{O,O}}$ ← $\mathbf{PTDF}$
8: **if** Determinant($(\mathbf{E} - \mathbf{PTDF}^{\mathbf{0}}_{\mathbf{O,O}})^{-1}$) > 0: **then**
9: $\mathbf{LODF}_{\mathbf{M,O}}$ = $\mathbf{PTDF}^{\mathbf{0}}_{\mathbf{M,O}}(\mathbf{E} - \mathbf{PTDF}^{\mathbf{0}}_{\mathbf{O,O}})^{-1}$
10: **else**
11: $\mathbf{LODF}_{\mathbf{M,O}}$ = 0
12: **end if**
13: $\mathbf{SLODF}[:, c_1 : c_k]$ = $\mathbf{LODF}_{\mathbf{M,O}}$
14: **return** $\mathbf{SLODF}$ ← make sparse
15: **end function**

As Figure 4.5 shows, these $\mathbf{SLODF}$ matrices are then vertically stacked to form a larger rectangular matrix, which is called the full LODF ($\mathbf{FLODF}$) matrix.

Algorithm 2 outlines the procedure for computing the $\mathbf{FLODF}$ matrix. Initially, in line 3, we determine the set of all possible contingencies Ω^C. In situations involving a substantial number of contingencies, performing a single matrix multiplication with the $\mathbf{FLODF}$ matrix to compute all post-contingency flows results in memory issues. In such cases, the $\mathbf{FLODF}$ matrix is built in multiple batches, with post-contingency flows computed batch-wise. Each batch comprises a specified number of $\mathbf{SLODF}$ matrices. In cases where a single matrix multiplication with $\mathbf{FLODF}$ is possible, the batch size equals $|\Omega^C|$. Lines 4-6 initialize the construction. Lines 8-17 create a $\mathbf{SLODF}$ matrix $\forall c \in \Omega^C$, and stack them to $\mathbf{FLODF}_{\mathbf{batch}}$. If the $\mathbf{FLODF}_{\mathbf{batch}}$ contains the specified amount of $\mathbf{SLODF}$s, it is appended as a submatrix to $\mathbf{FLODF}$. In line 18, any remaining $\mathbf{SLODF}$s are appended to the $\mathbf{FLODF}$.

After the $\mathbf{FLODF}$ matrix is constructed, a module called *pickle* is used for storage (line 20) and subsequent retrieval of the $\mathbf{FLODF}$ within the training loop.

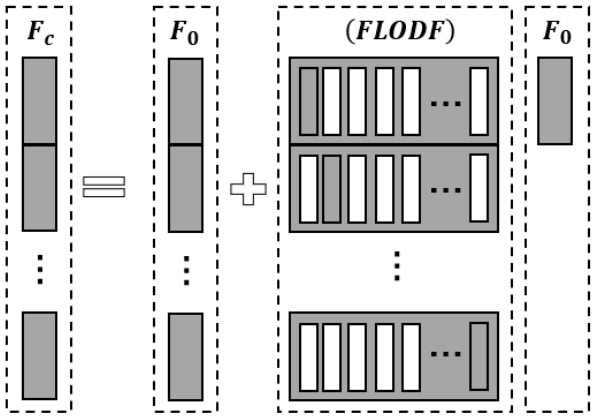


Figure 4.5: Schematic representation of the computation of all post-contingency flows with a single matrix multiplication. The $\mathbf{FLODF}$ is created by stacked $\mathbf{SLODF}$s for a N-1 case. The white columns are zero columns, the grey columns contain the LODFs.

Algorithm 2 Construction of the sparse $\mathbf{FLODF}$ matrix

Import: pickle
Import: data ← M2.1 - power system case description
Input: data, k
Obtain all contingency combinations Ω^C ← data, k
lodf batch = batch size # specified number of $\mathbf{SLODF}s$ per $\mathbf{FLODF_{batch}}$
$\mathbf{FLODF}$ = [ ]
$\mathbf{FLODF_{batch}}$ = ∅
counter = 0
for i in $range(|\Omega^C|)$ **do**
    $\mathbf{SLODF}$ = $createSLODF(i)$
    stack $\mathbf{SLODF}$ to $\mathbf{FLODF_{batch}}$
    counter += 1
    if counter == lodf batch: **then**
        $\mathbf{FLODF}$.append($\mathbf{FLODF_{batch}}$)
        $\mathbf{FLODF_{batch}}$ = ∅
        counter = 0
    end if
end for
if counter $>$ 0: **then** $\mathbf{FLODF}$.append($FLODF_{batch}$)
end if
pickle.dump($\mathbf{FLODF}$)

Memory Reduction with Sparsity FLODF As visualized in Figure 4.5, the construction of the $\mathbf{FLODF}$ matrix is designed to enable a single matrix multiplication to obtain all post-contingency flows. As the number of columns containing non-zero elements equals the number of outages k, this matrix is sparse. The number of contingency cases $|\Omega^C|$ holds combinatorial complexity and can be computed as:

$$\Omega^C(l,k) = \frac{|\Omega^L|!}{k!(|\Omega^L| - k)!} \tag{4.6}$$

Each $\mathbf{SLODF}$ contains k non-zero columns. Each column has $|\Omega^L|$ entries, with ($|\Omega^L|$ - k) possible non-zero entries. The maximum possible number of non-zero elements in a $\mathbf{SLODF}$ is $k(|\Omega^L| - k)$. The maximum possible amount of nonzero elements for the $\mathbf{FLODF}$ is $k|\Omega^C|(|\Omega^L| - k)$. The number

of elements in the $\mathbf{FLODF}$ is $|\Omega^C||\Omega^L|^2$. A lower bound of sparsity can be computed for the $\mathbf{FLODF}$ matrix:

$$sparsity = 1 - \frac{k(|\Omega^L| - k)}{|\Omega^L|^2} \tag{4.7}$$

This is a theoretical lower bound since for certain cases the LODF is zero due to the network topology, and in some contingency cases a part of the network gets disconnected. As previously mentioned, in the context of islanding cases, the LODFs are intentionally set to zero, and islanding is recognized by the singularity of the matrix $(\mathbf{E} - \mathbf{PTDF}^{\mathbf{0}}_{\mathbf{O,O}})^{-1}$. As k increases, more islanding cases occur, especially for smaller systems. To reduce the memory of the $\mathbf{FLODF}$ matrix, the matrix is expressed in sparse COO format, storing only the non-zero elements along with their row and column indices. Consequently, a sparse matrix multiplication is performed between the $\mathbf{FLODF}$ and the base case flows.

M2.4 -Data Generation
After the construction of the feasibility restoration layer and the $\mathbf{FLODF}$ matrix, the next step involves the generation of training and testing datasets, facilitated by function M1.2. Once generated, these datasets are stored using the pickle module for later retrieval during training of the proposed approach.

4.2.3. Constraint-Driven Learning

With the construction of the essential building blocks, the proposed approach is now prepared for training. The initial step, M3.1, specifies all training parameters and imports the fundamental components.

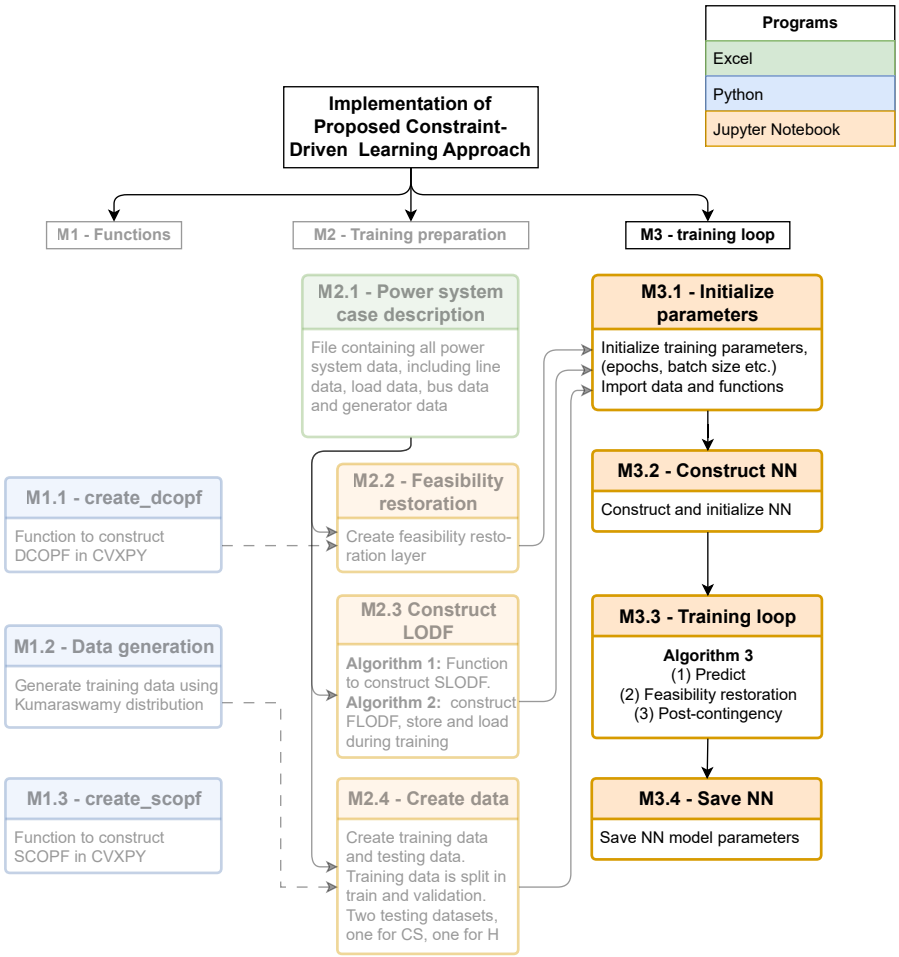


Figure 4.6: M3 - Training loop; steps taken during training of the proposed approach.

M3.2 - Neural Network Architecture

We employ a fully connected linear neural network, or MLP. Notably, as the dimensions of the input and output layer scale with the number of loads and generators within the system respectively, the number of trainable parameters increases with the system's size. Consequently, when scaling to larger test systems, a larger amount of training data is required to adequately train all these parameters, resulting in limited scalability of the MLP. To address the challenge of scalability, we explore the use of a GCNN, which parameter count is theoretically independent of the system's size. This property enhances the scalability of the proposed approach for larger grids.

To guide the GCNN in finding optimal solutions with respect to dispatch cost, we adopt a preprocessing step. Prior to passing the hidden features from the last convolutional layer to the readout layer, we concatenate a L2-normalized cost vector to the hidden feature matrix. This approach draws inspiration from the work presented in [54]. GCNNs are unique in that they share parameters and simultaneously learn features for all nodes. Intuitively, the inclusion of the cost introduces a form of prioritization, favoring the selection of more cost-effective generators.

M3.3 - Training Loop

(1) Predict As Figure 4.7 shows, the first step involves the prediction of the generator setpoints from the loads using a MLP. To ensure the MLP adheres to the generator limits, the MLP outputs a scaling factor $\alpha_n \in [0, 1]$. Both the sigmoid and scaled tanh activation functions possess the property of predicting a number between 0 and 1. Through the employement of one of these activation functions in the output of the MLP, the generator constraints are enforced. Subsequently, $\widehat{P}_{G_n}$ is computed as $\widehat{P}_{G_n} = \alpha_n \cdot (P_{G_n}^{max} - P_{G_n}^{min}) + P_{G_n}^{min}$.

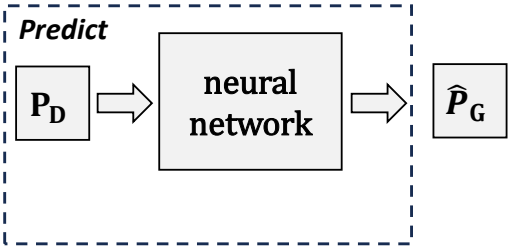


Figure 4.7: Schematic representation of prediction step.

(2) Feasibility Restoration After predicting the generator setpoints, as shown in Figure 4.8, these setpoints are forwarded to the previously constructed feasibility restoration layer from M2.2. The restoration outputs the restored generator setpoints together with the other variables in the optimization problem (i.e. phase angles $\boldsymbol{\delta}$ and line flows $\mathbf{F}$).

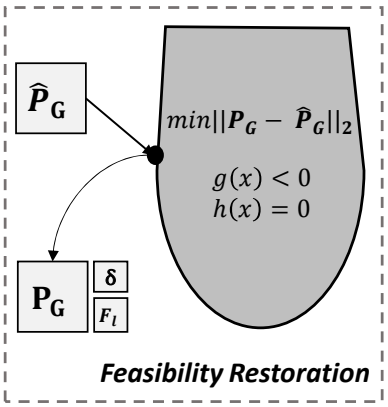


Figure 4.8: Schematic representation of feasibility restoration step.

(3) Post-Contingency Lastly, as visualized in Figure 4.9, using the $\mathbf{FLODF}$ matrix, all post-contingency flows are computed. For improved readability, Eq. (4.2) for computing the post-contingency flows is repeated below:

$$F_l^c = F_l^0 + LODF_{N-k} \times F_l^0 \quad \forall l \in \Omega^L \tag{4.8}$$

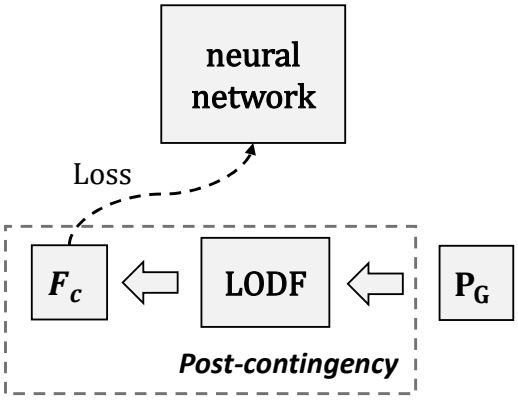


Figure 4.9: Schematic representation of post-contingency step.

The loss function utilized for training the entire approach is based on the penalty function method detailed in Section 2.3.2. This loss function considers violations of both DCOPF constraints and line flow limits in post-contingency cases. Since the feasibility restoration layer follows the MLP, the MLP remains agnostic to this restoration process. Consequently, the loss function incorporates violations of the DCOPF constraints related to the predicted generator setpoints $\widehat{\mathbf{P}}_\mathbf{G}$. To achieve this, the predicted base-case line flows are computed using Eq. (4.4). The complete loss function is formally defined in Eq. (4.3).

Computational Graph Memory Reduction An essential aspect of the training process involves reducing the memory requirements associated with the computational graph. A computational graph is a directed graph that visually represents a mathematical or computational model, illustrating the data flow and interconnection of various operations. Computational graphs find significant utility in deep learning, where they guide the gradient flow for backpropagation in tensors, as illustrated in Figure 4.10. However, when dealing with extensive tensors, such as those involving large $\mathbf{FLODF}$ matrices and multiple samples, substantial memory is required. To reduce memory requirements, Algorithm 3 is presented. Lines 7-8 in Algorithm 3 compute the post-contingency flows without the construction of the computational graph (requires grad = False). Lines 9-15 in Algorithm 3 remove rows corresponding to non-violating flows from the $\mathbf{FLODF}$ matrix and recompute the post-contingency flows using the computational graph (requires grad = True), constructing only a relatively small computational graph, therefore reducing memory requirements. A visualization of the reduced $\mathbf{FLODF}$ is given in Figure A.1.

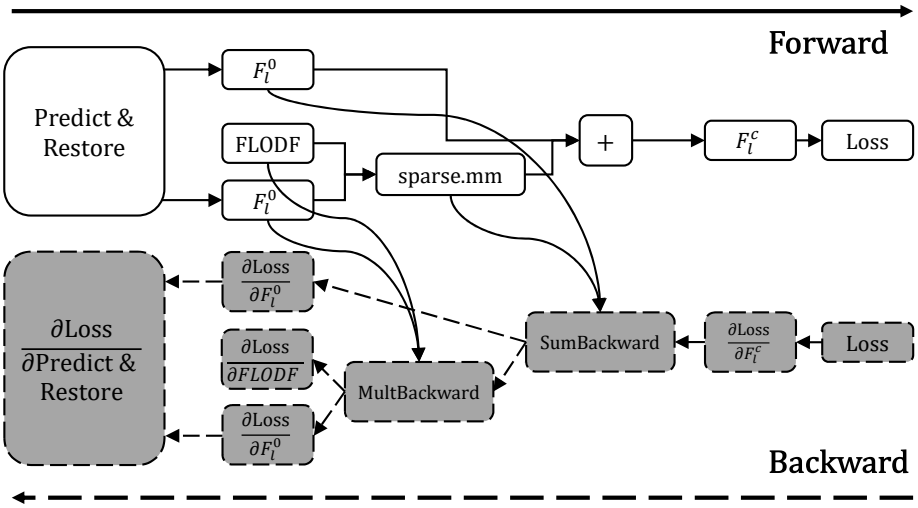


Figure 4.10: The computational graph of the proposed approach.

Algorithm 3 Reducing the computational graph

for $epoch$ in Epochs **do**
  MLP: $\boldsymbol{\alpha_n} \leftarrow \mathbf{P_D}$
  function CVXPYlayer($\mathbf{P_D}$, $\boldsymbol{\alpha_n}$)
    Input: $\mathbf{P_D}$, $\boldsymbol{\alpha_n}$
    return $\mathbf{P_G}$, $\mathbf{F^0}$, $\boldsymbol{\delta}$
  end function
  requires grad = False
  $\mathbf{F^c} \leftarrow \mathbf{F^0} + \mathbf{FLODF} \times \mathbf{F^0}$
  if $F_l^c < F_l^{max}$ **then**
    Obtain row indices non-violating F_l^c
    Remove non-violating rows $\mathbf{FLODF}$
    $\mathbf{FLODF_{reduced}} \leftarrow \mathbf{FLODF}$
  end if
  requires grad = True
  $\mathbf{F^c} \leftarrow \mathbf{F^0} + \mathbf{FLODF_{reduced}} \times \mathbf{F^0}$
  Loss: $\sum ReLU(|\mathbf{F^c}| - \mathbf{F^{max}})$
  Perform backward pass
end for

M3.4 - Save MLP
Finally, the MLP weights are stored for future inference and comparison with the proposed baselines.

4.3. Probabilistic Security Assessment

Expanding upon the preceding section, the workflow undergoes slight modifications to facilitate probabilistic security assessment. Incorporating probabilities into the approach enables decision-making with an N-k risk-based security criterion enhancing N-k security [1]. The goal is to achieve N-k security and heightened resilience to changing outage probabilities caused by unforeseen HILP events, thus enhancing power system operational resilience [3]. The modified sections are highlighted in Figure 4.11, and will be explained in this section.

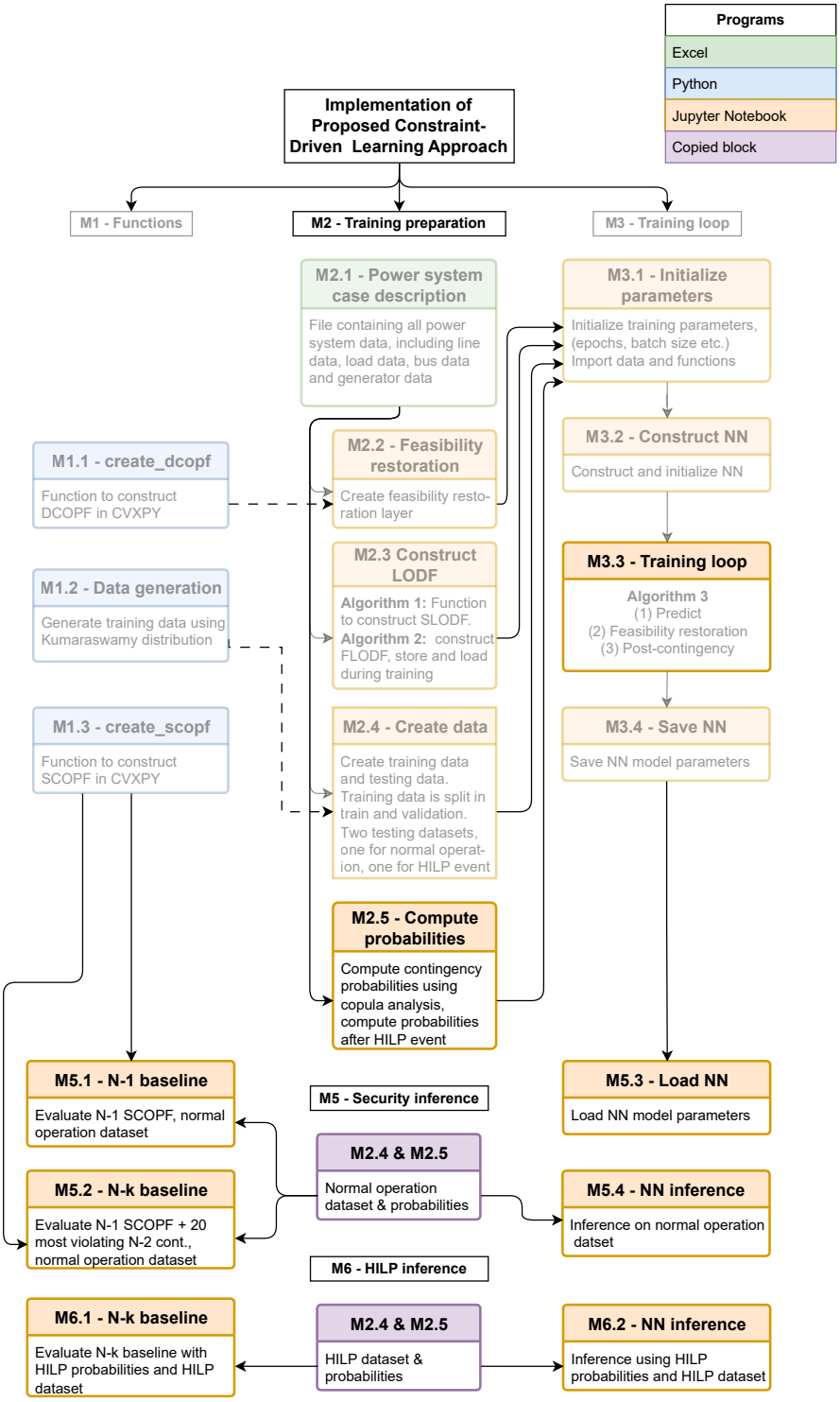


Figure 4.11: Schematic of the proposed approach; additional blocks needed for the probabilistic security assessment and the inference.

4.3.1. Joint Probabilities with Copulas

M2.5 - Compute Probabilities

Within the training preparation section M2, an additional Jupyter Notebook file is introduced designed for the computation of the contingency probabilities. To compute the joint line outage probabilities, the Gaussian copula function introduced in [21] is used. It is assumed that the inverse stress on transmis-

sion line l follows a Gaussian distribution and is given by a random variable $Y_l = N(\mu_l, \sigma_l)$. μ_l and σ_l are the mean and standard deviation respectively and the cumulative distribution function is given by:

$$\mathcal{F}_{Y_l}(y_l) = \frac{1}{2}\left[1 + erf\left(\frac{y_l - \mu_l}{\sigma_l\sqrt{2}}\right)\right] \tag{4.9}$$

For each line l, μ_l and σ_l are chosen so that $\mathcal{F}_{Y_l}(0)$ = p_l where p_l is the independent probability of line l going out. Assuming $\mu_l = 1\ \forall l$, σ_l is determined as:

$$\sigma_l = \frac{-1}{erf^{-1}(2p_l - 1)\sqrt{2}} \tag{4.10}$$

To compute the covariance between two probabilities, we assume exponential decay of correlation with distance between lines l and m:

$$\rho_{l,m} = \rho_0 e^{-d_{l,m}/L} \quad l, m \in \Omega^L \tag{4.11}$$

Here, ρ_0 represents maximal correlation at zero distance, L is the characteristic length, and $d_{l,m}$ is simplified as the shortest distance between any pair of buses of the lines ($d_{l,m}$ = $\min\{d(l_1, m_1), d(l_1, m_2), d(l_2, m_1), d(l_2, m_1)\}$) as shown in Figure 4.12. This exponential decay models spatial behavior seen in events like earthquakes, tornadoes, and hurricanes, which often exhibit geographic correlations [21, 22].

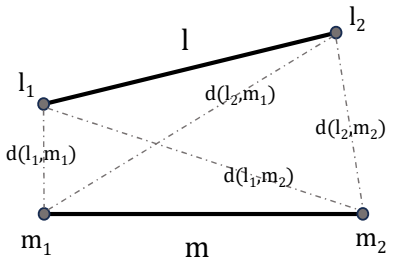


Figure 4.12: The distances between any pair of buses of two lines. The shortest distance is used, in the Figure d(l_1,m_1).

The covariance between lines is calculated as $cov(l, m) = \rho_{l,m}\sigma_l\sigma_m$. This equation allows for the construction of the covariance matrix $\mathbf{C}$. The copula is formed using the probability density function of the multivariate normal distribution given below.

$$f(y) = \frac{1}{\sqrt{(2\pi)^k|\mathbf{C}|}} exp\left(-\frac{1}{2}(\mathbf{y} - \mu)^T\mathbf{C}^{-1}(\mathbf{y} - \mu)\right) \tag{4.12}$$

The joint probability of k $= 2, 3$ outages is computed by integrating Eq. (4.12) over the corresponding region in the joint probability distribution:

$$\mathcal{F}_Y(0) = \int_{-\infty}^{0}\int_{-\infty}^{0}\cdots\int_{-\infty}^{0} f(y_1, y_2, \ldots, y_k)\, dy_1\, dy_2\, \ldots\, dy_k \tag{4.13}$$

Here, $\mathcal{F}_Y(0)$ is the joint probability of k outages. In this context, the joint probabilities for all combinations of k $= \{2, 3\}$ are computed, resulting in the formation of the vector $\mathbf{P_{N-k}}$. As a result, three distinct vectors, $\mathbf{P_{N-1}}$, $\mathbf{P_{N-2}}$, and $\mathbf{P_{N-3}}$, are created and finally stored for subsequent retrieval in the training loop.

4.3.2. Expected Violations

M3.3 - Modified Training Loop

The training loop in section M3.3 undergoes a small modification to facilitate the integration of probabilities during training. By multiplying the contingency probabilities with the post-contingency flow violations, the expected post-contingency flow violations are obtained. Using these expected violations the probability of the system being secure or not is assessed.

$$\mathbb{E}[\mathbf{F^{viol}}] = \|\mathbf{P_{N-k}} \cdot ReLU(|\mathbf{F^c}| - \mathbf{F^{max}})\|_1 \tag{4.14}$$

The loss function employed to minimize the expected post-contingency flow violations is similar to Eq. (4.3). The only modification is that the expected violations are used instead of the actual violations.

$$\begin{aligned} Loss = \lambda_0 \mathbf{c_G}^T \mathbf{P_G} + \lambda_1 \|ReLU(|\widehat{\mathbf{F^0}}| - \mathbf{F^{max}})\|_1 + \lambda_2 \|\mathbf{P_{N-k}} \cdot ReLU(|\mathbf{F^c}| - \mathbf{F^{max}})\|_1 \\ + \lambda_3 \|\Sigma \widehat{\mathbf{P}}_\mathbf{G} - \Sigma \mathbf{P_D}\|_1 \end{aligned} \tag{4.15}$$

The MLP is trained using the loss function described in Eq. (4.15), which accounts for all contingencies across multiple security levels including their probabilities.

4.3.3. Probabilistic Reliability Indicators

The probabilistic reliability indicators LOLE and EENS are evaluated during normal operation. Moreover, these indices are assessed when the proposed approach undergoes a HILP event. These indices are computed as follows [56]:

$$LOLE = \Delta T \sum_{c \in \Omega^C} P^c_{N-k}[\Sigma \mathbf{P^c_G} < \Sigma \mathbf{P^c_D}] \tag{4.16}$$

In this expression, $P^c_{N-k}[\Sigma \mathbf{P^c_G} < \Sigma \mathbf{P^c_D}]$ represents the probability of the generation capacity being smaller than the total load during contingency c. Here, ΔT denotes the total duration of the study period in hours. Additionally, the computation of EENS follows the formulation:

$$EENS = \sum_{c \in \Omega^C} (P^c_{shed} \cdot P^c_{N-k}) \tag{4.17}$$

The EENS is computed as the sum of the expected load shed during the study period for each contingency c, denoted as P^c_{shed}, multiplied by its respective probability P_{N-k}, considering all contingencies within the set Ω^C.

5

Case Studies

5.1. Case Study Settings

The deterministic approach is tested on the IEEE 39-bus and the IEEE 118-bus systems, while the probabilistic approach is solely tested on the IEEE 39-bus system [61]. The IEEE 39-bus system is modified, excluding the transformers. The test case characteristics are presented in Table 5.1. The single line diagrams are shown in Figure 5.2 and Figure 5.3 respectively.

To model the correlation pattern in power system loads, a Kumaraswamy(1.6, 2.8) distribution is used. Power levels are sampled within $\pm$ 0.25 of the nominal load with a Pearson correlation coefficient of 0.75. First, both systems are trained using a MLP with three hidden layers, dropout (0.2), ReLU activation and a scaled tanh activation in the output. For the 39-bus system, 1000 samples are used with a train-validation split of 800/200. 8 neurons per hidden layer result in 410 parameters. The flow limit for all lines is set to 1000MW. For the 118-bus system, 3200 samples are used with a train-validation split of 3000/200. 16 neurons per hidden layer result in a total of 2467 parameters. The flow limit is set to 200MW for most lines, with a higher flow limit for some critical lines. Next, both networks are trained using a GCNN. For the 39-bus case, the GCNN is constructed using three GraphConv layers, with 16, 8 and 8 hidden features respectively. One read-out layer is used, resulting in a total of 465 parameters. For the 118-bus case, the GCNN is constructed using three GraphConv layers, with 32, 16 and 16 hidden features respectively. One read-out layer is used, resulting in a total of 1689 parameters. Inference is done on a not previously seen data set of 1000 samples.

For the probabilistic study, inference is also done on a not previously seen data set of 1000 samples, representing 1000 hourly simulations. Equal outage probabilities are assumed for all lines, where $p_l = 0.005$ for $l \in \Omega^L$. These probabilities are then scaled according to individual line lengths $p_l^{scal} = p_l \frac{L_l - \mu_{len}}{\sigma_{len}}$ for $l \in \Omega^L$, assuming that longer lines have a higher outage probability than shorter lines. Here, L_l is the length of line l, μ_{len} is the mean length of all lines and σ_{len} is the standard deviation of the length of all lines. Joint probabilities for $k > 1$ are computed using the previously described copula analysis. We set the characteristic length L to 50 km and the maximum correlation ρ_0 to 0.15 resulting in the correlation plot shown in Figure 5.1. An outage frequency of 0.0051 /cctkmyr (circuit-kilometer-year) is chosen for all lines with an average repair time of 44h per individual line in outage [56]. It is assumed that for 10% line overloads no load shedding is necessary, but when a line is more than 10% overloaded, load shedding has to take place to avoid an outage.

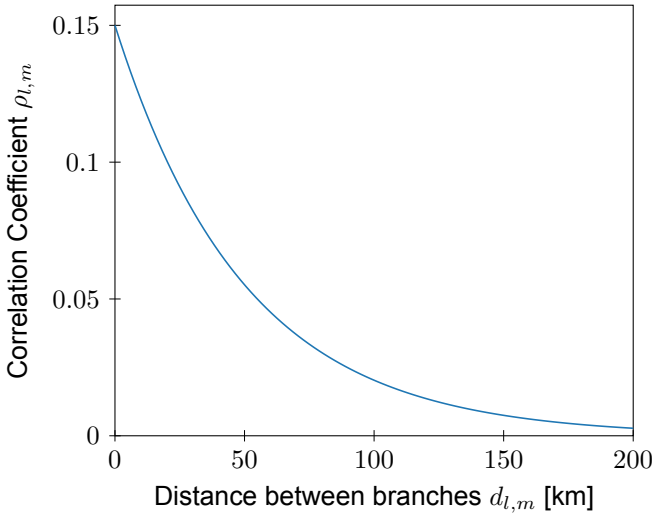


Figure 5.1: Plot showing the exponentially decaying correlation as distance increases.

The 39-bus system is tested on a 16GB RAM desktop with CPU. The 118-bus system is tested on the DelftBlue supercomputer with 16GB RAM and NVIDIA Volta V100S GPUs [62]. Both the DCOPF and SCOPF are implemented in the CVXPY 1.3.0 library [60] and the solver was ECOS 2.0.11. The CVXPYlayer is constructed using CVXPYlayers 0.1.5 [34]. The graph figures are generated using NETWORKX 3.1 [63]. The whole framework is implemented in PyTorch 1.13.1 [64]. The GCNN is made using Torch-Geometric 2.3.1 [65].

Table 5.1: System parameters for 39-bus and 118-bus test systems

	$\lvert\Omega^B\rvert$	$\lvert\Omega^G\rvert$	$\lvert\Omega^D\rvert$	$\lvert\Omega^L\rvert$
Case 39	27	10	21	33
Case 118	118	19	99	186

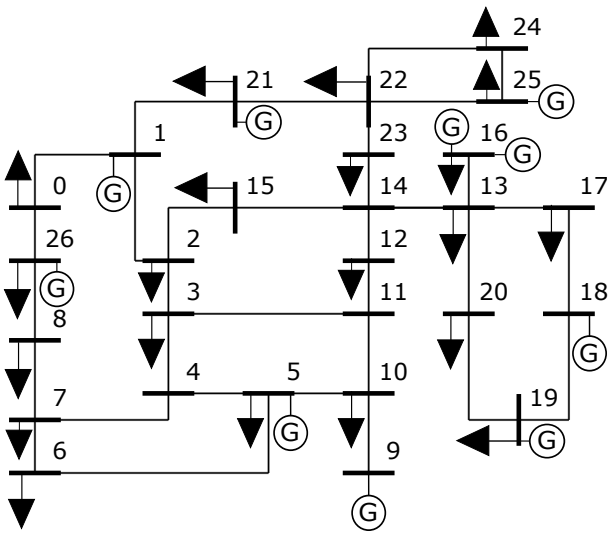


Figure 5.2: 39-bus system single line diagram. It is a modified version, excluding the transformers.

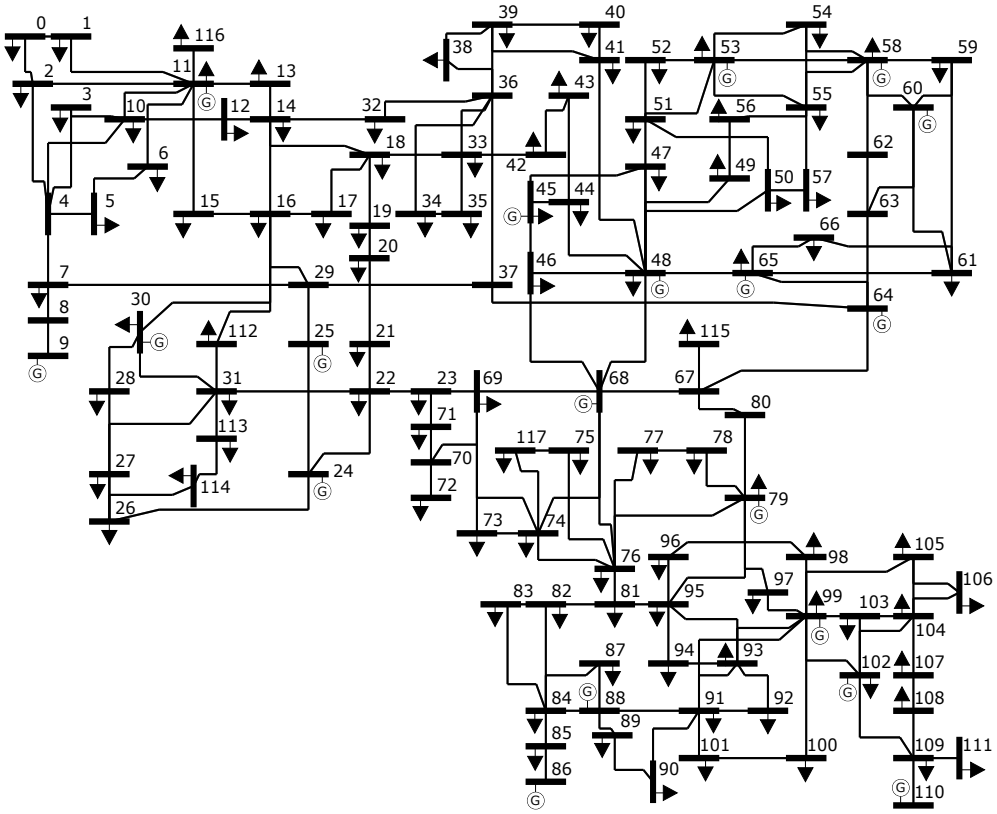


Figure 5.3: 118-bus system single line diagram.

5.1.1. Baselines

The baseline for the deterministic approach is a comprehensive N-1 SCOPF. When k >1 considering the complete set of contingencies becomes intractable utilizing off-the-shelf solvers, and the ground truth is obtained by performing a SCOPF with contingency screening (CS) using LODFs. The workflow of this CS method is illustrated in Figure 3.3. A heuristic (H) approach is used a second baseline, where the contingency list created by CS is used to solve a SCOPF on a new dataset, bypassing the iterative addition of contingencies. For the probabilistic approach, a baseline referred to as 'N-k SCOPF' is used. TSOs are integrating N-2 contingencies due to the heightened risk of multiple outages. The 'N-k SCOPF' includes a comprehensive N-1 SCOPF, encompassing the 20 most critical N-2 contingencies identified through a contingency filtering process.

To determine the optimal number of contingencies added per iteration, a case study is conducted on the N-2 and N-3 scenarios of the 39-bus system. The evaluation of the baseline's performance is focused around the trade-off between identifying violating post-contingency cases and computational time with the incremental addition of contingencies. The findings are depicted in Figure 5.4 and Figure 5.5. These figures illustrate that enhancing the solution's security comes at the cost of increased solving time for the optimization problem. Following the analysis, a decision is made to introduce 20 contingencies per iteration, resulting in a maximum of 60 post-contingency constraints added to the DCOPF.

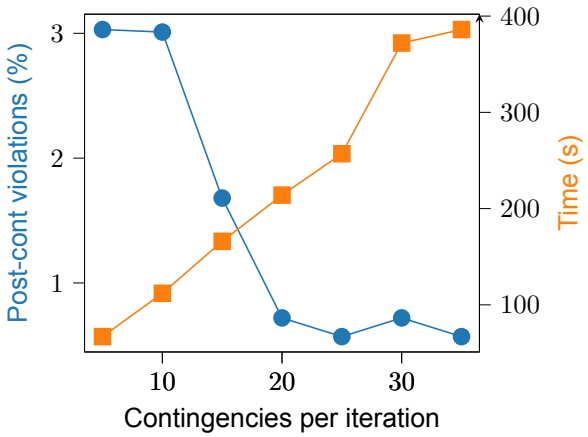


Figure 5.4: 39-bus N-2 post-contingency violations vs time to solve.

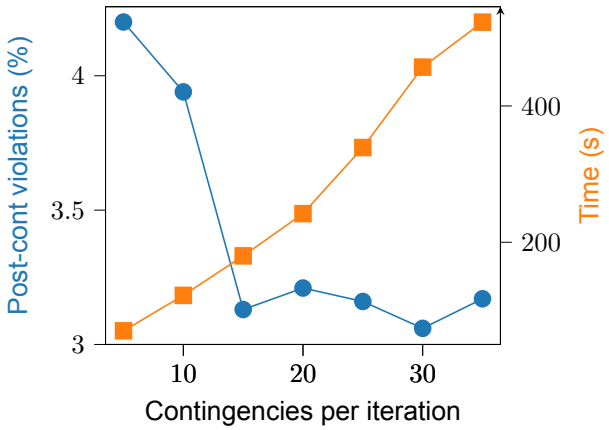


Figure 5.5: 39-bus N-3 post-contingency violations vs time to solve.

5.2. Memory Reduction

This case study assesses the achieved memory reduction by expressing the $\mathbf{FLODF}$ matrix in sparse format, and by reducing the computational graph. Table 5.2 presents the percentage of islanding cases, the sparsity of the $\mathbf{FLODF}$ matrix, and the achieved reduction in elements through its expression in COO format. A noteworthy decrease in the number of elements is evident.

Table 5.2: $\mathbf{FLODF}$ computed sparsity after construction

k	1	2	3	4	5	6
39-bus sparsity [%]	98.5	97.6	97.5	98.1	98.9	99.6
Reduction COO [%]	-95.6	-92.7	-92.4	-94.2	-96.7	-98.8
118-bus sparsity [%]	99.6	99.3	99.0	N/A	N/A	N/A
Reduction COO [%]	-98.8	-97.8	-97.0	N/A	N/A	N/A

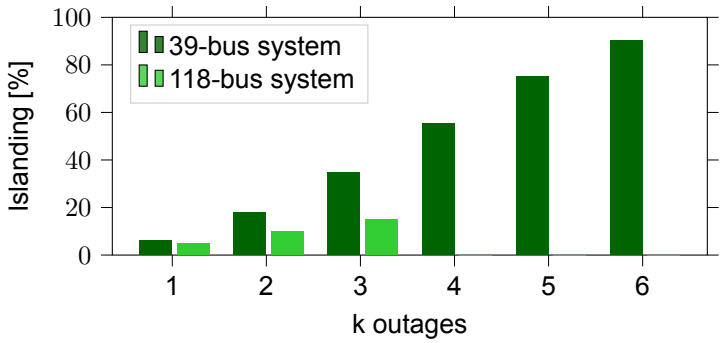


Figure 5.6: Percentage of contingency cases which are islanding cases as k increases.

For the 39-bus system, the 'reduced graph' algorithm is compared to a 'full graph' baseline with unmodified tensors using a batch size of 100 samples. For the N-5 case, a batch size of 50 is used due to memory issues of the 'full graph' baseline. Figure 5.7 shows that the 'reduced graph' algorithm is slightly slower during the forward pass as a result of the added operations. The increase in time is

small, as less than 1% of post-contingency flows violate constraints resulting in quick matrix multiplications with small matrices. In contrast, the N-5 case achieved a 100-fold memory reduction and 8x speedup in the backward pass by reducing the computational graph and the size of tensors involved in backpropagation.

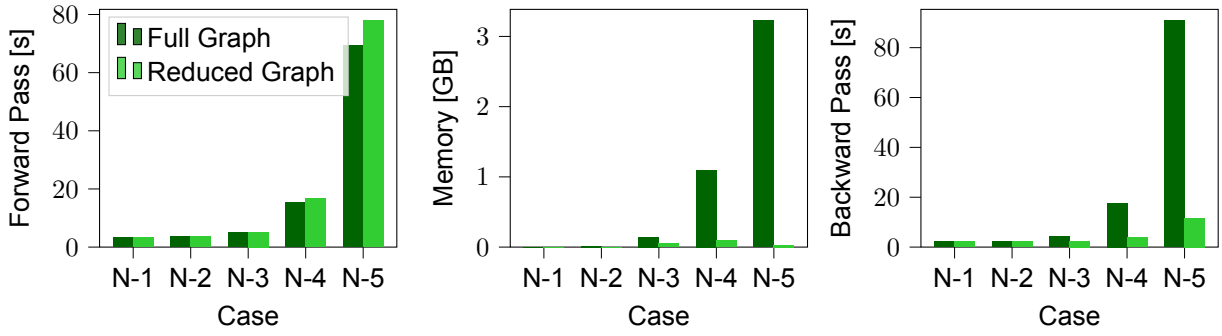


Figure 5.7: Comparison of the speed of the forward pass, the backward pass and the memory consumption of the full computational graph against the reduced computational graph.

5.3. Deterministic Post-Contingency Violations

The next case study assesses the success of the proposed approach in identifying post-contingency violations in comparison with the proposed baselines. For enhanced surveyability, the schematic pertaining to the model inference is shown in Figure 5.8.

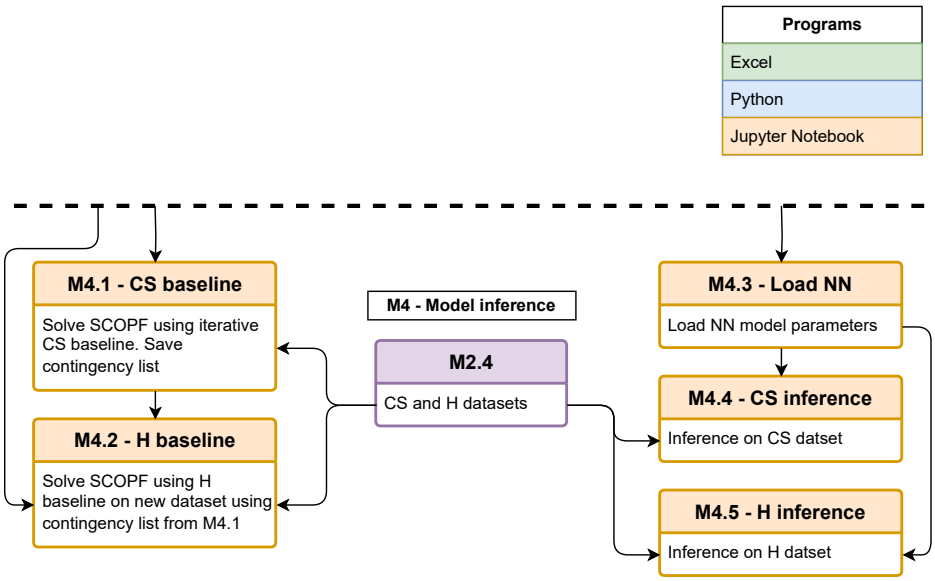


Figure 5.8: Schematic workflow corresponding to the deterministic post-contingency violation evaluation.

5.3.1. Multi-Layer Perceptron

Without Feasibility Restoration

To illustrate the challenge of obtaining a feasible solution when applying deep learning to constrained optimization problems, this case study assesses the proposed approach without the use of the feasibility restoration layer, denoted as MLP $\neq$ corr. A comparison with the CS baseline is shown in Table 5.3. The training parameters are shown in Table B.6.

Table 5.3: 39-bus system results proposed approach $\neq$ feasibility restoration layer

		N-1	N-2	N-3	N-4	N-5
Base case violations	SCOPF	0.00%	0.00%	0.00%	0.00%	0.00%
	MLP $\neq$ corr	39.80%	41.30%	42.20%	39.70%	39.30%
Post-cont violations	# contingencies	33	528	5,456	40,920	237,336
	SCOPF	0.06%	0.74%	3.30%	3.70%	3.51%
	MLP $\neq$ corr	0.32%	1.47%	3.36%	4.77%	4.79%
Time [s]	SCOPF	60	248	289	302	388
	MLP $\neq$ corr	5	5	7	22	90
	Increase	x12	x50	x41	x14	x4
Cost [$]	SCOPF	2041	2099	2144	2210	2209
	MLP $\neq$ corr	2156	2165	2164	2195	2151
	Increase	+5.63%	+3.14%	+0.93%	-0.68%	-2.63%

From the results in Table 5.3 it can be observed that the approach without feasibility restoration layer struggles to identify base case feasible solutions. Furthermore, when compared to the proposed approach that includes the feasibility restoration layer (Table B.8), it exhibits suboptimal performance in terms of dispatch cost and identifies fewer post-contingency cases with violations.

With Feasibility Restoration

Next, the results are presented of the proposed approach including the feasibility restoration layer.

39-bus system Figure 5.9 presents an absolute comparison between the proposed method and the CS baseline. Table 5.4 presents the relative differences. The Table containing all results can be found in Table B.8.

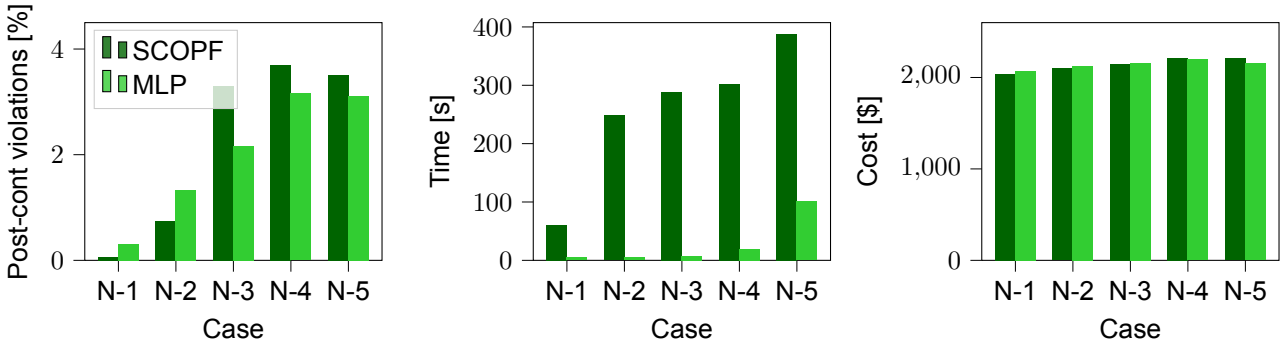


Figure 5.9: 39-bus system results proposed approach vs CS baseline.

Table 5.4: 39-bus MLP comparison of speed and cost increases with CS baseline

	N-1	N-2	N-3	N-4	N-5
Speed Increase	x12	x49	x41	x15	x4
Cost Change	+1.44%	+1.17%	+0.51%	-0.47%	-2.61%

Next, the proposed approach is compared to the H baseline, where the SCOPF is solved with a pre-

defined contingency list. Figure 5.10 presents the absolute values and Table 5.5 presents the relative differences. The full Table of results can be found in Table B.9.

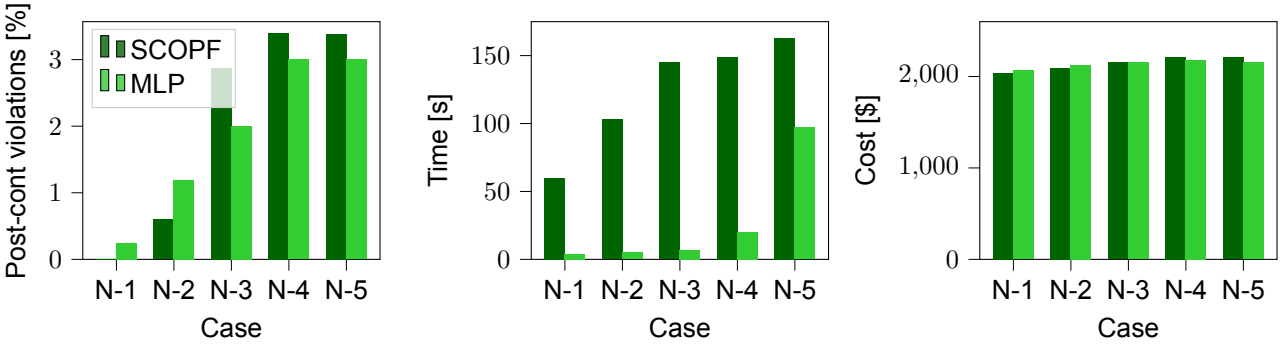


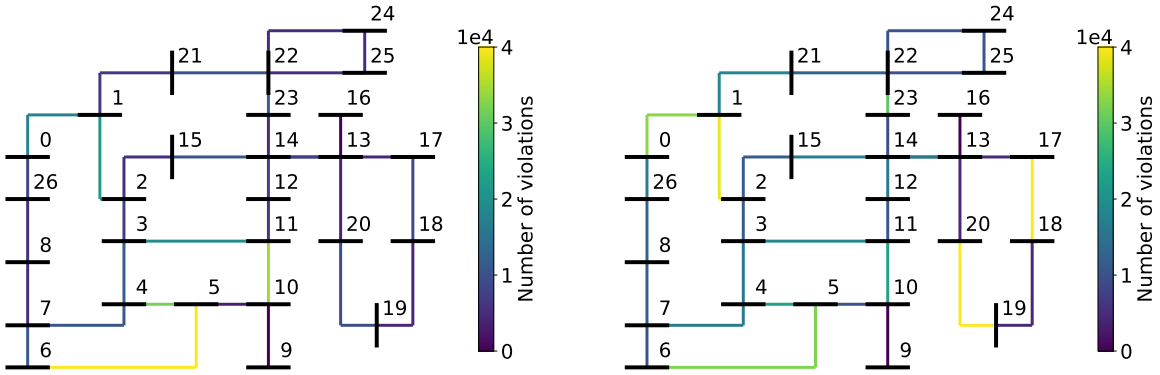
Figure 5.10: 39-bus system results proposed approach vs H baseline.

Table 5.5: 39-bus MLP comparison of speed and cost increases with H baseline

	N-1	**N-2**	**N-3**	**N-4**	**N-5**
Speed Increase	x15	x21	x21	x8	x2
Cost Change	+1.45%	+1.22%	+0.10%	-0.66%	-2.57%

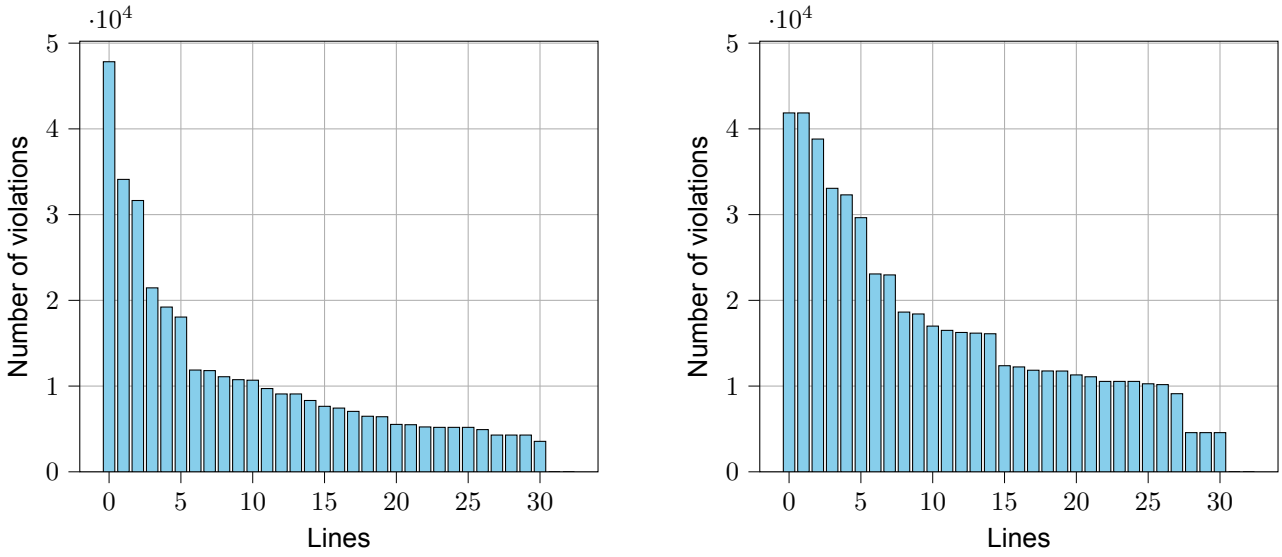
In the case of the 39-bus system, as Figure 5.9 and Figure 5.10 show, the proposed approach outperforms baselines for k > 2 by detecting more violating post-contingency cases, while remaining competitive for N-1 and N-2, with a maximum 0.7% absolute increase in violating cases for N-2. Moreover, the proposed approach exhibits speeds up to 21$\times$ faster than the H baseline for N-2 and N-3 cases. The H baseline, leveraging a pre-existing contingency list, significantly reduces computation time. The H method's efficiency results from the limited variability in loading profiles and spatial consistency of critical contingencies, effectively harnessing historical data.

Critical Lines 39-bus System To evaluate the location of the critical lines, the single line diagram in Figure A.9 is presented. It shows the number of violations for each line for the 39-bus N-3 case. The histogram in Figure A.10 shows the number of violations per line in descending order. These figures illustrate that the proposed approach identifies a different dispatch than the proposed baseline, resulting in different locations of critical lines. It is important to note that the goal of the proposed approach is not to mimic the proposed baseline. The single line diagrams and histograms for the 39-bus N-1 and N-2 cases can be found in the appendix in Figure A.5, Figure A.6, Figure A.7 and Figure A.8.



(a) Proposed approach. **(b)** Baseline.

Figure 5.11: 39-bus violations for the N-3 case, showing a decrease in the number violations.



(a) Proposed approach. **(b)** Baseline.

Figure 5.12: Number of violations per line, in descending order, for the 39-bus system N-3.

Security Level Comparison The models are trained for specific security levels, denoted as k $= \{1, 2, 3, 4, 5\}$. Figure 5.13 displays the percentage of identified violating cases for models trained at individual security levels and Table 5.2 shows the number of islanding cases. It reveals that the N-3 secure model offers security levels comparable to N-4 and N-5 models, primarily due to a notable number of islanding cases in N-4 and N-5. The Table containing all values can be found in Table B.10.

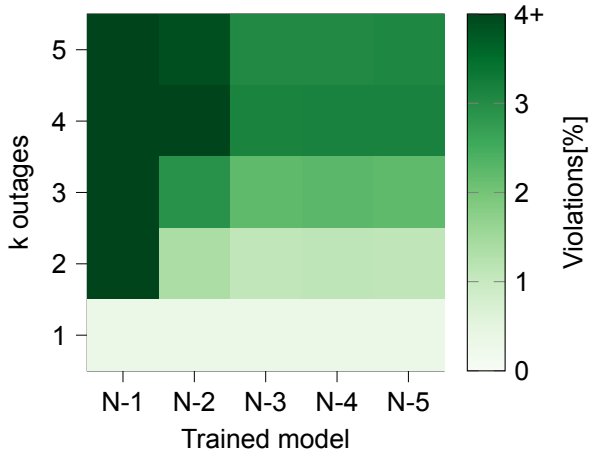


Figure 5.13: Comparison for 39-bus system of violating post-contingency cases for different contingency levels.

118-bus system Moving on to the 118-bus system, Figure 5.14 and Table 5.6 present the absolute and relative results respectively. The full Table of results can be found in Table B.12.

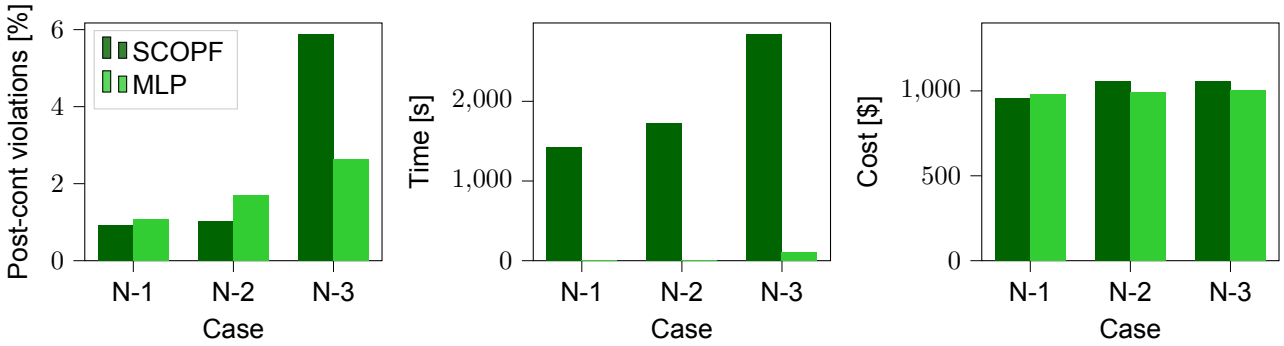


Figure 5.14: 118-bus system results proposed approach vs CS baseline.

Table 5.6: 118-bus MLP comparison of speed and cost increases with CS baseline

	N-1	**N-2**	**N-3**
Speed Increase	x158	x173	x27
Cost Change	+2.51%	-6.13%	-5.29%

Next, Figure 5.15 and Table 5.7 present the absolute and relative differences respectively of the H baseline compared to the proposed approach. Table B.13 presents the full results.

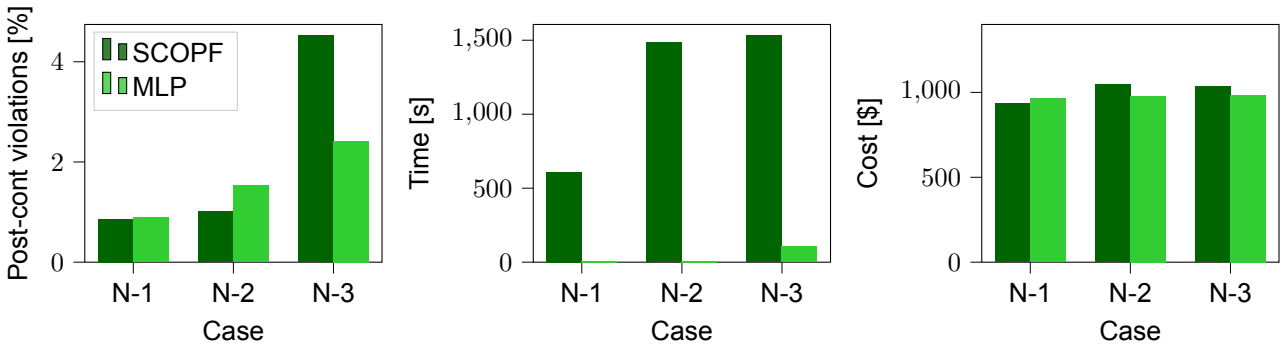


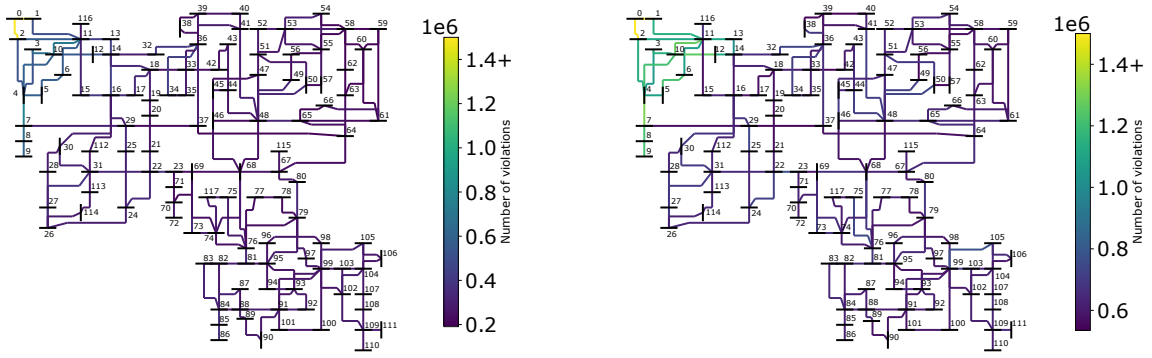
Figure 5.15: 118-bus system results proposed approach vs H baseline.

Table 5.7: 118-bus MLP comparison of speed and cost increases with H baseline

	N-1	**N-2**	**N-3**
Speed Increase	x76	x165	x15
Cost Change	+2.88%	-6.41%	-4.92%

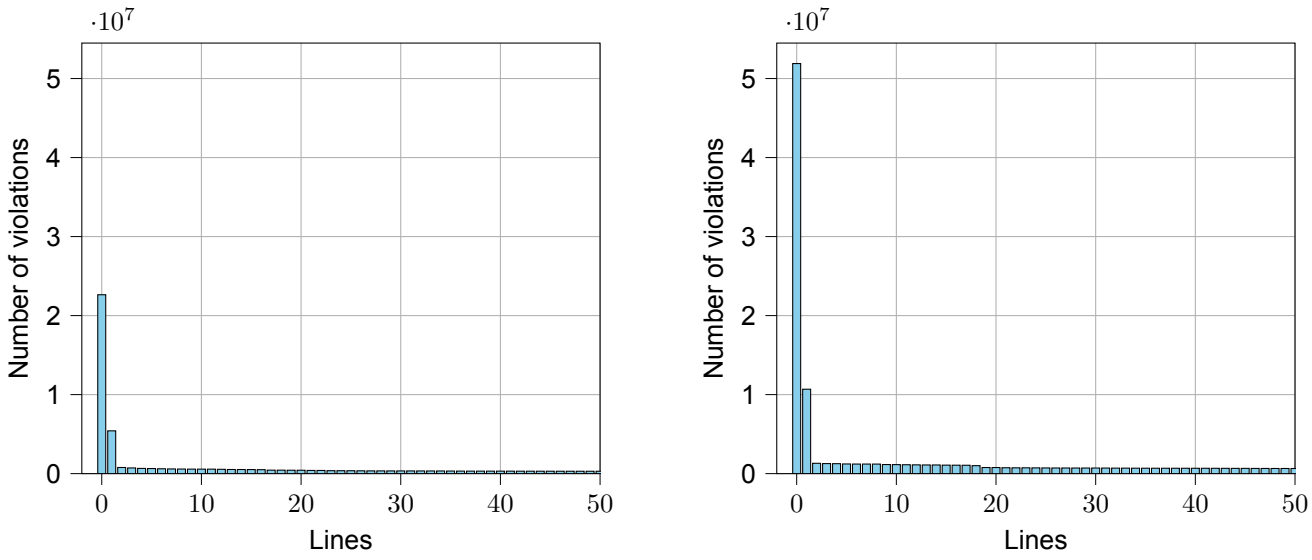
For the 118-bus N-1 case, the results of the proposed approach compared to the CS and H baseline are displayed in Figure 5.14 and Figure 5.15 respectively. Our proposed approach is 76$\times$ faster than the H baseline, with a slight increase in detected post-contingency violations and a 2.5% cost increase. In the 118-bus N-2 case, our approach is 176$\times$ faster, achieving a 6.1% cost improvement with a small increase in violating post-contingency cases. In the 118-bus N-3 case, our approach outperforms all baselines, notably reducing violating post-contingency cases by 1.9% in absolute terms.

Critical Lines 118-bus System To evaluate the location of the critical lines for the 118-bus system, the single line diagram in Figure 5.16 is presented. It shows the number of violations for each line for the 118-bus N-3 case. The histogram in Figure A.16 shows the number of violations per line in descending order. These figures reveal the identification of the same critical area in the North-West of the single line diagram. Notably, the proposed approach demonstrates a significant reduction in the number of violations within this critical area. The single line diagrams and histograms for the 118-bus N-1 and N-2 cases can be found in the appendix in Figure A.15, Figure A.12, Figure A.13 and Figure A.14.



(a) Proposed approach. **(b)** Baseline.

Figure 5.16: 118-bus violations for the N-3 case. The same critical area in the North-West of the single line diagram is identified by both methods. The proposed approach reduces the number of violations in that critical area.



(a) Proposed approach. **(b)** Baseline.

Figure 5.17: Number of violations per line, in descending order, for the 118-bus system N-3.

Security Level Comparison As shown in Figure 5.18, the N-3 trained model exhibits enhanced security against k = $\{1, 2\}$ outages compared to models specifically trained for those levels. Furthermore, the N-3 trained model achieves a dispatch cost similar to the N-2 trained model and only incurs a modest 2.5% cost increase compared to the N-1 trained model. This demonstrates that for a marginal cost increase, the N-3 model significantly enhances overall system security. The Table containing all values can be found in Table B.14.

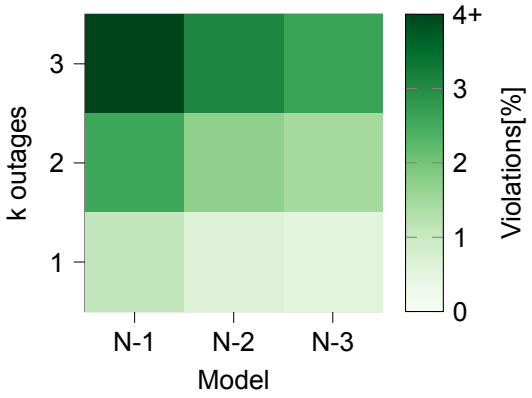


Figure 5.18: Comparison for 118-bus system of violating post-contingency cases for different contingency levels.

5.3.2. Graph Convolutional Neural Network

This case study incorporates a GCNN in the proposed approach. Based on the security level analysis presented earlier, it is established that achieving N-3 security is adequate for the 39-bus system. Consequently, the GCNN is assessed up to N-3 for the 39-bus system. The comparative analysis with the H baseline is depicted in Figure 5.19, while the comparison with the CS baseline is illustrated in Figure A.3. The results reveal the identification of numerous infeasible cases with a modest optimality gap in cost and a notable enhancement in computational speed. In contrast to the MLP, the GCNN exhibits a slight underperformance in terms of infeasible case detection and cost.

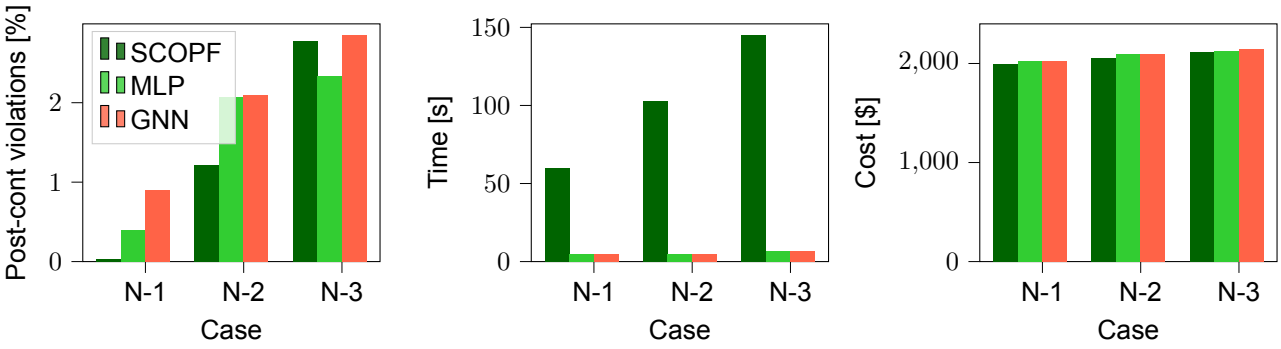


Figure 5.19: 39-bus system results proposed approach with GNN vs H baseline and MLP.

For the 118-bus system, the comparative analysis with the H baseline is shown in Figure 5.20, while the comparative analysis with the CS baseline is shown in Figure A.4. The proposed approach exhibits a notable underperformance specifically in detecting post-contingency violations for the N-2 case. Additionally, a suboptimal dispatch cost is observed when compared to the performance of the MLP.

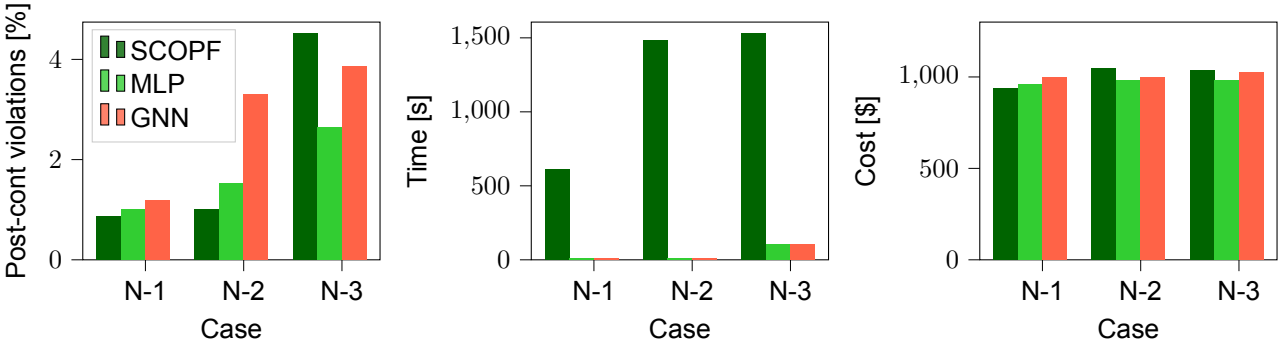


Figure 5.20: 118-bus system results proposed approach with GNN vs H baseline and MLP.

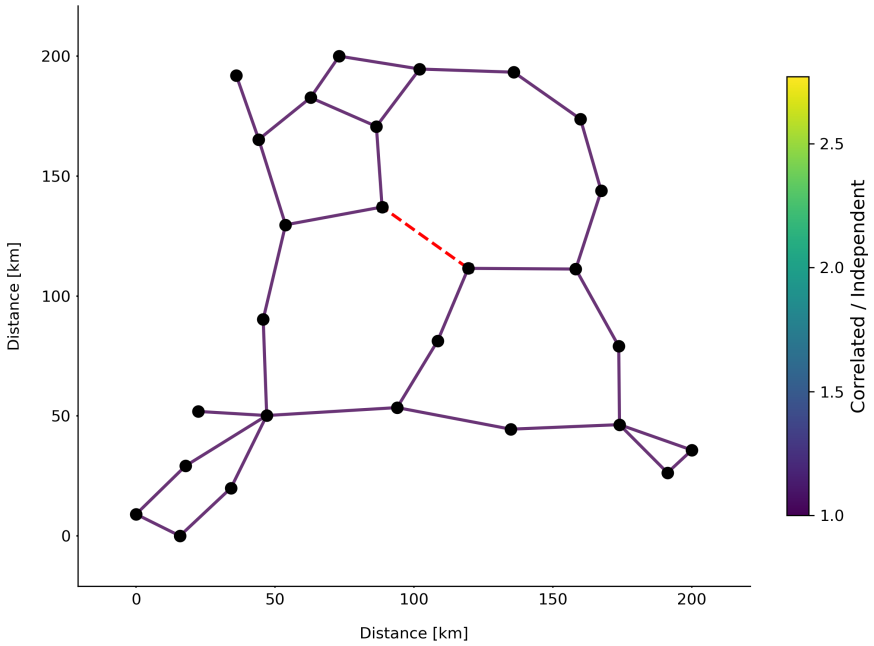
To summarize, the proposed approach incorporating a GCNN shows promise, but requires more research.

5.4. Probabilistic Security Assessment

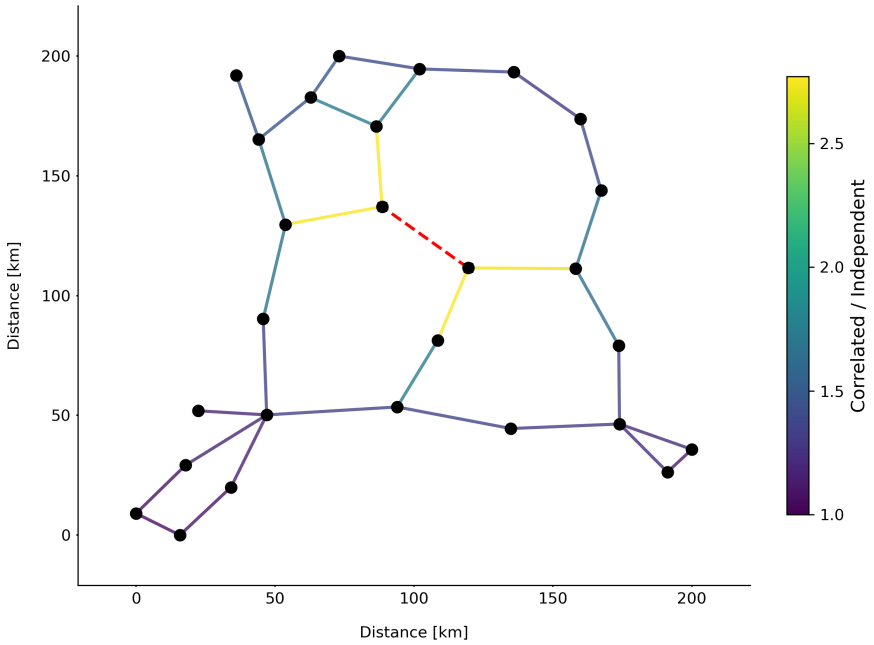
5.4.1. Copula Analysis

In the first case study of the probabilistic security assessment, the effect of the copula analysis on the joint probabilities is evaluated in comparison to the assumption of independent line failures. When independent line failures are assumed, the joint probability of multiple line failures is simply the product of the individual probabilities. In Figure 5.21, the probabilities computed using the copula analysis are compared to the independent probabilities, to visualize by how much the probabilities increase if a spatial correlation is incorporated.

In Figure 5.21a, the correlation is set to zero to see if the copula analysis computes the joint probabilities as independent probabilities. As can be seen from the figure, the ratio between the copula probabilities and the independent probabilities is 1, meaning the probabilities are the same, and the copula analysis considers the individual probabilities to be independent as expected. Furthermore, when a correlation is added to the copula analysis, it can be clearly seen in Figure 5.21b that the probabilities around the line in outage increase, and the probabilities of the lines far away from the outage can roughly be assumed to be independent.



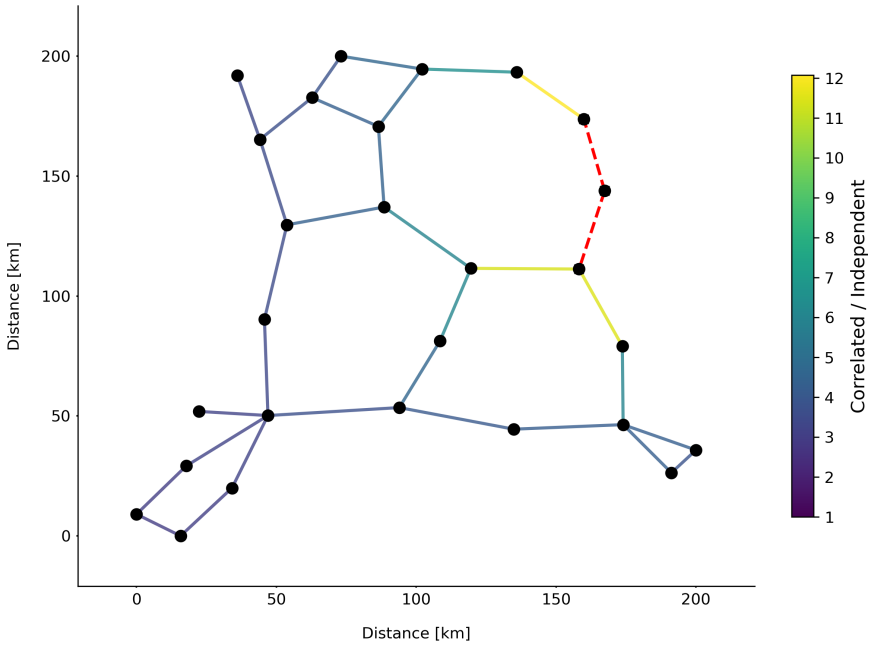
(a) Increased outage probability for N-2 failures with copula analysis and correlation set to 0.



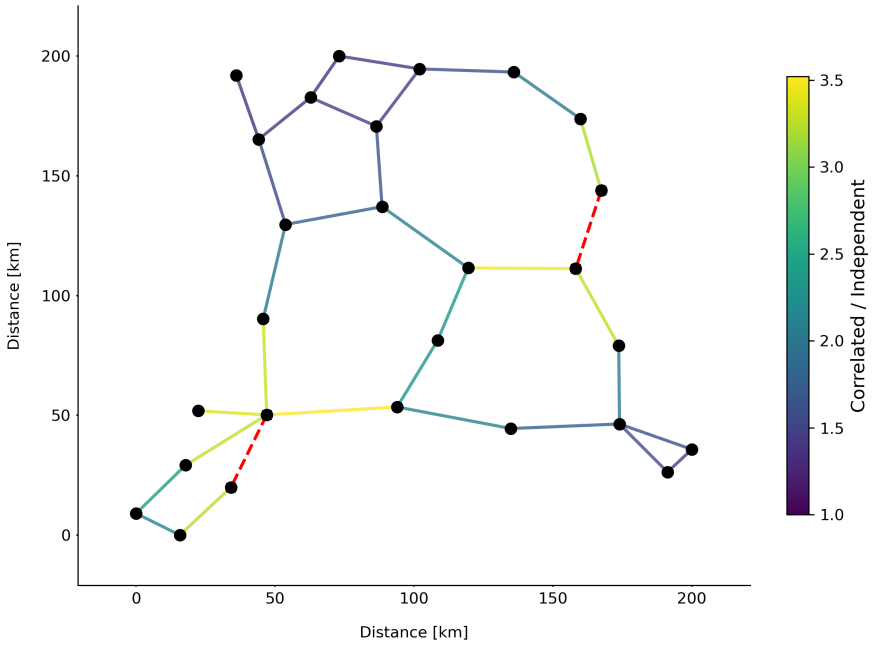
(b) Increased outage probability for N-2 failures with copula analysis and correlation set to 0.1.

Figure 5.21: Comparison of outage probabilities for specific N-2 cases.

When a N-3 case is considered, it can be seen from Figure 5.22a that if adjacent lines are in outage, the probability of outage of the surrounding lines significantly increase. When lines far apart experience an outage, roughly the same increase in probability as for N-2 cases is observed in Figure 5.22b.



(a) Increased outage probability for N-3 failures with copula analysis for adjacent lines



(b) Increased outage probability for N-3 failures with copula analysis for distant lines

Figure 5.22: Comparison of outage probabilities for specific N-3 cases.

This illustrates that incorporating correlations into outage probabilities leads to a substantial increase in the likelihood of N-k outages. By integrating these probabilities, derived from copula analysis, into the proposed risk-based framework, a more realistic depiction of multiple line outages can be achieved.

5.4.2. Risk-Based N-k SCOPF

For enhanced surveyability, the schematic of the workflow pertaining to the security assessment is shown in Figure 5.23.

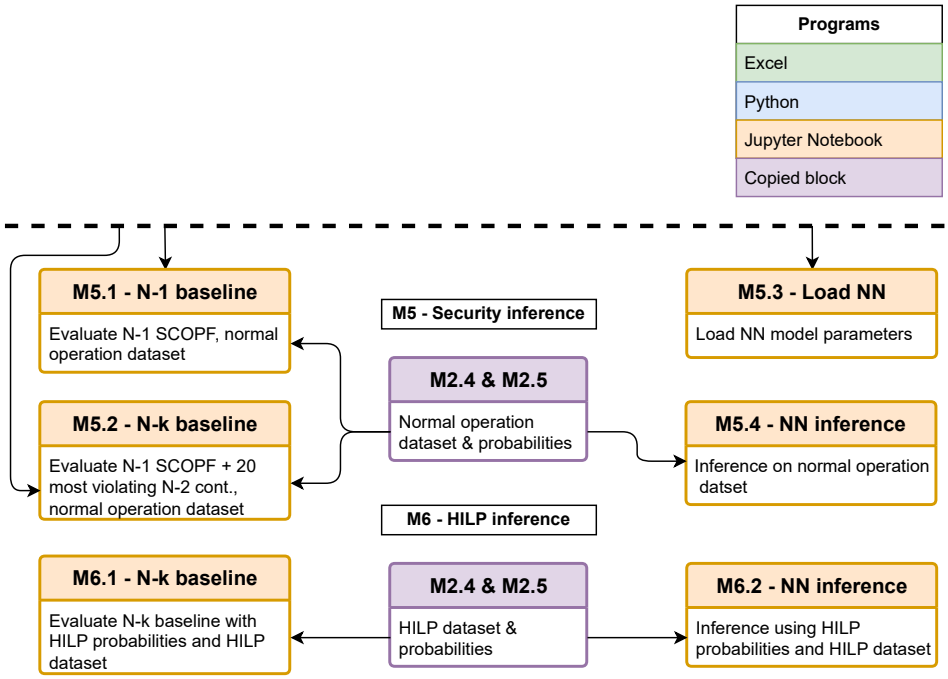


Figure 5.23: Schematic workflow corresponding to the probabilistic security assessment evaluation.

This case study assesses N-k SCOPFs with probability considerations. Instead of training the neural network solely on a single security level, we compute post-contingency flows for k $= \{1, 2, 3\}$ and then compute the expected flows by weighting them with their respective contingency probabilities.

Table 5.8: Comparison of LOLE and EENS between risk-based model and baselines

	LOLE [h/y]	EENS [MWh/y]
N-1 SCOPF	9.94	13.20
N-k SCOPF	1.08	1.85
Risk-Based Model	0.99	1.40

Table 5.9: Comparison of computational time and dispatch cost between between risk-based model and baselines

	N-1 SCOPF	N-k SCOPF	Risk-Based Model
Time [s]	61	139	8
Cost [$]	1968	2038	2076

Table 5.8 and Table 5.9 show that the N-1 SCOPF identifies the cheapest and least secure dispatch, lacking security against N-2 and N-3 contingencies. The proposed model demonstrates strong performance, showing slightly lower LOLE and EENS values compared to the N-k baseline, with only a 1.86% increase in dispatch cost and a 17$\times$ speedup.

5.4.3. Operational Resilience to HILP Events

To test the resilience of the proposed approach against HILP events, a case study is performed where the probabilities suddenly change due to an earthquake. The results are presented in Table 5.10. The earthquake simulation is represented using three concentric circles. Within the innermost circle, the individual probabilities of the lines increase significantly, while in the outermost circle, the individual probabilities show the least increase. The probabilities for k $= \{2, 3\}$ are recomputed. The risk-based model that is trained in the previous section is used, and the reliability indicators are recomputed. The proposed approach and the baseline exhibit similar LOLE values, but the proposed approach achieves a 21.5$\%$ reduction in EENS, highlighting its enhanced operational resilience against HILP events.

Table 5.10: Comparison of LOLE and EENS after HILP between risk-based model and N-k SCOPF baseline

	After HILP	
	LOLE [h/y]	EENS [MWh/y]
N-k SCOPF	13.37	28.28
Risk-Based Model	13.75	22.20

6

Conclusion and Discussion

6.1. Problem Statement

The transition to green energy is rapidly transforming the energy landscape, driven by the increased integration of renewable energy sources, distributed energy resources, and the electrification of various energy sectors. These developments pose a challenge to grid security, or operational reliability, making it harder to meet the N-1 security criterion, the failure of which has led to numerous blackouts. Furthermore, unforeseen weather events, now more frequent due to climate change, account for the second major cause of these blackouts. The growing grid complexity highlights the importance of N-k outages, but N-k security remains challenging due to its combinatorial complexity. Conventional methods struggle to efficiently scale with k. Additionally, HILP are occurring more frequently due to climate change, increasing the risk of blackouts which typically occur due to cascading failures or common cause events. This drives the need to develop a method which can approximate N-k SCOPFs, and to develop a method which enhances power systems security and resilience. In this thesis, a constraint-driven ML approach is introduced to address the challenges associated with N-k SCOPFs, considering all possible contingencies. Both a deterministic and probabilistic framework are implemented.

6.2. Proposed Approach

The proposed approach is based on the linearized DCOPF and uses a MLP to learn a mapping between loads as inputs and generator setpoints as outputs. A feasibility restoration layer is employed to restore the predicted generator setpoints to the base case feasible region. The set of all post-contingency flows is computed utilizing LODFs. Additionally, the loss function incorporates various terms, drawing inspiration from the penalty function method employed to transform constrained problems into unconstrained ones. These terms include dispatch cost, predicted violations in base case flows, imbalance between predicted generation and demand, and post-contingency flow violations. To additionally account for the probabilities of joint outages where k >1, a copula analysis is applied to compute these joint probabilities assuming a spatial correlation between individual line outages.

6.3. Research Questions and Answers

In this final chapter the core research questions are revisited. Based on the outcomes of the case studies, the following results emerge to the proposed research questions:

6.3.1. Constraint-Driven Deep Learning with LODFs

Objective 1: Develop a deterministic constraint-driven approach to approximate N-k SCOPFs, considering all possible contingencies using LODFs.

- Q1 Can LODFs be effectively employed to approximate N-k SCOPFs, incorporating all contingencies, in a scalable and computationally efficient manner?
- Q2 Can a constraint-driven ML approach reduce post-contingency violations?

Q1 - Scalable and Efficient LODFs The first research question is aimed to investigate whether LODFs can be effectively employed in a computationally efficient and scalable manner for approximating N-k SCOPFs considering all contingencies. The proposed approach effectively approximates N-k SCOPFs for the 39-bus and 118-bus systems up to k = 3. The approach is tractable by expressing the $\mathbf{FLODF}$ matrix in sparse format and by reducing the memory requirements related to the computational graph. Furthermore, by utilizing a MLP together with LODFs, the proposed approach efficiently incorporates many post-contingency inequality constraints. Incorporating as many inequality constraints as the proposed approach using conventional methods is not tractable. Particularly in extensive scenarios such as the 118-bus N-3 case, where over a million possible contingency cases exist, the proposed approach incorporating all these contingencies significantly outperforms conventional methods in terms of speed and post-contingency violation detection. Additionally, the topology-dependent LODFs need only be computed once or until the network topology changes.

That being said, the consideration of N-k contingencies holds combinatorial complexity, and different approaches for computing LODFs still rely on computationally inefficient matrix inversions [12, 25, 66, 67]. When scaling up to larger grids and for increasing k, the computation of all post-contingency flows becomes computationally challenging. While there is existing interest in accelerating N-k LODF computations [12], further research on fast N-k LODF computation remains necessary. Moreover, as the LODFs are topology dependent, they need to be recomputed if the topology changes. The $\mathbf{FLODF}$ matrix for the 118-bus N-3 case expressed in COO format is constructed in 20 minutes without GPU acceleration and requires 4.2GB of memory. Subsequently, the construction of the $\mathbf{FLODF}$ matrix for the N-4 case which considers a substantial 48 $\times 10^6$ contingency cases, resulted in memory issues when using 16GB of RAM. Although the occurrence of N-4 contingencies is rare, it indicates limited scalability when scaling up to large test systems. Moreover, as elaborated in Algorithm 2, the post-contingency flows are computed batch-wise for extensive case due to the occurrence of memory issues when performing a single sparse matrix multiplication with the complete $\mathbf{FLODF}$. This results in a slow computation of all post-contingency flows, and therefore in a long training and slow inference for extensive cases. Smart coding strategies, including potential parallelization, could optimize the computation process and yield significant speedups, especially when leveraging GPU acceleration. A final limitation is the exclusion of islanding cases in the proposed approach.

To conclude, utilizing LODFs the proposed approach effectively approximates a N-3 SCOPF for the 118-bus system considering all contingencies. However, due to memory requirements of the extensive LODF matrix, the N-3 case of the 118-bus approaches the limit of the approach. The proposed approach is promising, but to scale even further, more research is required.

Q2 - Constraint-Driven ML Approach The second research question aims to discover whether the proposed constraint-driven approach can reduce post-contingency violations. To assess the effectiveness of the constraint-driven ML approach, we conducted a comparative analysis with a conventional baseline, utilizing 39-bus and 118-bus test cases. In the 39-bus system, the proposed MLP-based approach achieved a remarkable 21$\times$ speed improvement over the fastest baseline, while remaining competitive in post-contingency violation detection and dispatch cost. For the 118-bus system, the proposed approach employing a MLP demonstrated strong performance in terms of dispatch cost, speed and post-contingency violation detection, making it a promising approach for larger systems. The notable 165$\times$ speedup for the N-2 case and a 6$\%$ decrease in cost with a 2$\%$ absolute reduction in post-contingency violations for the N-3 case are particularly significant. The slow solving time of the baselines underlined the slow convergence of conventional methods utilized for solving N-k SCOPFs. For the 118-bus N-2 and N-3 cases, as the solution space is significantly constrained by the incorporated contingencies, the solver in the baseline seems to prioritize feasibility over optimality in terms of dispatch cost. The case study, which introduced a GCNN into the proposed approach, revealed the potential for improved scalability.

While a constraint-driven approach offers the advantage of learning directly from the problem specification without the need for labeled training data, it's essential to note that, being a ML model, the strict enforcement of post-contingency constraints using the MLP is not guaranteed. When specific contingency security is required, a feasibility check remains unavoidable. In the event of post-contingency violations, redispatching is still necessary. Additionally, more research is necessary incorporating a

GCNN in the proposed approach. Theoretically, the number of parameters of a GNN should not scale with system size. However, for the 118-bus system, a greater number of parameters compared to the 39-bus system was required to achieve the results presented.

To conclude, the constraint-driven approach effectively reduces post-contingency violations. However, being a ML model, strict enforcement of post-contingency constraints can not be guaranteed.

6.3.2. Probabilistic Security Assessment

Objective 2: Perform a probabilistic security assessment to formulate an N-k risk-based security criterion, providing an alternative to the current deterministic N-1 security criterion.

- Q1 Does a probabilistic risk-based security criterion incorporating all contingencies improve power systems security?
- Q2 Does this proposed risk-based security criterion increase power systems resilience?

Q3 - Improved Power Systems Security The third research question aims at uncovering whether the proposed risk-based security criterion improves power systems security. During normal operation, the proposed risk-based security criterion demonstrates a significant decrease in LOLE and EENS compared to the current conventional N-1 security criterion. Additionally, similar values of LOLE and EENS are observed in comparison with the N-k baseline, along with a notable $17\times$ speedup and a small optimality gap in terms of dispatch cost of 1.86%. These results indicate that the proposed approach holds potential for increased reliability.

Nevertheless, the performed reliability study is not very extensive. The computation of the LOLE and the EENS is done utilizing contingency probabilities, while conventional reliability studies rely on various simulations considering different scenarios [2, 55]. Therefore, the performed case study serves only as an indication of the reliability of the proposed approach. In order to gain a more comprehensive understanding of the proposed approach's reliability, these reliability indices should be computed using extensive simulation methods, considering multiple future scenarios.

Q4 - Enhanced Power Systems Resilience The fourth research question aims at evaluating possible added resilience by the proposed approach. When faced with rapidly changing probabilities of a HILP event, the proposed approach exhibited comparable performance to the N-k baseline in terms of LOLE, while achieving a notable EENS reduction of approximately 21%. This suggests potential for increased resilience against unforeseen weather related events.

However, the same limitation holds as for the reliability study. A single case study to assess power systems resilience merely serves as an insight into the added resilience of the proposed approach. The growing interest in power systems resilience [1, 3, 4] has led to the introduction of various metrics to evaluate resilience, including reliability indices [2, 4, 57]. However, these indices, traditionally used for reliability evaluation, offer limited insights when evaluated against a single HILP event. To comprehensively assess the added resilience of the proposed approach, a more thorough evaluation is recommended.

Q2 - Additional Discussion Growing grid complexity underlines the increasing relevance of reliability and resilience studies [1]. While operational security and resiliency studies can be very extensive [2, 3, 4, 5, 57], the discussion below only focuses on the limitations of the proposed case studies.

The reliability case study involves inference from 1000 random samples, yet the assessment deviates from the conventional definition of reliability. True reliability assesses a system's ability to withstand disturbances and common failures during normal operation over a specified time period. In the proposed case study, all samples have the same outage probabilities, considering only load profile variations. In reality, power systems are faced with a lot more stochastic behaviour over prolonged normal operation periods, resulting from varying outage probabilities, and varying weather conditions [55]. Threats and component vulnerabilities influencing outage probabilities fluctuate over time. These factors fluctuate seasonally, geographically, and depend on equipment factors like thermal aging [5]. Moreover, the weather influences energy consumption, and the amount of energy produced by RES, resulting in even

more stochastic behaviour in load and generation. A comprehensive reliability study should at least encompass multiple scenarios for specified studied periods, accounting for changing outage probabilities and diverse weather conditions [55]. Such a case study allows for comparison of the reliability indices with other studies. Additionally, other reliability indices could be included in the evaluation [55, 56, 57].

While reliability pertains to normal operation, resiliency addresses specific and rare events. The performed case study evaluates two reliability indices after the occurrence of a single HILP event. However, in the context of short-term resiliency, three separate phases can be distinguished [1, 57]. These phases pertain to preventive, corrective and restorative actions. A truly resilient power system should predict, adapt, and recover from rare events with minimal human intervention [57]. The current case study, which solely examines reliability indices without incorporating recovery actions, provides only a partial evaluation of power systems' short-term resilience. Reliability based metrics give some indication of power systems resilience, but a comprehensive understanding of resilience requires a broader study [4].

To conclude, the proposed case studies indicate potential for heightened reliability and resilience. However, the case studies provide limited insight, and more research is recommended.

6.4. Future Work

Building on this thesis, an extension of the proposed approach involves the consideration of additional component outages (e.g. generator outages) into the approach. In alignment with conventional methods for preventive SCOPF, the present study exclusively focuses on the analysis of line outages [14, 15, 48]. This is particularly limiting during the energy transition, which is marked by an increased number generators due to the heightened integration of renewables. Reference [68] argues that generator outages should be considered in preventive SCOPF, but the inclusion of generator outages introduces binary variables, resulting in a more challenging MILP problem. Reference [16] mimicks a C&CGA for N-1 SCOPF to include binary variables. Similar approaches could be investigated to incorporate generator outages in the proposed approach.

Transitioning from preventive to corrective SCOPF, an avenue for future research involves the utilization of the proposed approach for corrective control. When a N-k violation takes place, the priority shifts away from dispatch cost considerations and the approach may be employed for corrective control during the restorative phase of an event. Consequently, the approach can be employed to enhance power systems resilience in this restorative phase [1].

Extending its utility, the proposed approach could be modified to function under diverse conditions, considering potential variations in system topology and contingency probabilities. The approach is currently trained for a specific system topology and for specific contingency probabilities. However, in the event of an outage or unforeseen changes in probabilities, retraining the approach becomes necessary. Moreover, in order to ensure efficient and reliable system operation, system operators perform transmission switching and topology control, further changing network topology [69]. Reference [70] introduces a ML approach for approximating the OPF problem under power grid topology reconfigurations. If the proposed approach can be effectively retrained after changed probabilities and reconfigurations, the approach could find broader purpose.

Shifting to alternative neural network architectures, future research could delve deeper into the use of GNNs within the proposed approach, providing a more scalable alternative to MLPs. Initial results show promise, but comprehensive testing is necessary to evaluate scalability and performance in terms of dispatch cost and post-contingeny violation detection. References [53, 71] propose methods which accurately approximate ACOPFs utilizing GNNs, but future research is necessary for N-k SCOPFs.

Finally, to address the main bottleneck considering scalability, future research must focus on more efficient methods for computing and storing LODFs. The current thesis employs a 'brute force' approach, necessitating numerous matrix inversions for LODF computation. To enhance scalability, investigating more scalable alternatives becomes imperative, particularly for larger test systems. One potential approach for computing N-k LODFs involves utilizing only N-1 LODFs, reducing the number of matrix inversions necessary. Additionally, there is room for exploring sophisticated techniques to compute the entire set of post-contingency flows, aiming to further reduce memory consumption and broaden

the applicability of the approach to even larger cases. TTD emerges as a potential research direction in this context. Future research could focus on constructing a TT decomposed tensor with sensitivity factors and developing a loss function in TT form.

References

[1] Florin Capitanescu. “Are We Prepared Against Blackouts During the Energy Transition?: Probabilistic Risk-Based Decision Making Encompassing Jointly Security and Resilience”. In: *IEEE Power and Energy Magazine* 21.3 (2023), pp. 77–86.

[2] Mathaios Panteli and Pierluigi Mancarella. “Modeling and evaluating the resilience of critical electrical power infrastructure to extreme weather events”. In: *IEEE Systems Journal* 11.3 (2015), pp. 1733–1742.

[3] Mingguo Hong et al. “Enhancement of operational resilience with the online weather look-ahead study”. In: *2021 IEEE Power & Energy Society General Meeting (PESGM)*. IEEE. 2021, pp. 01–05.

[4] Narayan Bhusal et al. “Power system resilience: Current practices, challenges, and future directions”. In: *IEEE Access* 8 (2020), pp. 18064–18086.

[5] Emanuele Ciapessoni et al. “Probabilistic risk-based security assessment of power systems considering incumbent threats and uncertainties”. In: *IEEE Transactions on Smart Grid* 7.6 (2016), pp. 2890–2903.

[6] Sanja Cvijic and Jinjun Xiong. “Security constrained unit commitment and economic dispatch through benders decomposition: A comparative study”. In: *2011 IEEE Power and Energy Society General Meeting*. IEEE. 2011, pp. 1–8.

[7] Qin Wang et al. “Solving corrective risk-based security-constrained optimal power flow with Lagrangian relaxation and Benders decomposition”. In: *International Journal of Electrical Power & Energy Systems* 75 (2016), pp. 255–264.

[8] J.F. Benders. “Partitioning procedures for solving mixed-variables programming problems ‘”. In: *Numerische mathematik* 4.1 (1962), pp. 238–252.

[9] Alexandre Velloso, Pascal Van Hentenryck, and Emma S Johnson. “An exact and scalable problem decomposition for security-constrained optimal power flow”. In: *Electric Power Systems Research* 195 (2021), p. 106677.

[10] Alinson Santos Xavier et al. “Transmission constraint filtering in large-scale security-constrained unit commitment”. In: *IEEE Transactions on Power Systems* 34.3 (2019), pp. 2457–2460.

[11] Diego A Tejada-Arango, Pedro Sánchez-Martın, and Andres Ramos. “Security constrained unit commitment using line outage distribution factors”. In: *IEEE Transactions on power systems* 33.1 (2017), pp. 329–337.

[12] P Kaplunovich and K Turitsyn. “Fast and reliable screening of N-2 contingencies”. In: *IEEE Transactions on Power Systems* 31.6 (2016), pp. 4243–4252.

[13] Liping Huang et al. “Robust Nk security-constrained optimal power flow incorporating preventive and corrective generation dispatch to improve power system reliability”. In: *CSEE Journal of Power and Energy Systems* (2022).

[14] Xiang Pan et al. “Deepopf: A deep neural network approach for security-constrained dc optimal power flow”. In: *IEEE Transactions on Power Systems* 36.3 (2020), pp. 1725–1735.

[15] Sidhant Misra, Line Roald, and Yeesian Ng. “Learning for constrained optimization: Identifying optimal active constraint sets”. In: *INFORMS Journal on Computing* 34.1 (2022), pp. 463–480.

[16] Alexandre Velloso and Pascal Van Hentenryck. “Combining deep learning and optimization for preventive security-constrained DC optimal power flow”. In: *IEEE Transactions on Power Systems* 36.4 (2021), pp. 3618–3628.

[17] Priya Donti et al. “Adversarially robust learning for security-constrained optimal power flow”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 28677–28689.

[18] James Kotary et al. “End-to-end constrained optimization learning: A survey”. In: *CoRR abs/2103.16378* (2021).

[19] Priya L. Donti, David Rolnick, and J Zico Kolter. “DC3: A learning method for optimization with hard constraints”. In: *International Conference on Learning Representations*. 2021. url: https://openreview.net/forum?id=V1ZHVxJ6dSS.

[20] Stephen Frank, Ingrida Steponavice, and Steffen Rebennack. “Optimal power flow: A bibliographic survey I: Formulations and deterministic methods”. In: *Energy systems* 3 (2012), pp. 221–258.

[21] Laurence A Clarfeld et al. “Risk of cascading blackouts given correlated component outages”. In: *IEEE Transactions on Network Science and Engineering* 7.3 (2019), pp. 1133–1144.

[22] Laurence A Clarfeld et al. “Assessing risk from cascading blackouts given correlated component failures”. In: *2018 Power Systems Computation Conference (PSCC)*. IEEE. 2018, pp. 1–7.

[23] Allen J Wood, Bruce F Wollenberg, and Gerald B Sheblé. *Power generation, operation, and control*. John Wiley & Sons, 2013.

[24] Spyros Chatzivasileiadis. “Optimization in modern power systems”. In: *Lecture Notes. Tech. Univ. of Denmark* (2018).

[25] Teoman Guler, George Gross, and Minghai Liu. “Generalized line outage distribution factors”. In: *IEEE Transactions on Power systems* 22.2 (2007), pp. 879–881.

[26] Ignacio Aravena and Daniel K. Molzahn et al. “Recent Developments in Security-Constrained AC Optimal Power Flow: Overview of Challenge 1 in the ARPA-E Grid Optimization Competition”. In: (June 2022). url: http://arxiv.org/abs/2206.07843.

[27] Gareth James et al. *An introduction to statistical learning*. Vol. 112. Springer, 2013.

[28] Alexander Jung. “Machine Learning: The Basics”. In: Machine Learning: Foundations, Methodologies, and Applications. Springer, Singapore, 2022. doi: https://doi.org/10.1007/978-981-16-8193-6.

[29] Xiang Pan, Tianyu Zhao, and Minghua Chen. “DeepOPF: Deep Neural Network for DC Optimal Power Flow”. In: (May 2019). url: http://arxiv.org/abs/1905.04479.

[30] Rahul Nellikkath and Spyros Chatzivasileiadis. “Physics-informed neural networks for ac optimal power flow”. In: *Electric Power Systems Research* 212 (2022), p. 108412.

[31] Ahmed S Zamzam and Kyri Baker. “Learning optimal solutions for extremely fast AC optimal power flow”. In: *2020 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. IEEE. 2020, pp. 1–6.

[32] Wenlong Liao et al. “A review of graph neural networks and their applications in power systems”. In: *Journal of Modern Power Systems and Clean Energy* 10.2 (2021), pp. 345–360.

[33] Brandon Amos and J Zico Kolter. “OptNet: Differentiable Optimization as a Layer in Neural Networks”. In: (2017).

[34] A. Agrawal et al. “Differentiable Convex Optimization Layers”. In: *Advances in Neural Information Processing Systems*. 2019.

[35] Priya L Donti, Jeff Schneider, and M Granger Morgan. *Bridging Deep Learning and Electric Power Systems*. 2022.

[36] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[37] Daniel Fontaine, LaurentMichel, and Pascal Van Hentenryck. “Constraint-based lagrangian relaxation”. In: *Principles and Practice of Constraint Programming: 20th International Conference, CP 2014, Lyon, France, September 8-12, 2014. Proceedings 20*. Springer. 2014, pp. 324–339.

[38] Ferdinando Fioretto, Terrence WK Mak, and Pascal Van Hentenryck. “Predicting ac optimal power flows: Combining deep learning and lagrangian dual methods”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 34. 01. 2020, pp. 630–637.

[39] Ferdinando Fioretto et al. “Lagrangian duality for constrained deep learning”. In: *Machine Learning and Knowledge Discovery in Databases. Applied Data Science and Demo Track: European Conference, ECML PKDD 2020, Ghent, Belgium, September 14–18, 2020, Proceedings, Part V*. Springer. 2021, pp. 118–135.

[40] I. V. Oseledets. “Tensor-train decomposition”. In: *SIAM Journal on Scientific Computing* 33 (5 2011), pp. 2295–2317. issn: 10648275. doi: 10.1137/090752286.

[41] Andrzej Cichocki et al. “Tensor networks for dimensionality reduction and large-scale optimization part 1 low-rank tensor decompositions”. In: *Foundations and Trends in Machine Learning* 9 (4-5 2016), pp. 249–429. issn: 19358245. doi: 10.1561/2200000059.

[42] Andrzej Cichocki et al. “Tensor networks for dimensionality reduction and large-scale optimizations: Part 2 applications and future perspectives”. In: *Foundations and Trends in Machine Learning* 9 (6 2017), pp. 431–673. issn: 19358245. doi: 10.1561/2200000067.

[43] Stephan Rabanser, Oleksandr Shchur, and Stephan Günnemann. “Introduction to Tensor Decompositions and their Applications in Machine Learning”. In: (Nov. 2017). url: http://arxiv.org/abs/1711.10781.

[44] Liping Huang et al. “Robust N-k security-constrained optimal power flow incorporating preventive and corrective generation dispatch to improve power system reliability”. In: *CSEE Journal of Power and Energy Systems* (2022). doi: 10.17775/cseejpes.2021.06560.

[45] Yuan Li and James D McCalley. “A general benders decomposition structure for power system decision problems”. In: *2008 IEEE International Conference on Electro/Information Technology*. IEEE. 2008, pp. 72–77.

[46] Dimitris Bertsimas, David B Brown, and Constantine Caramanis. “Theory and applications of robust optimization”. In: *SIAM review* 53.3 (2011), pp. 464–501.

[47] Qianfan Wang, Jean-Paul Watson, and Yongpei Guan. “Two-stage robust optimization for Nk contingency-constrained unit commitment”. In: *IEEE Transactions on Power Systems* 28.3 (2013), pp. 2366–2375.

[48] Florin Capitanescu et al. “Contingency filtering techniques for preventive security-constrained optimal power flow”. In: *IEEE Transactions on Power Systems* 22.4 (2007), pp. 1690–1697.

[49] Yeesian Ng et al. “Statistical learning for DC optimal power flow”. In: (2018), pp. 1–7.

[50] Deepjyoti Deka and Sidhant Misra. “Learning for DC-OPF: Classifying active sets using neural nets”. In: *2019 IEEE Milan PowerTech*. IEEE. 2019, pp. 1–6.

[51] Huiwen Jia and Siqian Shen. “Benders Cut Classification via Support Vector Machines for Solving Two-stage Stochastic Programs”. In: (Mar. 2021).

[52] Rahul Nellikkath and Spyros Chatzivasileiadis. “Physics-Informed Neural Networks for Minimising Worst-Case Violations in DC Optimal Power Flow”. In: *2021 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. IEEE. 2021, pp. 419–424.

[53] Damian Owerko, Fernando Gama, and Alejandro Ribeiro. “Optimal power flow using graph neural networks”. In: *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2020, pp. 5930–5934.

[54] Seonho Park et al. “Confidence-aware graph neural networks for learning reliability assessment commitments”. In: *IEEE Transactions on Power Systems* (2023).

[55] Tennet TSO B.V. “Rapport Monitoring Leveringszekerheid 2021”. In: *ESP-MA 2021-018* (2021).

[56] Bart W Tuinema et al. *Probabilistic Reliability Analysis of Power Systems*. Springer, 2020.

[57] Maedeh Mahzarnia et al. “A review of the measures to enhance power systems resilience”. In: *IEEE Systems Journal* 14.3 (2020), pp. 4059–4070.

[58] Kai Zhou, Ian Dobson, and Zhaoyu Wang. “The most frequent Nk line outages occur in motifs that can improve contingency selection”. In: *IEEE Transactions on Power Systems* (2023).

[59] Ian Dobson, Benjamin A Carreras, and David E Newman. “A loading-dependent model of probabilistic cascading failure”. In: *Probability in the Engineering and Informational Sciences* 19.1 (2005), pp. 15–32.

[60] Steven Diamond and Stephen Boyd. “CVXPY: A Python-embedded modeling language for convex optimization”. In: *Journal of Machine Learning Research* 17.83 (2016), pp. 1–5.

[61] Sogol Babaeinejadsarookolaee et al. “The power grid library for benchmarking ac optimal power flow algorithms”. In: *arXiv preprint arXiv:1908.02788* (2019).

[62] Delft High Performance Computing Centre (DHPC). *DelftBlue Supercomputer (Phase 1)*. `https://www.tudelft.nl/dhpc/ark:/44463/DelftBluePhase1`. 2022.

[63] Aric Hagberg, Pieter Swart, and Daniel S Chult. *Exploring network structure, dynamics, and function using NetworkX*. Tech. rep. Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

[64] Adam Paszke et al. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. url: `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`.

[65] Matthias Fey and Jan E. Lenssen. “Fast Graph Representation Learning with PyTorch Geometric”. In: *ICLR Workshop on Representation Learning on Graphs and Manifolds*. 2019.

[66] Jiachun Guo et al. In: *IEEE Transactions on Power Systems* 24 (3 2009), pp. 1633–1634. issn: 08858950. doi: `10.1109/TPWRS.2009.2023273`.

[67] Henrik Ronellenfitsch, Marc Timme, and Dirk Witthaut. “A dual method for computing power transfer distribution factors”. In: *IEEE Transactions on Power Systems* 32.2 (2016), pp. 1007–1015.

[68] Yury Dvorkin et al. “Optimizing primary response in preventive security-constrained optimal power flow”. In: *IEEE Systems Journal* 12.1 (2016), pp. 414–423.

[69] Kory W Hedman, Shmuel S Oren, and Richard P O’Neill. “A review of transmission switching and network topology optimization”. In: *2011 IEEE power and energy society general meeting*. IEEE. 2011, pp. 1–7.

[70] Yexiang Chen et al. “A meta-learning approach to the optimal power flow problem under topology reconfigurations”. In: *IEEE Open Access Journal of Power and Energy* 9 (2022), pp. 109–120.

[71] Maosheng Gao et al. “A Physics-Guided Graph Convolution Neural Network for Optimal Power Flow”. In: *IEEE Transactions on Power Systems* (2023).

A

Figures

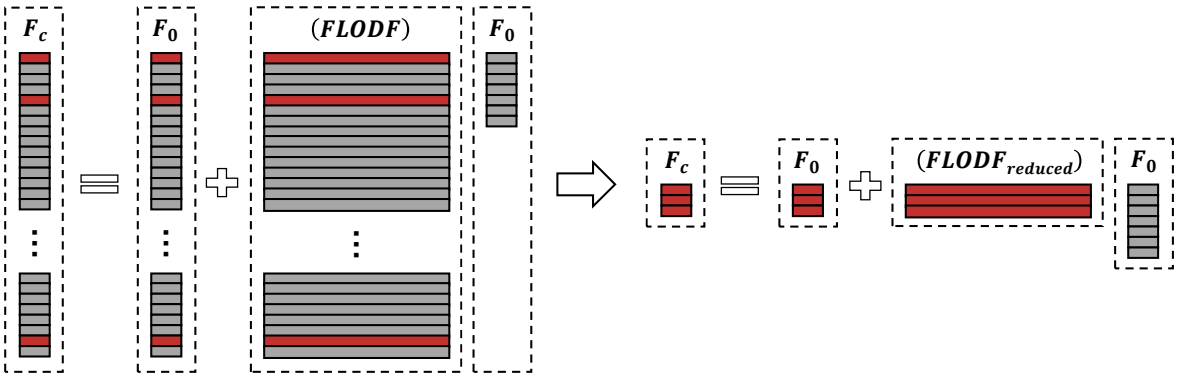
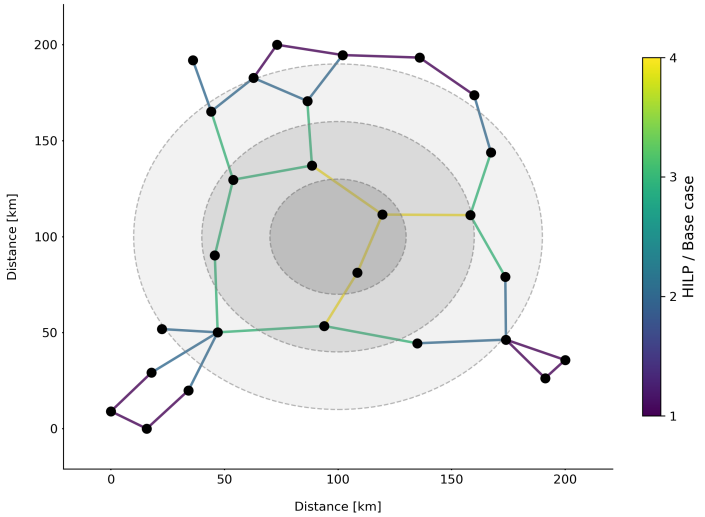
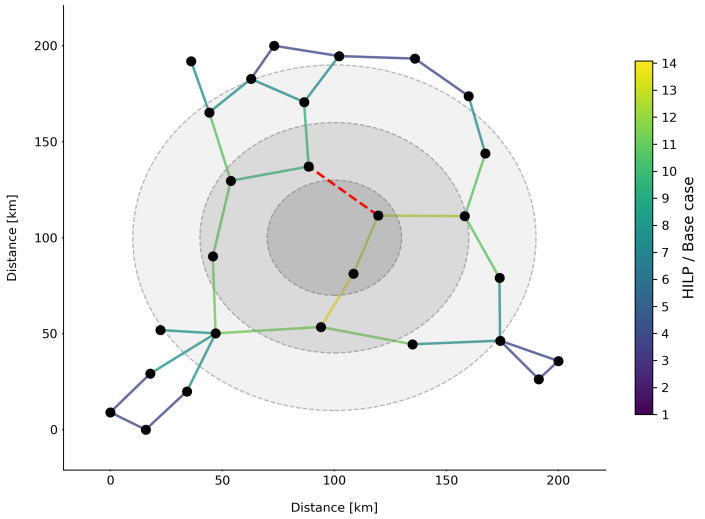


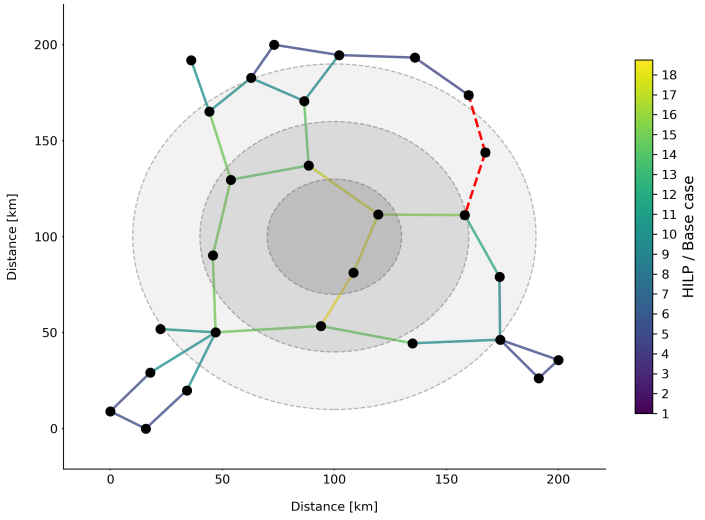
Figure A.1: Visualization of the reduced $\mathbf{FLODF}$. The lines in red represent the post-contingency flows in violation. By removing the rows corresponding to the non-violating post-contingency flows, the sizes of the tensors involved and their computational graph are significantly reduced.



(a) Increased outage probability for N-1 failures due to HILP event.



(b) Increased outage probability for specific N-2 failure due to HILP event.



(c) Increased outage probability for specific N-3 failure due to HILP event.

Figure A.2: Changing probabilities due to HILP event, simulated by an earthquake.

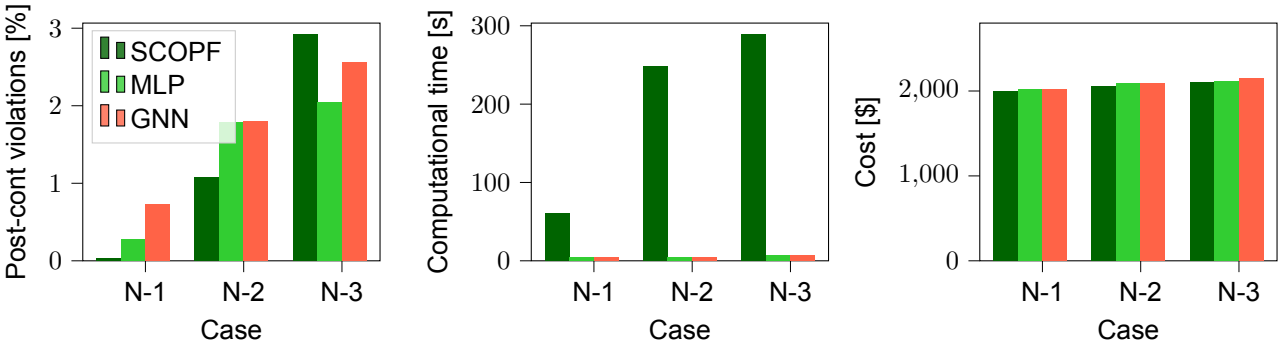


Figure A.3: 39-bus system results proposed method with GNN vs CS benchmark and MLP.

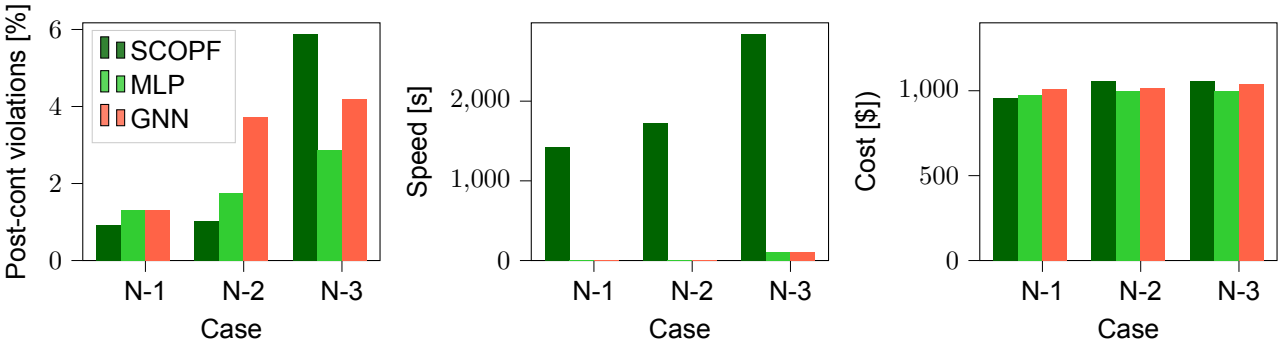
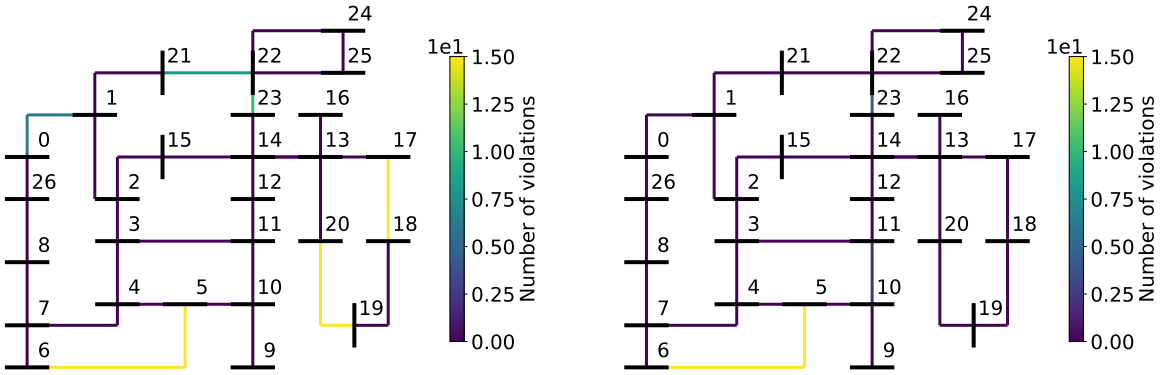
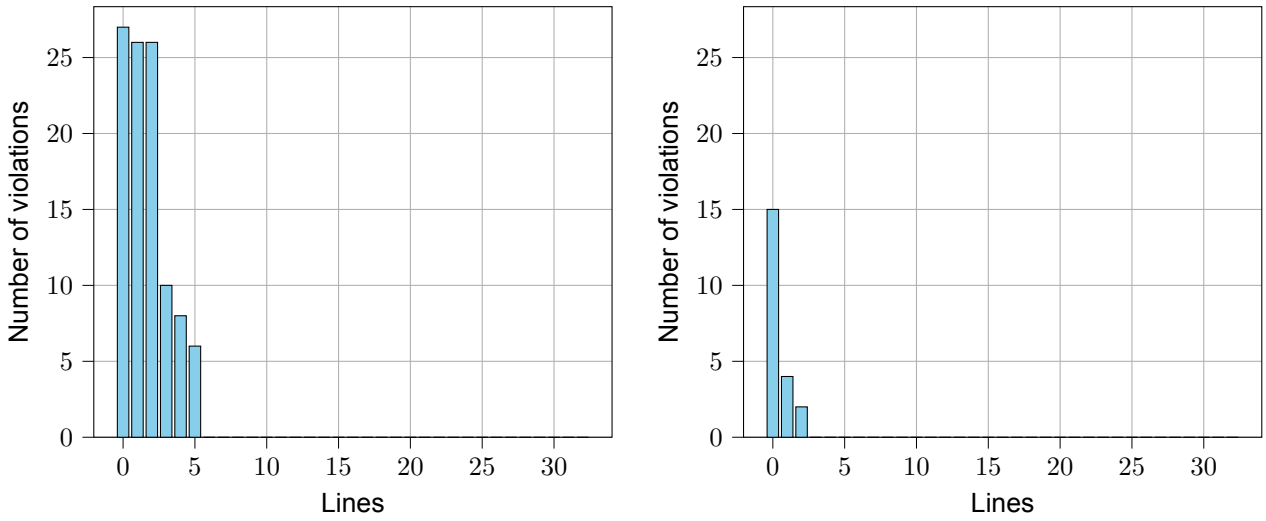


Figure A.4: 118-bus system results proposed method with GNN vs CS benchmark and MLP.



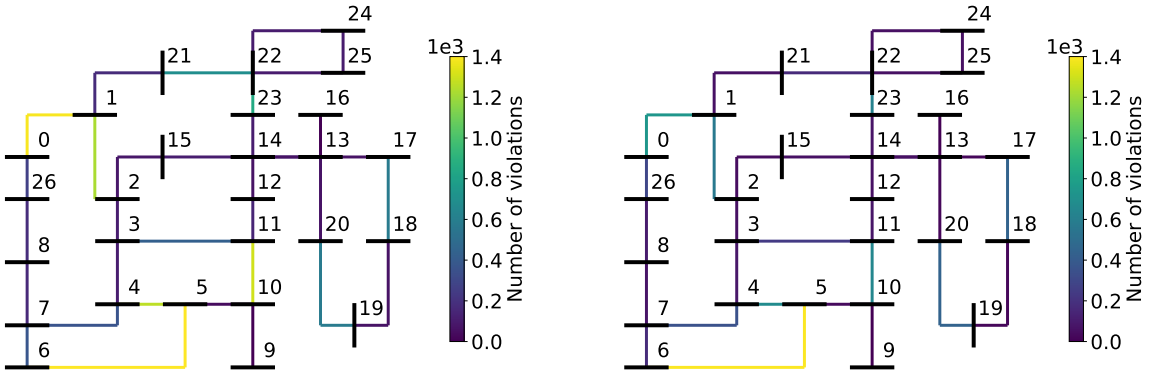
(a) Proposed approach. **(b)** Baseline.

Figure A.5: 39-bus violations for the N-1 case.



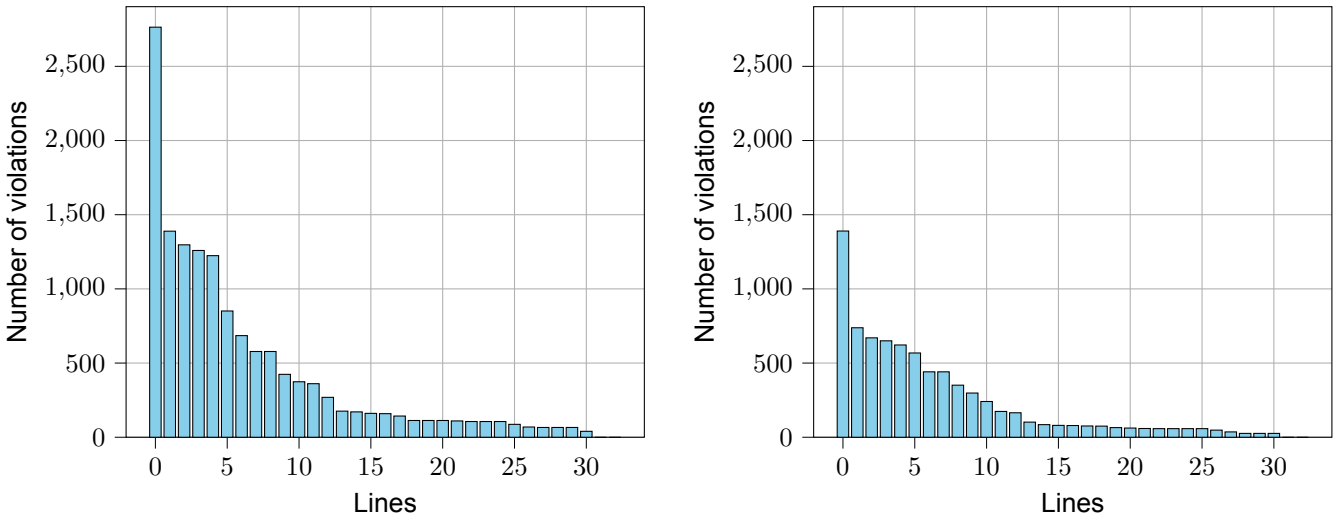
(a) Proposed approach. **(b)** Baseline.

Figure A.6: Number of violations per line, in descending order, for the 39-bus system N-1.



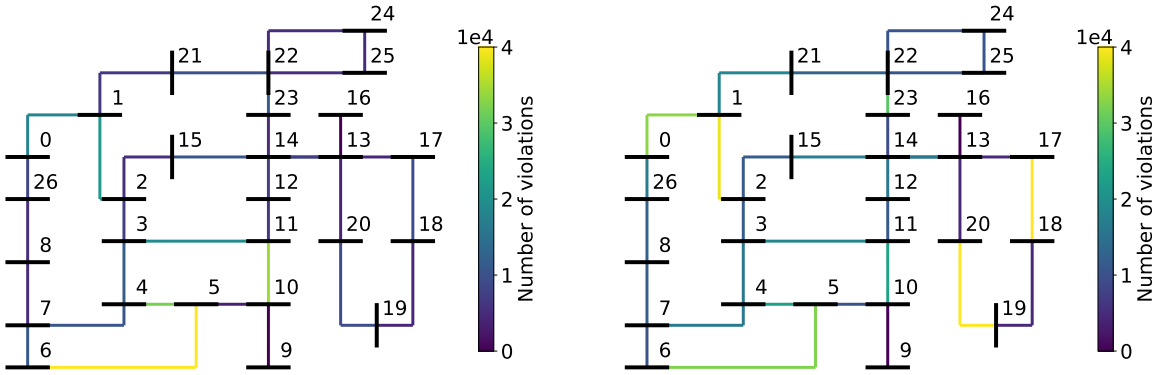
(a) Proposed approach. **(b)** Baseline.

Figure A.7: 39-bus violations for the N-2 case.



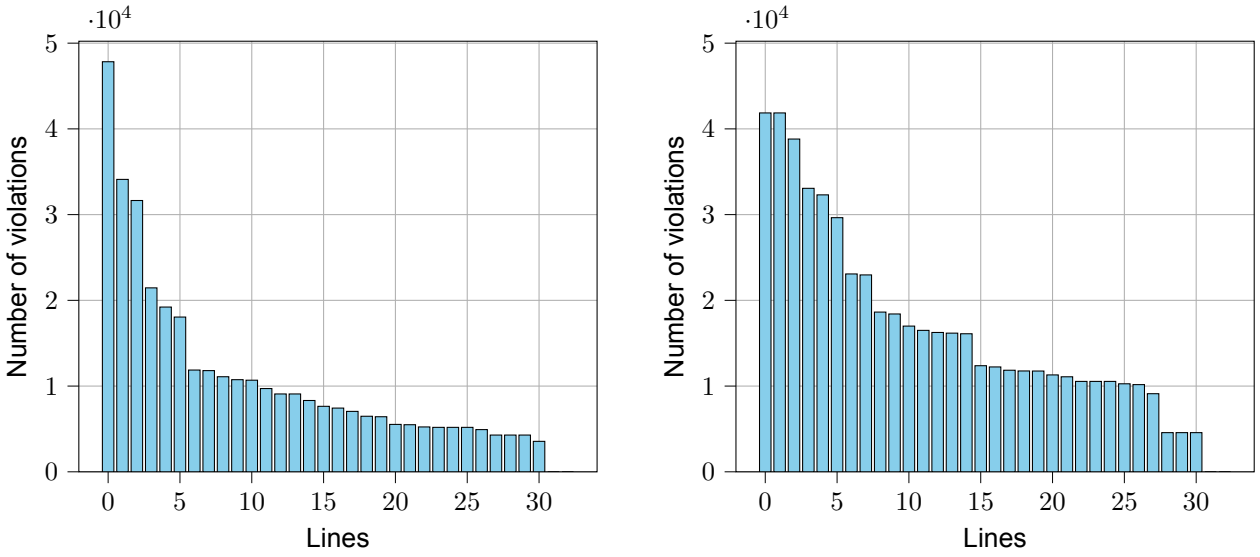
(a) Proposed approach. **(b)** Baseline.

Figure A.8: Number of violations per line, in descending order, for the 39-bus system N-2.



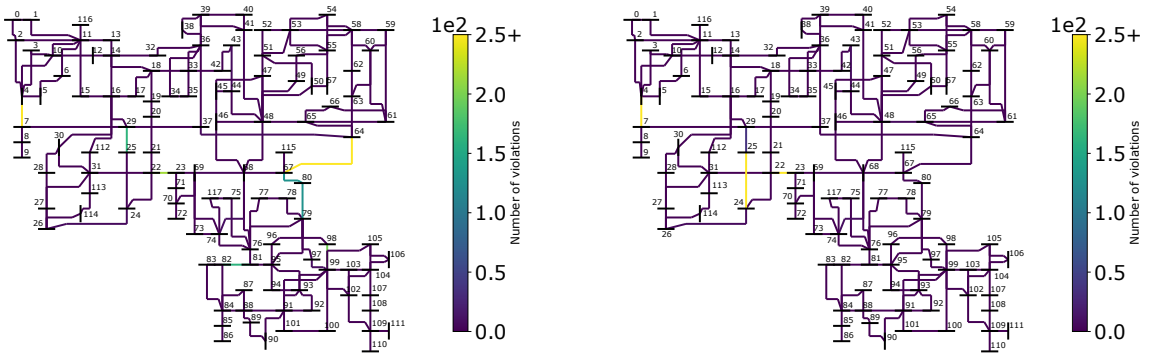
(a) Proposed approach. **(b)** Baseline.

Figure A.9: 39-bus violations for the N-3 case.



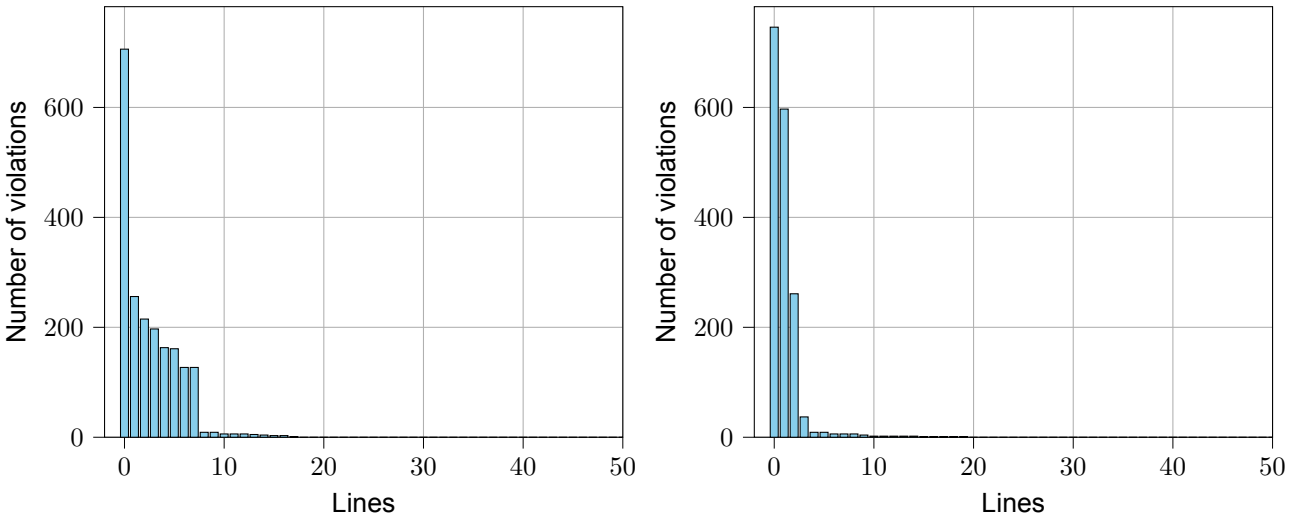
(a) Proposed approach. **(b)** Baseline.

Figure A.10: Number of violations per line, in descending order, for the 39-bus system N-3.



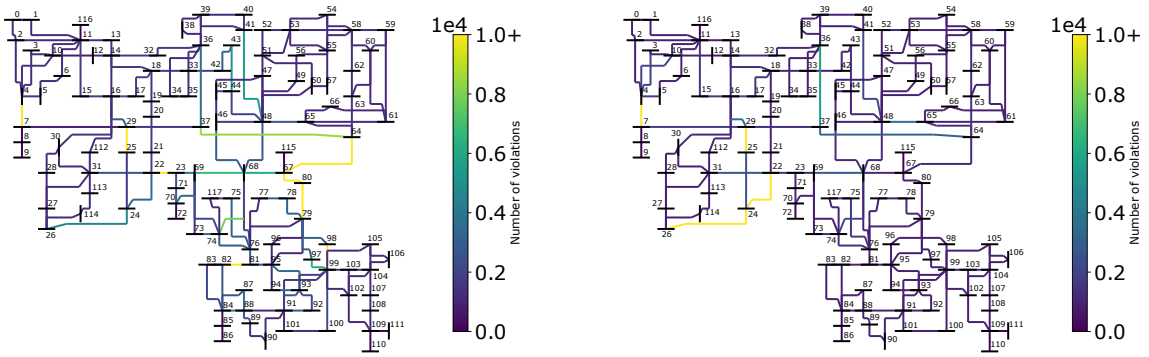
(a) Proposed approach. **(b)** Baseline.

Figure A.11: 118-bus violations for the N-1 case.



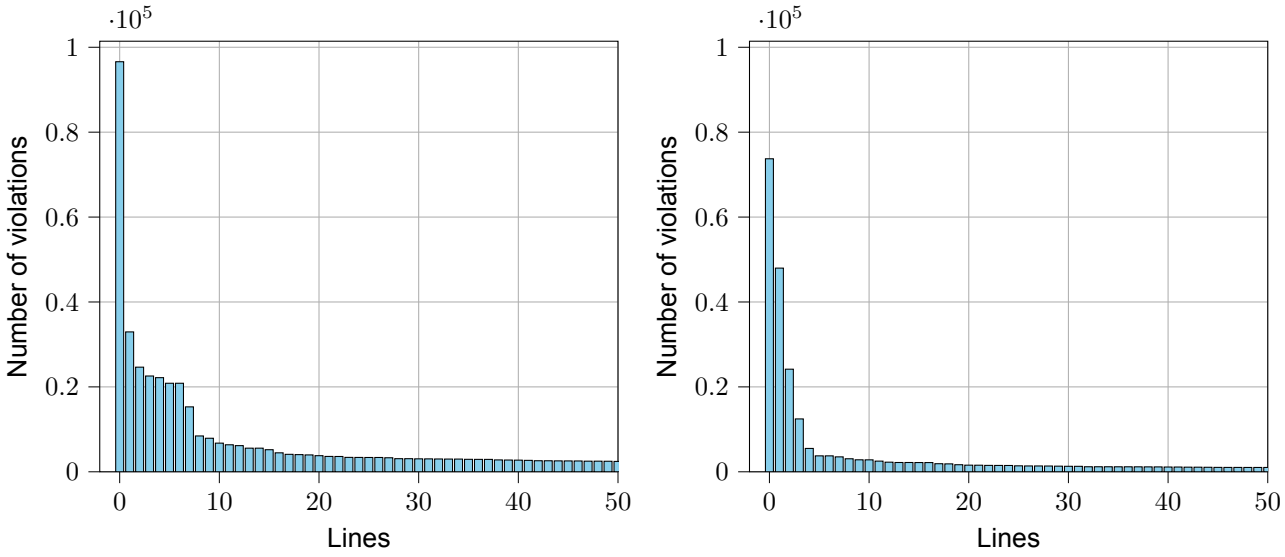
(a) Proposed approach. **(b)** Baseline.

Figure A.12: Number of violations per line, in descending order, for the 118-bus system N-1.



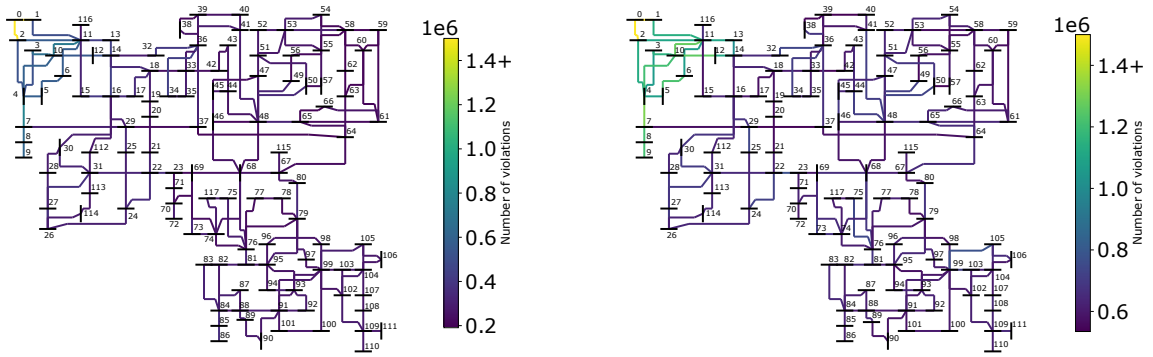
(a) Proposed approach. **(b)** Baseline.

Figure A.13: 118-bus violations for the N-2 case.



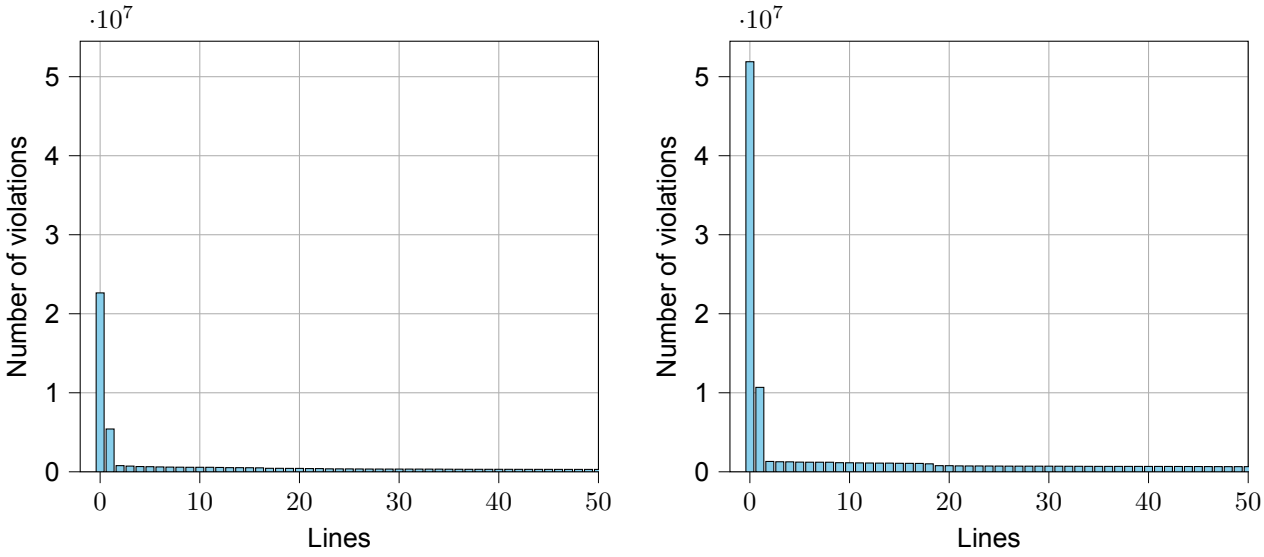
(a) Proposed approach. **(b)** Baseline.

Figure A.14: Number of violations per line, in descending order, for the 118-bus system N-2.



(a) Proposed approach. **(b)** Baseline.

Figure A.15: 118-bus violations for the N-3 case.



(a) Proposed approach. **(b)** Baseline.

Figure A.16: Number of violations per line, in descending order, for the 118-bus system N-3.

B

Tables

Table B.1: Fixed training parameters sensitivity analysis

		N-1	N-2	N-3
MLP	Epochs	100	150	200
	Batch size	100	100	100
Loss function	λ_1	0	0	0
	λ_2	1,000	100	10
	λ_3	0	0	0

Table B.2: $\mathbf{FLODF}$ sparsity lower bound

k	1	2	3	4	5	6
39-bus sparsity [%]	97.1	94.3	91.7	89.3	87.1	85.1
118-bus sparsity [%]	99.5	98.9	98.4	97.9	97.4	96.9

Table B.3: 39-bus system islanding cases

	N-1	N-2	N-3	N-4	N-5	N-6
Contingency Cases	33	528	5,456	40,920	237,336	1,107,568
Islanding Cases	2	94	1,902	22,623	178,581	999,868
Percentage Islanding [%]	6.06%	17.80%	34.86%	55.29%	75.24%	90.28%

Table B.4: 118-bus system islanding cases

	N-1	N-2	N-3
Contingency Cases	186	17,205	1,055,240
Islanding Cases	9	1,703	159,591
Percentage Islanding [%]	4.84%	9.90%	15.12%

Table B.5: Reduced vs full computational graph comparison for the 39-bus system

Average per epoch		**N-1**	**N-2**	**N-3**	**N-4**	**N-5**
Forward pass [s]	Full Graph	3.39	3.57	5.04	15.37	69.28
	Reduced Graph	3.40	3.58	5.19	16.76	77.87
Memory [GB]	Full Graph	0.0026	0.0148	0.1449	1.0987	3.2241
	Reduced Graph	0.0006	0.0021	0.0506	0.0900	0.0306
Backward pass [s]	Full Graph	2.25	2.40	4.26	17.68	90.81
	Reduced Graph	2.24	2.25	2.47	3.98	11.66

Table B.6: Training parameters approach $\neq$ feasibility restoration layer

		N-1	**N-2**	**N-3**	**N-4**	**N-5**
MLP	Epochs	100	150	200	200	200
	Batch size	100	100	100	100	100
	Time [s]	570	891	1,582	4,413	18,363
Loss function	λ_0	2	5	5	5	5
	λ_1	100	100	100	100	100
	λ_2	1,000	100	10	1	0.1
	λ_3	2,000	2,000	2,000	2,000	2,000

Table B.7: Training parameters feasibility restoration layer

		N-1	**N-2**	**N-3**	**N-4**	**N-5**
MLP	Epochs	100	150	200	200	200
	Batch size	100	100	100	100	100
	Time [s]	762	789	983	4,344	19,663
Loss function	λ_0	2	5	5	5	5
	λ_1	0	0	0	0	0
	λ_2	1,000	500	10	1	0.1
	λ_3	0	0	0	0	0

Table B.8: 39-bus system results proposed method vs CS baseline

		N-1	N-2	N-3	N-4	N-5
Base case violations	SCOPF	0.00%	0.00%	0.00%	0.00%	0.00%
	MLP	0.00%	0.00%	0.00%	0.00%	0.00%
Post-cont violations	# contingencies	33	528	5,456	40,920	237,336
	SCOPF	0.06%	0.74%	3.30%	3.70%	3.51%
	MLP	0.31%	1.33%	2.16%	3.16%	3.11%
Time [s]	SCOPF	60	248	289	302	388
	MLP	5	5	7	20	101
	Increase	x12	x49	x41	x15	x4
Cost [$]	SCOPF	2041	2099	2144	2210	2209
	MLP	2068	2123	2155	2198	2154
	Increase	+1.44%	+1.17%	+0.51%	-0.47%	-2.61%

Table B.9: 39-bus system results proposed method vs H baseline

		N-1	N-2	N-3	N-4	N-5
Base case violations	SCOPF	0.00%	0.00%	0.00%	0.00%	0.00%
	MLP	0.00%	0.00%	0.00%	0.00%	0.00%
Post-cont violations	# contingencies	33	528	5,456	40,920	237,336
	SCOPF	0.01%	0.60%	2.87%	3.40%	3.39%
	MLP	0.25%	1.19%	2.00%	3.00%	3.00%
Time [s]	SCOPF	60	103	145	149	163
	MLP	4	5	7	20	97
	Increase	x15	x21	x21	x8	x2
Cost [$]	SCOPF	2039	2097	2153	2212	2208
	MLP	2067	2123	2154	2197	2154
	Increase	+1.45%	+1.22%	+0.10%	-0.66%	-2.57%

Table B.10: Values of comparison for 39-bus system of violating post-contingency cases for different contingency levels

k	1	2	3	4	5
N-1 model infeasible cases [%]	0.33	5.10	10.80	13.26	10.75
N-2 model infeasible cases [%]	0.34	1.37	2.88	4.05	3.85
N-3 model infeasible cases [%]	0.33	1.08	2.19	3.11	3.03
N-4 model infeasible cases [%]	0.32	1.11	2.24	3.14	3.02
N-5 model infeasible cases [%]	0.34	1.10	2.20	3.13	3.05

Table B.11: Training parameters 118-bus system

		N-1	N-2	N-3
MLP	Epochs	100	100	100
	Batch size	100	100	100
	Training time [s]	18,400	25,200	67,500
Loss function	λ_0	1	1	1
	λ_1	100	100	100
	λ_2	500	5	0.05
	λ_3	10	10	10

Table B.12: 118-bus system results CS baseline

		N-1	N-2	N-3
Base case violations	SCOPF	0.0000%	0.0000%	0.0000%
	MLP	0.0000%	0.0000%	0.0000%
Post-cont violations	# contingencies	186	17,205	1,055,240
	SCOPF	0.9118%	1.0311%	5.8735%
	MLP	1.0774%	1.7028%	2.6370%
Time [s]	SCOPF	1423	1725	2841
	MLP	9	10	104
	Increase	x158	x173	x27
Cost [$]	SCOPF	956	1060	1058
	MLP	980	995	1002
	Increase	+2.51%	-6.13%	-5.29%

Table B.13: 118-bus system results H baseline

		N-1	N-2	N-3
Base case violations	SCOPF	0.0000%	0.0000%	0.0000%
	MLP	0.0000%	0.0000%	0.0000%
Post-cont violations	# contingencies	186	17,205	1,055,240
	SCOPF	0.8581%	1.0129%	4.5189%
	MLP	0.8876%	1.5288%	2.4203%
Time [s]	SCOPF	609	1484	1530
	MLP	8	9	104
	Increase	x76	x165	x15
Cost [$]	SCOPF	936	1045	1036
	MLP	963	978	985
	Increase	+2.88%	-6.41%	-4.92%

Table B.14: Values of comparison for 118-bus system of violating post-contingency cases for different contingency levels

k	1	2	3
N-1 model infeasible cases [%]	1.0774	2.5555	4.3070
N-2 model infeasible cases [%]	0.6425	1.7028	3.0781
N-3 model infeasible cases [%]	0.5425	1.4481	2.6370

Table B.15: 39-bus system results proposed method with GCNN vs CS baseline

		N-1	**N-2**	**N-3**
Base case violations	SCOPF	0.00%	0.00%	0.00%
	GCNN	0.00%	0.00%	0.00%
Post-cont violations	# contingencies	33	528	5,456
	SCOPF	0.03%	1.08%	2.92%
	GCNN	0.73%	1.80%	2.56%
Time [s]	SCOPF	60	248	289
	GCNN	5	5	7
	Increase	x12	x50	x41
Cost [$]	SCOPF	1993	2052	2099
	GCNN	2025	2089	2144
	Increase	+1.61%	+1.80%	+2.14%

Table B.16: GNN comparison of speed and cost increases compared to the CS baseline

	N-1	**N-2**	**N-3**
Speed Increase	x12	x49	x41
Cost Change	+2.41%	+1.80%	+0.57%

Table B.17: 39-bus system results proposed method with GCNN vs H baseline

		N-1	N-2	N-3
Base case violations	SCOPF	0.00%	0.00%	0.00%
	GCNN	0.00%	0.00%	0.00%
Post-cont violations	# contingencies	33	528	5,456
	SCOPF	0.03%	1.22%	2.78%
	GCNN	0.90%	2.09%	2.85%
Time [s]	SCOPF	60	103	145
	GCNN	5	5	7
	Increase	x12	x21	x21
Cost [$]	SCOPF	1998	2056	2112
	GCNN	2029	2093	2147
	Increase	+1.55%	+1.80%	+1.66%

Table B.18: GNN comparison of speed and cost increases compared to the H baseline

	N-1	N-2	N-3
Speed Increase	x15	x21	x21
Cost Change	+2.40%	+1.80%	+0.14%

C

Post-Contingency Flow Computation

Using PTDFs One way of computing the full set of post-contingency flows is using a $\mathbf{PTDF_c}$ matrix for every contingency case. Given the power injections $\Delta\mathbf{P} \in \mathbb{R}^{|\Omega^B|}$, the post-contingency flows using $\mathbf{PTDF_c}$ are computed as follows.

$$\mathbf{F^c} = \mathbf{PTDF_c}\Delta\mathbf{P} \tag{C.1}$$

To compute the $\mathbf{PTDF_c}$ matrix, the row and column corresponding to the line in outage are removed from the susceptance matrix $\mathbf{B} \in \mathbb{R}^{|\Omega^B|\times|\Omega^B|}$ to obtain $\mathbf{B_c}$. Next, the inverse is taken as $\mathbf{X_c} = \mathbf{B_c}^{-1}$ and the $\mathbf{PTDF_c}$ becomes.

$$\mathbf{PTDF_c} = \mathbf{B_{br}} \times \mathbf{A} \times \mathbf{X_c} \tag{C.2}$$

The relatively large $\mathbf{B_c}$ matrix is different for each contingency case, meaning it has to be inverted for each case posing a computational challenge. By stacking individual $\mathbf{PTDF_c}$ matrices, a $\mathbf{FPTDF_c}$ tensor (Full $\mathbf{PTDF_c}$) of size $\mathbb{R}^{|\Omega^L|\times|\Omega^c|\times|\Omega^B|}$ is constructed. To reduce the memory footprint when many contingencies have to be considered, TTD is used. First constructing the full tensor, and then applying TTD is not possible. Constructing the TT using randomnization is not suitable since it often results in large errors, which is not desirable for this application. Therefore, the $\mathbf{FPTDF_c}$ TT has to be constructed iteratively.

Using LODFs Alternatively, the post-contingency flows can be computed using LODFs as follows.

$$\mathbf{F^c} = \mathbf{F^0} + \mathbf{LODF_{M,O}} \times \mathbf{F^0} \tag{C.3}$$

When many contingency cases have to be considered, the tensor containing all $\mathbf{LODF_{M,O}}$ matrices becomes large. However, this time the matrix becomes sparse making it a good candidate for expressing it in sparse format. The choice is made to construct the $\mathbf{LODF_{M,O}}$ matrix such that it can be used for sparse 2D matrix multiplication. Using the proposed method to compute the LODFs, the matrix that needs to be inverted only has size k by k. This makes the process of computing sensitivity factors much faster.

Comparison PTDFs and LODFs Firstly, using PTDFs and LODFs, the exact same post-contingency flows are obtained, indicating that both methods have the same accuracy when computing post-contingency flows.

Secondly, the construction of the $\mathbf{PTDF_c}$ in TT format was successful for the 39-bus system. When scaling to the 118-bus system, the TT had to be iteratively constructed as described earlier. The TTD

itself and the rounding steps performed were slow, indicating that the $\mathbf{PTDF_c}$ tensor might not be the best candidate for TTD. For the 39-bus system, the construction of the N-3 $\mathbf{FLODF}$ matrix is roughly 3$\times$ as quick as the TT decomposed $\mathbf{PTDF_c}$ tensor, and the method utilizing the $\mathbf{FLODF}$ matrix achieved roughly 7$\times$ more compression than the $\mathbf{PTDF_c}$ TT method. For the 118-bus case, the construction of the N-2 $\mathbf{FLODF}$ matrix is roughly 29$\times$ as quick as the TT decomposed $\mathbf{PTDF_c}$ tensor. The construction of the TT decomposed $\mathbf{PTDF_c}$ tensor for the N-3 case was not successful because it was too slow, making the TTD method intractable for this thesis.

Lastly, computations in sparse format are quick and easy, whereas operations in TT format are still relatively new and unpractical for this application. No practical solution has been found to construct a suitable loss function from TT format. Consequently, the method using LODFs is used in the final framework.

D

Tensor Train Decomposition

The storage complexity of a tensor increases exponentially with the number of dimensions, a problem known as the curse of dimensionality. To mitigate this issue, various tensor decomposition schemes have been developed.

One such scheme is the Tensor Train Decomposition (TTD) [41, 42, 43], which allows tensors to be represented in a more compressed format, reducing computational complexity. TTD decomposes a large tensor into smaller tensors, referred to as cores, with lower ranks. When these cores are arranged sequentially, they resemble a ”train” of tensors, hence the name tensor train. The decomposition process involves iterative steps of unfolding the tensor (matricization) and performing singular value decomposition (SVD) [40] on each unfolded matrix. This iterative process results in a tensor train with N cores, obtained through N-1 steps. During the SVD step, some of the singular values are removed, reducing the rank between the cores. The storage complexity and accuracy depends largely on the retained ranks between the cores. The lower the rank, the lower the storage complexity, but also the lower the accuracy. Considering a tensor with same sized dimensions, the storage complexity is exponential $O(I^N)$, where N is the number of dimensions of the tensor and I is the size of each dimension. With TTD, this storage complexity is reduced to $O(NIR^2)$, where R is the rank between tensor cores.

Figure 2.4 illustrates a standard graphical representation of a tensor train. Each ”leg” corresponds to a dimension, and each circle represents a core. The depicted tensor train consists of 3 cores, i.e., 3 tensors, where the first core is a matrix with two legs, followed by a 3D tensor, and finally another matrix.

Basic operations and TT rounding Once a tensor is transformed to TT format, all operations should also be done in TT format to keep the advantage of the compression. Many basic operations can be easily performed in TT format, among which are addition of TTs, elementwise (Hadamard) product and the computation of the matrix norm. A multiplication between a TT and a matrix is another straightforward and fast operation. Consider the example below, where a 3D tensor expressed in TT format is multiplied with a 2D matrix. The 2D matrix represented in TT format by the letter G in Figure D.1, is simply ’absorbed’ by the second core of the 3D TT. The result is a new TT, where the size of the dimension I_{C2} gets replaced by I_{G2}. In full tensor form, the result is the same as multiplying this 2D matrix with every slice along the second dimension of the 3D tensor.

As a consequence of many of these operations, the ranks between the cores increase. An increase in ranks inherently means an increase in elements and consequently a reduced compression. To mitigate the issue of suboptimal ranks [40] resulting from basic operations, a TT rounding step can be performed. The process of rounding can also be referred to as *truncation*. Similarily to constructing a TT, during a rounding step SVD is applied and some singular values are removed to reduce the rank. The same trade-off between accuracy and complexity has to be made.

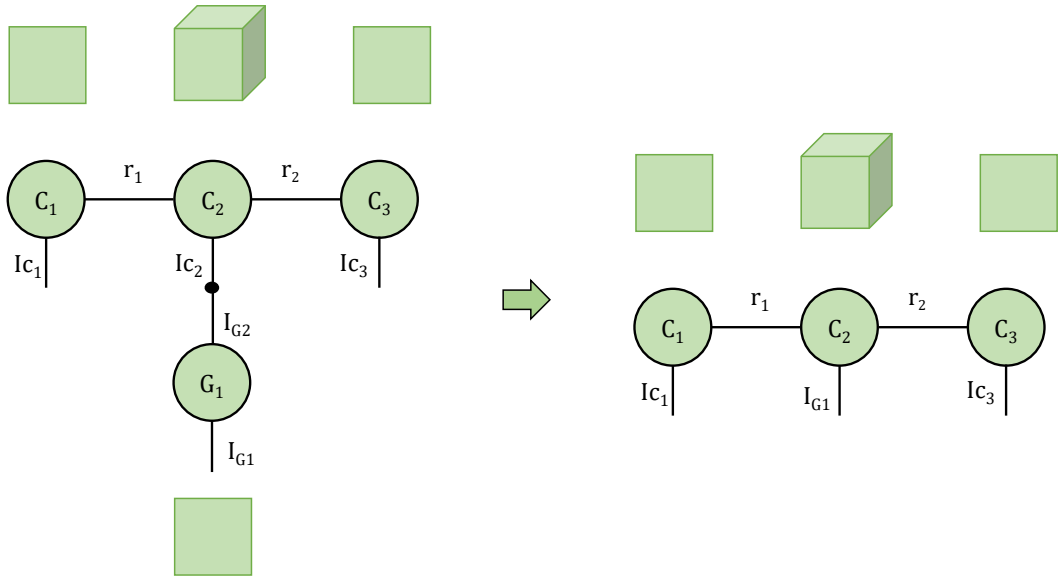


Figure D.1: Visualization of a TT matrix multiplication. The matrix G_1 gets absorbed by the second core C_2, resulting in a new TT. The bottom leg of core C_2 now has dimension I_{G_1}.

TT Construction When extensive tensors are involved which result in memory issues, first constructing the full tensor, and then applying TTD is not possible. In this case, the TT has to be constructed alternatively. Two approaches are briefly explained.

Firstly, the TT can be constructed using randomization. This is a stochastic process to approximate the solution of the TTD. How it exactly works lies outside the scope of this thesis. It's important to note that it would involve initializing the ranks of the TT beforehand without knowing what they should be. Defining the ranks is an iterative process which can be difficult without having any prior knowledge on what they should be.

Another option is to recursively build the TT. This would involve constructing a part of the full tensor, and transforming it into TT. Next, the next part of the tensor is constructed, it is transformed into TT and it is summed to the main TT. After every summation, the rank of the tensor train doubles, so a rounding step has to be performed to recompress the tensor train. During a rounding step some values are truncated and the error increases.

If a 3D tensor is considered, the time it takes to convert the tensor in TT format or to perform a rounding step depends on the structure and the data of the tensor. The more similar the slices of the tensor are, the faster the TTD and rounding is, and also the more compression you achieve with TTD. If you try to decompose a tensor into TT format and the matrices along the third dimension are very different, the TTD will take a long time, and not a lot of compression will be achieved.

E

Sensitivity Analysis

Two different frameworks are designed with different implementations of the CVXPYlayer.

Feasibility Restoration Layer The first framework incorporates the CVXPYlayer as a feasibility restoration layer. In this method, a NN maps the loads to generator setpoints ($\widehat{P}_G$). When the NN prediction lies beyond the feasibility boundaries constrained by the DCOPF equations, a restoration layer is employed to realign it within the feasible region. The convex DCOPF enables the utilization of a CVXPYlayer [34], a convex solver that computes the gradients through the solution of the optimization. The restoration layer minimizes the L2 norm between the predicted setpoints and the setpoints adhering to the feasible region. The restoration layer outputs the restored generator setpoints, along with other variables in the optimization problem, including phase angles ($\boldsymbol{\delta}$) and line flows ($\mathbf{F}$). The formulation of the optimization is outlined as follows:

$$\min_{n \in \Omega^G} \sum ||P_{G_n} - \widehat{P}_{G_n}||_2 \tag{E.1a}$$

subject to:

$$\mathbf{B} \cdot \boldsymbol{\delta} = \mathbf{P_G} - \mathbf{P_D} \tag{E.1b}$$

$$F_l^{min} \leq \frac{1}{x_{ij}}(\delta_i - \delta_j) \leq F_l^{max} \quad \forall i, j \in \Omega^B, \forall l \in \Omega^L \tag{E.1c}$$

$$P_{G_n}^{min} \leq P_{G_n} \leq P_{G_n}^{max} \quad \forall n \in \Omega^G \tag{E.1d}$$

Lower Bound Prediction Layer In the second approach, a NN maps the loads to the lower bounds of the generator setpoints. By doing so, the role of the neural network shifts to constraining the generator limits, thereby narrowing down the solution space of the DCOPF. This approach aims to modify the DCOPF to ensure that it only identifies generator setpoints that prevent post-contingency flow violations. The lower bound prediction layer operates within the same constraints as the feasibility restoration layer framework, albeit with a modified objective: instead of seeking the closest proximity to the feasible region, the focus shifts to finding the lowest dispatch cost. The loss function (Eq. (4.3)) employed in the feasibility restoration layer framework is also utilized in the lower bound prediction framework. However, it is good to realize that the objective function of the lower bound prediction layer has undergone a change. While the feasibility restoration layer aims to find a feasible solution, the objective in the lower bound prediction layer is to minimize the dispatch cost. The constraints (E.1b) - (E.1d) are the same as in the feasibility restoration layer method.

$$\min_{n \in \Omega^G} \sum c_{G_n} P_{G_n} \qquad \forall n \in \Omega^G \tag{E.2}$$

Sensitivity Analysis

A sensitivity analysis is conducted to compare the feasibility restoration layer framework and the lower bound prediction layer framework. The analysis focuses on the N-1, N-2, and N-3 cases of the 39-bus system. The penalty terms for post-contingency flow violations (λ_2) are fixed, while the cost penalty (λ_0) is varied from 0 to 1000. The analysis aims to assess the influence of the cost penalty and determine which framework performs best in finding secure and optimal solutions.

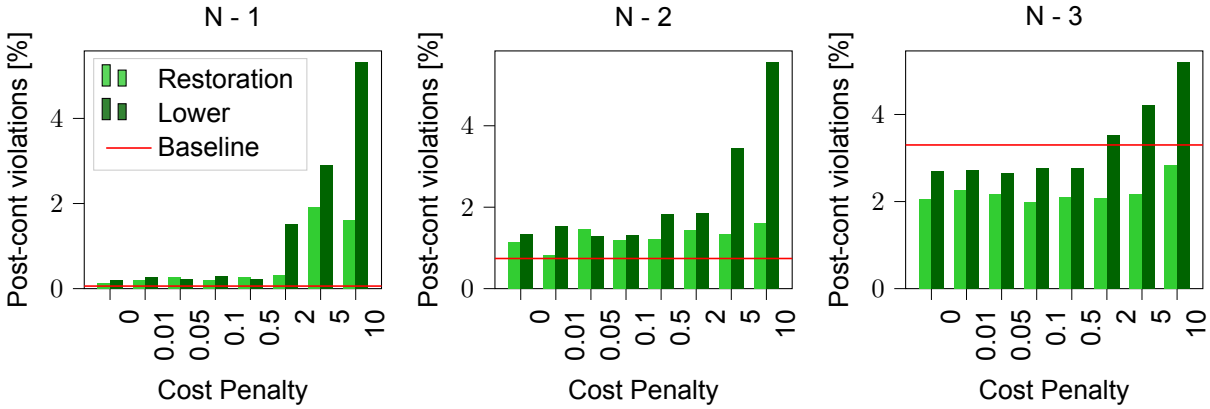


Figure E.1: Comparing the amount of identified violating post-contingency cases for both methods with a varying cost penalty term.

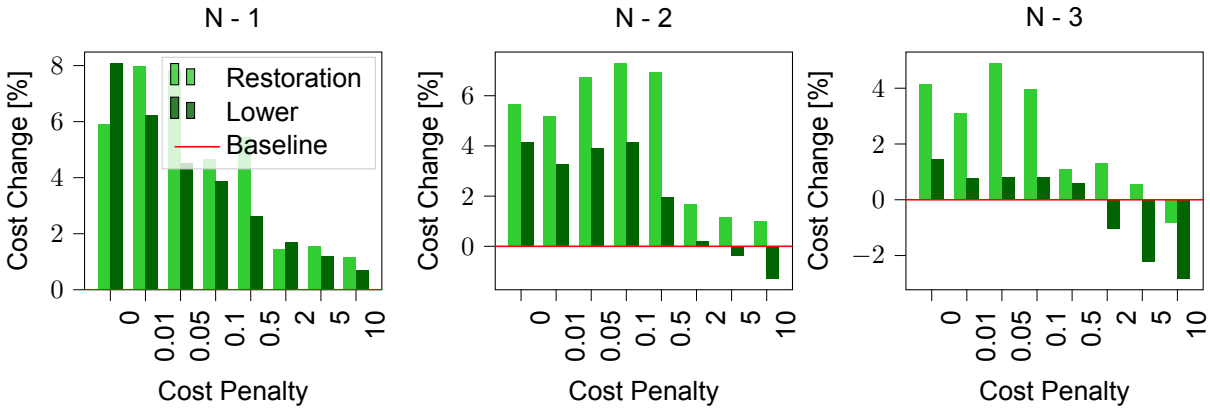


Figure E.2: Comparing the optimality of the dispatch cost for both methods with a varying cost penalty term.

The comparative analysis is presented in Figure E.1 and Figure E.2. Figure E.1 depicts that the lower bound prediction framework almost consistently identifies a higher number of infeasible cases, indicating less secure solutions compared to the feasibility restoration layer framework. On the other hand, Figure E.2 reveals that the lower bound framework often achieves more optimal solutions in terms of dispatch cost, despite its tendency to yield less secure solutions compared to the feasibility restoration framework. Understanding the sequential steps in both frameworks is crucial.

In the lower bound prediction framework, the NN only influences a subset of the generators, namely those operating at their lower bound, determined by the NN. Subsequently, within these bounds, the lower bound prediction layer aims to achieve an optimal cost-based dispatch solution. The focus of the final step in the framework is to find a cost efficient dispatch, and not to identify post-contingency feasible cases. Due to its limited impact on generators, the lower bound prediction framework sets a minimum threshold for identifying infeasible cases. Additionally, when a more secure post-contingency solution is obtained, this dispatch may push certain generators into base case infeasible regions.

In contrast, the feasibility restoration method directly influences all generators. Moreover, this training is guided by a loss function incorporating post-contingency violations and dispatch cost simultaneously. Afterwards, the restoration layer only corrects these generator setpoints to a base case feasible region. Therefore, it has a greater capacity to identify violating post-contingency cases, but achieving an optimal dispatch solution in terms of cost requires careful fine-tuning of the loss function.