LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

INSTITUT FÜR INFORMATIK
LEHRSTUHL FÜR MOBILE UND VERTEILTE SYSTEME

BACHELORARBEIT

A Path Towards Quantum Advantage for the Unit Commitment Problem

David Fischer

LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN
INSTITUT FÜR INFORMATIK
LEHRSTUHL FÜR MOBILE UND VERTEILTE SYSTEME

BACHELORARBEIT

A Path Towards Quantum Advantage for the Unit Commitment Problem

David Fischer

Aufgabensteller:	Prof. Dr. Claudia Linnhoff-Popien Priv.-Doz. Dr. Dirk André Deckert
Betreuer:	Jonas Stein Jago Silberbauer Philipp Altmann
Abgabetermin:	27. August 2024

Hiermit versichere ich, dass ich die vorliegende Bachelorarbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

München, den 27. August 2024

(Unterschrift des Kandidaten)

Abstract

Diese Arbeit stellt eine Lösung für das Unit-Commitment-Problem (UCP) im Bereich des Energienetzmanagements vor. Dabei handelt es sich um ein Optimierungsproblem, bei dem ein Gleichungssystem gelöst wird, um die Kosten für eine gegebene Lösung zu berechnen. Wir charakterisieren das UCP als ein Mixed-Integer Nonlinear Programming (MINLP)-Problem und lösen es mit Hilfe eines Quantensimulations-basierten Optimierungsansatzes (QuSO), wobei dieser eine Klasse von äquivalenten Problemen definiert, die mit dem vorgeschlagenen Algorithmus lösbar sind. Durch die Modellierung des Energienetzes in einem speziellen Graph erhalten wir hilfreiche Erkenntnisse über die Struktur und die Eigenschaften der Suszeptanzmatrix. Wir nutzen approximative Randbedingungen für den Gleichstrom (DC) in diesem Modell. Die vorgeschlagene Quantenroutine beginnt mit der Invertierung der reduzierten Suszeptanzmatrix mittels Quantum Singular Value Transformation (QSVT) unter Verwendung eines speziellen Matrixinversionspolynoms. Eine Quantum Phase Estimation Routine wird zusammen mit einem zusätzlichen QSVT Verfahren verwendet, um die Kostenfunktion zu konstruieren, die dann mit dem Quantum Approximate Optimization Algorithm (QAOA) optimiert wird. Dieser hybride quantenklassische Ansatz nutzt das Potenzial quantenmechanischer Verfahren, um die Effizienz bei der Lösung komplexer Optimierungsprobleme erheblich zu verbessern. Im Rahmen unserer Analyse bewerten wir die algorithmische Komplexität und demonstrieren das beachtliche Potenzial dieses Ansatzes zur Lösung von QuSO-Problemen. Besonders hervorzuheben ist, dass die QSVT-basierte Matrixinversion die Zeitkomplexität in Fällen exponentiell verringern kann, in denen klassische Methoden schlecht mit der Größe des Problems skalieren. Diese Reduktion der Komplexität könnte die Echtzeitoptimierung großer Energienetze ermöglichen und dadurch sowohl die betriebliche Effizienz als auch die Zuverlässigkeit signifikant steigern.

Abstract

This work presents a solution to the unit commitment problem (UCP) in energy grid management, an optimization problem that involves solving a system of equations to calculate costs for a given solution. We characterize the UCP as a Mixed-Integer Nonlinear Programming (MINLP) problem and solve it using a quantum simulation-based optimization (QuSO) approach, defining a class of equivalent problems solvable with the proposed algorithm. By modeling the energy grid as a specific graph, we gain valuable insights into the structure and characteristics of the susceptance matrix. We also incorporate approximate Direct Current (DC) power flow constraints into the model. The proposed quantum routine begins by inverting the reduced susceptance matrix via Quantum Singular Value Transformation (QSVT) using a specific matrix inversion polynomial. A quantum phase estimation routine, along with an additional QSVT procedure, is used to construct the cost function, which is then optimized using the Quantum Approximate Optimization Algorithm (QAOA). This hybrid quantum-classical approach leverages the computational power of quantum algorithms to enhance efficiency in solving such optimization problems. Our results evaluate the algorithm's complexity and demonstrate its significant potential for QuSO problems. Specifically, the QSVT matrix inversion can reduce time complexity exponentially in scenarios where classical methods scale poorly with problem size. This reduction in complexity could enable real-time optimization of large-scale energy grids, thereby improving operational efficiency and reliability.

Contents

1 Introduction

In recent years, the global energy landscape has undergone significant transformations, driven by the increasing demand for electricity, the integration of renewable energy sources, and the growing complexity of energy grids [Int23]. As countries work to achieve ambitious climate goals, upgrading and expanding energy grids has become increasingly important. These grids are not only getting larger but are also becoming more interconnected and reliant on diverse energy sources. The challenge of managing such complex systems has led to a critical examination of existing optimization techniques, particularly in addressing the Unit Commitment Problem (UCP), which lies at the heart of efficient energy grid management.

The UCP is a fundamental problem in the operation of power systems, where the objective is to determine the optimal operational state of power generation units to meet predicted energy demand while minimizing operational costs. This problem is inherently combinatorial and NP-hard [KM78], especially when accounting for the transmission costs involved in energy distribution. As a result, the computational challenge increases significantly with the size and complexity of modern energy grids. Generally, this problem is approached using Mixed-Integer Nonlinear Programming (MINLP), which, although effective in certain scenarios, struggles to scale efficiently [Urb18]. Moreover, these classical approaches often stumble when confronted with the nonlinear, high-dimensional nature of the UCP, especially under the constraints imposed by the integration of renewable energy sources, whose variability and unpredictability further complicate the optimization process [MER21]. An additional challenge in solving the UCP lies in the requirement to solve large-dimensional systems of linear equations (SLEs), which adds another layer of complexity. Classical methods, while robust, have shown limitations in handling these SLEs efficiently, particularly as grid sizes expand [HZZG22]. This challenge underscores the need for novel computational techniques that can manage such high-dimensional problems more effectively [SR+22].

The growing energy crisis, characterized by the volatility of fossil fuel prices, geopolitical tensions, and the increasing demands on energy systems, has exposed the limitations of conventional optimization strategies based on classical computing methods. These traditional approaches are increasingly inadequate for addressing the UCP, as they struggle to keep pace with the complexity of recent energy grids and the intricate requirements of modern power systems [P+21].

In response, quantum computing emerges as a promising alternative, offering the potential for significant speedup in solving certain classes of problems, including the UCP. By leveraging principles such as superposition, entanglement, and quantum interference, quantum algorithms can explore vast solution spaces more efficiently than classical algorithms [NC10]. This is particularly relevant for solving high-dimensional SLEs, which

are central to the UCP. The potential of quantum algorithms, such as the Quantum Approximate Optimization Algorithm (QAOA) [FGG14, FH19], to address these challenges presents a transformative approach to energy grid optimization. Moreover, these techniques could be extended to other optimization problems involving SLEs, broadening the impact of quantum computing in energy management.

1.1 Optimizing Electricity Systems

Given the critical role of optimization in managing modern energy grids, it is essential to understand the foundational problems that must be addressed. The following provides an explanation of the Unit Commitment Problem (UCP), alongside Economic Dispatch (ED) and Optimal Power Flow (OPF)—two closely related challenges that are integral to understanding the complexities of UCP. By understanding these problems and their complexities, we can better appreciate the significance of the proposed quantum approach in tackling these issues. We will provide a brief introduction, the mathematical definition of the UCP we are using is given in Section 4.1.

1.1.1 Introduction to Economic Dispatch

Economic Dispatch (ED) focuses on allocating generation resources to meet electricity demand at the lowest possible operational cost. It calculates the optimal power output $P_{G_i} \in \mathbb{R}$ from each generator G_i, assuming the cost of operation $c_i > 0$ as the primary objective. However, this model considers only the generation side and does not account for physical constraints of the power transmission network, such as line capacities and voltage levels, which are important limiting factors in reality. This simplification allows for rapid decision-making in real-time operations. ED also does not consider the unit start-up or shut-down processes and treats power generation capacity as a continuous variable [GSO17]. Assuming an energy grid with a set of $n \in \mathbb{N}$ connected generators $G_1, G_2, \ldots, G_n$, we can express an ED problem as follows:

$$\min_{P_{G_i}} \quad \sum_{i=1}^{n} c_i P_{G_i} \tag{1.1}$$

$$\text{subject to:} \tag{1.2}$$

$$P_{G_i}^{\min} \leq P_{G_i} \leq P_{G_i}^{\max} \quad \forall\, 1 \leq i \leq n \tag{1.3}$$

$$\sum_{i=1}^{n} P_{G_i} = P_D \tag{1.4}$$

where $c_i > 0$ is the cost per unit of power injected (usually in MWh) and $P_{G_i} \in \mathbb{R}$ is the power produced by generator G_i. The constraints include the equality constraint (Eq. 1.4) that ensures power and load balance, with $P_D > 0$ expressing the predicted power demand. The inequality constraint (Eq. 1.3) bounds the generated power by its minimum $P_{G_i}^{\min} > 0$ and maximum $P_{G_i}^{\max} > P_{G_i}^{\min}$ capacity for a given generator G_i.

We can express an ED in its linear form using:

$$\begin{aligned} \max_{x} \quad & c^T x \\ \text{s.t.} \quad & Ax \leq b \\ & x \geq 0 \end{aligned}$$

where we can use the matrix $A \in \mathbb{R}^{(2n+2)\times n}$ and vectors $b \in \mathbb{R}^{2n+2}$, $x \in \mathbb{R}^n$ as follows:

$$A = \begin{pmatrix} -1 & 0 & \cdots & 0 \\ 0 & -1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & -1 \\ 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \\ 1 & 1 & \cdots & 1 \\ -1 & -1 & \cdots & -1 \end{pmatrix}, \qquad b = \begin{pmatrix} -P_{G_1}^{\min} \\ -P_{G_2}^{\min} \\ \vdots \\ -P_{G_n}^{\min} \\ P_{G_1}^{\max} \\ P_{G_2}^{\max} \\ \vdots \\ P_{G_n}^{\max} \\ P_D \\ -P_D \end{pmatrix}, \qquad x = \begin{pmatrix} P_{G_1} \\ P_{G_2} \\ \vdots \\ P_{G_n} \end{pmatrix}$$

We will later further characterize these sets of feasible decision variables. Since this problem possesses all characteristics of a linear problem, it can be solved by the simplex algorithm. Conventionally, ED has been evaluated using methods including Lagrangian relaxation and gradient-based methods [CR90]. Newer, more enhanced methods include evolutionary algorithms, with particle swarm optimization proposed by [Gai03]. More sophisticated models, such as the following, incorporate some aspects of ED but take power flow into consideration instead of only economic dispatch.

1.1.2 Overview of Optimal Power Flow

Optimal Power Flow (OPF) represents a more sophisticated and comprehensive model for optimizing power system operations by incorporating the physical and engineering constraints of the power transmission network into the optimization process. OPF extends beyond simpler models, such as Economic Dispatch, by not only focusing on the allocation of generation resources but also accounting for the complex interrelationships within the power grid. This includes variables such as generator outputs, transmission line flows, and voltage levels across the network. OPF aims to balance economic efficiency with the reliable supply of electricity, ensuring that generation meets demand while adhering to the physical limitations of the power grid. It also accounts for temporal variations in demand, generation costs, and network transmission constraints, making it a more dynamic and realistic approach to power system optimization. By combining the principles of economic dispatch with power flow calculations, OPF optimizes the generation and distribution of electricity without exceeding the loadability limits of transmission lines. This approach ensures that power systems operate efficiently, even under varying conditions and constraints [GSO17].

The foundational work by Carpentier (1962) [Car62] laid the groundwork for OPF, with

additional developments by Huneault and Galiana (1991) [HG91]. The OPF problem can be further separated into two classes differentiating between an AC-OPF, which uses the full and untransformed AC power flow equations, and a DC-OPF, which uses a linear approximation of the AC power flow equations. Some microgrids, particularly those utilizing renewable energy sources like solar panels that generate DC power, may employ DC for local distribution [PPC23, AI21]. Consequently, the DC-OPF model can operate without relying on approximations, providing precise and effective optimization for power distribution. For our purposes, we will make use of the DC-OPF with the approximated AC power flow.

We can transform the Economic Dispatch problem into Optimal Power Flow by adding more constraints. Consider a n-bus system. To limit flow, the DC power flow approximations can be employed, expressed as Eq. 1.8 (see Section 2.3.1). This flow constraint ensures load balance at each bus through the inclusion of Eq. 1.9, where $B \in \mathbb{R}^{n \times n}$ represents the susceptance matrix (see Section 2.3). This equality constraint establishes the relationship between power and flow. The DC-OPF can then be formulated as follows:

$$\min_{P_{G_i}, \theta} \quad \sum_{i=1}^{n} c_i P_{G_i} \tag{1.5}$$

$$\text{subject to:} \tag{1.6}$$

$$P_{G_i}^{\min} \leq P_{G_i} \leq P_{G_i}^{\max} \quad \forall\, 1 \leq i \leq n \tag{1.7}$$

$$\left| \frac{(\theta_i - \theta_j)}{x_{ij}} \right| \leq P_{ij,\max} \quad \forall\, 1 \leq i, j \leq n \tag{1.8}$$

$$B \cdot \theta = P_G - P_D \tag{1.9}$$

where the values are given with:

- $\theta_i \in \mathbb{R}$: Phase angle at bus i. The vector $\theta \in \mathbb{R}^n$ includes θ_i for all $1 \leq i \leq n$.
- $x_{ij} \in \mathbb{R}$: Reactance of the transmission line between bus i and bus j.
- $P_{ij,\max} > 0$: Maximum permissible power flow through the transmission line connecting bus i to bus j.
- $P_G \in \mathbb{R}^n$: Vector of generated power at all buses, where $P_{G_i} \in \mathbb{R}$ is the generated power at bus i.
- $P_{G_i}^{\max} > P_{G_i}^{\min} > 0$: Minimum and maximum power generation limits at bus i.
- $P_D \in \mathbb{R}^n$: Vector of power demand at all buses, where $P_{D_i} \in \mathbb{R}$ is the power demand at bus i.
- $c_i > 0$: Cost coefficient for power generation at bus i.
- $B \in \mathbb{R}^{n \times n}$: Susceptance matrix, which represents the susceptance values between buses in the network.

Note that this is just an overview; a rigorous definition of all the values is given in Section 2.3. The constraint Eq. 1.8 ensures that the power flow on each transmission line does not exceed its maximum capacity, $P_{ij,\max}$. All other constraints and the objective function remain consistent with those defined in the Economic Dispatch (ED) problem.

1.1.3 Introduction to the Unit Commitment Problem

The Unit Commitment Problem (UCP) builds upon the ideas of ED and introduces the decision-making process regarding the on/off operational status of generating units over a specific discrete timeframe to meet predicted demand. UCP addresses the binary nature of a unit's operational status and incorporates additional information such as the costs associated with starting up and shutting down generating units. This added layer of decision-making complexity tries to identify the optimal subset of generators that should be operational to achieve minimum operating costs. UCP therefore aims to close the gap between the binary decision-making of unit statuses and the continuous optimization of power generation levels [WWS13]. Incorporating UCP into OPF models enables a detailed and dynamic approach to power system optimization. It considers temporal variations in demand and generation costs, ensuring an economically efficient and reliable electricity supply. This integration highlights the progression from simple ED models to complex OPF formulations, marking significant advancements in optimization techniques to meet the evolving demands of electricity systems.

In this work, we propose a quantum simulation-based optimization (QuSO) approach to tackle the UCP within energy grid management. Our method characterizes the UCP as a Mixed-Integer Nonlinear Programming (MINLP) problem and employs Quantum Singular Value Transformation (QSVT) to invert the reduced susceptance matrix, a key component in our quantum routine. By integrating quantum algorithms into the optimization process, we aim to significantly reduce the time complexity associated with solving the UCP, particularly in scenarios where classical methods scale poorly with problem size. Our approach not only addresses the specific challenges of the UCP but also suggests broader applications for quantum computing in tackling other complex optimization problems across various domains. The structure of this work is as follows:

- **Background**: We begin by providing a comprehensive overview of the foundational concepts, including Mixed-Integer Nonlinear Programming, spectral graph theory, and the mathematical modeling of energy grids. This section also introduces the basics of quantum computing, particularly the algorithms relevant to our approach.
- **Related Work**: Here, we review existing literature on UCP and its various solutions, including classical and quantum approaches.
- **Methodology**: This section details our proposed QuSO approach, including the formulation of the UCP as an optimization problem, the detailed construction of the quantum algorithm, and the integration of quantum and classical components.
- **Results**: We present the results of our approach, including a complexity analysis. Our findings demonstrate the potential of quantum computing to enhance the efficiency and scalability of solving the UCP.
- **Discussion**: We critically analyze the components of our algorithm and the implications of our results, particularly discussing the potential impact of quantum computing on energy grid management and the broader field of optimization.
- **Conclusion**: The work concludes with a summary of our contributions and a discussion of future research directions.

2 Background

2.1 Mixed-Integer Nonlinear Programming

Traditionally, the Unit Commitment Problem (UCP) has been addressed using linear programming techniques that approximate the operational characteristics of power plants [WWS13]. However, the increasing complexity of modern energy systems necessitates a more advanced approach, specifically the use of Mixed-Integer Nonlinear Programming (MINLP). Solving MINLP problems is classified as NP-hard, meaning that there is no algorithm that can optimize every possible MINLP problem within polynomial time constraints [KM78]. The complete formulation of the UCP as an MINLP problem can be found in Section 4.1.

Definition 2.1. *For $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$, a **polyhedral set** P is defined as:*

$$P = \{x \in \mathbb{R}^n : Ax \leq b\}.$$

Such a polyhedral set can be described as the intersection of a finite number of half-spaces $\mathrm{H} = \{x \in \mathbb{R}^n : a^{\mathrm{T}}x \leq b\}$. Each half-space can be defined by a linear inequality of the form:

$$a_1x_1 + a_2x_2 + \cdots + a_nx_n \leq b,$$

where $x_1, x_2, \ldots, x_n$ are unknown variables with coefficients $a_1, a_2, \ldots, a_n \in \mathbb{R}$ and a bound $b \in \mathbb{R}$ that defines the offset of the half-space from the origin.

Definition 2.2. *An optimization problem is called a **Mixed-Integer Nonlinear Programming (MINLP) problem** if it can be expressed in the following form:*

$$\min_x f(x) \tag{2.1a}$$
$$s.t. \quad g(x) \leq 0 \tag{2.1b}$$
$$x \in P \tag{2.1c}$$
$$x_i \in \mathbb{Z} \quad \forall i \in I \tag{2.1d}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ and $g : \mathbb{R}^n \to \mathbb{R}^m$ are algebraic functions. $P \subset \mathbb{R}^n$ is a polyhedral set and I is a finite set of indices.

The goal is to determine an input $x \in \mathbb{R}^n$ that achieves the minimization of $f(x)$, while concurrently satisfying the specified constraints of the polyhedral set. Component (2.1b) characterizes the constraints of the problem, involving various elements of the input vector x, which must satisfy all of these constraints. Subsequently, the polyhedral set (2.1c) adds bounds on the individual elements x_i within the vector x. Component (2.1d) clarifies that certain elements x_i must be integer values, differing from rational numbers as seen in a more general approach by [BKL+13].

2.2 Graph-Theoretical Modeling of Energy Grids

Definition 2.3. *A pair $G = (V, E, w)$ is called a **finite undirected weighted graph**, where V is a set of vertices such that $|V| = n \in \mathbb{N}$,*

$$E \subseteq \{(u, v) \in V \times V \mid u \neq v \wedge ((u, v) \in E \implies (v, u) \in E)\}$$

is the set of edges, and $w : E \to \mathbb{R} \setminus \{0\}$ is a weight function.

Remark 2.1. *Note that with this definition, the graph is undirected, meaning that if $(u, v) \in E$, then $(v, u) \in E$. This implies that each edge $(u, v) \in E$ is bidirectional. Therefore, E can be considered a set of unordered pairs of vertices, i.e., $\{\{u, v\} \mid (u, v) \in E\}$. The weight function w is symmetric, implying that for any $(u, v) \in E$, $w(u, v) = w(v, u)$. The graph does not contain self-loops, meaning that $(v, v) \notin E$ for any $v \in V$.*

While our definition focuses on real weights, which is sufficient since the only purpose will be to investigate the susceptance matrix in our context, this definition can be extended to include complex weights as shown in [BP24]. This extension complicates the estimation of Laplacian properties. For a detailed discussion on the estimation of complex-valued Laplacian matrices, see [HRP24]. To describe a graph in its structure efficiently, we use multiple matrices that describe connectivity between edges.

Definition 2.4. *Let $G = (V, E, w)$ with $|V| = n$ be a finite undirected weighted graph. The matrix $A_G \in \mathbb{R}^{n \times n}$ used to represent a finite graph is called the **adjacency matrix**.*

$$A_{ij} = \begin{cases} w(i, j) & \text{if } (i, j) \in E \\ 0 & \text{otherwise} \end{cases}$$

Definition 2.5. *Let $G = (V, E, w)$ with $|V| = n$ be a finite undirected weighted graph. The graph G is said to be **connected** if for any two vertices $u, v \in V$, there exists a sequence of vertices $(v_0 = u, v_1, v_2, \ldots, v_k = v)$ such that each pair $(v_i, v_{i+1}) \in E$.*

Definition 2.6 (Degree and Volume)**.** *Let $G = (V, E, w)$ be a finite undirected weighted graph with $|V| = n$.*

- *The **degree** $d(v)$ of a vertex $v \in V$ is defined as the sum of the weights of the edges incident to v:*
$$d(v) = \sum_{u \in V, (v,u) \in E} w(v, u).$$
- *The **volume** $\mathrm{vol}(S)$ of a set $S \subseteq V$ is the sum of the degrees of the vertices in S:*
$$\mathrm{vol}(S) = \sum_{w \in S} d(w).$$

Definition 2.7. *Let $G = (V, E, w)$ be a finite undirected weighted graph with $|V| = n$. The **degree matrix** $D_G \in \mathbb{R}^{n \times n}$ represents the weighted degrees of the vertices in G. The degree matrix D_G is given by:*

$$(D_G)_{ij} = \begin{cases} d(v_i) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases},$$

where $d(v_i)$ is the weighted degree of vertex $v_i \in V$.

Note that the definition above is the most commonly used one and often found in the literature, as in [Die17]. However, in many practical use cases, it can be highly dependent on the specific scenario in which the graph and especially its weights are being used, and therefore can be defined differently as used later on (see Remark 2.11). We can now combine the degree and adjacency matrix and express all our information through just one single matrix which is defined as follows:

Definition 2.8. *Let $G = (V, E, w)$ with $|V| = n$ be a finite undirected weighted graph, and $A_G \in \mathbb{R}^{n\times n}$ the adjacency matrix and $D_G \in \mathbb{R}^{n\times n}$ the degree matrix. The matrix $L_G \in \mathbb{R}^{n\times n}$ is called the* ***Laplacian matrix****:*

$$L_G = D_G - A_G$$

Remark 2.2 (Symmetry of Laplacian)**.** *The Laplacian matrix L_G of a finite undirected weighted graph $G = (V, E, w)$ with $|V| = n$ is symmetric. This is because $L_G = D_G - A_G$, where D_G is a diagonal matrix, hence symmetric. The adjacency matrix A_G is symmetric due to the undirected nature of the graph. Therefore, for all $i, j \in \{1, \dots, n\}$, $A_{ij} = A_{ji}$. Hence, $L_{ij} = D_{ij} - A_{ij} = L_{ji} = D_{ji} - A_{ji}$ for all $i, j \in \{1, \dots, n\}$.*

Remark 2.3 (Reality of Eigenvalues)**.** *The eigenvalues of the Laplacian matrix L_G of a finite undirected weighted graph $G = (V, E, w)$ are real. This follows from the fact that L_G is a symmetric matrix, and symmetric matrices have real eigenvalues.*

Lemma 2.1. *Let $G = (V, E, w)$ with $|V| = n$ be a finite undirected weighted graph. If $w : E \to \mathbb{R} \setminus (-\infty, 0]$, then the Laplacian matrix $L_G \in \mathbb{R}^{n\times n}$ of G is positive semidefinite.*

Proof. Let $G = (V, E, w)$ with $|V| = n$ be a finite undirected weighted graph with weight function $w : E \to \mathbb{R} \setminus (-\infty, 0]$, i.e., $w(e) > 0$ for all $e \in E$ and $L_G, D_G, A_G \in \mathbb{R}^{n\times n}$ the Laplacian, degree, and adjacency matrices of G. We consider the quadratic form of each matrix for $x \in \mathbb{R}^n$:

$$x^T L_G x = x^T (D_G - A_G)x = x^T D_G x - x^T A_G x$$

From the definition of D_G with the diagonal $(D_G)_{ii} = d(v_i) = \sum_{(i,j)\in E} w(i, j)$ it is:

$$x^T D_G x = \sum_{i=1}^{n} (D_G)_{ii} x_i^2 = \sum_{i=1}^{n} d(v_i) x_i^2 = \sum_{(i,j)\in E} w(i, j)(x_i^2 + x_j^2)$$

Since in an undirected graph, every edge (i, j) is bidirectional, meaning if $(i, j) \in E$, then $(j, i) \in E$ with the same weight $w(i, j) = w(j, i)$, for A_G we have:

$$\begin{aligned} x^T A_G x = \sum_{i=1}^{n}\sum_{j=1}^{n} A_{ij} x_i x_j &= \sum_{(i,j)\in E} w(i, j) x_i x_j + \sum_{(j,i)\in E} w(j, i) x_j x_i \\ &= \sum_{(i,j)\in E} w(i, j) x_i x_j + \sum_{(i,j)\in E} w(i, j) x_j x_i \\ &= \sum_{(i,j)\in E} 2w(i, j) x_i x_j \end{aligned}$$

Thus, we can rewrite the complete Laplacian L_G as follows:

$$x^T L_G x = \sum_{(i,j)\in E} w(i,j)(x_i^2 + x_j^2) + \sum_{(i,j)\in E} 2w(i,j)x_i x_j = \sum_{(i,j)\in E} w(i,j)(x_i^2 + 2x_i x_j + x_j^2)$$

Since $w(i,j) > 0$ and $(x_i - x_j)^2 \geq 0$ for all $(i,j) \in E$, we have:

$$x^T L_G x = x^T (D_G - A_G)x = \sum_{(i,j)\in E} w(i,j)(x_i - x_j)^2 \geq 0.$$

□

Corollary 2.1. *Let $G = (V, E, w)$ with $|V| = n$ be a finite undirected weighted graph. If $w : E \to \mathbb{R} \setminus (-\infty, 0]$, then the Laplacian matrix $L_G \in \mathbb{R}^{n\times n}$ of G has non-negative eigenvalues.*

Remark 2.4. *Note that we will be formulating our results for positive weights since, in the context of electrical networks, these weights, given as susceptances, are usually positive. In practical power grids, capacitive elements are more common for improving power quality and managing voltage stability, leading to positive susceptance values being typical in these systems [CSSDS+15, CK24]. However, the above result can be extended to graphs with negative weights under certain conditions, as detailed in [CKG16]. Specifically, for a connected graph $G = (V, E, W)$, the Laplacian matrix is positive semidefinite if and only if the following conditions are satisfied:*

- *The absolute values of the negative edge weights $|W(i,j)|$ are less than or equal to the reciprocals of the corresponding effective resistances $W_{ij}(G+)$ for all $(i,j) \in E^-$.*
- *There must be no cycle in G containing two or more edges with negative weights.*

In their paper, W denotes a diagonal matrix from $\mathbb{R}^{m\times m}$, where m is the number of edges in the graph, and each diagonal entry w_{kk} represents the weight of the edge ϵ_k. The positive and negative edges are separated into $E_+, E_- \subseteq E$; thus, $E_+ \cup E_- = E$ and $G_+ = (V, E_+, W_+)$ and $G_- = (V, E_-, W_-)$ are considered.
Formally, these conditions are stated in Theorem 3.2 and Lemma 3.3 of [CKG16].

The following theorem is supported by Lemma 3.1.1 and the discussions in Chapter 3 of [Spi19], where it is established that the Laplacian matrix of a connected graph has a unique zero eigenvalue with the eigenvector being the all-ones vector.

Theorem 2.1. *Let $G = (V, E, w)$ with $|V| = n$ be a finite undirected weighted graph. If $w : E \to \mathbb{R} \setminus (-\infty, 0]$ and G is connected, then the Laplacian matrix $L_G \in \mathbb{R}^{n\times n}$ of G has exactly one zero eigenvalue, and the corresponding eigenvector is $\mathbf{1} := (1, 1, \ldots, 1)^T \in \mathbb{R}^n$.*

Proof. Let $G = (V, E, w)$ with $|V| = n$ be a finite undirected weighted connected graph with weight function $w : E \to \mathbb{R} \setminus (-\infty, 0]$ and $L_G, D_G, A_G \in \mathbb{R}^{n\times n}$ the Laplacian, degree, and adjacency matrices of G. Let $\mathbf{0} := (0, 0, \ldots, 0)^T \in \mathbb{R}^n$ and $\mathbf{1} := (1, 1, \ldots, 1)^T \in \mathbb{R}^n$. We need to show that $L_G\mathbf{1} = \mathbf{0}$. Consider the i-th entry of the product $L_G\mathbf{1}$:

$$(L_G\mathbf{1})_i = \sum_{j=1}^{n} (L_G)_{ij} \cdot 1_j = \sum_{j=1}^{n} (D_G - A_G)_{ij} \cdot 1_j = \sum_{j=1}^{n} D_{ij} \cdot 1_j - \sum_{j=1}^{n} A_{ij} \cdot 1_j.$$

Since D_G is a diagonal matrix, $D_{ij} = d(v_i)$ if $i = j$ and 0 otherwise:

$$(L_G\mathbf{1})_i = D_{ii} \cdot 1_i - \sum_{j=1}^{n} A_{ij} \cdot 1_j = d(v_i) - \sum_{j=1}^{n} A_{ij}.$$

Since the sum of the weights of the edges incident to v_i is given by $d(v_i)$:

$$(L_G\mathbf{1})_i = d(v_i) - \sum_{j=1}^{n} A_{ij} = d(v_i) - \sum_{v_j \in V, (v_i,v_j)\in E} w(v_i, v_j) = d(v_i) - d(v_i) = 0.$$

Hence, $L_G\mathbf{1} = \mathbf{0}$ and therefore we have that 0 is an eigenvalue of L_G and $\mathbf{1}$ is a corresponding eigenvector. We now conclude that this is the only zero eigenvalue. For any vector $\mathbf{x} \in \mathbb{R}^n$ with $\mathbf{x} \neq \mathbf{0}$ such that $\mathbf{x}^T\mathbf{1} = 0$, we show that $\mathbf{x}^T L_G\mathbf{x} > 0$.
By definition, G is connected if for any two vertices $u, v \in V$, there exists a sequence of vertices $(v_0 = u, v_1, v_2, \ldots, v_k = v)$ such that each pair $(v_i, v_{i+1}) \in E$.
Consider the quadratic form $\mathbf{x}^T L_G\mathbf{x}$ for any vector $\mathbf{x} \in \mathbb{R}^n$:

$$\mathbf{x}^T L_G\mathbf{x} = \sum_{(u,v)\in E} w(u, v)(x_u - x_v)^2.$$

Since $w(u, v) > 0$, $(x_u - x_v)^2 \geq 0$, and equality holds if and only if $x_u = x_v$ for all edges $(u, v) \in E$. Therefore,

$$\mathbf{x}^T L_G\mathbf{x} = 0 \quad \text{if and only if} \quad x_u = x_v \quad \forall u, v \in V.$$

This implies that if $\mathbf{x}^T L_G\mathbf{x} = 0$, then $x_u = x_v$ for all vertices u and v in the same connected component. Since G is connected, this means $x_u = x_v$ for all $u, v \in V$, implying that $\mathbf{x}$ is a scalar multiple of $\mathbf{1}$. Thus, $\mathbf{x}^T L_G\mathbf{x} > 0$ for any vector $\mathbf{x} \in \mathbb{R}^n$ with $\mathbf{x} \neq \mathbf{0}$ orthogonal to the one vector $\mathbf{x}^T\mathbf{1} = 0$. Hence, L_G has exactly one zero eigenvalue with $\mathbf{1}$ as its eigenvector. □

Corollary 2.2. *Let $G = (V, E, w)$ with $|V| = n$ be a finite undirected weighted connected graph with $w : E \to \mathbb{R}\setminus(-\infty, 0]$. By excluding the row and column from L_G corresponding to the zero eigenvalue, the resulting matrix L'_G is positive definite. Hence, all eigenvalues of L'_G are strictly positive, making L'_G invertible. This property is used to construct the invertible susceptance matrix.*

Proof. We assume that the zero eigenvalue corresponds to the first row and column of L_G without loss of generality. The matrix L'_G is obtained by removing the first row and column of L_G. Let $0 \neq \mathbf{y} = (y_1, y_2, \ldots, y_{n-1}) \in \mathbb{R}^{n-1}$. We extend $\mathbf{y}$ to a vector $\mathbf{x} \in \mathbb{R}^n$ by setting $x_1 = -\sum_{i=1}^{n-1} y_i$ and $x_{i+1} = y_i$ for $i = 1, 2, \ldots, n-1$. Thus, $\mathbf{x}$ is constructed such that:

$$\mathbf{1}^T\mathbf{x} = x_1 + \sum_{i=1}^{n-1} x_{i+1} = -\sum_{i=1}^{n-1} y_i + \sum_{i=1}^{n-1} y_i = 0,$$

meaning $\mathbf{x}$ is orthogonal to $\mathbf{1}$. Since $\mathbf{x}$ is orthogonal to $\mathbf{1}$, and by Theorem 2.1, we have:

$$\mathbf{x}^T L_G\mathbf{x} > 0.$$

The quadratic form $\mathbf{x}^T L_G \mathbf{x}$ can be written in terms of the submatrix L'_G:

$$\mathbf{x}^T L_G \mathbf{x} = \mathbf{y}^T L'_G \mathbf{y}.$$

Therefore, since $\mathbf{y}$ is any non-zero vector in $\mathbb{R}^{n-1}$ and $\mathbf{x}^T L_G \mathbf{x} > 0$ for all non-zero $\mathbf{x}$ orthogonal to $\mathbf{1}$, it follows that:

$$\mathbf{y}^T L'_G \mathbf{y} > 0 \quad \text{for all non-zero} \quad \mathbf{y} \in \mathbb{R}^{n-1}.$$

This implies that L'_G is positive definite and therefore invertible. □

To solve the Unit Commitment Problem (UCP) with a DC power flow constraint, we utilize a technique involving the reduced susceptance matrix (see Section 2.3.2). We then invert this matrix using Quantum Singular Value Transformation (QSVT). For a finite, undirected, weighted, and connected graph, the second smallest eigenvalue of its Laplacian L_G is the smallest non-zero eigenvalue. This property also applies to the reduced Laplacian L'_G, as demonstrated in Corollary 2.2, and consequently to the reduced susceptance matrix, which is crucial for our quantum inversion method. The second smallest eigenvalue of the Laplacian L_G is particularly important because it influences the condition number of the reduced susceptance matrix, thereby affecting the complexity of our quantum algorithm. Therefore, understanding this eigenvalue is essential for our approach. The second smallest eigenvalue λ_2 of the Laplacian matrix L_G is known as the algebraic connectivity or Fiedler value [Fie73].

Definition 2.9 (Algebraic connectivity)**.** *Let $G = (V, E, w)$ with $|V| = n$ be a finite undirected weighted connected graph with $w : E \to \mathbb{R} \setminus (-\infty, 0]$ and $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$ be the eigenvalues of the Laplacian L_G, then λ_2 is called the* ***algebraic connectivity*** *of G.*

Remark 2.5. *The requirement for the graph G to be connected is crucial in defining algebraic connectivity, as the second smallest eigenvalue λ_2 of the Laplacian matrix L_G is positive only if the graph is connected. This follows from Theorem 2.1. If G is not connected, λ_2 would be zero, rendering the concept of algebraic connectivity meaningless.*

2.2.1 Condition Number Analysis for Laplacians

Definition 2.10. *Let $A \in \mathbb{R}^{n\times n}$ be an invertible matrix. The* ***condition number*** *$\kappa(A)$ of A with respect to a given matrix norm $\|\cdot\|$ is defined as:*

$$\kappa(A) = \|A\|\|A^{-1}\|,$$

where $\|A\|$ denotes the induced norm of the matrix A, and $\|A^{-1}\|$ denotes the induced norm of the inverse of A. For a given vector norm $\|\cdot\|_v$, the induced norm $\|A\|$ is defined by:

$$\|A\| = \sup_{x\neq 0} \frac{\|Ax\|_v}{\|x\|_v}.$$

Definition 2.11 (Normal Matrix)**.** *A matrix $A \in \mathbb{C}^{n\times n}$ is called a* ***normal matrix*** *if*

$$A^*A = AA^*,$$

where A^* denotes the conjugate transpose of A.

With the Spectral Theorem (Theorem 2.5.3 from [HJ13]), a matrix $A \in \mathbb{C}^{n\times n}$ can be characterized as normal if it is unitarily diagonalizable, so there exists a unitary matrix $U \in \mathbb{C}^{n\times n}$ such that

$$A = U\Lambda U^*,$$

where $\Lambda \in \mathbb{C}^{n\times n}$ is a diagonal matrix with the eigenvalues of A on the diagonal, and $U^*U = UU^* = I$, where $I \in \mathbb{R}^{n\times n}$ is the identity matrix. For normal matrices, the spectral norm, induced by the Euclidean vector norm, is commonly used, which simplifies the computation. The spectral norm of A, denoted $\|A\|_2$, is given by the largest singular value of A. For a normal matrix, this is the largest absolute value of its eigenvalues. Similarly, the spectral norm of A^{-1}, $\|A^{-1}\|_2$, is the reciprocal of the smallest absolute value of its eigenvalues. Therefore, the condition number of a normal matrix A can be formulated with respect to the spectral norm.

Definition 2.12 (Condition Number of a Normal Matrix)**.** *Let $A \in \mathbb{R}^{n\times n}$ be an invertible normal matrix. The* ***condition number*** *of A with respect to the spectral norm is defined as:*

$$\kappa(A) = \frac{\max\{|\lambda_i| \mid \lambda_i \in \sigma(A)\}}{\min\{|\lambda_i| \mid \lambda_i \in \sigma(A)\}},$$

where $\sigma(A)$ denotes the spectrum of A, i.e., the set of eigenvalues of A.

Note that the above condition number $\kappa(A)$ is well defined since A is invertible, which ensures that $0 < \min\{|\lambda_i|\}$.

Lemma 2.1. *Let $G = (V, E, w)$ with $|V| = n$ be a finite undirected weighted connected graph with $w : E \to \mathbb{R} \setminus (-\infty, 0]$. The Laplacian matrix L_G is normal.*

Proof. Let $G = (V, E, w)$ with $|V| = n$ be a finite undirected weighted connected graph with $w : E \to \mathbb{R} \setminus (-\infty, 0]$. We already know that L_G is symmetric from Remark 2.2. Thus, it is $L_G^T = L_G$. Note that $L_G L_G^T = L_G L_G = L_G^2$ and $L_G^T L_G = L_G L_G = L_G^2$. Therefore, it follows that $L_G L_G^T = L_G^T L_G = L_G^2$. Hence, L_G is normal. □

Remark 2.6. *Although the Laplacian L_G of a finite undirected weighted connected graph with positive weights satisfies most prerequisites, it is not invertible. However, we use the reduced Laplacian for our analysis (see Corollary 2.2). The condition number for the reduced Laplacian is then given as:*

$$\kappa(L'_G) = \frac{\lambda_n}{\lambda_2},$$

where we assume the eigenvalues $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ are ordered, with λ_2 being the second smallest and λ_n the largest eigenvalue. This relates directly to the eigenvalues of the reduced susceptance matrix and thus the complexity of our quantum algorithm. Therefore, it is crucial to find bounds for the largest eigenvalue λ_n and the smallest non-zero eigenvalue λ_2.

Remark 2.7. *If it is clear from the context which matrix is being referred to, we will denote the condition number simply as κ instead of $\kappa(A)$.*

2.2.1.1 Lower Bound for the Second Smallest Eigenvalue

To ensure the condition number κ of the reduced susceptance matrix remains manageable, it is crucial to investigate the second smallest eigenvalue λ_2 of the Laplacian matrix, as this will be the smallest eigenvalue for the matrix we aim to invert. A higher algebraic connectivity λ_2 implies a lower condition number κ, since $\kappa = \lambda_n/\lambda_2$ (see Remark 2.6). Therefore, we need to establish a lower bound for λ_2 that is greater than or equal to one. We follow the formulation given in [Spi19, Chu97].

Definition 2.13. *Let $A \in \mathbb{R}^{n\times n}$ be a symmetric matrix and let $x \in \mathbb{R}^n$ be a non-zero vector. The **Rayleigh quotient** $R(A,x)$ is defined as:*

$$R(A,x) = \frac{x^T A x}{x^T x}.$$

Lemma 2.2 (Rayleigh Quotient Lemma for Laplacians)**.** *Let $G = (V,E,w)$ with $|V| = n$ be a finite undirected weighted connected graph with $w : E \to \mathbb{R}\setminus(-\infty,0]$. The algebraic connectivity λ_2 of L_G can be characterized using the Rayleigh quotient as follows:*

$$\lambda_2 = \min_{\substack{x\in\mathbb{R}^n \\ x\perp \mathbf{1}}} R(L_G,x).$$

Proof. Let $G = (V,E,w)$ with $|V| = n$ be a finite undirected weighted connected graph with $w : E \to \mathbb{R}\setminus(-\infty,0]$ and L_G be the Laplacian of G. Since L_G is symmetric (Remark 2.2), it has a complete set of orthonormal eigenvectors $v_1, v_2, \ldots, v_n$ corresponding to real eigenvalues $\lambda_1 \le \lambda_2 \le \cdots \le \lambda_n$. The eigenvectors form an orthonormal basis for $\mathbb{R}^n$, meaning a vector $0 \neq x \in \mathbb{R}^n$ can be expressed as a linear combination of the eigenvectors:

$$x = \sum_{i=1}^{n} \alpha_i v_i \quad \text{where} \quad \alpha_i = v_i^T x.$$

Therefore, we can construct the Rayleigh quotient $R(L_G,x)$ as follows:

$$x^T L_G x = \left(\sum_{i=1}^{n} \alpha_i v_i\right)^T L_G \left(\sum_{i=1}^{n} \alpha_i v_i\right) = \sum_{i=1}^{n} \alpha_i^2 v_i^T L_G v_i = \sum_{i=1}^{n} \alpha_i^2 \lambda_i,$$
$$x^T x = \left(\sum_{i=1}^{n} \alpha_i v_i\right)^T \left(\sum_{i=1}^{n} \alpha_i v_i\right) = \sum_{i=1}^{n} \alpha_i^2 v_i^T v_i = \sum_{i=1}^{n} \alpha_i^2.$$

Thus, the Rayleigh quotient becomes:

$$R(L_G,x) = \frac{x^T L_G x}{x^T x} = \frac{\sum_{i=1}^{n} \alpha_i^2 \lambda_i}{\sum_{i=1}^{n} \alpha_i^2}.$$

To find λ_2, we include the condition that $x \perp \mathbf{1}$, which means x is orthogonal to the eigenvector corresponding to the smallest eigenvalue $\lambda_1 = 0$, since the eigenvector for λ_1 is $\mathbf{1}$ (see Theorem 2.1). This ensures that x is in the subspace orthogonal to the eigenspace associated with λ_1. That means that $\alpha_1 = v_1^T x = 0$. Therefore, $x \in \operatorname{span}\{v_2, v_3, \ldots, v_n\}$.

To minimize $R(L_G, x)$, note that $\lambda_i \geq \lambda_2$ for $i \geq 2$. Therefore:

$$R(L_G, x) \geq \lambda_2,$$

where equality holds when x is an eigenvector corresponding to λ_2. Thus, the minimum value of the Rayleigh quotient for $x \perp \mathbf{1}$ is λ_2, and it is achieved when x is the eigenvector associated with λ_2. Hence,

$$\lambda_2 = \min_{\substack{x \in \mathbb{R}^n \\ x \perp \mathbf{1}}} R(L_G, x).$$

□

Note that the above Lemma also follows as a special case from the more well-known Courant-Fischer Theorem (see Theorem 2.0.1 from [Spi19]).

Definition 2.14 (Weighted Shortest-Path Distance)**.** *Let $G = (V, E, w)$ with $|V| = n$ be a finite undirected weighted connected graph. Let $\mathrm{dist} : V \times V \to \mathbb{R}$ be a metric function on V. We can explicitly define the distance function $\mathrm{dist} : V \times V \to \mathbb{R}$ as the shortest path distance between two vertices:*

$$\mathrm{dist}(i, j) = \min_{P \in \mathcal{P}(i,j)} \sum_{(u,v) \in P} w(u, v),$$

where:

$$\mathcal{P}(i, j) = \{(i = v_0, v_1, v_2, \ldots, v_k = j) \mid (v_{l-1}, v_l) \in E \; \forall \, 1 \leq l \leq k\}$$

is the set of all paths from vertex i to vertex j.

Note that dist defines a metric function on V. Therefore, the pair (V, dist) forms a metric space. In fact, any well-defined metric function defined on the set of vertices together with the set forms a metric space as long as the graph is connected (see [Alo22, Chu97]).

Definition 2.15 (Diameter)**.** *The **diameter** of a finite undirected weighted connected graph $G = (V, E, w)$ is defined as:*

$$\mathrm{diam}(G) = \max_{u,v \in V} \mathrm{dist}(u, v)$$

where $\mathrm{dist}(u, v)$ is the weighted shortest-path distance between vertices u and v.

The following relationship, well known in spectral graph theory, connects the algebraic connectivity λ_2 with the diameter $\mathrm{diam}(G)$ of the graph. We can extend Theorem 4.2 from [Moh91] to our graph definition as follows:

Theorem 2.2. *Let $G = (V, E, w)$ be a finite undirected weighted connected graph with $|V| = n$, edge weights $w : E \to \mathbb{R} \setminus (-\infty, 0]$, and L_G the Laplacian matrix of G. Let $\mathrm{diam}(G)$ be the diameter of G. Then the following holds:*

$$\lambda_2 \geq \frac{4}{n \, \mathrm{diam}(G)}.$$

Proof. Let $G = (V, E, w)$ be a finite undirected weighted connected graph with $|V| = n$, edge weights $w : E \to \mathbb{R} \setminus (-\infty, 0]$, and L_G the Laplacian matrix of G. Let $\mathrm{diam}(G)$ be

the diameter of G. Consider a vector $x \in \mathbb{R}^n$ such that $x_i = \frac{1}{\sqrt{n}}$ for one subset of vertices and $x_j = -\frac{1}{\sqrt{n}}$ for another subset of vertices, ensuring $x \perp \mathbf{1}$ and $\|x\| = 1$.
The Rayleigh quotient for this vector is given through the quadratic form:

$$x^T L_G x = \sum_{(i,j)\in E} w_{ij}(x_i - x_j)^2 = \sum_{(i,j)\in E} w_{ij}\left(\frac{1}{\sqrt{n}} - \left(-\frac{1}{\sqrt{n}}\right)\right)^2 = \sum_{(i,j)\in E} w_{ij}\frac{4}{n}.$$

Since the sum of the weights w_{ij} for any path between the two farthest vertices (which defines the diameter) cannot exceed $\operatorname{diam}(G)$, we can bound the sum of the weights with the diameter:

$$x^T L_G x = \sum_{(i,j)\in E} w_{ij}\frac{4}{n} \leq \frac{4}{n} \cdot n \operatorname{diam}(G) = 4 \operatorname{diam}(G).$$

Hence,

$$\lambda_2 \geq \frac{x^T L_G x}{x^T x} \geq \frac{4}{n \operatorname{diam}(G)}.$$

□

Corollary 2.3. *The algebraic connectivity λ_2 can be bound by:*

$$\lambda_2 \geq 1 \quad \Leftrightarrow \quad n \operatorname{diam}(G) \leq 4.$$

Remark 2.8. *In the context of a large power grid, the characterization of the bound for the value of λ_2 provided by the inequality above is not that strong. As the number of vertices n corresponding to the buses in the power grid increases, the product $n \operatorname{diam}(G)$ tends to grow.*

The following inequality links the algebraic connectivity λ_2 to a combinatorial measure of connectivity. A higher λ_2 implies a higher Cheeger constant, suggesting better connectivity and a more robust network structure. Note that for a subset $S \subseteq E$ of edges, we denote $w(S)$ as the sum of the weights of these edges:

$$w(S) = \sum_{(u,v)\in S} w(u,v).$$

With that, the following definition generalizes the concept of the Cheeger constant (see [Chu97]) to weighted graphs by incorporating the edge weights into the boundary size.

Definition 2.16. *For a finite undirected weighted graph $G = (V, E, w)$, the **Cheeger constant** $h(G)$ is defined as:*

$$h(G) = \min_{\substack{S\subset V \\ 0<|S|\leq\frac{|V|}{2}}} \frac{w(\partial S)}{\operatorname{vol}(S)},$$

where $S \neq \varnothing$ and ∂S is the edge boundary of S, defined as the set of edges in E that have exactly one endpoint in S and the other endpoint in $V \setminus S$:

$$\partial S = \{(u,v) \in E \mid (u \in S \wedge v \in V \setminus S) \vee (v \in S \wedge u \in V \setminus S)\}.$$

Theorem 2.3 (Cheeger's Inequality)**.** *Let $G = (V, E, w)$ with $|V| = n$ be a finite undirected weighted connected graph with $w : E \to \mathbb{R} \setminus (-\infty, 0]$ and L_G be the Laplacian of G. The Cheeger constant $h(G)$ estimates λ_2 as follows:*

$$\frac{h(G)^2}{2} \leq \lambda_2 \leq 2h(G).$$

Proof. We use the proof idea proposed by [Chu07] using eigenvectors.
Therefore, let $G = (V, E, w)$ with $|V| = n$ be a finite undirected weighted connected graph with $w : E \to \mathbb{R} \setminus (-\infty, 0]$ and L_G be the Laplacian of G.
We first prove the upper bound. Let $S \subset V$ be the set achieving the minimum of the Cheeger constant, satisfying $0 < |S| \leq \frac{|V|}{2}$. Thus $h(G) = w(\partial S)/\mathrm{vol}(S)$, where $w(\partial S)$ is the total weight of edges leaving S, and $\mathrm{vol}(S)$ is the sum of the degrees of the vertices in S as given in Definition 2.16 of the Cheeger constant. We define $g \in \mathbb{R}^n$ as:

$$g := \chi_S - \frac{\mathrm{vol}(S)}{\mathrm{vol}(G)}\mathbf{1},$$

where χ_S is the characteristic function of S and $\mathbf{1} := (1, 1, \ldots, 1)^T \in \mathbb{R}^n$ is the vector of all ones. The Rayleigh quotient $R(L_G, g)$ is given by:

$$R(L_G, g) = \frac{g^T L_G g}{g^T g} = \frac{\sum_{(u,v)\in E} w_{uv}(g(u) - g(v))^2}{\sum_{v\in V} g(v)^2 d_v}.$$

For $u, v \in S$ or $u, v \notin S$, $g(u) - g(v) = 0$. When $u \in S$ and $v \notin S$, we get:

$$g(u) - g(v) = 1 - \left(-\frac{\mathrm{vol}(S)}{\mathrm{vol}(G)}\right) = 1 + \frac{\mathrm{vol}(S)}{\mathrm{vol}(G)}.$$

Hence, we can explicitly calculate the numerator of the quotient as follows:

$$\sum_{(u,v)\in\partial S} w_{uv}(g(u) - g(v))^2 = \sum_{(u,v)\in\partial S} w_{uv}\left(1 + \frac{\mathrm{vol}(S)}{\mathrm{vol}(G)}\right)^2 = \left(1 + \frac{\mathrm{vol}(S)}{\mathrm{vol}(G)}\right)^2 \cdot w(\partial S).$$

For a vertex $v \in S$, we have $g(v) = 1 - \mathrm{vol}(S)/\mathrm{vol}(G)$ and for a vertex $v \notin S$, we have that $g(v) = -\mathrm{vol}(S)/\mathrm{vol}(G)$. Thus, we can explicitly calculate the denominator of the quotient:

$$\begin{aligned}\sum_{v\in V} g(v)^2 d_v &= \sum_{v\in S}\left(1 - \frac{\mathrm{vol}(S)}{\mathrm{vol}(G)}\right)^2 d_v + \sum_{v\notin S}\left(-\frac{\mathrm{vol}(S)}{\mathrm{vol}(G)}\right)^2 d_v \\ &= \left(1 - \frac{\mathrm{vol}(S)}{\mathrm{vol}(G)}\right)^2 \cdot \mathrm{vol}(S) + \left(\frac{\mathrm{vol}(S)}{\mathrm{vol}(G)}\right)^2 \cdot (\mathrm{vol}(G) - \mathrm{vol}(S)).\end{aligned}$$

Therefore, from the above calculations, we can conclude that:

$$R(L_G, g) \leq 2h(G).$$

By definition of λ_2, it is $\lambda_2 \leq R(L_G, g) \leq 2h(G)$.
We now prove the lower bound. Let g denote an eigenvector associated with λ_2, the

second smallest eigenvalue of the Laplacian L_G. We use the notation where $g(v)$ refers to the component of the eigenvector g at the vertex v. Therefore, $L_G g = \lambda_2 g$ and $\sum_{v \in V} g(v)d(v) = 0$, where $d(v)$ is the degree of vertex v. We order the vertices such that:

$$g(v_1) \geq g(v_2) \geq \ldots \geq g(v_n).$$

Let $S_i = \{v_1, v_2, \ldots, v_i\} \subset V$ and define $h_i = \frac{w(\partial S_i)}{\text{vol}(S_i)}$, where $\text{vol}(S_i) = \sum_{v \in S_i} d(v)$ for $1 \leq i \leq n$. Define $r \in \{1, 2, \ldots, n\}$ such that $\text{vol}(S_r) \leq \text{vol}(G)/2$. With the orthogonality of g, we have:

$$\sum_{v \in V} g(v)d(v) = 0.$$

Thus,

$$\sum_{v \in V} g(v)^2 d(v) = \min_{c \in \mathbb{R}} \sum_{v \in V} (g(v) - c)^2 d(v) \leq \sum_{v \in V} (g(v) - g(v_r))^2 d(v).$$

We define the positive and negative parts of $g - g(v_r)$ as follows:

$$g_+(v) = \begin{cases} g(v) - g(v_r) & \text{if } g(v) \geq g(v_r) \\ 0 & \text{otherwise} \end{cases}.$$

$$g_-(v) = \begin{cases} |g(v) - g(v_r)| & \text{if } g(v) \leq g(v_r) \\ 0 & \text{otherwise} \end{cases}.$$

Without loss of generality, assume $R(g_+) \leq R(g_-)$. Then, $\lambda_2 \geq R(g_+)$, where

$$R(g_+) = \frac{\sum_{(u,v) \in E} w(u,v)(g_+(u) - g_+(v))^2}{\sum_{u \in V} g_+(u)^2 d(u)}.$$

Using the Cheeger constant $h(G)$ and the Cheeger ratio for subsets S_i, we have:

$$|\partial(S_i)| \geq h(G) \min(\text{vol}(S_i), \text{vol}(V) - \text{vol}(S_i)).$$

For the positive part g_+, this can be bounded as follows:

$$\lambda_2 \geq R(g_+) \geq \frac{\left(\sum_{i=1}^{n-1} (g_+(v_i)^2 - g_+(v_{i+1})^2) h(G) \min(\text{vol}(S_i), \text{vol}(V) - \text{vol}(S_i))\right)^2}{2\left(\sum_{u \in V} g_+(u)^2 d(u)\right)^2}.$$

Since $\text{vol}(S_i) \leq \frac{\text{vol}(V)}{2}$ for the optimal subset S_i,

$$\lambda_2 \geq \frac{h(G)^2}{2}.$$

□

Corollary 2.4. *The algebraic connectivity λ_2 can be bound by:*

$$h(G) \geq \sqrt{2} \quad \Longrightarrow \quad \lambda_2 \geq 1.$$

We can extend the above corollary to formulate an even stronger bound:

Corollary 2.5. *The algebraic connectivity λ_2 can be bound by:*

$$h(G) \geq \sqrt{2\Delta} \quad \wedge \quad \Delta \geq 1 \quad \Longrightarrow \quad \lambda_2 \geq 1,$$

where $\Delta = \max_i d(v_i)$ is the maximum degree of the graph G.

Proof. By substituting $h(G) \geq \sqrt{2\Delta}$ into the lower bound of Cheeger's inequality, we get:

$$\lambda_2 \geq \frac{h(G)^2}{2} \quad \Leftrightarrow \quad \lambda_2 \geq \frac{(\sqrt{2\Delta})^2}{2} \quad \Leftrightarrow \quad \lambda_2 \geq \Delta \geq 1.$$

□

Remark 2.9. *Note that we need to assume $\Delta \geq 1$. In large power systems, this can be assumed implicitly, as described below. Recall that Δ represents the following sum over the weighted edges:*

$$\Delta = \max_i d(v_i) = \max_i \sum_{u \in V, (v_i,u) \in E} w(v_i, u).$$

The susceptance (inverse of reactance) values in power systems are typically designed to ensure efficient power flow and stability. Even though individual susceptance values might be less than one, their cumulative sum for each node usually leads to a weighted degree greater than one, contributing to the overall robustness of the power grid [WKKVM15]. Additionally, large power systems are designed with multiple connections to ensure redundancy and reliability. This means that each node (or bus) is usually connected to several other nodes, resulting in higher weighted degrees. This can be observed in the IEEE 118-bus system (see Figure 2.1), which is a standard test case for power grid analysis. Note that the IEEE 118-bus test case is a specific power system model used for research and educational purposes, representing a simplified version of the American Electric Power system. This test case includes detailed information about buses, generators, loads, and other components necessary for power flow analysis [Uni, Chr].

2.2.1.2 Upper Bound for the Greatest Eigenvalue

To find a bound for the largest eigenvalue λ_n (see Remark 2.6) of the Laplacian matrix L_G, we can utilize several properties and known results about Laplacian eigenvalues. The Gershgorin Circle Theorem provides a way to bound eigenvalues based on the entries of the matrix. By adapting Theorem 6.1.1 from [HJ13], we can apply it specifically to Laplacian matrices associated with our graphs.

Definition 2.17. *Let $G = (V, E, w)$ be a finite undirected weighted graph with $|V| = n$ and $w : E \to \mathbb{R}_{>0}$ is a weight function. For each $i \in \{1, 2, \ldots, n\}$, the **Gershgorin disc** D_i is defined for $i \in \{1, 2, \ldots, n\}$ as:*

$$D_i = \{z \in \mathbb{C} \mid |z - d(v_i)| \leq d(v_i)\},$$

where $d(v_i)$ is the degree of $v_i \in V$ from Definition 2.7.

Note that this definition is consistent with the usual definition of Gershgorin circles, where the i-th Gershgorin disc is centered at $(L_G)_{ii} = d(v_i)$ with radius R_i, where:

$$R_i = \sum_{j \neq i} w(v_i, v_j) = d(v_i).$$

Theorem 2.4 (Gershgorin Circle Theorem for Laplacian Matrices)**.** *Let $G = (V, E, w)$ be a finite undirected weighted graph with $|V| = n$ and $w : E \to \mathbb{R}_{>0}$ is a weight function. For the Laplacian matrix L_G of the graph G, every eigenvalue λ of L_G lies within at least one of the Gershgorin discs:*

$$\lambda \in \bigcup_{i=1}^{n} D_i,$$

where each Gershgorin disc D_i for $i \in \{1, 2, \ldots, n\}$.

Proof. Let λ be an eigenvalue of L_G and $\mathbf{x} = (x_1, x_2, \ldots, x_n)^T \in \mathbb{C}^n$ with $\mathbf{x} \neq 0$ be the corresponding eigenvector. Therefore, $L_G\mathbf{x} = \lambda\mathbf{x}$. Given that, we can write the i-th entry as:

$$\sum_{j=1}^{n} (L_G)_{ij} x_j = d(v_i)x_i - \sum_{(v_i, v_j) \in E} w(v_i, v_j)x_j = \lambda x_i. \tag{2.2}$$

The abov Eq. 2.2 can be rearranged as follows:

$$\Leftrightarrow (d(v_i) - \lambda)x_i = \sum_{(v_i, v_j) \in E} w(v_i, v_j)x_j \Leftrightarrow |d(v_i) - \lambda||x_i| = \left| \sum_{(v_i, v_j) \in E} w(v_i, v_j)x_j \right| \tag{2.3}$$

$$\Leftrightarrow |d(v_i) - \lambda||x_i| \leq \sum_{(v_i, v_j) \in E} w(v_i, v_j)|x_j| \Leftrightarrow |d(v_i) - \lambda||x_i| \leq d(v_i) \max_{(v_i, v_j) \in E} |x_j| \tag{2.4}$$

Since $x_i \neq 0$, we can divide the result from Eq. 2.4 by $|x_i|$, which leaves us with:

$$|d(v_i) - \lambda| \leq d(v_i) \frac{\max_{(v_i, v_j) \in E} |x_j|}{|x_i|}. \tag{2.5}$$

Since λ must satisfy Eq. 2.5 for some $i \in \{1, 2, \ldots, n\}$ and considering $\max_i |x_j|/|x_i|$, it

follows that:

$$|d(v_i) - \lambda| \leq d(v_i).$$

Therefore, every eigenvalue λ of the Laplacian matrix L_G lies within at least one of the Gershgorin discs $D_i = \{z \in \mathbb{C} \mid |z - d(v_i)| \leq d(v_i)\}$. □

Corollary 2.6. *From the above theorem, it follows that all eigenvalues λ_i for $1 \leq i \leq n$ of the Laplacian matrix L_G lie within the interval:*

$$\lambda_i \in [0, 2\Delta],$$

where $\Delta = \max_i d(v_i)$ is the maximum degree of the graph G.

Proof. According to the Gershgorin Circle Theorem, each eigenvalue of L_G lies within at least one of the Gershgorin discs centered at $d(v_i)$ with radius $d(v_i)$, meaning the eigenvalues are within the interval:

$$0 \leq \lambda \leq 2 \max_{1 \leq i \leq n} d(v_i).$$

□

For a connected graph, the Perron-Frobenius theorem provides insights into the largest eigenvalue of the adjacency matrix A_G. Although this is not directly for the Laplacian, it can be related as follows:

Lemma 2.1. *Let $G = (V, E, w)$ with $|V| = n$ be a finite undirected weighted connected graph with $w : E \to \mathbb{R} \setminus (-\infty, 0]$ and A_G be the adjacency matrix, D_G the degree matrix, and L_G be the Laplacian matrix of G. Then, the eigenvalues λ_i of L_G are related to the eigenvalues μ_i of A_G by:*

$$\lambda_i = d(v_i) - \mu_i.$$

Proof. Let $G = (V, E, w)$ with $|V| = n$ be a finite undirected weighted connected graph with $w : E \to \mathbb{R} \setminus (-\infty, 0]$ and A_G, D_G, L_G the matrices as defined. Let $\mathbf{v} \in \mathbb{R}^n$ be an eigenvector of A_G with eigenvalue $\mu \in \mathbb{R}$, therefore $A_G\mathbf{v} = \mu\mathbf{v}$. Note that the Laplacian is given by definition as $L_G = D_G - A_G$, thus:

$$L_G\mathbf{v} = (D_G - A_G)\mathbf{v} = D_G\mathbf{v} - A_G\mathbf{v} = D_G\mathbf{v} - \mu\mathbf{v}.$$

For each vertex $v_i \in V$, the i-th component of the vector $D_G\mathbf{v}$ is $d(v_i)\mathbf{v}_i$, where $d(v_i)$ is the degree of vertex v_i. Thus:

$$(L_G\mathbf{v})_i = d(v_i)\mathbf{v}_i - \mu\mathbf{v}_i = (d(v_i) - \mu)\mathbf{v}_i.$$

Hence, $\mathbf{v}$ is also an eigenvector of L_G with the eigenvalue $\lambda = d(v_i) - \mu$. Therefore, the eigenvalues λ_i of L_G are related to the eigenvalues μ_i of A_G by:

$$\lambda_i = d(v_i) - \mu_i.$$

□

We can formulate Theorem 4.5.1 from [Spi19] for the adjacency matrix in our case as:

Theorem 2.5 (Perron-Frobenius)**.** *Let $G = (V, E, w)$ with $|V| = n$ be a finite undirected weighted connected graph with $w : E \to \mathbb{R} \setminus (-\infty, 0]$ and A_G be the adjacency matrix of G. Let $\mu_1 \leq \mu_2 \leq \ldots \leq \mu_n$ be the eigenvalues of A_G. Then:*

1. *The eigenvalue μ_n has a strictly positive eigenvector.*
2. *$\mu_n \geq -\mu_1$.*
3. *$\mu_n > \mu_{n-1}$.*

Proof. Let $G = (V, E, w)$ be a finite undirected weighted connected graph with $|V| = n$, $w : E \to \mathbb{R} \setminus (-\infty, 0]$, and A_G be the adjacency matrix of G. Let $\mu_1 \leq \mu_2 \leq \ldots \leq \mu_n$ be the eigenvalues and $x_1, x_2, \ldots, x_n$ the corresponding orthonormal eigenvectors of A_G. We again use the notation where $x_i(k)$ refers to the component of the eigenvector x_i at the index k.
1. Since μ_n is the largest eigenvalue of A_G, we need to show that there exists an eigenvector x_n with $x_n(i) > 0$ for all i. Note that since A_G is symmetric, we can use the Rayleigh coefficient for the largest eigenvalue μ_n of A_G to express:

$$\mu_n = \max_{x \in \mathbb{R}^n} R(A_G, x) = x_n^T A_G x_n = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} x_n(i) x_n(j).$$

Consider a vector $y \in \mathbb{R}^n$ with $y(i) := |x_n(i)|$ for all $1 \leq i \leq n$. Then the vector y has unit norm, $\|y\| = y^T y = x_n^T x_n = 1$. We need to show y is also an eigenvector corresponding to μ_n. To see this, we have:

$$\mu_n = x_n^T A_G x_n = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} x_n(i) x_n(j).$$

Since $a_{ij} \geq 0$ (given $w : E \to \mathbb{R} \setminus (-\infty, 0]$), replacing $x_n(i)$ with $|x_n(i)|$ does not decrease the value of the sum and might increase it. Thus, we can formulate the inequality:

$$\sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} x_n(i) x_n(j) \leq \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} |x_n(i)||x_n(j)| = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} y(i) y(j) = y^T A_G y.$$

Since μ_n is the maximum value of $x^T A_G x$ over all unit vectors x:

$$y^T A_G y \leq \mu_n.$$

Therefore, we can assemble the inequalities and formulate:

$$\mu_n = x_n^T A_G x_n \leq y^T A_G y = \mu_n \quad \Leftrightarrow \quad \mu_n = \mu_n,$$

which implies that y is an eigenvector corresponding to μ_n. Because y consists of the absolute values of the components of x_n, and $y(i)$ are all positive, this proves that μ_n has a positive eigenvector. We discuss why none of the entries of y can be zero. Suppose for contradiction that $y(i) = 0$ for some $1 \leq i \leq n$. Since G is connected, if $y(i) = 0$ for some $1 \leq i \leq n$, there must be at least one neighboring vertex k connected by an edge $(i, k) \in E$. Given that y has entries derived from the absolute values of x_n, and since x_n has no zero entries for a connected graph, y must have all strictly positive entries.

Therefore, the neighboring vertex k must have $y(k) > 0$. This implies:

$$(A_G y)_i = \sum_{\substack{j=1 \\ (i,j)\in E}}^{n} y(j) \geq y(k) > 0.$$

However, since y is an eigenvector corresponding to μ_n,

$$(A_G y)_i = \mu_n y(i) = \mu_n \cdot 0 = 0,$$

which leads to a contradiction. Therefore, $y(i) > 0$ for all i, showing that μ_n has a strictly positive eigenvector.
2. We again define a vector $y \in \mathbb{R}^n$ with $y(i) := |x_1(i)|$ for all $1 \leq i \leq n$. We again use the Rayleigh coefficient for the smallest eigenvalue μ_1 of A_G to express:

$$|\mu_1| = \left| \min_{x\in\mathbb{R}^n} R(A_G, x) \right| = |x_1^T A_G x_1| \leq \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}|x_1(i)||x_1(j)| = y^T A_G y.$$

We know $y^T A_G y \leq \mu_n$ because μ_n is the largest eigenvalue of A_G. Therefore:

$$|\mu_1| \leq \mu_n \quad \Leftrightarrow \quad \mu_n \geq -\mu_1.$$

3. We follow the procedure from the previous steps and define $y \in \mathbb{R}^n$ such that $y(i) = |x_{n-1}(i)|$ for all $1 \leq i \leq n$. Then again, we receive the inequality:

$$\mu_{n-1} = x_{n-1}^T A_G x_{n-1} \leq y^T A_G y \leq \mu_n.$$

By the first part of the theorem, we can take x_n to be strictly positive. Since eigenvectors associated with distinct eigenvalues are orthogonal, we have $\langle x_n, x_{n-1}\rangle = 0$. This orthogonality implies that both x_n and x_{n-1} are non-zero vectors, and x_{n-1} must have both positive and negative components. We consider two scenarios:
Scenario 1: Suppose all components of x_{n-1} are non-zero. Given that G is connected, there exists an edge $(i, j) \in E$ where $x_{n-1}(i) < 0$ and $x_{n-1}(j) > 0$. Consequently, we have $x_{n-1}(i)x_{n-1}(j) < |x_{n-1}(i)||x_{n-1}(j)|$, leading to the desired strict inequality, thereby showing that $\mu_{n-1} < \mu_n$.
Scenario 2: Assume $x_{n-1}(i) = 0$ for some $1 \leq i \leq n$. If all the inequalities we considered earlier are equalities, then y would be an eigenvector for μ_n with $y \geq 0$. From our earlier argument, if $y \geq 0$ and is an eigenvector corresponding to μ_n in a connected graph G, then none of the entries of y can be zero. Since this leads to $y(i) = x_{n-1}(i) = 0$ for some i, we reach a contradiction. □

Corollary 2.7. *Let $G = (V, E, w)$ be defined as in the theorem. Let Δ be the maximum degree of the graph G. The largest eigenvalue λ_n of the Laplacian matrix L_G for a graph G with maximum degree Δ is bounded by:*

$$\lambda_n \leq 2\Delta.$$

Proof. Let Δ be the maximum degree of the graph G. This means that for some vertex $v_{\max}$, $d(v_{\max}) = \Delta$. The Perron-Frobenius theorem tells us that the largest eigenvalue

μ_1 of A_G is positive and the smallest eigenvalue μ_n satisfies $\mu_n \geq -\mu_1$. The largest eigenvalue λ_n of L_G corresponds to the smallest eigenvalue μ_n of A_G since $\lambda_i = d(v_i) - \mu_i$ and subtracting a smaller μ_i gives a larger λ_i. Thus, it is $\lambda_n = d(v_{\max}) - \mu_n$. From the Perron-Frobenius theorem, we have that $\mu_n \geq -\mu_1$, therefore:

$$\lambda_n \leq \Delta - (-\mu_1) = \Delta + \mu_1.$$

Since μ_1 (the largest eigenvalue of A_G) is at most the maximum degree Δ, we have:

$$\lambda_n \leq \Delta + \Delta = 2\Delta.$$

□

2.2.2 Summary of Graph Theory Model to Energy Grids

Before moving on to integrating power flow into the model, it is important to revisit the components of an energy grid, which can be represented as a graph. This graph-based representation enables the application of graph theory concepts to effectively analyze and optimize the power grid. The primary components are buses and transmission lines.

Remark 2.10 (Energy Grid as a Graph)**.** *As we saw, an energy grid can be modeled as a finite undirected weighted graph $G = (V, E, w)$ (see Definition 2.3), where:*

- *V is the set of **buses** (nodes) representing various components in the grid (see [NS14]):*
  - ***Generation buses**: Buses where electricity is generated and injected into the grid. Therefore, a generator is connected to this bus.*
  - ***Load buses**: Buses where electricity is consumed.*
  - ***Reference bus**: A special bus that serves as the reference point for voltage magnitude and phase angle in the network.*
  - ***Transmission buses**: Buses that primarily function as connection points between transmission lines and other buses.*
- *$E \subseteq \{(u, v) \in V \times V \mid u \neq v \wedge (u, v) \in E \implies (v, u) \in E\}$ is the set of **transmission lines** (edges) connecting the buses, ensuring the graph is undirected.*
- *Each edge $(u, v) \in E$ is associated with a weight $w(u, v) \in \mathbb{R}_{>0}$, representing a value related to the transmission line between these buses. Typically, this weight is the susceptance of the transmission line between buses u and v.*

An energy grid modeled as a graph G must satisfy the following conditions:

1. ***Connectedness**: The graph G is connected, ensuring that there is a path between any pair of buses. This implies the potential for power flow throughout the grid. If any bus is isolated, it is considered part of a separate sub-grid, which can then be modeled as an independent graph.*
2. ***Non-negative Weights**: The weights $w(u, v)$ are positive, as susceptance and admittance are typically positive in practical power systems (see Remark 2.4).*

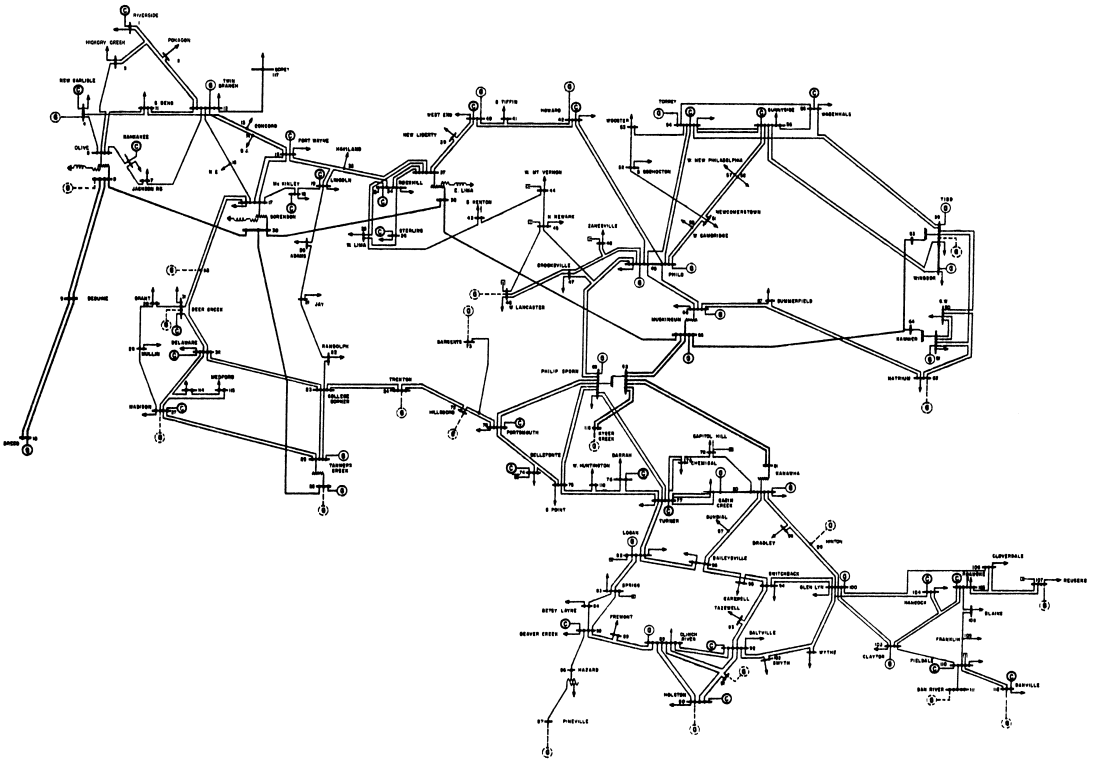

Figure 2.1: IEEE 118-bus system diagram as a reference for a weighted graph model. This test case represents a simplified version of the American Electric Power system in the Midwestern US as of December 1962. [Chr]

2.3 Integrating Power Flow into the Model

The power flowing between any transmission line is an important and limiting factor in describing the Optimal Power Flow (OPF). To include this element of power flow into our optimization problem, we first need to examine some concepts that describe such power in an energy grid. A fundamental part involves distinguishing between Direct Current (DC) and Alternating Current (AC) systems.

In contrast to DC, where the flow of electric charge is in a single direction, alternating current circuits use voltages and currents that are time-varying signals, which can be imagined as sine waves. These waves oscillate with a certain frequency (typically 50 Hz). The sine wave nature of AC allows for efficient transmission of energy over long distances, a crucial property that has shaped the modern electricity grid [GSO17].
To integrate these dynamics into OPF, we use the power flow equations, which are derived from Kirchhoff's laws but expressed in terms of phasors for AC systems (see Theorem 2.8). These equations take into account the complex relationship between voltages, currents, impedances of components, and loads. By solving the power flow equations, we can determine the distribution of voltages across the network and the flow of power between nodes, which is essential for optimizing the operation of the power system under various constraints, such as minimizing losses, maintaining voltage levels within acceptable limits, and ensuring the system operates within its thermal and stability bounds.

Definition 2.18. *The **Impedance** $Z \in \mathbb{C}$ is a measure of the total opposition that a circuit presents to the flow of alternating current:*

$$Z := R + iX = |Z|e^{i\theta_Z}$$

*where $R > 0$ is called the **resistance**, $X \in \mathbb{R}$ is called the **reactance**, and $\theta_Z \in (-\pi, \pi]$ is the phase angle of the impedance.*

Impedance is a central value in any circuit. Every device found in an electric power system has an impedance [vM06]. Unlike in DC circuits, where resistance is the only factor affecting the flow of electric current, AC circuits contain additional characteristics due to the presence of capacitance and inductance. These elements introduce phase shifts between voltage and current, making the concept of resistance inadequate for fully describing the opposition to current flow in AC circuits. Therefore, impedance can be viewed as an expanded version of resistance, encompassing its effects in alternating current scenarios [Gri12b].

Definition 2.19. *The **Admittance** $Y \in \mathbb{C}$ is a measure of how easily a circuit will allow current to flow.*

$$Y := \frac{1}{Z} = G + iB = |Y|e^{i\theta_Y}$$

*where $G \in \mathbb{R}$ is called the **conductance**, $B \in \mathbb{R}$ is called the **susceptance**, and $\theta_Y \in (-\pi, \pi]$ is the phase angle of the admittance.*

Admittance, as the inverse of impedance, represents the ease with which current flows through a component. It includes the conductance, which denotes the direct path for

current, and the susceptance, indicating the effect of capacitance and inductance. Therefore, admittance quantifies how readily an electrical circuit conducts current, where higher admittance suggests less opposition to current flow.
Since admittances or impedances, and thus conductance, susceptance, resistance, and reactance, are all defined through their relation between two given buses, an AC system consisting of multiple connected buses can be described as a matrix or vector. This allows for a representation of these values in the context of the entire power system, which makes it easier to analyze the flow of current and the distribution of voltages throughout the system. For example, the current $[I] \in \mathbb{C}^{n\times 1}$ and voltage $[V] \in \mathbb{C}^{n\times 1}$ can be expressed as vectors, and admittance as a matrix $Y \in \mathbb{C}^{n\times n}$. The elements of this matrix, Y_{ij}, represent the admittance between nodes i and j for $1 \leq i, j \leq n$ in an n-bus system.

Definition 2.20. *Consider an AC system consisting of n buses. Let $\mathcal{N}(i)$ represent the set of buses connected to bus i and y_{ij} the admittance of the line between buses i and j. The matrix $[Y] \in \mathbb{C}^{n\times n}$ is called the **Admittance matrix**, where:*

$$Y_{ij} = \begin{cases} \sum_{k\in\mathcal{N}(i)} y_{ik} & \text{if } i = j \\ -y_{ij} & \text{otherwise} \end{cases}$$

*Let b_{ij} be the susceptance of the line between buses i and j. The matrix $[B] \in \mathbb{R}^{n\times n}$ is called the **Susceptance Matrix**, where:*

$$B_{ij} = \begin{cases} \sum_{k\in\mathcal{N}(i)} b_{ik} & \text{if } i = j \\ -b_{ij} & \text{otherwise} \end{cases}$$

Note that in our case, the shunt admittance can usually be ignored because, for a high-voltage transmission line spanning less than 50 miles, the shunt capacitance of the line can be ignored. However, it should be modeled for lightly loaded distribution lines, such as underground lines [Gri12b]. For purposes of better readability, the susceptance matrix $[B] \in \mathbb{R}^{n\times n}$ is sometimes denoted by B.

Remark 2.11 (Susceptance matrix as a graph Laplacian)**.** *With this definition, the susceptance matrix B (as well as the admittance matrix) can be viewed as the Laplacian matrix L_G of a finite undirected weighted connected graph (see Definition 2.3 from Section 2.2), where the weights correspond to the susceptances of the transmission lines. Hence, all the results from Section 2.2 for a finite undirected weighted connected graph can be applied if we identify the weights with the susceptance. This works as follows for a connected n-bus AC system:*
We assume that the line susceptance B_{ij} between any two given buses i and j is positive ($B_{ij} > 0$). This assumption is reasonable, as described in Remark 2.4 from Section 2.2, since susceptance is usually positive under normal conditions. Let $G = (V, E, b)$ be a finite undirected weighted connected graph with $n = |V|$. Define the weight function by

$$b : E \to \mathbb{R}_{>0}, \quad (i, j) \mapsto B_{ij},$$

which maps each edge $(i, j) \in E$ to the positive susceptance B_{ij} between buses i and j.

Theorem 2.6 (Ohm's Law)**.** *$V = I \cdot Z$ where $V \in \mathbb{C}$ is the voltage across the resistor, $I \in \mathbb{C}$ is the current through the resistor, and $Z \in \mathbb{C}$ is the impedance.*

Proof. A complete derivation of this law can be found in [LSM17]. □

Note that Ohm's Law formulated above is a generalized form for AC circuits. It implies the common formulation of $V/R = I$ for DC circuits when the imaginary part of impedance vanishes.
The power in AC systems can be more complex to analyze than DC power because of these time-varying and thus complex characteristics. The concept of phasors becomes relevant in this context. In power system analysis, a phasor is the polar form of any complex value associated with a component. Through Euler's formula, we visualize a sine wave. In AC circuits, both voltage and current can be represented as phasors. The magnitude of a phasor reflects the amplitude of this wave, while the phase angle represents the time shift between the wave and a reference point in time, typically another wave. As already described with impedance and admittance, voltage V and current I can also be represented as phasors:

Remark 2.12 (Phasor Representation of Voltage and Current)**.**

$$V = |V|e^{i\theta_V} \qquad I = |I|e^{i\theta_I}$$

where $V \in \mathbb{C}$ is the voltage and $\theta_V \in (-\pi, \pi]$ is the voltage angle and $I \in \mathbb{C}$ is the current and $\theta_I \in (-\pi, \pi]$ is the current angle.

The following Kirchhoff's Laws will serve as a base for deriving the AC power flow equations. These laws provide a framework for analyzing the flow of electric current and the distribution of voltages in electrical circuits. Kirchhoff's Current Law ensures that all currents flowing into and out of any node are balanced, and Kirchhoff's Voltage Law focuses on the conservation of energy around circuit loops, asserting that the sum of all voltage changes around a loop equals zero.

Theorem 2.7 (Kirchhoff's Laws)**.** *For any closed n-bus AC system, let the voltages at the buses be $V_1, \ldots, V_n \in \mathbb{C}$ and the currents through the elements connected to a bus $i \in \{1, 2, \ldots, n\}$ be $I_1, \ldots, I_m \in \mathbb{C}$. Then the system satisfies:*

- *Kirchhoff's Current Law (KCL): At any bus i,*

$$\sum_{k=1}^{m} I_k = 0,$$

  where the summation is over all currents flowing into and out of bus i.
- *Kirchhoff's Voltage Law (KVL):*

$$\sum_{k=1}^{n} V_k = 0,$$

  where the summation is over all voltages around the system.

Proof. A complete derivation of these laws can be found in [Wad06]. □

Note that KVL applies to any closed loop in the system as well. These laws apply both to AC and DC systems. In DC systems, the voltages and currents are real numbers. The

total power in AC circuits is described by a complex value S, which is composed of real power P and reactive power Q as follows:

Definition 2.21. *The **Complex Power** $S \in \mathbb{C}$ is defined as the product of the voltage $V \in \mathbb{C}$ and the complex conjugate of the current $I \in \mathbb{C}$:*

$$S := V \cdot \bar{I} = P + iQ$$

*where $P \in \mathbb{R}$ is called the **real power** and $Q \in \mathbb{R}$ is the **reactive power**.*

Real Power is the average power actually consumed or produced by a circuit. It is the component of power that does work. Reactive Power Q is the power that oscillates back and forth between the reactive components and the power source. It does not do any net work but is necessary for the use of AC power systems [vM06].

Remark 2.13. *The notation $V_k \angle \phi_k$ is commonly used in the context of AC circuit analysis. This notation denotes a voltage (or current) at a given bus k, that has a magnitude of V_k and a phase of ϕ_k. Note that in this notation the magnitude is expressed as V_k and not $|V_k|$ like in the mentioned phasor notation. The angle ϕ_k represents the phase of the voltage relative to some reference, often taken as the current or another voltage in the circuit. In terms of the given definitions, this means:*

- *When describing Complex Power $S \in \mathbb{C}$, if there is a voltage source at bus 1 defined as $V_1 \angle \phi_1$, this means that the voltage source has a phase angle of $\phi_1 \in (-\pi, \pi]$ radians with respect to the reference. If the current flowing through a circuit from bus 1 to bus 2 has a phasor representation $I_1 \angle \theta_1$, with $\theta_1 \in (-\pi, \pi]$, the complex power delivered to the circuit can be calculated as:*

$$S = V_1 \cdot \bar{I_1} = |V_1| e^{i\phi_1} \cdot |I_1| e^{-i\theta_1} = |V_1||I_1| e^{i(\phi_1 - \theta_1)}$$

- *From $V_1 \angle \phi_1$, the Real Power $P \in \mathbb{R}$ and Reactive Power $Q \in \mathbb{R}$ in the circuit can then be computed as:*

$$P = |V_1||I_1| \cos(\phi_1 - \theta_1)$$
$$Q = |V_1||I_1| \sin(\phi_1 - \theta_1)$$

  This means that the real power is the component of the complex power that aligns with the voltage phasor, and the reactive power is the component that is perpendicular to it. This aligns with the common perception of AC, where voltages rise and fall periodically over time.

Now we can formulate the power flowing at a given bus by the AC Power Flow Equations. We use the most commonly used Voltage-Based Formulations from [MH19]. The following equations then express the power balance at each bus in a given system:

Theorem 2.8 (AC Power Flow Equations)**.** *For an n-bus AC system, the power balance at any given bus $k \in \{1, ..., n\}$ is described by:*

$$P_k^G - P_k^L = \sum_{l=1}^{n} |V_k||V_l| \left(G_{kl} \cos(\theta_k - \theta_l) + B_{kl} \sin(\theta_k - \theta_l)\right) \tag{2.6}$$

$$Q_k^G - Q_k^L = \sum_{l=1}^{n} |V_k||V_l| \left(G_{kl} \sin(\theta_k - \theta_l) + B_{kl} \cos(\theta_k - \theta_l)\right) \tag{2.7}$$

where $P_k^G \in \mathbb{R}$ and $Q_k^G \in \mathbb{R}$ denote the generated active and reactive power, and $P_k^L \in \mathbb{R}$ and $Q_k^L \in \mathbb{R}$ the active and reactive load at a given bus k. The voltage at bus s is given by $V_s \angle \theta_s$ with angle $\theta_s \in (-\pi, \pi]$. The conductance between the line between k and l is described as $G_{kl} \in \mathbb{R}$ and susceptance as $B_{kl} \in \mathbb{R}$. Note that if there is no line between k and l, then $G_{kl} = B_{kl} = 0$ does not interfere with the given definition.

Proof. Note that this is more of a derivation than a formal proof, as a mathematical proof in the traditional sense does not apply here. We first consider a simple two-bus system and then generalize it to arbitrary buses and lines. Assume a system with two buses, bus 1 and bus 2, and a transmission line between them. To model this line, we need to consider some kind of impedance $Z_{12} = R_{12} + iX_{12}$ with a resistive R_{12} and reactive X_{12} component in it (see Figure 2.2).

Thus the admittance $Y_{12} = \frac{1}{Z_{12}}$ is given as the inverse of the impedance. We assume there is some kind of voltage $V_1 \angle \phi_1$ and $V_2 \angle \phi_2$ with corresponding phase angles ϕ_1 and ϕ_2 being injected at the buses. To model the current I_{12} flowing from bus 1 to bus 2, we can use $I_{12} = \frac{V_1 \angle \phi_1 - V_2 \angle \phi_2}{Z} = \frac{V_{12}}{Z} = V_{12} \cdot Y$.

To further abstract these current formulas to an arbitrary bus k, we can state that

$$I_k = \sum_{l=1}^{n} Y_{kl} \cdot V_l$$

Now we can introduce a form of power to our model using Ohm's law from Theorem 2.6. $S_k = V_k \cdot \bar{I_k}$ describes the complex power at any bus k:

$$S_k = V_k \cdot \sum_{l=1}^{n} \bar{Y_{kl}} \cdot \bar{V_l}$$

This is because Ohm's Law states that $\frac{1}{Z} \cdot V = \frac{V}{Z} = I$. Since the admittance $Y_{kl} = G_{kl} + iB_{kl}$ is described through its conductance component G and susceptance B:

$$S_k = V_k \sum_{l=1}^{n} (G_{kl} - iB_{kl}) \bar{V_l} = \sum_{l=1}^{n} V_k \bar{V_l} (G_{kl} - iB_{kl})$$

We can further express S as its real and reactive power component using $S_k = P_k + iQ_k$ and extend the voltages $V_k = |V_k| e^{i\theta_k}$ and $V_l = |V_l| e^{i\theta_l}$ in their exponential representations:

$$\begin{aligned}
P_k = \mathrm{Re}(S_k) &= \sum_{l=1}^{n} \mathrm{Re}\left(V_k \bar{V_l} (G_{kl} - iB_{kl}) \right) \\
&= \sum_{l=1}^{n} \mathrm{Re}\left(|V_k| e^{i\theta_k} |V_l| e^{-i\theta_l} (G_{kl} - iB_{kl}) \right) \\
&= \sum_{l=1}^{n} \mathrm{Re}\left(|V_k||V_l| e^{i(\theta_k - i\theta_l)} (G_{kl} - iB_{kl}) \right) \\
Q_k = \mathrm{Im}(S_k) &= \sum_{l=1}^{n} \mathrm{Im}\left(|V_k||V_l| e^{i(\theta_k - i\theta_l)} (G_{kl} - iB_{kl}) \right)
\end{aligned}$$

With the identity $i = e^{i\frac{\pi}{2}}$, linearity of $\mathrm{Re}(x+y) = \mathrm{Re}(x) + \mathrm{Re}(y)$ for $x, y \in \mathbb{C}$, and the fact that $\cos(x + \frac{\pi}{2}) = -\sin(x)$ for any given $x \in \mathbb{R}$, we get the following result:

$$\begin{aligned}
P_k &= \sum_{l=1}^{n} \mathrm{Re}\left(|V_k||V_l|e^{i(\theta_k-\theta_l)}(G_{kl} - iB_{kl})\right) \\
&= \sum_{l=1}^{n} |V_k||V_l|\mathrm{Re}\left(G_{kl}e^{i(\theta_k-\theta_l)} - iB_{kl}e^{i(\theta_k-\theta_l)}\right) \\
&= \sum_{l=1}^{n} |V_k||V_l|\mathrm{Re}\left(G_{kl}e^{i(\theta_k-\theta_l)} - B_{kl}e^{i\frac{\pi}{2}}e^{i(\theta_k-\theta_l)}\right) \\
&= \sum_{l=1}^{n} |V_k||V_l|\mathrm{Re}\left(G_{kl}e^{i(\theta_k-\theta_l)} - B_{kl}e^{i(\theta_k-\theta_l)+\frac{\pi}{2}}\right) \\
&= \sum_{l=1}^{n} |V_k||V_l|\left(G_{kl}\cos(\theta_k-\theta_l) - B_{kl}\cos\left(\theta_k-\theta_l+\frac{\pi}{2}\right)\right) \\
&= \sum_{l=1}^{n} |V_k||V_l|(G_{kl}\cos(\theta_k-\theta_l) + B_{kl}\sin(\theta_k-\theta_l))
\end{aligned}$$

Formulation of the reactive power Q_k follows respectively with the same concept:

$$\begin{aligned}
Q_k &= \sum_{l=1}^{n} \mathrm{Im}\left(|V_k||V_l|e^{i(\theta_k-\theta_l)}(G_{kl} - iB_{kl})\right) \\
&= \sum_{l=1}^{n} |V_k||V_l|\mathrm{Im}\left(G_{kl}e^{i(\theta_k-\theta_l)} - B_{kl}e^{i\frac{\pi}{2}}e^{i(\theta_k-\theta_l)}\right) \\
&= \sum_{l=1}^{n} |V_k||V_l|\left(G_{kl}\sin(\theta_k-\theta_l) - B_{kl}\sin\left(\theta_k-\theta_l+\frac{\pi}{2}\right)\right) \\
&= \sum_{l=1}^{n} |V_k||V_l|\left(G_{kl}\sin(\theta_k-\theta_l) + B_{kl}\cos(\theta_k-\theta_l)\right)
\end{aligned}$$

If we consider P_k^G as the power that is generated and P_k^L the load at bus k, the total power is given by $P_k = P_k^G - P_k^L$. For the reactive power, $Q_k = Q_k^G - Q_k^L$ respectively. Thus, we now receive the full equations:

$$\begin{aligned}
P_k^G - P_k^L &= \sum_{l=1}^{n} |V_k||V_l|(G_{kl}\cos(\theta_k-\theta_l) + B_{kl}\sin(\theta_k-\theta_l)) \\
Q_k^G - Q_k^L &= \sum_{l=1}^{n} |V_k||V_l|(G_{kl}\sin(\theta_k-\theta_l) + B_{kl}\cos(\theta_k-\theta_l))
\end{aligned}$$

□

The AC Power Flow Equations can thus be characterized as a set of complex non-linear simultaneous equations designed to model the distribution of electrical power through a network as in [MKT+19]. However, for quick and intuitive analysis, a simpler linear relationship is often needed. This is where DC power flow analysis comes into play. DC power flow provides a simplified linear approximation of the AC power flow equations,

making it easier to analyze and solve. This simplification comes with a number of approximations, such as neglecting the reactive power component and assuming a uniform voltage magnitude across the network. Additionally, according to Kirchhoff's Current Law, for any bus in an electrical circuit, the sum of currents flowing into the node must be equal to the sum of currents flowing out of the node. In the context of power flow analysis, this principle translates to ensuring that the power injected into the system at any given bus must balance with the total power flowing in and out of all lines connected to that bus. This balance is crucial for the stability and efficient operation of the power system and will become an equality constraint in our overall optimization problem.

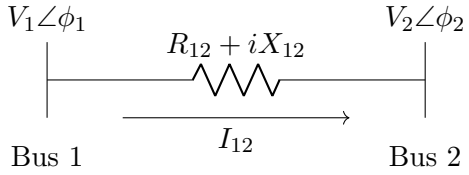

Figure 2.2: A simple two-bus model, showing an impedance $R_{12} + iX_{12}$ acting on the transmission line between bus 1 and bus 2 with current I_{12} flowing through it. Voltages $V_1\angle\phi_1$ and $V_2\angle\phi_2$ are injected at the given bus.

2.3.1 The DC Power Flow Approximation

In the following approximation, we need to make some assumptions and further investigate their behavior on the Power flow equations. The DC power flow approximation simplifies this by relating it to a DC circuit. Reactance X_{kl} in the AC system corresponds to resistance R_{kl} in a DC circuit. The difference in voltage angles $(\theta_k - \theta_l)$ corresponds to the voltage difference $(V_k - V_l)$ in a DC circuit, and the real power P_{kl} in the AC system corresponds to the current I_{kl} in the DC circuit. [WWS13]
In a DC circuit, the flow of current between two buses k and l can be expressed as:

$$I_{kl} = \frac{(V_k - V_l)}{R_{kl}}$$

Note that in this special DC case, I_{kl} should not be confused with the current in an AC circuit, which is complex. In the following, we show that under special circumstances (following Assumptions 1, 2, 3 and 4), one can simplify and express the AC power flow:

$$P_{kl} \approx \frac{(\theta_k - \theta_l)}{x_{kl}}$$

This equation is then linear and much easier to solve than the full AC power flow equations. It provides a good approximation for the real power flow in systems, as we see in the following paragraph. This approximation also significantly reduces the computational complexity compared to solving the full set of AC power flow equations, making it particularly useful for planning and real-time operations in power systems where detailed accuracy is not the primary concern and thus used in many operation models [DKS22].
We now can introduce the assumptions that let us linearize the AC power flow equations. Therefore consider an n-bus AC system and the set of AC power flow equations with its

parameters given as in Theorem 2.8:

Assumption 1: The total load in the AC system is low.

In an AC system operating under light load conditions, the total power consumed or transferred within the system is relatively low. This scenario directly impacts the reactive power Q_k at any given bus $k \in \{1, ..., n\}$, making it negligible in the overall power balance of the system.

Lemma 2.1. *For any $\varepsilon > 0$, if $|S_k| < \varepsilon$, then $|Q_k| < \varepsilon$.*

Proof. Assume there is an $\varepsilon > 0$ such that $|S_k| < \varepsilon$. Since with Definition 2.21 S_k can be expressed as $S_k = P_k + iQ_k$, it follows that

$$|S_k| = |P_k + iQ_k| = \sqrt{P_k^2 + Q_k^2} < \varepsilon.$$

Because $P_k^2 \geq 0$ and $\sqrt{x}$ is monotonic for all $x \in \mathbb{R}$, we get

$$|Q_k| = \sqrt{Q_k^2} \leq \sqrt{P_k^2 + Q_k^2} < \varepsilon.$$

□

Assumption 2: Resistances in transmission lines are negligible.

In AC transmission circuits, the reactance typically exceeds the resistance $R << X$, which means that the X/R ratio may surpass 10, as pointed out by [AC22]. This situation results in a very low line resistance, and consequently, the conductance G_{kl} between some buses $k, l \in \{1, ..., n\}$ is almost negligible. Under these circumstances, conductance is considered to be very small, often approximated as zero. Therefore, the admittance becomes mostly imaginary, characterized by a susceptance $B = -1/X$.

Lemma 2.2. *For any $\varepsilon > 0$, if $R_{kl} < \varepsilon$ and $R_{kl} << X_{kl}$, then $|G_{kl}| < \varepsilon$.*

Proof. Let $Z_{kl} = R_{kl} - iX_{kl}$ denote the impedance. Assume there is an $\varepsilon > 0$ such that $R_{kl} < \varepsilon$. Since the admittance Y_{kl} is defined as $Y_{kl} = \frac{1}{Z_{kl}}$ (see Def. 2.20), it follows that:

$$Y_{kl} = \frac{1}{Z_{kl}} = \frac{1}{R_{kl} - iX_{kl}} \cdot \frac{R_{kl} + iX_{kl}}{R_{kl} + iX_{kl}} = \frac{R_{kl} + iX_{kl}}{R_{kl}^2 + X_{kl}^2} = \underbrace{\frac{R_{kl}}{R_{kl}^2 + X_{kl}^2}}_{=G_{kl}} + i\frac{X_{kl}}{R_{kl}^2 + X_{kl}^2}$$

We can extract G_{kl} because $Y_{kl} = G_{kl} + iB_{kl}$ with Definition 2.20. This can be further simplified, noting that $R_{kl}^2 + X_{kl}^2 \geq X_{kl}^2$ since R_{kl}^2 is non-negative:

$$|G_{kl}| = \left| \frac{R_{kl}}{R_{kl}^2 + X_{kl}^2} \right| \leq \frac{|R_{kl}|}{X_{kl}^2}$$

Given that $R_{kl} < \varepsilon$ and assuming X_{kl} is significantly larger than R_{kl}, we can approximate:

$$|G_{kl}| < \frac{\varepsilon}{X_{kl}^2}$$

Since X_{kl} is significantly larger than R_{kl}, $\frac{\varepsilon}{X_{kl}^2}$ becomes very small. Thus, we have:

$$|G_{kl}| < \varepsilon$$

□

Assumption 3: All voltage magnitudes are close to nominal

In high-voltage transmission networks, the voltage magnitudes are generally regulated and maintained close to their nominal values, typically around one per unit, despite the effects of demand evolution and any unpredictable event [ZSMRZP21].

Assumption 4: The difference in voltage angles is small

Under typical operating scenarios, the voltage phasors' angular discrepancy at two interconnected buses k and j, denoted as $\theta_k - \theta_j$, is low. It usually stays below 10-15 degrees and rarely surpasses a 30-degree difference [Aly21]. Therefore, in this case, the angular gap across transmission circuits can be considered negligible.

Lemma 2.3 (Small-angle approximation)**.** *For small angles $\theta \in [0, 2\pi]$, the following approximations hold:* $\sin(\theta) \approx \theta$ *and* $\cos(\theta) \approx 1 - \theta^2/2$.

Proof. Since the functions $\sin, \cos \in C^\infty(\mathbb{R})$ are continuous and differentiable, we can consider the Taylor series expansion of $\sin(\theta)$ and $\cos(\theta)$ around $\theta = 0$:

$$\sin(\theta) = \theta + \mathcal{O}(\theta^3), \quad \cos(\theta) = 1 - \frac{\theta^2}{2!} + \mathcal{O}(\theta^4)$$

For very small θ, the terms θ^3 and beyond become negligible, thus the approximation $\sin(\theta) \approx \theta$ holds. For cosine, the terms θ^4 and beyond become negligible, leading to the approximation $\cos(\theta) \approx 1 - \theta^2/2$. □

Under the condition that we are considering purely reactive components, where the resistive part R of the impedance $Z = R + iX$ is negligible or zero, the relation between susceptance and reactance further simplifies. In this case, the admittance $Y = G + iB$ of a purely reactive component can be expressed in terms of its susceptance B by:

Lemma 2.4. *For any $\varepsilon > 0$, if $R_{kl} < \varepsilon$, then $B_{kl} \approx X_{kl}^{-1}$.*

Proof. Let $\varepsilon > 0$ and assume $R_{kl} < \varepsilon$. For an admittance $Y_{kl} = 1/Z_{kl}$, the following equation holds:

$$Y_{kl} = \frac{1}{Z_{kl}} = \frac{1}{Z_{kl}} \cdot \frac{\overline{Z_{kl}}}{\overline{Z_{kl}}} = \frac{\overline{Z_{kl}}}{|Z_{kl}|^2} = \frac{R_{kl} + iX_{kl}}{R_{kl}^2 + X_{kl}^2}$$

Because $Y_{kl} = G_{kl} + iB_{kl}$, the susceptance B_{kl} for $R_{kl} < \varepsilon$ small can be described as:

$$B_{kl} = \frac{X_{kl}}{R_{kl}^2 + X^2} \approx \frac{X_{kl}}{X_{kl}^2} = X_{kl}^{-1}$$

□

We can now formulate the DC Power Flow Approximation Theorem, incorporating all the mentioned assumptions to approximate the real power of an AC circuit as follows:

Theorem 2.9 (DC Power Flow Approximation)**.** *Consider an AC system with* $n \in \mathbb{N}$ *buses, where Assumptions 1, 2, 3, and 4 are all fulfilled:*

$$P_{kl} \approx \frac{\theta_k - \theta_l}{X_{kl}} \tag{2.8}$$

Proof. Consider all parameters given according to Theorem 2.8. Assume that $|S_{kl}| \approx 0$, $(\theta_k - \theta_l) \approx 0$, and $|R_{kl}| \approx 0$ are all small, and $|V_s| = 1$. We start the approximation with the definition of the AC Power Flow Equations mentioned in Theorem 2.8. Therefore, the power balance at any bus k is given by:

$$P_k^G - P_k^L = \sum_{l=1}^{n} |V_k||V_l| \left(G_{kl}\cos(\theta_k - \theta_l) + B_{kl}\sin(\theta_k - \theta_l)\right) \tag{2.9}$$

$$Q_k^G - Q_k^L = \sum_{l=1}^{n} |V_k||V_l| \left(G_{kl}\sin(\theta_k - \theta_l) + B_{kl}\cos(\theta_k - \theta_l)\right) \tag{2.10}$$

where $|V_s|$ is the magnitude of a voltage with angle θ_s at bus $1 \leq s \leq n$, and $Y_{kl} = G_{kl} + iB_{kl}$ is the admittance between k and l. Since $|S_{kl}| \approx 0$ with Assumption 1 and Lemma 2.1, Q_k and therefore Eq. 2.10 become negligible, and we end up with:

$$P_k^G - P_k^L \approx \sum_{l=1}^{n} |V_k||V_l| \left(G_{kl}\cos(\theta_k - \theta_l) + B_{kl}\sin(\theta_k - \theta_l)\right) \tag{2.11}$$

Applying Assumption 2, we conclude that $G_{kl} \approx 0$. Since $|R_{kl}| \approx 0$ and all voltage magnitudes $|V_s|$ are around one with Assumption 3, Eq. 2.11 for the real power transforms to:

$$P_k^G - P_k^L \approx \sum_{l=1}^{n} |V_k||V_l| B_{kl}\sin(\theta_k - \theta_l) \approx \sum_{l=1}^{n} B_{kl}\sin(\theta_k - \theta_l) \tag{2.12}$$

Because $(\theta_k - \theta_l) \approx 0$, phasor angles θ_k and θ_l can be approximated with the small-angle approximation $\sin(\theta_k - \theta_l) \approx (\theta_k - \theta_l)$ from Lemma 2.3, and Eq. 2.12 can be further reduced to:

$$\sum_{l=1}^{n} B_{kl}(\theta_k - \theta_l) \tag{2.13}$$

Using Assumption 4 with Lemma 2.4, we can express $B_{kl} = 1/X_{kl}$ as the inverse of X_{kl}, the reactance of the line between bus k and bus l. We can simplify Eq. 2.13 as follows:

$$P_k^G - P_k^L \approx \sum_{l=1}^{n} \frac{\theta_k - \theta_l}{X_{kl}} \tag{2.14}$$

Since we have the sum over all the power at a given bus k flowing, we can introduce a value $P_{kl} \in \mathbb{R}$ that describes this power flow between bus k and bus l and end up with a reduced form of Eq. 2.14:

$$P_k^G - P_k^L \approx \sum_{l=1}^{n} P_{kl} \tag{2.15}$$

where P_{kl} is expressed as in Eq. 2.8 from the Theorem:

$$P_{kl} \approx \frac{\theta_k - \theta_l}{X_{kl}} \tag{2.16}$$

□

Note that after all assumptions are met, the character of the power flow problem can be simplified to a System of Linear Equations (SLE). We receive a matrix $B \in \mathbb{R}^{n \times n}$ from Theorem 2.8 and $P := P_k^G - P_k^L \in \mathbb{R}^n$:

$$P_k^G - P_k^L = \sum_{l=1}^{n} B_{kl}(\theta_k - \theta_l) \iff B\,\theta = P$$

to be solved for θ by calculating B^{-1} if B is invertible.

2.3.2 Invertibility of the Susceptance Matrix

We now construct the invertible reduced susceptance matrix, which can be related to the graph Laplacian as described in Remark 2.11 from Section 2.3. The DC power flow equations in matrix form, $P = B\theta$, can lead to a singular matrix B under certain conditions. To ensure the invertibility of the matrix B, the following invertibility conditions must be met:

1. **Positive Susceptance:** The susceptance values in the matrix B must be non-zero for all transmission lines connecting buses.
2. **Connected Network:** The power system network graph must be connected.
3. **Reference Bus Definition:** A reference bus must be properly defined and excluded from the reduced matrix B.

Since we are considering a real AC power network, the susceptance is non-zero by Definition 2.19 from Section 2.3 and can be assumed positive as described in Remark 2.4 from Section 2.2. The connectedness is ensured by the construction of the power grid from Section 2.2. Therefore, the primary concern regarding the invertibility of B lies in the proper definition of the reference bus, as illustrated in the following remark.

Remark 2.14. *Assume $B \in \mathbb{R}^{n \times n}$ is the susceptance matrix. By construction, B is the Laplacian matrix of the graph representing the power network. Let $\mathcal{N}(i)$ represent the set of buses connected to bus i and b_{ij} the susceptance of the line between buses i and j, then B is given by:*

- *For $i \neq j$: $B_{ij} = -b_{ij}$*
- *For $i = j$: $B_{ii} = \sum_{j \in \mathcal{N}(i)} b_{ij}$*

The construction implies that the row sums of B are all zero by Thm. 2.7 from Section 2.3 (Kirchhoff's Law):

$$\sum_{j=1}^{n} B_{ij} = B_{ii} + \sum_{j \neq i} B_{ij} = \sum_{j \in \mathcal{N}(i)} b_{ij} + \sum_{j \in \mathcal{N}(i)} (-b_{ij}) = 0.$$

As a result, the rows of B are linearly dependent, making rank$(B) = n - 1$. Therefore, B is singular and not invertible.

To avoid that singularity, one bus is chosen as the reference bus (see Remark 2.10), and its voltage angle is fixed (typically to zero). This removal of one degree of freedom makes the reduced bus susceptance matrix B' non-singular, allowing for the solution of the power flow equations.

Lemma 2.1. *The bus susceptance matrix $B \in \mathbb{R}^{n \times n}$ of an n-bus DC system is invertible if there exists a reference bus $k \in \{1, 2, \ldots, n\}$ with a fixed voltage angle $\theta_k := 0$.*

Proof. Let $B \in \mathbb{R}^{n \times n}$ be the susceptance matrix of an n-bus DC power flow system. Let bus $k \in \{1, 2, \ldots, n\}$ be the reference bus with a fixed voltage angle $\theta_k := 0$. We reduce the system by one equation and one variable, resulting in a reduced bus susceptance matrix $B' \in \mathbb{R}^{(n-1)\times(n-1)}$.
Define B' by removing the k-th row and k-th column from B:

$$B' = B(\{1, \ldots, n\} \setminus \{k\}, \{1, \ldots, n\} \setminus \{k\})$$

where $B(S, T)$ denotes the submatrix of B formed by taking the rows indexed by the set $S = \{1, \ldots, n\} \setminus \{k\}$ and columns indexed by the set $T = \{1, \ldots, n\} \setminus \{k\}$.
To show B' is invertible, consider the properties of the Laplacian matrix. The original matrix B is a Laplacian matrix representing a connected graph with n nodes (see Section 2.1). The Laplacian matrix of a connected graph has exactly one zero eigenvalue, corresponding to the eigenvector $\mathbf{1}$ according to Thm. 2.1 from Section 2.2. When one node is removed (by fixing the reference bus), the remaining submatrix B' has full rank, $n - 1$, ensuring invertibility. More concretely, the determinant of the Laplacian matrix of a connected graph with one node removed is non-zero, which guarantees that B' is non-singular:

$$\text{rank}(B') = n - 1$$

Thus, the reduced matrix B' is invertible. Therefore, the bus susceptance matrix B of the n-bus DC power flow system is invertible if a reference bus is defined. This completes the proof. □

Remark 2.15. *The removal of the columns and rows corresponding to the reference bus is essential to ensure the non-singularity of the reduced susceptance matrix. Attempting to use the original matrix with just zero entries for the reference bus will still result in a singular matrix:*
Let $B \in \mathbb{R}^{n \times n}$ be the original susceptance matrix of an n-bus system. Setting the entries corresponding to the reference bus (bus 1) to zero, the modified matrix $\tilde{B}$ would have the form:

$$\tilde{B} = \begin{pmatrix} 0 & 0 & \cdots & 0 \\ 0 & B_{22} & \cdots & B_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & B_{n2} & \cdots & B_{nn} \end{pmatrix}$$

While this adjustment seems to eliminate the dependency on the reference bus, it does not resolve the singularity issue. Since the first row and first column are all zeros, it is clear that rank$(\tilde{B}) = n - 1 < n$, indicating that $\tilde{B}$ is still singular.

The only correct approach is to form the reduced susceptance matrix B' by removing the row and column corresponding to the reference bus. For example, if bus 1 is the reference bus, the reduced matrix $B' \in \mathbb{R}^{(n-1)\times(n-1)}$ is obtained by removing the first row and first column of B:

$$B' = \begin{pmatrix} B_{22} & \cdots & B_{2n} \\ \vdots & \ddots & \vdots \\ B_{n2} & \cdots & B_{nn} \end{pmatrix}$$

By removing the reference bus, the matrix B' corresponds to the Laplacian matrix of the remaining connected graph. Since the original graph is connected, the reduced graph remains connected, ensuring that B' has full rank:

$$rank(B') = n - 1$$

Thus, B' is non-singular and invertible. Consequently, the system of power flow equations can be solved without encountering singularities:

$$P' = B'\theta'$$

where $P' \in \mathbb{R}^{n-1}$ is the vector of net power injections excluding the reference bus, and $\theta' \in \mathbb{R}^{n-1}$ is the vector of voltage angles excluding the reference bus.

From the proof of Lemma 2.1 we also receive a way of making the susceptance matrix invertible by leaving out the rows and columns connected to the reference bus.

Definition 2.22. *Let $B \in \mathbb{R}^{n\times n}$ be the susceptance matrix of an n-bus DC power flow system, and let bus $k \in \{1, 2, \ldots, n\}$ be chosen as the reference bus with $\theta_k := 0$. Define the **reduced susceptance matrix** $B' \in \mathbb{R}^{(n-1)\times(n-1)}$ as follows:*

$$B' = B(\{1, \ldots, n\} \setminus \{k\}, \{1, \ldots, n\} \setminus \{k\})$$

where $B(S, T)$ with $S = T = \{1, \ldots, n\} \setminus \{k\}$ denotes the submatrix of B formed by taking the rows indexed by the set S and columns indexed by the set T. In other words, B' is obtained by removing the k-th row and k-th column from B.

Remark 2.16. *By Lemma 2.1, B' is invertible. The reduced system of equations can be written as:*

$$P' = B'\theta'$$

where $P' \in \mathbb{R}^{n-1}$ is the vector of net power injections excluding the reference bus, and $\theta' \in \mathbb{R}^{n-1}$ is the vector of voltage angles excluding the reference bus.

2.4 Introducing Quantum Algorithms

2.4.1 Basic Notation for Quantum-Gate-Algorithms

To use quantum algorithms, we need a system that can handle quantum information. Unlike traditional computers, which use bits that are either "on" or "off" (binary), quantum computers use qubits. Qubits can represent both $|0\rangle$ and $|1\rangle$ simultaneously through a concept called superposition. This allows quantum computers to explore many possibilities at once. Therefore, we will introduce formalism from quantum mechanics tailored to our purposes. We will loosely follow the notation of [NC10, ST94].

2.4.1.1 Operators on Finite-Dimensional Hilbert Spaces

Definition 2.23 (Hilbert Space)**.** *A* ***Hilbert Space*** $\mathcal{H} \subseteq \mathbb{C}^n$ *is a finite-dimensional vector space over* $\mathbb{C}$ *with an inner product* $\langle \cdot, \cdot \rangle : \mathcal{H} \times \mathcal{H} \to \mathbb{C}$*, which is complete with respect to the induced norm* $||v|| = \sqrt{\langle v, v \rangle}$ *for every* $v \in \mathcal{H}$*.*

Note that for our purposes it is sufficient to consider finite-dimensional Hilbert spaces as arbitrary subsets of $\mathbb{C}^n$. A more general definition that includes infinite dimensions and different fields can be found in [SS05]. If we do not specify the product for a given Hilbert space explicitly, we assume the standard scalar product as given.

Example 2.1 (State Space)**.** *The set* $\mathbb{C}^n$ *equipped with the standard scalar product:*

$$\langle u, v \rangle = \sum_{k=1}^{n} \bar{u}_k v_k$$

where $u, v \in \mathbb{C}^n$*, is a Hilbert space. Because a quantum state will be an element of a space of this form.*

Definition 2.24. *Let* $\mathcal{H}_1, \mathcal{H}_2$ *be Hilbert spaces with inner products. The map* $\Lambda : \mathcal{H}_1 \to \mathcal{H}_2$ *is called a (Linear)* ***Operator*** *if for all* $u, v \in \mathcal{H}_1$ *and* $a, b \in \mathbb{C}$ *the following holds:*

$$\Lambda(a \cdot u + b \cdot v) = a \cdot \Lambda(u) + b \cdot \Lambda(v).$$

The notation $+$ and $\cdot$ denotes vector addition and scalar multiplication in the respective Hilbert spaces. These operations are well-defined because Hilbert spaces are, by definition, vector spaces over the field of complex numbers $\mathbb{C}$. Consequently, linear combinations of elements within these spaces follow the usual rules of vector addition and scalar multiplication. Additionally, since the Hilbert spaces $\mathcal{H}_1$ and $\mathcal{H}_2$ are finite-dimensional, any linear operator defined on them is necessarily bounded. This follows from the fact that in finite-dimensional spaces, all linear operators are continuous and therefore bounded.

Lemma 2.2. *Let* $\mathcal{H}_1$ *and* $\mathcal{H}_2$ *be Hilbert spaces. Then every linear operator* $\Lambda : \mathcal{H}_1 \to \mathcal{H}_2$ *is bounded. This means there exists a constant* $C \geq 0$ *such that for all* $u \in \mathcal{H}_1$*,*

$$\|\Lambda(u)\|_{\mathcal{H}_2} \leq C \|u\|_{\mathcal{H}_1}.$$

Proof. Let $\mathcal{H}_1 \subseteq \mathbb{C}^n$ and $\mathcal{H}_2 \subseteq \mathbb{C}^m$ be Hilbert spaces. Thus, we can identify Λ with a matrix $A \in \mathbb{C}^{m \times n}$ that acts on vectors in $\mathbb{C}^n$. For any vector $u \in \mathcal{H}_1$, we have $\Lambda(u) = Au$.

The norm of $\Lambda(u)$ in $\mathcal{H}_2$ corresponds to the Euclidean norm of Au in $\mathbb{C}^m$, and the norm of u in $\mathcal{H}_1$ corresponds to the Euclidean norm of u in $\mathbb{C}^n$. Thus, we need to show that there exists a constant $C \geq 0$ such that:

$$\|Au\|_{\mathbb{C}^m} \leq C\|u\|_{\mathbb{C}^n} \quad \text{for all } u \in \mathbb{C}^n.$$

In finite dimensions, all norms are equivalent, so there exists a constant C' such that $\|A\| \leq C'$, where $\|A\|$ denotes the operator norm of the matrix A, which is given by:

$$\|A\| = \sup_{\|u\|_{\mathbb{C}^n}=1} \|Au\|_{\mathbb{C}^m}.$$

Thus, using the properties of the operator norm, for any $u \in \mathcal{H}_1 \cong \mathbb{C}^n$, we have:

$$\|\Lambda(u)\|_{\mathcal{H}_2} = \|Au\|_{\mathbb{C}^m} \leq \|A\|\|u\|_{\mathbb{C}^n} \leq C'\|u\|_{\mathcal{H}_1}.$$

Therefore, Λ is bounded, with $C = C'$. □

Definition 2.25 (Set of Linear Operators)**.** *Let $\mathcal{H}$ be a Hilbert space. The set $\mathcal{L}(\mathcal{H})$ denotes the set of all linear operators on $\mathcal{H}$:*

$$\mathcal{L}(\mathcal{H}) = \{\Lambda : \mathcal{H} \to \mathcal{H} \mid \Lambda \textit{is a linear operator}\}.$$

Lemma 2.3. *Let $\mathcal{H}$ be a Hilbert space. The set $\mathcal{L}(\mathcal{H})$ forms a vector space over $\mathbb{C}$.*

Proof. Let $\mathcal{H}$ be an n-dimensional Hilbert space. The set of all $n \times n$ matrices over $\mathbb{C}$ forms a vector space. Every bounded linear operator on the finite-dimensional Hilbert space $\mathcal{H}$ can be represented as an $n \times n$ matrix. Therefore, with Lemma 2.2, the vector space structure of $\mathcal{L}(\mathcal{H})$ is inherited from the space of $n \times n$ matrices. □

Corollary 2.8. *Let $\mathcal{H}$ be an n-dimensional Hilbert space. The space of $n \times n$ matrices with complex entries, $\mathcal{M}(n, \mathbb{C})$, is isomorphic to the space of linear operators on $\mathcal{H}$:*

$$\mathcal{L}(\mathcal{H}) \cong \mathcal{M}(n, \mathbb{C}).$$

Therefore, in the context of finite-dimensional Hilbert spaces, we can interpret any linear operator as a matrix with respect to a chosen basis. Moving forward, we will implicitly represent operators as matrices with respect to the standard basis. This approach facilitates the application of matrix operations, such as the conjugate transpose, simplifying our computations and discussions.

Definition 2.26. *Let $\mathcal{H}$ be a Hilbert Space and $\Pi : \mathcal{H} \to \mathcal{H}$ be an Operator. The operator Π is called a **Projection** if for all $u \in \mathcal{H}$ following holds:*

$$(\Pi \circ \Pi)(u) = \Pi(u),$$

Definition 2.27. *Let $\mathcal{H}$ be a Hilbert Space with a scalar product $\langle \cdot, \cdot \rangle$ and its induced norm. A surjective operator $U : \mathcal{H} \to \mathcal{H}$ is called **unitary** if:*

$$||U(x)|| = ||x||$$

This property characterizes a unitary operator as an isometry. For simplicity and to ensure we deal with square matrices, we will consider unitary operators $U : \mathcal{H} \to \mathcal{H}$. A more general definition can be found in [ST94].

Remark 2.17 (Tensor Product of Hilbert Spaces)**.** *Let $\mathcal{H}_1$ and $\mathcal{H}_2$ be Hilbert spaces. The tensor product $\mathcal{H}_1 \otimes \mathcal{H}_2$ is given as the Hilbert space constructed from the vector space tensor product of $\mathcal{H}_1$ and $\mathcal{H}_2$ with an inner product defined as follows: for any $\psi_1, \phi_1 \in \mathcal{H}_1$ and $\psi_2, \phi_2 \in \mathcal{H}_2$,*

$$\langle \psi_1 \otimes \psi_2, \phi_1 \otimes \phi_2 \rangle = \langle \psi_1, \phi_1 \rangle_{\mathcal{H}_1} \cdot \langle \psi_2, \phi_2 \rangle_{\mathcal{H}_2},$$

and extended linearly to the entire space $\mathcal{H}_1 \otimes \mathcal{H}_2$. The resulting space $\mathcal{H}_1 \otimes \mathcal{H}_2$ is complete with respect to the norm induced by this inner product, making it a Hilbert space.

Remark 2.18 (Dirac Notation)**.** *Dirac notation, also known as bra-ket notation, is a standard mathematical notation used in quantum mechanics to represent quantum states and their inner products. In this notation:*

- *A **state vector** $|u\rangle \in \mathcal{H}$ is a vector that represents the state of a quantum system within a Hilbert space $\mathcal{H}$. These vectors encapsulate all the information about the quantum state (see [Gri12a]). It is denoted by $|u\rangle$ and called a "ket"*
- *The dual vector (or conjugate transpose) of $|u\rangle$ is denoted by $\langle u|$, called a "bra".*
- *The inner product of two state vectors $|u\rangle$ and $|v\rangle$ is written as $\langle u|v\rangle$, which is equivalent to the standard inner product $\langle u, v\rangle$ in the Hilbert space $\mathcal{H}$.*

Also, Dirac notation is particularly useful when dealing with tensor products of Hilbert spaces. If $|x\rangle \in \mathcal{H}_1$ and $|y\rangle \in \mathcal{H}_2$, the tensor product of these states is denoted by $|x\rangle \otimes |y\rangle$, which can also be written as $|xy\rangle$ or $|x\rangle\,|y\rangle$.
A detailed explanation of this notation can be found in [Gri12a, NC10].

Example 2.2. *Let $\mathcal{H} := (\mathbb{C}^2)^{\otimes n}$ be a Hilbert space, which is the tensor product of n copies of $\mathbb{C}^2$. This space is equipped with the standard scalar product (see Example 2.1). Explicitly, the tensor product space $(\mathbb{C}^2)^{\otimes n}$ is constructed as follows:*

$$(\mathbb{C}^2)^{\otimes n} := \underbrace{\mathbb{C}^2 \otimes \mathbb{C}^2 \otimes \cdots \otimes \mathbb{C}^2}_{n \text{ times}} := \bigotimes_{i=1}^{n} \mathbb{C}^2$$

The set of all binary strings of length n, denoted by $\{0,1\}^n$, can be associated with the orthonormal basis vectors in this space. These binary strings are in the classical set $\{0,1\}^n$, which can be mapped to the quantum states in the Hilbert space $(\mathbb{C}^2)^{\otimes n}$.

Specifically, each binary string $b = b_1 b_2 \ldots b_n \in \{0,1\}^n$ corresponds to a basis vector $|b_1\rangle \otimes |b_2\rangle \otimes \cdots \otimes |b_n\rangle \in \mathcal{H}$, which we can write more compactly as $|b_1 b_2 \ldots b_n\rangle$. Here, each $|b_i\rangle$ is an element of the standard basis for $\mathbb{C}^2$, typically denoted as $|0\rangle$ and $|1\rangle$.

Explicitly, this mapping $f : \{0,1\}^n \to (\mathbb{C}^2)^{\otimes n}$ can be defined by:

$$f(b) = |b_1\rangle \otimes |b_2\rangle \otimes \cdots \otimes |b_n\rangle$$

where $b = b_1 b_2 \ldots b_n \in \{0,1\}^n$ and each $|b_i\rangle \in \mathbb{C}^2$ is either $|0\rangle$ or $|1\rangle$ depending on whether $b_i = 0$ or $b_i = 1$ for $i \in \{1, \ldots, n\}$. This mapping is bijective, meaning each binary string

uniquely corresponds to a distinct quantum state, and vice versa.

For example, the binary string $00\ldots0$ *corresponds to the basis vector* $|0\rangle\otimes|0\rangle\otimes\cdots\otimes|0\rangle = |00\ldots0\rangle$. *Similarly, the binary string* $00\ldots1$ *corresponds to* $|0\rangle\otimes|0\rangle\otimes\cdots\otimes|1\rangle = |00\ldots1\rangle$, *and so on, up to* $|1\rangle\otimes|1\rangle\otimes\cdots\otimes|1\rangle = |11\ldots1\rangle$. *These vectors form an orthonormal basis known as the* ***computational basis*** *(see [NC10]). The computational basis is fundamental in quantum computing as it allows us to represent and manipulate quantum states using classical binary strings, making the connection between quantum algorithms and classical computation.*

Theorem 2.10 (Properties of Unitary Operators)**.** *Let* $\mathcal{H}\subseteq\mathbb{C}^n$ *be a Hilbert space with the standard scalar product. Let* $U:\mathcal{H}\to\mathcal{H}$ *be an operator. Then the following statements are equivalent:*

(i) *U is a unitary operator.*

(ii) $U^*U = UU^* = I$.

(iii) *For all* $x, y\in\mathcal{H}$: $\langle Ux, Uy\rangle = \langle x, y\rangle$.

(iv) *U maps an orthonormal basis for* $\mathcal{H}$ *onto an orthonormal basis.*

Proof. (i) $\Rightarrow$ (ii) Assume U is a unitary operator. By the Definition 2.27 of a unitarity operator, U is surjective and preserves the norm. We know that $||Ux||^2 = \langle Ux, Ux\rangle$ for any x, and by unitarity, this equals $||x||^2 = \langle x, x\rangle$. Therefore, $\langle Ux, Ux\rangle = \langle x, x\rangle$. From the property of the adjoint, we have $\langle Ux, y\rangle = \langle x, U^*y\rangle$ for all x, y. By setting $y = Ux$, we get $\langle Ux, Ux\rangle = \langle x, U^*Ux\rangle$. Given $\langle Ux, Ux\rangle = \langle x, x\rangle$, it follows that $\langle x, U^*Ux\rangle = \langle x, x\rangle$, implying $U^*U = I$. A similar argument shows $UU^* = I$.

(ii) $\Rightarrow$ (iii) Given $U^*U = UU^* = I$, for any $x, y\in\mathcal{H}$, we have:

$$\langle Ux, Uy\rangle = \langle x, U^*Uy\rangle = \langle x, y\rangle.$$

This uses the property of the adjoint and the given condition $U^*U = I$.

(iii) $\Rightarrow$ (i) Assume for all $x, y\in\mathcal{H}$, $\langle Ux, Uy\rangle = \langle x, y\rangle$. Taking $y = x$, we get $||Ux||^2 = ||x||^2$, which means $||Ux|| = ||x||$. This directly implies U is a unitary operator by the given definition.

(iii) $\Rightarrow$ (iv) Given $\langle Ux, Uy\rangle = \langle x, y\rangle$ for all $x, y\in\mathcal{H}$, let $\{e_1,\ldots,e_n\}$ be an orthonormal basis for $\mathcal{H}$. Then for any e_i, e_j in the basis, $\langle Ue_i, Ue_j\rangle = \langle e_i, e_j\rangle = \delta_{ij}$, where δ_{ij} is the Kronecker delta. This shows U maps orthonormal bases to orthonormal bases.

(iv) $\Rightarrow$ (iii) Assume U maps an orthonormal basis $\{e_1,\ldots,e_n\}$ for $\mathcal{H}$ onto another orthonormal basis $\{Ue_1,\ldots,Ue_n\}$. For any $x, y\in\mathcal{H}$, they can be written as $x = \sum_i \alpha_i e_i$

and $y = \sum_j \beta_j e_j$, where $\alpha_i, \beta_j \in \mathbb{C}$. Then:

$$\begin{aligned}\langle Ux, Uy\rangle &= \left\langle U\left(\sum_i \alpha_i e_i\right), U\left(\sum_j \beta_j e_j\right)\right\rangle \\ &= \left\langle \sum_i \alpha_i U e_i, \sum_j \beta_j U e_j \right\rangle \\ &= \sum_{i,j} \alpha_i \overline{\beta_j} \langle Ue_i, Ue_j\rangle.\end{aligned}$$

Given $\langle Ue_i, Ue_j\rangle = \delta_{ij}$, this simplifies to $\sum_i \alpha_i \overline{\beta_i}$, which is exactly $\langle x, y\rangle$.

□

The following features will be important to characterize quantum gates later on.

Definition 2.28. *Let $U(n)$ denote the set of all unitary matrices in $\mathbb{C}^{n\times n}$, defined by:*

$$U(n) := \{U \in \mathbb{C}^{n\times n} \mid U^*U = UU^* = I\},$$

where U^* is the conjugate transpose of U, and I is the identity matrix.

Lemma 2.1 (Unitary Group)**.** *$U(n)$ forms a group under matrix multiplication.*

Proof. Let U and V be unitary operators, then their product UV is a unitary operator:

$$(UV)^*(UV) = V^*U^*UV = V^*(U^*U)V = V^*IV = V^*V = I,$$

where I is the identity matrix. Hence, the operation of multiplication is closed for unitary operators and associative by definition. The identity matrix I is a unitary operator and therefore denotes the identity element. Since it is $U^*U = UU^* = I$, the inverse of a unitary operator U is its conjugate transpose U^*, which is also unitary. □

Note that even $U(n)$ forms a group under matrix multiplication, it does not form a subvector space for the vector space of matrices: Multiplying a unitary matrix by a scalar typically results in a matrix that is not unitary. The sum of two unitary matrices is not necessarily unitary.

Definition 2.29 (Special Unitary Group)**.** *The special unitary group $SU(n)$ is defined as:*

$$SU(n) := \{U \in U(n) \mid \det(U) = 1\}.$$

Lemma 2.2. *$SU(n)$ is a subgroup of the unitary group $U(n)$.*

Proof. To show that $SU(n)$ is a subgroup of $U(n)$, we need to verify that it satisfies the subgroup criteria: Let $U, V \in SU(n)$. Then U and V are unitary, so UV is unitary with Lemma 2.1, and we have

$$\det(UV) = \det(U)\det(V) = 1 \cdot 1 = 1.$$

Hence, $UV \in SU(n)$. The identity matrix I is unitary and $\det(I) = 1$, so $I \in SU(n)$. If $U \in SU(n)$, then U is unitary, so U^* is its inverse in $U(n)$. Also,

$$\det(U^*) = \overline{\det(U)} = \overline{1} = 1.$$

Therefore, $U^* \in SU(n)$. Since $SU(n)$ satisfies closure, contains the identity element, and is closed under inverses, it is a subgroup of $U(n)$. □

Remark 2.19. *The special unitary group $SU(n)$ is particularly relevant in quantum mechanics and quantum computing because it describes unitary transformations that preserve not only the norm (as with $U(n)$) but also the global phase. Since global phase factors do not affect physical measurements, quantum gates are often modeled within $SU(n)$ to exclude these irrelevant phases. Moreover, $SU(n)$ plays a central role in the description of symmetries in quantum field theory and the Standard Model of particle physics. [NC10]*

2.4.1.2 Qubits and Quantum Information

In quantum computing, the fundamental unit of information is the qubit. The mathematical representation of a qubit's state space (see Example 2.1) is the Hilbert space $\mathcal{H}_2$. We use notation and formalism from [NC10, FGG14].

Definition 2.30 (Single-Qubit Hilbert Space)**.** *The Hilbert space $\mathcal{H}_2$ of a single qubit is a two-dimensional complex vector space, equipped with the standard scalar product:*

$$\mathcal{H}_2 = \mathbb{C}^2 = span\{|0\rangle, |1\rangle\},$$

where $|0\rangle$ and $|1\rangle$ are the standard basis vectors given as:

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

The state of a single qubit $|\psi\rangle \in \mathcal{H}_2$ can be expressed as a linear combination of these basis vectors:

Definition 2.31 (Qubit)**.** *Let $\mathcal{H}_2$ be the Hilbert space of a qubit. The state $|\psi\rangle \in \mathcal{H}_2$ is called a **qubit**:*

$$|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle,$$

where $\alpha_0, \alpha_1 \in \mathbb{C}$ are called amplitudes. The state vector $|\psi\rangle$ satisfies the normalization condition $\langle\psi|\psi\rangle = 1$, where $\langle\cdot|\cdot\rangle$ denotes the standard scalar product in $\mathcal{H}_2$. This ensures that the total probability of measuring the state equals one, implying $|\alpha_0|^2 + |\alpha_1|^2 = 1$.

Remark 2.20. *Note that the normalization condition $\langle\psi|\psi\rangle = 1$ follows from the State Postulate (see [NC10]).*

In a quantum system consisting of multiple qubits, the overall state space is constructed by taking the tensor product of the individual qubit spaces, which naturally inherits the standard scalar product.

Definition 2.32 (Multi-Qubit Hilbert Space)**.** *For an n-qubit system, the Hilbert space $\mathcal{H}_2^{\otimes n}$ is given by the tensor product:*

$$\mathcal{H}_2^{\otimes n} = (\mathbb{C}^2)^{\otimes n},$$

where each $\mathcal{H}_2$ is the two-dimensional Hilbert space associated with a single qubit. The tensor product space $\mathcal{H}_2^{\otimes n}$ is a 2^n-dimensional complex vector space, equipped with the standard scalar product inherited from $\mathcal{H}_2$ (see Definition 2.30).

Note that in some contexts, the Hilbert space $\mathcal{H}_2^{\otimes n}$ is equivalently denoted as $(\mathbb{C}^2)^{\otimes n}$. Both notations represent the same 2^n-dimensional complex vector space, which is the tensor product of n individual qubit Hilbert spaces.

Remark 2.21 (Relation Between $\mathcal{H}_2$ and $\mathcal{H}_2^{\otimes n}$)**.** *The multi-qubit Hilbert space $\mathcal{H}_2^{\otimes n}$ is constructed by taking the tensor product of n individual qubit spaces $\mathcal{H}_2$, each with its standard scalar product. This results in a combined space where the scalar product of two states $|\Phi\rangle, |\Psi\rangle \in \mathcal{H}_2^{\otimes n}$ is naturally defined as the product of the scalar products on each*

factor of the tensor product. An arbitrary n-qubit state $|\Phi\rangle \in \mathcal{H}_2^{\otimes n}$ *can be expressed in the computational basis (see Definition 2.33) as:*

$$|\Phi\rangle = \sum_{x \in \{0,1\}^n} \alpha_x |x\rangle,$$

where $\alpha_x \in \mathbb{C}$ *represents the amplitude associated with the basis state* $|x\rangle$*, and* $|x\rangle$ *is a computational basis vector formed by the tensor product of individual qubit states* $|0\rangle$ *and* $|1\rangle$ *as described in Definition 2.30.*

The computational basis for an n-qubit system is defined as follows:

Definition 2.33 (Computational Basis)**.** *Let* $\mathcal{H}_2^{\otimes n}$ *be the Hilbert space of an n-qubit quantum system. The* ***computational basis*** $\mathcal{B}_c^n$ *of* $\mathcal{H}_2^{\otimes n}$ *is the set of basis vectors:*

$$\mathcal{B}_c^n := \left\{ \bigotimes_{i=1}^{n} |b_i\rangle \mid b_i \in \{0,1\} \text{ for } i = 1, 2, \ldots, n \right\},$$

where each $|b_i\rangle \in \mathcal{H}_2$ *for* $i \in \{1, 2, \ldots, n\}$ *is one of the standard basis vectors:*

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

The scalar product between any two states in $\mathcal{H}_2^{\otimes n}$ *is the standard scalar product inherited from the individual qubit spaces* $\mathcal{H}_2$ *as described in Remark 2.17.*

Lemma 2.3. *The computational basis* $\mathcal{B}_c^n$ *defined for the n-qubit Hilbert space* $\mathcal{H}_2^{\otimes n}$ *forms a basis for* $\mathcal{H}_2^{\otimes n}$*.*

Proof. Let $\mathcal{B}_c^n$ be the computational basis defined for the n-qubit Hilbert space $\mathcal{H}_2^{\otimes n}$. Consider the following linear combination of the basis vectors in $\mathcal{B}_c^n$:

$$\sum_{x \in \{0,1\}^n} \alpha_x \, |x\rangle = 0,$$

where $|x\rangle = \bigotimes_{i=1}^{n} |b_i\rangle \in \mathcal{B}_c^n$ with $x = (b_1, b_2, \ldots, b_n)$, and $\alpha_x \in \mathbb{C}$ are coefficients. Since the vectors $|x\rangle$ are distinct and represent different binary combinations of the states $|0\rangle$ and $|1\rangle$, the only solution to this equation (by the property of linear independence of vectors in $\mathbb{C}^2$) is $\alpha_x = 0$ for all x. Therefore, the vectors in $\mathcal{B}_c^n$ are linearly independent. The dimension of the Hilbert space $\mathcal{H}_2^{\otimes n}$ is 2^n, because $\mathcal{H}_2^{\otimes n}$ is the tensor product of n 2-dimensional spaces (each corresponding to a qubit). The computational basis $\mathcal{B}_c^n$ contains exactly 2^n vectors, as there are 2^n distinct binary strings of length n. Since $\mathcal{B}_c^n$ contains 2^n linearly independent vectors in a 2^n-dimensional space, it must span the entire space. □

Since a qubit state $|\psi\rangle$ belongs to the Hilbert space $\mathcal{H}_2$, it can be expressed as a linear combination of the computational basis states $|0\rangle$ and $|1\rangle$, with two complex coefficients. This representation allows the qubit to be mapped onto the surface of a sphere known as the **Bloch Sphere**. On the Bloch Sphere, the poles correspond to the basis states $|0\rangle$ and $|1\rangle$, while any superposition of these states can be visualized as a point on the sphere, uniquely determined by the angles θ and ϕ.

Remark 2.22 (The Bloch Sphere)**.** *Consider the Hilbert space $\mathcal{H}_2$ for a single qubit. Given the computational basis (see Definition 2.33), a single qubit state $|\psi\rangle \in \mathcal{H}_2$ can be expressed as a linear combination of the basis states $|0\rangle$ and $|1\rangle$ with complex coefficients $\alpha_0, \alpha_1 \in \mathbb{C}$, as described in Definition 2.31. Introducing θ and ϕ as the polar and azimuthal angles in spherical coordinates, these angles determine the qubit's state on the surface of the Bloch sphere, where $\theta \in [0, \pi]$ and $\phi \in [0, 2\pi)$. Under the normalization condition $\langle\psi|\psi\rangle = 1$, the coefficients are given by:*

$$\alpha_0 = \cos\left(\frac{\theta}{2}\right), \quad \alpha_1 = e^{i\phi}\sin\left(\frac{\theta}{2}\right).$$

Here, θ controls the relative contribution of $|0\rangle$ and $|1\rangle$ in the superposition, effectively determining the latitude on the Bloch sphere (see Figure 2.3). The angle ϕ introduces a relative phase between the basis states, corresponding to a rotation around the z-axis of the sphere.

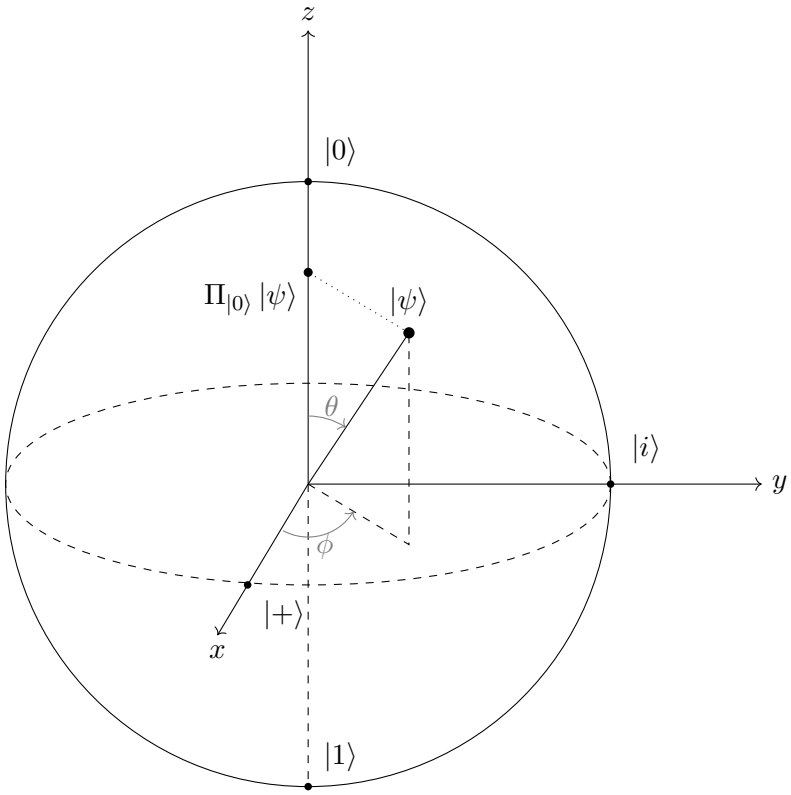

Figure 2.3: Visualization of an arbitrary single qubit state $|\psi\rangle$ as described in Remark 2.22 on the surface of the Bloch sphere, with corresponding angles θ and ϕ. The $|+\rangle$ and $|i\rangle$ states (see [NC10]) are on the x-axis and y-axis and the projection $\Pi_{|0\rangle}|\psi\rangle$ of $|\psi\rangle$ onto $|0\rangle$ on the z-axis. The poles of $|1\rangle$ and $|0\rangle$ on the z-axis represent the standard basis states. [RB20]

2.4.1.3 From Unitary Operators to Quantum Gates

In the domain of quantum computing, the fundamental operations on quantum data are represented by quantum gates, which form the structure of a quantum circuit. Qubits are manipulated and transformed by these gates to perform quantum computations, which are then measured at the end. Gates are expressed as linear operators acting on qubit states.

Definition 2.34 (Pauli Operators)**.** *The set of unitary operators $\{I, X, Y, Z\}$, where I is the identity and X, Y, Z are the three Pauli matrices, are called **Pauli Operators** or Pauli Basis:*

$$I := \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad X := \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y := \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z := \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

Remark 2.23. *Among the Pauli operators $\{I, X, Y, Z\}$, only the identity matrix I and the Pauli-Y matrix Y are elements of the special unitary group $SU(2)$ (see Definition 2.29), as they are unitary with a determinant of 1. The Pauli-X and Pauli-Z matrices, while unitary, have a determinant of -1 and therefore belong to the larger group $U(2)$ (see Definition 2.28).*

Note that the Pauli operators are sometimes denoted as σ^Z for the Pauli Z-operator, σ^X for the Pauli X-operator, and σ^Y for the Pauli Y-operator. With Lemma 2.3, we have already shown that the set of linear operators form a vector space, therefore we can further investigate some basis properties:

Lemma 2.4. *Let $\mathcal{H}$ be a 2-dimensional Hilbert space. The Pauli operators $\{I, X, Y, Z\}$ form a basis for the 4-dimensional space of linear operators $\mathcal{L}(\mathcal{H})$ on $\mathcal{H}$.*

Proof. Let $\mathcal{H}$ be a 2-dimensional Hilbert space, and let $A \in \mathcal{L}(\mathcal{H})$ be a linear operator on $\mathcal{H}$. We identify A as a 2×2 matrix. Therefore it can be written as:

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}, \tag{2.17}$$

where $a, b, c, d \in \mathbb{C}$. We want to express A as a linear combination of the Pauli operators:

$$A = \alpha I + \beta X + \gamma Y + \delta Z,$$

where $\alpha, \beta, \gamma, \delta \in \mathbb{C}$ are complex coefficients, which gives us:

$$\alpha \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} + \beta \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} + \gamma \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} + \delta \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = \begin{pmatrix} \alpha + \delta & \beta - i\gamma \\ \beta + i\gamma & \alpha - \delta \end{pmatrix}.$$

Since this expression needs to be equal to 2.17, we can solve for the coefficients and get:

$$\alpha = \frac{a+d}{2}, \quad \delta = \frac{a-d}{2}, \quad \beta = \frac{b+c}{2}, \quad \gamma = \frac{c-b}{2i}.$$

Hence A can be written as a linear combination of I, X, Y, and Z. To prove the linear

independence we suppose:

$$\begin{pmatrix} \alpha+\delta & \beta-i\gamma \\ \beta+i\gamma & \alpha-\delta \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}.$$

Therefore, we have the following equations:

$$\alpha+\delta=0, \quad \beta-i\gamma=0, \quad \beta+i\gamma=0, \quad \alpha-\delta=0.$$

Solving the first and last results in $\alpha=\delta=0$. With the second and third, it is $\beta=0$ and $\gamma=0$. Thus, linear independence is satisfied. Since the Pauli operators $\{I,X,Y,Z\}$ span the space of 2×2 matrices and are linearly independent, they form a basis for $\mathcal{M}(2,\mathbb{C})$ and with Corollary 2.8 also a basis for $\mathcal{L}(\mathcal{H})$. □

Lemma 2.5. *Let $\mathcal{H}_1$ and $\mathcal{H}_2$ be Hilbert spaces. The space of linear operators $\mathcal{L}(\mathcal{H}_1\otimes\mathcal{H}_2)$ on the tensor product of Hilbert spaces $\mathcal{H}_1\otimes\mathcal{H}_2$ is isomorphic to the tensor product of the spaces of linear operators on each Hilbert space:*

$$\mathcal{L}(\mathcal{H}_1\otimes\mathcal{H}_2)\cong\mathcal{L}(\mathcal{H}_1)\otimes\mathcal{L}(\mathcal{H}_2).$$

Proof. Consider two Hilbert spaces $\mathcal{H}_1$ and $\mathcal{H}_2$ with orthonormal bases $\{|i\rangle\}$ for $\mathcal{H}_1$ and $\{|j\rangle\}$ for $\mathcal{H}_2$. The tensor product space $\mathcal{H}_1\otimes\mathcal{H}_2$ has a basis $\{|i\rangle\otimes|j\rangle\}$.
An arbitrary linear operator A on $\mathcal{H}_1\otimes\mathcal{H}_2$ can be written in terms of these basis elements:

$$A=\sum_{i,j,k,l} A_{ij,kl}(|i\rangle\otimes|j\rangle)(\langle k|\otimes\langle l|),$$

where $A_{ij,kl}\in\mathbb{C}$ are complex coefficients. For $A_1\in\mathcal{L}(\mathcal{H}_1)$ and $A_2\in\mathcal{L}(\mathcal{H}_2)$, consider the following map:

$$\phi:\mathcal{L}(\mathcal{H}_1)\otimes\mathcal{L}(\mathcal{H}_2)\to\mathcal{L}(\mathcal{H}_1\otimes\mathcal{H}_2), \quad \phi(A_1\otimes A_2)(|u\rangle\otimes|v\rangle)=(A_1|u\rangle)\otimes(A_2|v\rangle).$$

This map is linear. Therefore, any basis of $\mathcal{L}(\mathcal{H}_1)$ and $\mathcal{L}(\mathcal{H}_2)$ gets mapped onto a basis for $\mathcal{L}(\mathcal{H}_1\otimes\mathcal{H}_2)$. For any $A\in\mathcal{L}(\mathcal{H}_1\otimes\mathcal{H}_2)$, we can express A in the form:

$$A=\sum_{i,k}(A_1)_{ik}|i\rangle\langle k|\otimes\sum_{j,l}(A_2)_{jl}|j\rangle\langle l|.$$

Thus, any operator A on $\mathcal{H}_1\otimes\mathcal{H}_2$ can be viewed as an element of $\mathcal{L}(\mathcal{H}_1)\otimes\mathcal{L}(\mathcal{H}_2)$, making ϕ bijective. □

Multi-qubit gates or operators can be constructed from tensor products of the Pauli Basis. The set of matrices obtained from all possible tensor products of Pauli matrices (including the identity) for n qubits forms a basis for the $2^n\times 2^n$ matrices, which describe operations on n-qubit systems. For n qubits, the Pauli basis consists of all possible n-fold tensor products of the single-qubit Pauli matrices:

$$\{I,X,Y,Z\}^{\otimes n}$$

This set contains 4^n elements because there are 4 choices of the Pauli matrices for each

of the n qubits.

Theorem 2.11. *Let $\mathcal{H}$ be a 2^n-dimensional Hilbert space, then $\{I, X, Y, Z\}^{\otimes n}$ forms a basis for the 4^n-dimensional space of linear operators $\mathcal{L}(\mathcal{H})$ on $\mathcal{H}$.*

Proof. We prove the statement by induction on the number of qubits $n \in \mathbb{N}$.
Base Case: For $n = 1$, $\mathcal{H}$ is a 2-dimensional Hilbert space. The Pauli operators $\{I, X, Y, Z\}$ form a basis for the 4-dimensional space of linear operators $\mathcal{L}(\mathcal{H})$ on $\mathcal{H}$.

Induction Hypothesis: Assume that for $(n-1)$ qubits, the Pauli operators of the tensor product $\{I, X, Y, Z\}^{\otimes(n-1)}$ form a basis for the 4^{n-1}-dimensional space of linear operators $\mathcal{L}(\mathcal{H})$.

Induction Step: Consider an n-qubit Hilbert space $\mathcal{H}$. We want to show that the Pauli operators $\{I, X, Y, Z\}^{\otimes n}$ form a basis for the 4^n-dimensional space of linear operators $\mathcal{L}(\mathcal{H})$. Given that $\mathcal{H}$ is a tensor product of n 2-dimensional Hilbert spaces, $\mathcal{H} = \mathcal{H}_1 \otimes \mathcal{H}_2 \otimes \cdots \otimes \mathcal{H}_n$. By the induction hypothesis, $\{I, X, Y, Z\}^{\otimes(n-1)}$ forms a basis for the space of linear operators on the $(n-1)$-qubit Hilbert space, which is 2^{n-1}-dimensional.
The Pauli operators for n qubits $\{I, X, Y, Z\}^{\otimes n}$ can be expressed as tensor products:

$$\{I, X, Y, Z\}^{\otimes n} = \{I, X, Y, Z\}^{\otimes(n-1)} \otimes \{I, X, Y, Z\}$$

Since $\{I, X, Y, Z\}^{\otimes(n-1)}$ forms a basis for the 4^{n-1}-dimensional space of linear operators on the $(n-1)$-qubit Hilbert space, and $\{I, X, Y, Z\}$ forms a basis for the 4-dimensional space of linear operators on a 2-dimensional Hilbert space, their tensor product $\{I, X, Y, Z\}^{\otimes n}$ forms a basis for the 4^n-dimensional space of linear operators on an n-qubit Hilbert space.
Thus, by induction, the Pauli operators $\{I, X, Y, Z\}^{\otimes n}$ form a basis for the 4^n-dimensional space of linear operators $\mathcal{L}(\mathcal{H})$ on the 2^n-dimensional Hilbert space $\mathcal{H}$.

□

Remark 2.24 (Quantum Gate/Quantum Operator)**.** *A quantum gate or quantum operator can be understood as a unitary operator $U : \mathcal{H} \to \mathcal{H}$ that acts on a vector in a Hilbert space $\mathcal{H}$ of dimension 2^n for a system of n qubits. This means it is a unitary element of $\mathcal{L}(\mathcal{H})$, the space of linear operators on the Hilbert space $\mathcal{H}$. Quantum gates are unitary transformations, which preserve the norm of probability amplitudes, ensuring that the probabilities of measurement outcomes sum to 1 (see Theorem 2.10). Therefore, a quantum gate can be represented as a unitary matrix. The action of quantum gates is described using matrix multiplication, and the effect of sequences of gates is computed by the product of their matrices. This framework forms the basis for constructing quantum circuits and performing quantum computations.*

For explicit computations, it is beneficial to have a convenient representation of the matrix exponential of the Pauli-Z operator. Therefore, consider the following lemma:

Lemma 2.1. *Let $\phi \in (-\infty, \infty)$ and Z be the Pauli-Z operator, then the matrix exponential can be expressed as:*

$$e^{i\phi Z} = \begin{pmatrix} e^{i\phi} & 0 \\ 0 & e^{-i\phi} \end{pmatrix}.$$

Proof. Let $\phi \in \mathbb{R}$ and Z be the Pauli-Z operator. Because the Pauli Z-Operator is Hermitian and unitary, it is $Z \cdot Z = \bar{Z}^T Z = I$, the identity. Therefore, if $k = 2n$ for some $n \in \mathbb{N}$, then $Z^{2n} = (Z^2)^n = I^n = I$, and if $k = 2n + 1$ for some $n \in \mathbb{N}$, then $Z^{2n+1} = (Z^2)^n \cdot Z = I^n \cdot Z = Z$. Because $e^{i\phi Z}$ is converging, we can split the series into its even and odd terms and receive:

$$e^{i\phi Z} = \sum_{k=0}^{\infty} \frac{(i\phi Z)^{2k}}{(2k)!} + \sum_{k=0}^{\infty} \frac{(i\phi Z)^{2k+1}}{(2k+1)!} = \cos(\phi) I + i \sin(\phi) Z$$

This can be further simplified using Euler's formula:

$$e^{i\phi Z} = \begin{pmatrix} \cos(\phi) + i\sin(\phi) & 0 \\ 0 & \cos(\phi) - i\sin(\phi) \end{pmatrix} = \begin{pmatrix} e^{i\phi} & 0 \\ 0 & e^{-i\phi} \end{pmatrix}$$

□

Note that a similar form of this lemma can be shown for all other Pauli operators, as they are all Hermitian and implicitly unitary. However, we will not require these results for our current calculations, so we omit the details.

2.4.2 Quantum Signal Processing

Quantum Signal Processing (QSP) was introduced by [Low17] as a generalization of composite pulse sequences. The core concept behind QSP involves the strategic alternation between two distinct types of single-qubit rotation operations.

Definition 2.35 (Signal Rotation Operator)**.** *The unitary operator W, referred to as the signal rotation operator, is represented as follows for $a \in [-1, 1]$:*

$$W(a) = \begin{pmatrix} a & i\sqrt{1-a^2} \\ i\sqrt{1-a^2} & a \end{pmatrix}, \tag{2.18}$$

The signal rotation operator performs a rotation around the x-axis in the Bloch sphere, with an angle of $\theta = -2\arccos(a)$.

Definition 2.36 (Signal Processing Rotation)**.** *The operator S, defined as the matrix exponential of the Pauli Z-operator, is called the signal processing rotation. For $\phi \in (-\infty, \infty)$ (the rotation angle), it is given as follows:*

$$S(\phi) = e^{i\phi Z}, \tag{2.19}$$

The signal processing rotation executes a rotation around the z-axis of the Bloch sphere, with the effective rotation angle being -2ϕ. Note that the signal rotation operator W has a consistent rotation angle θ, whereas the signal processing rotation S has a rotation angle ϕ adjusted based on a specifically chosen input sequence as follows:

Definition 2.37 (QSP Operation Sequence)**.** *A sequence of rotations, represented by a tuple of phase angles $\vec{\phi} = (\phi_0, \phi_1, \ldots, \phi_d) \in \mathbb{R}^{d+1}$ referred to as signals, defines the QSP operation sequence $U_{\vec{\phi}}$ as follows:*

$$U_{\vec{\phi}} := e^{i\phi_0 Z} \prod_{k=1}^{d} W(a) e^{i\phi_k Z}. \tag{2.20}$$

This sequence combines a set of given signals and converts them into processing rotations that can be used to approximate a polynomial. Utilizing the above characterization, Quantum Signal Processing (QSP) can be formulated as described in Theorem 1 from [MRTC21] and Theorem 3 from [GSLW18]. The QSP operation sequence $U_{\vec{\phi}}$, through its structured composition of signal and processing rotations, generates a matrix that can be expressed as a polynomial function of a, in the form:

Theorem 2.12 (Quantum Signal Processing)**.** *There exist signals $\vec{\phi} = (\phi_0, \phi_1, \ldots, \phi_d) \in \mathbb{R}^{d+1}$ such that for $a \in [-1, 1]$:*

$$U_{\vec{\phi}} = \begin{pmatrix} P(a) & iQ(a)\sqrt{1-a^2} \\ iQ^*(a)\sqrt{1-a^2} & P^*(a) \end{pmatrix},$$

where $P, Q \in \mathbb{C}[X]$ are polynomials satisfying the following conditions:

1. *The polynomials are of degree $deg(P) \leq d$ and $deg(Q) \leq d-1$.*
2. *The polynomial P has parity $d \mod 2$, and Q has parity $(d-1) \mod 2$.*

3. *For all* $a \in [-1,1] : |P(a)|^2 + (1-a^2)|Q(a)|^2 = 1$.

Proof. We use the idea presented in [GSLW18] to prove the theorem in both directions. " $\implies$ ": We prove the statement by induction over $d \in \mathbb{N}_0$. Let $\vec{\phi} \in \mathbb{R}^{d+1}$ be some arbitrary signals.
Base Case: Let $d = 0$. Obeying the convention that the empty product is equal to one and applying Lemma 2.1, the QSP Sequence is given by:

$$U_{\vec{\phi}} = e^{i\phi_0 Z} \prod_{k=1}^{0} W(a) e^{i\phi_k Z} = e^{i\phi_0 Z} \cdot 1 = \begin{pmatrix} e^{i\phi_0} & 0 \\ 0 & e^{-i\phi_0} \end{pmatrix}$$

with $P \equiv e^{i\phi_0}$ and $Q \equiv 0$ satisfying conditions 1-3.
Induction Hypothesis: Assume the proposition holds for some $d \in \mathbb{N}_0$, i.e., the QSP sequence $U_{\vec{\phi}}$ generates the defined matrix with $P, Q \in \mathbb{C}[X]$ satisfying conditions 1-3.
Induction Step: $d \mapsto d+1$ Thus we consider a signal $\vec{\phi} = (\phi_0, \phi_1, \ldots, \phi_{d+1}) \in \mathbb{R}^{(d+1)+1}$. We obtain the following QSP operation sequence using Lemma 2.1:

$$\begin{aligned}
U_{\vec{\phi}} &= e^{i\phi_0 Z} \prod_{k=1}^{d+1} W(a) e^{i\phi_k Z} = e^{i\phi_0 Z} \prod_{k=1}^{d} \left(W(a) e^{i\phi_k Z} \right) W(a) e^{i\phi_{d+1} Z} \\
&=^{(IH)} \begin{pmatrix} P(a) & iQ(a)\sqrt{1-a^2} \\ iQ^*(a)\sqrt{1-a^2} & P^*(a) \end{pmatrix} \begin{pmatrix} a & i\sqrt{1-a^2} \\ i\sqrt{1-a^2} & a \end{pmatrix} \begin{pmatrix} e^{i\phi_{d+1}} & 0 \\ 0 & e^{-i\phi_{d+1}} \end{pmatrix} \\
&= \begin{pmatrix} P(a) & iQ(a)\sqrt{1-a^2} \\ iQ^*(a)\sqrt{1-a^2} & P^*(a) \end{pmatrix} \begin{pmatrix} ae^{i\phi_{d+1}} & i\sqrt{1-a^2}e^{-i\phi_{d+1}} \\ i\sqrt{1-a^2}e^{i\phi_{d+1}} & ae^{-i\phi_{d+1}} \end{pmatrix} \\
&= \begin{pmatrix} \left(aP(a) + (a^2-1)Q(a)\right) e^{i\phi_{d+1}} & i\sqrt{1-a^2}\left(aQ(a) + P(a)\right) e^{-i\phi_{d+1}} \\ i\sqrt{1-a^2}\left(aQ^*(a) + P^*(a)\right) e^{i\phi_{d+1}} & \left(aP^*(a) + (a^2-1)Q^*(a)\right) e^{-i\phi_{d+1}} \end{pmatrix}
\end{aligned}$$

Therefore, we can denote the polynomials $P', Q' \in \mathbb{C}[X]$ of the $d+1$ iteration of the QSP Operation Sequence as:

$$\begin{aligned}
P'(a) &= \left(aP(a) + (a^2-1)Q(a)\right) e^{i\phi_{d+1}} \\
Q'(a) &= \left(aQ(a) + P(a)\right) e^{-i\phi_{d+1}}
\end{aligned}$$

These polynomials satisfy all the stated conditions 1-3 of the theorem:

1. Since the degree of $\deg(P) \leq d$ and $\deg(Q) \leq d-1$ we can conclude that:

$$\begin{aligned}
\deg(P') &= \max(\deg(P)+1, \deg(Q)+2) \leq d+1 \\
\deg(Q') &= \max(\deg(P), \deg(Q)+1) \leq d
\end{aligned}$$

2. The parity of $P'(a)$ and $Q'(a)$ follows from the parity of the original polynomials $P(a)$ and $Q(a)$. Which is defined by assumption with P having parity $d \mod 2$, and Q having parity $(d-1) \mod 2$.
3. Since for $U_{\vec{\phi}}$ the factors $W(a)$ and $e^{i\phi_k Z}$ for every $0 \leq k \leq d+1$ and $a \in [-1,1]$ are unitary, $U_{\vec{\phi}}$ describes a product of unitaries which is unitary according to Lemma 2.1. Therefore, the normalization condition is preserved according to Theorem 2.10.

" $\Longleftarrow$ ": We again prove the statement by induction over $deg(P) \in \mathbb{N}_0$. Let therefore $P', Q' \in \mathbb{C}[X]$ be given such that they follow conditions 1-3.
Base Case: Let $deg(P) = 0$. Using 3. for $a = 1$ it is

$$|P(1)|^2 + (1-1^2)|Q(1)|^2 = |P(1)|^2 = 1$$

therefore $|P(1)| = 1$. That means P is a constant polynomial of magnitude one therefore $P \equiv e^{i\phi_0}$ for some $\phi_0 \in \mathbb{R}$. And with 3. we get

$$|P(a)|^2 + (1-a^2)|Q(a)|^2 = 1 + (1-a^2)|Q(a)|^2 = 1 + (1-a^2)|Q(a)|^2 = 1$$

for any $a \in [-1, 1]$ and thus $Q \equiv 0$. Because P is even. we get $d \mod 2 = 0$, hence d is even and the signal $\vec{\phi} = (\phi_0, \frac{\pi}{2}, -\frac{\pi}{2}, \ldots, \frac{\pi}{2}, -\frac{\pi}{2}) \in \mathbb{R}^{d+1}$ is a solution, which is also valid if $d = 0$:

$$U_{\vec{\phi}} = e^{i\phi_0 Z} \prod_{k=1}^{d/2} \left(W(a) e^{i\phi_k Z} W(a) e^{-i\phi_k Z} \right) = e^{i\phi_0 Z} \cdot 1 = \begin{pmatrix} e^{i\phi_0} & 0 \\ 0 & e^{-i\phi_0} \end{pmatrix}$$

Induction Hypothesis: Assume the proposition holds for some the polynomials $P', Q' \in \mathbb{C}[X]$ where $deg(P) \in \mathbb{N}$, i.e., the polynomials satisfy 1-3 and are defined through a QSP Operation Sequence $U_{\vec{\phi}}$ with some $\vec{\phi} \in \mathbb{R}^{d+1}$.
Induction Step: $deg(P) \mapsto d+1$ Assume that $P, Q \in \mathbb{C}[X]$ are given with the leading coefficients p_ℓ and $q_{\ell-1}$, respectively. Assume without loss of generality that $1 \leq \deg(P) = \ell \leq d+1$. Then, we must have $\deg(Q) = \ell - 1$.
Note that property 3 from the theorem can be expressed for $a \in [-1, 1]$ as

$$|P(a)|^2 + (1-a^2)|Q(a)|^2 = 1 \quad \Leftrightarrow \quad P(a)P^*(a) + (1-a^2)Q(a)Q^*(a) = 1,$$

because $|P(a)|^2 = P(a) \cdot P^*(a)$ and $|Q(a)|^2 = Q(a) \cdot Q^*(a)$ for all $a \in [-1, 1]$, where $P^*(a)$ is the complex conjugate of $P(a)$ and $Q^*(a)$ is the complex conjugate of $Q(a)$, respectively. Since this property holds for infinitely many points, the polynomial on the left-hand side must be constant. Thus, we can write, with $R(a) := (1-a^2)$, for all $a \in [-1, 1]$:

$$PP^* + RQQ^* \equiv 1.$$

Note that in this term the highest order terms of the polynomials must cancel each other out. The highest degree term of $P(a)P^*(a)$ is $p_\ell \overline{p_\ell} a^{2\ell}$, where p_ℓ is the leading coefficient of $P(a)$. The leading term of $Q(a)Q^*(a)$ is $q_{\ell-1}\overline{q_{\ell-1}}a^{2(\ell-1)}$. When multiplied by $(1-a^2)$, the highest degree term in $(1-a^2)Q(a)Q^*(a)$ becomes:

$$-q_{\ell-1}\overline{q_{\ell-1}}a^{2\ell}.$$

Since the polynomial equation must hold identically for all $a \in [-1, 1]$, the highest degree terms on both sides of the equation must cancel each other out. Therefore, we must have:

$$p_\ell \overline{p_\ell} a^{2\ell} - q_{\ell-1}\overline{q_{\ell-1}}a^{2\ell} = 0 \quad \Leftrightarrow \quad p_\ell \overline{p_\ell} = q_{\ell-1}\overline{q_{\ell-1}} \quad \Leftrightarrow \quad |p_\ell| = |q_{\ell-1}|$$

Let $\phi'_{d+1} \in \mathbb{R}$ be such that $e^{2i\phi'_{d+1}} = \frac{p_\ell}{q_{\ell-1}}$, with Lemma 2.1 we can write:

$$\begin{pmatrix} P'(a) & iQ'(a)\sqrt{1-a^2} \\ iQ'^*(a)\sqrt{1-a^2} & P'^*(a) \end{pmatrix} := \begin{pmatrix} P(a) & iQ(a)\sqrt{1-a^2} \\ iQ^*(a)\sqrt{1-a^2} & P^*(a) \end{pmatrix} e^{-i\phi'_{d+1}Z} W^\dagger(a)$$
$$= \begin{pmatrix} P(a) & iQ(a)\sqrt{1-a^2} \\ iQ^*(a)\sqrt{1-a^2} & P^*(a) \end{pmatrix} \begin{pmatrix} e^{-i\phi'_{d+1}}a & -ie^{-i\phi'_{d+1}}\sqrt{1-a^2} \\ -ie^{i\phi'_{d+1}}\sqrt{1-a^2} & e^{i\phi'_{d+1}}a \end{pmatrix}$$
$$= \begin{pmatrix} e^{-i\phi'_{d+1}}aP(a) + e^{i\phi'_{d+1}}(1-a^2)Q(a) & i\left(e^{-i\phi'_{d+1}}aQ(a) - e^{i\phi'_{d+1}}P(a)\right)\sqrt{1-a^2} \\ i\left(e^{-i\phi'_{d+1}}aQ^*(a) - e^{i\phi'_{d+1}}P^*(a)\right)\sqrt{1-a^2} & e^{-i\phi'_{d+1}}aP^*(a) + e^{i\phi'_{d+1}}(1-a^2)Q^*(a) \end{pmatrix}$$

where

$$\begin{aligned} P'(a) &= e^{-i\phi'_{d+1}}aP(a) + e^{i\phi'_{d+1}}(1-a^2)Q(a) &&= e^{-i\phi'_{d+1}}\left(aP(a) + \tfrac{p_\ell}{q_{\ell-1}}(1-a^2)Q(a)\right) \\ Q'(a) &= e^{i\phi'_{d+1}}aQ(a) - e^{-i\phi'_{d+1}}P(a) &&= e^{-i\phi'_{d+1}}\left(\tfrac{p_\ell}{q_{\ell-1}}aQ(a) - P(a)\right) \end{aligned}$$

Note that the highest order terms P' and Q' cancel out again. Consider the leading terms in $P'(a)$: The leading term of $e^{-i\phi'_{d+1}}aP(a)$ is $e^{-i\phi'_{d+1}}a \cdot p_\ell a^\ell = e^{-i\phi'_{d+1}}p_\ell a^{\ell+1}$. The leading term of $e^{i\phi'_{d+1}}(1-a^2)Q(a)$ is given by:

$$e^{i\phi'_{d+1}}(1-a^2) \cdot q_{\ell-1}a^{\ell-1} = e^{i\phi'_{d+1}} \cdot q_{\ell-1}(a^{\ell-1} - a^{\ell+1})$$

Combining these, the highest order terms in $P'(a)$ are:

$$e^{-i\phi'_{d+1}}p_\ell a^{\ell+1} + e^{i\phi'_{d+1}}(-q_{\ell-1}a^{\ell+1})$$

Given that we already know that $|p_\ell| = |q_{\ell-1}|$ and assuming $p_\ell = q_{\ell-1}e^{i\phi'_{d+1}}$, we have:

$$\begin{aligned} e^{-i\phi'_{d+1}}p_\ell a^{\ell+1} - e^{i\phi'_{d+1}}q_{\ell-1}a^{\ell+1} &= e^{-i\phi'_{d+1}}q_{\ell-1}e^{i\phi'_{d+1}}a^{\ell+1} - e^{i\phi'_{d+1}}q_{\ell-1}a^{\ell+1} \\ &= q_{\ell-1}a^{\ell+1}(e^{-i(\phi'_{d+1}-\phi'_{d+1})} - e^{i\phi'_{d+1}}) \end{aligned}$$

For these terms to cancel out it needs to be $e^{-i(\phi'_{d+1}-\phi'_{d+1})} = 1 = e^{i\phi'_{d+1}}$, which is satisfied. The leading term of $e^{i\phi'_{d+1}}aQ(a)$ is $e^{i\phi'_{d+1}}a \cdot q_{\ell-1}a^{\ell-1} = e^{i\phi'_{d+1}}q_{\ell-1}a^\ell$. The leading term of $-e^{-i\phi'_{d+1}}P(a)$ is $-e^{-i\phi'_{d+1}}p_\ell a^\ell$. Combining these, the highest order terms in $Q'(a)$ are:

$$e^{i\phi'_{d+1}}q_{\ell-1}a^\ell - e^{-i\phi'_{d+1}}p_\ell a^\ell$$

Given $|p_\ell| = |q_{\ell-1}|$ and assuming $p_\ell = q_{\ell-1}e^{i\phi'_{d+1}}$, we have:

$$e^{i\phi'_{d+1}}q_{\ell-1}a^\ell - e^{-i\phi'_{d+1}}q_{\ell-1}e^{i\phi'_{d+1}}a^\ell = q_{\ell-1}a^\ell(e^{i\phi'_{d+1}} - 1)$$

For these terms to cancel out: $e^{i\phi'_{d+1}} = 1$, which is satisfied. Since the highest degree terms cancel out, the degrees of $P'(a)$ and $Q'(a)$ must be reduced by one:

$$\begin{aligned} \deg(P') &\leq \ell - 1 \leq d \\ \deg(Q') &\leq \ell - 2 \leq d - 1 \end{aligned}$$

Therefore, P' and Q' satisfy condition 1 of the theorem regarding d. The polynomial P

has parity $\ell \mod 2$. Since $\deg(P') = \ell - 1$, P' has parity $(\ell - 1) \mod 2$, which matches the parity condition for d. Similarly, Q has parity $(\ell - 1) \mod 2$. Since $\deg(Q') = \ell - 2$, Q' has parity $(\ell-2) \mod 2$, which matches the parity condition for $d-1$. Thus, condition 2 of the theorem is satisfied.
Note that $e^{-i\phi'_{d+1}Z}W^\dagger(a)$ is unitary as a product of unitary matrices according to Lemma 2.1. Therefore, Condition 3 is preserved due to the unitarity of the transformation, as established in Theorem 2.10. Applying the induction hypothesis, we get that the matrix equation equals

$$\begin{pmatrix} P'(a) & iQ'(a)\sqrt{1-a^2} \\ iQ'^*(a)\sqrt{1-a^2} & P'^*(a) \end{pmatrix} = e^{i\phi_0 Z} \left(\prod_{j=1}^{d+1} W(a) e^{i\phi_j Z} \right)$$

for some $\vec{\phi} \in \mathbb{R}^{d+1}$. Therefore, the signal sequence $\vec{\phi'} := (\phi_0, \phi_1, \phi_2, \ldots, \phi_d, \phi'_{d+1}) \in \mathbb{R}^{d+2}$ is valid for the polynomials. □

To summarize, the QSP-Theorem states that angles can be identified to realize complex polynomial transformations, which we can extract with projection $P(a) = \langle 0| U_{\vec{\phi}} |0\rangle$, where the maximal degree and parity of these transformations are directly governed by the number of these angles used. The approximation can be made arbitrarily close to the desired polynomial by increasing the number of unitary operations, subject to the limits of practical implementation and the effects of quantum noise. The phase factors can be calculated using the method described in Section 2.4.2.2. The Remez algorithm [Rem34] can be used for finding optimal polynomial approximations (see [Che66]).

2.4.2.1 Chebychev Polynomials and QSP

Chebychev polynomials are closely related to Quantum Signal Processing, especially as we will see in their connection to the powers W^d of the signal rotation operator (see Definition 2.35). Therefore, we introduce Chebychev polynomials following the formalism from [Tre19, Sau13].

Definition 2.38. *The **Chebyshev polynomial** of degree $n \in \mathbb{N}_0$, denoted by T_n, is defined for $\theta \in [0, \pi]$ and satisfies the equation:*

$$T_n(\cos\theta) = \cos(n\theta).$$

Remark 2.25. *Note that the cosine function $\cos(\theta)$ is continuous and monotonically decreasing on the interval $[0, \pi]$, with $\cos(0) = 1$ and $\cos(\pi) = -1$. By the intermediate value theorem, for each $x \in [-1, 1]$, there exists a unique $\theta \in [0, \pi]$ such that $x = \cos(\theta)$. Therefore, since $\theta \in [0, \pi]$ implies $-1 \leq \cos(\theta) \leq 1$, we can make the change of variables $x := \cos(\theta)$. Consequently, the Chebyshev polynomial $T_n(x)$ can be expressed for $x \in [-1, 1]$ equivalently as:*

$$T_n(x) = \cos\left(n \arccos(x)\right).$$

An easy way of imagining $T_n(x)$ as a function of x is to expand $\cos(n\theta)$ in powers of $\cos\theta$, and to write x in place of $\cos\theta$. Hence $T_n \in \mathbb{R}[X]$ of degree at most n with real coefficients. Then we get the following recursive formula:

Lemma 2.1. *For $x \in [-1, 1]$, the Chebyshev polynomials satisfy the recurrence relation:*

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)$$

where $T_0(x) = 1$ and $T_1(x) = x$.

Proof. Let $x \in [-1, 1]$ and $\theta \in [0, \pi]$. We identify $x := \cos(\theta)$. It is $T_0(x) = \cos(0) = 1$ and $T_1(x) = cos(\theta)$. For a given $n \geq 2$ consider the identity :

$$\cos((n+1)\theta) + \cos((n-1)\theta) = 2\cos\theta\cos(n\theta).$$

With the definition of Chebyshev polynomials, we have:

$$T_{n+1}(\cos\theta) + T_{n-1}(\cos\theta) = 2\cos\theta T_n(\cos\theta) \quad \Leftrightarrow \quad T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)$$

□

Chebyshev polynomials have many applications in approximation theory (see [Tre19]). They are particularly useful because the heights of the peaks of the function $T_n(x) = \cos(n\theta)$ for $x = \cos\theta$ are all equal to one.

Lemma 2.2. *For $x \in [-1, 1]$ and $\theta \in [0, \pi]$ with $x := \cos(\theta)$, the Chebyshev polynomials are bounded $|T_n(x)| \leq 1$.*

Proof. This follows by definition of $T_n(x)$, since $\cos(y)$ is bounded for every $y \in \mathbb{R}$, therefore $|T_n(x)| = |\cos(n\theta)| \leq 1$. □

Lemma 2.3. *For Chebyshev polynomials of the first kind, $T_j(-x) = (-1)^j T_j(x)$.*

Proof. We prove the statement by induction.
Base Case: For $j = 0$ and $j = 1$ we have:

$$T_0(-x) = 1 = (-1)^0 T_0(x)$$
$$T_1(-x) = -x = (-1)^1 x = (-1)^1 T_1(x)$$

Induction hypothesis: Assume that for some $j \geq 1$, the following holds for j and $j-1$:

$$T_j(-x) = (-1)^j T_j(x).$$

Induction step: We need to show that $T_{j+1}(-x) = (-1)^{j+1} T_{j+1}(x)$. We have:

$$\begin{aligned} T_{j+1}(-x) &= 2(-x)T_j(-x) - T_{j-1}(-x) \\ &= 2(-x)(-1)^j T_j(x) - (-1)^{j-1} T_{j-1}(x) \\ &= (-1)^{j+1} 2x T_j(x) - (-1)^{j-1} T_{j-1}(x) \\ &= (-1)^{j+1}(2x T_j(x) - T_{j-1}(x)) \\ &= (-1)^{j+1} T_{j+1}(x). \end{aligned}$$

□

Definition 2.39. *The **Chebyshev polynomial of the second kind** of degree $n \in \mathbb{N}_0$, denoted by U_n, is defined for $\theta \in [0, \pi]$ and satisfies the equation:*

$$U_n(\cos\theta) = \frac{\sin((n+1)\theta)}{\sin\theta}.$$

Remark 2.26. *Again we can make the change of variables $x := \cos(\theta)$, then the Chebyshev polynomial $U_n(x)$ can be expressed for $x \in [-1, 1]$ as:*

$$U_n(x) = \frac{\sin\left((n+1)\arccos(x)\right)}{\sqrt{1-x^2}}.$$

We can imagine $U_n(x)$ as a function of x by expanding it in powers of $\cos\theta$, and to write x in place of $\cos\theta$. Hence $U_n \in \mathbb{R}[X]$ of degree n with real coefficients. Then we get the following recursive formula:

Lemma 2.4. *For $x \in [-1, 1]$, the Chebyshev polynomials of the second kind satisfy the recurrence relation:*

$$U_{n+1}(x) = 2x U_n(x) - U_{n-1}(x)$$

where $U_0(x) = 1$ and $U_1(x) = 2x$.

Proof. Let $x \in [-1, 1]$ and $\theta \in [0, \pi]$. We identify $x := \cos(\theta)$. It is:

$$U_0(x) = \frac{\sin(1 \cdot \theta)}{\sin(\theta)} = 1 \quad \text{and} \quad U_1(x) = \frac{\sin(2\theta)}{\sin\theta} = 2\cos(\theta) = 2x$$

For a given $n \geq 2$ consider the identity:

$$\sin((n+2)\theta) = 2\cos\theta \sin((n+1)\theta) - \sin(n\theta).$$

With the definition of Chebyshev polynomials of the second kind, we have:

$$\begin{aligned} U_{n+1}(\cos\theta) &= \frac{\sin((n+2)\theta)}{\sin\theta} = \frac{2\cos\theta\sin((n+1)\theta) - \sin(n\theta)}{\sin\theta} \\ &= 2\cos\theta\frac{\sin((n+1)\theta)}{\sin\theta} - \frac{\sin(n\theta)}{\sin\theta} \\ &= 2xU_n(x) - U_{n-1}(x) \end{aligned}$$

□

We can demonstrate, as follows, that the Chebyshev polynomials are inherently constructed by the QSP sequence, as established in Theorem 2.12.

Lemma 2.5. *For any integer $d \geq 1$, the d-th power of the signal rotation operator $W(a)$ from Def. 2.35 is given by*

$$W^d(a) = \begin{pmatrix} T_d(a) & iU_{d-1}(a)\sqrt{1-a^2} \\ iU_{d-1}(a)\sqrt{1-a^2} & T_d(a) \end{pmatrix},$$

where $T_d(a)$ is the Chebyshev polynomial of the first kind with degree d and $U_{d-1}(a)$ is the Chebyshev polynomial of the second kind with degree $d-1$.

Proof. Let $a \in [-1,1]$. Recall that the signal rotation operator according by Definition 2.35, is then given as:

$$W(a) = \begin{pmatrix} a & i\sqrt{1-a^2} \\ i\sqrt{1-a^2} & a \end{pmatrix}, \tag{2.21}$$

We prove the statement by Induction over $d \in \mathbb{N}_0$.

Base Case: For $d = 1$, we have:

$$W^1(a) = W(a) = \begin{pmatrix} a & i\sqrt{1-a^2} \\ i\sqrt{1-a^2} & a \end{pmatrix},$$

which matches with the matrix since $T_1(a) = a$ and $U_0(a) = 1$ according to definition.

Induction Hypothesis: Assume that for some $din\mathbb{N}$ folowing holds,

$$W^d(a) = \begin{pmatrix} T_d(a) & iU_{d-1}(a)\sqrt{1-a^2} \\ iU_{d-1}(a)\sqrt{1-a^2} & T_d(a) \end{pmatrix}.$$

Induction Step: $d \mapsto d+1$ We calculate $W^{d+1}(a) = W^d(a)W(a)$ explicitly using the induction hypothesis.

$$W^{d+1}(a) =^{(IH)} \begin{pmatrix} T_d(a) & iU_{d-1}(a)\sqrt{1-a^2} \\ iU_{d-1}(a)\sqrt{1-a^2} & T_d(a) \end{pmatrix} \begin{pmatrix} a & i\sqrt{1-a^2} \\ i\sqrt{1-a^2} & a \end{pmatrix}$$

Calculating each matrix element one its own, we get the following terms:

$$\begin{aligned}
\left(W^{d+1}(a)\right)_{11} &= T_d(a)\cdot a + iU_{d-1}(a)\sqrt{1-a^2}\cdot i\sqrt{1-a^2} = aT_d(a) - (1-a^2)U_{d-1}(a),\\
\left(W^{d+1}(a)\right)_{12} &= T_d(a)\cdot i\sqrt{1-a^2} + iU_{d-1}(a)\sqrt{1-a^2}\cdot a = i\left(T_d(a) + aU_{d-1}(a)\right)\sqrt{1-a^2},\\
\left(W^{d+1}(a)\right)_{21} &= iU_{d-1}(a)\sqrt{1-a^2}\cdot a + T_d(a)\cdot i\sqrt{1-a^2} = i\left(aU_{d-1}(a) + T_d(a)\right)\sqrt{1-a^2},\\
\left(W^{d+1}(a)\right)_{22} &= iU_{d-1}(a)\sqrt{1-a^2}\cdot i\sqrt{1-a^2} + T_d(a)\cdot a = aT_d(a) - (1-a^2)U_{d-1}(a).
\end{aligned}$$

Note that $(W^{d+1}(a))_{11} = (W^{d+1}(a))_{22}$ and $(W^{d+1}(a))_{12} = (W^{d+1}(a))_{21}$. We use the change of variables where $a := cos(\theta)$ for some $\theta \in [0,\pi]$, therefore the polynomials can be expressed according to Remark 2.25 and Remark 2.26 as:

$$T_d(a) = \cos\left(d\arccos(a)\right), \quad U_{d-1}(a) = \frac{\sin\left(d\arccos(a)\right)}{\sqrt{1-a^2}}.$$

With this identification we can further simplify and get:

$$\begin{aligned}
\left(W^{d+1}(a)\right)_{11} &= a\cos\left(d\arccos(a)\right) - (1-a^2)\sin\left(d\arccos(a)\right)\left(\sqrt{1-a^2}\right)^{-1}\\
&= a\cos\left(d\arccos(a)\right) - \sqrt{1-a^2}\sin\left(d\arccos(a)\right)\\
&= \cos\left(\arccos(a)\right)\cos\left(d\arccos(a)\right) - \sin\left(\arccos(a)\right)\sin\left(d\arccos(a)\right)\\
&= \cos\left(\arccos(a) + d\arccos(a)\right)\\
&= \cos\left((d+1)\arccos(a)\right)\\
&= T_{d+1}(a)
\end{aligned}$$

Where we used that $a = \cos(\arccos(a))$ thus $\sqrt{1-a^2} = \sin(\arccos(a))$ and then applied the cosine addition formula.

$$\begin{aligned}
\left(W^{d+1}(a)\right)_{12} &= i\left(\cos\left(d\arccos(a)\right) + a\sin\left(d\arccos(a)\right)\left(\sqrt{1-a^2}\right)^{-1}\right)\sqrt{1-a^2}\\
&= i\left(\cos\left(d\arccos(a)\right)\sqrt{1-a^2} + a\sin\left(d\arccos(a)\right)\right)\\
&= i\left(\cos\left(d\arccos(a)\right)\sqrt{1-a^2} + \sin\left(d\arccos(a)\right)\left(a(\sqrt{1-a^2})^{-1}\sqrt{1-a^2}\right)\right)\\
&= i\left(\sin\left((d+1)\arccos(a)\right)\left(\sqrt{1-a^2}\right)^{-1}\right)\sqrt{1-a^2}\\
&= i\sin\left((d+1)\arccos(a)\right)\\
&= iU_d(a)\sqrt{1-a^2}
\end{aligned}$$

Where we used the same identification $a = \cos(\arccos(a))$ thus $\sqrt{1-a^2} = \sin(\arccos(a))$ and then applied the sine addition formula. Therefore we are left with:

$$W^{d+1}(a) = \begin{pmatrix} T_{d+1}(a) & iU_d(a)\sqrt{1-a^2} \\ iU_d(a)\sqrt{1-a^2} & T_{d+1}(a) \end{pmatrix},$$

□

Theorem 2.13. *The signal sequence* $\vec{\phi} = (0, 0, ..., 0) \in \mathbb{R}^{d+1}$ *creates* $P(a) := T_d(a)$ *the Chebyshev polynomial of the first kind and degree* d.

Proof. We prove the statement by Induction over $d \in \mathbb{N}$. Let the signals $U_{\vec{\phi}}$ for $\vec{\phi} = (0, 0, ..., 0) \in \mathbb{R}^{d+1}$ be given. Since for these signals $e^{i\phi_k Z} = e^{0 \cdot iZ} = 1$ (see Lemma 2.1) for every $0 \leq k \leq d$

$$U_{\vec{\phi}} = e^{i\phi_0 Z} \prod_{k=1}^{d} W(a) e^{i\phi_k Z} = \prod_{k=1}^{d} W(a) = W^d(a)$$

Base Case: For $d = 0$ and the signal $\vec{\phi} = (0)$ it is $W^0(a) = I_2$ the identity matrix, thus $P \equiv 1$, which matches $T_0(a) = 1$.

Induction Hypothesis: Assume the proposition holds for some $d \in \mathbb{N}$, i.e., the signals $\vec{\phi} = (0, 0, ..., 0) \in \mathbb{R}^{d+1}$ generate with the QSP Operation sequence the polynomial $P(a) = T_d(a)$ the Chebyshev polynomial of first kind and degree d.

Induction Step: $d \mapsto d + 1$ We consider the signal $\vec{\phi} = (0, 0, ..., 0) \in \mathbb{R}^{(d+1)+1}$. We apply Lemma 2.5, then the QSP operation sequence is given by:

$$W^{d+1}(a) = \begin{pmatrix} T_{d+1}(a) & iU_d(a)\sqrt{1-a^2} \\ iU_d(a)\sqrt{1-a^2} & T_{d+1}(a) \end{pmatrix}$$

Thus with the QSP Operation sequence the signals $\vec{\phi}$ generate the polynomial $P(a) = T_{d+1}(a)$ the Chebyshev polynomial of first kind and degree $d + 1$. □

Remark 2.27. *The use of Chebyshev polynomials in Quantum Signal Processing (see Theorem 2.12) offers significant benefits due to their special characteristics. When all phase signals in the QSP sequence are set to zero, the resulting sequence naturally forms Chebyshev polynomials (see Theorem 2.13), making them a fundamental part of the QSP method. Additionally, Chebyshev polynomials are known for providing the best uniform approximation to a continuous function according to the minimax criterion (see [Tre19, Sau13]), meaning they minimize the maximum possible error in approximation. Moreover, Chebyshev polynomials are highly numerically stable (see [Sau13]), which ensures that their use in polynomial approximations does not introduce large errors during computation.*

2.4.2.2 Determining the QSP Phase Factors

In this section, we detail the procedure to obtain the phase factors necessary for Quantum Signal Processing (QSP) using the procedure described in [DMWL21], given a target polynomial $f(x)$.

The notation $U_{\vec{\phi}}(x)$ in the QSP Theorem 2.12 refers to the QSP unitary matrix parameterized by the phase vector $\vec{\phi} = (\phi_0, \phi_1, \ldots, \phi_d) \in \mathbb{R}^{d+1}$, where $d \in \mathbb{N}$ is the degree of the target polynomial $f(x)$ to be approximated. This matrix is explicitly given by:

$$U_{\vec{\phi}}(x) = \begin{pmatrix} P(x) & iQ(x)\sqrt{1-x^2} \\ iQ^*(x)\sqrt{1-x^2} & P^*(x) \end{pmatrix},$$

where $P, Q \in \mathbb{C}[X]$ are polynomials in $x \in [-1, 1]$ that satisfy the conditions specified in the theorem. The vector $\vec{\phi}$ corresponds to a sequence of phases used to construct this matrix via a series of quantum operations, as described in the definition. The phase factors $\vec{\phi}$ from Theorem 2.12 are to be determined such that the polynomial encoded in the upper-left element of $U_{\vec{\phi}}$ approximates the target polynomial $f(x)$. Note that this upper-left element can be extracted using projectors:

$$\langle 0|U_{\vec{\phi}}(x)|0\rangle = P(x).$$

Remark 2.28. *In the context of Quantum Signal Processing, the target polynomial $f(x)$ is the polynomial that we aim to approximate using the QSP Theorem 2.12. The degree $d \in \mathbb{N}$ mentioned throughout the text refers to the degree of this target polynomial $f(x)$. The polynomials $P(x)$ and $Q(x)$, which appear in the QSP unitary matrix $U_{\vec{\phi}}(x)$, are constructed such that the matrix $U_{\vec{\phi}}(x)$ approximates $f(x)$ through its upper-left element.*

- *$P(x)$ is a polynomial of at most degree d.*
- *$Q(x)$ is a polynomial typically of at most degree $d-1$.*

These polynomials are intrinsically related to the phase factors $\vec{\phi}$ and are designed to ensure that the desired approximation of $f(x)$ is achieved when applying the QSP procedure. Thus, $f(x)$, $P(x)$, and $Q(x)$ are interconnected, with $P(x)$ and $Q(x)$ being the components of the QSP unitary matrix used to approximate the target polynomial $f(x)$.

Lemma 2.1 (Representation of Polynomials)**.** *Given a real polynomial $f \in \mathbb{R}[X]$ of degree $d \in \mathbb{N}$, there exist unique phase factors $\vec{\phi} = (\phi_0, \ldots, \phi_d) \in \mathbb{R}^{d+1}$ such that:*

$$Re\left[\langle 0|U_{\vec{\phi}}(x)|0\rangle\right] = f(x),$$

Proof. For a proof see [DMWL21]. □

Definition 2.40 (Loss Function)**.** *The discrepancy between the desired polynomial $f(x)$ from Remark 2.28 and the real part of the QSP polynomial $P(x)$ is quantified by a loss function $L : \mathbb{R}^{d+1} \to \mathbb{R}$, defined as:*

$$L(\vec{\phi}) = \frac{1}{N}\sum_{j=1}^{N}\left|Re\langle 0|U_{\vec{\phi}}(x_j)|0\rangle - f(x_j)\right|^2,$$

where $\{x_j\}_{j=1}^N \subseteq [-1,1]$ *are sample points.*

Theorem 2.14 (Optimization Problem)**.** *The optimal phase factors* $\vec{\phi}^* \in \mathbb{R}^{d+1}$ *are obtained by minimizing the loss function:*

$$\vec{\phi}^* = \arg\min_{\vec{\phi}} L(\vec{\phi}).$$

Remark 2.29. *This is a non-linear optimization problem where* $\vec{\phi}$ *is iteratively updated to reduce* $L(\vec{\phi})$ *(see Algorithm 2.1).*

Lemma 2.1 (Inversion Symmetry)**.** *If the polynomial* $f(x)$ *from Remark 2.28 has definite parity, the optimal phase factors* $\vec{\phi}^*$ *exhibit inversion symmetry:*

$$\vec{\phi}^* = (\phi_0, \phi_1, \ldots, \phi_{d-1}, \phi_d) \quad \textit{with} \quad \phi_i = \phi_{d-i}, \quad \forall i.$$

Proof. For a proof see [DMWL21]. □

Note that this symmetry from the above Lemma 2.1 reduces the number of independent parameters, enhancing the numerical stability of the optimization.

Theorem 2.15 (Complexity)**.** *The computational complexity of the optimization process is* $O(Nd^3)$*, where* d *is the degree of the polynomial* $f(x)$ *from Remark 2.28 and* N *is the number of sample points.*

Proof. For a proof see [DMWL21]. □

Remark 2.30. *It is important to note that the computational complexity of determining the phase factors using Algorithm 2.1 does not affect the overall complexity of the quantum algorithm proposed for solving the UCP. This is because the phase factors are precomputed and remain constant throughout the algorithm's execution. As a result, the complexity of the quantum algorithm is governed entirely by its operational structure and not by the preliminary phase factor computation.*

Algorithm 2.1: Optimization-Based Method for Finding QSP Phase Factors

Input : Polynomial $f \in \mathbb{R}[X]$ of degree $d \in \mathbb{N}$ from Remark 2.28, initial phase vector $\vec{\phi}^{(0)} = (\frac{\pi}{4}, 0, \ldots, 0, \frac{\pi}{4}) \in \mathbb{R}^{d+1}$, tolerance $\epsilon > 0$

Output : Optimized phase factors $\vec{\phi} = (\phi_0, \phi_1, \ldots, \phi_d) \in \mathbb{R}^{d+1}$

Complexity: The computational complexity of this procedure is $O(Nd^3)$, where d is the degree of the polynomial and N is the number of sample points.

Procedure :

1 Initialization: Choose an initial guess for the phase factors $\vec{\phi}^{(0)}$ to ensure symmetry, such as $\vec{\phi}^{(0)} = (\frac{\pi}{4}, 0, \ldots, 0, \frac{\pi}{4})$.

2 Define the loss function:

$$L(\vec{\phi}) = \frac{1}{N} \sum_{j=1}^{N} \left| \mathrm{Re}\langle 0 | U_{\vec{\phi}}(x_j) | 0 \rangle - f(x_j) \right|^2$$

where $\{x_j\}_{j=1}^{N}$ are sample points on the interval $[-1, 1]$.

3 Compute the gradient $\nabla_{\vec{\phi}} L(\vec{\phi})$ of the loss function with respect to the phase factors $\vec{\phi}$.

4 Update the phase factors iteratively using a quasi-Newton method such as L-BFGS ([LN89],[DMWL21]). $\vec{\phi}^{(k)}$ denotes the phase vector at the k-th iteration of the algorithm:

$$\vec{\phi}^{(k+1)} = \vec{\phi}^{(k)} - \alpha_k \nabla_{\vec{\phi}} L(\vec{\phi}^{(k)}),$$

where α_k is the step size determined by the optimization algorithm.

5 Repeat the gradient calculation and phase factor update until convergence:

$$L(\vec{\phi}^{(k)}) < \epsilon.$$

2.4.3 Quantum Singular Value Transformation

The Quantum Singular Value Transformation (QSVT) is a central component of our proposed algorithm, offering a crucial mechanism for efficient matrix inversion and polynomial approximations. QSVT builds on the foundational concepts of Quantum Signal Processing (QSP), extending its framework to enable more advanced transformations. In this section, we use the notation and formalism from [MRTC21] and [GSLW18]. We begin by establishing the fundamental concepts necessary for introducing QSVT.

Definition 2.41. *Let $A \in \mathbb{C}^{m\times n}$. The **singular values** σ_i of A, for all $1 \leq i \leq \min(m,n)$, are defined as the square roots of the eigenvalues of the matrix A^*A, where A^* denotes the conjugate transpose of A.*

Note that for a given matrix $A \in \mathbb{C}^{m\times n}$, the number of non-zero eigenvalues of A^*A is equal to $\text{rank}(A)$. Consequently, the number of non-zero singular values is also equal to $\text{rank}(A)$. These non-zero singular values, denoted as $\sigma_i > 0$ for $i \in \{1, 2, \ldots, \text{rank}(A)\}$, can be ordered such that $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{\text{rank}(A)} > 0$. Note that the above definition includes $\min(m,n)$ singular values in total, with any remaining singular values being zero if $\text{rank}(A) < \min(m,n)$.

Theorem 2.16 (Singular Value Decomposition)**.** *Let $A \in \mathbb{C}^{m\times n}$ and $r := \text{rank}(A)$. Let $\sigma_1, \sigma_2, \ldots, \sigma_r > 0$ be the non-zero singular values of A. Then there exist unitary matrices $W \in \mathbb{C}^{m\times m}$ and $V \in \mathbb{C}^{n\times n}$, and a diagonal matrix $\Sigma \in \mathbb{R}^{m\times n}$ with:*

$$\Sigma := \begin{pmatrix} \sigma_1 & 0 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & \sigma_2 & 0 & \cdots & 0 & \cdots & 0 \\ 0 & 0 & \sigma_3 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & \sigma_r & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 & \cdots & 0 \end{pmatrix}$$

such that $A = W\Sigma V^\dagger$.

Proof. For a complete proof, see Theorem 5.2 and Lemma 5.13 from [Bis21]. □

2.4.3.1 Block Encoding and Linear Combination of Unitaries

A matrix $A \in \mathbb{C}^{n\times m}$ can be block encoded into a unitary matrix $U_A \in \mathbb{C}^{N\times N}$ with $N \geq \max(n,m)$ and respect to a normalization factor $\lambda > 0$:

$$U_A = \begin{pmatrix} A/\lambda & \cdot \\ \cdot & \cdot \end{pmatrix}$$

where A/λ is the upper-left block of the matrix U_A. We use Definition 43 from [GSLW18] to define block encoding:

Definition 2.42 (Block Encoding)**.** *Let $A \in \mathbb{C}^{n\times m}$. A unitary matrix $U_A \in \mathbb{C}^{N\times N}$ with $N \geq \max(n,m)$ and $\lambda > 0$ is called a $(\lambda, a, \varepsilon)$-**Block Encoding** for A if for $\varepsilon > 0$ the*

following holds:

$$\left\| A - \lambda \left(\langle 0|^{\otimes a} \otimes I_n \right) U_A \left(|0\rangle^{\otimes a} \otimes I_m \right) \right\| \leq \varepsilon,$$

where $I_n \in \mathbb{R}^{n \times n}$ *and* $I_m \in \mathbb{R}^{m \times m}$ *are identity matrices.*

To see a detailed explanation of how the extraction of matrix A works, refer to the proof of the QSVT Theorem 2.17, where this process is explicitly demonstrated. A common approach to achieve a block encoding is to express an arbitrary matrix $A \in \mathbb{C}^{n \times n}$ as a combination of unitary matrices. This process, called Linear Combination of Unitaries (LCU), is described by [Boy23, CW12a].

Corollary 2.9 (Linear Combination of Unitaries)**.** *Let* $\mathcal{H}$ *be an* n*-dimensional Hilbert space. Any operator* $A \in \mathcal{L}(\mathcal{H})$ *can be expressed as a linear combination of* n^2 *unitary operators. Specifically, there exist complex coefficients* $\lambda_i \in \mathbb{C}$ *and unitary operators* $U_i \in \mathcal{L}(\mathcal{H})$ *for* $i \in \{1, 2, \ldots, n^2\}$ *such that:*

$$A = \sum_{i=1}^{n^2} \lambda_i U_i.$$

Proof. The corollary is implied by Theorem 2.11. The Pauli basis for the n-fold tensor product space $\mathcal{H}$ consists of all possible tensor products of the single-qubit Pauli operators $\{I, X, Y, Z\}$. The set $\{I, X, Y, Z\}^{\otimes n}$ forms a basis for $\mathcal{L}(\mathcal{H})$. Any operator $A \in \mathcal{L}(\mathcal{H})$ can be expressed as a linear combination of these basis elements. Each Pauli matrix is unitary, and the tensor product of unitary operators is also unitary. □

2.4.3.2 The QSVT Theorem

Remark 2.31. *Let* $A \in \mathcal{L}(\mathcal{H}_1, \mathcal{H}_2)$ *be a bounded linear operator between two Hilbert spaces* $\mathcal{H}_1$ *and* $\mathcal{H}_2$*, with* $r := \operatorname{rank}(A)$*. The SVD of* A *(see Theorem 2.16) provides a factorization* $A = U\Sigma V^\dagger$*, where* U *and* V *are mappings from finite-dimensional subspaces of* $\mathcal{H}_2$ *and* $\mathcal{H}_1$*, respectively, onto* $\mathbb{C}^r$*. The columns of* U *and* V*, denoted by* $\{|u_k\rangle\}$ *and* $\{|v_k\rangle\}$*, are orthonormal sets in their respective Hilbert spaces. According to the unitarity of* U *and* V*, the vectors* $\{|u_k\rangle\}_{k=1}^r \subset \mathcal{H}_2$ *and* $\{|v_k\rangle\}_{k=1}^r \subset \mathcal{H}_1$ *form orthonormal bases for the image and preimage subspaces corresponding to the non-zero singular values of* A*. Therefore, we can express the operator* A *in terms of its singular value decomposition as:*

$$A = \sum_{k=1}^{r} \sigma_k |u_k\rangle\langle v_k|,$$

where $\sigma_k > 0$ *are the singular values of* A*,* $|u_k\rangle \in \mathcal{H}_2$ *are the left singular vectors, and* $|v_k\rangle \in \mathcal{H}_1$ *are the right singular vectors. This decomposition is analogous to the spectral decomposition of a normal operator, with the singular values playing the role of eigenvalues.*

We now can formulate the process of Quantum Singular Value Transformation (QSVT), according to Theorem 17 from [GSLW18] and Theorem 4 from [MRTC21] as the following:

Theorem 2.17 (Quantum Singular Value Transformation (QSVT))**.** *Let $A \in \mathbb{R}^{n\times n}$. Given a $(\lambda, a, \varepsilon)$-block encoding $U_A \in \mathbb{C}^{N\times N}$ according to Definition 2.42 with $N \geq n$ of the matrix A and the projector*

$$\Pi = |0\rangle^{\otimes a} \langle 0|^{\otimes a} \otimes I_n$$

locating A inside U_A, then for odd d, we have for the signals $\vec{\phi} = (\phi_0, \phi_1, \ldots, \phi_d) \in \mathbb{R}^{d+1}$ (see Theorem 2.12):

$$U_{\vec{\phi}} = \Pi_{\phi_1} U_A \left[\prod_{k=1}^{(d-1)/2} \Pi_{\phi_{2k}} U_A^\dagger \Pi_{\phi_{2k+1}} U_A\right] = \begin{pmatrix} \mathrm{Poly}^{(\mathrm{SV})}(A) & \cdot \\ \cdot & \cdot \end{pmatrix},$$

where $\Pi_{\phi_k} := e^{i2\phi_k \Pi}$ are projector-controlled phase shift operators and $\mathrm{Poly}^{(\mathrm{SV})}(A)$ is defined for an odd polynomial as

$$\mathrm{Poly}^{(\mathrm{SV})}(A) := \sum_k \mathrm{Poly}(\sigma_k) |w_k\rangle \langle v_k|, \tag{2.22}$$

which applies a polynomial transform to the singular values of $A = \sum_k \sigma_k |v_k\rangle \langle w_k|$. The polynomial is of degree at most d and satisfies the conditions of P from Theorem 2.12. Similarly, for d even, we have for the signals $\vec{\phi} = (\phi_0, \phi_1, \ldots, \phi_d) \in \mathbb{R}^{d+1}$ (see Theorem 2.12):

$$U_{\vec{\phi}} = \left[\prod_{k=1}^{d/2} \Pi_{\phi_{2k-1}} U_A^\dagger \Pi_{\phi_{2k}} U_A\right] = \begin{pmatrix} \mathrm{Poly}^{(\mathrm{SV})}(A) & \cdot \\ \cdot & \cdot \end{pmatrix},$$

where $\mathrm{Poly}^{(\mathrm{SV})}(A)$ is defined for an even polynomial as

$$\mathrm{Poly}^{(\mathrm{SV})}(A) := \sum_k \mathrm{Poly}(\sigma_k) |v_k\rangle \langle v_k|, \tag{2.23}$$

which is also a polynomial transform of the singular values of A, but with the input and output spaces both being the right singular vector space, spanned by $\{|v_k\rangle\}$ (see Remark 2.31). Analogously, the polynomial is of degree at most d and satisfies the conditions of P from Theorem 2.12.

Proof. Given a matrix $A \in \mathbb{R}^{n\times n}$ with $r := \mathrm{rank}(A)$ and its SVD (see Theorem 2.16):

$$A = V \Sigma W^\dagger,$$

where $V \in \mathbb{R}^{n\times n}$ is a unitary matrix containing the right singular vectors, $W \in \mathbb{R}^{n\times n}$ is a unitary matrix containing the left singular vectors, and $\Sigma \in \mathbb{R}^{n\times n}$ is a diagonal matrix with

$$\Sigma = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_r),$$

the singular values $\sigma_k > 0$ on the diagonal for $k \in \{1, \ldots, r\}$. We can identify $A \in \mathcal{L}(\mathcal{H})$ (see Corollary 2.8) as a bounded linear operator between two Hilbert spaces $\mathcal{H} :=$

$\mathcal{L}(\mathbb{R}^n)$ (see Remark 2.31). According to the unitarity of V and W, the column vectors $\{|v_k\rangle\}_{k=1}^r \subset \mathcal{H}$ and $\{|w_k\rangle\}_{k=1}^r \subset \mathcal{H}$ form orthonormal bases for the image and preimage subspaces corresponding to the non-zero singular values of A. Therefore, we can express the operator A in terms of its singular value decomposition as:

$$A = V\Sigma W^\dagger = \sum_{k=1}^{r} \sigma_k |v_k\rangle\langle w_k|,$$

where $\sigma_k > 0$ are the singular values of A, $|v_k\rangle \in \mathcal{H}$ are the left singular vectors, and $|w_k\rangle \in \mathcal{H}$ are the right singular vectors. Therefore, V and W can then be expressed according to Remark 2.31:

$$V = \begin{pmatrix} | & | & & | \\ |v_1\rangle & |v_2\rangle & \dots & |v_r\rangle \\ | & | & & | \end{pmatrix}, \quad W = \begin{pmatrix} | & | & & | \\ |w_1\rangle & |w_2\rangle & \dots & |w_r\rangle \\ | & | & & | \end{pmatrix}$$

where each $|v_k\rangle$ and $|w_k\rangle$ is a column vector as described above.

Let a $(\lambda, a, \varepsilon)$-Block Encoding $U_A \in \mathbb{C}^{N\times N}$ with $N \geq n$ and $\lambda > 0$ be given for A according to Definition 2.42. Therefore, for $\varepsilon > 0$, the following holds:

$$\left\| A - \lambda \left(\langle 0|^{\otimes a} \otimes I_n \right) U_A \left(|0\rangle^{\otimes a} \otimes I_m \right) \right\| \leq \varepsilon,$$

where $I_n \in \mathbb{R}^{n\times n}$ and $I_m \in \mathbb{R}^{m\times m}$ are identity matrices.
The projector $\Pi \in \mathbb{R}^{N\times N}$ is defined according to the theorem as:

$$\Pi = |0\rangle^{\otimes a} \langle 0|^{\otimes a} \otimes I_n$$

where $I_n \in \mathbb{R}^{n\times n}$ is the identity matrix. We can calculate the tensor product and get the explicit matrix representation, where $N = 2^a \cdot n$:

$$\Pi = |0\rangle^{\otimes a} \langle 0|^{\otimes a} \otimes I_n = \underbrace{\begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}}_{2^a \times 2^a} \otimes \underbrace{\begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}}_{n\times n} = \underbrace{\begin{pmatrix} I_n & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}}_{N\times N}$$

Given the unitary matrix U_A, which is a block encoding of a matrix A, it can be expressed as:

$$U_A = \begin{pmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{pmatrix},$$

where $U_{11} \in \mathbb{C}^{n\times n}$ corresponds to the block encoding of A. The submatrices $U_{12} \in \mathbb{C}^{n\times(N-n)}$, $U_{21} \in \mathbb{C}^{(N-n)\times n}$, and $U_{22} \in \mathbb{C}^{(N-n)\times(N-n)}$ are the other blocks.
Applying the projector Π isolates the top-left block of U_A as follows:

$$\Pi U_A \Pi = \underbrace{\begin{pmatrix} I_n & 0 \\ 0 & 0 \end{pmatrix}}_{\Pi \in \mathbb{R}^{N\times N}} \cdot \underbrace{\begin{pmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{pmatrix}}_{U_A \in \mathbb{C}^{N\times N}} \cdot \underbrace{\begin{pmatrix} I_n & 0 \\ 0 & 0 \end{pmatrix}}_{\Pi \in \mathbb{R}^{N\times N}} = U_{11} = \frac{1}{\lambda} A.$$

With that in mind, let $\phi_k \in \mathbb{R}$ for all $k \in \{0, ..., d\}$ as defined in the theorem, we can compute the matrix representation of the projector phase shift:

$$\Pi_{\phi_k} := e^{i2\phi_k \Pi}.$$

The exponential of a matrix for Π_{ϕ_k} is given by:

$$\Pi_{\phi_k} = e^{i2\phi_k \Pi} = \sum_{t=0}^{\infty} \frac{(i2\phi_k \Pi)^t}{t!}.$$

We calculate the first few terms explicitly and follow the same reasoning as in Lemma 2.1. For $t = 0$ we have $\frac{(i2\phi_k \Pi)^0}{0!} = I$. For $t = 1$ we have $\frac{(i2\phi_k \Pi)^1}{1!} = i2\phi_k \Pi$. For $t = 2$ we have:

$$\frac{(i2\phi_k \Pi)^2}{2!} = \frac{(i2\phi_k)^2 \Pi^2}{2!} = \frac{(i2\phi_k)^2 \Pi}{2!} = \frac{(i2\phi_k)^2}{2!} \Pi.$$

Since $\Pi^2 = \Pi$, by the definition of a projector (see Definition 2.26), each higher power of Π is just equal to Π, meaning:

$$\frac{(i2\phi_k \Pi)^t}{t!} = \frac{(i2\phi_k)^t}{t!} \Pi \quad \text{for all } t \geq 1.$$

Excluding the identity matrix I, which comes from the $t = 0$ term, we can factor the series and relate it to the Taylor series for the exponential function:

$$\Pi_{\phi_k} = I + \sum_{t=1}^{\infty} \frac{(i2\phi_k)^t}{t!} \Pi = I + \left(\sum_{t=1}^{\infty} \frac{(i2\phi_k)^t}{t!} \right) \Pi = I + \left(e^{i2\phi_k} - 1 \right) \Pi.$$

We know the representation of Π thus, we obtain:

$$\Pi_{\phi_k} = e^{i2\phi_k \Pi} = I + (e^{i2\phi_k} - 1) \begin{pmatrix} I_n & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix}.$$

This expression can be explicitly written as a block matrix:

$$\Pi_{\phi_k} = \begin{pmatrix} e^{i2\phi_k} I_n & 0 & \cdots & 0 \\ 0 & I_n & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & I_n \end{pmatrix} \in \mathbb{R}^{N \times N}.$$

The top-left block is $e^{i2\phi_k} I_n$, applying the phase shift $e^{i2\phi_k}$ to the identity matrix. This matrix form reflects the phase shift of the projector in combination with the phase shift applied to the corresponding subspace given by the block encoding. We first consider the case where $d \in \mathbb{N}$ is even. Then the QSVT Sequence according to the theorem is given by:

$$U_{\vec{\phi}} = \left[\prod_{k=1}^{d/2} \Pi_{\phi_{2k-1}} U_A^\dagger \Pi_{\phi_{2k}} U_A \right]$$

The expression $\Pi_{\phi_{2k}} U_A^\dagger \Pi_{\phi_{2k+1}} U_A$ of the QSVT Sequence can be explicitly calculated as follows. We consider the k-th term in the product for even d:

$$\Pi_{\phi_{2k-1}} U_A^\dagger \Pi_{\phi_{2k}} U_A = \left(I + (e^{i2\phi_{2k-1}} - 1)\Pi\right) U_A^\dagger \left(I + (e^{i2\phi_{2k}} - 1)\Pi\right) U_A.$$

Expand this expression:

$$\begin{aligned}\Pi_{\phi_{2k-1}} U_A^\dagger \Pi_{\phi_{2k}} U_A &= \left(I + (e^{i2\phi_{2k-1}} - 1)\Pi\right) U_A^\dagger \left(I + (e^{i2\phi_{2k}} - 1)\Pi\right) U_A \\ &= U_A^\dagger U_A + (e^{i2\phi_{2k-1}} - 1)\Pi U_A^\dagger U_A + (e^{i2\phi_{2k}} - 1) U_A^\dagger \Pi U_A \\ &+ (e^{i2\phi_{2k-1}} - 1)(e^{i2\phi_{2k}} - 1)\Pi U_A^\dagger \Pi U_A.\end{aligned}$$

We already calculated U_A and Π in terms of their matrix representations:

$$\Pi U_A = \begin{pmatrix} U_{11} & 0 \\ 0 & 0 \end{pmatrix}, \quad U_A^\dagger \Pi = \begin{pmatrix} U_{11}^\dagger & 0 \\ 0 & 0 \end{pmatrix},$$

where U_{11} is related to A as $U_{11} = \frac{1}{\lambda} A$. Given that the SVD of A is given by $A = V \Sigma W^\dagger$ as described above, we can substitute U_{11} into the expression for $\Pi_{\phi_{2k}} U_A^\dagger \Pi_{\phi_{2k+1}} U_A$. We can calculate $U_{11}^\dagger U_{11}$ as follows since V is unitary with Lemma 2.1:

$$U_{11}^\dagger U_{11} = \left(\frac{1}{\lambda} W \Sigma^\dagger V^\dagger\right)\left(\frac{1}{\lambda} V \Sigma W^\dagger\right) = \frac{1}{\lambda^2} W \Sigma^\dagger \Sigma W^\dagger,$$

where $\Sigma^\dagger \Sigma$ is the square of the diagonal matrix of singular values. As described in [MRTC21], we can assume that the block encoding U_A for A is given as:

$$U_A = \begin{pmatrix} A & \sqrt{I - AA^T} \\ \sqrt{I - A^T A} & -A^T \end{pmatrix}$$

where the matrix square root is defined through its singular value decomposition as follows:

$$\sqrt{I - AA^T} := \sum_{k=1}^{n} \sqrt{1 - \sigma_k^2} |v_k\rangle\langle v_k|, \quad \sqrt{I - A^T A} := \sum_{k=1}^{n} \sqrt{1 - \sigma_k^2} |w_k\rangle\langle w_k|.$$

where σ_k are the singular values of A, and $|v_k\rangle$ and $|w_k\rangle$ are the left and right singular vectors of A, as described above. Then the term that dominates this expression is the first one:

$$U_A^\dagger U_A = \begin{pmatrix} U_{11}^\dagger U_{11} & U_{11}^\dagger U_{12} \\ U_{21}^\dagger U_{11} & U_{21}^\dagger U_{12} + U_{22}^\dagger U_{22} \end{pmatrix},$$

Therefore, after applying the sequence for $d/2$ steps, we can apply the QSP Theorem 2.12 which gives us for the signals $\vec{\phi} = (\phi_0, \phi_1, \ldots, \phi_d) \in \mathbb{R}^{d+1}$ such that for $\sigma_k \in [-1, 1]$ for all $k \in \{1, \ldots, r\}$ with $Q \equiv 1$ and ensures that the block in the upper-left corner of the resulting matrix is given by:

$$\Pi U_{\vec{\phi}} \Pi = V \Sigma' V^\dagger,$$

where $\Sigma' \in \mathbb{R}^{n \times n}$ is given as

$$\Sigma' = \operatorname{diag}(P(\sigma_1), P(\sigma_2), \ldots, P(\sigma_r)),$$

and $P(\sigma_k)$ is the polynomial from QSP applied to σ_k for all $k \in \{1, \ldots, r\}$. This corresponds to the polynomial applied to the singular values of A through the QSVT sequence.

$$\operatorname{Poly}^{(\mathrm{SV})}(A) := \sum_k \operatorname{Poly}(\sigma_k) \left|v_k\right\rangle \left\langle v_k\right|$$

To show for odd d, follow exactly the same way. The other direction can be shown with the same argument (see [GSLW18]). □

Remark 2.32 (Prerequisites to the Polynomial Used in QSVT and QSP)**.** *The QSP Theorem 2.12 specifies that since for QSVT the use of only P is relevant, we can use $Q \equiv 1$ to simplify the condition $|P|^2 + (1 - a^2)|Q|^2 = 1$ for $a \in [-1, 1]$ with $|P| \leq 1$. Therefore, $|P(x)| \leq 1$ for all $x \in [-1, 1]$. Thus, although the QSVT Theorem 2.17 states that P must satisfy all the conditions of P from Theorem 2.12, it is only relevant to show $|P(x)| \leq 1$ for all $x \in [-1, 1]$.*

2.4.3.3 Using QSVT for Matrix Inversion

The process of matrix inversion using Quantum Singular Value Transformation (QSVT) was proposed and detailed in [MRTC21]. This approach allows for the inversion of a given square matrix by constructing an approximation to its inverse. Traditionally, this task has been addressed by the HHL algorithm, developed by Harrow, Hassidim, and Lloyd [HHL09].

Definition 2.43 (ϵ-approximation)**.** *Let $f : D \to \mathbb{R}$ be a function defined on a domain $D \subseteq \mathbb{R}$, and let $\epsilon > 0$. A function $g : D \to \mathbb{R}$ is said to be an ϵ-approximation of f on D if for all $x \in D$:*

$$|g(x) - f(x)| \leq \epsilon.$$

This definition ensures that the maximum deviation between the approximating function $g(x)$ and the target function $f(x)$ is bounded by ϵ across the specified domain. The following lemma demonstrates how the inversion of a matrix can be achieved through the inversion of its SVD. It shows that inverting a matrix can be reduced to inverting its singular values, which is utilized by QSVT to perform matrix inversion.

Lemma 2.1 (SVD Inversion)**.** *Let $A \in \mathbb{C}^{n\times n}$ be invertible with $A = U\Sigma V^*$ as the SVD of A. Then the inverse of A is given by:*

$$A^{-1} = V\Sigma^{-1}U^*,$$

where $U \in \mathbb{C}^{n\times n}$ and $V \in \mathbb{C}^{n\times n}$ are unitary matrices and $\Sigma^{-1} \in \mathbb{C}^{n\times n}$ is defined by taking the reciprocal of each non-zero singular value in Σ.

Proof. Given $A = U\Sigma V^* \in \mathbb{C}^{n\times n}$, we need to prove that $A^{-1} = V\Sigma^{-1}U^*$. Since A is invertible, all singular values are non-zero, and Σ is invertible. We verify:

$$AA^{-1} = (U\Sigma V^*)(V\Sigma^{-1}U^*) = U\Sigma(V^*V)\Sigma^{-1}U^* = U\Sigma\Sigma^{-1}U^* = UIU^* = UU^* = I$$
$$A^{-1}A = (V\Sigma^{-1}U^*)(U\Sigma V^*) = V\Sigma^{-1}(U^*U)\Sigma V^* = V\Sigma^{-1}\Sigma V^* = VIV^* = VV^* = I$$

In both cases, we used the fact that U and V are unitary. □

Remark 2.33. *To summarize, for an invertible matrix $A = U\Sigma V^* \in \mathbb{C}^{n\times n}$, the Quantum Singular Value Transformation (QSVT) can be used to directly approximate the inverse A^{-1}. When we use QSVT to invert singular values, we employ a polynomial approximation of the function $f(x) = 1/x$. This approximation can invert singular values up to a precision ϵ. The resulting matrix, after applying QSVT, approximates the inverse A^{-1}. If we use a polynomial that exactly inverts the singular values (without any approximation error), the result is the exact inverse A^{-1} of A, as discussed in Lemma 2.1 above. Note that for an invertible matrix—in our case, the reduced susceptance matrix, which is invertible—there is no need to consider the Moore-Penrose pseudoinverse (see [SB02]). QSVT directly provides an approximation of A^{-1} with the desired precision ϵ. For a non-invertible or non-square matrix, QSVT would still provide an approximate inversion, resulting in a matrix close to the Moore-Penrose pseudoinverse A^+. However, since A is invertible in our context, we obtain an approximation of A^{-1} itself. For QSVT of non-square matrices, see [MRTC21].*

2.4.4 Quantum Approximate Optimization Algorithm

The Quantum Approximate Optimization Algorithm (QAOA) was introduced by Farhi et al. [FGG14] to provide approximate solutions for combinatorial optimization problems. The algorithm is designed to solve optimization problems by finding the ground state of a classical Hamiltonian $\hat{C}$, which encodes the cost function of the optimization problem. The objective is to find the bit string $z \in \{0,1\}^n$ that minimizes a given cost function $C(z)$. This can be translated into finding the ground state, which corresponds to the minimum eigenvalue of the Hamiltonian $\hat{C}$. We follow the notation of [FGG14] and introduce the two main components of the algorithm:

- The Cost Hamiltonian $\hat{C}$: This encodes the problem we aim to solve.
- The Mixing Hamiltonian $\hat{B}$: Commonly chosen as the transverse field Hamiltonian:

$$\hat{B} = -\sum_{i=1}^{n} \sigma_i^x,$$

where σ_i^x are the Pauli-X matrices acting on the i-th qubit. This Hamiltonian serves to explore the solution space (see [FGG14]).
The QAOA operates by applying a series of unitary transformations parameterized by angles γ and β, which are optimized to maximize the overlap with the ground state of $\hat{C}$.

2.5 Approximation Methods for Polynomial Construction

To establish a polynomial that can effectively invert matrices via Quantum Singular Value Transformation, we need to approximate continuous functions that are infinitely differentiable. This approximation relies heavily on the properties of functions within L^p-spaces. Therefore, we require the following foundational concepts from [LL01].

Definition 2.44. *A Borel set is any set in the σ-algebra generated by the open sets in $\mathbb{R}$.*

Definition 2.45. *The Lebesgue measure λ of an interval $[a,b] \subset \mathbb{R}$ is defined by*

$$\lambda([a,b]) = b - a.$$

Definition 2.46. *Let $1 \leq p < \infty$. The L^p space, denoted by $L^p(\mathbb{R})$, is the set of all measurable functions $f : \mathbb{R} \to \mathbb{R}$ such that*

$$\|f\|_p := \left(\int_{\mathbb{R}} |f(x)|^p \, dx \right)^{1/p} < \infty.$$

Definition 2.47. *The L^p-norm of a function $f \in L^p(\mathbb{R})$, where $1 \leq p < \infty$, is defined by*

$$\|f\|_p = \left(\int_{\mathbb{R}} |f(x)|^p \, dx \right)^{1/p}.$$

Definition 2.48. *The support of a function $f : \mathbb{R} \to \mathbb{C}$ is the closure of the set where f is non-zero:*

$$supp(f) = \overline{\{x \in \mathbb{R} : f(x) \neq 0\}}.$$

Algorithm 2.2: Quantum Approximate Optimization Algorithm (QAOA)

Input : Number of qubits n, depth p, cost Hamiltonian $\hat{C}$, mixing Hamiltonian $\hat{B}$

Output : Approximate solution to the optimization problem

Procedure :

1 **Initialization:** Start with an equal superposition of all possible states, usually achieved by applying a Hadamard gate $H^{\otimes n}$ to the n-qubit initial state $|0\rangle^{\otimes n}$.

2 **Parameterized Quantum Circuit:**

1. Apply the cost unitary $U_C(\gamma) = e^{-i\gamma\hat{C}}$.
2. Apply the mixing unitary $U_B(\beta) = e^{-i\beta\hat{B}}$.
3. Repeat the above two steps p times, where p is the depth of the QAOA circuit, leading to the final state:

$$|\psi(\vec{\gamma},\vec{\beta})\rangle = U_B(\beta_p)U_C(\gamma_p)\cdots U_B(\beta_1)U_C(\gamma_1)H^{\otimes n}|0\rangle^{\otimes n}$$

Here, $\vec{\gamma} = (\gamma_1, \gamma_2, \ldots, \gamma_p)$ and $\vec{\beta} = (\beta_1, \beta_2, \ldots, \beta_p)$ are the parameters to be optimized.

Measurement: Measure the final state in the computational basis to obtain bit strings.

Classical Optimization: Use a classical optimization algorithm, e.g., gradient descent, Nelder-Mead (see [ZWC+20]), to find parameters $(\vec{\gamma}^*, \vec{\beta}^*)$ that yield good approximations of the optimal parameters by maximizing the expectation value:

$$\langle\psi(\vec{\gamma},\vec{\beta})|\hat{C}|\psi(\vec{\gamma},\vec{\beta})\rangle.$$

3 Related work

The exploration of quantum computing for solving the Unit Commitment Problem has attracted significant attention due to its potential to outperform classical methods. In [AY19], the UCP was reformulated as a Quadratic Unconstrained Binary Optimization (QUBO) problem, applying QAOA on D-Wave processors. This approach showed promise but faced scalability issues due to hardware limitations. In [KGB+21], a hybrid method was introduced, combining QAOA for binary variables with classical optimization for continuous variables, which improved accuracy for small-scale problems. In [NZB22], a distributed quantum approach to UCP was proposed, breaking it down into QUBO subproblems solved using QAOA. This method allows for scalability and is particularly suited for large-scale energy systems where computational resources are limited. In [JPJL10], a quantum variant of Binary Particle Swarm Optimization (PSO) for UCP was introduced. This method performed well on systems with up to 100 units, though its computational time increased quadratically with system size, highlighting scalability concerns. QSVT, developed in [GSLW18] and expanded in [MRTC21], offers a flexible and efficient approach for solving linear systems, critical in UCP and power flow problems. Compared to HHL, QSVT handles a broader range of problems, making it well-suited for large-scale quantum optimization tasks. QSVT is increasingly integrated into power systems optimization, showing improvements in computational efficiency and accuracy over classical methods. Its flexibility makes it preferable over HHL in scenarios where matrix inversion is a key step. In [HHPT23], quantum surrogate models, combining classical surrogate models with quantum-enhanced optimization, have shown promise in improving solution speed and quality, particularly for sub-hourly commitments in renewable energy contexts. The integration of QAOA within hybrid frameworks has demonstrated significant improvements in computational efficiency and solution quality, especially given current quantum hardware limitations [HHPT23].

4 Methodology

4.1 Formulation of the Unit Commitment Optimization Problem

Definition 4.1 (Quantum simulation-based optimization)**.** *For an objective function $f : \mathbb{R}^N \times \mathbb{R}^L \to \mathbb{R}$, with $u : \mathbb{R}^M \to \mathbb{R}^L$, a quantum simulation-based optimization (QuSO) problem is defined as a special form of a MINLP problem, where f is efficiently implementable and may depend on the summary statistic result of the solution of a SLE:*

$$\begin{aligned} \min_{x} \quad & f(x, u(y)) \\ \text{subject to} \quad & A_x y = b_x, \\ & c_j(x, u(y)) \leq 0, \ \forall j \in [K], \\ & x_i \in [l_i, u_i] \subset \mathbb{R}, \\ & x_i \in \mathbb{Z}, \ \forall i \in I \subseteq [N], \end{aligned}$$

where $A_x \in \mathbb{R}^{R \times M}$ and $b_x \in \mathbb{R}^M$ represent a SLE that depends on the solution $x \in X$, where $x \mapsto A_x$ and $x \mapsto b_x$ are continuous.

Remark 4.1. *Definition 4.1 of a QuSO problem is compliant with Definition 2.2 from Section 2.1 of a MINLP problem, since the objective function $f(x, u(y))$ is demanded to be an algebraic function. The constraints $A_x y = b_x$ are defined as linear equations dependent on x. The constraints $c_j(x, u(y)) \leq 0$ are algebraic. The variable bounds $x_i \in [l_i, u_i] \subset \mathbb{R}$ define a polyhedral set and the variables x_i must be integers $\forall i \in I \subseteq [N]$. Thus, the QuSO problem can be viewed as a specific instance of a MINLP problem.*

In order to make the Susceptance Matrix invertible, we use the procedure described in Section 2.3.2 which assumes a connected power network graph, non-zero line susceptances for all buses, and a defined reference bus. The reference bus is explicitly defined in the constraints of the problem. For the formulation of the Unit Commitment Problem (UCP), we consider an $m+1$-bus AC system with $m \in \mathbb{N}$ and $m \geq 2$, under the relaxation assumptions described in Section 2.3.1 and thus make use of the DC power flow approximations from Theorem 2.8. We formulate the UCP as a generalization of the Economic Dispatch (ED) and Optimal Power Flow (OPF) problem, introducing a time component with discrete timesteps $t \in T$. Note that the following formulation of the UCP defines a MINLP according to Def. 2.2 from Section 2.1.

$$\min_{\mathbf{u},\mathbf{P}_\mathrm{G}} \quad \sum_{t\in\mathrm{T}} \left(\sum_{i\in\mathrm{G}} \left(c_{i,t}^{\mathrm{prod}} \cdot u_{i,t} + c_{i,t}^{\mathrm{start}} \cdot (1-u_{i,t-1})u_{i,t} + c_{i,t}^{\mathrm{stop}} \cdot u_{i,t-1}(1-u_{i,t}) \right) + \sum_{l\in\mathrm{L}} c_{l,t}^{\mathrm{trans}} \right)$$

subject to:

$$\begin{aligned}
&\sum_{i\in\mathrm{G}} P_{i,t} = \sum_{k\in\mathrm{B}} L_{k,t}, && \forall t \in \mathrm{T} \\
&P_i^{\min} \cdot u_{i,t} \le P_{i,t} \le P_i^{\max} \cdot u_{i,t}, && \forall i \in \mathrm{G},\ \forall t \in \mathrm{T} \\
&u_{i,t} \in \{0,1\}, && \forall i \in \mathrm{G},\ \forall t \in \mathrm{T} \\
&c_{l,t}^{\mathrm{trans}} \ge P_{l,t} \cdot C_l, && \forall l \in \mathrm{L},\ \forall t \in \mathrm{T} \\
&c_{l,t}^{\mathrm{trans}} \ge -P_{l,t} \cdot C_l, && \forall l \in \mathrm{L},\ \forall t \in \mathrm{T} \\
&B'\theta_t' = P_t' - L_t', && \forall t \in \mathrm{T} \\
&\theta_{m+1,t} = 0, && \forall t \in \mathrm{T}
\end{aligned}$$

Where:

- $m \in \mathbb{N}$ with $m \ge 2$ is a finite number of buses.
- $m+1$ is the index of the reference bus.
- $\mathrm{B} = \{1, 2, \ldots, m\}$ is the index set of buses.
- $\mathrm{G} \subseteq \mathrm{B}$ is the index set of generators.
- L is a finite index set of transmission lines.
- T is a finite set of discrete time steps.

- $P_{\min,i} \in \mathbb{R}$ is the minimum power output limit of generator i.
- $P_{\max,i} \in \mathbb{R}$ is the maximum power output limit of generator i.
- $c_{i,t}^{\mathrm{prod}} \in \mathbb{R}$ is the production cost of generator i at time t.
- $c_{i,t}^{\mathrm{start}} \in \mathbb{R}$ is the start-up cost of generator i at time t.
- $c_{i,t}^{\mathrm{stop}} \in \mathbb{R}$ is the shut-down cost of generator i at time t.
- $c_{l,t}^{\mathrm{trans}} \in \mathbb{R}$ is the transmission cost through line l at time t.
- $C_l \in \mathbb{R}$ is the cost per unit of power transmission per unit time for line l.

- $u_{i,t} \in \{0,1\}$, with $u_{i,t} = 0$ if off and $u_{i,t} = 1$ if on, the operational state of generator i at time t
- $\mathbf{u} \in \{0,1\}^{(|\mathrm{G}|\cdot|\mathrm{T}|)}$ with $\mathbf{u} = (u_{i,t})_{i\in\mathrm{G},t\in\mathrm{T}}$ the vector of all generator state variables.
- $P_{i,t} \in \mathbb{R}$ is the power generated by generator i at time t.
- $\mathbf{P}_\mathrm{G} \in \mathbb{R}^{(|\mathrm{G}|\cdot|\mathrm{T}|)}$ with $\mathbf{P}_\mathrm{G} = (P_{i,t})_{i\in\mathrm{G},t\in\mathrm{T}}$ the vector of all the power generated by generator i at time t.
- $L_{k,t} \in \mathbb{R}$ is the load at bus k and time t.
- $P_{l,t} \in \mathbb{R}$ is the power flowing through line l at time t.
- $\theta_{k,t} \in \mathbb{R}$ is the voltage angle at bus k at time t.

- $B' \in \mathbb{R}^{m\times m}$ (Def. 2.22 from Section 2.3.1) is the reduced susceptance matrix, obtained by excluding the rows and columns connected to the reference bus $m+1$.
- $\theta_t' = (\theta_{1,t}, \theta_{2,t}, \ldots, \theta_{m,t})^\top \in \mathbb{R}^m$ is the reduced vector of voltage angles $\theta_{k,t} \in \mathbb{R}$ at bus k and time t, excluding the reference bus $m+1$.

- $P'_t = (P_{1,t}\, u_{1,t},\, P_{2,t}\, u_{2,t},\, \ldots,\, P_{m,t}\, u_{m,t})^\top \in \mathbb{R}^m$ is the reduced vector of total power generation, where $P_{k,t} \in \mathbb{R}$ is the power injected by generator k at bus k and time t and $u_{k,t} = 0$ for $k \in B \setminus G$, excluding the reference bus $m+1$.
- $L'_t = (L_{1,t},\, L_{2,t},\, \ldots,\, L_{m,t})^\top \in \mathbb{R}^m$ is the reduced vector of total loads (power extraction), where $L_{k,t} \in \mathbb{R}$ is the power extracted at bus k and time t, excluding the reference bus $m+1$.

Remark 4.2. *Note that the Generator Parameters: $P_{\min,i}$, $P_{\max,i}$, $c_{i,t}^{prod}$, $c_{i,t}^{start}$, $c_{i,t}^{stop}$ as well as the Transmission Line Parameters: $c_{l,t}^{trans}$, C_l, B and the Load and Network Parameters: $L_{i,t}$, B' are all known values and determined by the setup of the underlying energy grid for a given $i \in G$, $l \in L$ and $t \in T$. The variables yet to be determined are the Operational State of Generators: $u_{i,t}$, $P_{i,t}$ as well as Voltage Angles: $\theta_{k,t}$ for a given $i \in G$, $k \in B$ and $t \in T$.*

A concrete representation of L, the index set of transmission lines, can be found in Section 2.2. The construction of $B' \in \mathbb{R}^{m \times m}$, the reduced susceptance matrix, which excludes the rows and columns connected to the reference bus $m+1$, can be found in Def. 2.22 from Sec. 2.3.2. The explicit derivation and concrete construction of the DC-Powerflow $B'\theta' = P'_t - L'_t$ for $t \in$ T in its matrix form is found in Section 2.3.1.
A broader formulation including more constraints and a specific representation of $c_{i,t}^{\text{prod}}$, $c_{i,t}^{\text{start}}$ and $c_{i,t}^{\text{stop}}$ as well as the transmission cost $c_{l,t}^{\text{trans}}$ can be found in [dBDD14] and [dBBDD14]. These constraints are consistent with our definition and can be added modularly for extended formalism.

Remark 4.3. *Note that for a given timestep $t \in T$, when we write out the matrix-vector product $B' \cdot \theta'_t = P'_t - L'_t$ in its summation form, it explicitly involves terms of the form $\theta_{i,t} - \theta_{j,t}$ as seen in Section 2.3.1. These terms represent the voltage angle differences between connected buses. For a specific bus i, the equation $B' \cdot \theta'_t = P'_t - L'_t$ can be written according to Lem. 2.1 from Section 2.3.1 as follows:*

$$\sum_{j=1}^{m} B_{ij}\theta_{j,t} = P_{i,t} - L_{i,t}, \quad \forall i \in \{1, 2, \ldots, m\},\ \forall t \in T$$

The elements B_{ij} of the matrix are determined by the network topology and the line susceptances. B_{ii} represent the sum of susceptances connected to bus i. B_{ij} are negative and represent the susceptance of the line between bus i and bus j (see Def. 2.20). Given that $B_{ij} = -\frac{1}{x_{ij}}$ for $i \neq j$ (where x_{ij} is the reactance of the line between buses i and j), and B_{ii} is the sum of the inverse reactances of the lines connected to bus i:

$$\sum_{j=1, j\neq i}^{m} \frac{\theta_{i,t} - \theta_{j,t}}{x_{ij}} = P_{i,t} - L_{i,t}$$

We can reformulate the UCP within the QuSO framework, making it suitable for processing by the proposed quantum algorithm. Note that a precise explanation of all components in this problem is given at the beginning of this section, we will use them implicitly for the next definition. The following definition encapsulates this transformation:

Definition 4.2 (UCP as Quantum simulation-based optimization)**.** *For an objective function $f : \mathbb{R}^{|G|\cdot|T|+|G|\cdot|T|} \times \mathbb{R}^{|B|\cdot|T|} \to \mathbb{R}$, with $u : \mathbb{R}^{|B|\cdot|T|} \to \mathbb{R}^{|B|\cdot|T|}$ with $u : y \mapsto y$,*

the following definition is a quantum simulation-based optimization (QuSO) problem:

$$f(x, u(y)) := \sum_{t\in T}\sum_{i\in G}\Big(c_{i,t}^{prod}\cdot u_{i,t} + c_{i,t}^{start}\cdot(1-u_{i,t-1})u_{i,t} + c_{i,t}^{stop}\cdot u_{i,t-1}(1-u_{i,t})\Big) + \sum_{t\in T}\sum_{l\in L} c_{l,t}^{trans}$$

where $x = (u_{i,t},\, P_{i,t})_{i\in G, t\in T}$ *for* $u_{i,t} \in \{0,1\} \subset \mathbb{Z}$, $P_{i,t} \in [l_i, u_i] \subset \mathbb{R}$ *with* $l_i < u_i$ *for* $i \in G$ *and* $t \in T$ *and where* $y = (\theta_{k,t})_{k\in B, t\in T}$ *for* $\theta_{k,t} \in \mathbb{R}$ *with* $k \in B$ *and* $t \in T$. *Then the QuSO is defined as the following:*

$$\begin{aligned}
&\min_{x} && f(x, u(y)) && \\
&\text{subject to} && \mathbf{B}'y = \Delta P && \\
& && \sum_{i\in G} P_{i,t} - \sum_{k\in B} L_{k,t} && \leq 0 \quad \forall t \in T \\
& && -\sum_{i\in G} P_{i,t} + \sum_{k\in B} L_{k,t} && \leq 0 \quad \forall t \in T \\
& && P_i^{\min} u_{i,t} - P_{i,t} && \leq 0 \quad \forall t \in T \\
& && -P_i^{\max} u_{i,t} + P_{i,t} && \leq 0 \quad \forall t \in T \\
& && P_{l,t} C_l - c_{l,t}^{trans} && \leq 0 \quad \forall t \in T \\
& && -c_{l,t}^{trans} - P_{l,t} C_l && \leq 0 \quad \forall t \in T \\
& && \theta_{m+1,t} && \leq 0 \quad \forall t \in T \\
& && -\theta_{m+1,t} && \leq 0 \quad \forall t \in T
\end{aligned}$$

where $\mathbf{B}' \in \mathbb{R}^{|B|\cdot|T|\times|B|\cdot|T|}$ *and* $\Delta P \in \mathbb{R}^{|B|\cdot|T|}$ *with* $\Delta P = (P_{k,t}\, u_{k,t} - L_{k,t})_{k\in B, t\in T}$ *represent a SLE that depends on the solution* $y \in \mathbb{R}^{|B|\cdot|T|}$. *This SLE is defined with the block matrices* $0 \in \mathbb{R}^{|B|\times|B|}$ *through* $B' \in \mathbb{R}^{|B|\times|B|}$, *the reduced susceptance matrix, in the following way:*

$$\mathbf{B}' = \begin{bmatrix} B' & 0 & 0 & \cdots & 0 \\ 0 & B' & 0 & \cdots & 0 \\ 0 & 0 & B' & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & B' \end{bmatrix}$$

Remark 4.4. *We will refer to* $\mathbf{B}' \in \mathbb{R}^{|B|\cdot|T|\times|B|\cdot|T|}$ *as the reduced block susceptance matrix.*

Remark 4.5. *In order to process the transmission line flows accordingly, we will use, corresponding to the notation of* $\mathbf{P}_G$, *the vector* $\mathbf{P}_L \in \mathbb{R}^{(|L|\cdot|T|)}$ *with* $\mathbf{P}_L = (P_{l,t})_{l\in L, t\in T}$ *the vector of all the power flowing through transmission line* l *at time* t. *This vector lists the power flow values* $P_{l,t}$ *in a lexicographical order by time* t *and line* l.

Remark 4.6. *In the above definition of the optimization problem we follow the convention*

to formulate $P_{l,t}C_l - c_{l,t}^{trans} \leq 0$ *and* $-c_{l,t}^{trans} - P_{l,t}C_l \leq 0$ *for* $t \in T$ *and* $l \in L$ *as inequality constraints. To process them more easily, we will use the equivalent form* $c_{l,t}^{trans} = |P_{l,t}C_l|$.

Remark 4.7. *The diagonal block structure of* $\mathbf{B}'$ *is convenient because it preserves the properties of the individual block matrix* B'. *Specifically,* $\mathbf{B}'$ *is invertible if and only if* B' *is invertible. Moreover, the eigenvalues of* $\mathbf{B}'$ *are the same as the eigenvalues of* B', *repeated* $|T|$ *times. Consequently, the condition number of* $\mathbf{B}'$ *is identical to the condition number of* B'.

Remark 4.8. *Note that the above inequality constraints* $c_{j,t}(x, u(y)) \leq 0$ *with* $j = 1, \ldots, 8$ *are defined for every timestep* $t \in T$. *This means that every constraint of the form* $c_{j,t}(x, u(y)) \leq 0$ *is given* $|T|$*-times i.e., the total number of these constraints is* $8 \cdot |T|$.

Remark 4.9. *The above version of the Unit Commitment Problem (UCP) is well-defined within the Definition 4.1 of a QuSO. In the UCP formulation, the objective function* $f(x, u(y))$ *maps* $x \in \mathbb{R}^{|G|\cdot|T|+|G|\cdot|T|}$ *and* $u(y) \in \mathbb{R}^{|B|\cdot|T|}$ *to* $\mathbb{R}$, *aligning with this requirement. In the UCP,* $u : \mathbb{R}^{|B|\cdot|T|} \to \mathbb{R}^{|B|\cdot|T|}$ *is defined as* $u(y) = y$, *satisfying this condition. The QuSO definition specifies constraints* $A_x y = b_x$ *and* $c_j(x, u(y)) \leq 0$. *In the UCP, the equality constraint* $\mathbf{B}'y = \Delta P$ *and various inequalities involving* $P_{i,t}$, $\theta_{m+1,t}$, *and other terms fit this representation. The QuSO definition includes decision variables* x, *which can be real numbers or integers within specific bounds. The UCP decision variables* $x = (u_{i,t}, P_{i,t})$ *and* $y = (\theta_{k,t})$ *include both binary and continuous variables, each constrained within specific bounds, consistent with the QuSO formulation.*

Remark 4.10 (Rescaling singular values of $\mathbf{B}'$)**.** *In order to process* $\mathbf{B}'$ *with the QSVT, the matrix inversion polynomial, which is defined in the following section, requires that all singular values* σ_i *for* $\mathbf{B}'$ *are within* $\sigma_i \in [\frac{1}{\kappa}, 1]$ *for* $\kappa \geq 1$, *where* κ *is the condition number of* $\mathbf{B}'$. *The singular values of a matrix* $\mathbf{B}'$ *are defined as the square roots of the eigenvalues of the matrix multiplied by its transpose. If* λ_i *are the eigenvalues of* $\mathbf{B}'^T\mathbf{B}'$, *then the singular values* σ_i *are given by* $\sigma_i = \sqrt{\lambda_i}$ *for* $1 \leq i \leq m$. *Since the eigenvalues* λ_i *of the matrix* $\mathbf{B}'^T\mathbf{B}'$ *are non-negative, because it is also a positive semi-definite matrix, the singular values* σ_i, *being the square roots of these non-negative eigenvalues, are also non-negative. Thus* $\sigma_i > 0$. *To transform the matrix such that all its singular values* σ_i *lie within the interval* $[1/\kappa, 1]$ *for* $\kappa \geq 1$, *we can rescale it as shown in Algorithm 4.3. The rescaling does not lose any information for our optimization problem, to avoid confusion with indexes we assume* $\mathbf{B}'$ *as rescaled implicitly.*

Algorithm 4.3: Rescaling $\mathbf{B}'$ to Ensure Singular Values Within $\left[\frac{1}{\kappa}, 1\right]$

Input : Matrix $\mathbf{B}' \in \mathbb{R}^{|B||T|\times|B||T|}$
Output : Rescaled matrix $\mathbf{B}'_r$ such that all singular values $\sigma_i \in \left[\frac{1}{\kappa}, 1\right]$
Procedure:

1. For $n = |B| \cdot |T|$. Compute the SVD Thm. 2.16 from Sec. 2.4.3 of $\mathbf{B}'$:

$$\mathbf{B}' = U\Sigma V^T$$

   where $U \in \mathbb{R}^{n\times n}$ and $V \in \mathbb{R}^{n\times n}$ are orthogonal matrices, and $\Sigma \in \mathbb{R}^{n\times n}$ is a diagonal matrix with singular values $\sigma_1, \sigma_2, \ldots, \sigma_n$.
2. Determine the largest singular value $\sigma_{\max} = \max(\sigma_1, \sigma_2, \ldots, \sigma_n)$.
3. Rescale the matrix $\mathbf{B}'$ to ensure the largest singular value is at most 1:

$$\tilde{\mathbf{B}}' = \frac{\mathbf{B}'}{\sigma_{\max}} = U\left(\frac{\Sigma}{\sigma_{\max}}\right)V^T$$

   Now, the singular values of $\tilde{\mathbf{B}}'$ are $\sigma'_i = \frac{\sigma_i}{\sigma_{\max}}$, with $\sigma'_{\max} = 1$.
4. Determine the smallest singular value $\sigma'_{\min}$ of $\tilde{\mathbf{B}}'$:

$$\sigma'_{\min} = \min(\sigma'_1, \sigma'_2, \ldots, \sigma'_n)$$

5. Check if $\sigma'_{\min} \geq \frac{1}{\kappa}$. If true, set $\mathbf{B}'_r = \tilde{\mathbf{B}}'$. If false, further rescale $\tilde{\mathbf{B}}'$:

$$\mathbf{B}'_r = \frac{\kappa}{\sigma'_{\min}} \cdot \tilde{\mathbf{B}}' = \frac{\kappa}{\sigma'_{\min}} \cdot \frac{\mathbf{B}'}{\sigma_{\max}}$$

6. The final matrix $\mathbf{B}'_r$ now has all singular values within the interval $\left[\frac{1}{\kappa}, 1\right]$.

Algorithm 4.4: Rescaling $\mathbf{B}'$ Using Condition Number Estimates

Input : Matrix $\mathbf{B}' \in \mathbb{R}^{|B||T| \times |B||T|}$
Output : Rescaled matrix $\mathbf{B}'_r$
Procedure:

1. Estimate the condition number $\kappa(\mathbf{B}')$ using the Rayleigh Quotient or Cheeger constant (see Section 2.2.1).
2. Rescale the matrix $\mathbf{B}'$ to ensure the largest singular value is at most 1:

$$\tilde{\mathbf{B}}' = \frac{\mathbf{B}'}{\sigma_{\max}}$$

   where $\sigma_{\max}$ is the largest singular value of $\mathbf{B}'$, estimated from the condition number $\kappa(\mathbf{B}')$.
3. Determine the smallest singular value $\sigma'_{\min}$ of $\tilde{\mathbf{B}}'$. If $\sigma'_{\min} \geq \frac{1}{\kappa(\mathbf{B}')}$, set $\mathbf{B}'_r = \tilde{\mathbf{B}}'$. If $\sigma'_{\min} < \frac{1}{\kappa(\mathbf{B}')}$, further rescale $\tilde{\mathbf{B}}'$:

$$\mathbf{B}'_r = \frac{\tilde{\mathbf{B}}'}{\sigma'_{\min} \cdot \kappa(\mathbf{B}')}$$

4. Output the final rescaled matrix $\mathbf{B}'_r$.

Remark 4.11 (Rescaling singular values of $\mathbf{B}'$ using estimated condition number)**.** *Given the high algebraic connectivity λ_2 (see Section 2.2) typically present in the power grids considered in this work, the matrix $\mathbf{B}'$ is expected to be well-conditioned. High algebraic connectivity, characterized by a significant second smallest eigenvalue λ_2 of the Laplacian matrix, ensures that the condition number $\kappa(\mathbf{B}')$ is naturally low (as described in Remark 2.9 from Section 2.2). Consequently, the singular values σ_i of $\mathbf{B}'$ are well-distributed within the interval $[1/\kappa(\mathbf{B}'), 1]$.*
By estimating the condition number using the Rayleigh Quotient or the Cheeger constant (see Section 2.2.1), we can effectively rescale the matrix to achieve the desired singular value distribution (see Algorithm 4.4). Importantly, these estimation and rescaling steps are computationally efficient, maintaining the overall efficiency of the quantum algorithm and ensuring that the exponential speedup is preserved.

4.2 Polynomial Approximation for QSVT

4.2.1 Overview of the Polynomial Approximations

In our quantum algorithm, we employ two distinct Quantum Singular Value Transformation (QSVT) procedures. The first is used to invert the reduced block susceptance matrix $\mathbf{B}' \in \mathbb{R}^{m\cdot|T|\times m\cdot|T|}$, and the second calculates the absolute value of the expression $c_{l,t}^{\text{trans}} = |P_{l,t} \cdot C_l|$ for $l \in \text{L}$ and $t \in \text{T}$ (see Section 4.2). These procedures require the construction of two separate polynomials tailored to these tasks.

Matrix Inversion Polynomial To invert a matrix using the Quantum Singular Value Transformation 2.17 from Section 2.4.3, we require a polynomial $P(x)$ that approximates the function $f(x) = \frac{1}{x}$ on the singular values of the reduced block susceptance matrix $\mathbf{B}'$. QSVT demands that the polynomial $P(x)$ be bounded by 1 in magnitude. However, directly approximating $P(x) \approx \frac{1}{x}$ is infeasible since typically $\frac{1}{\sigma_i} \geq 1$ for a singular value σ_i of $\mathbf{B}'$. To overcome this, we focus on approximating a function that behaves as $\frac{1}{2\kappa}\frac{1}{x}$ within the interval $[-1,1]\setminus\left[\frac{-1}{\kappa},\frac{1}{\kappa}\right]$, where κ is the condition number of $\mathbf{B}'$. The condition number is well defined by Lemma 2.1 from Section 2.3.2 and Theorem 2.1 from Section 2.2.

Remark 4.12. *This approach is also valid for any invertible matrix $A \in \mathbb{R}^{n\times n}$ with a finite condition number, and is not limited to the reduced block susceptance matrix $\mathbf{B}' \in \mathbb{R}^{m\cdot|T|\times m\cdot|T|}$ or its blocks $\mathbf{B}' \in \mathbb{R}^{m\times m}$.*

To construct a representation for $P_{\epsilon,\kappa}^{\text{MI}}(x)$, we proceed as follows: In the $P_{\epsilon,\kappa}^{\Theta}$ section, we begin by constructing part of the matrix inversion polynomial, which involves a Chebyshev polynomial approximation to the step function Θ_c. We utilize two techniques (mollifiers) to obtain an analytic C^∞ approximation, which is a variation of the error function. This error function is then approximated by a Chebyshev representation and developed into an approximation to the sign function Θ_0, producing an approximation to the rectangular function. Then we construct a Chebyshev approximation to $\frac{1}{x}$. Since $\frac{1}{x}$ is usually unbounded, we combine it with a polynomial that approximates a rectangular function in $[-1,1]$. Ultimately, we combine the $\frac{1}{x}$ approximation and the rectangular approximation to derive the matrix inversion polynomial. To summarize, the construction of the matrix inversion polynomial $P_{\epsilon,\kappa}^{\text{MI}}(x)$ involves several polynomial approximation steps:

1. Construct the polynomial approximation $P_{\epsilon,\kappa}^{\Theta}$ to the step function.
2. Construct the polynomial approximation $P_{\epsilon,\kappa}^{1/x}$ to the inverse function.
3. Construct the polynomial approximation $P_{\epsilon,\kappa}^{\text{rect}}$ using $P_{\epsilon,\kappa}^{\Theta}$.
4. Combine $P_{\epsilon,\kappa}^{\text{rect}}$ and $P_{\epsilon,\kappa}^{1/x}$ to construct the matrix inversion polynomial $P_{\epsilon,\kappa}^{\text{MI}}(x)$.

Absolute Value Approximation Polynomial To compute the cost function, we need to apply the absolute value to the parameters, necessitating another QSVT routine. Hence, we construct a polynomial that approximates the absolute value function. The absolute value approximation polynomial $P_{\epsilon}^{\text{abs}}(x)$ is designed for $x \in [-1,1]$.

4.2.2 The Matrix Inversion Polynomial

4.2.2.1 Approximation of the Step Function

The following approximation theorem for L^p spaces states that functions in L^p can be approximated by C^∞ functions through convolution with a given mollifier $j \in L^1(\mathbb{R}^n)$.

Theorem 4.1 (Approximation by C^∞-functions [Thm. 2.16 [LL01]])**.** *Let j be in $L^1(\mathbb{R}^n)$ with $\int_{\mathbb{R}^n} j = 1$. For $\varepsilon > 0$, define $j_\varepsilon(x) := \varepsilon^{-n} j(x/\varepsilon)$, so that $\int_{\mathbb{R}^n} j_\varepsilon = 1$ and $\|j_\varepsilon\|_1 = \|j\|_1$. Let $f \in L^p(\mathbb{R}^n)$ for some $1 \leq p < \infty$ and define the convolution for $x \in \mathbb{R}^n$:*

$$f_\varepsilon(x) = (j_\varepsilon * f)(x) = \int_{\mathbb{R}^n} j_\varepsilon(x-y) f(y)\, dy$$

Then the following holds:

- *$f_\varepsilon \in L^p(\mathbb{R}^n)$ and $\|f_\varepsilon\|_p \leq \|j\|_1 \|f\|_p$,*
- *$f_\varepsilon \to f$ strongly in $L^p(\mathbb{R}^n)$ as $\varepsilon \to 0$.*
- *If $j \in C_c^\infty(\mathbb{R}^n)$, then $f_\varepsilon \in C^\infty(\mathbb{R}^n)$*

Proof. For a complete proof see Theorem 2.16 from [LL01]. □

The above theorem is formulated using $\mathbb{R}^n$. It also applies for any measurable set $\Omega \subset \mathbb{R}^n$.

Remark 4.13 (Rem. to [Thm. 2.16 [LL01]])**.** *Let $\Omega \subset \mathbb{R}^n$ be a measurable set. Given $f \in L^p(\Omega)$, we can define $\tilde{f} \in L^p(\mathbb{R}^n)$ by $\tilde{f}(x) = f(x)$ for $x \in \Omega$ and $\tilde{f}(x) = 0$ for $x \notin \Omega$. Then, for $x \in \Omega$, define:*

$$f_\varepsilon(x) = (j_\varepsilon * \tilde{f})(x) = \int_{\mathbb{R}^n} j_\varepsilon(x-y) \tilde{f}(y)\, dy$$

The following holds:

- *$f_\varepsilon \in L^p(\Omega)$ and $\|f_\varepsilon\|_{L^p(\Omega)} \leq \|j\|_1 \|f\|_{L^p(\Omega)}$,*
- *$f_\varepsilon \to f$ strongly in $L^p(\Omega)$ as $\varepsilon \to 0$,*
- *If $j \in C_c^\infty(\mathbb{R}^n)$ and Ω is open, then $f_\varepsilon \in C^\infty(\Omega)$.*

Remark 4.14. *The interval $[-1,1]$, is a Borel set because it can be written as an intersection of open sets:*

$$[-1,1] = \bigcap_{n=1}^{\infty} \left(-1 - \frac{1}{n}, 1 + \frac{1}{n}\right).$$

That means it is measurable in $\mathbb{R}$ with respect to the Lebesgue measure. The Lebesgue measure of Ω is given by $\lambda([-1,1]) = 1 - (-1) = 2$. Note that the interval $(-1,1)$ is open and measurable. As an open set in $\mathbb{R}$, it is a Borel set and hence measurable with respect to the Lebesgue measure.

Since for Remark 4.13 the approximation by C^∞-functions on a measurable set $\Omega \subset \mathbb{R}^n$ is only achieved if it is open, we will apply the approximation to the interior of the closed interval and then generalize it to its boundary.

Lemma 4.1. *The function Θ_c for $c \in (-1,1)$ is defined as:*

$$\Theta_c : [-1,1] \to \mathbb{R}, \quad \Theta_c(x) = \begin{cases} -1 & \text{if } x < c \\ 0 & \text{if } x = c \\ 1 & \text{if } x > c \end{cases}$$

is in $\Theta_c \in L^p([-1,1])$ for $1 \leq p < \infty$.

Proof. We will use the following definition of the L^p norm with $1 \leq p < \infty$:

$$\|\Theta_c\|_{L^p} = \left(\int_{-1}^{1} |\Theta_c(x)|^p \, dx \right)^{\frac{1}{p}} = \left(\int_{-1}^{c} 1 \, dx + \int_{c}^{1} 1 \, dx \right)^{\frac{1}{p}} = 2^{\frac{1}{p}} < \infty$$

Hence $\Theta_c \in L^p([-1,1])$ for $1 \leq p < \infty$. □

Remark 4.15. *Since the integral of $|\Theta_c(x)|^p$ is finite and $\Theta_c(x)$ is measurable on $[-1,1]$ with respect to the Lebesgue measure, we conclude that $\Theta_c \in L^p([-1,1])$ for $1 \leq p < \infty$. The theorem for approximating functions in L^p by C^∞ functions can be applied to Θ_c.*

We first consider the following well known mollifier given by [Eva22].

Definition 4.3. *We call $\eta \in C^\infty(\mathbb{R}^n)$ the **standard mollifier** defined by*

$$\eta(x) := \begin{cases} C \exp\left(\frac{1}{\|x\|^2 - 1}\right) & \text{if } \|x\| < 1 \\ 0 & \text{if } \|x\| \geq 1, \end{cases}$$

the constant $C > 0$ selected so that $\int_{\mathbb{R}^n} \eta \, dx = 1$ and $\|\cdot\|$ is the Euclidean norm.

Remark 4.16. *Let $\eta \in C^\infty(\mathbb{R}^n)$ be the standard mollifier as defined above. For $\varepsilon > 0$, we define the scaled mollifiers $\eta_\varepsilon(x)$ according to Theorem 4.1 by*

$$\eta_\varepsilon(x) := \frac{1}{\varepsilon^n} \eta\left(\frac{x}{\varepsilon}\right).$$

These functions η_ε have the following properties, as described in [Eva22]: It is $\eta_\varepsilon \in C_c^\infty(\mathbb{R}^n)$. Specifically, $supp(\eta_\varepsilon) \subset B(0,\varepsilon)$. This follows directly from the definition of η, which has support in $B(0,1)$. By the scaling property, $\eta_\varepsilon(x) = 0$ for $|x| \geq \varepsilon$. The mollifiers η_ε are normalized such that $\int_{\mathbb{R}^n} \eta_\varepsilon(x) \, dx = 1$. This can be verified through a change of variables $y = \frac{x}{\varepsilon}$. Thus, the functions η_ε satisfy all requirements for a mollifier in Theorem 4.1 to achieve a C^∞ approximation.

Theorem 4.2 (C^∞-approximation of Θ_c by the Standard Mollifier)**.** *Let $\varepsilon > 0$. The function $(\Theta_c')_\varepsilon \in C^\infty((-1,1))$ is a C^∞-approximation to Θ_c according to Theorem 4.1:*

$$(\Theta_c')_\varepsilon(x) := C \int_{\frac{x-c}{\varepsilon}}^{\frac{x+1}{\varepsilon}} \exp\left(\frac{1}{u^2 - 1}\right) du,$$

where $C > 0$ is the normalization constant from the definition of the standard mollifier.

Proof. Let $c \in (-1,1)$. To approximate the step function Θ_c on the interval $(-1,1)$ using the standard mollifier, we follow Theorem 4.1 and Remark 4.13 on C^∞ approximation.

Let $\eta \in C^\infty(\mathbb{R}^n)$ be the standard mollifier and η_ε for $\varepsilon > 0$ its scaled version. Let $\Theta_c \in L^p((-1,1))$ be the step function from Lemma 4.1. By Remark 4.14, $(-1,1)$ is measurable. We extend Θ_c to $\mathbb{R}$ according to Remark 4.13. Define $\tilde{\Theta}_c \in L^p(\mathbb{R})$ for $c \in (-1,1)$ such that:

$$\tilde{\Theta}_c(x) = \begin{cases} \Theta_c(x) & \text{if } x \in (-1,1), \\ 0 & \text{if } x \notin (-1,1). \end{cases}$$

We calculate the convolution of $\tilde{\Theta}_c$ with η_ε as follows:

$$\begin{aligned}(\Theta_c')_\varepsilon(x) = (\eta_\varepsilon * \tilde{\Theta}_c)(x) &= \int_{\mathbb{R}} \eta_\varepsilon(x-y)\tilde{\Theta}_c(y)\,dy \\ &= \int_{-1}^{1} \frac{1}{\varepsilon}\eta\left(\frac{x-y}{\varepsilon}\right)\Theta_c(y)\,dy \\ &= \int_{-1}^{c} \frac{1}{\varepsilon}\eta\left(\frac{x-y}{\varepsilon}\right)(-1)\,dy + \int_{c}^{1} \frac{1}{\varepsilon}\eta\left(\frac{x-y}{\varepsilon}\right)(1)\,dy.\end{aligned}$$

We can substitute with $u = \frac{x-y}{\varepsilon}$. Then $dy = -\varepsilon du$. The limits of integration change accordingly: For $y < c$: $y = -1$ to c becomes $u = \frac{x+1}{\varepsilon}$ to $\frac{x-c}{\varepsilon}$. For $y > c$: $y = c$ to 1 becomes $u = \frac{x-c}{\varepsilon}$ to $\frac{x-1}{\varepsilon}$. Thus:

$$(\Theta_c')_\varepsilon(x) = -\int_{\frac{x+1}{\varepsilon}}^{\frac{x-c}{\varepsilon}} \eta(u)\,du + \int_{\frac{x-c}{\varepsilon}}^{\frac{x-1}{\varepsilon}} \eta(u)\,du.$$

We can insert the definition of η and are left with the following:

$$(\Theta_c')_\varepsilon(x) = C\int_{\frac{x-c}{\varepsilon}}^{\frac{x+1}{\varepsilon}} \exp\left(\frac{1}{u^2-1}\right)\,du.$$

for a normalization constant $C > 0$. Since $\eta \in C^\infty(\mathbb{R})$, the convolution $(\Theta_c')_\varepsilon$ will be $C^\infty((-1,1))$ by Theorem 4.1. □

Corollary 4.1. *Let $\varepsilon > 0$. The function $(\Theta_c')_\varepsilon \in C^\infty([-1,1])$ is a C^∞-approximation to Θ_c on the closed interval $[-1,1]$ when extended continuously to the boundaries.*

Proof. By the properties of the mollifier η_ε, the convolution $(\eta_\varepsilon * \tilde{\Theta}_c)(x)$ from the above proof is C^∞ on $(-1,1)$ by Theorem 4.1. Since $\tilde{\Theta}_c$ is supported on $[-1,1]$, the convolution essentially involves integration over $[-1,1]$. Thus, $(\Theta_c')_\varepsilon$ inherits the smoothness from η_ε on the open interval $(-1,1)$. Consider the behavior as $x \to 1$. For x near 1, the convolution integral becomes:

$$(\Theta_c')_\varepsilon(x) = \int_{-1}^{1} \frac{1}{\varepsilon}\eta\left(\frac{x-y}{\varepsilon}\right)\Theta_c(y)\,dy.$$

Since η has compact support, $\eta\left(\frac{x-y}{\varepsilon}\right) \neq 0$ only when $|x-y| < \varepsilon$. As $x \to 1$, the integral only considers y close to 1, where $\Theta_c'(y) = 1$ for $y > c$ and $\Theta_c'(y) = -1$ for $y < c$. The smoothness of η ensures that this transition is smooth.

For the derivatives of $(\Theta_c')_\varepsilon(x)$, we differentiate under the integral sign:

$$(\Theta_c')_\varepsilon^{(k)}(x) = \int_{-1}^{1} \frac{1}{\varepsilon^{k+1}} \eta^{(k)}\left(\frac{x-y}{\varepsilon}\right) \Theta_c(y)\, dy.$$

As $x \to 1$, the behavior of the derivatives can be analyzed by considering the smoothness of the integrand. The integral of the product of the k-th derivative of the smooth function η and the compactly supported function Θ_c over the interval $[x-\varepsilon, x+\varepsilon]$ remains smooth. This is due to the fact that $\eta^{(k)}\left(\frac{x-y}{\varepsilon}\right)$ scales with $\frac{1}{\varepsilon^{k+1}}$ while $\Theta_c(y)$ is evaluated over a shrinking interval as $x \to 1$, ensuring that the integrand remains smooth and the resulting derivatives has smooth behavior.
A similar argument holds for $x \to -1$. Near $x = -1$, the integral involves y close to -1, where $\Theta_c'(y)$ transitions smoothly due to the properties of η. The differentiation under the integral sign ensures smoothness for all derivatives. Since $(\Theta_c')_\varepsilon$ is C^∞ on $(-1, 1)$ and extends smoothly to $x = \pm 1$ along with its derivatives, we conclude that $(\Theta_c')_\varepsilon \in C^\infty([-1, 1])$. □

Definition 4.4 (Error Function)**.** *The error function* $\operatorname{erf} : \mathbb{R} \to \mathbb{R}$ *is defined as:*

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2}\, dt.$$

Remark 4.17. *The simplified expression for $(\Theta_c')_\varepsilon(x)$ from Theorem 4.2 does not correspond to elementary functions. While the error function itself arises from the Gaussian integral, the integral of η does not match this form. The integrand is not directly relatable to the Gaussian or error function. This convolution is inherently complex and must be handled as a numerical or special function integral without a simpler closed-form representation, which is therefore not suitable for our purpose.*

Therefore, we need to lower our requirements and define the following mollifier without compact support. The Approximation by C^∞-functions Theorem 4.1 then gives us, nevertheless, a strong convergence in $L^p(\mathbb{R}^n)$.

Lemma 4.1 (Gaussian Mollifier)**.** *The function defined by*

$$\varphi : \mathbb{R}^n \to \mathbb{R}, \quad \varphi(x) := (2\pi)^{-n/2} e^{-\frac{\|x\|^2}{2}}$$

where $\|\cdot\|$ is the Euclidean norm, is called the Gaussian Mollifier and satisfies:

- $\varphi \in C^\infty$.
- $\varphi \in L^1(\mathbb{R}^n)$ *with* $\int_{\mathbb{R}^n} \varphi(x)\, dx = 1$.

For $\varepsilon > 0$, with $\varphi_\varepsilon(x) := \varepsilon^{-n}\varphi(x/\varepsilon)$, it holds that $\int_{\mathbb{R}^n} \varphi_\varepsilon(x)\, dx = 1$ and $\|\varphi_\varepsilon\|_1 = \|\varphi\|_1$.

Proof. First note that the Gaussian function $e^{-\frac{\|x\|^2}{2}} \in C^\infty(\mathbb{R}^n)$ because the exponential function e^z is infinitely continuously differentiable for all $z \in \mathbb{R}$, and $(2\pi)^{-n/2}$ is a constant. Thus, $\varphi \in C^\infty(\mathbb{R}^n)$. The solution of the one-dimensional integral is commonly known:

$$\int_{-\infty}^{\infty} e^{-\frac{x_i^2}{2}}\, dx_i = \sqrt{2\pi}.$$

Since $\|x\|^2 = x_1^2 + x_2^2 + \cdots + x_n^2$, the integrand can be separated into a product of n identical one-dimensional integrals:

$$\int_{\mathbb{R}^n} e^{-\frac{\|x\|^2}{2}}\, dx = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} e^{-\frac{x_1^2+x_2^2+\cdots+x_n^2}{2}}\, dx_1\, dx_2\, \cdots\, dx_n = (2\pi)^{n/2}$$

Therefore, we can calculate:

$$\int_{\mathbb{R}^n} \varphi(x)\, dx = \left(\frac{1}{2\pi}\right)^{n/2} \int_{\mathbb{R}^n} e^{-\frac{\|x\|^2}{2}}\, dx = \left(\frac{1}{2\pi}\right)^{n/2} (2\pi)^{n/2} = 1.$$

Since the integral is finite, $\varphi \in L^1(\mathbb{R}^n)$. Now consider $\varphi_\varepsilon(x) = \varepsilon^{-n}\varphi(x/\varepsilon)$. Using the change of variables $u = x/\varepsilon$, so $x = \varepsilon u$ and $dx = \varepsilon^n\, du$:

$$\begin{aligned}
\int_{\mathbb{R}^n} \varphi_\varepsilon(x)\, dx &= \int_{\mathbb{R}^n} \varepsilon^{-n}\varphi(x/\varepsilon)\, dx \\
&= \int_{\mathbb{R}^n} \varepsilon^{-n}\left(\frac{1}{2\pi}\right)^{n/2} e^{-\frac{\|x/\varepsilon\|^2}{2}}\, dx \\
&= \int_{\mathbb{R}^n} \varepsilon^{-n}\left(\frac{1}{2\pi}\right)^{n/2} e^{-\frac{\|u\|^2}{2}}\varepsilon^n\, du \\
&= \int_{\mathbb{R}^n} \left(\frac{1}{2\pi}\right)^{n/2} e^{-\frac{\|u\|^2}{2}}\, du = 1
\end{aligned}$$

Therefore, $\int_{\mathbb{R}^n} \varphi_\varepsilon(x)\, dx = 1$. Finally, we show that $\|\varphi_\varepsilon\|_1 = \|\varphi\|_1$ again using the change of variables $u = x/\varepsilon$:

$$\begin{aligned}
\|\varphi_\varepsilon\|_1 &= \int_{\mathbb{R}^n} |\varphi_\varepsilon(x)|\, dx \\
&= \int_{\mathbb{R}^n} \varepsilon^{-n}|\varphi(x/\varepsilon)|\, dx \\
&= \int_{\mathbb{R}^n} \varepsilon^{-n}|\varphi(u)|\varepsilon^n\, du \\
&= \int_{\mathbb{R}^n} |\varphi(u)|\, du = \|\varphi\|_1
\end{aligned}$$

□

Remark 4.18. *Observe that for any radius $R > 0$, $\varphi(x) \neq 0$ for $x \in B_R(0)$ outside the ball, indicating that it does not have compact support. This characteristic is important for achieving a C^∞-approximation as stated in Theorem 4.1.*

Nevertheless, we will demonstrate that it is still possible to obtain a C^∞ approximation.

Theorem 4.3 (Approximation of Θ_c by the Gaussian Mollifier)**.** *Let $\varepsilon > 0$ and $c \in (-1, 1)$. The function $(\Theta_c)_\varepsilon : [-1, 1] \to \mathbb{R}$ is defined with:*

$$(\Theta_c)_\varepsilon(x) := \operatorname{erf}\left(\frac{x-c}{\varepsilon\sqrt{2}}\right)$$

and an approximation to Θ_c according to Theorem 4.1 in the sense that $(\Theta_c)_\varepsilon \to (\Theta_c)$ strongly in $L^p(\mathbb{R})$ as $\varepsilon \to 0$.

Proof. We now approximate Θ_c on the interval $[-1,1]$ using the Gaussian mollifier. We again follow Theorem 4.1 and Remark 4.13 on C^∞ approximation. Let $\varphi \in C^\infty(\mathbb{R}^n)$ be the Gaussian mollifier and φ_ε for $\varepsilon > 0$ its scaled version. Let $\Theta_c \in L^p([-1,1])$ be the step function from Lemma 4.1, which is measurable by Remark 4.14. Define $\tilde{\Theta}_c \in L^p(\mathbb{R})$ for $c \in (-1,1)$ such that:

$$\tilde{\Theta}_c(x) = \begin{cases} \Theta_c(x) & \text{if } x \in [-1,1], \\ 0 & \text{if } x \notin [-1,1]. \end{cases}$$

We calculate the convolution of $\tilde{\Theta}_c$ with φ_ε as follows:

$$\begin{aligned}(\Theta_c)_\varepsilon(x) = (\varphi_\varepsilon * \tilde{\Theta}_c)(x) &= \int_{-\infty}^{\infty} \varphi_\varepsilon(x-y)\tilde{\Theta}_c(y)\,dy \\ &= \underbrace{\int_{-\infty}^{c} \varphi_\varepsilon(x-y)(-1)\,dy}_{I_1} + \underbrace{\int_{c}^{\infty} \varphi_\varepsilon(x-y)(1)\,dy}_{I_2}\end{aligned}$$

With the substitution $u = x - y$ and $dy = -du$ we can calculate both integrals:

$$I_1 = -\int_{-\infty}^{c} \frac{1}{\varepsilon\sqrt{2\pi}} e^{-\frac{(x-y)^2}{2\varepsilon^2}}\,dy = -\int_{x-c}^{\infty} \frac{1}{\varepsilon\sqrt{2\pi}} e^{-\frac{u^2}{2\varepsilon^2}}\,du = -I_2$$

Note that these integrals represent the area under a Gaussian curve, which can be expressed in terms of the error function:

$$\begin{aligned}I_1 &= -\int_{x-c}^{\infty} \frac{1}{\varepsilon\sqrt{2\pi}} e^{-\frac{u^2}{2\varepsilon^2}}\,du = -\frac{1}{2}\left(1 - \operatorname{erf}\left(\frac{x-c}{\varepsilon\sqrt{2}}\right)\right) \\ I_2 &= \int_{-\infty}^{x-c} \frac{1}{\varepsilon\sqrt{2\pi}} e^{-\frac{v^2}{2\varepsilon^2}}\,dv = \frac{1}{2}\left(1 + \operatorname{erf}\left(\frac{x-c}{\varepsilon\sqrt{2}}\right)\right)\end{aligned}$$

Adding these two integrals together gives:

$$(\Theta_c)_\varepsilon(x) = -\frac{1}{2}\left(1 - \operatorname{erf}\left(\frac{x-c}{\varepsilon\sqrt{2}}\right)\right) + \frac{1}{2}\left(1 + \operatorname{erf}\left(\frac{x-c}{\varepsilon\sqrt{2}}\right)\right) = \operatorname{erf}\left(\frac{x-c}{\varepsilon\sqrt{2}}\right)$$

□

Remark 4.19. *Although the Gaussian mollifier used in the theorem above does not have compact support, the C^∞ approximation Theorem 4.1 ensures that for a given $c \in (-1,1)$ and $\varepsilon > 0$, the approximation $(\Theta_c)_\varepsilon \to \Theta_c$ converges strongly in $L^p(\mathbb{R})$ as $\varepsilon \to 0$. However, this theorem does not imply that the approximated function $(\Theta_c)_\varepsilon$ is itself C^∞.*

To clarify why $(\Theta_c)_\varepsilon$ is indeed C^∞ when using a Gaussian mollifier, consider the approximation of Θ_c using the Gaussian mollifier, expressed in terms of the error function:

$$(\Theta_c)_\varepsilon(x) = erf\left(\frac{x-c}{\varepsilon\sqrt{2}}\right).$$

The error function erf is both continuous and differentiable, with its first derivative given

by:

$$\frac{\mathrm{d}}{\mathrm{d}z}\, erf(z) = \frac{2}{\sqrt{\pi}} e^{-z^2},$$

which is C^∞ as a well-known combination of C^∞ functions. Therefore, $(\Theta_c)_\varepsilon$ is a C^∞ function due to the smoothness of the error function, providing a smooth C^∞ approximation to the discontinuous function Θ_c.

Therefore, we will now find an approximation of Chebyshev polynomials to the error function $\mathrm{erf}(k\,[x-\delta])$ for $k > 0$ and $\delta \in [-1, 1]$. To construct this approximation polynomial, we utilize special functions. We begin with the following definition adapted from [AW05, Leb72]:

Definition 4.5 (Modified Bessel Functions of the First Kind)**.** *The modified Bessel functions of the first kind, denoted $I_\nu(x)$, are defined for $x \in \mathbb{C}$ by:*

$$I_\nu(x) = \sum_{m=0}^{\infty} \frac{1}{m!\,\Gamma(m+\nu+1)} \left(\frac{x}{2}\right)^{2m+\nu},$$

where $\nu \in \mathbb{R}$. Here, $\Gamma(z)$ represents the Gamma function, which is given by:

$$\Gamma(z) = \int_0^\infty t^{z-1} e^{-t}\, dt, \quad \textit{for } \mathrm{Re}(z) > 0,$$

and is defined for all $z \in \mathbb{C} \setminus \{0, -1, -2, \ldots\}$.

Remark 4.20. *Note that this series converges for all $x \in \mathbb{C}$ and $\nu \in \mathbb{R}$, ensuring that the series characterization of $I_\nu(x)$ is well-defined. For further details, see [Leb72].*

We can adapt the error function approximation from [Low17], which is derived through the Jacobi-Anger expansion of the exponential decay function:

$$e^{-\beta(x+1)} = e^{-\beta} \left(I_0(\beta) + 2 \sum_{j=1}^{\infty} I_j(\beta) T_j(-x) \right),$$

where $\beta > 0$, $x \in [-1, 1]$, $I_j(\beta)$ are the modified Bessel functions of the first kind, and $T_j(x)$ are the Chebyshev polynomials of the first kind. Using this expression, and through a change of variables and integration of the whole series, we obtain the expression for the error function, which gives us the following approximation (Corollary 6.25 and Corollary 2.26 from [Low17]) to $\mathrm{erf}(k\,[x-\delta])$ using Chebyshev polynomials (see Figure 4.1):

Lemma 4.1 (Polynomial approximation to the error function)**.** *Let $\epsilon > 0$. For $k > 0$ and $\delta \in [-1, 1]$, the polynomial $p_{erf,k,\delta,n}$ of odd degree $n = \mathcal{O}(\sqrt{(k^2 + \log(1/\epsilon)) \log(1/\epsilon)})$ is an ϵ-approximation to $erf(k\,[x-\delta])$ for $x \in [-1, 1]$. The polynomial $p_{erf,k,\delta,n}(x)$ is given by*

$$p_{erf,k,\delta,n}(x) := \frac{2ke^{-k^2/2}}{\sqrt{\pi}} \left(I_0(k^2/2)\, x + \sum_{j=1}^{(n-1)/2} I_j(k^2/2)\, (-1)^j \left(\tfrac{T_{2j+1}(x-\delta)}{2j+1} - \tfrac{T_{2j-1}(x-\delta)}{2j-1} \right) \right),$$

where $I_j(x)$ is the j-th modified Bessel function of the first kind, and $T_j(x)$ is the j-th Chebyshev polynomial of the first kind.

Proof. A proof can be found in Lemma 6.22, Corollary 6.25, and Corollary 2.26 of [Low17]. □

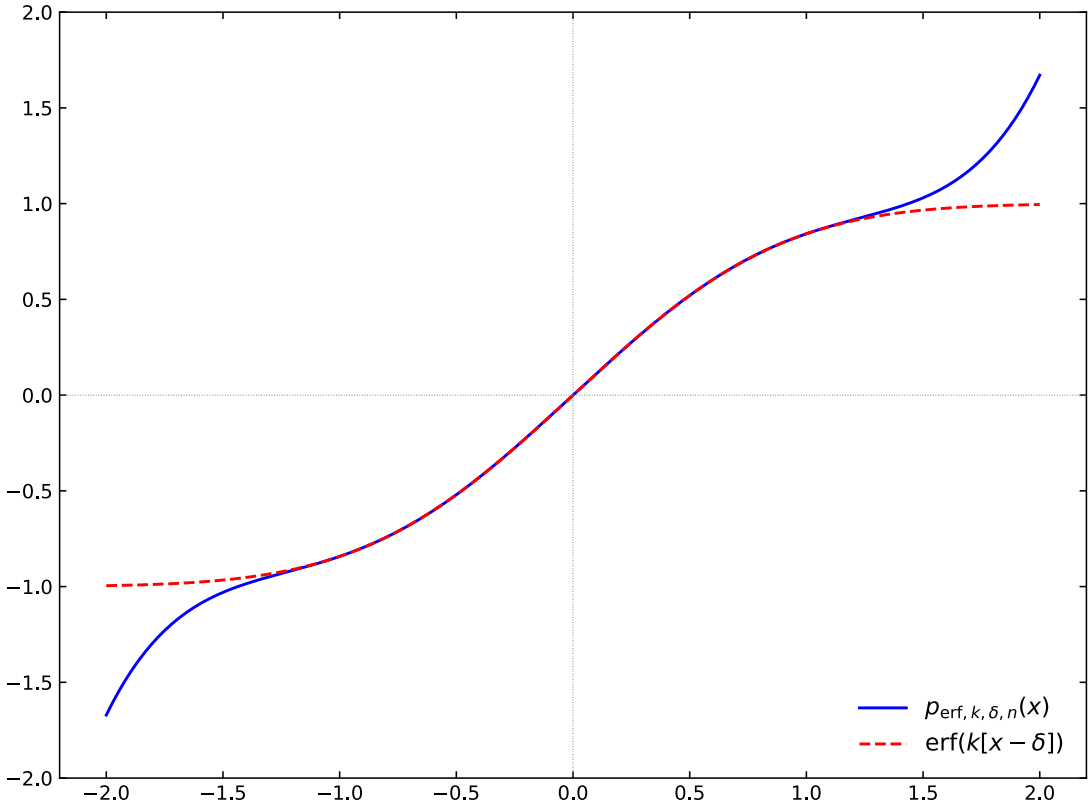

Figure 4.1: Comparison between the polynomial approximation $p_{\mathrm{erf},k,n}(x)$ from Lemma 4.1 and the error function $\mathrm{erf}(k[x-\delta])$ from Definition 4.4. The approximation is computed using parameters derived from Corollary 6.25 in [Low17], with $k = 1$, $\epsilon = 1 \times 10^{-2}$, $\delta = 0$, and $n = \sqrt{k^2 + \log(1/\epsilon)\log(1/\epsilon)}$. The x-axis represents the x values, and the y-axis represents the function values.

The idea now is to use $p_{\text{erf},k,\delta,n}$ and to define a $k > 0$ that gets arbitrarily small with respect to some $\epsilon, \kappa > 0$. By doing so, we will obtain an approximation to the sign function (see Figure 4.2). Note that this sign function is given by Θ_0 from Lemma 4.1.

Definition 4.6 (Sign function)**.** *The function* $\text{sgn} : [-1,1] \to \mathbb{R}$ *is defined as:*

$$\text{sgn}(x) := \Theta_0(x)$$

To define such a $k > 0$ we follow Corollary 6.27 from [Low17]:

Lemma 4.2 (Polynomial approximation to the sign function)**.** *Let* $\epsilon > 0$*. For* $\kappa > 0$ *and* $\delta \in [-1,1]$ *with*

$$k := \frac{\sqrt{2}}{\kappa} \log^{1/2}\left(\frac{2}{\pi\epsilon^2}\right)$$

the polynomial of odd degree $n = \mathcal{O}(1/\kappa \log(1/\epsilon))$*, given by* $p_{sgn,\kappa,\delta,n}(x) := p_{erf,k,\delta,n}(x)$ *is an* ϵ*-approximation to* $sgn(x-\delta)$ *for* $x \in [-1,1]$*.*

Proof. A proof can be found in Lemma 6.19 and Corollary 6.27 of [Low17]. □

Lemma 4.3. $p_{sgn,\kappa,\delta,n}$ *is an odd polynomial.*

Proof. With Lemma 2.3 from Section 2.4.2.1 we know that Chebyshev polynomials have the property $T_j(-x) = (-1)^j T_j(x)$. Thus, for odd indices, $T_{2j+1}(-(x-\delta)) = -T_{2j+1}(x-\delta)$ and $T_{2j-1}(-(x-\delta)) = -T_{2j-1}(x-\delta)$. Since $I_0(k^2/2)\,x$ is linear in x, it is also an odd function. The sum involves only terms of $T_{2j+1}(x-\delta)$ and $T_{2j-1}(x-\delta)$, which are odd, preserving the overall odd parity. Thus, $p_{\text{erf},k,\delta,n}(x)$ is an odd polynomial. Therefore, $p_{\text{sgn},\kappa,\delta,n}(x)$ is also an odd polynomial. □

Let $\kappa \in (0,2]$. Consider the rectangular function $\text{rect}(x/w)$, which is a rectangular function scaled by a factor $w \in [0, 2-\kappa]$:

$$\text{rect}\left(\frac{x}{w}\right) = \begin{cases} 1 & \text{if } |x| \leq \frac{w}{2}, \\ 0 & \text{if } |x| > \frac{w}{2}. \end{cases}$$

We can combine two such sign function approximations from above and obtain a scaled rectangular function (see Figure 4.3), as described in the following Corollary 6.28 from [Low17]:

Lemma 4.4. *Let* $\epsilon > 0$*. For* $\kappa \in (0,2]$ *and* $w \in [0, 2-\kappa]$ *the even polynomial*

$$p_{\text{rect},w,\kappa,n}(x) = \frac{1}{2}\left(p_{\text{sgn},\kappa,(w+\kappa)/2,n+1}(x) + p_{\text{sgn},\kappa,(w+\kappa)/2,n+1}(-x)\right) \tag{4.1}$$

of even degree $n = \mathcal{O}(1/\kappa \log(1/\epsilon))$ *is an* ϵ*-approximation to* $\text{rect}(x/w)$ *for* $x \in [-1,1]$*.*

Proof. A proof can be found in Lemma 6.20 from [Low17]. □

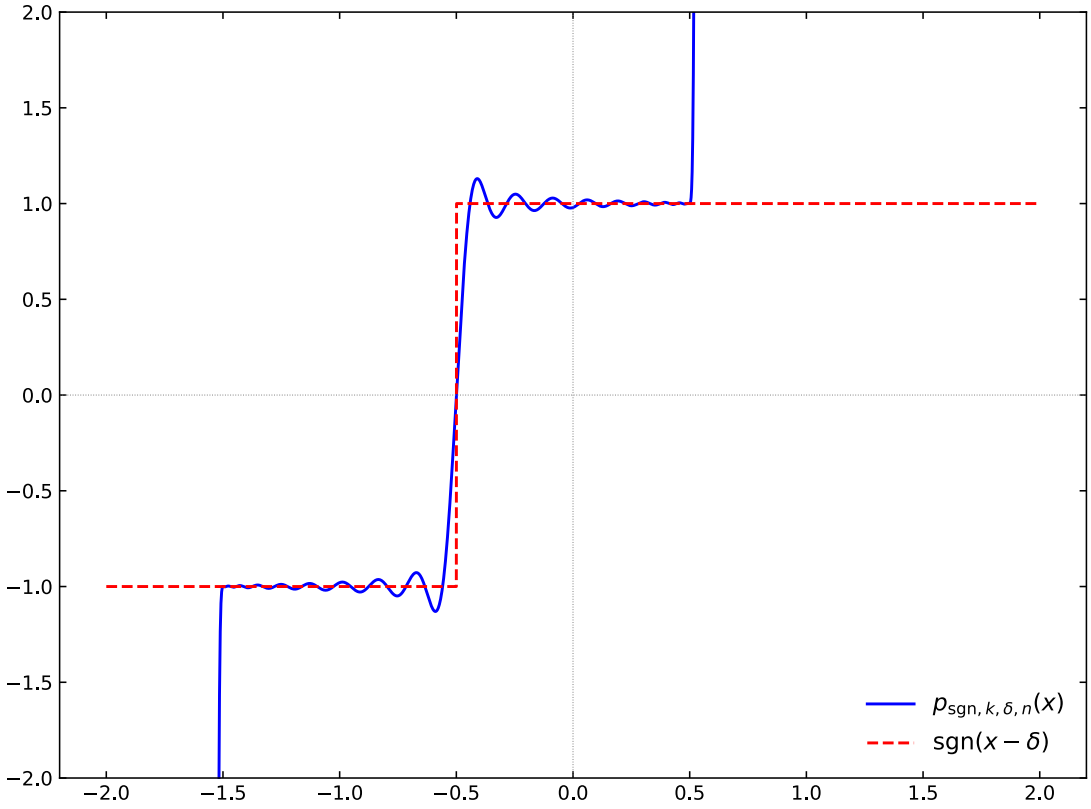

Figure 4.2: Comparison between the polynomial approximation $p_{\mathrm{sgn},k,\delta,n}(x)$ from Lemma 4.2 and the sign function $\mathrm{sgn}(x-\delta)$ from Definition 4.6. The approximation is computed for $\epsilon = 1 \times 10^{-562}$ and using parameters derived from Corollary 6.27 in [Low17], with $\delta = -1/2$, $k = (\sqrt{2}/2)\log^{1/2}(2/(\pi\epsilon^2))$, and $n = \lceil (1/k)\log(1/\epsilon) \rceil$ (odd). The x-axis represents the x values, and the y-axis represents the function values.

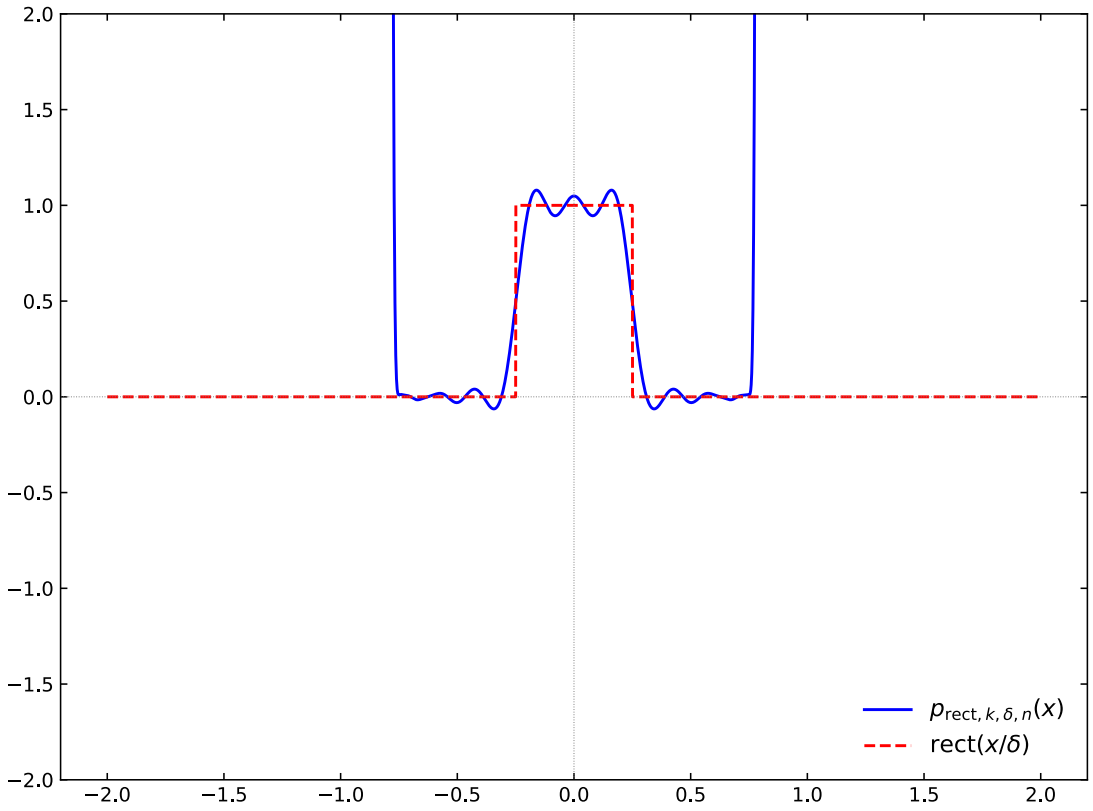

Figure 4.3: Comparison between the polynomial approximation $p_{\mathrm{rect},k,\delta,n}(x)$ from Lemma 4.4 and the rectangular function $\mathrm{rect}(x/\delta)$. The approximation is computed for $\epsilon = 1\times10^{-562}$ and using parameters derived from Corollary 6.28 in [Low17], with $\delta = 1/2$, $k = (\sqrt{2}/2)\log^{1/2}(2/(\pi\epsilon^2))$, and $n = \lceil(1/k)\log(1/\epsilon)\rceil$ (odd). The x-axis represents the x values, and the y-axis represents the function values. Note that any observed discrepancies, particularly around the discontinuities, might stem from the numerical implementation and the challenges associated with approximating functions with high precision for extremely small ϵ values.

4.2.2.2 Construction of the Inversion Polynomial

Building on the methodologies proposed by [GSLW18] and [MRTC21], we aim to construct a polynomial that closely approximates $\frac{1}{x}$. Our specific goal is to achieve an $\frac{\epsilon}{2\kappa}$-approximation of the function $f(x) := \frac{1}{2\kappa}\frac{1}{x}$, where κ denotes the condition number of the matrix B'. [GSLW18] developed a polynomial approximation to achieve this. For clarity in our notation, we assume $\kappa > 0$, as that will be the condition number later on and the condition number (see Definition 2.10 from Section 2.2)is always positive.

Definition 4.7. *Let $b_{\epsilon,\kappa} \in \mathbb{N}$. For $x \in [-1,1] \setminus \left[\frac{-1}{\kappa}, \frac{1}{\kappa}\right]$ we define:*

$$g_{\epsilon,\kappa}(x) := \frac{1-(1-x^2)^{b_{\epsilon,\kappa}}}{x}$$

This function approximates $\frac{1}{x}$ effectively over the range $x \in [-1,1]\setminus\left[\frac{-1}{\kappa}, \frac{1}{\kappa}\right]$ for sufficiently large $b_{\epsilon,\kappa} \in \mathbb{N}$ as seen in the following:

Lemma 4.5. *For $0 < \epsilon < \frac{1}{2}$ the function $g_{\epsilon,\kappa}(x)$ is an ϵ-approximation for $\frac{1}{x}$ on the interval $x \in [-1,1] \setminus \left[\frac{-1}{\kappa}, \frac{1}{\kappa}\right]$, where $b_{\epsilon,\kappa} = \mathcal{O}\left(\kappa^2 \log\left(\frac{\kappa}{\epsilon}\right)\right)$.*

Proof. Let $0 < \epsilon < \frac{1}{2}$. For $x \in [-1,1] \setminus \left[\frac{-1}{\kappa}, \frac{1}{\kappa}\right]$, we analyze the behavior of $g_{\epsilon,\kappa}(x)$ as x approaches the boundaries of the excluded interval, i.e., as $x \to \frac{1}{\kappa}^-$ or $x \to -\frac{1}{\kappa}^+$. We observe that

$$\lim_{x\to\frac{1}{\kappa}^-}(1-x^2)^{b_{\epsilon,\kappa}} = 1 \quad \text{and} \quad \lim_{x\to-\frac{1}{\kappa}^+}(1-x^2)^{b_{\epsilon,\kappa}} = 1.$$

For larger values of $b_{\epsilon,\kappa}$, the expression $(1-x^2)^{b_{\epsilon,\kappa}}$ remains close to 1 except near the endpoints $x = \pm 1$. Thus, as $b_{\epsilon,\kappa} \to \infty$, the function $g_{\epsilon,\kappa}(x)$ tends towards

$$\lim_{b_{\epsilon,\kappa}\to\infty} g_{\epsilon,\kappa}(x) = \frac{1}{x}.$$

We next consider the Taylor expansion of $(1-x^2)^{b_{\epsilon,\kappa}}$ around x near the boundaries $\frac{1}{\kappa}^-$ or $-\frac{1}{\kappa}^+$:

$$(1-x^2)^{b_{\epsilon,\kappa}} = 1 - b_{\epsilon,\kappa}x^2 + \frac{b_{\epsilon,\kappa}(b_{\epsilon,\kappa}-1)}{2}x^4 + O(x^6).$$

Therefore, for large $b_{\epsilon,\kappa}$, we have:

$$g_{\epsilon,\kappa}(x) \approx \frac{1-(1-b_{\epsilon,\kappa}x^2)}{x} = \frac{b_{\epsilon,\kappa}x^2}{x} = b_{\epsilon,\kappa}x.$$

We need to choose $b_{\epsilon,\kappa}$ such that $\left|g_{\epsilon,\kappa}(x) - \frac{1}{x}\right| < \epsilon$, and since it is $(1-x^2)^{b_{\epsilon,\kappa}} \leq e^{-b_{\epsilon,\kappa}x^2}$ for $x \in [-1,1] \setminus \left[\frac{-1}{\kappa}, \frac{1}{\kappa}\right]$, we have:

$$\left|\frac{1-(1-x^2)^{b_{\epsilon,\kappa}}}{x} - \frac{1}{x}\right| < \epsilon \iff \left|\frac{(1-x^2)^{b_{\epsilon,\kappa}}}{x}\right| < \epsilon \iff \left|\frac{e^{-b_{\epsilon,\kappa}x^2}}{x}\right| < \epsilon.$$

Hence, it must be $e^{-b_{\epsilon,\kappa}x^2} < \epsilon|x|$. Consider the worst-case scenario where $x \to \frac{1}{\kappa}^-$ approaches from the left. Note that by the symmetry of the problem, the other limit as $x \to -\frac{1}{\kappa}^+$ from the right can be computed similarly.

$$\lim_{x\to\frac{1}{\kappa}^-}\left(-b_{\epsilon,\kappa}x^2\right) < \lim_{x\to\frac{1}{\kappa}^-}\log(\epsilon x). \tag{4.2}$$

Note that for the limit on the right-hand side of the above Equation 4.2, we have:

$$\lim_{x\to\frac{1}{\kappa}^-}\log(\epsilon x) = \log\left(\epsilon\cdot\frac{1}{\kappa}\right) = \log(\epsilon) - \log(\kappa).$$

Considering the left-hand side of Equation 4.2, as $x \to \frac{1}{\kappa}^-$:

$$\lim_{x\to\frac{1}{\kappa}^-}\left(-b_{\epsilon,\kappa}x^2\right) = -b_{\epsilon,\kappa}\left(\frac{1}{\kappa}\right)^2 = -\frac{b_{\epsilon,\kappa}}{\kappa^2}.$$

Therefore, we are left with the following inequality:

$$-\frac{b_{\epsilon,\kappa}}{\kappa^2} < \log(\epsilon) - \log(\kappa) \iff b_{\epsilon,\kappa} > \kappa^2\left(\log(\kappa) - \log(\epsilon)\right) \iff b_{\epsilon,\kappa} > \kappa^2\log\left(\frac{\kappa}{\epsilon}\right).$$

Thus, choosing $b_{\epsilon,\kappa} = \lceil\kappa^2\log\left(\frac{\kappa}{\epsilon}\right)\rceil$ ensures that $g_{\epsilon,\kappa}(x)$ ϵ-approximates $\frac{1}{x}$ for $x \in [-1,1] \setminus \left[\frac{-1}{\kappa},\frac{1}{\kappa}\right]$. Consequently, $g_{\epsilon,\kappa}(x)$ is an ϵ-approximation to $\frac{1}{x}$ in the specified range, as required. □

Although $g_{\epsilon,\kappa}(x)$ is not a polynomial yet, we can find an ϵ-approximation to it with the following polynomial proposed by [MRTC21], which we call the inversion polynomial (see 4.4):

Definition 4.8 (Inversion Polynomial)**.** *Let $0 < \epsilon < \frac{1}{2}$. For $x \in [-1,1] \setminus \left[\frac{-1}{\kappa},\frac{1}{\kappa}\right]$, we define*

$$P^{1/x}_{2\epsilon,\kappa}(x) := 4\sum_{j=0}^{D_{\epsilon,\kappa}}(-1)^j\left[2^{-2b_{\epsilon,\kappa}}\sum_{i=j+1}^{b_{\epsilon,\kappa}}\binom{2b_{\epsilon,\kappa}}{b_{\epsilon,\kappa}+i}\right]T_{2j+1}(x),$$

where $T_i(x)$ is the Chebyshev polynomial of the first kind of order i, and

$$b_{\epsilon,\kappa} := \mathcal{O}\left(\kappa^2\log\left(\frac{\kappa}{\epsilon}\right)\right),$$
$$D_{\epsilon,\kappa} := \left\lceil\sqrt{b_{\epsilon,\kappa}\log\left(\frac{4b_{\epsilon,\kappa}}{\epsilon}\right)}\right\rceil = \mathcal{O}(\kappa\log(\kappa/\epsilon)).$$

Lemma 4.6. *$P^{1/x}_{2\epsilon,\kappa}(x)$ is an ϵ-approximation to $g_{\epsilon,\kappa}(x)$ for $x \in [-1,1] \setminus \left[\frac{-1}{\kappa},\frac{1}{\kappa}\right]$.*

Proof. For a complete proof see [MRTC21, Low17]. □

Lemma 4.7. *$P^{1/x}_{2\epsilon,\kappa}(x)$ is a 2ϵ-approximation to $\frac{1}{x}$ for $x \in [-1,1] \setminus \left[\frac{-1}{\kappa},\frac{1}{\kappa}\right]$.*

Proof. Since with Lemma 4.5, $P^{1/x}_{2\epsilon,\kappa}(x)$ is a polynomial that ϵ-approximates $g_{\epsilon,\kappa}(x)$ over the interval $x \in [-1,1] \setminus \left[\frac{-1}{\kappa}, \frac{1}{\kappa}\right]$, we get with the triangle inequality:

$$\left|P^{1/x}_{2\epsilon,\kappa}(x) - \frac{1}{x}\right| \leq \left|P^{1/x}_{2\epsilon,\kappa}(x) - g_{\epsilon,\kappa}(x)\right| + \left|g_{\epsilon,\kappa}(x) - \frac{1}{x}\right| < \epsilon + \epsilon = 2\epsilon$$

□

Lemma 4.8. *$\frac{1}{2\kappa}P^{1/x}_{\frac{\epsilon}{2},2\kappa}(x)$ is an ϵ-approximation for $\frac{1}{2\kappa}\frac{1}{x}$ on $x \in [-1,1] \setminus \left[\frac{-1}{2\kappa}, \frac{1}{2\kappa}\right]$.*

Proof. Let $x \in [-1,1] \setminus \left[\frac{-1}{\kappa}, \frac{1}{\kappa}\right]$. From Corollary 4.7, we know that $P^{1/x}_{2\epsilon,\kappa}(x)$ is a 2ϵ-approximation to $\frac{1}{x}$. Therefore:

$$\left|\frac{1}{2\kappa}P^{1/x}_{2\epsilon,\kappa}(x) - \frac{1}{2\kappa}\frac{1}{x}\right| = \frac{1}{2\kappa}\left|P^{1/x}_{2\epsilon,\kappa}(x) - \frac{1}{x}\right| < \frac{1}{2\kappa}\cdot 2\epsilon = \frac{\epsilon}{\kappa}.$$

By substituting ϵ with $\frac{\epsilon}{2}$ in the polynomial approximation, we get that $P^{1/x}_{\frac{\epsilon}{2},2\kappa}(x)$ is an $\frac{\epsilon}{2}-$ approximation to $\frac{1}{x}$. Thus,

$$\left|\frac{1}{2\kappa}P^{1/x}_{\frac{\epsilon}{2},2\kappa}(x) - \frac{1}{2\kappa}\frac{1}{x}\right| = \frac{1}{2\kappa}\left|P^{1/x}_{\frac{\epsilon}{2},2\kappa}(x) - \frac{1}{x}\right| < \frac{1}{2\kappa}\cdot\frac{\epsilon}{2} = \frac{\epsilon}{4\kappa}.$$

Since the function $\frac{1}{2\kappa}\frac{1}{x}$ is scaled by $\frac{1}{2\kappa}$, this gives us an ϵ-approximation.

□

The following lemma is essential as it establishes the requirement $\kappa \geq 1$ for the polynomial to be bounded, which is necessary for applying the QSVT Theorem 2.17 from Section 2.4.3.

Lemma 4.9. *Let $0 < \epsilon < \frac{1}{2}$ and $\kappa \geq 1$. Then $\left|\frac{1}{2\kappa}P^{1/x}_{\frac{\epsilon}{2},2\kappa}(x)\right| \leq 1$ for $x \in [-1,1] \setminus \left[\frac{-1}{\kappa}, \frac{1}{\kappa}\right]$.*

Proof. Let $0 < \epsilon < \frac{1}{2}$. Since $\epsilon < \frac{1}{2}$, we can bound the polynomial for $x \in [-1,1] \setminus \left[\frac{-1}{\kappa}, \frac{1}{\kappa}\right]$. Using the triangle inequality and the bounds on the approximation error, we have:

$$\left|\frac{1}{2\kappa}P^{1/x}_{2\epsilon,2\kappa}(x)\right| \leq \left|\frac{1}{2\kappa}\frac{1}{x}\right| + \epsilon < \left|\frac{1}{2\kappa}\frac{1}{x}\right| + \frac{1}{2} \leq \left|\frac{1}{2\kappa}\right| + \frac{1}{2} \leq 1$$

□

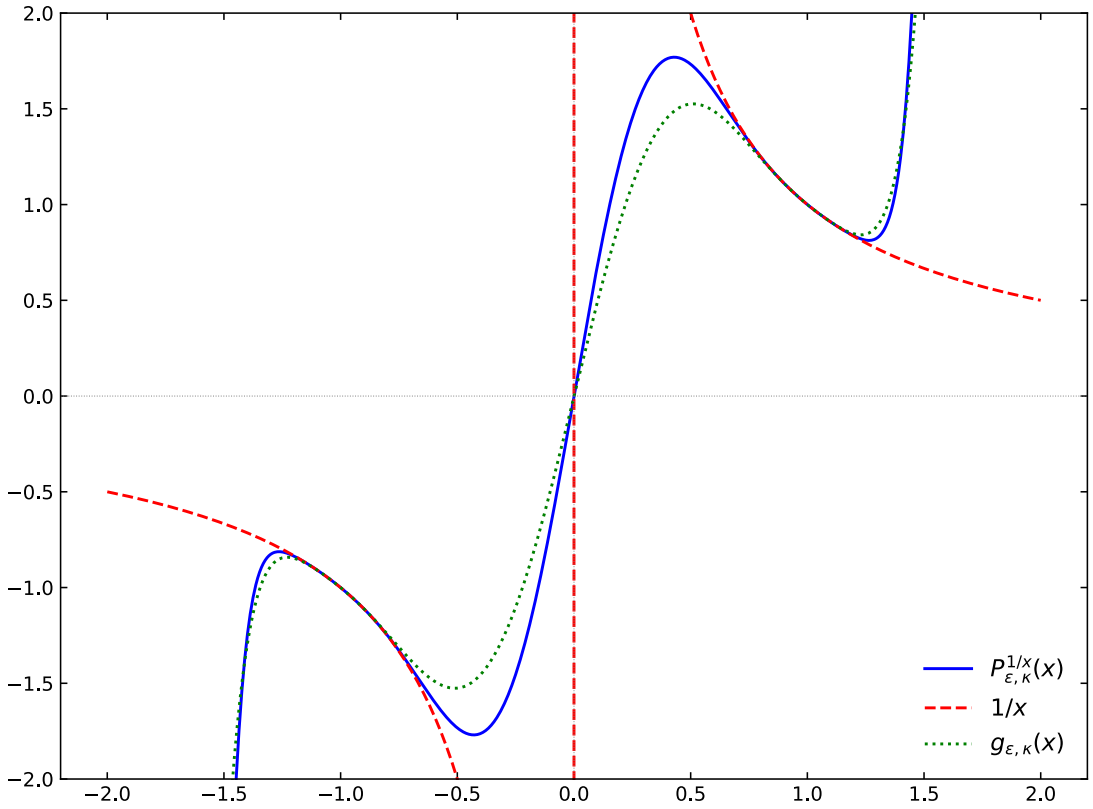

Figure 4.4: Comparison between the polynomial approximation $P^{1/x}_{\varepsilon,\kappa}(x)$ from Definition 4.8 and the function $1/x$. The approximation is computed according to Corollary 4.7 using parameters $\varepsilon = 1 \times 10^{-3}$, $\kappa = 1$, $b = \lceil \kappa^2 \log(\frac{\kappa}{\varepsilon}) \rceil$, and $D = \lceil \sqrt{b \log(4b/\varepsilon)} \rceil$. As a reference, there is the function $g_{\varepsilon,\kappa}$ from Lemma 4.5 with $\varepsilon = 1 \times 10^{-2}$, $\kappa = 1$, and $b = \lceil \kappa^2 \log(\frac{\kappa}{\varepsilon}) \rceil$. The x-axis represents the x values, and the y-axis represents the function values.

4.2.2.3 Construction of the Matrix Inversion Polynomial

Again let $\kappa > 0$ for the following. However, the inversion polynomial may not be bounded for $x \in \left[\frac{-1}{2\kappa}, \frac{1}{2\kappa}\right]$. To enforce this bound, we multiply the polynomial by an even function that remains close to 1 for $x \in [-1,1] \setminus \left[\frac{-1}{\kappa}, \frac{1}{\kappa}\right]$ and approaches 0 for $x \in \left[\frac{-1}{2\kappa}, \frac{1}{2\kappa}\right]$. This is achieved using a polynomial approximation of a rectangular function as proposed by [MRTC21]. To make the construction of the step function approximation polynomial more comprehensible, we used the notation of [Low17] in Section 4.2.2.1 for defining a polynomial approximation p_{rect} and p_{sgn} (see Lemma 4.11 / Lemma 4.2). We need to modify those slightly to make them compliant with the notation of [MRTC21]. We need a slightly more specialized rectangular approximation than the basic one given in Lemma 4.4. We will adapt Lemma 4.2 for this and use $P^{\Theta}_{\epsilon,\frac{1}{4\kappa}} := p_{\text{sgn},\frac{1}{4\kappa},0,n}$ as our sign approximation component to construct the following.

Definition 4.9. *For $\epsilon \in (0, \sqrt{2/e\pi}]$, $\kappa \geq 1$ and $x \in [-1,1] \setminus \left[\frac{-1}{\kappa}, \frac{1}{\kappa}\right]$, we define:*

$$P^{rect}_{\epsilon,\kappa}(x) := \frac{1}{1+\frac{\epsilon}{2}}\left(1 + \frac{1}{2}\left(P^{\Theta}_{\epsilon,\frac{1}{4\kappa}}\left(x - \tfrac{3}{4\kappa}\right) + P^{\Theta}_{\epsilon,\frac{1}{4\kappa}}\left(-x - \tfrac{3}{4\kappa}\right)\right)\right),$$

where $P^{\Theta}_{\epsilon,\frac{1}{4\kappa}} := p_{sgn,\frac{1}{4\kappa},0,n}$ is the polynomial approximation from Lemma 4.2.

Lemma 4.10. *For the polynomial $P^{\Theta}_{\epsilon,\frac{1}{4\kappa}}$ and $x \in [-1,1] \setminus \left[\frac{-1}{\kappa}, \frac{1}{\kappa}\right]$ the following holds:*

- *$P^{\Theta}_{\epsilon,\frac{1}{4\kappa}}$ is odd.*
- *$P^{\Theta}_{\epsilon,\frac{1}{4\kappa}}(x)$ is an ϵ-approximation to $\Theta_0(x)$.*
- *$|P^{\Theta}_{\epsilon,\frac{1}{4\kappa}}(x)| \leq 1$.*

Proof. $P^{\Theta}_{\epsilon,\frac{1}{4\kappa}}$ being odd follows from Lemma 4.3. The ϵ-approximation follows from Lemma 4.2. The boundedness follows from Lemma 2.2 from Section 2.4.2.1 and since $x \in [-1,1]$, all other terms are bounded by 1 as well. □

Lemma 4.11. *Let $0 < \epsilon < \frac{1}{2}$. For $P^{rect}_{\epsilon,\kappa}(x)$, the following holds:*

1. *$P^{rect}_{\epsilon,\kappa}(x) \in [1-\epsilon, 1]$ for $x \in [-1,1] \setminus \left[\frac{-1}{\kappa}, \frac{1}{\kappa}\right]$.*
2. *$P^{rect}_{\epsilon,\kappa}(x) \in [0, \epsilon]$ for $x \in \left[\frac{-1}{2\kappa}, \frac{1}{2\kappa}\right]$.*
3. *$P^{rect}_{\epsilon,\kappa}(x)$ has an even degree $\mathcal{O}(\kappa \log(1/\epsilon))$ for $x \in [-1,1]$.*

Proof.

1. Let $x \in [-1,1] \setminus \left[\frac{-1}{\kappa}, \frac{1}{\kappa}\right]$. Both $x - \frac{3}{4\kappa}$ and $-x - \frac{3}{4\kappa}$ are outside the interval $\left[-\frac{1}{4\kappa}, \frac{1}{4\kappa}\right]$. By definition of $P^{\Theta}_{\epsilon,\frac{1}{4\kappa}}(x - \frac{3}{4\kappa}) \approx 1$ and $P^{\Theta}_{\epsilon,\frac{1}{4\kappa}}(-x - \frac{3}{4\kappa}) \approx 1$. Therefore, it is:

$$P^{\text{rect}}_{\epsilon,\kappa}(x) \approx \frac{1}{1+\frac{\epsilon}{2}}\left(1 + \frac{1}{2}(1+1)\right) = \frac{1}{1+\frac{\epsilon}{2}} \cdot 2 = \frac{2}{1+\frac{\epsilon}{2}} \in [1-\epsilon, 1].$$

2. Let $x \in \left[\frac{-1}{2\kappa}, \frac{1}{2\kappa}\right]$. Both $x - \frac{3}{4\kappa}$ and $-x - \frac{3}{4\kappa}$ are close to $\pm\frac{3}{4\kappa}$ respectively, which is far from $\pm\frac{1}{4\kappa}$. Hence, by Definition 4.9, we have $P^{\Theta}_{\epsilon,\frac{1}{4\kappa}}(x - \frac{3}{4\kappa}) \approx 0$ and $P^{\Theta}_{\epsilon,\frac{1}{4\kappa}}(-x - \frac{3}{4\kappa}) \approx 0$. Therefore:

$$P^{\text{rect}}_{\epsilon,\kappa}(x) \approx \frac{1}{1+\frac{\epsilon}{2}}\left(1+\frac{1}{2}(0+0)\right) = \frac{1}{1+\frac{\epsilon}{2}}.$$

To establish that this expression lies within the interval $[1-\epsilon, 1]$, consider that the function

$$f(\epsilon) := \frac{1}{1+\frac{\epsilon}{2}}$$

is strictly decreasing because its derivative is

$$f'(\epsilon) = -\frac{1}{2(1+\frac{\epsilon}{2})^2} < 0 \quad \text{for all} \quad \epsilon > 0.$$

For any $0 < \epsilon < \frac{1}{2}$, we can expand $f(\epsilon)$ as:

$$\frac{1}{1+\frac{\epsilon}{2}} = 1 - \frac{\epsilon}{2} + \mathcal{O}(\epsilon^2).$$

For small ϵ, the higher-order terms $\mathcal{O}(\epsilon^2)$ are negligible, so we approximate:

$$\frac{1}{1+\frac{\epsilon}{2}} \approx 1 - \frac{\epsilon}{2}.$$

Since $1-\epsilon \leq 1-\frac{\epsilon}{2}$ for $0 < \epsilon < \frac{1}{2}$, it follows that:

$$1-\epsilon \leq \frac{1}{1+\frac{\epsilon}{2}} \leq 1,$$

confirming that $P^{\text{rect}}_{\epsilon,\kappa}(x)$ indeed lies within $[1-\epsilon, 1]$.

3. The polynomials $P^{\Theta}_{\epsilon,\frac{1}{4\kappa}}(x)$ used in $P^{\text{rect}}_{\epsilon,\kappa}(x)$ are constructed to have an even degree $\mathcal{O}(\kappa \log(1/\epsilon))$. Since $P^{\text{rect}}_{\epsilon,\kappa}(x)$ is a combination of such polynomials shifted by constants, its degree remains even and in the same order.

□

We can construct the final polynomial for matrix inversion as follows:

Definition 4.10. *Let $0 < \epsilon < \frac{1}{2}$. For a matrix $A \in \mathbb{R}^{n\times n}$ invertible with the condition number $\kappa = \frac{\sigma_{\max}}{\sigma_{\min}} \geq 1$, where $\sigma_{\max}$ and $\sigma_{\min}$ are the largest and smallest non-zero singular values of A, the matrix inversion polynomial is defined for $x \in [-1,1] \setminus \left[\frac{-1}{\kappa}, \frac{1}{\kappa}\right]$ as:*

$$P^{MI}_{\epsilon,\kappa}(x) := \frac{1}{2\kappa} P^{1/x}_{\frac{\epsilon}{2},2\kappa}(x) P^{rect}_{\epsilon',\kappa}(x),$$

where $\epsilon' = \min\left(\frac{2\epsilon}{5\kappa}, \frac{\kappa}{2\cdot D_{\frac{\epsilon}{4},2\kappa}}\right) = \mathcal{O}\left(\frac{\epsilon}{\kappa}\right)$.

Note that $D_{\frac{\epsilon}{4},2\kappa}$ is the degree of the polynomial from $P^{1/x}$ (see Definition 4.8). Also, note that the term $\frac{2\epsilon}{5\kappa}$ in the above definition ensures an $\frac{\epsilon}{2\kappa}$-approximation to $\frac{1}{2\kappa}\frac{1}{x}$ over the range of singular values, while the other term ensures that the polynomial is bounded for $x \in \left[\frac{-1}{\kappa}, \frac{1}{\kappa}\right]$.

Lemma 4.12. *$P^{MI}_{\epsilon,\kappa}(x)$ has degree $d = \mathcal{O}(\kappa \log(\kappa/\epsilon))$.*

Proof. The degree of $P^{\mathrm{MI}}_{\epsilon,\kappa}(x)$ is determined by the degrees of its parts $P^{1/x}_{\frac{\epsilon}{2},2\kappa}(x)$ and $P^{\mathrm{rect}}_{\epsilon',\kappa}(x)$. The degree of $P^{1/x}_{\frac{\epsilon}{2},2\kappa}(x)$ is by construction, given by $D_{\epsilon,\kappa} = \mathcal{O}(\kappa \log(\kappa/\epsilon))$. With Lemma 4.11, the degree of $P^{\mathrm{rect}}_{\epsilon',\kappa}(x)$ is $\mathcal{O}(\kappa \log(1/\epsilon'))$. Since $\epsilon' = \mathcal{O}\left(\frac{\epsilon}{\kappa}\right)$, we have:

$$\log(1/\epsilon') = \log(\kappa/\epsilon) = \log(\kappa) + \log(1/\epsilon).$$

Thus, the combined degree of $P^{\mathrm{MI}}_{\epsilon,\kappa}(x)$ is calculated as the sum of its components:

$$d = \mathcal{O}(\kappa \log(\kappa/\epsilon) + \kappa \log(\kappa/\epsilon)) = \mathcal{O}(\kappa \log(\kappa/\epsilon)).$$

□

Theorem 4.4. *Let $0 < \epsilon < \frac{1}{2}$. The matrix inversion polynomial $P^{MI}_{\epsilon,\kappa}(x)$ satisfies all requirements for the application of the QSVT Theorem 2.17:*

1. *$P^{MI}_{\epsilon,\kappa}$ is odd.*
2. *$|P^{MI}_{\epsilon,\kappa}(x)| \leq 1$ for all $x \in [-1,1]$.*
3. *$P^{MI}_{\epsilon,\kappa}(x)$ is a $\frac{\epsilon}{2\kappa}$-approximation to $\frac{1}{2\kappa}\frac{1}{x}$ for $x \in [-1,1] \setminus \left[\frac{-1}{\kappa}, \frac{1}{\kappa}\right]$.*

Proof. We prove all the points separately:

1. Consider the components of $P^{\mathrm{MI}}_{\epsilon,\kappa}(x)$ separately. $P^{1/x}_{2\epsilon,\kappa}(x)$ is constructed from Chebyshev polynomials of odd order ($T_{2j+1}(x)$), which ensures it has odd parity. $P^{\mathrm{rect}}_{\epsilon,\kappa}(x)$ could potentially be constructed to be even. Thus, the product of an odd function ($P^{1/x}_{2\epsilon,\kappa}(x)$) and an even function ($P^{\mathrm{rect}}_{\epsilon,\kappa}(x)$) results in an odd function. Therefore, $P^{\mathrm{MI}}_{\epsilon,\kappa}(x)$ has odd parity.
2. We need to show that $|P^{\mathrm{MI}}_{\epsilon,\kappa}(x)| \leq 1$ for $x \in [-1,1]$. From Lemma 4.11, $P^{\mathrm{rect}}_{\epsilon,\kappa}(x)$ is bounded by 1 for $x \in [-1,1]$. For $x \in \left[\frac{-1}{2\kappa}, \frac{1}{2\kappa}\right]$, with Lemma 4.11, the polynomial $P^{\mathrm{rect}}_{\epsilon,\kappa}(x)$ ensures that $P^{\mathrm{MI}}_{\epsilon,\kappa}(x)$ is bounded by ϵ, thus $\left|\frac{1}{2\kappa} P^{1/x}_{\frac{1}{2}\epsilon,2\kappa}(x) P^{\mathrm{rect}}_{\epsilon',\kappa}(x)\right| \leq 1$. For $x \in [-1,1] \setminus \left[\frac{-1}{\kappa}, \frac{1}{\kappa}\right]$, $P^{1/x}_{\frac{1}{2}\epsilon,2\kappa}(x)$ $\frac{\epsilon}{2\kappa}$-approximates $\frac{1}{x}$, as shown in Lemma 4.8.

3. We need to show that $P^{\mathrm{MI}}_{\epsilon,\kappa}(x)$ is a $\frac{\epsilon}{2\kappa}$-approximation to $\frac{1}{2\kappa}\frac{1}{x}$ for $x \in [-1,1] \setminus \left[\frac{-1}{\kappa}, \frac{1}{\kappa}\right]$. From Corollary 4.7, $P^{1/x}_{\frac{1}{2}\epsilon,2\kappa}(x)$ is a $\frac{\epsilon}{2}$-approximation to $\frac{1}{x}$ for $x \in [-1,1] \setminus \left[\frac{-1}{\kappa}, \frac{1}{\kappa}\right]$. $P^{\mathrm{rect}}_{\epsilon,\kappa}(x)$ is in the desired range $\left[\frac{-1}{\kappa}, \frac{1}{\kappa}\right]$ as shown in Lemma 4.11.

□

Remark 4.21. *Note that points 1 and 3 in the above theorem are not explicit requirements for QSVT, as stated in Remark 2.32. However, point 1 allows us to choose the correct circuit for the QSVT operator since it distinguishes between odd and even (see Theorem 2.17 from Section 2.4.3). Point 3 ensures that the use of the polynomial is justified.*

4.2.3 The Absolute Value Approximation Polynomial

In this subsection, we construct the QSVT absolute value approximation polynomial, which is crucial for implementing quantum algorithms that utilize Quantum Singular Value Transformation (QSVT). This polynomial provides an efficient and accurate approximation of the absolute value function $|\cdot| : [-1, 1] \to \mathbb{R}$, meeting the requirements of the QSVT Theorem 2.17.

Definition 4.11 (Fourier series)**.** *Let $f \in L^1([0, 2\pi])$ be a 2π-periodic function. The Fourier series of f, is given by*

$$(\mathcal{F}f)(x) = \frac{a_0}{2} + \sum_{k=1}^{\infty}(a_k \cos kx + b_k \sin kx),$$

where the Fourier coefficients a_k and b_k for $k \in \mathbb{N}$ are defined as

$$a_k = \frac{1}{\pi}\int_0^{2\pi} f(x)\cos kx\, dx \quad \textit{and} \quad b_k = \frac{1}{\pi}\int_0^{2\pi} f(x)\sin kx\, dx.$$

Theorem 4.5. *The Fourier series of $f(x) := |\cos(x)|$ for $x \in \mathbb{R}$ is:*

$$(\mathcal{F}f)(x) = \frac{2}{\pi} + \frac{4}{\pi}\sum_{k=1}^{\infty}\frac{(-1)^{k+1}}{4k^2 - 1}\cos(2kx)$$

Proof. Consider $f(x) := |\cos(x)|$, which is $f \in L^1([0, 2\pi])$ and a 2π-periodic function. The Fourier coefficients a_k and b_k for $k \in \mathbb{N}$ can be determined according to the definition. We first calculate the a_k coefficients:

$$\begin{aligned} a_k &= \frac{1}{\pi}\int_0^{2\pi} f(x)\cos(kx)\, dx \\ &= \frac{1}{\pi}\int_0^{2\pi} |\cos(x)|\cos(kx)\, dx \\ &= \frac{2}{\pi}\Big(\underbrace{\int_0^{\frac{\pi}{2}} \cos(x)\cos(kx)\, dx}_{I_1} - \underbrace{\int_{\frac{\pi}{2}}^{\pi} \cos(x)\cos(kx)\, dx}_{I_2}\Big) \end{aligned}$$

We calculate I_1 using the trigonometric identity:

$$\cos(x)\cos(kx) = \frac{1}{2}\left(\cos((k+1)x) + \cos((k-1)x)\right)$$

Therefore,

$$I_1 = \frac{1}{2}\Big(\underbrace{\int_0^{\frac{\pi}{2}} \cos((k+1)x)\, dx}_{I_{11}} + \underbrace{\int_0^{\frac{\pi}{2}} \cos((k-1)x)\, dx}_{I_{12}}\Big)$$

Substitution with $u := (k+1)x$ leaves us with:

$$I_{11} = \int_0^{\frac{\pi}{2}} \cos(u)\frac{du}{k+1} = \frac{1}{k+1}\int_0^{\frac{\pi}{2}} \cos(u)\, du = \frac{1}{k+1}\sin\left((k+1)\frac{\pi}{2}\right).$$

Substitution with $v := (k-1)x$ gives us:

$$I_{12} = \int_0^{\frac{\pi}{2}} \cos(v) \frac{dv}{k-1} = \frac{1}{k-1} \int_0^{\frac{\pi}{2}} \cos(v)\, dv = \frac{1}{k-1} \sin\left((k-1)\frac{\pi}{2}\right).$$

Therefore,

$$I_1 = \frac{1}{2}(I_{11} + I_{12}) = \frac{1}{2}\left(\frac{1}{k+1}\sin\left((k+1)\frac{\pi}{2}\right) + \frac{1}{k-1}\sin\left((k-1)\frac{\pi}{2}\right)\right).$$

Now we calculate I_2 which is the same integral as I_1 evaluated at different values:

$$\begin{aligned} I_2 &= \frac{1}{2}\left(\int_{\frac{\pi}{2}}^{\pi} \cos(kx+x)\, dx + \int_{\frac{\pi}{2}}^{\pi} \cos(kx-x)\, dx\right) \\ &= \frac{1}{2}\left(\tfrac{1}{k+1}\left(\sin((k+1)\pi) - \sin\left((k+1)\tfrac{\pi}{2}\right)\right) + \tfrac{1}{k-1}\left(\sin((k-1)\pi) - \sin\left((k-1)\tfrac{\pi}{2}\right)\right)\right) \end{aligned}$$

Therefore we have:

$$\begin{aligned} a_k &= \frac{2}{\pi}(I_1 - I_2) \\ &= \frac{1}{\pi}\left(\frac{2\sin\left((k+1)\frac{\pi}{2}\right)}{k+1} + \frac{2\sin\left((k-1)\frac{\pi}{2}\right)}{k-1}\right) \\ &= \frac{2}{\pi}\left(\frac{\sin\left((k+1)\frac{\pi}{2}\right)}{k+1} + \frac{\sin\left((k-1)\frac{\pi}{2}\right)}{k-1}\right) \end{aligned}$$

Note that for $k = 2m+1$ with $m \in \mathbb{N}$, it is

$$\frac{\sin\left((2m+1+1)\frac{\pi}{2}\right)}{2m+1+1} + \frac{\sin\left((2m+1-1)\frac{\pi}{2}\right)}{2m+1-1} = \frac{\sin\left((2(m+1))\frac{\pi}{2}\right)}{2(m+1)} + \frac{\sin\left((2m)\frac{\pi}{2}\right)}{2m} = 0.$$

Therefore, we only need to consider $k \mapsto 2k$ for our terms:

$$\begin{aligned} a_k &= \frac{2}{\pi}\left(\frac{\sin\left((2k+1)\frac{\pi}{2}\right)}{2k+1} + \frac{\sin\left((2k-1)\frac{\pi}{2}\right)}{2k-1}\right) \\ &= \frac{2}{\pi}\left(\frac{(-1)^k}{2k+1} + \frac{-(-1)^k}{2k-1}\right) \\ &= \frac{2(-1)^k}{\pi}\left(\frac{1}{2k+1} - \frac{1}{2k-1}\right) \\ &= \frac{2(-1)^k}{\pi}\left(\frac{(2k-1)-(2k+1)}{(2k+1)(2k-1)}\right) \\ &= \frac{2(-1)^k}{\pi} \cdot \frac{-2}{(2k+1)(2k-1)} \\ &= \frac{4(-1)^{k+1}}{\pi} \cdot \frac{1}{4k^2-1} \\ &= \frac{4(-1)^{k+1}}{\pi(4k^2-1)} \end{aligned}$$

Also, since $|\cos(x)|$ is even, the b_k terms vanish. Therefore, we only need to calculate a_0, which is the following:

$$\begin{aligned} a_0 &= \frac{1}{\pi}\int_0^{2\pi} |\cos(x)|\,dx \\ &= \frac{2}{\pi}\Big(\int_0^{\frac{\pi}{2}} \cos(x)\,dx - \int_{\frac{\pi}{2}}^{\pi} \cos(x)\,dx\Big) \\ &= \frac{2}{\pi}\left(\sin(x)\Big|_0^{\frac{\pi}{2}} - \sin(x)\Big|_{\frac{\pi}{2}}^{\pi}\right) \\ &= \frac{4}{\pi}. \end{aligned}$$

□

Corollary 4.2. *The Chebyshev series of $f(x) := |x|$ for $x \in [-1,1]$ is given by:*

$$f_\infty(x) = \frac{2}{\pi} + \frac{4}{\pi}\sum_{k=1}^{\infty} \frac{(-1)^{k+1}}{4k^2-1} T_{2k}(x)$$

Proof. This follows with a change of variables since $T_n(\cos\theta) = \cos(n\theta)$ for $\theta \in [0,\pi]$. □

Remark 4.22. *Note that the Fourier series and Chebyshev series are related in a special way when the function $f : \mathbb{R} \to \mathbb{R}$ is even and 2π-periodic. This is because, in such cases, the sine terms in the Fourier series disappear (since $\sin(nx)$ is an odd function and integrates to zero over a symmetric interval), leaving only cosine terms. The cosine terms in the Fourier series correspond to the terms in the Chebyshev series. A more extensive review of this relationship can be found in [Boy00].*

Therefore, we can use this as the polynomial to approximate the absolute value as follows:

Definition 4.12. *For $x \in [-1,1]$, the absolute value approximation polynomial is defined as:*

$$P_\epsilon^{abs}(x) = \frac{2}{\pi} + \frac{4}{\pi}\sum_{k=1}^{d} \frac{(-1)^{k+1}}{4k^2-1} T_{2k}(x)$$

with $d = \mathcal{O}\left(\left\lceil \frac{1}{\pi\epsilon} \right\rceil\right)$ as the degree of the polynomial.

Note that the absolute value approximation polynomial is illustrated in Figure 4.5.

Lemma 4.1. *The degree of $P_\epsilon^{abs}(x)$ is even.*

Proof. Let $d \in \mathbb{N}$ be the degree of $P_\epsilon^{\text{abs}}(x)$. This polynomial is represented as a finite sum of Chebyshev polynomials $T_{2k}(x)$. The term with the highest degree is $T_{2d}(x)$, which has degree $2d$. Thus, the degree is even. □

Lemma 4.2. *$P_\epsilon^{abs}(x)$ is an ϵ-approximation to $|x|$ for $x \in [-1,1]$.*

Proof. Let $\epsilon > 0$. Consider the Chebyshev series $f_\infty(x)$ for $x \in [-1, 1]$. The truncated series $f_N(x)$ after N terms is:

$$f_N(x) = \frac{2}{\pi} + \frac{4}{\pi} \sum_{k=1}^{N} \frac{(-1)^{k+1}}{4k^2 - 1} T_{2k}(x)$$

Define $E_N(x) := ||x| - f_N(x)|$ as the error term after N iterations, then it is:

$$E_N(x) = ||x| - f_N(x)| = |f_\infty(x) - f_N(x)| = \left| \frac{4}{\pi} \sum_{k=N+1}^{\infty} \frac{(-1)^{k+1}}{4k^2 - 1} T_{2k}(x) \right|$$

We want to achieve $|E_N(x)| < \epsilon$. Since the Chebyshev polynomials satisfy $|T_{2k}(x)| \leq 1$ for $x \in [-1, 1]$, we can bound the error term:

$$|E_N(x)| \leq \frac{4}{\pi} \sum_{k=N+1}^{\infty} \left| \frac{(-1)^{k+1}}{4k^2 - 1} \right| = \frac{4}{\pi} \sum_{k=N+1}^{\infty} \frac{1}{4k^2 - 1} < \epsilon$$

For large k, the terms $\frac{1}{4k^2-1}$ can be approximated by $\frac{1}{4k^2}$. The series $\sum_{k=N+1}^{\infty} \frac{1}{k^2}$ is a well-known p-series with $p = 2$, which converges and can be bounded by:

$$\sum_{k=N+1}^{\infty} \frac{1}{4k^2} = \frac{1}{4} \sum_{k=N+1}^{\infty} \frac{1}{k^2} \leq \frac{1}{4} \int_N^{\infty} \frac{1}{x^2} dx = \frac{1}{4} \cdot \frac{1}{N}$$

So we have:

$$|E_N(x)| \leq \frac{4}{\pi} \sum_{k=N+1}^{\infty} \frac{1}{4k^2 - 1} \leq \frac{4}{\pi} \cdot \frac{1}{4} \cdot \frac{1}{N} = \frac{1}{\pi N} < \epsilon$$

To satisfy $|E_N(x)| < \epsilon$, we need $N > 1/\pi\epsilon$. Thus, for $N = \mathcal{O}(\lceil 1/\pi\epsilon \rceil)$, the polynomial $P_\epsilon^{\text{abs}}(x)$ is an ϵ-approximation to $|x|$ for $x \in [-1, 1]$. □

Theorem 4.6. *The absolute value approximation polynomial $P_\epsilon^{abs}(x)$ satisfies all requirements for the application of the QSVT Theorem 2.17:*

1. *$P_\epsilon^{abs}(x)$ is even.*
2. *$|P_\epsilon^{abs}(x)| \leq 1$ for all $x \in [-1, 1]$.*
3. *$P_\epsilon^{abs}(x)$ is an ϵ-approximation to $|x|$ for $x \in [-1, 1]$.*

Proof. Note that 1. is the result of Lemma 4.1 and 3. follows from Lemma 4.2.
We prove 2. explicitly. Let therefore $x \in [-1, 1]$. Consider the sum:

$$\sum_{k=1}^{d} \frac{(-1)^{k+1}}{4k^2 - 1} T_{2k}(x).$$

The Chebyshev polynomials $T_k(x)$ satisfy $|T_k(x)| \leq 1$ for any $k \in \mathbb{N}$ by definition. Therefore, the absolute value of each term in the sum is bounded by 1. To bound the entire sum, we consider the series:

$$\left| \sum_{k=1}^{d} \frac{(-1)^{k+1}}{4k^2 - 1} T_{2k}(x) \right| \leq \sum_{k=1}^{d} \left| \frac{(-1)^{k+1}}{4k^2 - 1} T_{2k}(x) \right| \leq \sum_{k=1}^{d} \frac{1}{4k^2 - 1}$$

We know from the last proof that for large k, $\frac{1}{4k^2-1}$ behaves similarly to $\frac{1}{4k^2}$, which is a convergent p-series with $p = 2$. Therefore, the sum $\sum_{k=1}^{\infty} \frac{1}{4k^2-1}$ is convergent and thus bounded by $1/N$ with $N = \mathcal{O}(\lceil 1/\pi\epsilon \rceil)$ for some $\epsilon > 0$ and its partial sums are bounded as well. Therefore, taking the absolute value of $P_\epsilon^{\text{abs}}(x)$ leaves us with:

$$\left|P_\epsilon^{\text{abs}}(x)\right| \leq \left|\frac{2}{\pi}\right| + \left|\frac{4}{\pi}\sum_{k=1}^{d}\frac{(-1)^{k+1}}{4k^2-1}T_{2k}(x)\right| \leq \frac{2}{\pi} + \frac{4}{\pi}\sum_{k=1}^{d}\frac{1}{4k^2-1} \leq \frac{2}{\pi} + \frac{4}{\pi N}.$$

Given that $\frac{2}{\pi} + \frac{4}{\pi N} \leq \frac{2}{\pi} \leq 1$ for an appropriate choice of ϵ.

□

Remark 4.23. *Note that points 1 and 3 in the above theorem are not explicit requirements for QSVT, as stated in Remark 2.32. However, point 1 allows us to choose the correct circuit for the QSVT operator since it distinguishes between odd and even (see Theorem 2.17 from Section 2.4.3). Point 3 ensures that the use of the polynomial is justified.*

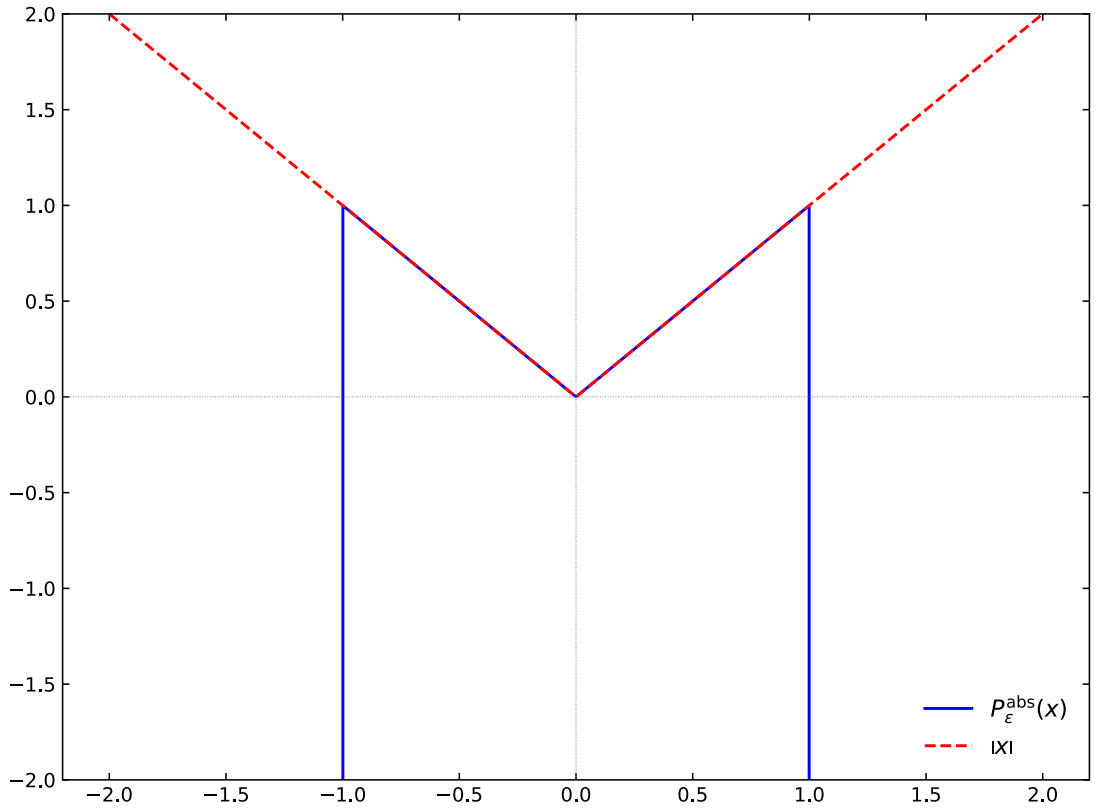

Figure 4.5: Comparison between the absolute value approximation polynomial $P_\epsilon^{\text{abs}}(x)$ from Definition 4.12 and the absolute value $|x|$. The approximation is computed with $\epsilon = 1 \times 10^{-4}$ and $d = \lceil 1/\pi\epsilon \rceil$. The x-axis represents the x values, and the y-axis represents the function values.

4.3 Overview of the Quantum Algorithm

The following is an overview of the components of the proposed quantum algorithm to solve the UCP from Definition 4.2 as shown in Figure 4.6:

State Preparation: Prepare the quantum state corresponding to the vector $\Delta P = (P_{k,t}\, u_{k,t} - L_{k,t})_{k\in \mathrm{B}, t\in \mathrm{T}}$. This step encodes the elements of ΔP into the amplitudes of a quantum state, ensuring proper normalization.

Block Encoding of $\mathbf{B}'$: To perform the inversion of the reduced susceptance matrix $\mathbf{B}'$, employ the Linear Combination of Unitaries (LCU) approach for block encoding. This involves expressing $\mathbf{B}'$ as a sum of unitary matrices and constructing the necessary operators for efficient block encoding.

Applying QSVT to Calculate $\mathbf{B}'^{-1}$: Using the block encoding of $\mathbf{B}'$, apply Quantum Singular Value Transformation (see "QSVT 1" in Figure 4.6) to compute the inverse of $\mathbf{B}'$. Construct a polynomial that approximates the matrix inversion and apply it using QSVT to calculate the voltage angles.

Block Encoding of the P_L-Vector: Construct a diagonal matrix corresponding to the power flow vector P_L as diagonal block encoding of the matrix D from Section 4.5. Prepare a quantum state with amplitudes matching the entries of P_L and use this state for efficient block encoding of the diagonal matrix.

Applying QSVT to Calculate $|P_{l,t}|$: Apply QSVT (see "QSVT 2" in Figure 4.6) to compute the absolute values of the power flow vector entries, ensuring correct accounting of the power flow through each transmission line in subsequent calculations.

Applying Quantum Amplitude Estimation (QAE): Given the absolute values of the power flow vector in quantum state form, employ QAE to calculate the scalar product $\langle |P_{l,t}|, C_l \rangle$ with the transmission cost coefficients, which is then incorporated into the cost Hamiltonian as H_{trans} in Section 4.8.1.5.

Applying the Quantum Approximate Optimization Algorithm (QAOA): Construct the cost Hamiltonian including terms representing operational costs, power balance, generator output limits, and transmission costs. Optimize the decision variables by constructing the QAOA state through alternating applications of phase separator and mixer operators. Calculate the expectation value of the cost Hamiltonian and use classical optimization techniques to find the optimal parameters. Measure the optimal quantum state, corresponding to the minimized cost Hamiltonian, to obtain the optimal operational states of the generators.

Remark 4.24 (Verification of the Algorithm)**.** *Given the input values specified in Section 4.1, the proposed quantum algorithm is confirmed to be well-defined and correct. It generates an approximated optimized set of operational states* $\{u_{i,t}\}_{i\in G,t\in T}$ *and the corresponding power outputs* $\{P_{i,t}\}_{i\in G,t\in T}$*, ensuring that all conditions outlined in Definition 4.2 are satisfied. The correctness of each component of the algorithm has been verified through the following:*

- ***State Preparation****: Theorem 4.7 confirms the accurate preparation of the initial quantum state based on the given input parameters.*
- ***Quantum Singular Value Transformation (QSVT)****: Theorem 2.17 ensures that QSVT correctly transforms the singular values as required.*
- ***Matrix Inversion Polynomial****: Theorem 4.4 verifies the correct approximation of matrix inversion by the polynomial used in QSVT.*
- ***Absolute Value Polynomial****: Theorem 4.6 confirms the accuracy of the polynomial involved.*
- ***Quantum Phase Estimation (QPE)****: Theorem 4.1, along with Lemma 4.2 and Corollary 4.5, establishes the accuracy of QPE in phase estimation, essential for constructing the cost function.*
- ***Quantum Approximate Optimization Algorithm (QAOA)****: The existence of a solution using QAOA is affirmed in Remark 4.41.*

Together, these verifications ensure that the algorithm functions correctly and achieves the desired outcomes under the specified conditions.

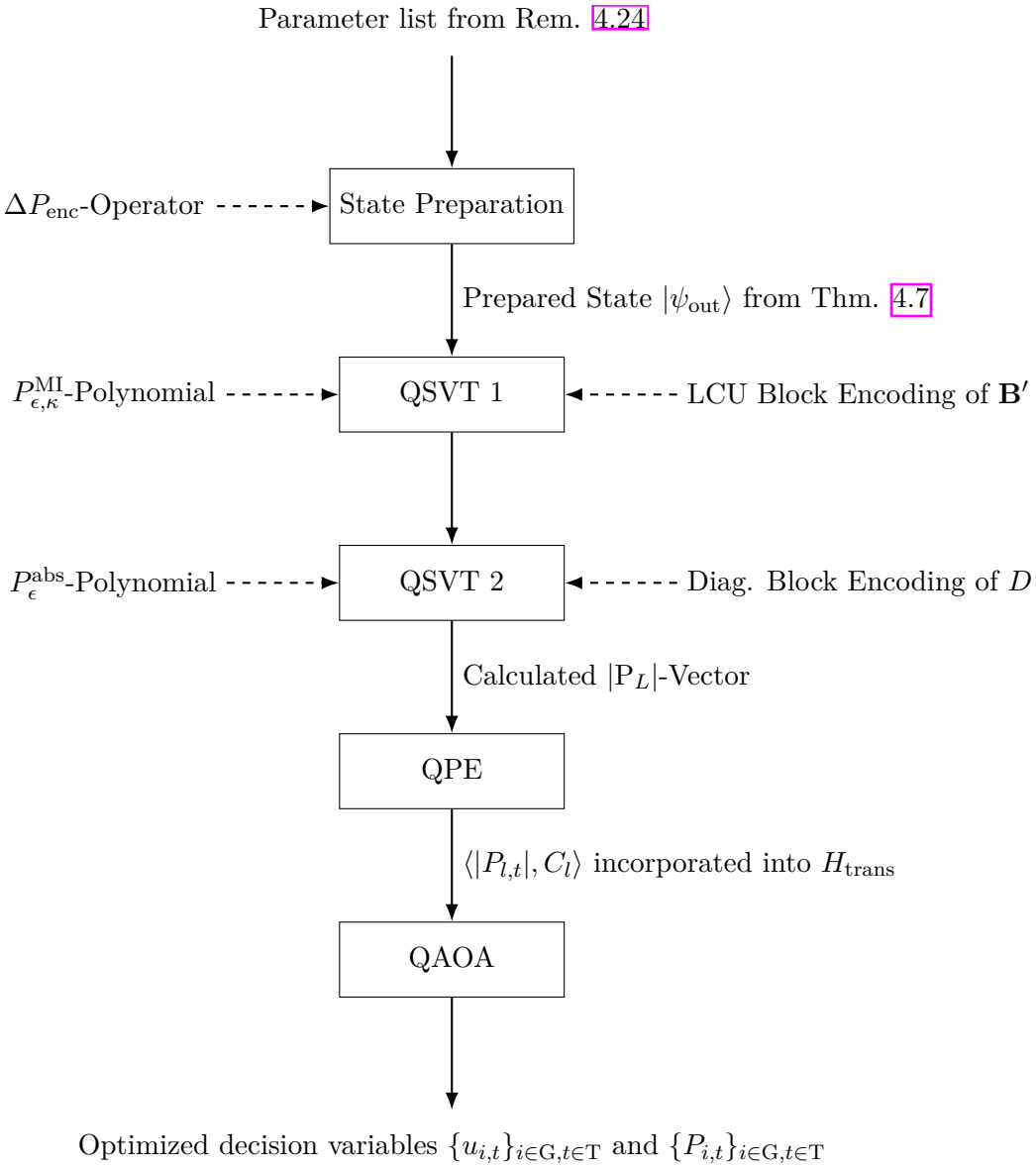

Figure 4.6: Flowchart illustrating the Quantum Optimization Algorithm for solving the UCP as described in the above Overview Section 4.3. Note that this figure is for visualization purposes only and does not represent the exact input data or computational processes of the algorithms. The diagram is simplified and sequentially arranged for clarity; however, it should be understood that the actual process involves nested and parallel operations that are more complex than demonstrated here.

4.4 The QSVT Matrix Inversion Procedure

4.4.1 Quantum State Preparation

Given $\Delta P \in \mathbb{R}^{|B|\cdot|T|}$ with $\Delta P = (P_{k,t}\, u_{k,t} - L_{k,t})_{k\in \mathrm{B}, t\in \mathrm{T}}$. We use the amplitude encoding method for quantum state preparation, following the methodology described in [SP18] and introduced by [MVBS04]. This method utilizes $R_y(\eta_i^{\Delta P})$ rotation gates about the y-axis to encode the elements of ΔP into the amplitudes of a quantum state. The required rotation angles, $\eta_i^{\Delta P}$, are derived to ensure the resulting quantum state corresponds to a normalized version of ΔP.

Definition 4.13. *Given the vector $\Delta P \in \mathbb{R}^{|B|\cdot|T|}$, the amplitude encoding operator ΔP_{enc} is defined by the sequence of unitary operations that prepare the quantum state $|\Delta P\rangle$. The operator ΔP_{enc} is defined as:*

$$\Delta P_{enc} : |0\rangle^{\otimes \log_2(|B|\cdot|T|)} \mapsto \frac{1}{\|\vec{V}\|} \sum_{k\in B, t\in T} V_{k,t} \,|k,t\rangle$$

where $|k,t\rangle$ is a given basis vector indexed by the pair (k,t) and

$$V_{k,t} := \begin{cases} P_{k,t} & \text{if } k \in G \\ -L_{k,t} & \text{if } k \notin G \end{cases}$$

where $P_{k,t}$ is the net power generated at the bus k and timestep t and $L_{k,t}$ the load. $\|\vec{V}\|$ is the norm of all the vectors including all $V_{k,t}$ for all $k \in B$.

Remark 4.25 (Index mapping)**.** *To prepare this quantum state, each rotation angle $\eta_j^{\Delta P}$ is given for $j \in \{1, 2, \ldots, |B| \cdot |T|\}$. Therefore we can use the following index mapping:*

$$j = (t-1)\cdot|B| + k \text{ for } k \in B \text{ and } t \in T$$

If the index set $T \not\subset \mathbb{N}$ use the position of the given index.

The value of $\eta_j^{\Delta P}$ is calculated according to the formula described in [MVBS04] where the $V_{k,t}$ are calculated for all $k \in B$ and $t \in T$ as by definition:

$$\eta_j^{\Delta P} = 2 \arcsin\left(\frac{V_j}{\sqrt{\sum_{k\in \mathrm{B}, t\in \mathrm{T}} |V_{k,t}|^2}}\right)$$

This formula also accounts for the normalization from the i-th component onwards. The implementation uses a series of controlled $R_y(\eta_i^{\Delta P})$ rotations. Note that R_z-rotations are not necessary in this circuit since the input values are real. The amplitude encoding operator ΔP_{enc} is incorporated into the whole state preparation circuit (see Figure 4.7).

Theorem 4.7. *The state preparation operator (see Figure 4.7) creates the following state* $|\psi_{out}\rangle \in \mathbb{C}^{2(n+1)}$ *as output of the first register assuming that all buses are connected to a generator where* $n = (|B| \cdot |T|)$*:*

$$|\psi_{out}\rangle = \frac{1}{\|\vec{V}\|} \begin{pmatrix} P_0 u_0 \\ P_1 u_1 \\ P_2 u_2 \\ P_3 u_3 \\ \vdots \\ P_n u_n \\ (1-u_0)P_0 \\ (1-u_1)P_1 \\ \vdots \\ (1-u_n)P_n \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

where $\|\vec{V}\|$ *is the normalization factor of the amplitude encoding operator and the indexes are mapped to one dimension using the mapping from Remark 4.25.*

Proof. Let $n = |B| \cdot |T|$. The initial state of the system can be written as:

$$|\psi_0\rangle = |0\rangle \otimes |0\rangle^{\otimes \log_2(n)} \otimes |u_0\rangle \otimes |u_1\rangle \otimes \cdots \otimes |u_n\rangle$$

The ΔP_{enc} operator encodes the vector of $V_{k,t}$ for $k \in B$ and $t \in T$ into the second register where $|k,t\rangle$ is a given basis vector indexed by the pair (k,t) and $V_{k,t} = P_{k,t}$ if $k \in G$ and $-L_{k,t}$ if $k \notin G$:

$$\Delta P_{\text{enc}} |0\rangle^{\otimes \log_2(n)} = \frac{1}{\|\vec{V}\|} \sum_{k \in \mathrm{B}, t \in \mathrm{T}} V_{k,t} |k,t\rangle$$

Assuming $B = G$ (i.e., all buses are connected to a generator), then $V_{k,t} = P_{k,t}$ for every $k \in B$ and $t \in T$. The state after this operation is:

$$|\psi_1\rangle = |0\rangle \otimes \left(\frac{1}{\|\vec{V}\|} \sum_{k \in \mathrm{B}, t \in \mathrm{T}} P_{k,t} |k,t\rangle \right) \otimes |u_0\rangle \otimes |u_1\rangle \otimes \cdots \otimes |u_n\rangle$$

Next, we apply a series of controlled-X gates, with the first qubit as the target and the second register and the $|u_i\rangle$ qubits as controls. The effect of these controlled-X gates can be understood as follows: Each controlled-X gate will flip the first qubit if and only if the corresponding control qubits (from the second register and the $|u_i\rangle$ qubits) are in the $|1\rangle$ state. Using the index mapping from Remark 4.25 so that $i = (t-1) \cdot |B| + k$ for $k \in \mathrm{B}$ and $t \in \mathrm{T}$, the state after applying all controlled-X gates can be expressed as:

$$|\psi_2\rangle = \frac{1}{\|\vec{V}\|} \sum_{i=0}^{n} P_i |u_i \oplus 0\rangle |i\rangle \otimes |u_0\rangle \otimes |u_1\rangle \otimes \cdots \otimes |u_n\rangle$$

Since the initial target qubit is in the $|0\rangle$ state, the $\oplus$ operation with u_i either flips or maintains the state based on the value of u_i:

$$|\psi_2\rangle = \frac{1}{\|\vec{V}\|} \sum_{i=0}^{n} P_i \, |u_i\rangle \, |i\rangle \otimes |u_0\rangle \otimes |u_1\rangle \otimes \cdots \otimes |u_n\rangle$$

This means the first qubit will be in the state $|u_i\rangle$ corresponding to the control qubits' state. To account for the $(1 - u_i)$ terms, we recognize that the state can be decomposed into parts where the qubit is $|0\rangle$ or $|1\rangle$ for each u_i:

$$|\psi_3\rangle = \frac{1}{\|\vec{V}\|} \sum_{i=0}^{n} P_i \left(u_i \, |1\rangle + (1 - u_i) \, |0\rangle\right) |i\rangle \otimes |u_0\rangle \otimes |u_1\rangle \otimes \cdots \otimes |u_n\rangle$$

When we combine these results, the first register's state after considering both u_i and $(1 - u_i)$ contributions is:

$$|\psi_{\text{out}}\rangle = \frac{1}{\|\vec{V}\|} \begin{pmatrix} P_0 u_0 & \dots & P_n u_n & (1 - u_0) P_0 & \dots & (1 - u_n) P_n & 0 & \dots & 0 \end{pmatrix}$$

□

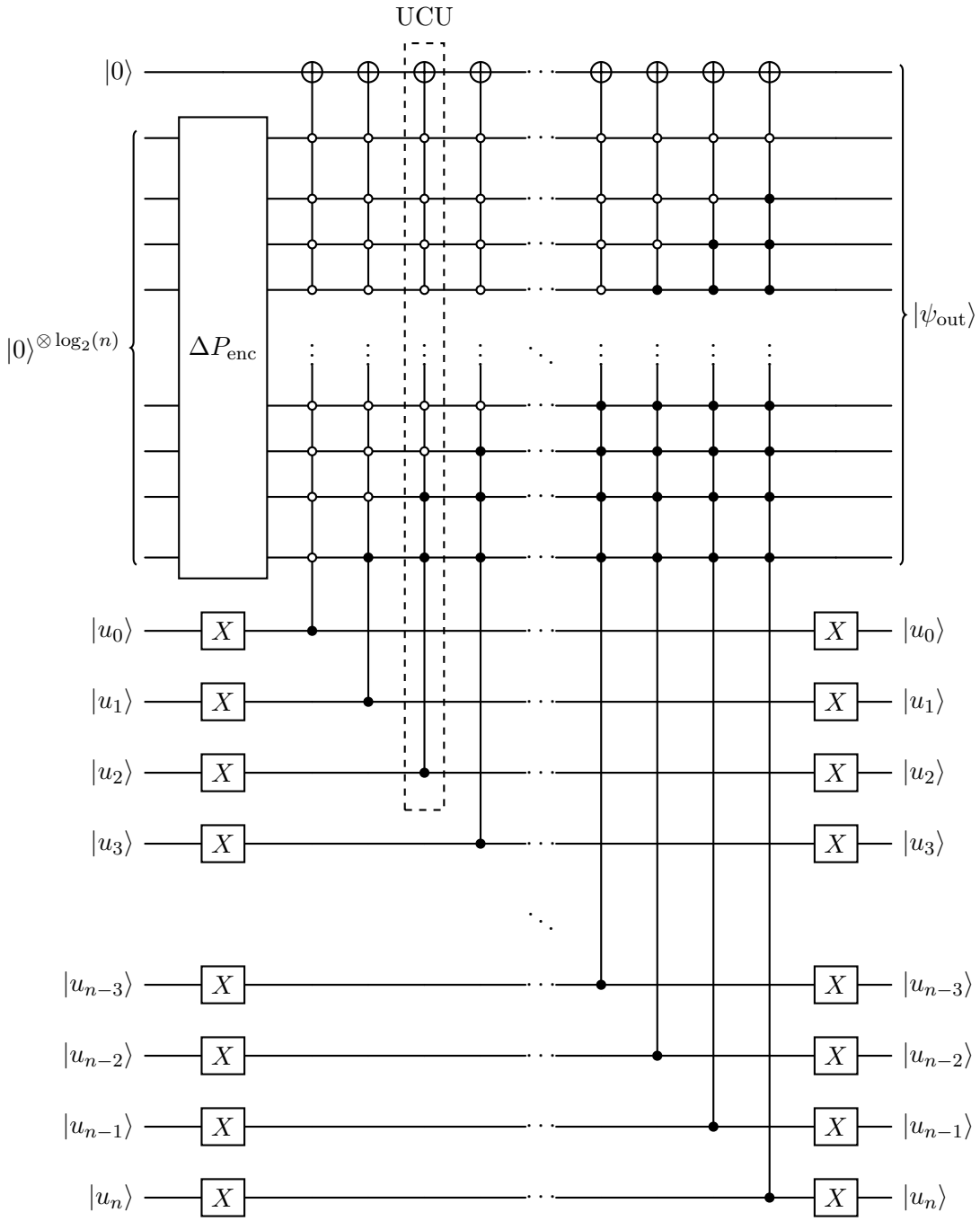

Figure 4.7: Quantum Circuit for State Preparation, where X are Pauli X-Gates and ΔP_{enc} is the amplitude encoding operator for $n = |G| \cdot |B|$. The output state $|\psi_{\text{out}}\rangle$ is described in Theorem 4.7.

Algorithm 4.5: Quantum State Preparation

Input : The values of $P_{k,t}$, u_i, and $L_{k,t}$ for $k \in B$, $i \in G$, and $t \in T$

Output : Quantum state $|\psi_{\text{out}}\rangle$ encoded in a quantum register, see Theorem 4.7 for an explicit form of the output state.

Complexity: See Theorem 5.1 from Section 5.1

Procedure :

Prepare the initial state as $|0\rangle \otimes |0\rangle^{\otimes \log_2(n)} \otimes |u_0\rangle \otimes |u_1\rangle \otimes \cdots \otimes |u_n\rangle$, where $n = |B| \cdot |T|$.

Calculate the Euclidean norm $\|\vec{V}\|$ of the vector $\vec{V}$ where

$$V_{k,t} := \begin{cases} P_{k,t} & \text{if } k \in G \\ -L_{k,t} & \text{if } k \notin G \end{cases}.$$

foreach $k \in B$ *and* $t \in T$ **do**

  Encode $V_{k,t}$ into the amplitude of the basis state $|k,t\rangle$ in the second register using the ΔP_{enc} operator:

$$\Delta P_{\text{enc}} |0\rangle^{\otimes \log_2(n)} = \frac{1}{\|\vec{V}\|} \sum_{k \in B, t \in T} V_{k,t} |k,t\rangle$$

foreach $i = 0$ *to* n **do**

  Apply a controlled-X gate with the first qubit as the target and the second register qubits and the $|u_i\rangle$ qubits as controls:

$$\text{Apply } C^n X \text{ gates to transform } |0\rangle \text{ to } |u_i \oplus 0\rangle$$

Account for both u_i and $1 - u_i$ contributions:

$$|\psi_3\rangle = \frac{1}{\|\vec{V}\|} \sum_{i=0}^{n} P_i \left(u_i |1\rangle + (1 - u_i) |0\rangle\right) |i\rangle \otimes |u_0\rangle \otimes |u_1\rangle \otimes \cdots \otimes |u_n\rangle$$

Combine the results to form the output state $|\psi_{\text{out}}\rangle$.

4.4.2 LCU Block Encoding of the Reduced Block Susceptance Matrix

We use the Linear Combination of Unitaries (LCU) approach (see Section 2.4.3.1) for block encoding the matrix $\mathbf{B}'$, following the methodology described in [CW12b] and discussed in [DMWL21]. To achieve the block encoding of the matrix $\mathbf{B}'$ according to Corollary 2.9, we first need to express it as an LCU, with $K \leq (|B| \cdot |T|)^2$ as follows:

$$\mathbf{B}' = \sum_{k=1}^{K} y_k U_k,$$

where $y_k \in \mathbb{C}$ are complex coefficients and $U_k \in \mathbb{R}^{|B||T| \times |B||T|}$ are unitary matrices. Let $\alpha = \sum_{k=1}^{K} |y_k|$ be a normalization factor and $a_k = \frac{y_k}{\alpha}$ be the normalized coefficients. For implementing the LCU-Operators, we follow a slightly modified version of [DMWL21].

Definition 4.14. *The Prepare operator for the matrix $\mathbf{B}'$ is defined to create a state:*

$$\mathrm{PREP}_{\mathbf{B}'} \in \mathcal{L}(\mathbb{C}^{2^n}, \mathbb{C}^K), \quad \mathrm{PREP}_{\mathbf{B}'}|0\rangle = \sum_{k=1}^{K} \sqrt{a_k}|k\rangle,$$

Definition 4.15. *The Select operator for the matrix $\mathbf{B}'$ is a controlled unitary defined as:*

$$\mathrm{SEL}_{\mathbf{B}'} \in \mathcal{L}(\mathbb{C}^K \otimes \mathbb{C}^n), \quad \mathrm{SEL}_{\mathbf{B}'} = \sum_{k=1}^{K} |k\rangle\langle k| \otimes U_k,$$

Note that for $\mathrm{PREP}_{\mathbf{B}'}$ the state $|0\rangle$ is the initial state of an ancillary register and $|k\rangle$ are computational basis states, for $\mathrm{SEL}_{\mathbf{B}'}$ the $|k\rangle\langle k|$ acts on an ancillary register and U_k on the target register as seen in the exemplary circuit 4.8.

Theorem 4.8. *For the matrix $\mathbf{B}'$ with $n = |B| \cdot |T|$ the unitary operator $U_{\mathbf{B}'}$ defined as:*

$$U_{\mathbf{B}'} := (\mathrm{PREP}_{\mathbf{B}'}^{\dagger} \otimes I_n) \cdot \mathrm{SEL}_{\mathbf{B}'} \cdot (\mathrm{PREP}_{\mathbf{B}'} \otimes I_n)$$

creates a block encoding for $\mathbf{B}'$, where $I_n \in \mathbb{R}^{n \times n}$ are the identity matrices.

Proof. To prove that $U_{\mathbf{B}'}$ is a block encoding of $\mathbf{B}'$, we need to show that:

$$\left\| \mathbf{B}' - \alpha \left(\langle 0|^{\otimes a} \otimes I \right) U_{\mathbf{B}'} \left(|0\rangle^{\otimes a} \otimes I \right) \right\| \leq \epsilon,$$

where $\alpha = \sum_{k=1}^{K} |y_k|$ and the operators are defined with $a_k = \frac{y_k}{\alpha}$ as:

$$\mathrm{PREP}_{\mathbf{B}'} |0\rangle = \sum_{k=1}^{K} \sqrt{a_k} |k\rangle, \quad \mathrm{SEL}_{\mathbf{B}'} = \sum_{k=1}^{K} |k\rangle \langle k| \otimes U_k, \quad \mathrm{PREP}_{\mathbf{B}'}^{\dagger} = \sum_{k=1}^{K} \sqrt{a_k} \langle k|.$$

Let $|\psi\rangle \in \mathbb{C}^n$ be an arbitrary quantum state. We apply $\mathrm{PREP}_{\mathbf{B}'} \otimes I_n$ to $|0\rangle \otimes |\psi\rangle$, and then $\mathrm{SEL}_{\mathbf{B}'}$ to this state:

$$\mathrm{SEL}_{\mathbf{B}'}\Big((\mathrm{PREP}_{\mathbf{B}'} \otimes I_n)(|0\rangle \otimes |\psi\rangle)\Big) = \mathrm{SEL}_{\mathbf{B}'} \left(\sum_{k=1}^{K} \sqrt{a_k} |k\rangle \otimes |\psi\rangle \right) = \sum_{k=1}^{K} \sqrt{a_k} |k\rangle \otimes U_k |\psi\rangle.$$

Applying the un-preparation operator $\text{PREP}^\dagger_{\mathbf{B}'} \otimes I_n$ to the above state and using the fact that $\langle k|j\rangle = \delta_{kj}$ (Kronecker delta), we get:

$$(\text{PREP}^\dagger_{\mathbf{B}'} \otimes I_n) \left(\sum_{k=1}^{K} \sqrt{a_k} \, |k\rangle \otimes U_k \, |\psi\rangle \right) = \left(\sum_{k=1}^{K} \sqrt{a_k} \, \langle k| \right) \left(\sum_{j=1}^{K} \sqrt{a_j} \, |j\rangle \otimes U_j \, |\psi\rangle \right)$$
$$= \sum_{k=1}^{K} a_k U_k \, |\psi\rangle \, .$$

Combining the results from the previous steps leaves us with:

$$U_{\mathbf{B}'}(|0\rangle \otimes |\psi\rangle) = \Big((\text{PREP}^\dagger_{\mathbf{B}'} \otimes I_n) \cdot \text{SEL}_A \cdot (\text{PREP}_{\mathbf{B}'} \otimes I_n) \Big) (|0\rangle \otimes |\psi\rangle)$$
$$= |0\rangle \otimes \left(\sum_{k=1}^{K} a_k U_k \right) |\psi\rangle \, .$$

We project onto $\langle 0|^{\otimes a} \cdot |0\rangle^{\otimes a}$:

$$\left(\langle 0|^{\otimes a} \otimes I \right) U_{\mathbf{B}'} \left(|0\rangle^{\otimes a} \otimes I \right) = \sum_{k=1}^{K} a_k U_k = \sum_{k=1}^{K} \frac{y_k}{\alpha} U_k = \frac{1}{\alpha} \sum_{k=1}^{K} y_k U_k = \alpha^{-1} \mathbf{B}'.$$

Since with the following the condition is satisfied:

$$\left\| \mathbf{B}' - \alpha \left(\langle 0|^{\otimes a} \otimes I \right) U_{\mathbf{B}'} \left(|0\rangle^{\otimes a} \otimes I \right) \right\| = \left\| \mathbf{B}' - \alpha(\alpha^{-1} \mathbf{B}') \right\| = 0 \leq \epsilon.$$

The operator $U_{\mathbf{B}'}$ defines a block encoding for $\mathbf{B}'$. □

The following circuit, $\text{PREP}_{\mathbf{B}'}$ prepares the state $\sum_{k=1}^{K} \sqrt{a_k} \, |k\rangle$ and $\text{SEL}_{\mathbf{B}'}$ is a controlled unitary operation $U = \sum_{k=1}^{K} |k\rangle \langle k| \otimes U_k$. Then $\text{PREP}^\dagger_{\mathbf{B}'}$ unprepares the ancilla state back to $|0\rangle^{\otimes \log K}$. Thus with:

$$U_{\mathbf{B}'} = (\text{PREP}^\dagger_{\mathbf{B}'} \otimes I_n) \cdot \text{SEL}_{\mathbf{B}'} \cdot (\text{PREP}_{\mathbf{B}'} \otimes I_n),$$

the matrix $\mathbf{B}'$ is block encoded as:

$$\mathbf{B}' = \alpha(\langle 0^{\otimes \log K}| \otimes I) U_{\mathbf{B}'} (|0^{\otimes \log K}\rangle \otimes I)$$

where $\alpha = \sum_{k=1}^{K} |y_k|$, the normalization factor.

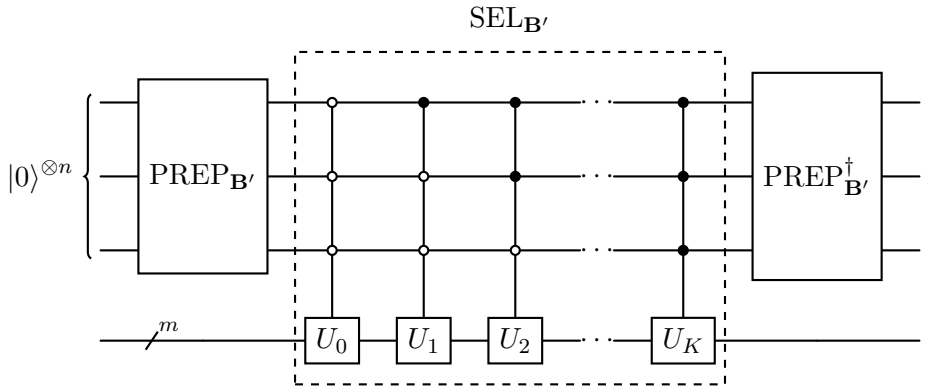

Figure 4.8: Exemplary Quantum Circuit for LCU-Block Encoding $\mathbf{B}'$ using the LCU method, with $n = \log_2(|B| \cdot |T|)$.

Algorithm 4.6: Quantum Circuit for LCU-Block Encoding of Matrix $\mathbf{B}'$

Input : The matrix $\mathbf{B}'$ that can be decomposed as $\mathbf{B}' = \sum_{k=1}^{K} y_k U_k$ where y_k are complex coefficients and U_k are unitary matrices.

Output : A quantum state proportional to $\mathbf{B}'|\Delta P\rangle$, specifically $\alpha^{-1}\mathbf{B}'|\Delta P\rangle$ where $\alpha = \sum_{k=1}^{K} |y_k|$.

Complexity: See Theorem 5.2 from Section 5.1

Procedure :

1 Calculate the normalization factor $\alpha = \sum_{k=1}^{K} |y_k|$ and the normalized coefficients $a_k = \frac{y_k}{\alpha}$.

2 Apply the Prepare operator PREP$_{\mathbf{B}'}$ to create the state PREP$_{\mathbf{B}'}|0\rangle = \sum_{k=1}^{K} \sqrt{a_k}|k\rangle$.

3 Apply the Select operator SEL$_{\mathbf{B}'} = \sum_{k=1}^{K} |k\rangle\langle k| \otimes U_k$ to the combined ancillary and target registers.

4 Apply the Unprepare operator PREP$^{\dagger}_{\mathbf{B}'}$ to return the ancillary register to the $|0\rangle$ state.

4.4.3 QSVT-Based Inversion of the Reduced Block Susceptance Matrix

We assume that for $\mathbf{B}'$ all its singular values $\sigma_i \in [\frac{1}{\kappa}, 1]$ for $\kappa \geq 1$, where κ is the condition number, the matrix is already rescaled according to Remark 4.10.

Remark 4.26. *For the inversion of $A \in \mathbb{R}^{n\times n}$ in [MRTC21], the matrix $A^\dagger$ is used. However, this is unnecessary for our case, because, if we consider a real-valued, full-rank, square matrix $A \in \mathbb{R}^{n\times n}$. Since A is real, $A = A^T$. For a full-rank matrix, all singular values are non-zero, ensuring invertibility. The distinction between A and $A^\dagger$ is important for complex-valued matrices or ensuring unitary evolution in quantum algorithms. For real-valued matrices, where $A = A^T$, operations involving A maintain the requisite properties without needing $A^\dagger$.*

The block encoding $U_{\mathbf{B}'}$ of Theorem 4.8, together with the matrix inversion polynomial from Definition 4.10 with Theorem 4.10 satisfy all prerequisites for the QSVT Theorem 2.17. Therefore with these, we can define a QSVT Operator:

Corollary 4.3. *Given the $U_{\mathbf{B}'}$ block encoding of Theorem 4.8 from Section 4.4.2 and the matrix inversion polynomial $P^{MI}_{\epsilon,\kappa}$ of Definition 4.10, the operator $\mathrm{QSVT}_{P^{MI}_{\epsilon,\kappa}}$ performing the Quantum Singular Value Transformation (see Theorem 2.17) can be constructed according to Figure 4.9. The projectors $\Pi_{\phi_k} = |0\rangle^{\otimes m}\langle 0|^{\otimes m} \otimes e^{i\phi_k\sigma_Z}$ are given with phase angles $\phi_k \in \mathbb{R}$ for $1 \leq k \leq \lfloor d/2 \rfloor$ locating $\mathbf{B}'$ inside the block encoding $U_{\mathbf{B}'}$. The operator is defined for $d = \mathcal{O}(\kappa \log(\kappa/\epsilon))$ odd, the degree of the matrix inversion polynomial as (see Theorem 4.4):*

$$\mathrm{QSVT}_{P^{MI}_{\epsilon,\kappa}}(U_{\mathbf{B}'}) = \Pi_{\phi_1} U_{\mathbf{B}'} \left[\prod_{k=1}^{(d-1)/2} \Pi_{\phi_{2k}} U^\dagger_{\mathbf{B}'} \Pi_{\phi_{2k+1}} U_{\mathbf{B}'} \right] = \begin{pmatrix} P^{MI(\mathrm{SV})}_{\epsilon,\kappa}(\mathbf{B}') & \cdot \\ \cdot & \cdot \end{pmatrix}$$

where $P^{MI(\mathrm{SV})}_{\epsilon,\kappa}(\mathbf{B}')$ is given as a block encoding for the singular value decomposition:

$$\mathbf{B}' = U\Sigma V^T$$

where $U \in \mathbb{R}^{|B||T|\times|B||T|}$ and $V \in \mathbb{R}^{|B||T|\times|B||T|}$ are orthogonal matrices, and a diagonal matrix $\Sigma \in \mathbb{R}^{|B||T|\times|B||T|}$ with singular values $\sigma_1, \sigma_2, \ldots, \sigma_{(|B||T|)}$ on its diagonal:

$$P^{MI(\mathrm{SV})}_{\epsilon,\kappa}(\mathbf{B}') := \sum_{j=1}^{|B||T|} P^{MI}_{\epsilon,\kappa}(\sigma_j) \, |u_j\rangle \langle v_j| \,, \tag{4.3}$$

Proof. Follows from Theorem 2.17. □

To calculate the phase factors for the QSVT-Sequence, one can apply the method described in Section 2.4.2.2.

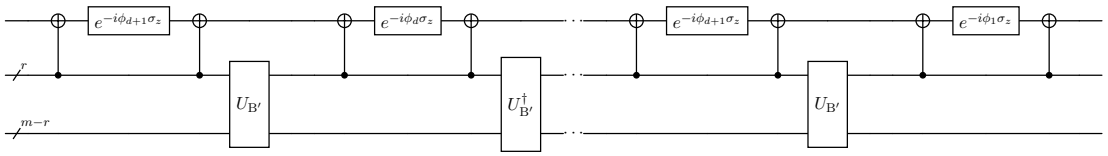

Figure 4.9: Quantum Circuit the $\mathrm{QSVT}_{P^{\mathrm{MI}}_{\epsilon,\kappa}}(U_{\mathbf{B}'})$-Operator if with the degree d of $P^{\mathrm{MI}}_{\epsilon,\kappa}$ odd, where $r = \log_2(|B||T|)$ and m-ancillas from the LCU Block encoding.

Algorithm 4.7: $\mathrm{QSVT}_{P^{\mathrm{MI}}_{\epsilon,\kappa}}$ Operator

Input: A Block encoding $U_{\mathbf{B}'}$ for $\mathbf{B}'$ with a condition number $\kappa \geq 1$ for the matrix $\mathbf{B}'$ and an error $\epsilon < 1/2$.
Output: A block encoded $\frac{\epsilon}{2\kappa}$-approximation of $\frac{\kappa}{2}\mathbf{B}'^{-1}$.
Complexity: See Theorem 5.3 from Section 5.1
Procedure :

1 Determine a unitary block encoding $U_{\mathbf{B}'}$ for $\mathbf{B}'$. (see Thm. 4.8)
2 Calculate the phase angles $\phi_k \in \mathbb{R}$ for $1 \leq k \leq \lfloor d/2 \rfloor$ using the method described in Section 2.4.2.2.
3 Apply the QSVT sequence to with this block encoding to compute $(P^{\mathrm{MI}}_{\epsilon,\kappa})^{(\mathrm{SV})}(\mathbf{B}')$, with the matrix inversion polynomial $P^{\mathrm{MI}}_{\epsilon,\kappa}(x)$ from Definition 4.10

4.5 Block Encoding of the Load Power Vector

Remark 4.27. *After applying the QSVT procedure, we obtain the pseudoinverse of the matrix* $\mathbf{B}'$ *(see Section 2.4.3.3). This allows us to calculate the voltage angles for each time step and, consequently, for each transmission line* $l \in L$ *between buses* $i, j \in B$*. At each time step* $t \in T$*, we calculate the power flow* $P_{l,t}$ *using the voltage angles and line susceptance* B_l*:*

$$P_{l,t} = B_l(\theta_{i,t} - \theta_{j,t})$$

where $\theta_{i,t}$ *and* $\theta_{j,t}$ *are the voltage angles at the buses connected by line* l*. The transmission cost for line* l *at time* t *is:*

$$c^{trans}_{l,t} = |P_{l,t} \cdot C_l|$$

We call $\mathrm{P}_L \in \mathbb{R}^{|L||T|}$ the Load Power Vector, which we will construct now. We again use block encoding for the vector $\mathrm{P}_L := (P_{l,t})_{l\in\mathrm{L},t\in\mathrm{T}}$ where $\mathrm{L} = \{l_1, l_2, \ldots, l_{|L|}\}$ is the index set of all transmission lines and $\mathrm{T} = \{t_1, t_2, \ldots, t_{|T|}\}$ the index set of all time steps. The vector can be expressed as:

$$\mathrm{P}_L = (P_{l_1,t_1}, P_{l_2,t_1}, \ldots, P_{l_{|L|},t_1}, P_{l_1,t_2}, \ldots, P_{l_{|L|},t_2}, \ldots, P_{l_1,t_{|T|}}, P_{l_2,t_{|T|}}, \ldots, P_{l_{|L|},t_{|T|}})^\top$$

This vector explicitly lists each element $P_{l,t}$ in a sequential manner, starting from t_1 and iterating through all l values, then moving to t_2 and so on, until $t_{|T|}$.
Define the set of indices where all entries of P_L are zero, i.e., the value $P_{l,t} = 0$ where no power flows through line l at times t:

$$Z = \{(l,t) \in L \times T \mid P_{l,t} = 0\}$$

Given the set Z, we can construct a vector P'_L by excluding the zero entries. Then, P'_L is formed by excluding the elements $P_{l,t}$ where $(l,t) \in Z$:

$$\mathrm{P}'_L = (P_{l,t} \mid (l,t) \in (L \times T) \setminus Z)$$

The resulting vector is $\mathrm{P}'_L \in \mathbb{R}^{|L||T|-|Z|}$. Thus, P'_L is the refined vector obtained by removing all $|Z|$-zero entries from P_L.
In order to process that refined vector using a block encoding, we construct a quadratic diagonal matrix with the entries $(P_{l,t})_{(l\in\mathrm{L},t\in\mathrm{T})}$ without Z of the vector P'_L on its diagonal. We need to create a matrix $D \in \mathbb{R}^{(|L|\cdot|T|-|Z|)\times(|L|\cdot|T|-|Z|)}$ with the elements of the vector on the diagonal and all other elements zero as follows:

$$D = \mathrm{diag}(P_{l_1,t_1}, P_{l_2,t_1}, \ldots, P_{l_{|L|},t_1}, P_{l_1,t_2}, P_{l_2,t_2}, \ldots, P_{l_{|L|},t_2}, \ldots, P_{l_1,t_{|T|}}, P_{l_2,t_{|T|}}, \ldots, P_{l_{|L|},t_{|T|}})$$

Let $n = |L| \times |T| - |Z|$. Then the matrix D can be represented as:

$$D = \begin{pmatrix} P_{l_1,t_1} & 0 & 0 & \cdots & 0 \\ 0 & P_{l_2,t_1} & 0 & \cdots & 0 \\ 0 & 0 & P_{l_3,t_1} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & P_{l_n,t_n} \end{pmatrix}$$

This construction ensures that D is a diagonal matrix with no zero entries on its diagonal, thus has full rank and is invertible. Also, the singular values are given through

Lemma 4.1. *Let $n = |L| \times |T| - |Z|$. The singular values $\sigma(D)$ of D are given through*

$$\sigma(D) = \{|P_{l_1,t_1}|, |P_{l_2,t_1}|, \ldots, |P_{l_n,t_n}|\}$$

Proof. The singular values are given as the square root of the eigenvalues of $D^T D$. For a diagonal matrix, the eigenvalues of:

$$D^T D = \mathrm{diag}(P^2_{l_1,t_1}, P^2_{l_2,t_1}, P^2_{l_3,t_1}, \ldots, P^2_{l_n,t_n})$$

are simply its diagonal elements. Applying the square root gives us the result. □

The following theorem provides a method proposed by [RR23] to block encode a diagonal matrix using a state preparation unitary. By preparing a quantum state whose amplitudes correspond to the diagonal entries of the matrix D, we can use the theorem to construct an efficient block encoding of D. This block encoding can then be used in our algorithm. Using this instead of LCU gives us more efficiency as seen later:

Theorem 4.9 ([RR23])**.** *Given an n-qubit quantum state specified by a state-preparation unitary U, such that $|\psi\rangle^{\otimes n} = U|0\rangle^{\otimes n} = \sum_{j=0}^{N-1} \psi_j |j\rangle^{\otimes n}$ (with $\psi_j \in \mathbb{C}$), we can prepare a $(1, n+3, 0)$-block-encoding U_A of the diagonal matrix $A = diag(\psi_0, \ldots, \psi_{N-1})$ with $O(n)$ circuit depth and a total of $O(1)$ queries to a controlled-U gate.*

Proof. The complete proof involves the construction of a number of truncated lemmas; therefore, for a complete proof see [RR23]. □

To use this theorem for block encoding the diagonal matrix D, we need a quantum state preparation unitary U such that it prepares the state $|\psi\rangle_n = \sum_{j=0}^{N-1} \psi_j |j\rangle_n$ where the ψ_j are the diagonal entries of D:

$$|\psi\rangle_n = \sum_{j=0}^{n-1} P_{l_j,t_j} |j\rangle_n$$

where $n = |L||T| - |Z|$. If we can prepare this state using a unitary U, then we receive a $(1, n+3, 0)$-block-encoding U_D of the diagonal matrix $D = \text{diag}(\psi_0, \ldots, \psi_{N-1})$ with $\psi_j = P_{l_j,t_j}$.

Theorem 4.10. *For the diagonal matrix D from above with $n = |L||T| - |Z|$ the unitary operator U_D creates a $(1, n+3, 0)$-block encoding for D, where $I_n \in \mathbb{R}^{n \times n}$ is the identity matrix.*

Proof. Follows from Theorem 4.9. □

Thus, with the above theorem, the matrix D is block encoded as:

$$D = \alpha(\langle 0^{\otimes \log K}| \otimes I) U_D (|0^{\otimes \log K}\rangle \otimes I)$$

where α is the normalization factor.

4.6 Applying QSVT for Absolute Value Computation

We again assume that for D all its singular values $\sigma_i \in [\frac{1}{\kappa}, 1]$ for $\kappa \geq 1$, where κ is the condition number. If D is not already rescaled, apply the procedure as described in Algorithm 4.10. The block encoding U_D of Theorem 4.10, together with the identity, which satisfies all prerequisites for the QSVT Theorem 2.17:

Corollary 4.4. *Given the U_D diagonal block encoding of Theorem 4.10 and the absolute value approximation polynomial P_ϵ^{abs} of Definition 4.12 from Section 4.2.3, the operator $\mathrm{QSVT}_{P_\epsilon^{abs}}$ performing the Quantum Singular Value Transformation (Theorem 2.17) can be constructed according to Figure 4.10. The projectors $\Pi_{\phi_k} = |0\rangle^{\otimes m}\langle 0|^{\otimes m} \otimes e^{i\phi_k\sigma_Z}$ are given with phase angles $\phi_k \in \mathbb{R}$ for $1 \leq k \leq \lfloor d/2 \rfloor$ locating D inside the block encoding U_D. The operator is defined for d even, where d is the degree of the absolute value approximation polynomial (see Theorem 4.6) as:*

$$\mathrm{QSVT}_{P_\epsilon^{abs}}(U_D) = \left[\prod_{k=1}^{d/2} \Pi_{\phi_{2k-1}} U_D^\dagger \Pi_{\phi_{2k}} U_D\right] = \begin{pmatrix} P_\epsilon^{abs\,(\mathrm{SV})}(D) & \cdot \\ \cdot & \cdot \end{pmatrix}$$

where $P_\epsilon^{abs\,(\mathrm{SV})}(D)$ is given as a block encoding for the singular value decomposition:

$$D = I_{|L||T|-|Z|} \cdot diag(|P_{l_1,t_1}|, |P_{l_2,t_1}|, \ldots, |P_{l_{|L|-|Z|},t_{|T|-|Z|}}|) \cdot I_{|L||T|-|Z|}$$

where $I_n \in \mathbb{R}^{(|L||T|-|Z|)\times(|L||T|-|Z|)}$ is the identity matrix:

$$P_\epsilon^{abs\,(\mathrm{SV})}(D) := \sum_{(l,t)\in(L\times T)\setminus Z} P_\epsilon^{abs}(P_{l,t}) e_j e_t^\top, \tag{4.4}$$

Proof. Follows from Theorem 2.17. □

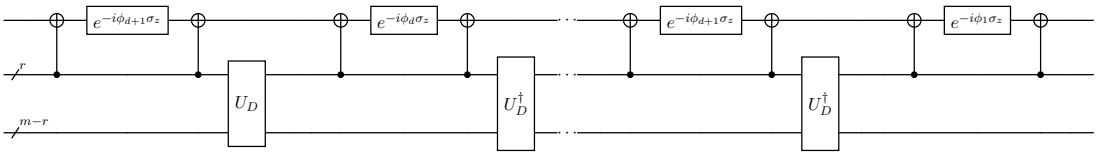

Figure 4.10: Quantum Circuit for the $\mathrm{QSVT}_{P_\epsilon^{\mathrm{abs}}}(U_D)$-Operator with the degree d of $P_\epsilon^{\mathrm{abs}}$ being even (see Lemma 4.6), where $r = \log_2(|B||T|)$ and m-ancillas from the diagonal block encoding.

Algorithm 4.8: QSVT$_{P_\epsilon^{\text{abs}}}$ Operator

Input: A block encoding U_D for D and an error $\epsilon > 0$.
Output: A block encoded ϵ-approximation of $|P_{l,t}|$.
Complexity: See Theorem 5.4 from Section 5.1
Procedure :

1 Determine a unitary block encoding U_D for D. (see Thm. 4.9)
2 Calculate the phase angles $\phi_k \in \mathbb{R}$ for $1 \leq k \leq \lfloor d/2 \rfloor$ using the method described in Section 2.4.2.2.
3 Apply the QSVT sequence to this block encoding to compute $P_\epsilon^{\text{abs(SV)}}(D)$, with the absolute value approximation polynomial $P_\epsilon^{\text{abs}}(x)$ from Definition 4.12

4.7 Applying the Quantum Amplitude Estimation

Given that the $P_{l,t}$ values are obtained from the QSVT and are thus already in a quantum state form, we can proceed directly to the subsequent steps without separate state preparation for $P_{l,t}$. Our goal is to calculate $\langle |P_{l,t}|, C_l \rangle$, which is then inserted and completes the cost function (see Def. 4.2).

Remark 4.28. *For the detailed definition of the Hilbert space for the power flow variables $P_{l,t}$, refer to Remark 4.37. We assume a Hilbert Space $\mathcal{H}$ of dimension $dim(\mathcal{H}) = 2^n$.*

Remark 4.29. *To avoid potential confusion with the notation $|P_{l,t}|$ when used within kets, we define $\text{abs}(P_{l,t}) := |P_{l,t}|$ in the following section.*

Definition 4.16. *The quantum state representing the transmission cost C_l for line $l \in L$ is prepared using the unitary operator U_{C_l}:*

$$U_{C_l} : \mathcal{H} \to \mathcal{H}, \quad |0\rangle^{\otimes n} \mapsto |C_l\rangle$$

Definition 4.17. *Define the Hermitian operator $U_H : \mathcal{H} \to \mathcal{H}$ for the Hadamard test as:*

$$U_H = U_{C_l}^\dagger U_{\text{abs}(P_{l,t})}$$

where $U_{\text{abs}(P_{l,t})}$ is the identity operator since $|\text{abs}(P_{l,t})\rangle$ is already in quantum state form from the QSVT procedure in the step before.

Remark 4.30. *The Hadamard test creates the superposition state necessary for estimating the real part of the inner product of the two quantum states:*

$$\frac{1}{\sqrt{2}} \left(|0\rangle \otimes |\text{abs}(P_{l,t})\rangle + |1\rangle \otimes U_H \, |\text{abs}(P_{l,t})\rangle \right) \in \mathcal{H} \otimes \mathbb{C}^2$$

Definition 4.18. *The Grover operator $\mathcal{Q} : \mathcal{H} \to \mathcal{H}$ is defined as:*

$$\mathcal{Q} := \mathcal{A} U_0 \mathcal{A}^\dagger U_{\psi_1}$$

where:

- *$\mathcal{A} : \mathcal{H} \to \mathcal{H}$ is the unitary operator that prepares the state $|\psi\rangle$.*

- $U_0 : \mathcal{H} \to \mathcal{H}, \quad U_0 = I - 2\left|0\right\rangle^{\otimes n}\left\langle 0\right|^{\otimes n}$ *is the diffusion operator, flipping the sign of the amplitude of the* $\left|0\right\rangle^{\otimes n}$ *state.*
- $U_{\psi_1} : \mathcal{H} \to \mathcal{H}, \quad U_{\psi_1} = Z \otimes I^{\otimes(n-1)}$ *applies a phase flip conditioned on the first qubit being* $\left|1\right\rangle$*.*

Lemma 4.1 (Quantum Amplitude Estimation [Thm. 12 [BHMT02]])**.** *Given an n-qubit unitary operator* $\mathcal{A} : \left|0\right\rangle^{\otimes n} \mapsto \left|\psi\right\rangle$*, where* $\left|\psi\right\rangle := \sqrt{1-a}\left|\psi_0\right\rangle + \sqrt{a}\left|\psi_1\right\rangle$*, we can compute a basis-encoded* ε*-approximation of* $\frac{2^m}{\pi}\arcsin\left(\sqrt{a}\right)$ *with success probability* $1-\delta$ *for any* $0 < \delta < 1$ *by applying Quantum Phase Estimation (QPE) on* $\left|\psi\right\rangle$ *and* $\mathcal{Q} := -\mathcal{A}U_0\mathcal{A}^\dagger U_{\psi_1}$*.*

Lemma 4.2 ([LHF24])**.** *Given* $\left|\psi\right\rangle \in \mathcal{H}$ *via* $U : \left|0\right\rangle^{\otimes n} \mapsto \left|\psi\right\rangle$*, we can find an* ε*-approximation of*

$$\frac{2^m}{\pi} \cdot \arcsin\left(\sqrt{\frac{1}{2} + \frac{\langle\psi|H|\psi\rangle}{2}}\right)$$

with success probability $1-\delta$ *for any* $0 < \delta < 1$ *by applying QAE on* $\mathcal{Q} := \mathcal{A}U_0\mathcal{A}^\dagger U_{\psi_1}$*, where* $U_0 := I - 2\left|0\right\rangle^{\otimes n}\left\langle 0\right|^{\otimes n}$*,* $U_{\psi_1} := Z \otimes I^{\otimes(n-1)}$*, and* $\mathcal{A}$ *represents a Hadamard test of* $\left|0\right\rangle^{\otimes m}\left|\psi\right\rangle$ *with* U_H*, a given* $(1, m, 0)$*-block-encoding of the subnormalized Hermitian matrix* $H \in \mathbb{C}^{2^n \times 2^n}$*.*

Corollary 4.5 ([LHF24])**.** *Given* $U_\psi : \left|0\right\rangle^{\otimes n} \mapsto \left|\psi\right\rangle$ *and* $U_\varphi : \left|0\right\rangle^{\otimes n} \mapsto \left|\varphi\right\rangle$*, we can compute an* ε*-approximation of*

$$\frac{2^m}{\pi} \cdot \arcsin\left(\sqrt{\frac{1}{2} + \frac{|\langle\varphi|\psi\rangle|}{2}}\right)$$

with success probability $1-\delta$ *for any* $0 < \delta < 1$ *by applying the algorithm proposed in Lemma 4.2 to* $m = 0$*,* $U = I^{\otimes n}$ *and* $U_H := U_\varphi^\dagger U_\psi$*.*

Remark 4.31. *Using Quantum Amplitude Estimation (QAE), we aim to obtain an* ε*-approximation of the scalar product* $\langle|P_{l,t}|, C_l\rangle$*. The goal is to estimate the amplitude* $a \in [0, 1]$*:*

$$\mathcal{A} : \mathcal{H} \to \mathcal{H}, \quad \left|0\right\rangle^{\otimes n} \mapsto \left|\mathrm{abs}(P_{l,t})\right\rangle := \sqrt{1-a}\left|\psi_0\right\rangle + \sqrt{a}\left|\psi_1\right\rangle$$

where the value a is related to the scalar product $|\langle C_l|P_{l,t}\rangle|$*.*
We use the Grover operator $\mathcal{Q}$ *iteratively and apply quantum phase estimation (QPE) to obtain an* ε*-approximation of:*

$$\frac{2^m}{\pi}\arcsin\left(\sqrt{a}\right)$$

We then obtain the ε*-approximation of the scalar product:*

$$\frac{2^m}{\pi}\arcsin\left(\sqrt{\frac{1}{2} + \frac{|\langle C_l|P_{l,t}\rangle|}{2}}\right)$$

and solve for $|\langle C_l|P_{l,t}\rangle|$*.*

Algorithm 4.9: QAE for Cost Calculation

Input : Quantum state $|\text{abs}(P_{l,t})\rangle$ with $\text{abs}(P_{l,t}) := |P_{l,t}|$ from QSVT, transmission cost state preparation unitary U_{C_l}, number of qubits n, precision parameter ε, success probability parameter δ

Output : ε-approximation of the scalar product $\langle |P_{l,t}|, C_l \rangle$

Complexity: See Theorem 5.5 from Section 5.1

Procedure :

1 Prepare the quantum state for the transmission cost $|C_l\rangle$ using the unitary operator:

$$U_{C_l} : \mathbb{C}^{2^n} \to \mathbb{C}^{2^n}, \quad |0\rangle^{\otimes n} \mapsto |C_l\rangle$$

2 Define the Hermitian operator for the Hadamard test:

$$U_H = U_{C_l}^\dagger U_{\text{abs}(P_{l,t})}$$

3 Perform the Hadamard test to prepare the superposition state:

$$\frac{1}{\sqrt{2}} \left(|0\rangle \otimes |\text{abs}(P_{l,t})\rangle + |1\rangle \otimes U_H |\text{abs}(P_{l,t})\rangle \right)$$

4 Define the Grover operator $\mathcal{Q}$:

$$\mathcal{Q} := \mathcal{A} U_0 \mathcal{A}^\dagger U_{\psi_1}$$

- $\mathcal{A} : \mathbb{C}^{2^n} \to \mathbb{C}^{2^n}$ is the unitary operator that prepares the state.
- $U_0 = I - 2 |0\rangle^{\otimes n} \langle 0|^{\otimes n}$ is the diffusion operator.
- $U_{\psi_1} = Z \otimes I$ applies a phase flip conditioned on the first qubit being $|1\rangle$.

Use Quantum Amplitude Estimation (QAE) to estimate the amplitude $a \in [0, 1]$:

$$\mathcal{A} : \mathbb{C}^{2^n} \to \mathbb{C}^{2^n}, \quad |0\rangle^{\otimes n} \mapsto |\text{abs}(P_{l,t})\rangle := \sqrt{1-a}\, |\psi_0\rangle + \sqrt{a}\, |\psi_1\rangle$$

Apply the Grover operator $\mathcal{Q}$ iteratively and use Quantum Phase Estimation (QPE) to obtain an ε-approximation of:

$$\frac{2^m}{\pi} \arcsin\left(\sqrt{a}\right)$$

where the value $a \in [0, 1]$ is related to the scalar product $|\langle C_l | P_{l,t} \rangle|$.
Obtain the ε-approximation of the scalar product:

$$\frac{2^m}{\pi} \arcsin\left(\sqrt{\frac{1}{2} + \frac{|\langle C_l | P_{l,t} \rangle|}{2}} \right)$$

Output the ε-approximation of $\langle |P_{l,t}|, C_l \rangle$.

4.8 Applying the Quantum Approximate Optimization Algorithm

4.8.1 Constructing the Cost Hamiltonian

To incorporate the constraints of the UCP formulation (see Definition 4.2) into the QAOA framework (see Section 2.4.4), we need to formulate the constraints as penalty terms within a Hamiltonian. This will ensure that the constraints are respected during the optimization process. Therefore, these penalties will be included in the cost Hamiltonian (as described in [ZWC+20, Luc14, Con24]).

4.8.1.1 QAOA Representation of the Generator States

The operational state of each generator is binary:

$$u_{i,t} \in \{0,1\}, \quad \forall i \in \mathrm{G},\ \forall t \in \mathrm{T}$$

This constraint is naturally handled by encoding $u_{i,t}$ into qubits where $\hat{u}_{i,t}$ is a quantum operator representing the binary variable. Each $\hat{u}_{i,t}$ is mapped to a single qubit, with $|0\rangle$ representing $u_{i,t} = 0$ (off) and $|1\rangle$ representing $u_{i,t} = 1$ (on).
The Ising model (see [Con24, NC10]) is a mathematical model used in statistical mechanics and quantum computing, where each spin (or binary variable) $s_i \in \{-1, +1\}$. The Ising Hamiltonian is typically written as (see [Con24]):

$$H_{\text{Ising}} = -\sum_i h_i s_i - \sum_{i<j} J_{ij} s_i s_j$$

However, in our formulation of the UCP (see Definition 4.2), the binary variables $u_{i,t} \in \{0,1\}$ are given for $i \in \mathrm{G}$ at time $t \in \mathrm{T}$. To translate this into an Ising form, we relate the binary variable $u_{i,t}$ to an Ising spin $s_{i,t}$ as follows:

$$s_{i,t} = 2u_{i,t} - 1 \quad \text{or equivalently,} \quad u_{i,t} = \frac{1}{2}(1 + s_{i,t})$$

In the quantum context, where $s_{i,t}$ is represented by the Pauli-Z matrix $\sigma^z_{i,t}$ with eigenvalues ± 1 (see Remark 2.23):

$$\hat{u}_{i,t} = \frac{1}{2}(I_u - \sigma^z_{i,t})$$

This form directly maps the problem into an Ising model (see [Luc14]) where $\sigma^z_{i,t}$ acts as the spin variable. The Hamiltonian for the UCP, formulated using $\hat{u}_{i,t}$, is then in Ising form, allowing the problem to be addressed using quantum algorithms that solve Ising models, such as the QAOA. To define $\hat{u}_{i,t}$, first consider the following Hilbert space:

Remark 4.32 (Hilbert Space for Binary Variables)**.** *Let $\mathcal{H}_2$ denote the two-dimensional Hilbert space associated with a single qubit (see Definition 2.31), defined as:*

$$\mathcal{H}_2 = \mathit{span}\{|0\rangle, |1\rangle\}$$

Since we consider a system with $|G|$ generators and $|T|$ time steps, the total Hilbert space

$\mathcal{H}_u$ for the binary variables is the tensor product of the individual qubit spaces:

$$\mathcal{H}_u = \left(\bigotimes_{i \in G} \bigotimes_{t \in T} \mathcal{H}_2 \right)$$

This space has dimension $\dim(\mathcal{H}_u) = 2^{|G| \cdot |T|}$.

Definition 4.19 (Pauli-Z Operator for Generator States)**.** *Given $\mathcal{H}_u$ as described in Remark 4.32, the operator $\sigma^z_{i,t} \in \mathcal{L}(\mathcal{H}_u)$ is the Pauli-Z operator acting on the qubit corresponding to generator $i \in G$ at time $t \in T$. The index j for this qubit is given by the index mapping in Remark 4.25:*

$$j = (t-1) \cdot |G| + i,$$

where $i \in G$ and $t \in T$. The operator $\sigma^z_{i,t}$ is then expressed as:

$$\sigma^z_{i,t} = I_2^{\otimes(j-1)} \otimes \sigma^z \otimes I_2^{\otimes(n-j)},$$

where $n = |G| \times |T|$ and j is the index calculated using the mapping from Remark 4.25. σ^z is the Pauli-Z operator (see Definition 2.34) acting on the j-th qubit, and $I_2 \in \mathcal{L}(\mathcal{H}_2)$ is the identity operator.

This definition ensures that the Pauli-Z operator acts specifically on the qubit corresponding to the generator i at time t, with identity operators acting on all other qubits.

Definition 4.20. *The operator $\hat{u}_{i,t} : \mathcal{H}_u \to \mathcal{H}_u$ where $\mathcal{H}_u$ is the Hilbert space described in Remark 4.32, is defined for a generator $i \in G$ at time $t \in T$ as:*

$$\hat{u}_{i,t} := \frac{1}{2}(I_u - \sigma^z_{i,t})$$

where $\sigma^z_{i,t}$ is the Pauli-Z operator acting on the qubit corresponding to generator $i \in G$ at time $t \in T$ from Definition 4.19, and $I_u \in \mathcal{L}(\mathcal{H}_u)$ is the identity operator.

Lemma 4.3. *The operator $\hat{u}_{i,t} \in \mathcal{L}(\mathcal{H}_u)$ is linear.*

Proof. Let $\psi, \phi \in \mathcal{H}_u$. Then:

$$\hat{u}_{i,t}(\psi + \phi) = \frac{1}{2}(I_u - \sigma^z_{i,t})(\psi + \phi) = \frac{1}{2}(I_u\psi + I_u\phi - \sigma^z_{i,t}\psi - \sigma^z_{i,t}\phi) = \hat{u}_{i,t}\psi + \hat{u}_{i,t}\phi$$

For any $\alpha \in \mathbb{C}$ and $\psi \in \mathcal{H}_u$:

$$\hat{u}_{i,t}(\alpha\psi) = \frac{1}{2}(I_u - \sigma^z_{i,t})(\alpha\psi) = \frac{1}{2}(\alpha I_u\psi - \alpha\sigma^z_{i,t}\psi) = \alpha\hat{u}_{i,t}\psi$$

□

Remark 4.33. *In the Hilbert space $\mathcal{H}_u$ (see Remark 4.32), the operator $\hat{u}_{i,t}$ acts as $\frac{1}{2}(I_2 - \sigma^z)$ on the qubit associated with generator $i \in G$ at time $t \in T$ and as the identity on all other qubits:*

$$\hat{u}_{i,t} = I_2 \otimes \cdots \otimes \frac{1}{2}(I_2 - \sigma^z) \otimes \cdots \otimes I_2$$

The construction of $\hat{u}_{i,t}$ ensures that it correctly represents the binary state of the generator in the Ising model form, allowing it to be used effectively in the QAOA algorithm (see [Luc14, Con24]).

We can define the Hamiltonian for the cost function or objective function of our optimization problem as follows:

Definition 4.21 (Cost Function Hamiltonian)**.**

$$H_{objective} := \sum_{t \in T} \sum_{i \in G} \left(c_{i,t}^{prod} \cdot \hat{u}_{i,t} + c_{i,t}^{start} \cdot (1 - \hat{u}_{i,t-1}) \hat{u}_{i,t} + c_{i,t}^{stop} \cdot \hat{u}_{i,t-1} (1 - \hat{u}_{i,t}) \right)$$

where $\hat{u}_{i,t} \in \mathcal{L}(\mathcal{H}_u)$ is the quantum operator from Definition 4.20 associated with the qubit representing the binary state of generator $i \in G$ at time $t \in T$. All other parameters are given in Definition 4.2 from Section 4.1.

4.8.1.2 QAOA Representation of the Power Outputs

To define the power output operator $\hat{P}_{i,t}$ for $i \in$ G and $t \in$ T within the QAOA framework, we proceed by constructing the operator using the Hilbert space structure and Pauli-Z matrices, similar to how the binary state operator $\hat{u}_{i,t}$ was defined. First, we need to discretize the power output variables $P_{i,t}$ (see [Sau13, PTVF07]):

Lemma 4.4 (Discretized Power Output Variables)**.** *Let $q_P \in \mathbb{N}$. Given the continuous power output variable $P_{i,t} \in [P_{\min,i}, P_{\max,i}]$ (see Definition 4.1) for $i \in G$ and $t \in T$, the variable can be expressed as:*

$$P_{i,t} = P_{\min,i} + \sum_{j=1}^{q_P} \Delta P_i \cdot 2^{j-1} \cdot b_{j,i,t},$$

where $\Delta P_i = \frac{P_{\max,i} - P_{\min,i}}{2^{q_P} - 1}$ and $b_{j,i,t} \in \{0, 1\}$ represents the j-th binary digit of the power output. This representation ensures that the entire range $[P_{\min,i}, P_{\max,i}]$ is covered by the binary variables.

Proof. Let $N \in \mathbb{N}$ be an integer. Note that it can be represented using $q_P \in \mathbb{N}$ binary digits, which has a standard binary expansion given by (see [PTVF07]):

$$N = \sum_{j=1}^{q_P} 2^{j-1} z_j,$$

where $z_j \in \{0, 1\}$ represents the j-th binary digit of N. To represent the continuous variable $P_{i,t}$ within the range $[P_{\min,i}, P_{\max,i}] \subset [0, \infty)$, we scale this binary expansion by defining a step size ΔP_i as:

$$\Delta P_i = \frac{P_{\max,i} - P_{\min,i}}{2^{q_P} - 1}.$$

Note that the step size ΔP_i ensures that the entire range is covered in discrete increments. We can then express $P_{i,t}$ as:

$$P_{i,t} = P_{\min,i} + \sum_{j=1}^{q_P} \Delta P_i \cdot 2^{j-1} \cdot b_{j,i,t},$$

where $b_{j,i,t} \in \{0,1\}$ are the binary variables corresponding to the value of $P_{i,t}$.
Consider the case when all $b_{j,i,t} = 0$:

$$P_{i,t} = P_{\min,i} + \sum_{j=1}^{q_P} \Delta P_i \cdot 2^{j-1} \cdot 0 = P_{\min,i}.$$

Consider the case when all $b_{j,i,t} = 1$:

$$\begin{aligned} P_{i,t} &= P_{\min,i} + \sum_{j=1}^{q_P} \Delta P_i \cdot 2^{j-1} = P_{\min,i} + \Delta P_i \cdot (2^{q_P} - 1) \\ &= P_{\min,i} + \left(\frac{P_{\max,i} - P_{\min,i}}{2^{q_P} - 1} \right) \cdot (2^{q_P} - 1) = P_{\min,i} + (P_{\max,i} - P_{\min,i}) = P_{\max,i} \end{aligned}$$

Thus, the expression for $P_{i,t}$ correctly maps the entire binary space to the continuous range $[P_{\min,i}, P_{\max,i}]$. □

Remark 4.34 (Hilbert Space for Binary-Encoded Power Output Variables)**.** *Each binary variable $b_{j,i,t}$ corresponds to the state of a qubit, where $|0\rangle$ represents $b_{j,i,t} = 0$ and $|1\rangle$ represents $b_{j,i,t} = 1$. The Hilbert space for q_P qubits is therefore:*

$$\mathcal{H}_{2^{q_P}} = span\{|0\rangle, |1\rangle, \ldots, |2^{q_P} - 1\rangle\},$$

where each basis state $|k\rangle \in \mathcal{H}_2$ with $k \in \{0, 1, \ldots, 2^{q_P} - 1\}$ (see Definition 2.33) represents a possible binary configuration of the qubits.

Remark 4.35 (Hilbert Space for Power Output Variables)**.** *Power output variables $P_{i,t}$ associated with generator $i \in G$ at time $t \in T$ need to be represented with a certain precision. Assume each power output variable is encoded using basis encoding (see [NC10]), where $q_P \in \mathbb{N}$ qubits are used to represent the binary expansion of $P_{i,t}$ (see [Sau13]):*

$$P_{i,t} = P_{\min,i} + \Delta P_i \sum_{j=1}^{q_P} 2^{j-1} b_{j,i,t},$$

where $b_{j,i,t} \in \{0,1\}$ represents the j-th binary digit of the power output. Each $b_{j,i,t}$ corresponds to the state of a qubit, where $|0\rangle$ represents $b_{j,i,t} = 0$ and $|1\rangle$ represents $b_{j,i,t} = 1$. The Hilbert space for q_P qubits is therefore:

$$\mathcal{H}_{2^{q_P}} = span\{|0\rangle, |1\rangle, \ldots, |2^{q_P} - 1\rangle\},$$

where each basis state $|k\rangle \in \mathcal{H}_{2^{q_P}}$ with $k \in \{0, 1, \ldots, 2^{q_P} - 1\}$ represents a possible binary configuration of the qubits. Since there are $|G|$ generators and $|T|$ time steps, the total number of power output variables is $|G| \cdot |T|$. Thus, the combined Hilbert space for all power output variables is:

$$\mathcal{H}_P = \left(\bigotimes_{i \in G} \bigotimes_{t \in T} \mathcal{H}_{2^{q_P}} \right).$$

The dimension of this Hilbert space is $\dim(\mathcal{H}_P) = (2^{q_P})^{|G| \cdot |T|} = 2^{q_P \cdot |G| \cdot |T|}$.

Definition 4.22 (Pauli-Z Operator for Power Output)**.** *Given the Hilbert space $\mathcal{H}_{2^{q_P}}$ as described in Remark 4.35, the operator $\sigma^z_{j,i,t} \in \mathcal{L}(\mathcal{H}_{2^{q_P}})$ is the Pauli-Z operator acting on the qubit associated with the j-th binary digit $b_{j,i,t}$ of the power output of generator $i \in G$ at time $t \in T$. Using the index mapping from Remark 4.25, the position k of the qubit corresponding to the j-th binary digit is given by:*

$$k = (t-1) \cdot |G| \cdot q_P + (i-1) \cdot q_P + j,$$

where $j \in \{1, 2, \ldots, q_P\}$, $i \in G$, and $t \in T$. The operator $\sigma^z_{j,i,t}$ is then expressed as:

$$\sigma^z_{j,i,t} = I_2^{\otimes(k-1)} \otimes \sigma^z \otimes I_2^{\otimes(n-k)},$$

where $n = |G| \times |T| \times q_P$ and k is the index calculated using the mapping described above. σ^z is the Pauli-Z operator acting on the k-th qubit, and $I_2 \in \mathcal{L}(\mathcal{H}_2)$ is the identity operator.

This ensures that the Pauli-Z operator $\sigma^z_{j,i,t}$ specifically acts on the qubit representing the j-th binary digit of the power output for generator i at time t, with identity operators acting on all other qubits. To translate the classical representation from Lemma 4.4 into the quantum domain, we associate each binary digit $b_{j,i,t}$ with a qubit, where $b_{j,i,t} = 0$ corresponds to the qubit state $|0\rangle \in \mathcal{H}_2$ and $b_{j,i,t} = 1$ corresponds to $|1\rangle$. The corresponding quantum operator for $b_{j,i,t}$ (see [Luc14, Con24]) is then:

$$\hat{b}_{j,i,t} = \frac{1}{2}(I_2 - \sigma^z_{j,i,t}),$$

where $\sigma^z_{j,i,t}$ is the Pauli-Z operator with eigenvalues ± 1. This operator effectively maps the qubit states to the binary values, with $\sigma^z_{j,i,t}$ corresponding to the quantum representation of the classical binary digit $b_{j,i,t}$. Given this mapping, the power output operator $\hat{P}_{i,t}$ is constructed by substituting the quantum operator $\hat{b}_{j,i,t}$ for each binary digit:

$$\begin{aligned}\hat{P}_{i,t} &= P_{\min,i} \cdot I_{2^{q_P}} + \Delta P_i \sum_{j=1}^{q_P} 2^{j-1} \cdot \hat{b}_{j,i,t} \\ &= P_{\min,i} \cdot I_{2^{q_P}} + \frac{\Delta P_i}{2} \sum_{j=1}^{q_P} 2^{j-1}(I_2 - \sigma^z_{j,i,t})\end{aligned}$$

We can therefore define this operator as follows:

Definition 4.23 (Quantum Operator for Power Output)**.** *The operator $\hat{P}_{i,t} : \mathcal{H}_{2^{q_P}} \to \mathcal{H}_{2^{q_P}}$ where $\mathcal{H}_{2^{q_P}}$ is the Hilbert space described in Remark 4.35, is defined for a generator $i \in G$ at time $t \in T$ as:*

$$\hat{P}_{i,t} := P_{\min,i} \cdot I_{2^{q_P}} + \frac{\Delta P_i}{2} \sum_{j=1}^{q_P} 2^{j-1} \left(I_2 - \sigma^z_{j,i,t}\right),$$

where $P_{\min,i}$ is the minimum power output of generator $i \in G$ (see Section 4.1), ΔP_i is the power increment for each binary digit, $\sigma^z_{j,i,t}$ is the Pauli-Z operator acting on the jth qubit as defined in Definition 4.22, and $I_{2^{q_P}} \in \mathcal{L}(\mathcal{H}_{2^{q_P}})$ is the identity operator on the Hilbert space $\mathcal{H}_{2^{q_P}}$.

Lemma 4.5. *The operator* $\hat{P}_{i,t} \in \mathcal{L}(\mathcal{H}_{2^{q_P}})$ *is linear.*

Proof. Let $\psi, \phi \in \mathcal{H}_{2^{q_P}}$. Then:

$$\begin{aligned}
\hat{P}_{i,t}(\psi+\phi) &= \left(P_{\min,i} \cdot I_{2^{q_P}} + \frac{\Delta P_i}{2} \sum_{j=1}^{q_P} 2^{j-1} \left(I_2 - \sigma^z_{j,i,t}\right)\right)(\psi+\phi) \\
&= P_{\min,i} \cdot I_{2^{q_P}}(\psi+\phi) + \frac{\Delta P_i}{2} \sum_{j=1}^{q_P} 2^{j-1} \left(I_2(\psi+\phi) - \sigma^z_{j,i,t}(\psi+\phi)\right) \\
&= P_{\min,i} \cdot (I_{2^{q_P}}\psi + I_{2^{q_P}}\phi) + \frac{\Delta P_i}{2} \sum_{j=1}^{q_P} 2^{j-1} \left(I_2\psi - \sigma^z_{j,i,t}\psi + I_2\phi - \sigma^z_{j,i,t}\phi\right) \\
&= \hat{P}_{i,t}\psi + \hat{P}_{i,t}\phi.
\end{aligned}$$

Consider $\alpha \in \mathbb{C}$ and $\psi \in \mathcal{H}_{2^{q_P}}$. We have:

$$\begin{aligned}
\hat{P}_{i,t}(\alpha\psi) &= \left(P_{\min,i} \cdot I_{2^{q_P}} + \frac{\Delta P_i}{2} \sum_{j=1}^{q_P} 2^{j-1} \left(I_2 - \sigma^z_{j,i,t}\right)\right)(\alpha\psi) \\
&= \alpha \left(P_{\min,i} \cdot I_{2^{q_P}} + \frac{\Delta P_i}{2} \sum_{j=1}^{q_P} 2^{j-1} \left(I_2 - \sigma^z_{j,i,t}\right)\right)\psi \\
&= \alpha \hat{P}_{i,t}\psi.
\end{aligned}$$

□

4.8.1.3 Power Balance Constraint

The power balance constraint ensures that the total generated power meets the total load at each time step:

$$\sum_{i \in \mathrm{G}} P_{i,t} = \sum_{k \in \mathrm{B}} L_{k,t}, \quad \forall t \in \mathrm{T}$$

This constraint can be incorporated as a penalty term in the cost Hamiltonian H_C:

Definition 4.24 (Power Balance Constraint as Penalty Term)**.**

$$H_{balance} := \sum_{t \in T} \lambda_{balance} \left(\sum_{i \in G} \hat{P}_{i,t} - \sum_{k \in B} L_{k,t}\right)^2$$

where $\lambda_{balance} \in \mathbb{R}^+$ *is a penalty parameter and* $\hat{P}_{i,t} \in \mathcal{L}(\mathcal{H}_{2^{q_P}})$ *as defined in Definition 4.23.*

4.8.1.4 Generator Output Limits

The following constraints ensure that the power output of each generator stays within its minimum and maximum limits:

$$P_i^{\min} \cdot u_{i,t} \leq P_{i,t} \leq P_i^{\max} \cdot u_{i,t}, \quad \forall i \in \mathrm{G},\ \forall t \in \mathrm{T}$$

This can be incorporated as two penalty terms:

Definition 4.25 (Generator Output Limits as Penalty Term)**.**

$$H_{limits} := \sum_{t \in T} \sum_{i \in G} \left(\alpha_{\max} \left(\hat{P}_{i,t} - P_{\max,i} \cdot \hat{u}_{i,t} \right)^2 + \beta_{\min} \left(P_{\min,i} \cdot \hat{u}_{i,t} - \hat{P}_{i,t} \right)^2 \right)$$

where $\alpha_{\max}, \beta_{\min} \in \mathbb{R}$ *are penalty parameters.* $\hat{P}_{i,t} \in \mathcal{L}(\mathcal{H}_{2^{q_P}})$ *is described in Definition 4.23 and* $\hat{u}_{i,t} \in \mathcal{L}(\mathcal{H}_u)$ *is described in Definition 4.20.*

4.8.1.5 Transmission Cost

The transmission cost through each line should reflect the power flow through that line:

$$c_{l,t}^{\text{trans}} := |P_{l,t}| \cdot C_l, \quad \forall l \in \text{L}, \ \forall t \in \text{T}$$

To ensure the transmission cost correctly penalizes the power flow, we include it directly in the cost Hamiltonian:

Definition 4.26 (Transmission Cost Constraints as Penalty Term)**.**

$$H_{trans} := \sum_{t \in T} \sum_{l \in L} \left(|\hat{P}_{l,t}| \cdot C_l \right)$$

where $C_l \in \mathbb{R}^+$ *is the cost per unit power flow for line* $l \in L$*, and* $\hat{P}_{l,t}$ *are quantum operators acting on the Hilbert space* $\mathcal{H}_{P_l}$ *associated with the power flow variables.*

Remark 4.36. *In this context,* $\hat{P}_{l,t}$ *denotes the quantum operator corresponding to the classical power flow variable* $P_{l,t}$*. While* $P_{l,t}$ *represents the actual power flow as a real number,* $\hat{P}_{l,t}$ *is its quantum analog, used within the quantum algorithm to encode and manipulate this value in a quantum state. Upon measurement,* $\hat{P}_{l,t}$ *gives the classical value* $P_{l,t}$*.*

Remark 4.37 (Hilbert Space for Power Flow Variables in Amplitude Encoding)**.** *The power flow variables* $P_{l,t}$ *are represented using amplitude encoding (see Section 4.4.1). The Hilbert space dimension* $\mathcal{H}_{P_l}$ *is determined by the number of qubits* $n_{P_l} \in \mathbb{N}$ *used to encode the entire system:*

$$\dim(\mathcal{H}_{P_l}) = 2^{n_{P_l}}.$$

4.8.1.6 Combined Cost Hamiltonian

Combining all the terms, the cost Hamiltonian H_C is the sum of the components representing production, start-up, shut-down, power balance, generator limits, and transmission costs:

Definition 4.27 (Combined Cost Hamiltonian)**.**

$$H_C := H_{objective} + H_{balance} + H_{limits} + H_{trans}$$

where H_C is called the cost Hamiltonian, an operator which acts on a Hilbert space $\mathcal{H}'$ defined through the Hilbert spaces of the individual operators.

Remark 4.38. *The total Hilbert space for the entire system is the tensor product of all individual Hilbert spaces:*

$$\mathcal{H}' = \mathcal{H}_u \otimes \mathcal{H}_P \otimes \mathcal{H}_{P_l}$$

The dimension of the total Hilbert space is therefore:

$$\begin{aligned}\dim(\mathcal{H}') &= \dim(\mathcal{H}_u) \cdot \dim(\mathcal{H}_P) \cdot \dim(\mathcal{H}_{P_l}) \\ &= 2^{|G|\cdot|T|} \cdot 2^{q_P\cdot|G|\cdot|T|} \cdot 2^{n_{P_l}} \\ &= 2^{|G|\cdot|T|+q_P\cdot|G|\cdot|T|+n_{P_l}}\end{aligned}$$

4.8.2 Obtaining the Optimized Decision Variables and Cost

To obtain the cost associated with the approximate optimal parameters $\vec{\gamma}^*$ and $\vec{\beta}^*$ before performing the final measurement, we can incorporate the expectation value calculation directly into the classical optimization routine. This will provide an estimate of the cost for the state corresponding to $\vec{\gamma}^*$ and $\vec{\beta}^*$.

Remark 4.39. *The initial state $|\psi_0\rangle \in (\mathbb{C}^2)^{\otimes n}$ is a superposition of all possible states:*

$$|\psi_0\rangle = \frac{1}{\sqrt{2^n}} \sum_{z \in \{0,1\}^n} |z\rangle$$

where $n = |G| \cdot |T|$.

Definition 4.28 (Phase Separator)**.** *For a given parameter $\gamma \in [\gamma_{\min}, \gamma_{\max}] \subset \mathbb{R}$ we define:*

$$U(H_C, \gamma) = e^{-i\gamma H_C}$$

Definition 4.29 (Mixer Hamiltonian)**.** *The mixer Hamiltonian is defined as:*

$$H_B = -\sum_j X_j$$

where X_j is the Pauli-X operator acting on the j-th qubit. Its unitary operator for a given parameter $\beta \in [\beta_{\min}, \beta_{\max}] \subset \mathbb{R}$ is:

$$U(H_B, \beta) = e^{-i\beta H_B}$$

Note that this is a well-defined matrix exponential because H_C is a Hermitian operator. The matrix exponential of H_B is well defined by Lemma 2.1 from Section 2.4.1.

Definition 4.30 (QAOA State)**.** *The QAOA state is constructed by alternating applications of the phase separator and mixer operators*
:

$$|\psi_p(\vec{\gamma}, \vec{\beta})\rangle = U(H_B, \beta_p)U(H_C, \gamma_p) \cdots U(H_B, \beta_1)U(H_C, \gamma_1)|\psi_0\rangle$$

Here, $p \in \mathbb{N}$ is the number of QAOA layers, $\vec{\gamma} = (\gamma_1, \gamma_2, \ldots, \gamma_p)$ and $\vec{\beta} = (\beta_1, \beta_2, \ldots, \beta_p)$ are the sets of parameters to be optimized.

Definition 4.31 (Expectation Value of the Cost Hamiltonian)**.** *The expectation value of the cost Hamiltonian H_C with respect to the QAOA state $|\psi_p(\vec{\gamma}, \vec{\beta})\rangle$ is calculated as follows:*

$$\langle H_C \rangle_{\vec{\gamma},\vec{\beta}} = \langle \psi_p(\vec{\gamma}, \vec{\beta}) | H_C | \psi_p(\vec{\gamma}, \vec{\beta}) \rangle$$

This expectation value represents the average cost for a given set of parameters $\vec{\gamma}$ and $\vec{\beta}$.

Remark 4.40. *The expectation value $\langle H_C \rangle_{\vec{\gamma},\vec{\beta}}$ in the QAOA can be computed using a quantum circuit as follows. After preparing the QAOA state $|\psi_p(\vec{\gamma}, \vec{\beta})\rangle$, the cost Hamiltonian H_C is measured by decomposing it into a sum of Pauli operators. Each operator's expectation value is estimated through repeated measurements on the quantum circuit. The overall expectation value is then the weighted sum of these measurements, which is used in a classical optimization loop to adjust the parameters $\vec{\gamma}$ and $\vec{\beta}$.*

The following problem is then to be solved by classical methods:

Definition 4.32 (Classical Optimization Problem)**.** *The optimization problem can be described as:*

$$(\vec{\gamma}^*, \vec{\beta}^*) = \arg\min_{\vec{\gamma},\vec{\beta}} \langle H_C \rangle_{\vec{\gamma},\vec{\beta}}$$

Remark 4.41 (Existence of a Solution)**.** *We can conclude that the expectation value*

$$f(\vec{\gamma}, \vec{\beta}) := \langle H_C \rangle_{\vec{\gamma},\vec{\beta}} = \langle \psi_p(\vec{\gamma}, \vec{\beta}) | H_C | \psi_p(\vec{\gamma}, \vec{\beta}) \rangle$$

is bounded because the Cost Hamiltonian H_C operates on a finite-dimensional Hilbert space, and its expectation values cannot exceed certain limits. Also the expectation value $f(\vec{\gamma}, \vec{\beta})$ as a function of the parameters $\vec{\gamma}$ and $\vec{\beta}$ is continuous, because it involves matrix exponentials and inner products of finite-dimensional vectors, which are continuous operations. We also demand the parameters within bounded intervals:

$$\gamma_i \in [\gamma_{\min}, \gamma_{\max}] \quad \textit{and} \quad \beta_i \in [\beta_{\min}, \beta_{\max}]$$

for all $i \in \{1, 2, \ldots, p\}$ with $p \in \mathbb{N}$ the number of QAOA layers. Therefore:

$$[\gamma_{\min}, \gamma_{\max}]^p \times [\beta_{\min}, \beta_{\max}]^p$$

the parameter space is closed and bounded, thus compact. For a continuous function on a compact set, the extreme value theorem guarantees that the function attains its minimum values. Thus, the function $f(\vec{\gamma}, \vec{\beta})$ will attain its minimum within that set. Therefore we can assert that there exists a solution $(\vec{\gamma}^*, \vec{\beta}^*)$ that minimizes the expectation value of the Cost Hamiltonian H_C in the QAOA framework:

$$(\vec{\gamma}^*, \vec{\beta}^*) = \arg\min_{\vec{\gamma},\vec{\beta}} \langle H_C \rangle_{\vec{\gamma},\vec{\beta}}$$

This ensures that an optimal set of parameters exists, making the problem well-defined and solvable.

Remark 4.42 (Minimal Expectation Value)**.** *The corresponding minimal expectation value is:*

$$Cost_{\min} = \langle H_C \rangle_{\vec{\gamma}^*,\vec{\beta}^*}$$

Remark 4.43 (Steps for Classical Optimization)**.** *To obtain the cost at the optimal parameters $\vec{\gamma}^*$ and $\vec{\beta}^*$, we follow these steps:*

1. *Initialize the parameters $\vec{\gamma}$ and $\vec{\beta}$.*
2. *Evaluate the expectation value $\langle H_C \rangle_{\vec{\gamma},\vec{\beta}}$ using a quantum simulator or quantum hardware for the current parameters $\vec{\gamma}$ and $\vec{\beta}$.*
3. *Use a classical optimization algorithm (e.g., gradient descent, Nelder-Mead) to adjust the parameters $\vec{\gamma}$ and $\vec{\beta}$ to minimize the expectation value (see [Sau13]).*
4. *Store the minimum expectation value encountered during the optimization process. This is the cost associated with the optimal parameters $\vec{\gamma}^*$ and $\vec{\beta}^*$.*

Definition 4.33 (Optimal Quantum State)**.** *After finding the optimal parameters $\vec{\gamma}^*$ and $\vec{\beta}^*$, the corresponding quantum state is:*

$$|\psi_p(\vec{\gamma}^*, \vec{\beta}^*)\rangle$$

Remark 4.44. *As the number of layers p in QAOA increases, the algorithm's performance improves, with the expected value $\langle H_C \rangle_{\vec{\gamma},\vec{\beta}}$ converging to the optimal value C_{max} as $p \to \infty$. This implies that QAOA can theoretically reach the optimal solution with sufficiently large p. For details, see [FH19].*

Remark 4.45 (Measurement of the Optimal State)**.** *Measure this optimal quantum state multiple times to obtain a distribution over possible solutions. Each measurement gives a bitstring representing a specific configuration of the binary variables $u_{i,t}$, which corresponds to the operational state of the generators.*

5 Results

5.1 Complexity Analysis of Individual Components

5.1.1 Complexity Analysis of the State Preparation

For our optimization problem (see Section 4.1), the state preparation (see Section 4.4.1) involves encoding a vector ΔP into a quantum state and then applying a series of unitary operations controlled by multiple qubits. The amplitude encoding operator ΔP_{enc} (see Def. 4.13) requires $\log_2(n)$ qubits, where $n = |B| \cdot |T|$. The circuit depth for amplitude encoding is $\mathcal{O}(n)$. The amplitude encoding involves single-qubit R_y rotations. Uniformly Controlled Unitary (UCU) operations (for a definition see [YZ23]) are used to apply a set of unitary transformations controlled by multiple qubits. A UCU is a set of unitary operations controlled by multiple qubits (see [YZ23]). The state preparation circuit involves multiple UCUs. Consequently, we can apply the following lemma to achieve an efficient implementation. We utilize the following lemma from [YZ23]:

Lemma 5.1 (Lem. 11 [YZ23])**.** *For all $x \in \{0,1\}^q$, suppose that W^x can be implemented by a standard p-qubit quantum circuit of depth d. Then for any $M \geq pq$, the (q,p)-UCU $\sum_{x\in\{0,1\}^q} |x\rangle\langle x| \otimes W^x$ can be implemented by a standard quantum circuit of depth $\mathcal{O}\left(\log p + dq + \frac{dp2^q}{M}\right)$ and size $\mathcal{O}(dp2^q)$ with M ancillary qubits. In particular, if we have sufficiently many ancillary qubits, then the circuit depth for a (q,p)-UCU is $\mathcal{O}(\log p + dq)$.*

Remark 5.1. *A standard p-qubit quantum circuit refers to a quantum circuit that operates on p qubits using a sequence of quantum gates from a universal gate set, including single-qubit gates (such as Pauli-X, Pauli-Y, Pauli-Z, Hadamard, and R_y rotations) and two-qubit gates like CNOT. The circuit depth is defined by the number of sequential layers of operations, where each layer consists of gates that can be applied in parallel across disjoint qubits. This structure ensures that the quantum circuit efficiently implements the desired unitary operations on the p qubits. A rigorous definition can be found in [YZ23].*

Remark 5.2. *The operator W^x denotes a unitary operation applied to a set of p qubits, conditional on the control qubits being in the state corresponding to the binary string $x \in \{0,1\}^q$. In the context of uniformly controlled unitary (UCU) operations, W^x encapsulates the specific transformation enacted based on the control state x, thereby enabling complex controlled operations within the quantum circuit. A rigorous definition can be found in [YZ23].*

Theorem 5.1 (Complexity Analysis of the State Preparation)**.** *The state preparation circuit (see Section 4.4.1) for ΔP has the following complexity:*

- *Circuit depth:* $\mathcal{O}(\log_2(|B| \cdot |T|) + \log_2(|G| + 1) + \log_2(m))$
- *Qubit count:* $\mathcal{O}(\log_2(|B| \cdot |T|) + \log_2(m) + \log_2(|G| + 1))$
- *Cumulative error:* $\mathcal{O}(\log_2(|B| \cdot |T|)\epsilon_{UCU} + \log_2(m)\epsilon_{UCU})$

where $\epsilon_{UCU} \geq 0$ represents the error associated with the uniformly controlled unitary (UCU) operations. The number of buses is m, and the index sets are defined as stated in Def. 4.2.

Proof. We prove this by combining the complexities from the amplitude encoding and UCU operations (see Figure 4.7). The amplitude encoding operator ΔP_{enc} encodes the vector ΔP into the amplitudes of a quantum state using R_y rotations. Given the vector $\Delta P \in \mathbb{R}^{|B|\cdot|T|}$ (see Section 4.4.1), the encoding requires, according to the construction of the circuit following [MVBS04] and [SP18], $\log_2(|B| \cdot |T|)$ qubits. The circuit depth for amplitude encoding is then $\mathcal{O}(|B| \cdot |T|)$. This depth is linear in the number of elements in ΔP because each element requires a sequence of rotations and controlled operations to encode its amplitude. We can improve this further. With Remark 5.1, we can identify the Uniformly Controlled Unitary (UCU) operations inside the amplitude encoding operator ΔP_{enc} and apply Lemma 5.1 from [YZ23]. Note that $q = \log_2(|B| \cdot |T|)$, $p = 1$ (for the single-qubit rotations in amplitude encoding), and $d = \mathcal{O}(1)$. If sufficient ancillary qubits are available ($m \geq pq = \log_2(|B| \cdot |T|)$), the depth simplifies to $\mathcal{O}(\log_2(|B| \cdot |T|))$. The amplitude encoding involves single-qubit R_y rotations. Each rotation identified as UCU can be associated with an error $\epsilon_{UCU} \geq 0$. We implicitly allow this error to be zero as it will be neglected in most cases. The cumulative error for $n = |B| \cdot |T|$ rotations is then $\mathcal{O}(|B| \cdot |T|\epsilon_{UCU})$. We utilize Lemma 5.1 from [YZ23] for the rest of the state preparation circuit. Applying this, we consider the following parameters: $q = \log_2(m)$ (where m is the number of buses), $d = \log_2(m)$ (the depth of the individual unitary operations), and $p = |G| + 1$ (the number of generators + 1). According to the lemma, the UCU operations then require $\log_2(m) + \log_2(|G| + 1)$ qubits and a circuit depth of $\mathcal{O}(\log p + dq) = \mathcal{O}(\log_2(|G| + 1) + \log_2(m))$. Each controlled unitary operation can be associated with an error $\epsilon_{UCU} \geq 0$. Again, we implicitly allow this error to be zero as it will be neglected in most cases. The cumulative error is then $\mathcal{O}(\log_2(m)\epsilon_{UCU})$. By combining the analyses for the UCU operations, we derive the overall complexity and error metrics for the state preparation operator. The total circuit depth is:

$$\mathcal{O}(\log_2(|B|\cdot|T|))+\mathcal{O}(\log_2(|G|+1)+\log_2(m)) = \mathcal{O}(\log_2(|B|\cdot|T|)+\log_2(|G|+1)+\log_2(m))$$

The total qubit count necessary for implementing the full state preparation circuit (see Figure 4.7) is:

$$\mathcal{O}(\log_2(|B|\cdot|T|))+\mathcal{O}(\log_2(m)+\log_2(|G|+1)) = \mathcal{O}(\log_2(|B|\cdot|T|)+\log_2(m)+\log_2(|G|+1))$$

The cumulative error is therefore:

$$\mathcal{O}(\log_2(|B| \cdot |T|)\epsilon_{UCU}) + \mathcal{O}(\log_2(m)\epsilon_{UCU}) = \mathcal{O}(\log_2(|B| \cdot |T|)\epsilon_{UCU} + \log_2(m)\epsilon_{UCU})$$

□

	ΔP_{enc}-Operator	UCU Operations
Circuit Depth	$\mathcal{O}(\log_2(\lvert B\rvert \cdot \lvert T\rvert))$	$\mathcal{O}(\log_2(\lvert G\rvert + 1) + \log_2(m))$
Qubit Count	$\mathcal{O}(\log_2(\lvert B\rvert \cdot \lvert T\rvert))$	$\mathcal{O}(\log_2(m) + \log_2(\lvert G\rvert + 1))$
Potential Errors	$\mathcal{O}(\log_2(\lvert B\rvert \cdot \lvert T\rvert)\epsilon_{UCU})$	$\mathcal{O}(\log_2(m)\epsilon_{UCU})$
Total Circuit Depth	$\mathcal{O}(\log_2(\lvert B\rvert \cdot \lvert T\rvert) + \log_2(\lvert G\rvert + 1) + \log_2(m))$	
Total Qubit Count	$\mathcal{O}(\log_2(\lvert B\rvert \cdot \lvert T\rvert) + \log_2(m) + \log_2(\lvert G\rvert + 1))$	
Total Potential Errors	$\mathcal{O}(\log_2(\lvert B\rvert \cdot \lvert T\rvert)\epsilon_{UCU} + \log_2(m)\epsilon_{UCU})$	

Table 5.1: Complexity Analysis for Quantum State Preparation of ΔP

5.1.2 Complexity Analysis of the LCU Block Encoding

In the context of the LCU block encoding (see Section 4.4.2) for the matrix $\mathbf{B}'$, the parameter K represents the number of terms in the linear combination. The worst-case scenario is when $K = (|B| \cdot |T|)^2$, which represents the maximum number of terms if every element in the matrix $\mathbf{B}'$ needs to be represented by a distinct unitary matrix. The best-case scenario for K would be when the matrix $\mathbf{B}'$ is highly sparse and has a very structured form, such that it can be decomposed into a much smaller number of unitary matrices. In the ideal best-case scenario, the number of terms K could be minimal, such as $K = \mathcal{O}(|B| \cdot |T|)$, if each row or column of $\mathbf{B}'$ can be represented with a single unitary.

Theorem 5.2 (Complexity Analysis for LCU Block Encoding of $\mathbf{B}'$)**.** *The LCU block encoding for* $\mathbf{B}'$ *has the following complexity:*

- *Circuit depth:* $\mathcal{O}(Kn)$
- *Qubit count:* $\mathcal{O}(\log_2(K) + \log_2(|B| \cdot |T|))$
- *Cumulative error:* $\mathcal{O}(K(\epsilon_{PREP} + n\epsilon_{SEL} + \epsilon_{UNPREP}))$

where $\mathcal{O}(|B| \cdot |T|) \leq K \leq \mathcal{O}((|B| \cdot |T|)^2)$ *and* $n = |B| \cdot |T|$. $\epsilon_{PREP}, \epsilon_{SEL}, \epsilon_{UNPREP} \geq 0$ *are associated with the respective operations.*

Proof. The proof involves analyzing the complexity contributions from the Prepare, Select, and Unprepare operators in the LCU block encoding process (see Section 4.4.2). The Circuit Depth can be determined as follows:
The state preparation step involves creating the state $\text{PREP}_{\mathbf{B}'} |0\rangle = \sum_{k=1}^{K} \sqrt{a_k} |k\rangle$. This requires $\mathcal{O}(K)$ gates where K is the number of terms in the LCU decomposition. In the worst case, $K = (|B| \cdot |T|)^2$. In the best case, $K = |B| \cdot |T|$. The controlled unitary operation $\text{SEL}_{\mathbf{B}'} = \sum_{k=1}^{K} |k\rangle\langle k| \otimes U_k$ has a circuit depth depending on the complexity of each U_k. If each U_k can be implemented with a depth of $\mathcal{O}(n)$, the total depth is $\mathcal{O}(Kn)$. The unprepare operation has the same complexity as the prepare operation, $\mathcal{O}(K)$. Thus, the total circuit depth for the LCU block encoding is:

$$\mathcal{O}(K) + \mathcal{O}(Kn) + \mathcal{O}(K) = \mathcal{O}(Kn)$$

The Qubit Count can be determined as follows. $\text{PREP}_{\mathbf{B}'}$ and $\text{SEL}_{\mathbf{B}'}$ require ancillary qubits to store the index k. This needs $\log_2(K)$ qubits. For $K \leq (|B| \cdot |T|)^2$, we need $2\log_2(|B| \cdot |T|)$ qubits. The target register requires $n = \log_2(|B| \cdot |T|)$ qubits. Therefore, the total qubit count is:

$$\mathcal{O}(\log_2(K) + n) = \mathcal{O}(\log_2((|B| \cdot |T|)^2) + \log_2(|B| \cdot |T|)) = \mathcal{O}(2\log_2(|B| \cdot |T|))$$

Combining the operations for $\epsilon_{\text{PREP}}, \epsilon_{\text{SEL}} \geq 0$. Assuming they can as well be neglected, we can describe the cumulative error as:

$$\mathcal{O}(K\epsilon_{\text{PREP}}) + \mathcal{O}(Kn\epsilon_{\text{SEL}}) + \mathcal{O}(K\epsilon_{\text{PREP}}) = \mathcal{O}(K(2\epsilon_{\text{PREP}} + n\epsilon_{\text{SEL}}))$$

□

	$\text{PREP}_{\mathbf{B}'}$**-Operator**	$\text{SEL}_{\mathbf{B}'}$**-Operator**
Circuit Depth	$O(K)$	$O(Kn)$
Qubit Count	$O(\log_2(K))$	-
Potential Errors	$O(K\epsilon_{\text{PREP}})$	$O(Kn\epsilon_{\text{SEL}})$
Total Circuit Depth	$O(Kn)$	
Total Qubit Count	$O(2\log_2(\lvert B\rvert \cdot \lvert T\rvert))$	
Total Potential Errors	$\mathcal{O}(K(2\epsilon_{\text{PREP}} + n\epsilon_{\text{SEL}}))$	

Table 5.2: Complexity Analysis for Block Encoding of $\mathbf{B}'$ where $\mathcal{O}(|B| \cdot |T|) \leq K \leq \mathcal{O}((|B| \cdot |T|)^2)$ and $n = |B| \cdot |T|$. The errors $\epsilon_{\text{PREP}}, \epsilon_{\text{SEL}}, \epsilon_{\text{UNPREP}} \geq 0$ are associated with the respective operations.

5.1.3 Complexity Analysis of QSVT for Matrix Inversion

The following is the Complexity Analysis for the Quantum Singular Value Transformation (QSVT) of $\mathbf{B}'$ (see Section 2.4.3.3):

Theorem 5.3 (Complexity Analysis of QSVT Matrix Inversion)**.** *The QSVT matrix inversion for* $\mathbf{B}'$ *has the following complexity:*

- *Circuit depth:* $\mathcal{O}(K + \kappa \log(\kappa/\epsilon_{MI}))$
- *Qubit count:* $\mathcal{O}(\log_2(K) + \log_2(|B| \cdot |T|))$
- *Cumulative error:* $\mathcal{O}(\epsilon_{LCU} + \epsilon_{QSVT1})$

where $\epsilon_{LCU} \geq 0$ *with* $\epsilon_{LCU} = \mathcal{O}(K(2\epsilon_{PREP} + n\epsilon_{SEL}))$ *for* $n = |B| \cdot |T|$ *and* $\epsilon_{QSVT1} > 0$ *is related to* κ, ϵ_{MI} *are errors assigned to the LCU and QSVT operations and* $\mathcal{O}(|B| \cdot |T|) \leq K \leq \mathcal{O}((|B| \cdot |T|)^2)$.

Proof. We prove this by combining the complexities from the LCU block encoding and QSVT application (see Section 4.4.2 for explicit definition). For the LCU block encoding, we decompose $\mathbf{B}'$ as a sum of K unitary matrices, requiring $\mathcal{O}(K)$ for $\mathcal{O}(|B| \cdot |T|) \leq K \leq \mathcal{O}((|B| \cdot |T|)^2)$ circuit depth and $\mathcal{O}(\log_2(K))$ qubits. The error associated with this block encoding is $\epsilon_{LCU} = \mathcal{O}(K(2\epsilon_{\text{PREP}} + n\epsilon_{\text{SEL}})) \geq 0$ for $n = |B| \cdot |T|$. For the QSVT application, the degree of the polynomial for matrix inversion is $d = \mathcal{O}(\kappa \log(\kappa/\epsilon_{MI}))$, where $\epsilon_{MI} > 0$ is the precision of the matrix inversion polynomial approximation and κ is the condition number of B′ which is leading to a circuit depth of $\mathcal{O}(\kappa \log(\kappa/\epsilon_{MI}))$ and the qubit count needed remains $\mathcal{O}(\log_2(|B| \cdot |T|))$. The error associated with the QSVT is $\epsilon_{QSVT1} \geq 0$. Combining these, the total circuit depth is:

$$\mathcal{O}(K) + \mathcal{O}(\kappa \log(\kappa/\epsilon_{MI})) = \mathcal{O}(K + \kappa \log(\kappa/\epsilon_{MI}))$$

The total qubit count necessary is thus $\mathcal{O}(\log_2(K) + \log_2(|B| \cdot |T|))$. □

	LCU Block Encoding	**QSVT Application**
Circuit Depth	$\mathcal{O}(K)$	$\mathcal{O}(\kappa \log(\kappa/\epsilon_{MI}))$
Qubit Count	$\mathcal{O}(\log_2(K))$	$\mathcal{O}(\log_2(\lvert B\rvert \cdot \lvert T\rvert))$
Potential Errors	$\mathcal{O}(\epsilon_{LCU})$	$\mathcal{O}(\epsilon_{QSVT})$
Total Circuit Depth	$\mathcal{O}(K + \kappa \log(\kappa/\epsilon_{MI}))$	
Total Qubit Count	$\mathcal{O}(\log_2(K) + \log_2(\lvert B\rvert \cdot \lvert T\rvert))$	
Total Potential Errors	$\mathcal{O}(\epsilon_{LCU} + \epsilon_{QSVT1})$	

Table 5.3: Complexity Analysis for QSVT Matrix Inversion where $n \leq K \leq n^2$ for $n := |B| \cdot |T|$ and $\epsilon_{LCU} = \mathcal{O}(K(2\epsilon_{\text{PREP}} + n\epsilon_{\text{SEL}}))$.

5.1.4 Complexity Analysis of QSVT for Absolute Value Calculation

The following is the Complexity Analysis for Quantum Singular Value Transformation (QSVT) of D (see Section 4.5 and Section 4.6).

Theorem 5.4 (Complexity Analysis of QSVT Absolute Value Calculation)**.** *The QSVT for calculating $|P_{l,t}|$ using the block encoding of D has the following complexity:*

- *Circuit depth:* $\mathcal{O}(|L| \cdot |T| - |Z| + 1/\epsilon_{abs})$
- *Qubit count:* $\mathcal{O}(\log_2(|L| \cdot |T| - |Z|))$
- *Cumulative error:* $\mathcal{O}(\epsilon_{diag} + \epsilon_{QSVT2})$

where $\epsilon_{diag} \geq 0$ and $\epsilon_{QSVT2} > 0$ is related to ϵ_{abs} are errors assigned to the block encoding and QSVT operations respectively.

Proof. For the block encoding, we encode the diagonal matrix D using the method described in Theorem 4.9. The preparation of the state requires $\mathcal{O}(|L| \cdot |T| - |Z|)$ circuit depth and $\mathcal{O}(\log_2(|L| \cdot |T| - |Z|))$ qubits according to the theorem. The error associated with this block encoding is $\epsilon_{diag} \geq 0$. Assuming it can be neglected. For the QSVT application, the degree of the polynomial for the absolute value approximation (see Theorem 4.6) is $\mathcal{O}(\lceil 1/\pi\epsilon_{abs} \rceil)$, leading to a circuit depth of $\mathcal{O}(\lceil 1/\pi\epsilon_{abs} \rceil)$. The qubit count needed remains $\mathcal{O}(\log_2(|L| \cdot |T| - |Z|))$. The error associated with the QSVT is $\epsilon_{QSVT2} > 0$ and is related to ϵ_{abs}, the precision of the absolute value approximation. Combining these, the total circuit depth is:

$$\mathcal{O}(|L| \cdot |T| - |Z|) + \mathcal{O}(\lceil 1/\pi\epsilon_{abs} \rceil) = \mathcal{O}(|L| \cdot |T| - |Z| + 1/\epsilon_{abs})$$

The total qubit count necessary is $\mathcal{O}(\log_2(|L| \cdot |T| - |Z|))$ and the cumulative error is therefore $\mathcal{O}(\epsilon_{diag}) + \mathcal{O}(\epsilon_{QSVT}) = \mathcal{O}(\epsilon_{diag} + \epsilon_{QSVT})$. □

	Block Encoding of D	**QSVT Application**
Circuit Depth	$\mathcal{O}(\lvert L\rvert \cdot \lvert T\rvert - \lvert Z\rvert)$	$\mathcal{O}(\lceil 1/\pi\epsilon_{abs}\rceil))$
Qubit Count	$\mathcal{O}(\log_2(\lvert L\rvert \cdot \lvert T\rvert - \lvert Z\rvert))$	$\mathcal{O}(\log_2(\lvert L\rvert \cdot \lvert T\rvert - \lvert Z\rvert))$
Potential Errors	$\mathcal{O}(\epsilon_{BE})$	$\mathcal{O}(\epsilon_{QSVT2})$
Total Circuit Depth	$\mathcal{O}(\lvert L\rvert \cdot \lvert T\rvert - \lvert Z\rvert + \lceil 1/\pi\epsilon_{abs}\rceil)$	
Total Qubit Count	$\mathcal{O}(\log_2(\lvert L\rvert \cdot \lvert T\rvert - \lvert Z\rvert))$	
Total Potential Errors	$\mathcal{O}(\epsilon_{diag} + \epsilon_{QSVT2})$	

Table 5.4: Complexity Analysis for QSVT Absolute Value Calculation where ϵ_{diag} is the error from the diagonal Block Encoding

5.1.5 Complexity Analysis of the Quantum Amplitude Estimation

The following is the Complexity Analysis for Quantum Amplitude Estimation (see Section 4.7).

Theorem 5.5 (Complexity Analysis of QAE for $\langle |P_{l,t}|, C_l\rangle$)**.** *The QAE for calculating $\langle |P_{l,t}|, C_l\rangle$ has the following complexity:*

- *Circuit depth:* $\mathcal{O}(1/\epsilon_{QAE})$
- *Qubit count:* $\mathcal{O}(\log_2(|G| \cdot |T| \cdot q))$
- *Cumulative error:* $\mathcal{O}(\epsilon_{QAE})$

where $\epsilon_{QAE} > 0$ is the desired precision.

Proof. We prove this by analyzing the steps involved in the QAE algorithm. For the circuit depth, the most significant contribution comes from the Grover operator iterations. Each iteration involves applying the unitary $\mathcal{A}$, the diffusion operator U_0, the inverse of the unitary $\mathcal{A}^\dagger$, and the phase flip operator U_{ψ_1}. The circuit depth for each Grover iteration is therefore dominated by the depth of $\mathcal{A}$ and $\mathcal{A}^\dagger$, which is $\mathcal{O}(1/\epsilon_{QAE})$. For the qubit count, the number of qubits required to represent the states $|P_{l,t}\rangle$ and $|C_l\rangle$ is determined by the size of the problem. Specifically, we need $n = |G|\cdot|T|\cdot q$ qubits where q is the number of qubits needed per value. Thus, the total qubit count is $\mathcal{O}(\log_2(|G|\cdot|T|\cdot q))$. The cumulative error of the QAE algorithm is $\mathcal{O}(\epsilon_{QAE})$, which is determined by the precision $\epsilon_{QAE} > 0$ required for the scalar product estimation. □

	QAE Application
Circuit Depth	$\mathcal{O}(1/\epsilon_{QAE})$
Qubit Count	$\mathcal{O}(\log_2(\lvert G\rvert \cdot \lvert T\rvert \cdot q))$
Potential Errors	$\mathcal{O}(\epsilon_{QAE})$
Total Circuit Depth	$\mathcal{O}(1/\epsilon_{QAE})$
Total Qubit Count	$\mathcal{O}(\log_2(\lvert G\rvert \cdot \lvert T\rvert \cdot q))$
Total Potential Errors	$\mathcal{O}(\epsilon_{QAE})$

Table 5.5: Complexity Analysis for QAE Absolute Value Calculation where $\epsilon_{QAE} > 0$ is the desired precision.

5.1.6 Complexity Analysis of the QAOA Application

This section provides a detailed complexity analysis of QAOA for optimizing the cost Hamiltonian H_C, focusing on the quantum circuit depth, qubit requirements, and classical optimization overhead.

Theorem 5.6 (Complexity Analysis of the QAOA for Optimizing Cost Hamiltonian)**.** *The QAOA for optimizing the cost Hamiltonian H_C has the following complexity:*

- *Circuit depth:* $\mathcal{O}(p \cdot d_{H_C} + p \cdot d_{H_B})$
- *Qubit count:* $\mathcal{O}(|G| \cdot |T| + q_P \cdot |G| \cdot |T| + n_{P_l})$
- *Classical optimization complexity:* $\mathcal{O}(ClassicalOpt(\vec{\gamma}, \vec{\beta}))$

where p is the number of QAOA layers, d_{H_C} is the depth of the cost Hamiltonian, d_{H_B} is the depth of the mixer Hamiltonian, and $ClassicalOpt(\vec{\gamma}, \vec{\beta})$ is the complexity of the classical optimization algorithm.

Proof. We analyze the complexity in terms of circuit depth, qubit count, and considering the classical optimization needed for optimizing the parameters. The QAOA circuit consists of $p \in \mathbb{N}$ layers, each comprising one application of the cost Hamiltonian H_C and one application of the mixer Hamiltonian H_B. Therefore, the total circuit depth is $\mathcal{O}(p \cdot d_{H_C} + p \cdot d_{H_B})$ where d_{H_C} and d_{H_B} represent the depths of applying the cost and mixer Hamiltonians. The qubit count is determined by the number of generators $|G|$, time steps $|T|$, and the precision q_P used for power output variables. The total number of qubits required thus is $\mathcal{O}(|G| \cdot |T| + q_P \cdot |G| \cdot |T| + n_{P_l})$, where n_{P_l} is from the amplitude encoding of the power flow variables (see Remark 4.37). The classical optimization algorithm adjusts the parameters $\vec{\gamma}$ and $\vec{\beta}$ to minimize the expectation value of the cost Hamiltonian. The complexity of this step depends on the specific algorithm used. Generally, if the optimization involves evaluating the cost function multiple times, it has a complexity of $\mathcal{O}(\text{ClassicalOpt}(\vec{\gamma}, \vec{\beta}))$, dependent on the algorithm used for optimization. □

	QAOA Application
Circuit Depth	$\mathcal{O}(p \cdot d_{H_C} + p \cdot d_{H_B})$
Qubit Count	$\mathcal{O}(\lvert G\rvert \cdot \lvert T\rvert + q_P \cdot \lvert G\rvert \cdot \lvert T\rvert + n_{P_l})$
Classical Optimization Complexity	$\mathcal{O}(\text{ClassicalOpt}(\vec{\gamma}, \vec{\beta}))$
Total Circuit Depth	$\mathcal{O}(p \cdot d_{H_C} + p \cdot d_{H_B})$
Total Qubit Count	$\mathcal{O}(\lvert G\rvert \cdot \lvert T\rvert + q_P \cdot \lvert G\rvert \cdot \lvert T\rvert + n_{P_l})$
Classical Optimization Complexity	$\mathcal{O}(\text{ClassicalOpt}(\vec{\gamma}, \vec{\beta}))$

Table 5.6: Complexity Analysis for a QAOA Application where p is the number of QAOA layers, d_{H_C} and d_{H_B} are the depths of the cost and mixer Hamiltonians, and $\text{ClassicalOpt}(\vec{\gamma}, \vec{\beta})$ is the complexity of the classical optimization algorithm.

5.2 Combined Complexity Analysis of the Quantum Algorithm

The following is the complexity analysis of the entire quantum algorithm solving the UCP (see Def. 4.2), excluding the classical optimization, as it has no influence on the quantum part of the algorithm.

Theorem 5.7 (Complexity Analysis of the Entire Quantum Algorithm for UCP)**.** *The overall complexity of the quantum algorithm for solving the Unit Commitment Problem (UCP) through state preparation, LCU block encoding, QSVT, QAE, and QAOA has the following combined complexity. The Circuit depth for the entire algorithm is:*

$$\begin{aligned}\mathcal{O}(&\log_2(|B|\cdot|T|)+\log_2(|G|+1)+\log_2(m)+Kn\\&+\kappa\log(\kappa/\epsilon_{MI})+|L|\cdot|T|-|Z|\\&+1/\epsilon_{abs}+1/\epsilon_{QAE}+p\cdot d_{H_C}+p\cdot d_{H_B})\end{aligned}$$

The Qubit count for the entire algorithm is:

$$\begin{aligned}\mathcal{O}(&\log_2(|B|\cdot|T|)+\log_2(m)+\log_2(|G|+1)+\log_2(K)\\&+\log_2(|B|\cdot|T|)+\log_2(|L|\cdot|T|-|Z|)\\&+|G|\cdot|T|+q_P\cdot|G|\cdot|T|+n_{P_l})\end{aligned}$$

The Cumulative error for the entire algorithm is:

$$\begin{aligned}\mathcal{O}(&\log_2(|B|\cdot|T|)\epsilon_{UCU}+\log_2(m)\epsilon_{UCU}+K(2\epsilon_{PREP}+n\epsilon_{SEL})\\&+\epsilon_{LCU}+\epsilon_{QSVT1}+\epsilon_{diag}+\epsilon_{QSVT2}+\epsilon_{QAE})\end{aligned}$$

where the terms ϵ_{UCU}, ϵ_{PREP}, ϵ_{SEL}, ϵ_{MI}, ϵ_{abs}, ϵ_{diag}, ϵ_{QSVT1}, ϵ_{QSVT2}, and ϵ_{QAE} represent the errors associated with the respective operations.

Proof. We analyze the combined complexity by summing up the contributions from each step in the algorithm Thm. 5.1, Thm. 5.2, Thm. 5.3, Thm. 5.4, Thm. 5.5 and Thm. 5.6. By combining these complexities, we obtain the stated circuit depth, qubit count, and cumulative error for the entire quantum algorithm. □

The thorough analysis of the susceptance matrix can be found in Section 2.2. Multiple conditions are presented under which the matrix and its condition number satisfy the following:

Corollary 5.1 (Best-Case Scenario for Overall Complexity)**.** *In the best-case scenario, the complexity of the quantum algorithm for UCP is significantly reduced. We make the following assumptions:*

- *The matrix $\mathbf{B}'$ is highly structured, allowing $K = \mathcal{O}(|B|\cdot|T|)$.*
- *The condition number κ of $\mathbf{B}'$ is low, leading to $\kappa = \mathcal{O}(1)$.*
- *The error terms are minimal or negligible.*
- *The power flow variables $P_{l,t}$ have minimal zero entries, leading to $|Z| \approx 0$.*

Under these assumptions, the best-case complexity is:

- *Circuit depth:* $\mathcal{O}(\log_2(|B|\cdot|T|)+\log_2(|G|+1)+\log_2(m)+|B|^2\cdot|T|^2+1/\epsilon_{MI}+|L|\cdot|T|+1/\epsilon_{abs}+1/\epsilon_{QAE}+p\cdot d_{H_C}+p\cdot d_{H_B})$
- *Qubit count:* $\mathcal{O}(\log_2(|B|\cdot|T|)+\log_2(m)+\log_2(|G|+1)+\log_2(|B|\cdot|T|)+\log_2(|B|\cdot|T|)+\log_2(|L|\cdot|T|)+|G|\cdot|T|+q_P\cdot|G|\cdot|T|+n_{P_l})$

Proof. The corollary follows from Theorem 5.7 under the given assumptions. In the best-case scenario, the structured nature of $\mathbf{B}'$ allows for $K=\mathcal{O}(|B|\cdot|T|)$, reducing the complexity of the LCU block encoding. With a low condition number $\kappa=\mathcal{O}(1)$ (see Section 2.2), the depth of QSVT for matrix inversion becomes $\mathcal{O}(\log(1/\epsilon_{MI}))$. Minimal zero entries in $P_{l,t}$ lead to $|Z|\approx 0$, simplifying the block encoding for D. □

Corollary 5.2 (Worst-Case Scenario for Overall Complexity)**.** *In the worst-case scenario, the complexity of the quantum algorithm for UCP is maximized. We make the following assumptions:*

- *The matrix* $\mathbf{B}'$ *is unstructured, leading to* $K=\mathcal{O}((|B|\cdot|T|)^2)$.
- *The condition number* κ *of* $\mathbf{B}'$ *is high.*
- *The error terms are significant.*
- *The power flow variables* $P_{l,t}$ *have many zero entries, leading to* $|Z|\approx|L|\cdot|T|$.

Under these assumptions, the worst-case complexity is:

- *Circuit depth:* $\mathcal{O}(\log_2(|B|\cdot|T|)+\log_2(|G|+1)+\log_2(m)+(|B|\cdot|T|)^3+\kappa\log(\kappa/\epsilon_{MI})+|L|\cdot|T|+1/\epsilon_{abs}+1/\epsilon_{QAE}+p\cdot d_{H_C}+p\cdot d_{H_B})$
- *Qubit count:* $\mathcal{O}(\log_2(|B|\cdot|T|)+\log_2(m)+\log_2(|G|+1)+2\log_2(|B|\cdot|T|)+\log_2(|L|\cdot|T|)+|G|\cdot|T|+q_P\cdot|G|\cdot|T|+n_{P_l})$

Proof. The corollary follows from Theorem 5.7 under the given assumptions. In the worst-case scenario, the unstructured nature of $\mathbf{B}'$ leads to $K=\mathcal{O}((|B|\cdot|T|)^2)$, significantly increasing the complexity of the LCU block encoding. A high condition number κ results in a deeper QSVT circuit. Significant error terms further complicate the algorithm. Many zero entries in $P_{l,t}$ lead to $|Z|\approx|L|\cdot|T|$, increasing the complexity of the block encoding for D. □

The following provides a simplified complexity analysis that abstracts away quantum-specific details, focusing instead on the primary factors that influence the algorithm's performance.

Corollary 5.3 (Simplified Best-Case Complexity with Fixed Precision)**.** *Assume the following conditions:*

- *The matrix* $\mathbf{B}'$ *is well-structured, minimizing decomposition complexity.*
- *The condition number* κ *of* $\mathbf{B}'$ *is low, ensuring efficient matrix inversion.*
- *All precision-related parameters are set to a fixed machine precision* $\epsilon>0$*, such as* $\epsilon:=10^{-6}$.

Under these assumptions, the complexity of the quantum algorithm for solving the UCP is given by:

- *Circuit depth:* $\mathcal{O}\left(\log_2(|B|\cdot|T|)+|B|^2\cdot|T|^2\right)$
- *Qubit Count:* $\mathcal{O}\left(\log_2(|B|\cdot|T|)+|G|\cdot|T|\right)$

Proof. The corollary follows from Theorem 5.7 under the given assumptions and from simplifying the detailed complexity analysis by focusing on the dominant factors and with fixed ϵ:

- Circuit Depth: The depth primarily depends on $\log_2(|B| \cdot |T|)$ due to the number of buses and time steps. The quadratic term $|B|^2 \cdot |T|^2$ reflects the complexity of operations related to the system size. The precision term $\frac{1}{\epsilon}$, which accounts for the machine precision required in quantum operations.
- Qubit Count: The qubit requirements scale logarithmically with the system size $\log_2(|B| \cdot |T|)$ and linearly with the number of generators and time steps $|G| \cdot |T|$.

□

	Total Complexity
Circuit Depth	$\mathcal{O}(\log_2(\lvert B\rvert \cdot \lvert T\rvert) + \log_2(\lvert G\rvert + 1) + \log_2(m) + Kn$ $+\kappa \log(\kappa/\epsilon_{MI}) + \lvert L\rvert \cdot \lvert T\rvert - \lvert Z\rvert$ $+\frac{1}{\epsilon_{abs}} + \frac{1}{\epsilon_{QAE}} + p \cdot d_{H_C} + p \cdot d_{H_B})$
Qubit Count	$\mathcal{O}(\log_2(\lvert B\rvert \cdot \lvert T\rvert) + \log_2(m) + \log_2(\lvert G\rvert + 1)$ $+ \log_2(K) + \log_2(\lvert B\rvert \cdot \lvert T\rvert)$ $+ \log_2(\lvert L\rvert \cdot \lvert T\rvert - \lvert Z\rvert) + \lvert G\rvert \cdot \lvert T\rvert$ $+q_P \cdot \lvert G\rvert \cdot \lvert T\rvert + n_{P_l})$
Cumulative Error	$\mathcal{O}(\log_2(\lvert B\rvert \cdot \lvert T\rvert)\epsilon_{UCU} + \log_2(m)\epsilon_{UCU}$ $+K(2\epsilon_{\text{PREP}} + n\epsilon_{\text{SEL}})$ $+\epsilon_{LCU} + \epsilon_{QSVT1} + \epsilon_{diag}$ $+\epsilon_{QSVT2} + \epsilon_{QAE})$

Table 5.7: Overall Complexity Analysis for the Entire Quantum Algorithm for UCP.

6 Discussion

The core objective of this research was to explore and develop a quantum simulation-based optimization (QuSO) approach for solving the Unit Commitment Problem (UCP), particularly within the context of energy grid management. By characterizing the UCP as a Mixed-Integer Nonlinear Programming (MINLP) problem and utilizing Quantum Singular Value Transformation (QSVT) and the Quantum Approximate Optimization Algorithm (QAOA), the work aimed to reduce the computational complexity traditionally associated with such problems. The application of graph theory in modeling the energy grid is particularly advantageous. Utilizing the susceptance matrix, derived from the Laplacian matrix, offers a powerful framework for analyzing the grid's structural properties. This approach leverages algebraic connectivity and eigenvalue bounds to achieve computational efficiency, especially in large-scale grids. However, the reliance on the DC power flow approximation, while necessary for tractability, introduces a limitation compared to more accurate AC power flow models. QSVT plays a pivotal role by enabling efficient matrix inversion, crucial for optimizing the UCP. The polynomial approximation of the inversion process, particularly through Chebyshev polynomials, ensures both accuracy and efficiency. The exponential speedup in time complexity provided by QSVT marks a significant advancement. However, the success of QSVT depends on the condition number of the matrix being inverted. While bounds are provided, real-world scenarios might present challenges if these bounds are not tight enough. Furthermore, QSVT's efficiency is closely tied to the effectiveness of the block encoding within the quantum circuits. An efficient block encoding is essential, as it directly impacts matrix inversion and other linear algebra operations central to UCP optimization. If the block encoding is not optimized, the advantages of QSVT could be reduced, leading to increased circuit depth and reduced accuracy, which are critical concerns on Noisy Intermediate-Scale Quantum (NISQ) devices [Pre18]. The integration of QAOA with the quantum-constructed cost function offers a novel hybrid quantum-classical approach to solving the UCP. Representing generator states using the Ising model and modeling constraints as penalties in the cost Hamiltonian within the QAOA framework is an effective method that leverages the discrete nature of the UCP. However, the effectiveness of QAOA is highly dependent on the initial parameterization and quantum circuit depth, which may pose challenges for near-term implementations on quantum devices.

7 Conclusion

This research advances the theoretical foundations of quantum computing applied to energy grid optimization. The exponential reduction in computational complexity achieved through QSVT and the potential of QAOA to address non-linear, non-convex optimization problems are significant contributions. These findings suggest that quantum computing can theoretically provide a viable path toward solving large-scale UCPs more efficiently than classical methods, potentially enabling real-time optimization in complex energy grids. From a practical standpoint, optimizing unit commitment in real-time would substantially improve operational efficiency and reliability, especially in grids with high renewable energy penetration. However, the transition from theoretical models to practical applications requires overcoming significant challenges, including the limitations of current quantum hardware and the need for more robust algorithms capable of handling real-world complexities. Future work should focus on addressing these challenges by exploring more accurate AC power flow models, even if this increases complexity. Additionally, optimizing the quantum algorithms, particularly the block encoding in QSVT and the parameterization of QAOA, is crucial. Given the current limitations of NISQ devices, research into error mitigation techniques and more efficient quantum circuit designs is necessary. Finally, collaboration with industry partners and testing on actual power grid data will be essential in validating these algorithms in real-world scenarios. The future of quantum computing in energy grid management is promising, but realizing its full potential will require continued innovation in both algorithm design and quantum hardware development.

List of Algorithms

List of Figures

List of Tables

Bibliography

[AC22] Paweł Albrechtowicz and Piotr Cisek. An impact of the line resistance on the power flow calculations with installed phase-shifting transformer in different voltage levels power systems. *Electric Power Systems Research*, 209:107970, 2022.

[AI21] Fahad Saleh Al-Ismail. Dc microgrid planning, operation, and control: A comprehensive review. *IEEE Access*, 9:36154–36172, 2021.

[Alo22] Juan M. Alonso. Metric spaces and sparse graphs, 2022.

[Aly21] Ammar Alyousef. *E-Mobility Management: Towards a Grid-friendly Smart Charging Solution*. PhD thesis, Universität Passau, 2021.

[AW05] George B. Arfken and Hans J. Weber. *Mathematical Methods for Physicists*. Elsevier Academic Press, 6th edition, 2005.

[AY19] Akshay Ajagekar and Fengqi You. Quantum computing for energy systems optimization: Challenges and opportunities. *Energy*, 179:76–89, July 2019.

[BHMT02] Gilles Brassard, Peter Høyer, Michele Mosca, and Alain Tapp. Quantum amplitude amplification and estimation, 2002.

[Bis21] James Bisgard. *Analysis and Linear Algebra: The Singular Value Decomposition and Applications*, volume 94 of *Student Mathematical Library*. American Mathematical Society, Central Washington University, Ellensburg, WA, 2021.

[BKL+13] Pietro Belotti, Christian Kirches, Sven Leyffer, Jeff Linderoth, James Luedtke, and Ashutosh Mahajan. Mixed-integer nonlinear optimization. *Acta Numerica*, 22:1–131, 2013.

[Boy00] John P. Boyd. *Chebyshev and Fourier Spectral Methods*. Dover Publications, Inc., second edition, 2000.

[Boy23] Gregory Boyd. Low-overhead parallelisation of lcu via commuting operators. *arXiv preprint arXiv:2312.00696*, 2023.

[BP24] Lucas Böttcher and Mason A. Porter. Complex networks with complex weights. *Physical Review E*, 109(2):024314, 2024.

[Car62] J. Carpentier. Contribution a l'etude du dispatching economique. *Bulletin de la Societe Francaise des Electriciens*, 8:431–447, Aug 1962.

[Che66] Elliott Ward Cheney. *Introduction to Approximation Theory*. International Series in Pure and Applied Mathematics. McGraw-Hill Book Company, 1966. Original from University of Michigan, Digitized 20 Nov. 2007.

[Chr] Richard D. Christie. Ieee 118-bus test case. `https://labs.ece.uw.edu/pstca/pf118/pg_tca118bus.htm`. Accessed: 2024-07-18.

[Chu97] Fan R. K. Chung. *Spectral Graph Theory*, volume 92. American Mathematical Society, Providence, RI, 1997.

[Chu07] Fan R. K. Chung. Four proofs for the cheeger inequality and graph partition algorithms. *Mathematics, Computer Science*, 2007.

[CK24] Muddu Chethan and Ravi Kuppan. A review of facts device implementation in power systems using optimization techniques. *Journal of Engineering and Applied Science*, 71(1):18, 2024.

[CKG16] Yongxin Chen, Sei Zhen Khong, and Tryphon T. Georgiou. On the definiteness of graph laplacians with negative weights: Geometrical and passivity-based approaches. In *2016 American Control Conference (ACC)*, pages 2488–2493, 2016.

[Con24] Claudio Conti. *Quantum Machine Learning: Thinking and Exploration in Neural Network Models for Quantum Science and Quantum Computing*. Quantum Science and Technology. Springer, New York, NY, USA, 2024.

[CR90] Badrul Chowdhury and Saifur Rahman. A review of recent advances in economic dispatch. *Power Systems, IEEE Transactions on*, 5:1248 – 1259, 12 1990.

[CSSDS+15] Lucas Cuadra, Sancho Salcedo-Sanz, Javier Del Ser, Silvia Jiménez-Fernández, and Zong Woo Geem. A critical review of robustness in power grids using complex networks concepts. *Energies*, 8(9):9211–9265, 2015.

[CW12a] Andrew M. Childs and Nathan Wiebe. Hamiltonian Simulation Using Linear Combinations of Unitary Operations. *Quantum Information and Computation*, 12:901–924, 2012.

[CW12b] Andrew M. Childs and Nathan Wiebe. Hamiltonian simulation using linear combinations of unitary operations. *Quantum Information and Computation*, 12(11-12):901–924, 2012.

[dBBDD14] Kenneth Van den Bergh, Kenneth Bruninx, Erik Delarue, and William D'haeseleer. A mixed-integer linear formulation of the unit commitment problem. Technical Report WP EN2014-07, TME Working Paper - Energy and Environment, 2014. Last update: February 2014. An electronic version of the paper may be downloaded from the TME website: `http://www.mech.kuleuven.be/tme/research/`.

[dBDD14] Kenneth Van den Bergh, Erik Delarue, and William D'haeseleer. Dc power flow in unit commitment models. Technical Report WP EN2014-12, TME Working Paper - Energy and Environment, 2014. Last update: May 2014. An electronic version of the paper may be downloaded from the TME website: http://www.mech.kuleuven.be/tme/research/.

[Die17] Reinhard Diestel. *Graph Theory*. Graduate Texts in Mathematics. Springer Berlin, Heidelberg, 5 edition, 2017.

[DKS22] Ishanki De Mel, Oleksiy V. Klymenko, and Michael Short. Balancing accuracy and complexity in optimisation models of distributed energy systems and microgrids with optimal power flow: A review. *Sustainable Energy Technologies and Assessments*, 52:102066, 2022.

[DMWL21] Yulong Dong, Xiang Meng, K. Birgitta Whaley, and Lin Lin. Efficient phase-factor evaluation in quantum signal processing. *Phys. Rev. A*, 103:042419, Apr 2021.

[Eva22] Lawrence C. Evans. *Partial Differential Equations*, volume 19 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, Rhode Island, 2 edition, 2022.

[FGG14] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm, 2014.

[FH19] Edward Farhi and Aram W Harrow. Quantum supremacy through the quantum approximate optimization algorithm, 2019.

[Fie73] Miroslav Fiedler. Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, 23(2):298–305, 1973.

[Gai03] Zwe-Lee Gaing. Gaing, z.-l.: Particle swarm optimization to solving the economic dispatch considering the generator constraints. ieee tans. on power syst. 18(3), 1187-1195. *Power Systems, IEEE Transactions on*, 18:1187 – 1195, 09 2003.

[Gri12a] Robert B. Griffiths. Hilbert space quantum mechanics, 2012. Lecture notes for course qmd111, Version of 20 August 2012.

[Gri12b] Leonard L. Grigsby, editor. *Electric Power Generation, Transmission, and Distribution*. CRC Press, 3 edition, 6 2012.

[GSLW18] András Gilyén, Yuan Su, Guang Hao Low, and Nathan Wiebe. Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics. *arXiv preprint arXiv:1806.01838*, June 2018.

[GSO17] J. Duncan Glover, Mulukutla S. Sarma, and Thomas Overbye. *Power System Analysis and Design*. Cengage Learning, 6th edition, 2017.

[HG91] M. Huneault and F.D. Galiana. A survey of the optimal power flow literature. *IEEE Transactions on Power Systems*, 6(2):762–770, 1991.

[HHL09] Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.*, 103:150502, Oct 2009.

[HHPT23] Pascal Halffmann, Patrick Holzer, Kai Plociennik, and Michael Trebing. *A Quantum Computing Approach for the Unit Commitment Problem*, page 113–120. Springer International Publishing, 2023.

[HJ13] Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, Cambridge, New York, Melbourne, Madrid, Cape Town, Singapore, Sao Paulo, Delhi, Mexico City, second edition, 2013.

[HRP24] Morad Halihal, Tirza Routtenberg, and H. Vincent Poor. Estimation of complex-valued laplacian matrices for topology identification in power systems. *arXiv preprint arXiv:2308.03392*, 2024.

[HZZG22] Jiangwei Hou, Qiaozhu Zhai, Yuzhou Zhou, and Xiaohong Guan. A fast solution method for large-scale unit commitment based on lagrangian relaxation and dynamic programming, 2022.

[Int23] International Energy Agency. *World Energy Outlook 2023*. OECD Publishing, 2023.

[JPJL10] Yun-Won Jeong, Jong-Bae Park, Se-Hwan Jang, and Kwang Y. Lee. A new quantum-inspired binary pso: Application to unit commitment problems for power systems. *IEEE Transactions on Power Systems*, 25(3):1486–1495, 2010.

[KGB+21] Samantha Koretsky, Pranav Gokhale, Jonathan M. Baker, Joshua Viszlai, Honghao Zheng, Niroj Gurung, Ryan Burg, Esa Aleksi Paaso, Amin Khodaei, Rozhin Eskandarpour, and Frederic T. Chong. Adapting quantum approximation optimization algorithm (qaoa) for unit commitment, 2021.

[KM78] Ravindran Kannan and Clyde L. Monma. On the computational complexity of integer programming problems. In *Optimization and Operations Research*, Berlin, Heidelberg, 1978. Springer Berlin Heidelberg.

[Leb72] N.N. Lebedev. *Special Functions and Their Applications*. Dover Publications, 1972.

[LHF24] Yidong Liao, Min-Hsiu Hsieh, and Chris Ferrie. Quantum optimization for training quantum neural networks. *Quantum Machine Intelligence*, 6(1), June 2024.

[LL01] Elliott H. Lieb and Michael Loss. *Analysis: Second Edition*, volume 14 of *Graduate Studies in Mathematics*. American Mathematical Society, Princeton, NJ, 2001.

[LN89] Dong C. Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1):503–528, 1989.

[Low17] Guang Hao Low. *Quantum Signal Processing by Single-Qubit Dynamics*. Ph.d. thesis, Massachusetts Institute of Technology, Massachusetts, USA, 2017.

[LSM17] Samuel J. Ling, Jeff Sanny, and William Moebs. *University Physics Volume 2*. OpenStax, 2017.

[Luc14] Andrew Lucas. Ising formulations of many np problems. *Frontiers in Physics*, 2, 2014.

[MER21] German Morales-España and Andres Ramos. Challenges and solutions in the integration of renewable energy sources in the electricity grid. *Renewable Energy*, 163:324–333, 2021.

[MH19] Daniel K. Molzahn and Ian A. Hiskens. *A Survey of Relaxations and Approximations of the Power Flow Equations*. Now Foundations and Trends, 2019.

[MKT+19] R. Muzzammel, I. Khail, M.H. Tariq, A.M. Asghar, and A. Hassan. Design and power flow analysis of electrical system using electrical transient and program software. *Energy and Power Engineering*, 11:186–199, 2019.

[Moh91] Bojan Mohar. Eigenvalues, diameter, and mean distance in graphs. *Graphs and Combinatorics*, 7:53–64, 1991.

[MRTC21] John M. Martyn, Zane M. Rossi, Andrew K. Tan, and Isaac L. Chuang. Grand unification of quantum algorithms. *PRX Quantum*, 2(4), December 2021.

[MVBS04] Mikko Mottonen, Juha J. Vartiainen, Ville Bergholm, and Martti M. Salomaa. Transformation of quantum states using uniformly controlled rotations, 2004.

[NC10] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2010.

[NS14] T. K. Nagsarkar and M. S. Sukhija. *Power System Analysis*. Oxford University Press, 2 edition, 2014.

[NZB22] Nima Nikmehr, Peng Zhang, and Mikhail A. Bragin. Quantum distributed unit commitment: An application in microgrids. *IEEE Transactions on Power Systems*, 37(5):3592–3603, 2022.

[P+21] Narayana Padhy et al. A review of optimization techniques applied to unit commitment in modern power systems. *IEEE Transactions on Power Systems*, 36(5):3896–3908, 2021.

[PPC23] Vitor Fernão Pires, Armando Pires, and Armando Cordeiro. Dc microgrids: Benefits, architectures, perspectives and challenges. *Energies*, 16(3), 2023.

[Pre18] John Preskill. Quantum computing in the nisq era and beyond. *Quantum*, 2:79, August 2018.

[PTVF07] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes: The Art of Scientific Computing*. Cambridge University Press, Cambridge, New York, Melbourne, Madrid, Cape Town, Singapore, Sao Paulo, third edition, 2007.

[RB20] Janosh Riebesell and Stefan Bringuier. Collection of standalone tikz images, 2020. Version 0.1.0, DOI: 10.5281/zenodo.7486911.

[Rem34] E. Remez. Sur le calcul effectif des polynomes d'approximation de tchebichef. *CR Acad. Sci. Paris*, 199:337–340, 1934.

[RR23] Arthur G. Rattew and Patrick Rebentrost. Non-linear transformations of quantum amplitudes: Exponential improvement, generalization, and applications, 2023.

[Sau13] Timothy Sauer. *Numerical Analysis*. Pearson Deutschland GmbH, Germany, 2 edition, 2013.

[SB02] Josef Stoer and Roland Bulirsch. *Introduction to Numerical Analysis*, volume 12 of *Texts in Applied Mathematics*. Springer, 3rd edition, 2002.

[SP18] Maria Schuld and Francesco Petruccione. *Supervised Learning with Quantum Computers*. Quantum Science and Technology. Springer Cham, 1 edition, 2018. Physics and Astronomy.

[Spi19] Daniel A. Spielman. Spectral and algebraic graph theory. Yale University Website, 2019. Incomplete Draft.

[SR+22] David I Stern, Arthur J Ragauskas, et al. The global energy crisis: impacts and responses. *Energy Policy*, 144:111710, 2022.

[SS05] Elias M. Stein and Rami Shakarchi. *Real Analysis: Measure Theory, Integration, and Hilbert Spaces*, volume 3 of *Princeton Lectures in Analysis*. Princeton University Press, Princeton and Oxford, 2005. Printed on acid-free paper.

[ST94] J. J. Sakurai and San Fu Tuan. *Modern Quantum Mechanics*. Addison-Wesley series in advanced physics. Addison-Wesley, Reading, Massachusetts, revised edition, 1994. Library of Congress Cataloging-in-Publication Data: 93-17803 CIP.

[Tre19] Lloyd N. Trefethen. *Approximation Theory and Approximation Practice, Extended Edition*. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2019.

[Uni] Texas A&M University. Ieee 118-bus system. `https://electricgrids.engr.tamu.edu/electric-grid-test-cases/ieee-118-bus-system/`. Accessed: 2024-07-18.

[Urb18] Luca Urbanucci. Limits and potentials of mixed integer linear programming methods for optimization of polygeneration energy systems. *Energy Procedia*, 148:1199–1205, 2018. ATI 2018 - 73rd Conference of the Italian Thermal Machines Engineering Association.

[vM06] Alexandra von Meier. *Electric Power Systems: A Conceptual Introduction*. Wiley-IEEE Press, Hoboken, NJ, 1st edition, 2006.

[Wad06] C.L. Wadhwa. *Basic Electrical Engineering*. New Age International, New Delhi, India, 4th edition, 2006.

[WKKVM15] Xiangrong Wang, Yakup Koç, Robert E. Kooij, and Piet Van Mieghem. A network approach for power grid robustness against cascading failures. In *2015 7th International Workshop on Reliable Networks Design and Modeling (RNDM)*, pages 208–214, 2015.

[WWS13] A.J. Wood, B.F. Wollenberg, and G.B. Sheblé. *Power Generation, Operation, and Control*. Wiley, 2013.

[YZ23] Pei Yuan and Shengyu Zhang. Optimal (controlled) quantum state preparation and improved unitary synthesis by quantum circuits with any number of ancillary qubits. *Quantum*, 7:956, March 2023.

[ZSMRZP21] Fco. Javier Zarco-Soto, José L. Martínez-Ramos, and Pedro J. Zarco-Periñán. Chapter 7 - voltage control in active distribution networks. In P. Sanjeevikumar, C. Sharmeela, Jens Bo Holm-Nielsen, and P. Sivaraman, editors, *Power Quality in Modern Power Systems*, pages 193–217. Academic Press, 2021.

[ZWC+20] Leo Zhou, Sheng-Tao Wang, Soonwon Choi, Hannes Pichler, and Mikhail D. Lukin. Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices. *Phys. Rev. X*, 10:021067, Jun 2020.