Quantum Singular Value Transformation Methods for Quantum Phase Estimation

Bachelor Thesis by
Vivian Sattler

submitted to the
Institute of Theoretical Physics
Faculty of Mathematics and Physics
Gottfried Wilhelm Leibniz University Hannover

Examiner: Prof. Dr. Tobias J. Osborne
Supervisor: M. Sc. Shawn Skelton

December 14, 2023

Contents

1 Introduction

Quantum computation arose in the 1980's, exploring the possible advantages of building computational devices to execute algorithms using quantum systems [1]. However, it was not until 1994 that the discovery of Shor's algorithm constituted a milestone in showing how quantum computers allow significant (up to exponential) speedups over classical computers. It promised faster solutions to many computational problems with a wide range of applications [2] and therefore naturally sparked the interest of researchers and industry alike [3].

The field has blossomed tremendously in the past decades, both from the theoretical and the experimental point of view [4]. While quantum computers with few qubits have successfully performed factoring tasks in 2001 already [2, 5], quantum computers of today use superconducting qubits to operate on up to over 100 qubits [2, 6]. However, only large-scale quantum devices with orders of magnitudes more qubits promise to hold significant advantages over their classical counterparts [7]. Although it is still not clear to what extent one will be able to build viable quantum computers, the recent progress in overcoming experimental challenges gives hope that it might be possible to implement some quantum algorithms exceeding classical limitations in the foreseeable future [4].

At the same time, the vast research interest catalyzed the discovery of novel quantum algorithms and subroutines [4]. Just a handful of them form the building blocks for the majority of known quantum algorithms, namely quantum search, quantum phase estimation and Hamiltonian simulation. They do not exhibit structural similarities at first glance, but strikingly, they can all be formulated in terms of a framework called quantum singular value transformation (QSVT) [1]. Developed by Gilyén et al. in 2018, QSVT is a procedure that allows for the polynomial transformation of a non-unitary matrix contained within a bigger unitary operator. Because the set of achievable polynomials is quite broad, QSVT can be applied in numerous scenarios. The resulting algorithms have appealing properties, such as being "conceptually simple and efficient" [8]. Since almost all quantum algorithms can be formulated in terms of QSVT, it is also referred to as "a grand unification of quantum algorithms"[1].

In this thesis, QSVT is presented while focusing on its application to a quantum phase estimation (QPE) algorithm proposed by Martyn et al. [1]. QPE forms a subroutine of many quantum algorithms, estimating the unknown phase φ of an eigenstate $|u\rangle$ of a unitary U where $U|u\rangle = e^{i2\pi\varphi}|u\rangle$. Along the way, we also survey the textbook approach to QPE in more detail.

After an introduction to the theory of quantum computation in chapter 2, chapter 3 lays the groundwork for QSVT. Next, we present QPE (chapter 4), followed by the discussion of the QSVT-based QPE algorithm in chapter 5. We conclude this thesis by analyzing the computational cost of the given algorithm (chapter 6) before summarizing our work in chapter 7.

2 Prerequisites

This chapter develops the prerequisites necessary to follow the thesis. Moving forward, it is assumed that the reader has been exposed to the basic principles of quantum theory and the corresponding linear algebra. Section 2.1 briefly introduces the classical circuit model of computation. Although we will not cover classical theory of computation in detail here, we use it to draw an analogy between classical and quantum computation to motivate the quantum circuit model, which is subject of the following chapter 2.2. Both sections 2.1 and 2.2 are adopted from the lecture [9], and based on the introductory chapters in [3], if not specified otherwise. Lastly, we introduce quantum computational cost analysis in chapter 2.3.

2.1 Universality Theorem

A widespread mathematical model of classical computation is the circuit model. It describes a computer as a Boolean circuit which is comprised of wires and logic gates. The wires carry the information within the circuit, while the gates process the information by performing operations on the wires that they are connected to. The term "Boolean" refers to the state of a wire taking either one of the values in $\{0, 1\}$, which is called a bit. Conveniently, the classical circuit model directly translates to the intuitive idea of computer hardware design using electrical wires and components, such as transistors manipulating the voltages that are applied to the wires. Known classical logic gates are for example AND, OR, XOR or NAND gates, which all perform certain logical operations on the wire states. More generally speaking, a logic gate is defined as a (vectorial) Boolean function $f : \{0,1\}^m \to \{0,1\}^n$ that operates on m input bits and gives a result on n output bits. Still, the NAND gate is of particular interest at this point because it is vital for the classical universality theorem: It states that the NAND gate is universal for classical computation, provided certain conditions apply.[1] This is a meaningful result, as it expresses that any logic gate on a fixed amount of bits can be implemented using only NAND gates. We will soon find that there exists a similar theorem in the quantum case.

We now transfer the classical circuit model to the quantum circuit model. In principle, the analogy is straightforwardly achieved by turning bits into quantum bits, and gates into quantum gates. Notwithstanding, we have to make sure that the quantum circuit model complies to the rules of quantum mechanics, adding some additional constraints.

To begin with, the fundamental building blocks of a quantum circuit in analogy to the bits are the quantum bits, also called qubits. A single qubit is an element of the Hilbert space $\mathbb{C}^2$. It can be written in terms of the basis $\{|0\rangle, |1\rangle\}$, called the computational basis, as

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\,, \tag{1}$$

where $\alpha, \beta \in \mathbb{C}$ satisfy the normalization condition $|\alpha|^2 + |\beta|^2 = 1$. Accordingly, one can also express a one-qubit state in vector representation by the tuple $(\alpha, \beta)^{\mathrm{T}}$. Identifying the possible states $\{|0\rangle, |1\rangle\}$ of a qubit with the possible bit values $\{0, 1\}$, it is easily observed that in contrast to in the classical case, a qubit can exist in a superposition of the two basis states. Measurement

[1] The conditions include the availability of the Fanout gate, which allows the generation of multiple identical copies of the input state, and the Crossover gate, interchanging the values of two bits. What is more, access to additional working bits and preparation of their initial values is required. Those bits are called ancilla bits, and their quantum equivalent is introduced in section 2.2. For the proof of the classical universality theorem, see [3].

of the qubit with respect to the computational basis results in a measurement statistic that gives $|0\rangle$ with a probability of $P_0 = |\alpha|^2$, and and $|1\rangle$ with a probability of $P_1 = |\beta|^2$, while the superposition collapses into the basis state corresponding to the measurement. Naturally, the state of an n-qubit system is described by an element of the tensor product Hilbert space $\left(\mathbb{C}^2\right)^{\otimes n}$. An arbitrary n-qubit computational basis state is often abbreviated as

$$|q_1\rangle \otimes |q_2\rangle \otimes ... |q_n\rangle = |q_1 q_2 ... q_n\rangle \tag{2}$$

for $q_i \in \{0, 1\} \, \forall i \in \{1, ...n\}$. Following the description of a qubit by equation (1), a multi-qubit state is hence represented as a vector as $(\alpha_1, \beta_1)^{\mathrm{T}} \otimes ... \otimes (\alpha_n, \beta_n)^{\mathrm{T}}$.

The quantum equivalent of a logical gate is called a quantum gate. In accordance with the postulates of quantum mechanics, the evolution of states in a closed quantum system is given by unitary operations ($U^{-1} = U^{\dagger}$) on the state space. A quantum gate acting on n qubits is thus a group element of $U(2^n)$ and can be represented as a unitary $2^n \times 2^n$-matrix with respect to the computational basis.

The remaining key elements of the quantum circuit model are the ability to prepare qubits in the computational basis states, as well as to perform measurements in the computational basis, and lastly, classical resources. All constituents combined form a quantum computer, also often referred to as a quantum device. In theory, a quantum device is able to perform all classical computations. However, it is often convenient to outsource some calculations to a classical device. This is why, practically, a quantum device also often contains a classical component, e.g. the instruments used to control the quantum system.

Now, after setting up an analogy between the classical and the quantum circuit model, it remains to show that the analogy succeeds. The argument needed for this is delivered by the universality theorem. It states that, like in the classical case, there exists a universal gate set in quantum computation. This means that any unitary operation can be approximated to arbitrary accuracy by a quantum circuit implementing solely gates from a given gate set. Therefore, the gate set is called universal, and its elements are the elementary gates. More specifically, the universality theorem states that single-qubit gates and CNOT gates are universal for quantum computation.[2] As a matter of fact, up to a constant factor, at most $n^2 4^n$ single-qubit gates and CNOT gates suffice to implement an arbitrary n-qubit unitary. The quantum universality theorem hence justifies working with the quantum circuit model, as it provides a way of implementing any of the infinitely many arbitrary unitary operations by decomposing it into a finite number of elementary gates. However, this decomposition can be quite complex and it is not unique. The choice of which elementary gates to use depends on the quantum circuit that one desires to execute as well as the available quantum device, determining which elementary gates are possible and, ideally, easy to implement.

Hereafter, the quantum circuit model is utilized as a mathematical abstraction for the quantum computer. The efficient experimental realization of the quantum circuits is a large area of research in itself which will not be covered in detail here.

[2] The quantum gates mentioned here are established in section 2.2.
Similar to the classical case, the universality theorem holds under the condition that access to a Crossover gate and ancilla state preparation is granted.

2.2 A Toolkit for Constructing Quantum Circuits

This chapter gives a short introduction to the basic elements of quantum circuits, starting with quantum wires that carry the qubits. Rather than a physical wire, the quantum wires often represent movement in time in between applying quantum gates. Within the set of quantum wires one distinguishes between the system register, where the quantum states of interested are stored and manipulated, and the ancilla register, containing the auxiliary qubits needed to perform the desired unitary operations on the system register [10].

In Figure 1, the most common single-qubit operators are listed. They are represented as unitary 2×2-matrices with respect to the computational basis, acting on one-qubit states in their vector representations. Next to the matrix representations of the gates are their respective circuit diagram depictions: In analogy to the circuit symbols of logical gates in an electrical circuit, quantum gates are drawn as boxes connected to horizontal lines depicting the quantum wires that they act on. They often contain a letter or expression to indicate the specific gate action. Quantum circuit diagrams are to be understood as executed in order from left to right, where vertically aligned gates on different qubits are run in parallel.

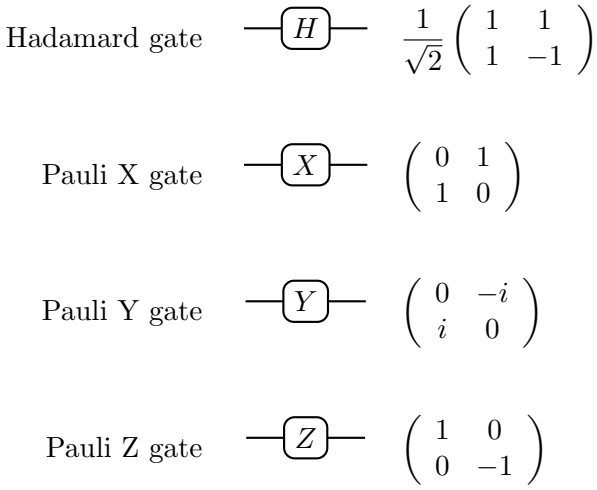


Figure 1: Circuit diagram symbols and matrix representations of prominent single-qubit quantum gates.

The Hadamard gate is often used when designing quantum algorithms, as it yields an equal superposition state when applied to the zero state:

$$H|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) =: |+\rangle . \tag{3}$$

And similarly:

$$H|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) =: |-\rangle . \tag{4}$$

These two resulting states form a different basis of the one-qubit Hilbert space called the Hadamard basis $\{|+\rangle, |-\rangle\}$. More generally, applying a Hadamard gate to n qubits in the zero state yields an equal superposition of all computational basis states $|k\rangle := |q_1 q_2 ... q_n\rangle$,

$$H^{\otimes n}|0\rangle^{\otimes n} = \frac{1}{\sqrt{2^n}} \sum_k |k\rangle . \tag{5}$$

The matrix representations of the Pauli gates are given by the Pauli matrices. The Pauli X gate is often referred to as the bit flip gate or quantum NOT gate, as it interchanges the coefficients of the basis states of the qubit. To investigate the action of a quantum gate on a quantum state, one can not only take their matrix-vector-product, but one can also express both in bra-ket notation. The second option is chosen here to illustrate the Pauli X gate:

$$X(\alpha\,|0\rangle + \beta\,|1\rangle) = (|1\rangle\langle 0| + |0\rangle\langle 1|)(\alpha\,|0\rangle + \beta\,|1\rangle) = \alpha\,|1\rangle + \beta\,|0\rangle\,. \tag{6}$$

The Pauli gates admit Pauli rotation operators by their exponentiation. Using the identity

$$e^{iAx} = \cos(x)\mathbb{1} + i\sin(x)A \tag{7}$$

which holds for all $x \in \mathbb{R}$ and any square matrix A fulfilling $A^2 = \mathbb{1}$, one defines the Pauli rotation gates:

$$\begin{aligned} R_x(\theta) &= e^{\frac{-i\theta X}{2}} = \begin{pmatrix} \cos(\frac{\theta}{2}) & -i\sin(\frac{\theta}{2}) \\ -i\sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{pmatrix} \\ R_y(\theta) &= e^{\frac{-i\theta Y}{2}} = \begin{pmatrix} \cos(\frac{\theta}{2}) & -\sin(\frac{\theta}{2}) \\ \sin(\frac{\theta}{2}) & \cos(\frac{\theta}{2}) \end{pmatrix} \\ R_z(\theta) &= e^{\frac{-i\theta Z}{2}} = \begin{pmatrix} e^{\frac{-i\theta}{2}} & 0 \\ 0 & e^{\frac{i\theta}{2}} \end{pmatrix}. \end{aligned} \tag{8}$$

An illustrative geometrical interpretation of a qubit and single-qubit operations is delivered by the Bloch sphere. Given the normalization condition $|\alpha|^2 + |\beta|^2 = 1$ of the single-qubit state in equation (1), one can exploit the properties of the sine and cosine functions to rewrite the one-qubit state as

$$|\psi\rangle = e^{i\gamma}\left(\cos(\theta/2)\,|0\rangle + e^{i\varphi}\sin(\theta/2)\,|1\rangle\right), \tag{9}$$

such that the parameters φ and θ can be identified with angles that determine a point on the two-dimensional unit sphere. As it does not affect the measurement statistics of the quantum state, the relative phase $e^{i\gamma}$ is often discarded. When applying the Pauli rotation gates to a qubit, the vector corresponding to the single-qubit state rotates around the Bloch sphere by the angle θ about the respective axis. In fact, every single-qubit gate can be expressed as a product of Pauli rotations up to a global phase, highlighting the importance of the Bloch sphere picture.

Another type of gate that will reappear in the subsequent algorithms is the controlled gate. It performs an operation on one or more target qubits dependent on the input state of one or more control qubits. The most prevalent controlled gate is the controlled-NOT gate, abbreviated as CNOT. Conditioned on one control qubit (here, the first qubit) being in the $|1\rangle$ state, it flips the bit of one target qubit:

$$\begin{aligned} &\mathrm{CNOT}(\alpha\,|00\rangle + \beta\,|01\rangle + \gamma\,|10\rangle + \delta\,|11\rangle) \\ &\qquad = (|0\rangle\langle 0| \otimes \mathbb{1} + |1\rangle\langle 1| \otimes X)(\alpha\,|00\rangle + \beta\,|01\rangle + \gamma\,|10\rangle + \delta\,|11\rangle) \\ &\qquad = \alpha\,|00\rangle + \beta\,|01\rangle + \gamma\,|11\rangle + \delta\,|10\rangle\,. \end{aligned} \tag{10}$$

The CNOT gate is visualized in Figure 2 below.

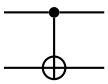
CNOT  $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$

Figure 2: Circuit diagram depiction and matrix representation of the CNOT gate. The dot marks the control qubit, the crossed circle the target qubit while indicating the NOT gate.

The notion of the controlled gate can be extended to arbitrary quantum gates manipulating multiple qubits. Generally speaking, if one assumes an $(n+k)$-qubit register and a quantum gate U acting on k qubits in a state $|\psi\rangle$, then an n-qubit-controlled-U gate is defined as:

$$C^n(U)\,|q_1 q_2 ... q_n\rangle\,|\psi\rangle = |q_1 q_2 ... q_n\rangle\,U^{q_1 q_2 ... q_n}\,|\psi\rangle\,. \tag{11}$$

The action of the gate is hence to apply U to $|\psi\rangle$ if all control qubits $|q_1\rangle\,, ... |q_n\rangle$ are in the $|1\rangle$ state, and to leave $|\psi\rangle$ unchanged otherwise. In circuit notation, this corresponds to Figure 3.

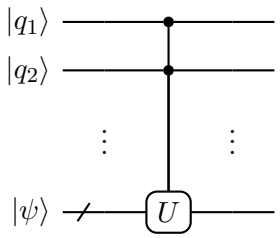


Figure 3: Circuit diagram depiction of a multi-qubit-controlled-U gate. The crossed quantum wire indicates multiple qubits drawn as one wire.

Lastly, when we refer to a measurement in the remainder of this thesis, we mean a measurement with respect to the computational basis. Its circuit depiction is as given in Figure 4:

Figure 4: Circuit diagram depiction of quantum measurement.

2.3 Quantum Computational Cost Analysis

An essential matter when developing a circuit on both classical and quantum computational devices is analyzing its cost. However, quantifying the cost of a quantum algorithm is non-trivial, since there exist multiple definitions of how it is measured. We briefly introduce them here, based on the discussion in [10].

Recall that the universality theorem ensures the existence of a decomposition of any quantum circuit into individual quantum gates of a universal gate set. Still, it is common that a quantum algorithm assumes access to some unitary operator U without or while only partially knowing the decomposition into elementary gates. To which extent we can trace U back to its full decomposition and the choice of the elementary gate set depends on the problem given. The remaining "black box"-operation is often referred to as the oracle. Counting only the number of oracle calls

within an algorithm gives its query complexity, a common measure of the computational cost of quantum algorithms. If the full decomposition of the circuit into elementary gates is known, their respective numbers yield the gate complexity, another cost measure. We do not define further what is meant by the cost of an elementary gate, which reflects that the efforts necessary for their experimental implementation depends on the quantum device. Treating the computational cost in this abstract manner has the advantage that one achieves a device-independent evaluation of computational cost. Since oracles can be difficult to implement, there is often a large disparity between query complexity and gate complexity. The query complexity thus provides a lower bound on the total gate complexity of the quantum algorithm.

Another possible way of analyzing the computational cost is by circuit depth, defined by "the maximum number of gates along any path from an input to an output" [10]. Since gates acting on different qubits can be carried out in parallel (i.e. simultaneously) in the quantum circuit model, the circuit depth acts as a measure of how much time the quantum device needs to run a certain algorithm. This reflects an important limitation of quantum devices: decoherence. Decoherence generally describes "processes corrupting the desired evolution of the system" [3]. It occurs because quantum devices can never be perfectly decoupled from their environment, and their interaction with it leads to the loss of the definite phase relations between quantum states. As these set phase relations are crucial for performing quantum computation, guaranteeing sufficiently long coherence times for given circuit depths imposes a serious constraint on quantum algorithm development [3]. Suppressing the effects of quantum decoherence is the objective of fault-tolerant quantum computation, a broad area of research that we will not elaborate on here. In the following, we will however rely on key results of fault-tolerant quantum computation and assume that error-correcting protocols have been applied, granting that all errors in our later discussion stem from approximations or the probabilistic nature of quantum measurement statistics [10]. Accordingly, we do not investigate circuit depth further, and instead use query complexity and gate complexity to analyze the computational cost of the quantum phase estimation algorithm presented later on.

3 Quantum Singular Value Transformation

The quantum singular value transformation is the extension of quantum signal processing (QSP), a technique first formulated by Low et al. in 2016 [11]. QSP intersperses rotations around different axes of the Bloch sphere of a single-qubit system, leading to a set of resulting composite gates that is "quite rich" [8]. By precisely manipulating the unitary $U(\theta)$ dependent on some signal parameter of interest θ, it is possible to tailor the measurement statistics reflecting the transition properties of a single-qubit system that is subject to the QSP gate sequence. This way, one can amplify the desired signal and reduce possible errors which hinder the control over the quantum system [11, 12]. Among the sources responsible for those errors are decoherence as well as systematic errors such as fluctuating experimental control parameters or fabricational deficiencies [13, 12]. Composite gates can help to mitigate these errors and even introduce new operations not achievable with single gates [14]. This furnishes a wide array of applications of QSP, ranging from Nuclear Magnetic Resonance Spectroscopy to quantum imaging, over quantum sensing to quantum computation [11]. Accordingly, many composite gate sequences have been designed [1], and even more are expected to be found by the evolving research on them [8].

Not long after the proposal of Low et al., Low and Chuang extended their findings beyond the single-qubit system by developing the idea of qubitization [15]. Combining QSP and qubitization, Gilyén et al. generalized their results establishing the quantum singular value transformation. QSVT applies a polynomial to the singular values of a matrix that is encoded within a bigger multi-qubit unitary operator, using the phase angle set from QSP [8]. It has been found that the vast majority of known quantum algorithms can be formulated in terms of QSVT [1], often giving rise to efficient and conceptually simple algorithms. They are competitive with their existing equivalents and may even have the potential to improve them regarding performance and needed resources [16]. Examples for their applications, as listed in [4, 8], include Hamiltonian simulation [17, 15, 8], amplitude amplification and estimation [8, 18], Gibbs sampling [8], solving quantum linear systems [8, 1], algorithms for topological data analysis [19, 20, 21], certain quantum walk results [8], quantum machine learning [8] and quantum phase estimation [1, 16], the latter of which we will explore in detail in chapter 5.

In the following, we will develop the framework for QSP and QSVT, based on the formulation by Gilyén et al. in [8, 1], inspired by [22]. After introducing the mathematical background for QSP in chapter 3.1, we will first address the basic concepts of singular value decomposition (chapter 3.2) and block encoding (chapter 3.3) before concluding with the foundations of QSVT (chapter 3.4) and its implementation (chapter 3.5).

3.1 Quantum Signal Processing

The keynote of QSP is to alternate two different types of rotational operators, namely the signal rotation operator W and the signal processing rotation operator S. While $W(\theta)$ performs a rotation around a fixed angle θ, the rotation angle of $S(\phi)$ varies, predetermined by a given sequence that is denoted by $\vec{\phi} = (\phi_0, ...\phi_d) \in \mathbb{R}^{d+1}$. As a result, one receives a composite gate

$$U_{\vec{\phi}}(\theta) := S(\phi_0)W(\theta)S(\phi_1)W(\theta)...S(\phi_{d-1})W(\theta)S(\phi_d). \tag{12}$$

It is common that $W(\theta)$ rotates around the x-axis of the Bloch sphere, whereas $S(\phi)$ performs a rotation around its z-axis. In this convention, often labeled as the W_x-convention, the signal rotation and signal processing rotation operator take the form

$$W(\theta) := e^{i\theta/2X} = \begin{pmatrix} \cos(\theta/2) & i\sin(\theta/2) \\ i\sin(\theta/2) & \cos(\theta/2) \end{pmatrix} \qquad S(\phi) := e^{i\phi Z} = \begin{pmatrix} e^{i\phi} & 0 \\ 0 & e^{-i\phi} \end{pmatrix}, \tag{13}$$

or equivalently,

$$W(x) := \begin{pmatrix} x & i\sqrt{1-x^2} \\ i\sqrt{1-x^2} & x \end{pmatrix}, \tag{14}$$

where $x := \cos(\theta/2) \in [-1, 1]$.

The following key theorem, originally developed by Low et al. in [11], characterizes the type of unitaries achievable using the sequence of alternating rotations and the corresponding constraints:

Theorem 1 (Quantum signal processing [8]). *For $d \in \mathbb{N}$, there exists an angle set $\vec{\phi} = (\phi_0, ...\phi_d) \in \mathbb{R}^{d+1}$ such that $\forall x \in [-1, 1]$:*

$$U_{\vec{\phi}}(x) = e^{i\phi_0 Z} \prod_{j=1}^{d} \left(W(x) e^{i\phi_j Z} \right) = \begin{pmatrix} P(x) & iQ(x)\sqrt{1-x^2} \\ iQ^*(x)\sqrt{1-x^2} & P^*(x) \end{pmatrix} \tag{15}$$

if and only if the complex polynomials $P(x)$, $Q(x) \in \mathbb{C}[x]$ fulfill

(i) $deg(P) \leq d$ and $deg(Q) \leq d-1$

(ii) P has parity d mod 2 and Q has parity $(d-1)$ mod 2

(iii) $\forall x \in [-1, 1] : |P(x)|^2 + (1-x^2)|Q(x)|^2 = 1$,

where $P^*(x)$ and $Q^*(x)$ denote the polynomials created when complex conjugating the coefficients of $P(x)$ and $Q(x)$, respectively.[3]

It is easy to see that constraint *(iii)* ensures the unitarity of the operator above.

Remarkably, the QSP theorem above holds both ways: It is straightforward to recognize that, given a set of rotation angles $\vec{\phi}$, the resulting composite gate contains polynomials of x in its entries. Nonetheless, the theorem also guarantees the existence of an angle set for given suitable polynomials $P(x)$ and $Q(x)$. Finding the correct angle set is however non-trivial. An algorithm for this is inferred by the proof of the QSP theorem in [8] based on the work of Low et al. [11]. Further efficient classical methods [4] to find the phase angles have been presented in various works such as [23, 24, 25, 26, 27], which we will not discuss in more detail here.

Given a polynomial $P(x)$ satisfying the conditions stated in theorem 1, then one must first find an equally suitable polynomial $Q(x)$ in order to apply it. The following theorem ensures the existence of such a polynomial $Q(x)$.

Theorem 2 (Existence of QSP Polynomials [8]). *For $d \in \mathbb{N}$ and a polynomial $P \in \mathbb{C}[x]$, there exists a polynomial $Q \in \mathbb{C}[x]$ such that both P and Q satisfy the conditions from theorem 1 if and only if P satisfies conditions (i) and (ii) from theorem 1 and additionally*

[3] Note that $Q(x)$ only appears in a product with $\sqrt{1-x^2}$. It is hence only determined for $x \in (-1, 1)$ and should thus be treated as the restriction $Q(x)|_{(-1,1)} \in \mathbb{C}[x]$, to be precise.

(v) $\forall x \in [-1,1] : |P(x)| \leq 1$

(vi) $\forall x \in (-\infty,-1] \cup [1,\infty) : |P(x)| \geq 1$

(vii) If d is even, then $\forall x \in \mathbb{R} : P(ix)P^*(ix) \geq 1$.

Conversely: Let $d \in \mathbb{N}$ and if $Q \in \mathbb{C}[x]$, then there exists a $P \in \mathbb{C}[x]$ such that both P and Q satisfy the conditions from theorem 1 if and only if Q satisfies conditions (i) and (ii) from theorem 1 and additionally

(viii) $\forall x \in [-1,1] : \sqrt{1-x^2}|Q(x)| \leq 1$

(ix) If d is odd, then $\forall x \in \mathbb{R} : (1+x^2)Q(ix)Q^*(ix) \geq 1$.

The proof can be found in [8], which simultaneously infers an algorithm for finding Q. This is referred to as the completion step, often forming the first and crucial part of the phase angle finding algorithm, followed by the so-called decomposition [25]. The existing methods rely on optimization or polynomial factorization [26] and are covered by the aforementioned sources for angle finding algorithms.

Another noteworthy remark is that there exist other QSP conventions than the W_x-convention. Commonly known is for instance the reflection convention, in which we regard the signal rotation operator as a reflection,

$$R(x) := \begin{pmatrix} x & \sqrt{1-x^2} \\ \sqrt{1-x^2} & -x \end{pmatrix}. \tag{16}$$

It is easy to see that the conventions are equivalent, since $R(x)$ can be obtained from $W(x)$ by $R(x) = -ie^{i\frac{\pi}{4}Z}W(x)e^{i\frac{\pi}{4}Z}$. The reflection convention becomes especially relevant when QSVT is derived from QSP, for which Gilyén et al. adopt the reflection convention [8]. In practice, to find the angles in the reflection convention, one computes them in the W_x-convention first and then transfers them by a simple mapping.

Often we are rather interested in the achievable polynomial transformation $\mathrm{Poly}(x)$ of the signal x within a subsystem than the unitary $U_{\vec{\phi}}(x)$ itself. It is obtained by measuring the QSP register in a chosen basis, called the signal basis, like $\langle \cdot | U_{\vec{\phi}}(x) | \cdot \rangle$. Which conditions $\mathrm{Poly}(x)$ has to meet depends on the selected combination of signal basis and QSP convention. For instance, using the W_x-convention and measuring in the computational basis, we can instantly see from equation (15) that $\mathrm{Poly}(x) = \langle 0 | U_{\vec{\phi}}(x) | 0 \rangle = P(x)$. The polynomial transformation is thus limited by the constraints imposed on $P(x)$ and $Q(x)$ by theorems 1 and 2. This already significantly narrows down the set of feasible polynomial transformations. A better choice for the signal basis is e.g. the Hadamard basis, in which the resulting polynomial becomes $\mathrm{Poly}(x) = \langle + | U_{\vec{\phi}}(x) | + \rangle = \Re(P(x)) + i\Re(Q(x))\sqrt{1-x^2}$. One can show that with this particular combination of $P(x)$ and $Q(x)$, each being subject to the aforementioned constraints, one achieves a polynomial with less restrictions [1]. We therefore adopt the Hadamard basis as a signal basis in this thesis.

3.2 Singular Value Decomposition

In order to later define the quantum singular value transformation, the singular value decomposition (SVD) is introduced. It is a mathematical tool akin to the better-known eigendecomposition

of a matrix, but applicable to arbitrary (rectangular) matrices.

Definition 1 (Singular value decomposition [8]). *Let A be a matrix $A \in \mathbb{C}^{q\times p}$. Then there exist unitary matrices $U \in \mathbb{C}^{q\times q}$ and $V \in \mathbb{C}^{p\times p}$ such that A can be factorized as*

$$A = U\Sigma V^\dagger \tag{17}$$

where $\Sigma \in \mathbb{R}^{q\times p}$ denotes a diagonal matrix containing $c := \min\{q,p\}$ singular values of A. The singular values are non-negative real numbers uniquely determined by A.

However, the SVD itself is not unique. One can e.g. choose the SVD such that the singular values appear in descending order, $\sigma_1 \geq \sigma_2 ... \geq \sigma_c$, which fixes Σ.

The columns of U are known as left singular vectors and analogously, right singular vectors refer to the columns of V. They are denoted as $\{|u_l\rangle\}_{l=1,...q}$ and $\{|v_j\rangle\}_{j=1,...p}$, respectively. As both U and V are unitary by premise, the left and right singular vectors each form orthonormal basis sets, which gives rise to writing

$$A = \sum_{i=1}^{c} \sigma_i |u_i\rangle\langle v_i| . \tag{18}$$

With the objective of performing a polynomial transformation on the singular values of a matrix in the context of QSVT, the singular value transformation is defined below. It provides a general way to apply a function to the singular values of a given matrix.

Definition 2 (Singular value transformation by even/odd functions [8]). *Let $f : \mathbb{R} \to \mathbb{C}$ be an even or odd function and $A \in \mathbb{C}^{q\times p}$ with an SVD as in definition 1. Then, the singular value transformation of A by f is defined as*

$$f^{(SV)}(A) := \begin{cases} \sum_{i=1}^{c} f(\sigma_i) |u_i\rangle\langle v_i| & \text{if } f \text{ odd} \\ \sum_{i=1}^{p} f(\sigma_i) |v_i\rangle\langle v_i| & \text{if } f \text{ even} \end{cases} \tag{19}$$

where $c = \min\{q,p\}$ and we define $\sigma_i := 0 \,\forall\, c < i \leq p$. Note that for even functions, $f^{(SV)}(A)$ is an endomorphism of the right singular vector space.

3.3 Block Encoding

Recall that one can only implement operations as quantum gates on a quantum device if they are unitary. Still, it can be of interest to implement non-unitary matrices, performed by a technique that is called projected unitary encoding. Its key idea is to embed the desired non-unitary matrix A inside a larger unitary matrix U, which can be executed on a quantum device [8].

Definition 3 (Projected unitary encoding [8]). *Let U be a unitary operator and Π, $\tilde{\Pi}$ orthogonal projectors. We then call $\{U, \Pi, \tilde{\Pi}\}$ a projected unitary encoding of A if $A = \tilde{\Pi}U\Pi$.*

U then takes the form

$$U = \begin{array}{c} \\ \tilde{\Pi} \end{array} \begin{array}{c} \Pi \\ \begin{pmatrix} A & \cdot \\ \cdot & \cdot \end{pmatrix} \end{array}, \tag{20}$$

and it is sometimes referred to as the projected unitary encoding as well. The projectors

$$\tilde{\Pi} := \sum_{l=1}^{q} |u_l\rangle\langle u_l| \quad \Pi := \sum_{j=1}^{p} |v_j\rangle\langle v_j| \tag{21}$$

locate A within U, where $\{|u_l\rangle\}$ and $\{|v_j\rangle\}$ again denote the left and right singular vectors of A. Often the upper left corner of U is chosen to represent A, so that the projectors are given by $\tilde{\Pi} = \Pi = |0\rangle\langle 0|^{\otimes a} \otimes \mathbb{1}$ for $a \in \mathbb{N}$ determined by the dimensions of U and A. This is the special case of projected unitary encodings called a block encoding [8]. For completeness, it is formally defined below.

Definition 4 (Block encoding [8])**.** *Let A be an operator acting on s qubits, $\alpha, \varepsilon \in \mathbb{R}_+$ and $a \in \mathbb{N}$. Then, the $(s+a)$-qubit unitary U is an (α, a, ε)-block encoding of A, if*

$$\big|\big|A - \alpha(\langle 0|^{\otimes a} \otimes \mathbb{1})U(|0\rangle^{\otimes a} \otimes \mathbb{1})\big|\big| \leq \varepsilon. \tag{22}$$

The factor α rescales A such that $||U|| = 1$ is ensured. Furthermore, note that the definition above assumes A to be a square $2^s \times 2^s$-matrix. This caveat can be overcome by embedding $A \in \mathbb{C}^{r\times t}$ in a bigger matrix $A' \in \mathbb{C}^{2^s \times 2^s}$, where $r, t \leq 2^s$, by filling all entries of A' apart from A with zeroes.[4]

3.4 Quantum Singular Value Transformation

The quantum singular value transformation introduced by Gilyén et al. [8] can be viewed as the extension of QSP to the multi-qubit case, polynomially transforming not a scalar but a whole matrix that was block encoded into a bigger matrix. It originates from combining the already established QSP with qubitization, a concept first introduced by Low and Chuang in [15]. Martyn et al. summarized their work in [1], explaining how they considered a hermitian operator $\mathcal{H}$ block-encoded within a bigger unitary U and found that one can associate a Bloch sphere (i.e. a qubit basis) with every eigenspace of $\mathcal{H}$, giving the concept its name. The known formalism of QSP can then be applied to the single-qubit subspaces, giving rise to the quantum eigenvalue transformation theorem. It describes how to synthesize polynomial transformations of the eigenvalues of $\mathcal{H}$ within U. Gilyén et al. went on to generalize the results of [15] to arbitrary complex-valued matrices in [8], exchanging the eigenvalue decomposition for the singular value decomposition and thus developing QSVT.

Loosely speaking, one can transfer the picture of QSP to QSVT by identifying a projected unitary encoding U with the signal rotation operator W and interspersing it with the analogue of the signal processing rotation operator S. The latter is substituted by projector-controlled phase shift operations

$$\tilde{\Pi}_\phi := e^{i\phi(2\tilde{\Pi}-\mathbb{1})} \quad \Pi_\phi := e^{i\phi(2\Pi-\mathbb{1})}, \tag{23}$$

where Π, $\tilde{\Pi}$ are the projectors of the block-encoding. We can easily confirm that, per construction,

[4] A' is a faithful representation of A since $(A+B)' = A' + B'$ and $(A \cdot C)' = A' \cdot C'$ for any $A, B \in \mathbb{C}^{r\times t}$ and $C \in \mathbb{C}^{t\times l}$, $l \leq 2^t$.

Π_ϕ applies the phase $e^{i2\phi}$ to all states that lie in the image of Π ($\tilde{\Pi}_\phi$ works analogously):

$$\begin{aligned}\Pi_\phi = e^{i\phi(2\Pi-\mathbb{1})} &= \cos(\phi)\mathbb{1} + i\sin(\phi)(2\Pi-\mathbb{1}) \\ &= \frac{1}{2}\left(e^{i\phi}+e^{-i\phi}\right)\mathbb{1} + \frac{1}{2}\left(e^{i\phi}-e^{-i\phi}\right)(2\Pi-\mathbb{1}) \\ &= e^{-i\phi}\mathbb{1} + \left(e^{i\phi}-e^{-i\phi}\right)\Pi.\end{aligned} \tag{24}$$

Π maps all $|\psi_\Pi\rangle \in \mathrm{Im}(\Pi)$ to itself and all $|\psi_\perp\rangle \notin \mathrm{Im}(\Pi)$ to zero. This way,

$$\Pi_\phi |\psi_\Pi\rangle = e^{i\phi}|\psi_\Pi\rangle \quad \text{and} \quad \Pi_\phi |\psi_\perp\rangle = e^{-i\phi}|\psi_\perp\rangle, \tag{25}$$

or equivalently after multiplying with a global phase of $e^{i\phi}$, we obtain the desired phase shift of

$$\Pi_\phi |\psi_\Pi\rangle = e^{i2\phi}|\psi_\Pi\rangle \quad \text{and} \quad \Pi_\phi |\psi_\perp\rangle = |\psi_\perp\rangle. \tag{26}$$

The projector-controlled phase shift gates generalize the signal processing rotation operator in the sense that they too apply rotations about a predetermined angle ϕ. In contrast to the single-qubit case, we have to account for the fact that the matrix we want to manipulate is block encoded within another by picking out the right subspaces with the projectors as per definition of Π_ϕ and $\tilde{\Pi}_\phi$. After this intuitive description, we present the main theorem of QSVT.

Theorem 3 (Quantum singular value transformation [1])**.** *Let U be a projected unitary encoding of a matrix $A \in \mathbb{C}^{q\times p}$. The "alternating phase modulation sequence $U_{\vec{\phi}}$" [8] performs the singular value transformation* $\mathrm{Poly}^{(SV)}(A)$ *of the projected unitary A in the following way, given that* $\mathrm{Poly}^{(SV)}(A)$ *is of degree at most d and obeys the conditions of P from theorem 1.*
For odd d:

$$U_{\vec{\phi}} = \tilde{\Pi}_{\phi_1} U \left[\prod_{k=1}^{(d-1)/2} \Pi_{\phi_{2k}} U^\dagger \tilde{\Pi}_{\phi_{2k+1}} U\right] = \begin{matrix}\Pi \\ \tilde{\Pi}\end{matrix} \begin{pmatrix} \mathrm{Poly}^{SV}(A) & \cdot \\ \cdot & \cdot \end{pmatrix}. \tag{27}$$

For even d:

$$U_{\vec{\phi}} = \left[\prod_{k=1}^{d/2} \Pi_{\phi_{2k-1}} U^\dagger \tilde{\Pi}_{\phi_{2k}} U\right] = \begin{matrix}\Pi \\ \tilde{\Pi}\end{matrix} \begin{pmatrix} \mathrm{Poly}^{SV}(A) & \cdot \\ \cdot & \cdot \end{pmatrix}. \tag{28}$$

$\mathrm{Poly}^{(SV)}$ *is called the target polynomial.*

Crucially, the theorem shows that the QSVT sequence performs a polynomial transformation not on the full quantum system, but on a clearly identified subsystem. Note that, in particular, QSVT uses precisely the angle set $\vec{\phi}$ from QSP, determined by $\mathrm{Poly}^{(SV)}$ as in theorem 1. The proof employs rather involved considerations of the action of the phase modulation sequence on the partaking vector spaces, which is why we do not provide further mathematical insights on QSVT within this thesis and instead refer to [8] for this.

3.5 Implementation

After having motivated the foundations of QSVT it remains to bridge the gap from the theorem to the implementation on a quantum device. The circuits shown in this chapter are not fully generalized but suffice for our purposes of later demonstrating QSVT on the example of phase estimation.

Consider first the projector-controlled phase shift operation Π_ϕ as defined in equation (23). It is implemented by involving an additional ancilla qubit and two instances of projector-controlled-NOT gates (C_Π-NOT) around a Z-rotation about ϕ, as pictured in Figure 5.

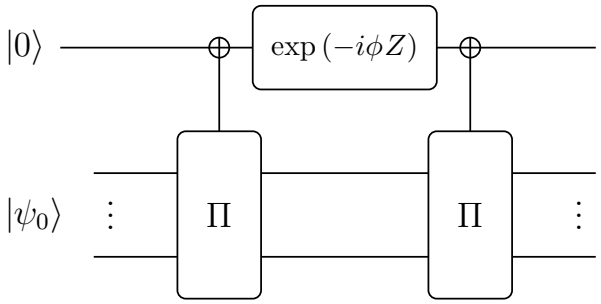


Figure 5: Quantum circuit implementation of Π_ϕ for QSVT [1]. Here, $|\psi_0\rangle$ denotes an arbitrary initial state. $\tilde{\Pi}_\phi$ is implemented analogously.

The C_Π-NOT gate flips the target qubit if the control qubits lie in the image of Π, defined as

$$C_\Pi\text{-NOT} := X \otimes \Pi + \mathbb{1} \otimes (\mathbb{1} - \Pi). \tag{29}$$

One can easily check that the circuit above executes the desired phase shift by writing out all quantum gates from Figure 5 and using that Π has the properties of a projector:

$$\begin{aligned}\Pi_\phi &= (X \otimes \Pi + \mathbb{1} \otimes (\mathbb{1} - \Pi)) \left(e^{-i\phi Z} \otimes \mathbb{1}\right) (X \otimes \Pi + \mathbb{1} \otimes (\mathbb{1} - \Pi)) \\ &= X e^{-i\phi Z} X \otimes \Pi^2 + + e^{-i\phi Z} X \otimes (\mathbb{1} - \Pi)\Pi + X e^{-i\phi Z} \otimes \Pi(\mathbb{1} - \Pi) + e^{-i\phi Z} \otimes (\mathbb{1} - \Pi)(\mathbb{1} - \Pi) \\ &= e^{-i\phi Z} \otimes (\mathbb{1} - \Pi) + X e^{-i\phi Z} X \otimes \Pi.\end{aligned} \tag{30}$$

Applying equation (30) to $|0\rangle \otimes |\psi_\Pi\rangle$ and $|0\rangle \otimes |\psi_\perp\rangle$ delivers the same result as equation (25):

$$\begin{aligned}\left[e^{-i\phi Z} \otimes (\mathbb{1} - \Pi) + X e^{-i\phi Z} X \otimes \Pi\right] |0\rangle \otimes |\psi_\Pi\rangle &= X e^{-i\phi Z} X |0\rangle \otimes |\psi_\Pi\rangle \\ &= |0\rangle \otimes e^{i\phi} |\psi_\Pi\rangle \\ \left[e^{-i\phi Z} \otimes (\mathbb{1} - \Pi) + X e^{-i\phi Z} X \otimes \Pi\right] |0\rangle \otimes |\psi_\perp\rangle &= e^{-i\phi Z} |0\rangle \otimes |\psi_\perp\rangle \\ &= |0\rangle \otimes e^{-i\phi} |\psi_\perp\rangle .\end{aligned} \tag{31}$$

The correct action of the circuit is thus confirmed.

In accordance with theorem 3, the complete QSVT circuit is comprised of alternating Π_ϕ and $\tilde{\Pi}_\phi$ with the projected unitary encoding U and its adjoint $U^\dagger$, as depicted below in Figure 6.

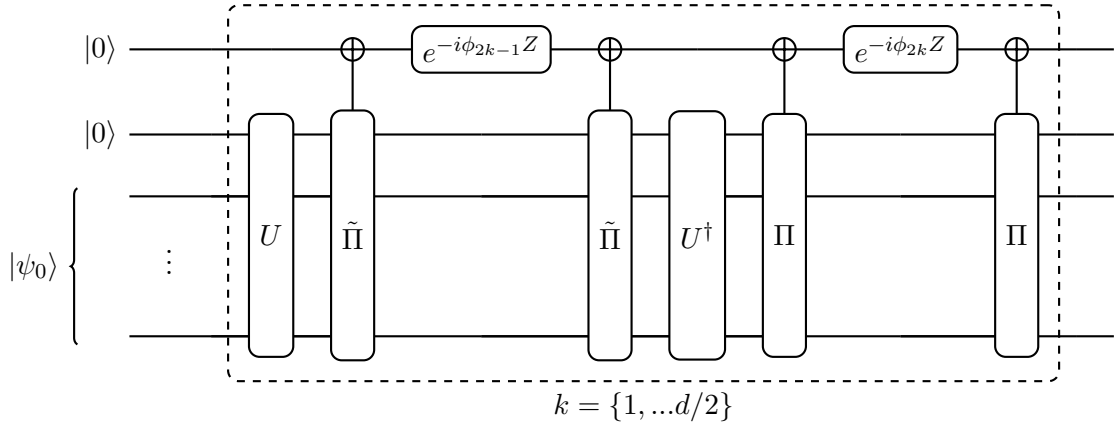


Figure 6: Quantum circuit implementing a simplified QSVT sequence for the even case. The dotted line marks that the gate sequence inside of it is repeated for $k = \{1, ...d/2\}$. $|\psi_0\rangle$ denotes an arbitrary state.

4 Quantum Phase Estimation

Quantum Phase Estimation is a procedure used as a subroutine of many quantum algorithms. Given a unitary U with an eigenstate $|u\rangle$, the corresponding eigenvalue must have magnitude 1, i.e. it can be written as a complex phase:

$$U|u\rangle = e^{i2\pi\varphi}|u\rangle . \tag{32}$$

Since the phase φ cannot be obtained by measurement, the QPE algorithm is used to find an estimate for its unknown value [3]. Some of the numerous applications of QPE are outlined in the following.

In the context of Hamiltonian simulation, we assume access to the unitary $U = e^{i2\pi\mathcal{H}}$, where $\mathcal{H}$ denotes the Hamiltonian of some quantum system of interest with eigenvalues λ_j. Then, $U|u_j\rangle = e^{i2\pi\lambda_j}|u_j\rangle$ becomes a phase estimation problem [10]. The gained information on the eigenvalues, including especially the ground state energy, of a Hamiltonian admits conclusions about the stable configurations and reaction behavior of quantum systems. Hence, the quantum chemical applications of QPE are of high industrial relevance, for instance in pharmaceutics and material sciences [4].

Furthermore, the well-known Shor algorithm for finding prime factors of a composite integer can be reduced to a QPE problem [2]. Shor's algorithm has had a great impact on the field of quantum computation and has inspired a wide range of algorithms itself. Developed by Peter Shor in 1994, it showed that the factoring algorithm and the so-called discrete logarithm problem can be efficiently solved on a quantum computer, while both problems are not considered to be efficiently solvable on a classical device [2]. The exponential speedup provided by Shor's factoring algorithm contributed to manifesting the intrinsic advantage of quantum computers over classical computers [7]. Further applications of QPE include amplitude estimation, Gibbs sampling and the HHL algorithm for solving linear systems of equations, to name but a few [10, 4].

In the ensuing discussions of QPE, φ is assumend to be an arbitrary real number within $[0, 1)$ due to the periodicity of the exponential function $e^{i2\pi\varphi}$. Just like one can represent an integer number in binary code, one can use the binary fraction representation to express φ as a bit string $\varphi = \sum_j^m 2^{-j}\varphi_j = 0.\varphi_1\varphi_2...\varphi_m$, where $\varphi_j \in \{0, 1\}\,\forall j$, also abbreviated as $.\varphi_1\varphi_2...\varphi_m$. The set of all φ with such a finite representation is called dyadic rationals, and it is dense in $[0, 1) \subset \mathbb{R}$ [28]. Still, we allow $m \to \infty$ in order to capture all possible $\varphi \in [0, 1)$ [1]. In the following, the standard QPE approach (chapter 4.2) is presented. It is based on the quantum Fourier transform, which is therefore introduced in chapter 4.1.

4.1 Quantum Fourier Transform

The discrete Fourier transform is often used as a subroutine of classical algorithms. Similarly, there exists a quantum version of it, whose applications go beyond quantum phase estimation. It forms an essential part of many quantum algorithms such as Shor's factoring algorithm [10]. We introduce the quantum Fourier transform (QFT) here, based on the discussions in [10] and [3].

Classically, the Fourier transform takes a vector of $N \in \mathbb{N}$ complex numbers $x_0, ...x_{N-1}$ and

gives $y_0, ...y_{N-1}$ as the transformed data according to

$$y_j = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{i2\pi \frac{kj}{N}} x_k. \tag{33}$$

Analogously, the QFT efficiently performs a discrete Fourier transform of quantum mechanical amplitudes x_j by the mapping

$$\sum_{j=0}^{N-1} x_j \left|j\right\rangle \longrightarrow \sum_{k=0}^{N-1} y_k \left|k\right\rangle, \tag{34}$$

where $\{\left|0\right\rangle, ... \left|N-1\right\rangle\}$ is an orthonormal basis.

Definition 5 (Discrete quantum Fourier transform [10])**.** *Let $\left|j\right\rangle \in \left(\mathbb{C}^2\right)^{\otimes n}$ denote any computational basis state, and let $N = 2^n$.*[5] *Then, the discrete quantum Fourier transform is defined as the unitary U_{FT} acting on $\left|j\right\rangle$ such that*

$$U_{FT} \left|j\right\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{i2\pi \frac{kj}{N}} \left|k\right\rangle, \tag{35}$$

and similarly, the inverse discrete quantum Fourier transform is given by

$$U_{FT}^{\dagger} \left|j\right\rangle = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{-i2\pi \frac{kj}{N}} \left|k\right\rangle. \tag{36}$$

In particular, from equation (5) follows

$$U_{FT} \left|0\right\rangle^{\otimes n} = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \left|k\right\rangle = H^{\otimes n} \left|0\right\rangle. \tag{37}$$

In order to derive a quantum circuit implementation of the QFT, it is helpful to rewrite it in terms of its action on the individual qubits. For this, we employ the binary representation of integers, e.g.

$$k = k_{n-1}k_{n-2}...k_0. = \sum_{i=0}^{n-1} k_i 2^i \quad \text{where } k_i \in \{0,1\}\, \forall i \tag{38}$$

for k and similarly for j. Moreover, this gives rise to writing $\left|k\right\rangle = \left|k_{n-1}k_{n-2}...k_0\right\rangle$ and leads to

$$\begin{aligned} \frac{kj}{N} &= k_0 \frac{j}{2^n} + k_1 \frac{j}{2^{n-1}} + ... + k_{n-1} \frac{j}{2} \\ &= k_0(.j_{n-1}...j_0) + k_1(j_{n-1}.j_{n-2}...j_0) + ... + k_{n-1}(j_{n-1}...j_1.j_0). \end{aligned} \tag{39}$$

Using the periodicity of the exponential function, the exponential can thus be written as

$$e^{i2\pi \frac{kj}{N}} = e^{i2\pi k_0(.jn-1...j_0)} e^{i2\pi k_1(.j_{n-2}...j_0)} ... e^{i2\pi k_{n-1}(.j_0)}. \tag{40}$$

[5] Here, we have selected the computational basis and $N = 2^n$, although the definition of the QFT generalizes to arbitrary orthonormal bases and $N \in \mathbb{N}$. If $N \neq 2^n$, one can fill up the missing data up to the next even power of 2 with zeroes [3].

Consequently, the "product representation" [3] of the QFT is obtained as follows:

$$\begin{aligned} U_{FT}\left|j\right\rangle =& \frac{1}{\sqrt{N}} \sum_{k_0,k_1,\ldots k_{n-1}} e^{i2\pi k_0(.j_{n-1}\ldots j_0)} e^{i2\pi k_1(.j_{n-2}\ldots j_0)} \ldots e^{i2\pi k_{n-1}(.j_0)} \left|k_{n-1}k_{n-2}\ldots k_0\right\rangle \\ =& \frac{1}{\sqrt{N}} \left(\sum_{k_{n-1}} e^{i2\pi k_{n-1}(.j_0)} \left|k_{n-1}\right\rangle \right) \otimes \left(\sum_{k_{n-2}} e^{i2\pi k_{n-2}(.j_1 j_0)} \left|k_{n-2}\right\rangle \right) \\ & \otimes \cdots \otimes \left(\sum_{k_0} e^{i2\pi k_0(.j_{n-1}\ldots j_1 j_0)} \left|k_0\right\rangle \right) \\ =& \frac{1}{\sqrt{N}} \left(\left|0\right\rangle + e^{i2\pi(.j_0)}\left|1\right\rangle\right) \otimes \left(\left|0\right\rangle + e^{i2\pi(.j_1 j_0)}\left|1\right\rangle\right) \otimes \ldots \otimes \left(\left|0\right\rangle + e^{i2\pi(.j_{n-1}\ldots j_0)}\left|1\right\rangle\right). \end{aligned} \tag{41}$$

As a last prerequisite for the implementation of the QFT, the R_k gates are defined. They are Pauli-Z-rotations up to a global phase:

$$R_k := \begin{pmatrix} 1 & 0 \\ 0 & e^{i2\pi/2^k} \end{pmatrix}. \tag{42}$$

The implementation of the QFT is best understood by retracing the one- and two-qubit case. If $n = 1$, the state one desires to acquire by the QFT circuit is

$$U_{FT}\left|j_0\right\rangle = \frac{1}{\sqrt{2}}\left(\left|0\right\rangle + e^{i2\pi(.j_0)}\left|1\right\rangle\right),$$

in accordance with equation (41). Note that this is precisely the result of $\left|j_0\right\rangle$ undergoing a Hadamard gate:

$$\left|j_0\right\rangle \longrightarrow \frac{1}{\sqrt{2}}\left(\left|0\right\rangle + e^{i2\pi(.j_0)}\left|1\right\rangle\right) = \frac{1}{\sqrt{2}} \left\{ \begin{array}{ll} \left|0\right\rangle + \left|1\right\rangle & \text{if } j_0 = 0 \\ \left|0\right\rangle - \left|1\right\rangle & \text{if } j_0 = 1 \end{array} \right\} = H\left|j_0\right\rangle. \tag{43}$$

Hence, applying the Hadamard gate to the qubit suffices to implement the QFT in the single-qubit case. Next, if $n = 2$, one requires a circuit implementing

$$U_{FT}\left|j_1\right\rangle \otimes \left|j_0\right\rangle = \frac{1}{\sqrt{4}}\left(\left|0\right\rangle + e^{i2\pi(.j_0)}\left|1\right\rangle\right) \otimes \left(\left|0\right\rangle + e^{i2\pi(.j_1 j_0)}\left|1\right\rangle\right). \tag{44}$$

This action is achieved by additionally applying an R_2 gate on the qubit assigned to $\left|j_1\right\rangle$, controlled by the $\left|j_0\right\rangle$-qubit.

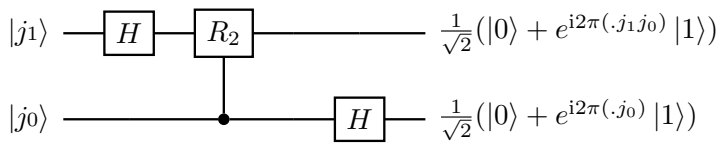


Figure 7: Quantum circuit for QFT with $n = 2$ [10].

Note that Figure 7 shows the two-qubit QFT circuit, but the order of outcome states is reversed. This can be remedied by applying a SWAP gate that interchanges the qubit states. The full quantum circuit for the QFT is achieved similarly and depicted below. Not pictured are the

$\mathcal{O}(n/2)$ SWAP gates that ensure the right ordering of the final qubit states [10].[6]

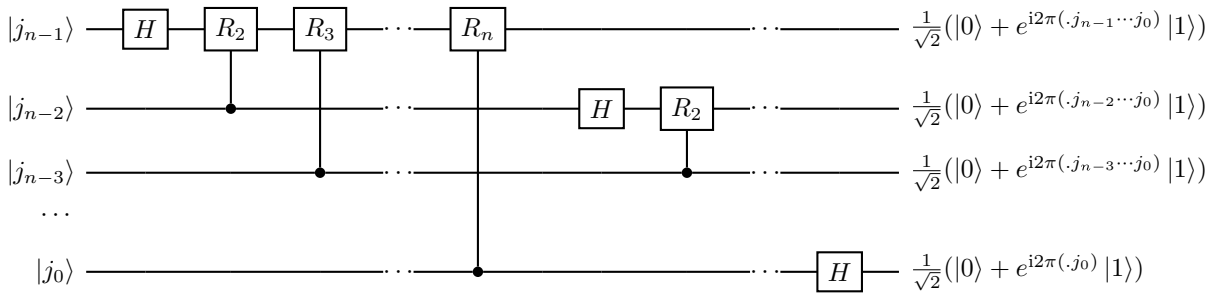


Figure 8: Quantum circuit for QFT before applying swap operations [10].

The quantum Fourier transform has a gate complexity of $\mathcal{O}(n^2)$ [10], providing an asymptotic exponential advantage over the classical Fast Fourier transform (FFT) which scales as $\mathcal{O}(2^n n)$ [3].

4.2 Quantum Phase Estimation Based on Quantum Fourier Transform

In the following, the standard quantum phase estimation procedure is presented, based on the description in [10] and [3]. Recall that the objective of QPE is to obtain an estimate of the unknown phase φ associated to a certain eigenstate $|u\rangle$ of U. To be consistent with the notation of the figures taken from [10], we adopt $|\psi\rangle = |u\rangle$ only in this subchapter. Moreover, assume that φ is a dyadic rational with n digits, $\varphi = .\varphi_{n-1}\varphi_n ... \varphi_0$, and that U occupies l qubits. Also presuming access to the powers of U, one can construct a unitary operation $\mathcal{U}$ acting on $l+n$ qubits:

$$\mathcal{U} = \sum_{j=0}^{2^n-1} |j\rangle\langle j| \otimes U^j, \tag{45}$$

where, as in the definition of the QFT, j labels the computational basis states.

The phase estimation algorithm requires two registers: An n-qubit ancilla register, prepared in the $|0\rangle^{\otimes n}$-state, and an l-qubit system register initially in the state $|\psi\rangle$. The first stage of the QPE algorithm is a quantum Fourier transform acting on the ancilla register, or equivalently, a multi-qubit Hadamard gate transform as shown by equation (37). In the product representation, the resulting state is thus

$$\frac{1}{\sqrt{2^n}} (|0\rangle + |1\rangle)^{\otimes n}. \tag{46}$$

Secondly, $\mathcal{U}$ is applied to both registers. One can show that $\mathcal{U}$ can be rewritten as a series of U raised to successive powers of two acting on the system register, controlled by the ancilla register, and implemented as in Figure 9:

[6] The "big O"-notation is used to mark upper bounds of a function. One says "$f(n) \in \mathcal{O}(g(n))$" if there are constants c and n_0 such that $\forall n \geq n_0$, $f(n) \leq cg(n)$.

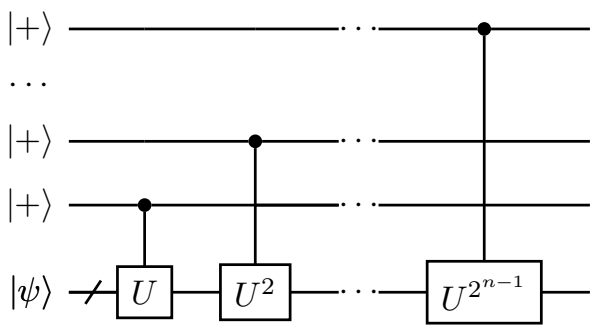


Figure 9: Quantum circuit of $\mathcal{U}$ applying successive powers of two of U to the system register, controlled by the ancilla register. The first stage of the algorithm has already been performed in this diagram. Adapted from [10].

Because the system register is assumed to be prepared in the eigenstate $|\psi\rangle$ of U, one can invoke the eigenvalue equation $U|\psi\rangle = e^{i2\pi\varphi}|\psi\rangle$ to understand the action of $\mathcal{U}$ on the ancilla space. For instance, the action of the circuit in Figure 9 for $n = 1$ is:

$$\begin{aligned}
\mathcal{U}\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes |\psi\rangle &= \left(|0\rangle\langle 0| \otimes \mathbb{1} + |1\rangle\langle 1| \otimes U^{2^0}\right)\left(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes |\psi\rangle\right) \\
&= \frac{1}{\sqrt{2}}\left(|0\rangle \otimes |\psi\rangle + |1\rangle \otimes U^{2^0}|\psi\rangle\right) \\
&= \frac{1}{\sqrt{2}}\left(|0\rangle \otimes |\psi\rangle + |1\rangle \otimes e^{i2\pi\varphi}|\psi\rangle\right) \\
&= \frac{1}{\sqrt{2}}\left(|0\rangle + e^{i2\pi\varphi}|1\rangle\right) \otimes |\psi\rangle\,.
\end{aligned} \tag{47}$$

The resulting phase can be assigned to the ancilla qubit, while the system register remains in state $|\psi\rangle$. Similarly, applying $\mathcal{U}$ to the full register yields the ancilla register's state

$$\begin{aligned}
&\frac{1}{\sqrt{2^n}}\left(|0\rangle + e^{i2\pi(.\varphi_0)}|1\rangle\right) \otimes \left(|0\rangle + e^{i2\pi(.\varphi_1\varphi_0)}|1\rangle\right) \otimes ... \otimes \left(|0\rangle + e^{i2\pi(.\varphi_{n-1}...\varphi_0)}|1\rangle\right) \\
&\qquad = \frac{1}{\sqrt{2^n}}\sum_{j=0}^{2^n-1} e^{i2\pi\varphi j}|j\rangle\,,
\end{aligned} \tag{48}$$

and it can be compactly rewritten as above by reversing a similar calculation to equation (41).

As a third step of the QPE procedure, the ancilla register undergoes an inverse Fourier transform. But comparing the current ancilla register's state with the definition of the QFT (35) reveals that it is precisely $U_{FT}|\varphi\rangle$. Hence, applying $U_{FT}^\dagger$ gives exactly $|\varphi\rangle$, the desired phase result. A subsequent measurement of the ancilla space thus completes the QPE algorithm.

The full circuit diagram summarizes all steps of quantum phase estimation based on QFT.

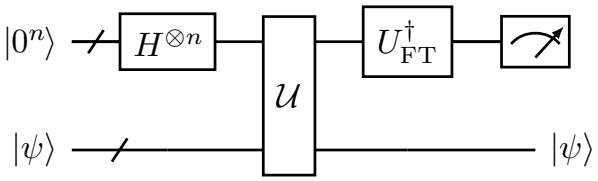


Figure 10: Quantum circuit for quantum phase estimation based on the quantum Fourier transform. Adapted from [10].

Until now, we have assumed the system register to be prepared in the eigenstate $|\psi\rangle$ of U and φ to be a dyadic rational, leading to an exact estimate of φ that is obtained with certainty. In practice, these conditions are seldomly met. We strive to relax both constraints towards greater generality.

Fortunately, a slight adaptation of the procedure gives a "pretty good approximation to φ with high probability" [3] if φ is non-dyadic. To achieve an estimate of φ with accuracy 2^{-t} (i.e. a t-bit estimate) with a success probability of at least $1-\delta$ for some small $\delta > 0$, it suffices to employ a number of

$$t + \left\lceil \log\left(2 + \frac{1}{2\delta}\right)\right\rceil \tag{49}$$

ancilla qubits [3]. The dominant cost of the QPE algorithm is usually generated by the queries of U, whose number scales with $\mathcal{O}(1/\varepsilon)$, where ε denotes the precision of the estimate [4].

What is more, phase estimation can be extended to the system register being in a general superposition state of eigenstates: If $|\psi\rangle = \sum_j \alpha_j |\psi_j\rangle$ such that $|\psi_j\rangle$ are eigenstates of U with eigenvalues $e^{i2\pi\varphi_j}$, respectively, then measuring the ancillae as part of the QPE algorithm yields φ_k with probability $|\alpha_k|^2$. Relaxing both assumptions simultaneously is possible, but subject to some more caveats and demands a more involved analysis [10].

In summary, the standard approach to QPE prepares an n-bit estimate of φ with success probability at least $1-\delta$ on the ancilla register. After a Hadamard transform, a clever series of controlled applications of powers of U is followed by an inverse Fourier transform. The desired phase information is prepared on the ancilla space and read out by a measurement. Specifically, all digits of φ are encoded on the ancillae at the same time. The number of ancilla qubits n is determined by the desired accuracy of the estimate of φ.

Kitaev established another version of QPE in [29]. Its main difference to QFT-based QPE is that the estimate of φ is obtained digit by digit. The algorithm repeatedly employs a simple circuit containing a controlled-U^{2^j} operation which stores the phase estimation on the control qubit, to be eventually read out by a measurement. Remarkably, this ancilla qubit is reused for each digit, motivating the name "iterative quantum phase estimation" [10]. This produces a lower qubit resource cost than standard QPE and can be advantageous for quantum devices with short coherence times. This iterative approach will become relevant in the next chapter.

5 Quantum Phase Estimation Based on QSVT

In the previous chapter, we have developed the standard QPE algorithm based on QFT and briefly introduced iterative phase estimation by Kitaev. This following chapter now elaborates on an example of an iterative QPE algorithm based on QSVT, established by Martyn et al. in 2021. Albeit adapting minor changes, we closely follow the algorithm design presented in [1]. If not specified otherwise, this paper serves as the source for this chapter.

5.1 Outline

Recall from chapter 4 that the QPE problem consists of a unitary U with a corresponding l-qubit eigenstate $|u\rangle$ such that $U|u\rangle = e^{i2\pi\varphi}|u\rangle$ where φ is unknown. The QSVT-based QPE algorithm assumes access to U and its powers, as well as φ to consist of m bits in binary fraction representation, $\varphi = 0.\varphi_1\varphi_2...\varphi_m$, where m can be infinite. In contrast to the textbook QPE algorithm, this iterative version performs a predetermined number of n iterations to calculate an n-bit estimate of φ bit by bit, starting with the least significant one. Each individual estimated digit of φ is obtained by leveraging the power of a QSVT circuit, controlled by an ancilla qubit: A polynomial transformation of the singular values of a specific block encoded matrix is used to encode the estimate of the current digit of interest on the ancilla space. Measurement of the ancilla qubit then allows for the readout of the bit estimate which is contained within a classical parameter θ. At the end of each iteration, θ is updated accordingly to track the current approximate value of φ, and fed forward to the next iteration. Eventually, one is in possession of an approximation to φ.

The description above shows that the QSVT-based QPE algorithm is indeed an example for iterative QPE algorithms. The algorithm can be implemented two ways: On one hand, using one ancilla qubit that is used n times by preparing it in the required initial state over again after each iteration. Alternatively, one can employ n ancillae all initialized from the beginning of the algorithm and generate one phase estimate on one ancilla qubit each. Choosing the second option here, the quantum circuit executing the algorithm takes roughly the following form.

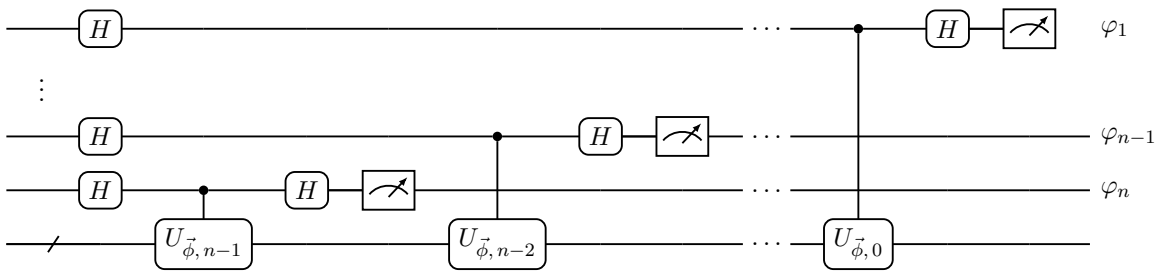


Figure 11: Quantum circuit sketch of the QSVT-based QPE algorithm by [1]. $U_{\vec{\phi},j}$ denotes the QSVT procedures with phase angles $\vec{\phi}$ in iteration j, from $j = n-1$ to $j = 0$. The unconventional downwards labeling of iterations is due to the algorithm design.

5.2 Generation of the Phase Estimate

The key element of the QSVT-based QPE algorithm is the generation of a bit estimate of φ on an ancilla qubit in each of the n iterations. This is achieved by a QSVT sequence, transforming

the singular values of a certain matrix, which is chosen to be

$$A_j(\theta) := \frac{1}{2}(\mathbb{1} + e^{-2\pi i\theta}U^{2^j}) \tag{50}$$

in iteration j, respectively. The reason for this particular choice lies within its singular values. One can confirm that $A_j(\theta)$ has a singular value

$$\sigma_j(\theta) = |\cos(\pi(2^j\varphi - \theta))| \tag{51}$$

corresponding to the right singular vector $|v_j\rangle = |u\rangle$ and left singular vector $|w_j\rangle = e^{i\pi(2^j\varphi-\theta)}|u\rangle$. Crucially, $\sigma_j(\theta)$ encodes both θ, the classical parameter tracking the current estimate of φ, and $2^j\varphi = 0.\varphi_{j+1}...\varphi_m$, the $m-j$ least significant bits of φ.

θ is initialized as 0. In order to update it accordingly, θ is halved at the beginning of each iteration, so that its first decimal equals zero: $\theta \leftarrow \theta/2 = 0.0\theta_2...\theta_{n-j}$. At the end of each iteration, θ_1 takes the value of the bit estimate obtained by measuring the currently involved ancilla qubit. This way, it is ensured that θ always stores the current estimate of φ.

Owing to their structure, the singular values $\sigma_j(\theta)$ allow for a clever subtraction of the previously estimated digits within θ from $2^j\varphi$. This captures the essence of why the algorithm gives the best possible n-bit estimate of φ according to the rounding rule. The bit estimate φ_{j+1} can be directly related to $\sigma_j(\theta)$ by an expression using the sign function, which is defined as

$$\Theta(x) = \begin{cases} 1, & x > 0 \\ 0, & x = 0 \\ -1, & x < 0. \end{cases} \tag{52}$$

The sign function is therefore used as a target function of QSVT to encode the phase information on the ancillae.

The following rigorous analysis of the generation of the phase estimate aims to clarify the argument above. Note first that there are two cases to distinguish between, $m \leq n$ and $n < m$.

(i) $m \leq n$

In the first case, more or equally many iterations are performed than there are digits of φ to estimate. Knowing that one iteration corresponds to one digit of the approximation, this scenario allows for an exact approximation. To be more precise:

Theorem 4 ([1])**.** *If $m \leq n$ and the sign function can be implemented exactly by QSVT, then at the end of the algorithm $\theta = \varphi$.*

The proof follows directly by induction by the following theorem, simultaneously illustrating the algorithm's ingenuity.

Theorem 5 ([1])**.** *If $\varphi = 0.\varphi_1\varphi_2...\varphi_n$, then at the end of iteration j, the approximate value yields $\theta = 0.\varphi_{j+1}\varphi_{j+2}...\varphi_n$.*

Proof. Firstly, if $\varphi = 0.\varphi_1\varphi_2...\varphi_m$ with $m < n$, complete φ to an n-bit number by adding zeroes as the remaining digits $\varphi_{m+1}...\varphi_n$, hence keeping the value of φ unchanged.

Base case. Assume that the QPE algorithm has already been initialized, so that $\theta = 0.0$. The corresponding singular value to $A_{n-1}(0)$ is

$$\sigma_{n-1}(\theta) = |\cos(\pi(2^{n-1}\varphi - \theta))| = |\cos(\pi(0.\varphi_n - 0.0))| = \left|\cos\left(\frac{\varphi_n \pi}{2}\right)\right| = \begin{cases} 1 & \varphi_n = 0 \\ 0 & \varphi_n = 1. \end{cases} \tag{53}$$

From this equation follows that singular value and bit estimate are directly related by the sign function,

$$\Theta\left(\sigma_{n-1}(\theta) - \frac{1}{\sqrt{2}}\right) = \left\{ \begin{array}{ll} -1 & \varphi_n = 1 \\ 1 & \varphi_n = 0 \end{array} \right\} = 1 - 2\varphi_n. \tag{54}$$

The algorithm exploits this connection, implementing the sign function $\Theta\left(\sigma_{n-1}(\theta) - \frac{1}{\sqrt{2}}\right)$ with QSVT to encode φ_n on the ancilla space.[7] This is carried out by the following quantum circuit, which is a piece taken out of the whole algorithm's circuit in Figure 11.

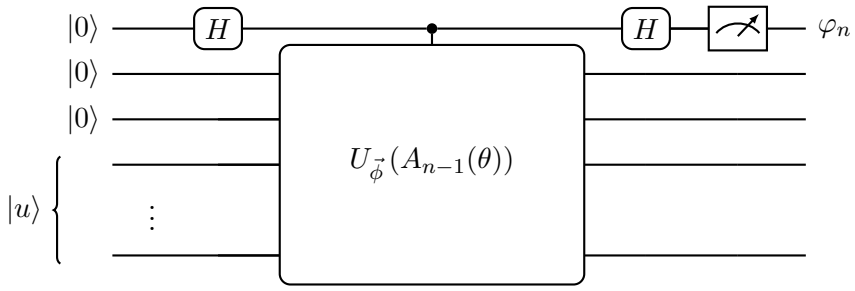


Figure 12: The quantum circuit in iteration $n-1$ generating the phase digit φ_n, inspired by [1].

The following calculations show how the circuit in Figure 12 generates φ_n. Enclosed by the two Hadamard gates is a controlled-QSVT procedure, denoted by $C\text{-}U_{\vec{\phi}}$, implemented on $2+l$ qubits initialized in $|0\rangle \otimes |0\rangle \otimes |u\rangle$. First, the action of the controlled circuit is broken down further, using that H acts on $|0\rangle$ and $|1\rangle$ as given by equations (3) and (4).

$$\begin{aligned} &\left((H \otimes \mathbb{1})(C\text{-}U_{\vec{\phi}})(H \otimes \mathbb{1})\right) |0\rangle \otimes |0\rangle \otimes |0\rangle \otimes |u\rangle \\ &\quad = \left((H|0\rangle\langle 0|H) \otimes \mathbb{1} + (H|1\rangle\langle 1|H) \otimes U_{\vec{\phi}}\right) |0\rangle \otimes |0\rangle \otimes |0\rangle \otimes |u\rangle \\ &\quad = \frac{1}{2}\Big(|0\rangle\langle 0| \otimes \left(\mathbb{1} + U_{\vec{\phi}}\right) + |1\rangle\langle 0| \otimes \left(\mathbb{1} - U_{\vec{\phi}}\right) \\ &\qquad + |1\rangle\langle 1| \otimes \left(\mathbb{1} + U_{\vec{\phi}}\right) + |0\rangle\langle 1| \otimes \left(\mathbb{1} - U_{\vec{\phi}}\right)\Big) |0\rangle \otimes |0\rangle \otimes |0\rangle \otimes |u\rangle \\ &\quad = \frac{1}{2}\left(|0\rangle \otimes \left(\mathbb{1} + U_{\vec{\phi}}\right) |0\rangle \otimes |0\rangle \otimes |u\rangle + |1\rangle \otimes \left(\mathbb{1} - U_{\vec{\phi}}\right) |0\rangle \otimes |0\rangle \otimes |u\rangle\right) \end{aligned} \tag{55}$$

$U_{\vec{\phi}}(A_{n-1}(\theta))$ performs a transformation of the singular values of $A_{n-1}(\theta)$, by alternating rotations on the subspace associated to $|0\rangle \otimes |0\rangle \otimes |\cdot\rangle_l$. This employs the target function $\Theta\left(A_{n-1}(\theta) - \frac{1}{\sqrt{2}}\right)$ on the last l qubits.[8] When expanding $U_{\vec{\phi}}$ in terms of its action on all subspaces $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\} \otimes |\cdot\rangle_l$, we may disregard all contributions to $U_{\vec{\phi}}|\cdot\rangle_{2+l}$ apart

[7] To be precise, the sign function itself actually cannot be implemented by QSVT. Instead, a symmetrized polynomial approximation $P^{PE}_{\epsilon,\Delta}$ of Θ is used as the target function. For pedagogical simplicity, we will continue using Θ for now until establishing the phase estimation polynomial in chapter 5.3. It will also be discussed there which constraints this imposes on theorems 4 and 6.

[8] The original source [1] uses $\Theta\left(\frac{1}{\sqrt{2}} - A_{n-1}(\theta)\right)$. We have adopted the changes necessary to account for this difference throughout the thesis.

the one from the first subspace, given that the projection of the initial state $|00\rangle$ to all other subspaces vanishes, such that

$$U_{\vec{\phi}} |00\rangle \otimes |u\rangle = |00\rangle \otimes \Theta\left(A_{n-1}(\theta) - \frac{1}{\sqrt{2}}\right) |u\rangle . \tag{56}$$

$\Theta\left(A_{n-1}(\theta) - \frac{1}{\sqrt{2}}\right)$ is then applied to $|u\rangle$ as in definition 2, using the SVD of $A_{n-1}(\theta)$ (definition 1). Since the singular value $\sigma_{n-1}(\theta)$ corresponds to singular vectors proportional to $|u\rangle$, only the summand associated to $\sigma_{n-1}(\theta)$ remains. Furthermore, one can employ equation (54) to evaluate $\Theta\left(\sigma_{n-1}(\theta) - \frac{1}{\sqrt{2}}\right)$:

$$\begin{aligned} |00\rangle \otimes \Theta\left(A_{n-1}(\theta) - \frac{1}{\sqrt{2}}\right) |u\rangle &= |00\rangle \otimes \left(\sum_{i=1}^{p} \Theta(\sigma_i - \frac{1}{\sqrt{2}}) |v_i\rangle \langle v_i|\right) |u\rangle \\ &= |00\rangle \otimes \Theta\left(\sigma_{n-1}(\theta) - \frac{1}{\sqrt{2}}\right) |u\rangle \\ &= \Theta\left(\sigma_{n-1}(\theta) - \frac{1}{\sqrt{2}}\right) |00\rangle \otimes |u\rangle \\ &= (1 - 2\varphi_n) |00\rangle \otimes |u\rangle . \end{aligned} \tag{57}$$

Inserting this result back into equation (55) yields

$$\begin{aligned} &\frac{1}{2}\Big(|0\rangle \otimes \left(\mathbb{1} + U_{\vec{\phi}}\right) |0\rangle \otimes |0\rangle \otimes |u\rangle + |1\rangle \otimes \left(\mathbb{1} - U_{\vec{\phi}}\right) |0\rangle \otimes |0\rangle \otimes |u\rangle\Big) \\ &= \frac{1}{2}\Big(|0\rangle \otimes (1 + (1 - 2\varphi_n)) |0\rangle \otimes |0\rangle \otimes |u\rangle + |1\rangle \otimes (1 - (1 - 2\varphi_n)) |0\rangle \otimes |0\rangle \otimes |u\rangle\Big) \\ &= \frac{1}{2}\Big((2 - 2\varphi_n) |0\rangle + 2\varphi_n |1\rangle\Big) |0\rangle \otimes |0\rangle \otimes |u\rangle \\ &= \Big((1 - \varphi_n) |0\rangle + \varphi_n |1\rangle\Big) |0\rangle \otimes |0\rangle \otimes |u\rangle \end{aligned} \tag{58}$$

as the action of the quantum circuit in Figure 12. Lastly, measuring the first ancilla qubit in the computational basis gives the bit estimate with certainty, since its final state is

$$(1 - \varphi_n) |0\rangle + \varphi_n |1\rangle = \begin{cases} |0\rangle & \varphi_n = 0 \\ |1\rangle & \varphi_n = 1. \end{cases} \tag{59}$$

This concludes the proof of the base case.

Induction step. Assume that the theorem holds for iteration $j + 1$, so that at the end of it $\theta = 0.\varphi_{j+2}\varphi_{j+3}...\varphi_n$. At the beginning of the subsequent iteration j, θ is shifted and becomes $\theta = 0.0\varphi_{j+2}...\varphi_n$. The singular value in iteration j is

$$\begin{aligned} \sigma_j(\theta) = |\cos(\pi(2^j\varphi - \theta))| &= |\cos(\pi(0.\varphi_{j+1}\varphi_{j+2}...\varphi_n - 0.0\varphi_{j+2}...\varphi_n))| \\ &= \left|\cos\left(\frac{\varphi_{j+1}\pi}{2}\right)\right| = \begin{cases} 1 & \varphi_{j+1} = 0 \\ 0 & \varphi_{j+1} = 1. \end{cases} \end{aligned} \tag{60}$$

As a result of the specific shape of the singular value $\sigma_j(\theta)$ and θ being updated accordingly by the algorithm, one obtains the same direct relation between $\sigma_j(\theta)$ and φ_{j+1} as in the base case (53). Hence, we can mirror the proof from the base case by substituting $A_{n-1}(\theta)$ for $A_j(\theta)$ to

confirm that the quantum circuit depicted in Figure 11 delivers the desired phase estimate. This completes the proof for $m \leq n$. □

(ii) $n < m$

The second case becomes relevant if the algorithm performs less iterations than there are digits of the phase φ. This can be due to limited qubit resources, restricting the number of iterations, or too many – possibly infinitely many – digits of φ. Consequently, the algorithm cannot give an exact phase result. Instead, it implements a rounding rule by construction, generating an n-bit estimate of φ.

The rounding rule in the binary system works precisely as in the decimal system. Granted one wants to round a number between 0 and 1 in binary fraction representation to the digit j, then the rule is as follows: If the digit in place $j+1$ is 0, then the estimate is given by simply cutting off the number after digit j. In contrast, if the digit is 1, then the estimate is obtained by rounding up: If digit j is equal to 0 it is increased to take value 1, and if it is already equal to 1, it takes on value 0 but the rounding up gets carried over to the next bit in place $j-1$. In case there are multiple successive digits of 1, the rounding carries through until a digit is 0. Lastly, all digits after and including $j+1$ are discarded. The rounding rule gives rise to an upper bound of the error of the estimate, as stated by the following theorem.

Theorem 6 ([1])**.** *If $n < m$ (including $m \to \infty$) and the sign function can be implemented exactly by QSVT, then at the end of the algorithm $|\theta - \varphi| \leq 2^{-n-1}$.*

The statement follows from the following theorem when choosing the last iteration $j = 0$, which is proven by induction.

Theorem 7 ([1])**.** *Let φ be an m-bit number such that $n < m$ (including $m \to \infty$). Furthermore, let $\tilde{r}_j.\tilde{\varphi}_{j+1}\tilde{\varphi}_{j+2}...\tilde{\varphi}_n$ denote that $2^j\varphi = 0.\varphi_{j+1}\varphi_{j+2}...\varphi_n...\varphi_m$ was rounded to $n-j$ decimals, such that $\tilde{r}_j$ before the decimal point is possibly 1 if rounding was carried over. Then after iteration j the current estimate of φ is $\theta = 0.\tilde{\varphi}_{j+1}\tilde{\varphi}_{j+2}...\tilde{\varphi}_n$ and, after each iteration, the rounded φ approximates the original $n-j$ decimals of interest of φ by $|\tilde{r}_j.\tilde{\varphi}_{j+1}\tilde{\varphi}_{j+2}...\tilde{\varphi}_n - 0.\varphi_{j+1}\varphi_{j+2}...\varphi_n...\varphi_m| \leq 2^{j-n-1}$.*

Theorem 7 reveals that, when $n < m$, the algorithm generates $2^j\varphi$ rounded to $n-j$ digits (mod 1) in each iteration j. We outline the key aspects of its proof here, simultaneously giving an insight on how the algorithm carries out the rounding by construction. For the detailed case-by-case analysis, see section 8.2 of the appendix.

When $n < m$, $\sigma_j(\theta)$ does not take a simple form as in equation (53) anymore due to the contribution of the trailing digits of $2^j\varphi$, namely $\varphi_{j+2}...\varphi_m$. Since they each take arbitrary values of $\{0,1\}$, $\sigma_j(\theta)$ now covers a continuous range of $[0,1]$. This will ultimately be accounted for by having selected the sign function as a target function. The following graphic illustrates how $\sigma_j(\theta)$ depends on the argument of the cosine function as defined by equation (51).

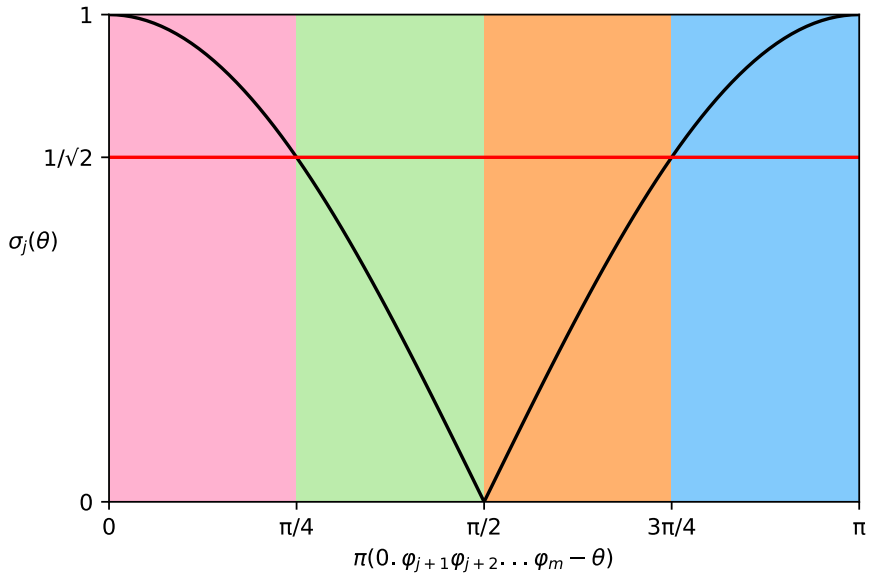


Figure 13: Dependence of the singular value $\sigma_j(\theta)$ on $2^j\varphi$ and θ. The colored regions mark the four possible combinations of $\varphi_{j+1}\varphi_{j+2}$, dictating the value of $\sigma_{j-1}(\theta)$. From left to right: 00, 01, 10, 11.

The inductive proof of theorem 7 starts with the base case, $j = n-1$, where $2^j\varphi = 0.\varphi_n\varphi_{n+1}...\varphi_m$ and $\theta = 0.0$. It transpires that the only digits determining the value of $\sigma_{n-1}(\theta)$ are φ_n and φ_{n+1}. This is because the contributions of the remaining digits are small enough to not increase the value of $2^{n-1}\varphi$ to or above the next multiple of $\pi/4$. As this is directly related to how the singular values stand in comparison to $\frac{1}{\sqrt{2}}$, it affects the value of the sign function and thus ultimately the bit estimate. We have marked all possible combinations $\varphi_n\varphi_{n+1}$ in Figure 13 by colored bars. The generation of the according bit estimates is also retraced in the table below:

$\varphi_n\varphi_{n+1}$	$2^{n-1}\varphi \in$	$\sigma_{n-1}(\theta) = \lvert\cos(\pi(2^{n-1}\varphi))\rvert \in$	$\Theta\left(\sigma_{n-1}(\theta) - \frac{1}{\sqrt{2}}\right)$	Generated estimate φ_n
00	$\left[0, \frac{1}{4}\right)$	$\left(\frac{1}{\sqrt{2}}, 1\right]$	1	0
01	$\left[\frac{1}{4}, \frac{1}{2}\right)$	$\left(0, \frac{1}{\sqrt{2}}\right]$	-1 or 0	1 or indefinite
10	$\left[\frac{1}{2}, \frac{3}{4}\right)$	$\left[0, \frac{1}{\sqrt{2}}\right)$	-1	1
11	$\left[\frac{3}{4}, 1\right)$	$\left[\frac{1}{\sqrt{2}}, 1\right)$	1 or 0	0 or indefinite

Table 1: The four possible combinations of $\varphi_n\varphi_{n+1}$ and the corresponding steps towards the generation of the bit estimate φ_n, given that $n < m$ while m is finite. If m is infinite, all domains are closed. If $\sigma_{n-1}(\theta) = \frac{1}{\sqrt{2}}$, the bit estimate is not generated with certainty, thus labeled "indefinite". For further information on how to remedy this, see section 8.2 of the appendix.

Note that employing $\Theta\left(\sigma_{n-1}(\theta) - \frac{1}{\sqrt{2}}\right)$ as a target function ensures the generation of the correct bit estimate, even if $\sigma_{n-1}(\theta)$ is continuous when $n < m$: By comparing the outer two columns of Table 1, one can see that the generated estimate equals the result of rounding $\varphi_n\varphi_{n+1}$ to

one digit. This is exactly as desired according to the rounding rule. The proof details in the appendix confirm this more rigorously. We also prove the general case for $n < m$ there, since it resembles the base case's proof.

5.3 Caveats

Owing to equation (54), one in principle desires the QSVT procedure within the QPE algorithm to implement the sign function with a shifted argument by $\frac{1}{\sqrt{2}}$ as a target function. Nonetheless, one can only achieve polynomial transformations with QSVT by definition (see theorem 3). Since theorems 4 and 6 prove that the algorithm works under the premise that the sign function can be implemented exactly by QSVT, it is natural to question the algorithm's functionality. Luckily, one can find a suitable polynomial to approximate the sign function, following the approach of Low [30]. Low makes use of a specific function called the error function (see definition 6) that approximates the sign function, and then polynomially approximates the error function itself. How the polynomial is constructed exactly is outlined in section 8.1 of the appendix. As a result, we obtain an arbitrarily precise ϵ-approximation of the shifted sign function by the phase estimation polynomial $P^{PE}_{\epsilon,\Delta}(x)$ for $x \in [-1, -\frac{\Delta}{2} + \frac{1}{\sqrt{2}}] \cup [\frac{\Delta}{2} + \frac{1}{\sqrt{2}}, 1]$ after fixing a small Δ. The remaining interval of length Δ centered around $\frac{1}{\sqrt{2}}$ might seem to impose a problem at first glance because we are not guaranteed a working polynomial approximation there. This is to be discussed in the remainder of this chapter. Using the phase estimation polynomial remedies the lack of an exact sign function implementation, but at the cost of introducing additional constraints.

To specify these constraints further, we delve into the output of the algorithm. A run is considered successful if it calculates a phase estimate θ such that the error $|\theta - \varphi| \leq 2^{-n-1}$, as stated in theorem 6. Conversely, an error higher than this is labels the run a failure. Notably, the algorithm bears a non-zero failure probability ρ which is introduced by the approximations made in the construction of the phase estimation polynomial $P^{PE}_{\varepsilon,\Delta}$. In accordance, ρ is determined by the input parameters Δ and ε.

Fortunately, the error can be mitigated by abiding to certain bounds. To begin with, a non-zero Δ coincides with an interval where the phase estimation polynomial is not a good approximation to the shifted sign function. However, if Δ is chosen small enough, one can find an acceptable upper bound on the induced error of the first iteration:

Theorem 8 ([1])**.** *If Δ is chosen such that it satisfies*

$$\Delta < \left(\cos\left(\frac{3\pi}{16} - \frac{1}{\sqrt{2}}\right)\right) \approx 0.25, \tag{61}$$

then $\sigma_j \in [-\frac{1}{\sqrt{2}} - \frac{\Delta}{2}, \frac{1}{\sqrt{2}} + \frac{\Delta}{2}]$, i.e., the polynomial approximation failing, is only possible at iteration $j = n - 1$. If an error happens at this iteration, then at the end of the algorithm $|\theta - \varphi| \leq 1/2^n$, assuming no more errors are made at later iterations.

The proof of theorem 8 can be found in appendix B2 of [1]. From here on, we will assume that Δ is chosen such that theorem 8 is fulfilled.

Secondly, a non-zero ε dictates how the phase estimation polynomial deviates from the ideal outside of $[-\frac{1}{\sqrt{2}} - \frac{\Delta}{2}, \frac{1}{\sqrt{2}} + \frac{\Delta}{2}]$ and is thus another source of error. The following closer examination allows to set ε depending on the desired success probability of the algorithm. In $j < n - 1$,

$\sigma_j \notin [-\frac{1}{\sqrt{2}} - \frac{\Delta}{2}, \frac{1}{\sqrt{2}} + \frac{\Delta}{2}]$ by cause of theorem 8. We select the case $\sigma_j < -\frac{1}{\sqrt{2}} - \frac{\Delta}{2}$; the other case follows analogously. Consequently, σ_j is within the range where $P^{PE}_{\varepsilon,\Delta}(x)$ ε-approximates the negative side of the sign function and, according to equation (54), the correct bit measurement would yield $|1\rangle$. Nonetheless, there exists a non-zero failure probability ρ_j, i.e. the probability of measuring $|0\rangle$, which can be determined exactly.

Iteration j brings the control ancilla qubit to the (unnormalized) state

$$\frac{1}{2}\Big(\big(1 - P^{PE}_{\varepsilon,\Delta}(\sigma_j)\big)\,|0\rangle + \big(1 + P^{PE}_{\varepsilon,\Delta}(\sigma_j)\big)\,|1\rangle\Big), \tag{62}$$

which can be seen by comparing to equations (57) and (58) when substituting the sign function by the phase estimation polynomial. In consonance with the laws of quantum mechanics, the probability of measuring $|1\rangle$ in the computational basis thus yields:

$$\rho_j = \left| \frac{\frac{1}{2}\big(1 - P^{PE}_{\varepsilon,\Delta}(\sigma_j)\big)}{\sqrt{\left|\frac{1}{2}\big(1 + P^{PE}_{\varepsilon,\Delta}(\sigma_j)\big)\right|^2 + \left|\frac{1}{2}\big(1 - P^{PE}_{\varepsilon,\Delta}(\sigma_j)\big)\right|^2}} \right|^2 = \frac{1}{2}\frac{\big(1 + P^{PE}_{\varepsilon,\Delta}(\sigma_j)\big)^2}{1 + \big(P^{PE}_{\varepsilon,\Delta}(\sigma_j)\big)^2}. \tag{63}$$

An upper bound for this expression can now be found using the ε-bound of $P^{PE}_{\varepsilon,\Delta}(x)$ in the current domain of interest, $-1 \leq P^{PE}_{\varepsilon,\Delta}(\sigma_j) \leq -1 + \varepsilon$:

$$\rho_j = \frac{1}{2}\frac{\big(1 + P^{PE}_{\varepsilon,\Delta}(\sigma_j)\big)^2}{1 + \big(P^{PE}_{\varepsilon,\Delta}(\sigma_j)\big)^2} \leq \frac{1}{2}\frac{\varepsilon^2}{1 + \big(-1 + \varepsilon\big)^2} \leq \frac{1}{2}\varepsilon^2 \quad \forall j \in \{n-2, \ldots 0\}. \tag{64}$$

The total failure probability adds up to the sum of failure probabilities in the individual iterations. For $j = n - 1$, which was excluded from the analysis above, one can note that if $\sigma_j \in [-\frac{1}{\sqrt{2}} - \frac{\Delta}{2}, \frac{1}{\sqrt{2}} + \frac{\Delta}{2}]$, the choice of Δ as in theorem 8 already guarantees the success of the algorithm. If $\sigma_j \notin [-\frac{1}{\sqrt{2}} - \frac{\Delta}{2}, \frac{1}{\sqrt{2}} + \frac{\Delta}{2}]$, then the same argument applies as for all other iterations and therefore $\rho_{n-1} \leq \frac{1}{2}\varepsilon^2$. In conclusion, the total failure probability yields $\rho = n\varepsilon^2/2$. Most importantly, from this expression follows that a choice of $\varepsilon \leq \sqrt{2\delta/n}$ ensures the success of the algorithm with probability at least $1 - \delta$, i.e. $\rho \leq \delta$. We can thus treat δ, the desired upper bound on the failure probability, as an input parameter of the algorithm. Combined with the bound for Δ, the algorithm gives an n-bit estimate θ of φ satisfying $|\theta - \varphi| \leq 1/2^n$ with probability $1 - \delta$. Note that theorem 6 originally proposes $|\theta - \varphi| \leq 1/2^{(n+1)}$ as an error bound. The polynomial approximation hence successfully remedies the lack of an exact sign function implementation, at the cost of sacrificing one digit of precision.

Now that we have completed the discussion of the algorithm, it is fully summarized below.

Algorithm 1: QSVT-based QPE algorithm. Inspired by [1].

Input : Access to controlled-U^{2^j} operations, an eigenstate $|u\rangle$ of U with eigenvalue $e^{i2\pi\varphi}$, and n ancilla qubits to control a QSVT procedure on each. δ, $\epsilon \leq \sqrt{2\delta/n}$ and $\Delta < \left(\cos\left(\frac{3\pi}{16} - \frac{1}{\sqrt{2}}\right)\right) \approx 0.25$.

Output : With success probability at least $1-\delta$, the algorithm gives an n-bit estimate θ of φ such that $|\theta - \varphi| \leq 1/2^n$.

1 Initialization of $\theta = 0$.
2 Apply a Hadamard gate to each ancilla qubit.
3 **for** $j = n-1$ down to 0 **do**
3.1 Shift the digits of theta: $\theta/2 \rightarrow \theta$.
3.2 Perform a QSVT circuit:
Construct a block encoding of $A_j(\theta)$.
Perform the polynomial transformation $P^{PE}_{\epsilon,\Delta}$ of $A_j(\theta)$, conditioned on one ancilla qubit.
3.3 Apply a Hadamard gate to the prior mentioned ancilla qubit.
3.4 Measure the ancilla qubit in the computational basis.
3.5 Update $\theta = 0.\theta_1\theta_2...\theta_{n-j}$ such that θ_1 takes on the value of the measurement outcome, φ_{j+1}.

Despite our proposed algorithm being heavily inspired by [1], we adapted a minor change worth commenting on. The original algorithm executes two additional steps after ending the for-loop in order to generate a bit estimate for the 2^0-digit of φ. This extra step also demands the availability of another ancilla qubit. By disregarding this here, we obtain a phase estimate mod 1, conforming to theorem 7. At the end of the day, our QSVT-based algorithm is one example of various alternative phase estimation algorithms and it remains the choice of the user which one to select over another. We justify our changes here with the increasing conceptual simplicity, confirmed by the main author of the paper [31].

5.4 Implementation

To conclude this chapter on the QSVT-based QPE algorithm, we discuss how it is implemented on a quantum device in preparation for the ensuing cost analysis in the next chapter. Since $A_j(\theta)$ is not unitary, a block encoding is needed for its implementation. Recall that $A_j(\theta) = \frac{1}{2}(\mathbb{1} + e^{-2\pi i\theta}U^{2^j})$. One can confirm that

$$W_j(\theta) = \frac{1}{2}\begin{pmatrix} \mathbb{1} + e^{-2\pi i\theta}U^{2^j} & \mathbb{1} - e^{-2\pi i\theta}U^{2^j} \\ \mathbb{1} - e^{-2\pi i\theta}U^{2^j} & \mathbb{1} + e^{-2\pi i\theta}U^{2^j} \end{pmatrix} \tag{65}$$

is indeed a valid $(2,1,0)$-block encoding of $A_j(\theta)$ according to definition 4 by checking $\tilde{\Pi}W_j(\theta)\Pi = A_j(\theta)$ with the projectors $\Pi = \tilde{\Pi} = |0\rangle\langle 0| \otimes \mathbb{1}$. On a quantum device, we accomplish the implementation of $W_j(\theta)$ by the following circuit in Figure 14.

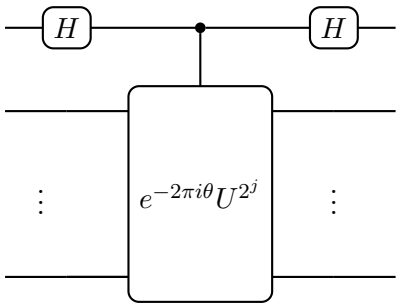


Figure 14: Desired implementation of the block encoding $W_j(\theta)$ of $A_j(\theta)$ [1].

Recall from chapter 5.1 that there are multiple ways to implement the algorithm. Our description so far has assumed the availability of multiple ancilla qubits on which the digits of the phase estimate are generated and measured, although one could also reuse one ancilla qubit for the same purpose. In both cases, the quantum superposition state is destroyed by the measurement wherefore the information about the previously estimated digits, stored within θ, has to be classically fed forward to the next iteration in order to prepare $e^{-2\pi i\theta}U^{2^j}$ as above.

Alternatively, we can slightly alter the algorithm design to enable the quantum feedforward of θ. While still employing multiple ancilla qubits, all measurements are shifted to the end of the quantum circuit, so that the individual estimated phase digits remain encoded on the ancilla space. This way, gates controlled by the ancilla qubits can be used to feed θ forward to subsequent iterations by design. Hence, we adopt the implemention of $W_j(\theta)$ by a more complex circuit using the operators R_k previously defined in equation (42). The resulting quantum circuit is depicted below.

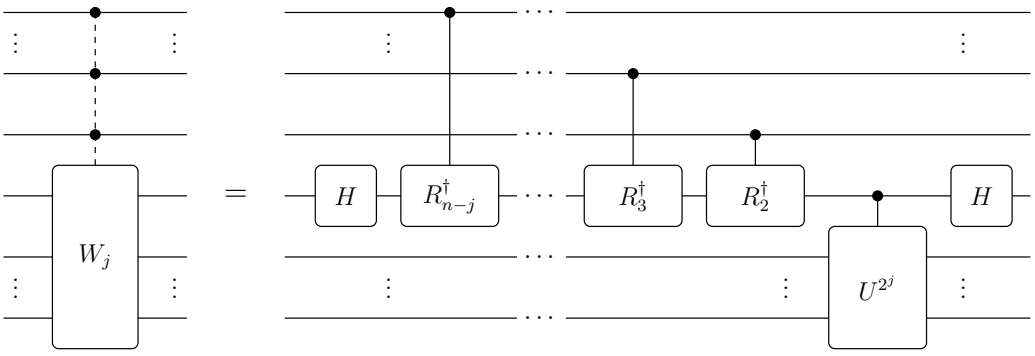


Figure 15: Implementation of the block encoding $W_j(\theta)$ of $A_j(\theta)$ [1]. The dotted controls on the left side indicate the multiple controlled-R_k gates within the circuit, while expressing that W_j as a whole is not to be understood as a controlled gate. This circuit assumes the bottom $1+l$ qubits to be initialized in the $|0\rangle \otimes |u\rangle$ -state, even though not pictured here.

Finally, we provide the full circuit diagram of the QSVT- based QPE algorithm using multiple ancilla qubits and the quantum feedforward of the information from previous iterations as per Figure 15. This circuit will form the base of the computational cost analysis in the following chapter.

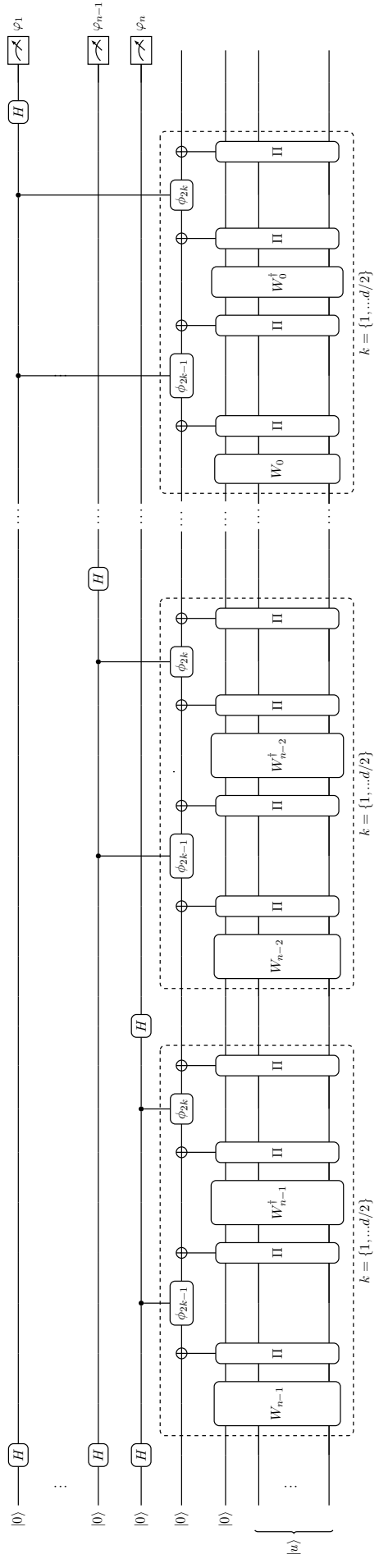


Figure 16: Full circuit diagram of the QSVT-based QPE algorithm. Inspired by [1]. The QSVT procedures are marked by the dashed boxes, assuming d is even. The block encodings W_j are implemented as in Figure 15, omitting the dashed controls of W_j here.

6 Cost Analysis of a QSVT-Based QPE Algorithm

After the iterative quantum phase estimation algorithm by Martyn et al. was explored in detail, this chapter takes a step further and inspects the computational cost of the algorithm. Along the way we also summarize the underlying assumptions that were made throughout the thesis. First, we study the qubit resources required for running the algorithm (chapter 6.1). Secondly, there follows a quantitative examination of its computational cost (chapter 6.2). For this, the algorithm is decomposed up to the oracle U, while still taking contributions of the single- and two-qubit gates into account to achieve a more accurate lower bound on the total gate complexity. Hence, the chosen measure of computational cost can be regarded as a combination of gate complexity and query complexity, although we will refer to it as gate complexity in the following. This chapter predominantly contains original work, based on the findings in the previous chapters. Hence, the quantum circuit in Figure 16, adapted from [1], serves as a main reference.

6.1 Qubit Resources

We have seen that the number of ancilla qubits dictates the amount of digits of the resulting phase estimate. Apart from these n ancilla qubits, there are more to be taken into account. Every QSVT protocol is implemented by alternating rotations, controlled by one ancilla qubit (see Figure 6). As this control qubit can be reused for all QSVT procedures, it only adds one to the total qubit count. Lastly, there is one additional ancilla qubit that originates from the block encoding $W_j(\theta)$ as seen in Figure 15, and can again be reused for all iterations. In total, this results in $n + l + 2$ qubits needed to run the algorithm, as seen in the full circuit diagram (Fig. 16).

6.2 Gate Complexity

Estimating the gate complexity is crucial when it comes to determining if a quantum algorithm is implementable and also to see if it provides an advantage over already established algorithms. Hereafter, the gate complexity of algorithm is broken down in terms of one-qubit gates, two-qubit gates and the oracle U, as it is not possible to decompose the latter gate more without making further assumptions about it. The cost of a gate is denoted by $\mathcal{C}(...)$ and $\mathcal{C}^1$ and $\mathcal{C}^2$ refer to the cost of one- and two-qubit gates, respectively. While neglecting contributions of the measurements, we give a lower bound of the gate complexity of the quantum circuit in Figure 16.

To begin with, as seen on the far left side of the circuit diagram (Fig. 16), we assume the ancilla register to be prepared in $|0\rangle^{\otimes(n+2)}$ and the remaining l qubits to be prepared in the eigenstate $|u\rangle$. This state preparation comes with a cost, however this is not subject of our investigation in this thesis. For one possible approach to state preparation see e.g. [32].

After the preparation of states, a Hadamard gate is applied to each of the n ancilla qubits directly after the initialization of the algorithm, as well as after performing the QSVT protocol. This generates a cost of $2n$ one-qubit gates, i.e. $2n\,\mathcal{C}^1$.

6.2.1 Gate Complexity of the QSVT Sequence

Furthermore, there is one QSVT procedure in each iteration of the algorithm of which we are to determine the gate complexity hereinafter. Specifically, note that it depends on the iteration. As per the circuit diagram in Figure 16, the QSVT procedure within iteration j executes all of the

following gates $d/2$ times: An application of each $W_j(\theta)$ and $W_j^\dagger(\theta)$, four projector-controlled-NOT gates (C_Π-NOT), and two Z-rotation gates by angle ϕ each controlled by an ancilla qubit, denoted by C-ϕ. The gate complexity of the QSVT circuit in iteration j hence becomes

$$\mathcal{C}(\mathrm{QSVT}_j) = \frac{d}{2}\left[2\mathcal{C}(C\text{-}\phi) + 4\mathcal{C}(C_\Pi\text{-NOT}) + \mathcal{C}(W_j(\theta) + \mathcal{C}(W_j^\dagger(\theta))\right]. \tag{66}$$

d stands for the degree of the phase estimation polynomial with the corresponding parameters Δ and ε, given as

$$d = \mathcal{O}\left(\frac{1}{\Delta}\log(1/\varepsilon)\right) \tag{67}$$

by theorem 11 in section 8.1 of the appendix. We assume that the exact d is to be found prior to running the algorithm.

To specify $\mathcal{C}(\mathrm{QSVT}_j)$ further, we investigate the contributions of its individual constituents from left to right according to equation (66). To begin with, the cost of a one-qubit-controlled rotation gate C-ϕ is counted as a two-qubit gate, $\mathcal{C}(C\text{-}\phi) = \mathcal{C}^2$.

6.2.1.1 Gate Complexity of the Projector-Controlled Phase Shift Gate

Next, the cost of the projector-controlled-NOT gates is investigated. With its definition from equation (29) and $\Pi = |0\rangle\langle 0| \otimes \mathbb{1}_l$, its action can be rewritten as

$$\begin{aligned} C_\Pi\text{-NOT} &= X \otimes \Pi + \mathbb{1}_1 \otimes (\mathbb{1}_{l+1} - \Pi) = X \otimes |0\rangle\langle 0| \otimes \mathbb{1}_l + \mathbb{1}_1 \otimes (\mathbb{1}_1 \otimes \mathbb{1}_l - |0\rangle\langle 0| \otimes \mathbb{1}_l) \\ &= (X \otimes |0\rangle\langle 0| + \mathbb{1}_1 \otimes |1\rangle\langle 1|) \otimes \mathbb{1}_l, \end{aligned} \tag{68}$$

where $\mathbb{1}_t$ refers to the identity operation on t qubits. We have factored out the action of C_Π-NOT on the l-qubit subspace to see that it is the identity operation, whilst the first two qubits undergo a C-NOT operation, conditioned on the second qubit being in the $|0\rangle$ state. Therefore, we count the cost of the C_Π-NOT as a two-qubit gate here, $\mathcal{C}(C_\Pi\text{-NOT}) = \mathcal{C}^2$.

6.2.1.2 Gate complexity of the Block Encoding

According to Figure 15, the implementation of the block encoding W_j is comprised of two Hadamard gates, $n-j-1$ one-qubit-controlled R_k gates and a one-qubit-controlled U^{2^j} gate. The Hadamard gates are again regarded as single-qubit gates and the one-qubit-controlled R_k gates as two-qubit gates. In contrast, analyzing the cost of the controlled U^{2^j} gate is more involved.

In this thesis we assume the cost of all powers of U to be $\mathcal{C}(U^{2^j}) = \mathcal{C}(U)\,\forall j$, as e.g. proposed in [33]. The reasonability of this assumption can be justified for instance if U has some special structure, such that its matrix elements have a unified description by some function f. Then one may assume that U^{2^j} can be obtained efficiently by another function $g(f)$, leading to the same cost for all powers of U.

However, we are not able to specify the additional cost of controlling U^{2^j} without more information about U. We thus ultimately provide a lower bound of the cost of implementing W_j given by

$$\mathcal{C}(W_j) = 2\mathcal{C}(H) + (n-j-1)\mathcal{C}(C\text{-}R_l) + \mathcal{C}(U^{2^j}) = 2\mathcal{C}^1 + (n-j-1)\mathcal{C}^2 + \mathcal{C}(U). \tag{69}$$

Furthermore, we assign the same cost to the implementation of $W_j^\dagger$ as to W_j. It is generally known that taking the adjoint of a sequence of operators yields the order-reversed sequence

of the adjoint operators, respectively ($(U_1U_2...U_k)^\dagger = U_k^\dagger...U_2^\dagger U_1^\dagger$). Consequently, the cost of implementing the adjoint sequence of a composite gate is comprised of the sum of costs of its adjoint constituents. The Hadamard gate is self-adjoint and the adjoint of a controlled-R_l gate can be shown to become another controlled-Z-rotation gate. Hence, it is reasonable to state $\mathcal{C}(W_j^\dagger) = \mathcal{C}(W_j)$, based on the accepted premise that implementing U and its adjoint $U^\dagger$ is equally costly.

6.2.2 Summary

In conclusion, the total gate complexity of the QSVT-based QPE algorithm implemented as in Figure 16 and inspired by [1] amounts to

$$\begin{aligned}
\mathcal{C}(\text{QPE}) &= 2n\mathcal{C}(H) + \sum_{j=0}^{n-1} \mathcal{C}(\text{QSVT}_j) \\
&= 2n\mathcal{C}^1 + \sum_{j=0}^{n-1} \frac{d}{2} \left[2\mathcal{C}(C\phi) + 4\mathcal{C}(C_\Pi\text{-NOT}) + 2\mathcal{C}(W_j(\theta))\right] \\
&= 2n\mathcal{C}^1 + \frac{d}{2}\left[2n\mathcal{C}(C\phi) + 4n\mathcal{C}(C_\Pi\text{-NOT}) + 2\sum_{j=0}^{n-1}\mathcal{C}(W_j(\theta))\right] \\
&= 2n\mathcal{C}^1 + d\left[n\mathcal{C}^2 + 2n\mathcal{C}^2 + \sum_{j=0}^{n-1}\left(2\mathcal{C}^1 + (n-j-1)\mathcal{C}^2 + \mathcal{C}(U)\right)\right] \\
&= 2n\mathcal{C}^1 + d\left[n\mathcal{C}^2 + 2n\mathcal{C}^2 + 2n\mathcal{C}^1 + n\mathcal{C}(U) + \left(\sum_{j=0}^{n-1}(n-j-1)\right)\mathcal{C}^2\right] \\
&= 2n\mathcal{C}^1 + d\left[n\mathcal{C}^2 + 2n\mathcal{C}^2 + 2n\mathcal{C}^1 + n\mathcal{C}(U) + \left(\frac{1}{2}(n-1)n\right)\mathcal{C}^2\right] \\
&= (2n(d+1))\mathcal{C}^1 + dn\left(3 + \frac{1}{2}(n-1)\right)\mathcal{C}^2 + dn\mathcal{C}(U)
\end{aligned} \tag{70}$$

where d is the degree of the phase estimation polynomial, and n is given as an input to the algorithm.

Recall that we have based our cost analysis on the circuit in Figure 16, which assumed even d. Fortunately, $\mathcal{C}(\text{QSVT}_j)$ and therefore also the total gate complexity remain the same for odd d. This is because according to the QSVT theorem (3), a general odd QSVT circuit contains $(d-1)/2$ $\tilde{\Pi}_\phi$- and U-operations and $(d+1)/2$ Π_ϕ- and $U^\dagger$-operations (where this U here denotes a general projected unitary encoding as in the QSVT theorem instead of the QPE unitary). This generates a gate complexity of $\mathcal{C}(\text{QSVT}_j) = \frac{d-1}{2}(\mathcal{C}(\tilde{\Pi}_\phi) + \mathcal{C}(W_j)) + \frac{d+1}{2}(\mathcal{C}(\Pi_\phi) + \mathcal{C}(W_j^\dagger))$. However, since we have adopted $\mathcal{C}(W_j) = \mathcal{C}(W_j^\dagger)$ and $\tilde{\Pi}_\phi = \Pi_\phi$, the expression for $\mathcal{C}(\text{QSVT}_j)$ boils down to $d(\mathcal{C}(\Pi_\phi)+\mathcal{C}(W_j))$. This can be proven to be the same as for even d by reformulating equation 66 using $\mathcal{C}(C\text{-}\phi) + 2\mathcal{C}(C_\Pi\text{-NOT}) = \mathcal{C}(\Pi_\phi)$ and $\mathcal{C}(W_j) = \mathcal{C}(W_j^\dagger)$.

The underlying assumptions of this central result can be summarized to the availability of $n+l+2$ qubits for running the algorithm and access to controlled powers of U, $\mathcal{C}(U^{2^j}) = \mathcal{C}(U)\,\forall j$ and $\mathcal{C}(U) = \mathcal{C}(U^\dagger)$. It provides a lower bound on the gate complexity that neglects the costs of preparing initial qubit states, controlling the powers of U, finding the exact polynomial degree d,

as well as the n measurements.

Note that the last term of the gate complexity in equation (70), $dn\mathcal{C}(U)$, describes the query complexity of the QSVT-based QPE algorithm. Inserting theorem 8 and $\varepsilon \leq \sqrt{2\delta/n}$ into equation (67) thus yields $d = \mathcal{O}(\log(\frac{1}{\varepsilon}))$. Consequently, the query complexity of the algorithm scales as $\mathcal{O}(n \log(\frac{1}{\varepsilon}))$, making it competitive with other variants of QPE under the same assumptions.

7 Conclusion and Outlook

In this thesis we introduced the quantum singular value transformation as an extension of quantum signal processing. Although it is considered a "grand unification of quantum algorithms"[1], our focus was directed towards its application to quantum phase estimation. QPE is a crucial subroutine used in many quantum algorithms for various purposes that aims to estimate the phase of an eigenvalue to a given unitary. While adapting minor changes to it for enhanced simplicity, we elucidated the QSVT-based QPE algorithm proposed by Martyn et al. in [1]. Significantly, this example of an iterative phase estimation approach employs QSVT to encode the individual digits of the phase estimate on the ancilla subspace. This is enabled by the cleverly chosen singular values of the block encoded matrix A_j that are transformed according to QSVT with a target polynomial approximating the sign function. The ensuing quantum measurements read out the estimated digits and store them within a regularly updated classical parameter that feeds forward to the next iteration by design. Eventually, an estimate of the desired phase is found with a success probability related to the additional parameters introduced by the polynomial approximation to the sign function. We were able to specify a lower bound for the gate complexity of the algorithm depending on the degree of the phase estimation polynomial, the number of ancilla qubits available and the cost of the unitary U (see equation 70). From this equation we were also able to infer the query complexity that scales as $\mathcal{O}(n \log(\frac{1}{\varepsilon}))$ and reveals that this QPE algorithm is competitive with other non-QSVT-based versions. Moreover, we summarized the needed qubit resources and displayed the underlying assumptions giving rise to the remaining overhead to the gate complexity.

Moving forward, it might be interesting to investigate changes of the algorithm's output if the lower part of the quantum register is not assumed to be prepared in the eigenstate of U, but rather in a general superposition state. As discussed for the standard QPE version, we expect this to have an impact on the success probability of the algorithm. This reflects how it is in practice often unrealistic to prepare a quantum system in an exact eigenstate [10].

Moreover, one could try to specify the cost of implementing powers of U further, possibly leading to a more concrete lower bound for the gate complexity. For this purpose, a commonly made assumption is, for instance, U being a sparse matrix [4]. On the other hand, if one does not want to assume properties of U, one may explore the idea of a "quantum random access memory (QRAM)", following e.g. the proposal in [34]. The quantum memory refers to an additional qubit register where the powers of U applied to appropriate initial states are prepared in advance on. This way, they can be called by the algorithm. QRAM is said to have the potential to possibly accelerate existing algorithms [34] and is therefore one example of the many intriguing branches of research of quantum computation that can be further looked into.

8 Appendix

8.1 Construction of the Phase Estimation Polynomial

This subsection examines the phase estimation polynomial used in the QPE algorithm depicted in Figure 1. As mentioned in chapter 5.3 we follow the work of Low [30]. For the detailed underlying proofs of the subsequent arguments, please refer to the original source.

Recall that the QSVT circuit within the QPE algorithm prepares the information of the current digit φ_j on the corresponding ancilla qubit by using equation (54) and implementing the sign function $\Theta\left(x - \frac{1}{\sqrt{2}}\right)$. Generally speaking, Low makes use of the error function to approximate the sign function. The error function is a standard mathematical function that is defined as follows.

Definition 6 (Error function [35])**.**

$$\operatorname{erf}(kx) = \frac{2}{\sqrt{\pi}} \int_0^{kx} e^{-y^2} dy$$

For the construction of the polynomial, the following theorem will be crucial :

Theorem 9 (Approximation of the sign function with the error function; Lemma 6.19 in [30])**.** $\forall \Delta > 0, \varepsilon \in (0, \sqrt{2/(e\pi)}]$*, the error function satisfies*

$$\varepsilon \geq \max_{|x| \geq \Delta/2} |\operatorname{erf}(kx) - \Theta(x)|$$

for $k = \frac{\sqrt{2}}{\Delta} \log^{\frac{1}{2}}(\frac{2}{\pi\varepsilon^2})$.

Essentially, this lemma states that the sign function can be approximated to arbitrary precision ε by the error function outside of a symmetric interval around zero whose length is determined by a positive parameter Δ[9]. The approximation remains correct albeit the arbitrary choice of Δ because the parameter k is determined by ε and Δ.

The next step is a polynomial approximation of the error function, such that combining it with theorem 9 then yields a polynomial approximation of the sign function. But observe (e.g. in equation 54) that the QPE algorithm involves a sign function that undergoes a shift of the argument by $\frac{1}{\sqrt{2}}$. Therefore, a polynomial approximation to the regular error function does not suffice. Instead, a polynomial approximation of the shifted error function is of need:

Theorem 10 (Polynomial approximation of the shifted error function $\operatorname{erf}(k(x-\delta))$; Corollary 6.26 in [30])**.** $\forall k > 0, \varepsilon \in (0, \mathcal{O}(1)], \delta \in [-1, 1]$ *the polynomial* $p_{\operatorname{erf},k,\delta,n}(x) := p_{\operatorname{erf},2k,n}((x-\delta)/2)$ *of odd degree* $n = \mathcal{O}(\sqrt{(k^2 + \log(1/\varepsilon))\log(1/\varepsilon)})$ *satisfies*

$$\varepsilon \geq \max_{x \in [-1,1]} |p_{\operatorname{erf},k,\delta,n}(x) - \operatorname{erf}(k(x-\delta))|.$$

[9]The notation Δ is chosen for consistency with [1]. In [30], this parameter is denoted by κ.

The polynomial $p_{\mathrm{erf},k,\delta,n}$ *is defined as*

$$\begin{aligned} p_{\mathrm{erf},k,n}(x) = \frac{2ke^{-k^2/2}}{\sqrt{\pi}} \big(& (I_0(k^2/2)x \\ & + \sum_{j=1}^{(n-1)/2} I_j(k^2/2)(-1)^j \left(\frac{T_{2j+1}(x)}{2j+1} - \frac{T_{2j-1}(x)}{2j-1} \right), \end{aligned} \tag{71}$$

where $I_j(x)$ *denote the modified Bessel functions of the first kind and* $T_j(x)$ *the Chebyshev polynomials of first kind.*

Connecting the two theorems above yields the desired polynomial approximation of the sign function. Let $x' := x - \delta$. Then, from theorem 10 follows that $\forall k > 0, \Delta > 0, \varepsilon \in (0, \mathcal{O}(1)], \delta \in [-1, 1]$:

$$\begin{aligned} \varepsilon &\geq \max_{x\in[-1,1]} |p_{\mathrm{erf},k,\delta,n}(x) - \mathrm{erf}(kx')| \\ &= \max_{x\in[-1,1]} |p_{\mathrm{erf},k,\delta,n}(x) - \mathrm{erf}(kx') + \Theta(x') - \Theta(x')| \\ &= \max_{x\in[-1,1]} |p_{\mathrm{erf},k,\delta,n}(x) - \Theta(x') - (\mathrm{erf}(kx') - \Theta(x'))| \\ &\geq \max_{x\in[-1,1]} \Big\{ |p_{\mathrm{erf},k,\delta,n}(x) - \Theta(x')| - |(\mathrm{erf}(kx') - \Theta(x')| \Big\} \\ &\geq \max_{x\in[-1,-\Delta/2+\delta]\cup[\Delta/2+\delta,1]} \Big\{ |p_{\mathrm{erf},k,\delta,n}(x) - \Theta(x')| - \varepsilon' \Big\}, \end{aligned} \tag{72}$$

After adding a zero in equation (72), the next step makes use of the triangular inequality, before theorem 9 comes to play. Here, the absolute value of the deviation of the error function from the sign function can be bounded by an arbitrarily small ε' in the interval $|x'| \geq \Delta/2$, $\forall \Delta > 0$. The shifted argument x' is accounted for by adapting the range of x.

In summary, the theorem below delivers the polynomial approximation to the shifted sign function for $x \in [-1, -\Delta/2 + \delta] \cup [\Delta/2 + \delta, 1]$:

Theorem 11 (Polynomial approximation of the shifted sign function; Corollary 6.27 in [30])**.** *The polynomial* $p_{\mathrm{erf},k,\delta,n}(x)$ *of odd degree of order* $n = \mathcal{O}\left(\frac{1}{\Delta}\log(1/\varepsilon)\right)$ *fulfills* $\forall \varepsilon \in (0, \mathcal{O}(1)], \delta \in [-1, 1], \Delta > 0$:

$$\varepsilon'' := \varepsilon + \varepsilon' \geq \max_{x\in[-1,-\Delta/2+\delta]\cup[\Delta/2+\delta,1]} |p_{\mathrm{erf},k,\delta,n}(x) - \Theta(x-\delta)|, \tag{73}$$

where $k = \frac{\sqrt{2}}{\Delta}\log^{\frac{1}{2}}(\frac{2}{\pi\varepsilon^2})$.

The exact degree of $p_{\mathrm{erf},k,\delta,n}(x)$ is specified by equation (6.47) in [30] and can be reconstructed by understanding the proofs leading up to theorem 10. However, we will not discuss this in detail here.

The discussion in [30] about the phase estimation polynomial construction ends with theorem 11, yet [1] appends another step to it. Note that the latter source uses an equivalent notation for the polynomial which will be used hereinafter. In this notation, the polynomial becomes $P^{\Theta}_{\varepsilon,\Delta}(x) := p_{\mathrm{erf},k,\delta,n}(x)$. The notations convey the same information since k is a function of ε and Δ and n is determined by k and ε.

The phase estimation polynomial needs to fulfill the requirements of theorem 3 for a feasible implementation within the QSVT circuit. On that account, [1] establishes the symmetrization of $P^{\Theta}_{\varepsilon,\Delta}(x)$ to guarantee definite parity. The ensuing symmetrized phase estimation polynomial is defined in [1] by

$$P^{PE}_{\varepsilon,\Delta}(x) := \frac{1}{1+\frac{\varepsilon}{4}}\left(\left(1-\frac{\varepsilon}{4}+P^{\Theta}_{\varepsilon/2,\Delta}\left(x-\frac{1}{\sqrt{2}}\right)-P^{\Theta}_{\varepsilon/2,\Delta}\left(x+\frac{1}{\sqrt{2}}\right)\right)\right). \tag{74}$$

In accordance with [1], $P^{PE}_{\varepsilon,\Delta}(x)$ behaves like $P^{\Theta}_{\varepsilon,\Delta}(x-\frac{1}{\sqrt{2}})$ for $x \geq 0$. Since the singular values that the polynomial transformation is applied to are positive by construction, the approximation succeeds and we refer to $P^{PE}_{\varepsilon,\Delta}(x)$ as the phase estimation polynomial.

8.2 Proof of Theorem 7

Here we elaborate on the details complementing the proof of theorem 7 in chapter 5.2.

Proof. **Base case** ($j = n-1$). Assume the algorithm has been initialized previously, and $m < \infty$ for now. As before, the singular value in this iteration is given by

$$\sigma_{n-1} = |\cos(\pi(2^{n-1}\varphi - \theta))| = |\cos(\pi(0.\varphi_n\varphi_{n+1}...\varphi_m - 0.0))|. \tag{75}$$

Here, we lay focus on the decomposition into the contributions of the particular bits to σ_{n-1}:

$$\sigma_{n-1} = \left|\cos\left(\pi\left(\frac{\varphi_n}{2}+\frac{\varphi_{n+1}}{4}+\sum_{k=2}^{m-n}\frac{\varphi_{n+k}}{2^{k+1}}\right)\right)\right|. \tag{76}$$

This emphasizes how the value of σ_{n-1} is determined by the bit values of φ. Specifically, we are interested in whether σ_{n-1} is greater or less than $\frac{1}{\sqrt{2}}$, as it dictates the value of the sign function $\Theta\left(\sigma_{n-1}-\frac{1}{\sqrt{2}}\right)$ and thus the bit estimate value.

To start with, note that the value of the sum in equation (76) is bounded by a geometric series argument:

$$\sum_{k=2}^{m-n}\frac{\varphi_{n+k}}{2^{k+1}} \leq \sum_{k=2}^{m-n}\frac{1}{2^{k+1}} < \sum_{k=2}^{\infty}\frac{1}{2^{k+1}} = \frac{1}{4}. \tag{77}$$

Therefore, the contribution of the corresponding digits is never high enough to increase the argument of the cosine in equation (76) up to or above the next multiple of $\frac{\pi}{4}$. Hence, it can be assigned to one of four cases, determined only by the bit values of φ_n and φ_{n+1}. This leads to the four cases discussed in Table 1 and marked in Figure 13.

Consider for instance the first case, where $\varphi_n\varphi_{n+1} = 00$. The binary fraction representation of $2^{n-1}\varphi$ then corresponds to $0\cdot\frac{1}{2}+0\cdot\frac{1}{4}+\sum_{k=2}^{m-n}\varphi_{n+k}/2^{k+1}$ and its value thus lies in $\left[0,\frac{1}{4}\right)$ due to the aforementioned geometric series argument. Hence, $\varphi_n\varphi_{n+1} = 00$ is associated to the red bar in Figure 13. It is easy to see from the image that the value of σ_{n-1} is greater than $\frac{1}{\sqrt{2}}$, so that $\sigma_{n-1}-\frac{1}{\sqrt{2}} > 0$, and the sign function takes value 1. Due to equation (54), the bit estimate that is obtained by the algorithm then yields 0, so that $\theta = 0.0$. Note that this generated estimate agrees with the desired estimate by the rounding rule: Since the base case delivers a one-bit estimate of $2^{n-1}\varphi = 0.\varphi_n\varphi_{n+1}...\varphi_m$ and the neighboring digit of the first digit, φ_{n+1}, is 0, there is no rounding up to be expected at φ_n, so that $r_{n-1}.\tilde{\varphi}_n = 0.0$ and φ_n and $\tilde{\varphi}_n$ are indeed equal.

An analogous argument applies to the case where $\varphi_n\varphi_{n+1} = 10$. Here, the orange bar in Figure 13 indicates the possible range for σ_{n-1}, such that the sign function returns -1 and again, the generated estimate matches the desired one. No rounding up occurs and the rounding rule is satisfied for both $\varphi_n\varphi_{n+1} = 00$ and $\varphi_n\varphi_{n+1} = 10$.

What is more, the error bound proposed by theorem 7 also conforms to these two cases:

$$|r_{n-1}.\tilde{\varphi}_n - 0.\varphi_n...\varphi_m| = |0.0 - 0.00\varphi_{n+2}...\varphi_m| = \sum_{k=2}^{m-n} \frac{\varphi_{n+k}}{2^{k+1}} \leq \frac{1}{4} = 2^{j-n-1}. \tag{78}$$

Next, consider $\varphi_n\varphi_{n+1} = 01$. In Figure 13, the green section is assigned to this case. For all values of σ_{n-1} in the range $\left(0, \frac{1}{\sqrt{2}}\right)$, the estimate φ_n is 1, so that $\theta = 0.1$. This matches the statement of theorem 7, as rounding to one digit yields $r.\tilde{\varphi}_n = 0.1$ and hence $\varphi_n = \tilde{\varphi}_n$. One can confirm that the error bound is fulfilled as well:

$$|r_{n-1}.\tilde{\varphi}_n - 0.\varphi_n...\varphi_m| = |0.1 - 0.01\varphi_{n+2}...\varphi_m| = \left|\frac{1}{2} - \frac{1}{4} - \sum_{k=2}^{m-n} \frac{\varphi_{n+k}}{2^{k+1}}\right| \leq \frac{1}{4} = 2^{j-n-1}. \tag{79}$$

Lastly, regard $\varphi_n\varphi_{n+1} = 11$, corresponding to the blue bar in Figure 13. For all $\sigma_{n-1} \in \left(\frac{1}{\sqrt{2}}, 1\right)$, the sign function returns 1 and the generated estimate yields $\varphi_n = 0$, so that $\theta = 0.0$. On the other hand, rounding to one decimal yields $r_{n-1}.\tilde{\varphi}_n = 1.0$, so again $\varphi_n = \tilde{\varphi}_n$ is fulfilled. Note that $r_{n-1} = 1$ due to the rounding carryover ensures that the error bound is still maintained:

$$|r_{n-1}.\tilde{\varphi}_n - 0.\varphi_n...\varphi_m| = |1.0 - 0.11\varphi_{n+2}...\varphi_m| = \left|1 - \frac{1}{2} - \frac{1}{4} - \sum_{k=2}^{m-n} \frac{\varphi_{n+k}}{2^{k+1}}\right| \leq \frac{1}{4} = 2^{j-n-1}. \tag{80}$$

One special case has yet to be discussed for the argumentation to be complete, namely if $\sigma_{n-1} = \frac{1}{\sqrt{2}}$. This scenario results in an ambiguous outcome statistic of θ which allows both $\varphi_n = 0$ and $\varphi_n = 1$, reflecting the general intrinsic ambiguity of rounding that occurs when rounding up or down are equally accurate. According to Table 1, it arises for finite m if either $2^{n-1}\varphi = 0.\varphi_n\varphi_{n+1}...\varphi_m = 0.0100...00$ or $2^{n-1}\varphi = 0.1100...0$, and for $m \to \infty$ when $2^{n-1}\varphi = 0.00111...$ or $2^{n-1}\varphi = 0.10111...$. Luckily, the problem is detectable by observing the measurement statistics and we can react to it by adding an additional ancilla qubit and iteration to the algorithm, if the resources allow it. The algorithm then returns an estimate that is one digit more precise as originally intended.

Induction step ($j < n - 1$). Proceeding to the induction step, assume that the statement holds for all iterations prior and including iteration $j+1$, i.e. we can presuppose that the estimate yields $\theta = 0.\tilde{\varphi}_{j+2}\tilde{\varphi}_{j+3}...\tilde{\varphi}_n$ after iteration $j+1$ and that $|\tilde{r}_{j+1}.\tilde{\varphi}_{j+2}\tilde{\varphi}_{j+3}...\tilde{\varphi}_n - 0.\varphi_{j+2}\varphi_{j+3}...\varphi_n...\varphi_m| \leq 2^{(j+1)-n-1}$. We now want to confirm that the iteration j still fulfills the theorem statement and reflects the desired rounding rule.

Similar to the base case, we investigate the contributions of the individual digits to the singular

value in the following iteration, given by

$$\begin{aligned}\sigma_j(\theta) &= |\cos(\pi(2^j\varphi-\theta))| = |\cos(\pi(0.\varphi_{j+1}\varphi_{j+2}...\varphi_n...\varphi_m - 0.0\tilde{\varphi}_{j+2}\tilde{\varphi}_{j+3}...\tilde{\varphi}_n))| \\ &= \left|\cos\left(\pi\left(\frac{\varphi_{j+1}}{2}+\frac{\varphi_{j+2}-\tilde{\varphi}_{j+2}}{4}+\sum_{k=3}^{n-j}\frac{\varphi_{k+j}-\tilde{\varphi}_{k+j}}{2^k}+\sum_{i=n-j+1}^{m-j}\frac{\varphi_{j+i}}{2^i}\right)\right)\right|. \end{aligned} \tag{81}$$

But in higher iterations, $\theta \neq 0$ and therefore the digits $j+2$ until n include the difference between the digits of φ and the current estimate. This way, whether rounding has occurred in previous iterations or not has an impact on the digits to be estimated within the following iterations. The following analysis reveals how this is crucial for the correct implementation of rounding carryovers.

Akin to the base case, the majority of terms in the argument of the cosine is bounded by a geometric series argument, and the contributions of interest are made by φ_{j+1}, φ_{j+2} and $\tilde{\varphi}_{j+2}$. Firstly, consider $\varphi_{j+2} = \tilde{\varphi}_{j+2}$, so that φ_{j+1} solely dictates σ_j. According to equation (81), the argument of the cosine thus lies in the red or orange section of Figure 13, and the generated estimate yields $\theta_1 = \varphi_{j+1}$. Additionally, $\varphi_{j+2} = \tilde{\varphi}_{j+2}$ reveals that no rounding is carried over from the previous iterations, so that $\varphi_{j+1} = \tilde{\varphi}_{j+1}$ and combined with the induction hypothesis (stating that $\theta = 0.\theta_1\tilde{\varphi}_{j+2}\tilde{\varphi}_{j+3}...\tilde{\varphi}_n$) $\theta = 0.\tilde{\varphi}_{j+1}\tilde{\varphi}_{j+2}...\tilde{\varphi}_n$ is verified.

The error bound can be confirmed easily, too: Note that because $\varphi_{j+2} = \tilde{\varphi}_{j+2}$, $\varphi_{j+1} = \tilde{\varphi}_{j+1}$ and $\tilde{r}_{j+1} = 0$ due to the absence of rounding carryover in the previous iteration,

$$\tilde{r}_j.\tilde{\varphi}_{j+1}...\tilde{\varphi}_n - 0.\varphi_{j+1}....\varphi_n...\varphi_m = \frac{1}{2}(\tilde{r}_{j+1}.\tilde{\varphi}_{j+2}...\tilde{\varphi}_n - 0.\varphi_{j+2}....\varphi_n...\varphi_m). \tag{82}$$

Therefore, the error bound for the inductive step follows directly from the bound from the induction hypothesis:

$$\begin{aligned}|\tilde{r}_j.\tilde{\varphi}_{j+1}\tilde{\varphi}_{j+2}...\tilde{\varphi}_n - 0.\varphi_{j+1}\varphi_{j+2}...\varphi_n...\varphi_m| &= \frac{1}{2}|\tilde{r}_{j+1}.\tilde{\varphi}_{j+2}...\tilde{\varphi}_n - 0.\varphi_{j+2}....\varphi_n...\varphi_m| \\ &\leq \frac{1}{2}2^{(j+1)-n-1} = 2^{j-n-1}.\end{aligned} \tag{83}$$

This discussion has shown that the cases in which no rounding in the previous digit has happened, there is no rounding carried over, as expected. Let us now consider the scenarios in which the rounding differs from the original number, i.e., when $\tilde{\varphi}_{j+2} \neq \varphi_{j+2}$.

On one hand, $\tilde{\varphi}_{j+2} = 1$ and $\varphi_{j+2} = 0$ occurs if, in the previous iteration, rounding up was carried through from digit $n+1$ to digit $j+2$. Quantitatively, this means that $\varphi_{j+3} = ... = \varphi_{n+1} = 1$, such that the rounded value of $2^{j+1}\varphi$ to $n-(j+1)$ digits was $\tilde{r}_{j+1}.\tilde{\varphi}_{j+2}...\tilde{\varphi}_n = 0.100...0$. Additionally, there is no possibility of rounding carryover to digit $j+1$, such that rounding $2^j\varphi$ to $n-j$ digits yields $r_j.\varphi_{j+1}\tilde{\varphi}_{j+2}...\tilde{\varphi}_n = 0.\varphi_{j+1}100..0$, and specifically, $\tilde{\varphi}_{j+3} = ...\tilde{\varphi}_n = 0$. Following equation (81), this corresponds to the red or green section of Figure 13, depending on the value of φ_{j+1}, and the generated estimate yields $\theta_1 = \varphi_{j+1}$, which confirms $\theta = 0.\tilde{\varphi}_{j+1}...\tilde{\varphi}_n$. Proving that the error bound is fulfilled works exactly as in the previous case (see equations (82) and (83)).

On the other hand, $\tilde{\varphi}_{j+2} = 0$ and $\varphi_{j+2} = 1$ implies that rounding was even carried over past digit $j+2$ in the prior iteration, i.e. $2^{j+1}\varphi = 0.1\varphi_{j+3}..\varphi_m$ was rounded to $\tilde{r}_{j+1}.\tilde{\varphi}_{j+2}...\tilde{\varphi}_n = 1.0\tilde{\varphi}_{j+3}...\tilde{\varphi}_n$, so that $\tilde{\varphi}_{j+2} = ... = \tilde{\varphi}_n = 0$. This demands for $\varphi_{j+2} = ... = \varphi_{n+1} = 1$. The yet indeterminate

value of φ_{j+1} decides on whether rounding stops at digit $j+1$ or is carried over one digit further.

When $\varphi_{j+1} = 0$, the rounding stops at $j+1$, so that rounding $2^j\varphi$ to $n-j$ digits gives $\tilde{r}_j.\tilde{\varphi}_{j+1}...\tilde{\varphi}_n = 0.100...0$. This corresponds to the green section in Figure 13 such that the algorithm returns $\theta_1 = 1 = \varphi_{j+1}$, which fulfills $\theta = 0.\tilde{\varphi}_{j+1}\tilde{\varphi}_{j+2}...\tilde{\varphi}_n$ as demanded. It remains to check the error bound using the known information about the individual digits and their estimates:

$$\begin{aligned}
|\tilde{r}_j.\tilde{\varphi}_{j+1}\tilde{\varphi}_{j+2}...\tilde{\varphi}_n - 0.\varphi_{j+1}\varphi_{j+2}...\varphi_n...\varphi_m| &= |0.100...0 - 0.011...1\varphi_{n+2}...\varphi_m| \\
&= \left|\frac{1}{2} - \sum_{k=2}^{n-j+1}\frac{1}{2^k} - \sum_{i=n-j+2}^{m-j}\frac{\varphi_{j+i}}{2^i}\right| \\
&= \left|\frac{1}{2^{n-j+1}} - \sum_{i=n-j+2}^{m-j}\frac{\varphi_{j+i}}{2^i}\right| \\
&\leq 2^{j-n-1},
\end{aligned} \tag{84}$$

where we have used the convergence of the finite geometric series for rewriting the series and finding the upper bound.

Next, take $\tilde{\varphi}_{j+2} = 0$, $\varphi_{j+2} = 1$ into account, but with $\varphi_{j+1} = 1$. In this scenario, $\tilde{r}_j.\tilde{\varphi}_{j+1}...\tilde{\varphi}_n = 1.00..0$ is the rounded value of $2^j\varphi = 0.11...\varphi_{n+2}...\varphi_m$ to $n-j$ digits, i.e., the rounding is carried through completely. The significant digits of $2^j\varphi - \theta$ are as such that they correspond to the blue section of Figure 13, and the algorithm returns $\theta_1 = 0$, wherefore θ is as expected. That the error bound is fulfilled is shown almost identically to the prior case in equation (84):

$$\begin{aligned}
|\tilde{r}_j.\tilde{\varphi}_{j+1}\tilde{\varphi}_{j+2}...\tilde{\varphi}_n - 0.\varphi_{j+1}\varphi_{j+2}...\varphi_n...\varphi_m| &= |1.00...0 - 0.11...1\varphi_{n+2}...\varphi_m| \\
&= \left|1 - \frac{1}{2} - \sum_{k=2}^{n-j+1}\frac{1}{2^k} - \sum_{i=n-j+2}^{m-j}\frac{\varphi_{j+i}}{2^i}\right| \\
&= \left|\frac{1}{2^{n-j+1}} - \sum_{i=n-j+2}^{m-j}\frac{\varphi_{j+i}}{2^i}\right| \\
&\leq 2^{j-n-1}.
\end{aligned} \tag{85}$$

In contrast to the base case, theorem 8 guarantees that $\sigma_j \neq \frac{1}{\sqrt{2}}$ for $j < n-1$, which is why there is no need for a separate analysis of potentially problematic cases here. Hence, the proof is complete. □

References

[1] John M. Martyn et al. "A Grand Unification of Quantum Algorithms". In: *PRX Quantum* 2.4 (Dec. 3, 2021), p. 040203. ISSN: 2691-3399. DOI: 10.1103/PRXQuantum.2.040203. arXiv: 2105.02859[quant-ph]. URL: http://arxiv.org/abs/2105.02859.

[2] Ronald de Wolf. *Quantum Computing: Lecture Notes*. Jan. 16, 2023. DOI: 10.48550/arXiv.1907.09415. arXiv: 1907.09415[quant-ph]. URL: http://arxiv.org/abs/1907.09415.

[3] Michael A. Nielsen and Isaac L. Chuang. *Quantum computation and quantum information*. 10th anniversary ed. Cambridge ; New York: Cambridge University Press, 2010. 676 pp. ISBN: 978-1-107-00217-3.

[4] Alexander M. Dalzell et al. *Quantum algorithms: A survey of applications and end-to-end complexities*. Oct. 4, 2023. DOI: 10.48550/arXiv.2310.03011. arXiv: 2310.03011[quant-ph]. URL: http://arxiv.org/abs/2310.03011.

[5] Lieven M. K. Vandersypen et al. "Experimental Realization of an Order-Finding Algorithm with an NMR Quantum Computer". In: *Physical Review Letters* 85.25 (Dec. 18, 2000), pp. 5452–5455. ISSN: 0031-9007, 1079-7114. DOI: 10.1103/PhysRevLett.85.5452. arXiv: quant-ph/0007017. URL: http://arxiv.org/abs/quant-ph/0007017.

[6] Youngseok Kim et al. "Evidence for the utility of quantum computing before fault tolerance". In: *Nature* 618.7965 (June 2023). Number: 7965 Publisher: Nature Publishing Group, pp. 500–505. ISSN: 1476-4687. DOI: 10.1038/s41586-023-06096-3. URL: https://www.nature.com/articles/s41586-023-06096-3.

[7] Tobias J. Osborne. *Lecture series: Quantencomputing und Quantenlogik mit gespeicherten Ionen. Quantum algorithms*. Leibniz Universität Hannover, Dec. 16, 2022.

[8] András Gilyén et al. "Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics". In: *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*. June 23, 2019, pp. 193–204. DOI: 10.1145/3313276.3316366. arXiv: 1806.01838[quant-ph]. URL: http://arxiv.org/abs/1806.01838.

[9] Tobias J. Osborne. *Lecture series: Quantencomputing und Quantenlogik mit gespeicherten Ionen. Quantum gate operations and the circuit model*. Leibniz Universität Hannover, Oct. 28, 2022.

[10] Lin Lin. *Lecture Notes on Quantum Algorithms for Scientific Computation*. Jan. 20, 2022. DOI: 10.48550/arXiv.2201.08309. arXiv: 2201.08309[physics,physics:quant-ph]. URL: http://arxiv.org/abs/2201.08309.

[11] Guang Hao Low, Theodore J. Yoder, and Isaac L. Chuang. "The methodology of resonant equiangular composite quantum gates". In: *Physical Review X* 6.4 (Dec. 28, 2016), p. 041067. ISSN: 2160-3308. DOI: 10.1103/PhysRevX.6.041067. arXiv: 1603.03996[quant-ph]. URL: http://arxiv.org/abs/1603.03996.

[12] Xin Wang et al. "Composite pulses for robust universal control of singlet–triplet qubits". In: *Nature Communications* 3.1 (Aug. 14, 2012). Number: 1 Publisher: Nature Publishing Group, p. 997. ISSN: 2041-1723. DOI: 10.1038/ncomms2003. URL: https://www.nature.com/articles/ncomms2003.

[13] Y Tomita, J T Merrill, and K R Brown. "Multi-qubit compensation sequences". In: *New Journal of Physics* 12.1 (Jan. 19, 2010), p. 015002. ISSN: 1367-2630. DOI: 10.1088/1367-2630/12/1/015002. URL: https://iopscience.iop.org/article/10.1088/1367-2630/12/1/015002.

[14] Malcolm H. Levitt. "Composite pulses". In: *Progress in Nuclear Magnetic Resonance Spectroscopy* 18.2 (Jan. 1, 1986), pp. 61–122. ISSN: 0079-6565. DOI: 10.1016/0079-6565(86)80005-X. URL: https://www.sciencedirect.com/science/article/pii/007965658680005X.

[15] Guang Hao Low and Isaac L. Chuang. "Hamiltonian Simulation by Qubitization". In: *Quantum* 3 (July 12, 2019), p. 163. ISSN: 2521-327X. DOI: 10.22331/q-2019-07-12-163. arXiv: 1610.06546[quant-ph]. URL: http://arxiv.org/abs/1610.06546.

[16] Patrick Rall. "Faster Coherent Quantum Algorithms for Phase, Energy, and Amplitude Estimation". In: *Quantum* 5 (Oct. 19, 2021), p. 566. ISSN: 2521-327X. DOI: 10.22331/q-2021-10-19-566. arXiv: 2103.09717[quant-ph]. URL: http://arxiv.org/abs/2103.09717.

[17] Guang Hao Low and Isaac L. Chuang. *Optimal Hamiltonian Simulation by Quantum Signal Processing*. arXiv.org. June 8, 2016. DOI: 10.1103/PhysRevLett.118.010501. URL: https://arxiv.org/abs/1606.02685v2.

[18] Patrick Rall and Bryce Fuller. *Amplitude Estimation from Quantum Signal Processing*. arXiv.org. July 18, 2022. DOI: 10.22331/q-2023-03-02-937. URL: https://arxiv.org/abs/2207.08628v3.

[19] Ryu Hayakawa. *Quantum algorithm for persistent Betti numbers and topological data analysis*. arXiv.org. Oct. 31, 2021. DOI: 10.22331/q-2022-12-07-873. URL: https://arxiv.org/abs/2111.00433v2.

[20] Sam McArdle, András Gilyén, and Mario Berta. *A streamlined quantum algorithm for topological data analysis with exponentially fewer qubits*. arXiv.org. Sept. 26, 2022. URL: https://arxiv.org/abs/2209.12887v1.

[21] Dominic W. Berry et al. *Analyzing Prospects for Quantum Advantage in Topological Data Analysis*. arXiv.org. Sept. 27, 2022. URL: https://arxiv.org/abs/2209.13581v3.

[22] Daniel List. "Implementation of Quantum Singular Value Transformation". MasterThesis. Leibniz Universität Hannover, Aug. 11, 2023.

[23] Jeongwan Haah. "Product Decomposition of Periodic Functions in Quantum Signal Processing". In: *Quantum* 3 (Oct. 7, 2019), p. 190. ISSN: 2521-327X. DOI: 10.22331/q-2019-10-07-190. arXiv: 1806.10236[quant-ph]. URL: http://arxiv.org/abs/1806.10236.

[24] Rui Chao et al. *Finding Angles for Quantum Signal Processing with Machine Precision*. Mar. 8, 2020. arXiv: 2003.02831[quant-ph]. URL: http://arxiv.org/abs/2003.02831.

[25] Yulong Dong et al. "Efficient phase-factor evaluation in quantum signal processing". In: *Physical Review A* 103.4 (Apr. 22, 2021), p. 042419. ISSN: 2469-9926, 2469-9934. DOI: 10.1103/PhysRevA.103.042419. arXiv: 2002.11649[physics,physics:quant-ph]. URL: http://arxiv.org/abs/2002.11649.

[26] Lexing Ying. "Stable factorization for phase factors of quantum signal processing". In: *Quantum* 6 (Oct. 20, 2022), p. 842. ISSN: 2521-327X. DOI: 10.22331/q-2022-10-20-842. arXiv: 2202.02671[quant-ph]. URL: http://arxiv.org/abs/2202.02671.

[27] Jiasu Wang, Yulong Dong, and Lin Lin. "On the energy landscape of symmetric quantum signal processing". In: *Quantum* 6 (Nov. 3, 2022). Publisher: Verein zur Förderung des Open Access Publizierens in den Quantenwissenschaften, p. 850. DOI: 10.22331/q-2022-11-03-850. URL: https://quantum-journal.org/papers/q-2022-11-03-850/.

[28] C Avalos-Ramos, J A Félix-Algandar, and J A Nieto. "Dyadic Rationals and Surreal Number Theory". In: *IOSR Journal of Mathematics* (Sept. 2020).

[29] A. Yu Kitaev. *Quantum measurements and the Abelian Stabilizer Problem*. Nov. 20, 1995. arXiv: quant-ph/9511026. URL: http://arxiv.org/abs/quant-ph/9511026.

[30] Guang Hao Low. "Quantum signal processing by single-qubit dynamics". Accepted: 2018-04-27T18:10:33Z. Thesis. Massachusetts Institute of Technology, 2017. URL: https://dspace.mit.edu/handle/1721.1/115025.

[31] John M. Martyn. *Private Correspondance*. E-mail. July 26, 2023.

[32] Sam McArdle, András Gilyén, and Mario Berta. *Quantum state preparation without coherent arithmetic*. Oct. 26, 2022. DOI: 10.48550/arXiv.2210.14892. arXiv: 2210.14892[quant-ph]. URL: http://arxiv.org/abs/2210.14892.

[33] P.W. Shor. "Algorithms for quantum computation: discrete logarithms and factoring". In: *Proceedings 35th Annual Symposium on Foundations of Computer Science*. Proceedings 35th Annual Symposium on Foundations of Computer Science. Nov. 1994, pp. 124–134. DOI: 10.1109/SFCS.1994.365700. URL: https://ieeexplore.ieee.org/document/365700.

[34] Koustubh Phalak, Avimita Chatterjee, and Swaroop Ghosh. "Quantum Random Access Memory for Dummies". In: *Sensors* 23.17 (Jan. 2023). Number: 17 Publisher: Multidisciplinary Digital Publishing Institute, p. 7462. ISSN: 1424-8220. DOI: 10.3390/s23177462. URL: https://www.mdpi.com/1424-8220/23/17/7462.

[35] Larry C. Andrews. *Special Functions of Mathematics for Engineers*. SPIE Press, 1998. 512 pp. ISBN: 978-0-8194-2616-1.