Title

Quantum Signal Processing Algorithm and Its Applications

Author

Dong, Yulong

Publication Date

2023

Peer reviewed|Thesis/dissertation

Quantum Signal Processing Algorithm and Its Applications

by

Yulong Dong

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Applied Mathematics

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Lin Lin, Co-chair
Professor K. Birgitta Whaley, Co-chair
Professor Jon Wilkening

Fall 2023

Quantum Signal Processing Algorithm and Its Applications

Abstract

Quantum Signal Processing Algorithm and Its Applications

by

Yulong Dong

Doctor of Philosophy in Applied Mathematics

University of California, Berkeley

Professor Lin Lin, Co-chair

Professor K. Birgitta Whaley, Co-chair

Recent years have witnessed significant advancements in quantum algorithms for scientific computing. Central to these quantum algorithms is Quantum Signal Processing (QSP), which provides a unified framework for designing and understanding quantum algorithms through the exact implementation of matrix polynomials on quantum computers. Asymptotic analyses of QSP-based quantum algorithms indicate their potential to achieve optimal results for various tasks, for example, Hamiltonian simulation. An additional advantage of QSP is its minimal requirement for ancilla qubits, making it more feasible for implementation on near-to-intermediate term quantum architectures. QSP was first introduced theoretically in a seminal paper of Low and Chuang in 2017, and the task of designing scalable algorithms for its explicit implementation was initially considered challenging. However, in recent years, significant strides have been made in solving and comprehending this issue, effectively paving the way for the practical application of QSP.

QSP can be conceptualized as a nonlinear polynomial approximation framework. It maps a set of parameters, known as phase factors, to a polynomial function that adheres to minimal requirements. The essence of solving QSP problems lies in the inversion process: given a target function, the goal is to identify an appropriate set of phase factors such that the QSP-parametrized function closely approximates the target. This dissertation focuses on both the numerical resolution and theoretical exploration of QSP problems, along with their wide-ranging applications, from near-term quantum technologies to fault-tolerant quantum computing regimes. Included

within this scope are various fundamental applications in quantum numerical linear algebra and quantum benchmarking. Furthermore, while QSP offers a simple and elegant theoretical framework for designing and understanding quantum algorithms, it also exhibits rich structures. These structures can be leveraged to accelerate not only the practical deployment of quantum technologies but also to advance theoretical research in this field. This aspect also falls within the scope of this dissertation.

This dissertation is organized as follows: Chapter 1 introduces the developments associated with QSP problems, providing motivation and a brief overview of the results presented in this dissertation. Chapter 2 offers a review of the theory of QSP and its variants. Following this, Chapter 3 presents several topics that serve as streamlined subroutines for enhancing QSP applications. In Chapter 4, a comprehensive overview of efficient iterative algorithms recently developed for solving QSP phase factors is provided, along with their convergence analysis and numerical justifications. Chapter 5 delves into important structural features of QSP problems, highlighting their role in accelerating practical implementations of QSP. The final two chapters showcase applications of QSP through two examples: Chapter 6 introduces a state-of-the-art quantum algorithm for ground-state preparation and energy estimation, highly suitable for early fault-tolerant quantum devices. In Chapter 7, an innovative benchmarking method is presented, which utilizes a random input model and a simplified QSP-based circuit to gauge the performance of quantum computers in scientific computing tasks. Additionally, this chapter provides a thorough analysis of the statistical properties of the ensemble of random input models on quantum computers.

Contents

Acknowledgments

Firstly, I extend my heartfelt gratitude to my research advisors, Professor Lin Lin and Professor K. Birgitta Whaley. Their extensive knowledge and unwavering support in research and beyond have been invaluable. Without their guidance, the work detailed in this dissertation would not have been possible, nor would I have been able to pursue a research and career path that aligns so closely with my interests and enduring spirit.

I am thankful to all the instructors from the courses I attended at Berkeley over the past five years. Their dedication to teaching and cross-disciplinary insights have enriched my intuition and knowledge base, instilling in me the confidence to venture into the unknowns of my research. Special thanks to Professor Jon Wilkening, whose rigorous course series on numerical differential equations profoundly deepened my understanding of numerical analysis. His service as chair of my qualifying exam committee and on my dissertation committee is greatly appreciated.

I am fortunate to have met many outstanding researchers. Their discussions and collaborations have been a source of inspiration and intellectual enjoyment. I would like to express my gratitude once more to my advisors, Professor Lin Lin and Professor K. Birgitta Whaley, and to my collaborators Dr. Zhiyan Ding, Dr. Jonathan A. Gross, Dr. Robert Kosut, Dr. Murphy Yuezhen Niu, Dr. Yu Tong, Jiasu Wang, Xiang Meng, and Hongkang Ni. Special thanks to Murphy for hosting my internships at Google Research, which were filled with fruitful discussions. My group mates, particularly Jiahao Yao and Qinyi Zhu, who joined Berkeley alongside me under Professor Lin's guidance, have shared their expertise generously in discussions and seminars, which is greatly appreciated. I am also thankful to my office mates Dun Tang and Ziwen Zhao for making the office a lively and supportive space.

During the pandemic, a challenging period of isolation, I found strength and emotional support from new friends made online. I am sincerely grateful to them for their companionship and support during these trying times.

Lastly, my deepest gratitude goes to my parents, especially my mother, Mrs. Hongxia Xu, for her immense dedication and support throughout my life, shaping me into the person I am today. I also extend my heartfelt dedication to Kehan, whose presence has brought romance and joy to my everyday life.

Chapter 1

Introduction

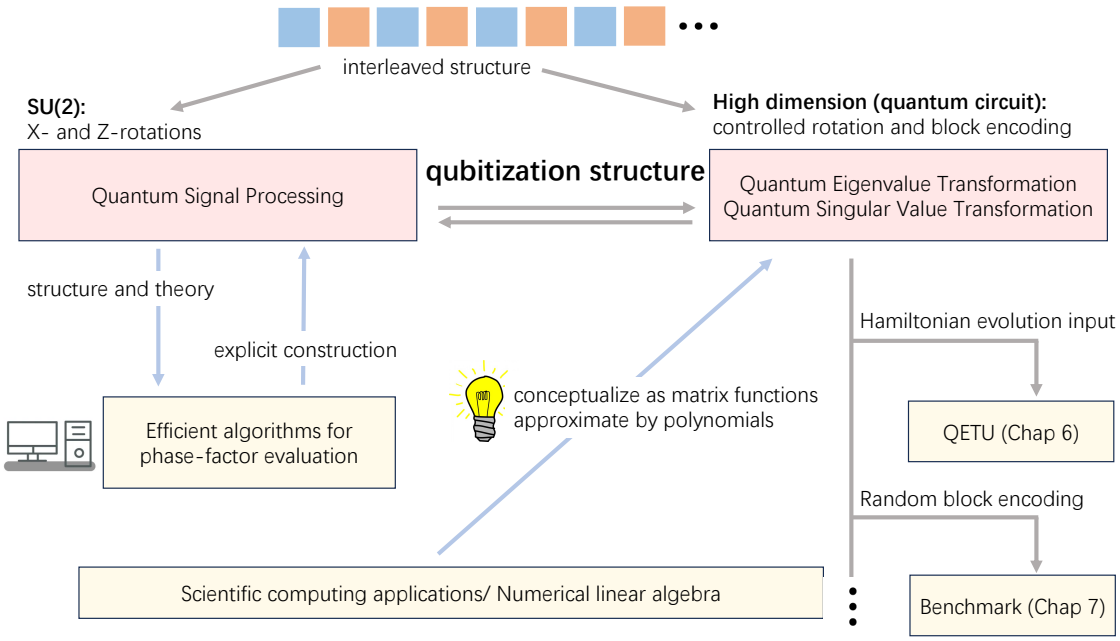


Figure 1.1: An illustrative roadmap of the key ideas and concepts presented in this dissertation. These topics are introduced in the opening chapters and further expanded upon throughout the work.

Recent progress in quantum algorithms has enabled construction of efficient quantum circuit representations for a large class of non-unitary matrices, which significantly

expands the potential range of applications of quantum computers beyond the original goal of efficient simulation of unitary dynamics envisaged by Benioff [12] and Feynman [59]. The basic tool for representation of non-unitary matrices and hence of non-unitary quantum operators is called block encoding [67]. It describes the process in which one embeds a non-unitary matrix A into the upper-left block of a larger unitary matrix U_A, and then expresses the quantum circuit in terms of U_A.

Computation of matrix functions, *i.e.*, evaluation of $F(A)$, where $F(x)$ is a smooth (real-valued or complex-valued) function, is a central task in numerical linear algebra [79]. Numerous computational tasks can be performed by generating approximations to matrix functions. These include application of a broad range of operators to quantum states: *e.g.*, $e^{-\mathrm{i}tA}$ for the Hamiltonian simulation problem; $e^{-\beta A}$ for the thermal state preparation problem; A^{-1} for the matrix inverse (also called the quantum linear system problem, QLSP); and the spectral projector of A for the principal component analysis, to name a few.

Several routes to construct a quantum circuit for $F(A)$ have been developed. These include methods using phase estimation (*e.g.*, the HHL algorithm [76] for the matrix inverse), the method of linear combination of unitaries (LCU) [41, 14], and the method of quantum signal processing (QSP) [101, 99, 67, 105]. Among these methods, QSP stands out as so far the most general approach capable of representing a broad class of matrix functions via the eigenvalue or singular value transformations of A, while using a minimal number of ancilla qubits. The basic idea of QSP is to approximate the desired function $F(x)$ by a polynomial function $f(x)$, and then find a circuit to encode $f(A)$ *exactly* (assuming an exact block encoding U_A). Treating the block encoding U_A as an oracle, the application of QSP has given rise to linear system solving, Hamiltonian system simulation, ground-state energy estimation, and quantum benchmarking [101, 67, 95, 105, 51, 50, 52, 108, 49]. Recently, the construction of QSP has been studied and generalized using advanced theoretical tools [130, 131, 132].

Despite these fast growing successes, practical application of QSP on quantum computers, whether these are near- or long-term machines, faced a significant challenge. A QSP circuit is defined using a series of adjustable phase factors. Once these phase factors are known, the QSP circuit can be directly implemented using U_A together with a set of multi-qubit control gates and single qubit phase rotation gates. However, the inverse problem, *i.e.*, finding the phase factors associated with a given polynomial function $f(x)$ is a considerably more challenging task. The original work of Low and Chuang [101] demonstrated the existence of the phase factors but was not constructive. Initial efforts to find constructive procedures were not encouraging. Thus it was reported in [42] that it was prohibitive to obtain a QSP circuit of length that is larger than 30 for the Jacobi-Anger expansion [101] of the Hamiltonian

simulation problem, and concluded "the difficulty of computing the angles needed to perform the QSP algorithm prevents us from taking full advantage of the algorithm in practice, so it would be useful to develop a more efficient classical procedure for specifying these angles". Consequently, the phase-factor evaluation was originally conceived to be a challenging task. In the past few years, significant progress has been made to develop efficient algorithms to find phase factors. These algorithms fall into two categories: factorization methods [67, 73, 33, 156], and iterative methods [53, 54].

For a given real polynomial $f(x)$, factorization methods construct phase factors from the roots of $1 - f^2(x)$ in the complex plane, and the roots must be obtained at high precision. As a result, as the polynomial d increases, direct implementation of factorization-based methods is not numerically stable and requires $\mathcal{O}(d \log(d/\epsilon))$ bits of precision [67, 73] (ϵ is the target accuracy). There have been two recent improvements of factorization-based methods: the capitalization method [33], and the Prony method [156]. Empirical results indicate that both methods are numerically stable and are applicable to large degree polynomials. Furthermore, the performance of factorization-based methods does not deteriorate near the fully-coherent regime where the maxnorm of the target polynomial is one $\|f\|_\infty = 1$.

Contrasting with the complex construction of factorization-based methods, iterative methods stand out for their intuitiveness, numerical stability, and ease of implementation. The idea is to directly tackle the nonlinear system, or by employing an equivalent optimization formulation. This formulation aims to minimize the discrepancy between the parametric ansatz and the targeted characteristic. However, due to the complex energy landscape [153], direct optimization from random initial guesses can easily get stuck at local minima and can only be used for low degree polynomials. Ref. [53, 153] propose and study the *symmetric* QSP where the set of phase factors are subjected to a symmetry condition that reduces the degrees of freedom. Ref. [53] further observes that starting from a carefully chosen but problem-independent initial guess, standard optimization methods such as the LBFGS method [115] can be robust and stable and can be applied to very high degree polynomials. Recently, we propose the fixed point iteration (FPI) algorithm that directly tackles the nonlinear system and show that the symmetric phase factors have a well-defined limit as the polynomial degree increases towards infinity when the polynomial approaches a smooth (non-polynomial) function. However, it is important to note that in many examples near the fully-coherent regime, the assumptions of those theoretical results are violated. Consequently, gradient-based optimization methods and the FPI method may exhibit slow convergence or fail to converge altogether.

QSP in the fully-coherent regime (or near fully-coherent regime, where $\|f\|_\infty = 1-\delta$ for a small $\delta > 0$) finds applications in quantum algorithms for Hamiltonian sim-

ulation [104] and time-marching based simulation of non-Hermitian dynamics [58]. As discussed in Ref. [55], this problem is particularly challenging in the fully-coherent regime where the nonlinear system is very ill-conditioned (see the numerical section in in Ref. [55] for an illustration of this phenomenon). To overcome these difficulties, we introduce in Ref. [55] a Newton's method tailored for efficiently solving the nonlinear system to solve the nonlinear system. Specifically, we demonstrate that starting with a problem-independent initial guess, as proposed in Ref. [53], Newton's method achieves rapid convergence across all parameter regimes using standard double-precision arithmetic operations.

In addition to developing efficient algorithms for phase-factor evaluation in QSP, another avenue for enhancing QSP's implementability on near-term or long-term quantum machines involves improving the circuit structure by exploring its variants. As of now, the fabrication of full-scale fault-tolerant quantum computers remains a formidable technical challenge for the foreseeable future, and it is reasonable to expect that early fault-tolerant quantum computers share the following characteristics: (1) The number of logical qubits is limited. (2) It can be difficult to execute certain controlled operations (e.g., multi-qubit control gates), whose implementation requires a large number of non-Clifford gates. Note that non-Clifford gates are challenging to implement in quantum computing primarily due to their complex physical realization and their incompatibility with many standard quantum error correction techniques, which complicates maintaining computational accuracy and fidelity [114]. Besides these, the maximum circuit depth of early-fault-tolerant quantum computers, which is determined by the maximum coherence time of the devices, may still be limited. Therefore it is still important to reduce the circuit depth, sometimes even at the expense of a larger total runtime (via a larger number of repetitions). Quantum algorithms tailored for early fault-tolerant quantum computers [31, 10, 24, 89, 93, 149, 158, 151] need to properly take these limitations into account, and the resulting algorithmic structure can be different from those designed for fully fault-tolerant quantum computers.

To gain access to the quantum Hamiltonian H, a standard input model is the *block encoding* (BE) model, which directly encodes the matrix H (after proper rescaling) as a submatrix block of a larger unitary matrix U_H [99, 32]. Combined with techniques such as linear combination of unitaries (LCU) [14], quantum signal processing [101] or quantum singular value transformation [67], one can implement a large class of matrix functions of H on a quantum computer. This leads to quantum algorithms for ground-state preparation and ground-state energy estimation with near-optimal query complexities to U_H [94]. The block encoding technique is also very useful in many other tasks such as Hamiltonian simulation, solving linear systems, preparing the Gibbs state, and computing Green's function and the correlation functions [143,

32, 128, 66, 101]. However, the block encoding of a quantum Hamiltonian (e.g., a sparse matrix) often involves a relatively large number of ancilla qubits, as well as multi-qubit controlled operations that lead to a large number of two-qubit gates and long circuit depths [67], and is therefore not suitable in the early fault-tolerant setting.

A widely used alternative approach for accessing the information in H is the time evolution operator $U = \exp(-i\tau H)$ for some time τ. This input model will be referred to as the *Hamiltonian evolution* (HE) model. While Hamiltonian simulation can be performed using quantum signal processing for sparse Hamiltonians with optimal query complexity [101], such an algorithm queries a block encoding of H, which defeats the purpose of employing the HE model. On the other hand, when H can be efficiently decomposed into a linear combination of Pauli operators, the time evolution operator can be efficiently implemented using, e.g., the Trotter product formula [96, 40] without using any ancilla qubit. This remarkable feature has inspired quantum algorithms for performing a variety of tasks using controlled time evolution and one ancilla qubit. A textbook example of such an algorithm is the Hadamard test. Inspired by these considerations, we developed a tool in Ref. [51] known as the Quantum Eigenvalue Transformation of Unitary Matrices with Real Polynomials (QETU), which uses a controlled Hamiltonian evolution as the input model, a single ancilla qubit and no multi-qubit control operations, and is thus suitable for early fault-tolerant quantum devices. This leads to a simple quantum algorithm that outperforms all previous algorithms with a comparable circuit structure for estimating the ground-state energy. For a class of quantum spin Hamiltonians, we propose a new method that exploits certain anti-commutation relations and further removes the need of implementing the controlled Hamiltonian evolution. Coupled with a Trotter-based approximation of the Hamiltonian evolution, the resulting algorithm can be very suitable for early fault-tolerant quantum devices.

Beyond simplifying circuit complexity through the Hamiltonian evolution input model for deterministic tasks, circuit complexity can also be reduced by adopting random input models. By forgoing the determinism of computational tasks, introducing additional randomness simplifies the complexity, resulting in a less rigid structure for the desired quantum circuits. A practical application of this approach is in assessing quantum computer performance for scientific computing tasks, leveraging random block encoding combined with QSVT circuits. In Ref. [52], we demonstrate an application for QSVT on near term quantum devices that allows benchmarking of Hamiltonian simulation for a class of Hamiltonians that are relevant to recent efforts to demonstrate supremacy of quantum computation over classical computation [8]. This is the class of random Hamiltonians generated from block encoding of random unitary operators that correspond to random unitary circuits. We show that for this

class of Hamiltonians it is possible to formulate a simple metric, called the quantum unitary evolution score (QUES), for the success of quantum unitary evolution. This metric is the primary output from the Hamiltonian simulation benchmark, and is directly related to the circuit fidelity. This allows verification of Hamiltonian simulation on near-term quantum devices without any need for classical computation, and the approach can be scaled to a large number of qubits.

Notations

In this dissertation, we adhere to the following notational conventions: For any matrix $A \in \mathbb{C}^{m\times n}$ the transpose, Hermitian conjugate, and complex conjugate are denoted as $A^\top$, $A^\dagger$, A^* (or $\bar{A}$), respectively. These notations apply to operations on vectors as well. In the context of quantum computing, we represent the basis vectors in $\mathbb{C}^2$ as follows:

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \ |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \ \langle 0| = |0\rangle^\dagger = (1,0)\,, \text{ and } \ \langle 1| = |1\rangle^\dagger = (0,1)\,.$$

For a 2-by-2 matrix $A = (a_{ij})$, the notation $\langle i|A|j\rangle = a_{ij}$ is used for any $i, j \in \{0, 1\}$.

For a matrix $A \in \mathbb{C}^{m\times n}$, the operator norm $\|\cdot\|_2$, induced by the vector ℓ^2 norm, is defined as

$$\|A\|_2 := \sup_{\|x\|_2=1} \|Ax\|_2\,. \tag{1.1}$$

Furthermore, the matrix 1-norm, induced by the vector ℓ^1 norm, is given by

$$\|A\|_1 := \max_{\|x\|_1=1} \|Ax\|_1\,.$$

For any function f defined over $[-1, 1]$, its infinity norm, or maxnorm, is denoted as $\|f\|_\infty := \max_{-1\le x\le 1} |f(x)|$. Given any $x \in [-1, 1]$, the Chebyshev polynomial of the first kind, $T_n(x)$, is defined as $T_n(x) = \cos(n \arccos(x))$.

Chapter 2

Quantum signal processing and its variants

Quantum Signal Processing (QSP) offers a modern, unified framework for understanding and designing quantum algorithms applicable to a broad range of fields. At the core of QSP are two pivotal theories: the reduction of high-dimensional matrices to SU(2) (commonly known as qubitization in standard literature) and an approximation theory within SU(2). Qubitization establishes a duality between high-dimensional applications and a simpler two-dimensional analytical framework. The approximation theory, meanwhile, simplifies the optimization of complex quantum algorithms to the task of finding the best polynomial approximations for specific target functions. Although the existence of these structures was initially reported in Ref. [101], their explicit characterization was presented later in Ref. [67]. This elegant theoretical foundation has since attracted a dedicated subcommunity of researchers, driving both practical and theoretical advancements in QSP. Our work contributes significantly to the theoretical generalizations of QSP-based methods.

This chapter aims to review the theory of QSP and its variants. It is organized as follows: Section 2.1 provides a brief overview of theoretical progress in implementing polynomial matrix transformations on quantum computers. Section 2.2 introduces and proves the cosine-sine decomposition (CSD), a key mathematical structure underlying qubitization. Section 2.3 leverages CSD to conceptualize qubitization and offer a mathematical derivation of QSP. Notably, though Ref. [141] in 2023 presents a derivation of QSP using CSD, our independent study in 2021 paralleled these findings, summarized in an internal note that these two sections draw upon. Section 2.4 introduces a theory similar to QSP but with a symmetric constraint on the parameters, resulting in significant theoretical developments as reported in Ref. [153]. Alongside the original QSP, Section 2.6 discusses an alternative input model leading

to quantum eigenvalue transformation for unitary matrices (QETU), as elaborated in Ref. [51]. This advancement offers an efficient and optimal algorithm for ground-state preparation and energy estimation on early fault-tolerant quantum computers, detailed in Chapter 6. Finally, Section 2.6 presents a multi-level QSP-based method for ground-state preparation that exponentially improves a system dependence.

Please note that Sections 2.1 and 2.5 are based on [51] (joint work with Lin Lin and Yu Tong) and Section 2.4 is based on [153] (joint work with Jiasu Wang and Lin Lin).

2.1 Brief summary of polynomial matrix transformations on quantum computers

In the past few years, there have been significant algorithmic advancements in efficient representation of certain polynomial matrix transformations on quantum computers, which finds applications in Hamiltonian simulation, solving linear systems of equations, eigenvalue problems, to name a few. Several routes to construct a quantum circuit have been developed. These include methods using phase estimation (e.g., the HHL algorithm [76] for the matrix inverse), the method of linear combination of unitaries (LCU) [41, 14], and the method of quantum signal processing (QSP) [101, 99, 67]. Among these methods, QSP stands out as so far the most general approach capable of representing a broad class of matrix functions via the eigenvalue or singular value transformations, while using a minimal number of ancilla qubits. The commonality of these QSP-based approaches is to (1) encode a certain polynomial using a product of parameterized SU(2) matrices, and (2) lift the SU(2) representation to matrices of arbitrary dimensions (a procedure called "qubitization" [99] which is related to quantum walks [140, 37]). This framework often leads to a very concise quantum circuit, and can unify a large class of quantum algorithms that have been developed in the literature [67, 105]. For clarity of the presentation, the term quantum signal processing (QSP) will specifically refer to the SU(2) representation. It is worth noting that the depending on the structure of the matrix and the input model, the resulting quantum circuits can be different. Block encoding [99, 67] is a commonly used input model for representing non-unitary matrices on a quantum computer.

When a polynomial of interest is represented by QSP, we can use the block encoding input model to implement the polynomial transformation of a Hermitian matrix, which gives the quantum eigenvalue transformation (QET) [99]. Similarly the polynomial transformation of a general matrix (called singular value transforma-

tion) gives the quantum singular value transformation (QSVT) [67]. In fact, for a Hermitian matrix with a block encoding input model, the quantum circuits of QET and QSVT can be the same.

It is worth noting that the original presentation of QSP [101] combines together the SU(2) representation and a trigonometric polynomial transformation of a Hermitian matrix H, and the input model is provided by a quantum walk operator [37]. If H is a s-sparse matrix, the use of a walk operator is actually not necessary, and QET/QSVT gives a more concise algorithm than that in Ref. [101].

Using the Hamiltonian evolution input model, QETU algorithm, proposed in Ref. [51], provides a circuit structure that is similar to that in [101, Figure 1], and the derivation of QETU is both simpler and more constructive. Note that Ref. [101] only states the existence of the parameterization without providing an algorithm to evaluate the phase factors, and the connection with the more explicit parameterization such as those in [67, 73] has not been shown in the literature. QETU algorithm directly connects to the parameterization in [67], and in particular, QSP with symmetric phase factors [53, 153]. This gives rise to a concise way for representing real polynomial transformations that is encountered in most applications.

The QETU technique is also related to QSVT. From the Hamiltonian evolution input model $U = e^{-iH}$, we can first use one ancilla qubit and linear combination of unitaries to implement a block encoding of $\cos(H) = (U + U^\dagger)/2$. Using another ancilla qubit, we can use another ancilla qubit and QET/QSVT to implement $H = \arccos(\cos(H))$ approximately. In other words, from the Hamiltonian evolution U we can implement the matrix logarithm of U to approximately block encode H. Then we can implement a matrix function $f(H)$ using the block encoding above and another layer of QSVT. QETU simplifies the procedure above by directly querying U. The concept of "qubitization" [99] appears very straightforwardly in QETU. It also saves one ancilla qubit and gives perhaps a slightly smaller circuit depth.

2.2 Cosine-sine decomposition

In this section, we introduce a useful theorem to present a mathematical derivation of Quantum Signal Processing (QSP). At the core of QSP lies a standard form of high-dimensional unitary matrices expressed in terms of the direct sum of SU(2). The Cosine-Sine Decomposition (CSD) provides a mathematical framework to understand and derive this structure of decomposition. We begin with the CSD of rectangular matrices having orthogonal columns, specifically where $U^\dagger U = I$. Subsequently, we extend this to derive the CSD of unitary matrices by leveraging this result.

Theorem 2.2.1 (CSD of rectangular matrix with orthonormal columns)**.** *For $q \geq p$, let $U \in \mathbb{C}^{(p+q)\times p}$ be a matrix satisfying $U^\dagger U = I_p$, which is referred to as a complex matrix with orthonormal columns. There exists a decomposition*

$$U = \begin{pmatrix} W_1 & 0 \\ 0 & W_2 \end{pmatrix} \begin{pmatrix} C \\ S \\ 0 \end{pmatrix} V^\dagger.$$

Here, $W_1, V \in \mathbb{C}^{p\times p}$, $W_2 \in \mathbb{C}^{q\times q}$ are unitary matrices and $C = \mathrm{diag}\{c_1, \cdots, c_p\}, S = \mathrm{diag}\{s_1, \cdots, s_p\}$ are diagonal matrices so that $c_j^2 + s_j^2 = 1\ \forall j$.

Proof. Let the rows of the matrix U be partitioned as

$$U = \begin{pmatrix} U_1 \\ U_2 \end{pmatrix} \in \mathbb{C}^{(p+q)\times p} \text{ where } U_1 \in \mathbb{C}^{p\times p}, U_2 \in \mathbb{C}^{q\times p}.$$

Let the singular value decomposition (SVD) of U_1 be $U_1 = W_1 C V^\dagger$ where $W_1, V \in \mathbb{C}^{p\times p}$ and C is a p-by-p diagonal matrix. Consider the QR factorization of the matrix $U_2 V$ as $U_2 V = W_2 R$ where $W_2 \in \mathbb{C}^{q\times q}$ is a unitary matrix and $R \in \mathbb{C}^{q\times p}$ is an upper triangular matrix. Then, $U_2 = W_2 R V^\dagger$. As a consequence of the orthonormality of rows, it holds that

$$U^\dagger U = U_1^\dagger U_1 + U_2^\dagger U_2 = V\left(C^2 + R^\dagger R\right) V^\dagger = I_p,$$

which means that $R^\dagger R = I_p - C^2$ is diagonal.

Because R is upper triangular, applying induction can show that all off-diagonal elements of the upper triangular matrix R must be zero. Let $S = \mathrm{diag}(R) = \sqrt{R^\dagger R}$. The proof is completed. □

Theorem 2.2.2 (CSD of unitary matrix)**.** *Let $q \geq p$ and $U \in \mathbb{C}^{(p+q)\times(p+q)}$ be any unitary matrix. There exists a decomposition*

$$U = \begin{pmatrix} W_1 & 0 \\ 0 & W_2 \end{pmatrix} \begin{pmatrix} C & S & 0 \\ -S & C & 0 \\ 0 & 0 & I_{q-p} \end{pmatrix} \begin{pmatrix} V_1^\dagger & 0 \\ 0 & V_2^\dagger \end{pmatrix}. \tag{2.1}$$

Here, $W_1, V_1 \in \mathbb{C}^{p\times p}, W_2, V_2 \in \mathbb{C}^{q\times q}$ are unitary matrices and $C = \mathrm{diag}\{c_1, \cdots, c_p\}, S = \mathrm{diag}\{s_1, \cdots, s_p\}$ are diagonal matrices so that $c_j^2 + s_j^2 = 1\ \forall j$.

Proof. Consider the following partition

$$U = \begin{pmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{pmatrix} \text{ where } U_{11} \in \mathbb{C}^{p\times p}, U_{22} \in \mathbb{C}^{q\times q}.$$

We then apply Theorem 2.2.1 separately to the matrices $\begin{pmatrix} U_{11} \\ U_{21} \end{pmatrix}$ and $(U_{11}\ U_{12})$.

Let the SVDs be given by:

$$U_{11} = W_1 C V_1^\dagger,\ U_{21} = W_2 (S\ 0)^\top V_1^\dagger = (-W_2)(-S\ 0)^\top V_1^\dagger,\ U_{12} = W_1 (S\ 0) V_2^\dagger.$$

Here, the singular values of U_{11} is sorted in an ascending order and let r be the number of singular values strictly less than 1. Then,

$$C = \mathrm{diag}\{\sigma_1, \sigma_2, \cdots, \sigma_r, \underbrace{1, \cdots, 1}_{p-r}\} = \begin{pmatrix} \widetilde{C} & 0 \\ 0 & I_{p-r} \end{pmatrix},\ \text{where } 0 \le \sigma_1 \le \sigma_2 \le \cdots \le \sigma_r < 1, \tag{2.2}$$

and

$$S = \mathrm{diag}\{\sqrt{1-\sigma_1^2}, \sqrt{1-\sigma_2^2}, \cdots, \sqrt{1-\sigma_r^2}, \underbrace{0, \cdots, 0}_{p-r}\} = \begin{pmatrix} \widetilde{S} & 0 \\ 0 & 0 \end{pmatrix}, \tag{2.3}$$

where $\widetilde{S}$ is nonsingular. According to Theorem 2.2.1, it holds that

$$U = \begin{pmatrix} W_1 & 0 \\ 0 & -W_2 \end{pmatrix} \begin{pmatrix} C & S & 0 \\ -S & -W_2^\dagger U_{22} V_2 \\ 0 \end{pmatrix} \begin{pmatrix} V_1^\dagger & 0 \\ 0 & V_2^\dagger \end{pmatrix}.$$

Let us partition $-W_2^\dagger U_{22} V_2$ as

$$-W_2^\dagger U_{22} V_2 = \begin{pmatrix} Q_{11} & Q_{12} & Q_{13} \\ Q_{21} & Q_{22} & Q_{23} \\ Q_{31} & Q_{32} & Q_{33} \end{pmatrix}$$

where $Q_{11} \in \mathbb{C}^{r\times r}, Q_{22} \in \mathbb{C}^{(p-r)\times(p-r)}, Q_{33} \in \mathbb{C}^{(q-p)\times(q-p)}$. The fact that

$$\begin{pmatrix} C & S & 0 \\ -S & -W_2^\dagger U_{22} V_2 \\ 0 \end{pmatrix} = \begin{pmatrix} \widetilde{C} & 0 & \widetilde{S} & 0 & 0 \\ 0 & I_{p-r} & 0 & 0 & 0 \\ -\widetilde{S} & 0 & Q_{11} & Q_{12} & Q_{13} \\ 0 & 0 & Q_{21} & Q_{22} & Q_{23} \\ 0 & 0 & Q_{31} & Q_{32} & Q_{33} \end{pmatrix},$$

is a unitary matrix implies a system of equations

$$\begin{cases} \widetilde{S}\widetilde{C} - \widetilde{S} Q_{11} = 0 \\ \widetilde{S} Q_{12} = 0 \\ \widetilde{S} Q_{13} = 0 \\ \widetilde{S}^2 + Q_{11}^\dagger Q_{11} + Q_{21}^\dagger Q_{21} + Q_{31}^\dagger Q_{31} = I_r \end{cases} \Rightarrow \begin{cases} Q_{11} = \widetilde{C} \\ Q_{12} = 0, Q_{13} = 0, Q_{21} = 0, Q_{31} = 0 \end{cases}.$$

Hence

$$\widetilde{Q} := \begin{pmatrix} Q_{22} & Q_{23} \\ Q_{32} & Q_{33} \end{pmatrix} \in \mathbb{C}^{(q-r)\times(q-r)}$$

is a unitary matrix. Combining these derived results, the following equation holds

$$\begin{aligned} U &= \begin{pmatrix} W_1 & 0 \\ 0 & -W_2 \begin{pmatrix} I_r & 0 \\ 0 & \widetilde{Q} \end{pmatrix} \end{pmatrix} \begin{pmatrix} \widetilde{C} & 0 & \widetilde{S} & 0 & 0 \\ 0 & I_{p-r} & 0 & 0 & 0 \\ -\widetilde{S} & 0 & \widetilde{C} & 0 & 0 \\ 0 & 0 & 0 & I_{p-r} & 0 \\ 0 & 0 & 0 & 0 & I_{q-p} \end{pmatrix} \begin{pmatrix} V_1^\dagger & 0 \\ 0 & V_2^\dagger \end{pmatrix} \\ &= \begin{pmatrix} W_1 & 0 \\ 0 & -W_2 \begin{pmatrix} I_r & 0 \\ 0 & \widetilde{Q} \end{pmatrix} \end{pmatrix} \begin{pmatrix} C & S & 0 \\ -S & C & 0 \\ 0 & 0 & I_{q-p} \end{pmatrix} \begin{pmatrix} V_1^\dagger & 0 \\ 0 & V_2^\dagger \end{pmatrix}. \end{aligned}$$

It proves the theorem. □

2.3 Qubitization and quantum signal processing

Qubitization is a framework wherein a unitary matrix can be transformed into a standard form, comprising a direct sum of SU(2) submatrices, through unitary-matrix transformations. Although quantum computers can perform various tasks efficiently, their quantum mechanical foundation mandates that all realizable quantum gates are unitary matrices. As a result, information in quantum computing is represented either as unitary matrices or their submatrices. Therefore, analyzing the structural relationship between unitary matrices and their submatrices is crucial in the development of quantum algorithms. In this section, we demonstrate the qubitization structure using the Cosine-Sine Decomposition (CSD) established in the previous section. Within this framework, we will introduce the formalism of Quantum Signal Processing (QSP).

Qubitization

In the realm of quantum computing, the submatrix of a unitary matrix is commonly referred to as a "block encoding" which is pivotal in representing and manipulating quantum information.

Definition 2.3.1 (Block encoding)**.** *Given an n-qubit matrix A, if we can find $\alpha, \epsilon \in \mathbb{R}_+$, and an $(m+n)$-qubit unitary matrix U_A so that*

$$\|A - \alpha \left(\langle 0^m| \otimes I_N\right) U_A \left(|0^m\rangle \otimes I_N\right) \|_2 \leq \epsilon, \tag{2.4}$$

then U_A is called an (α, m, ϵ)-block-encoding of A.

Here, an n-qubit matrix is defined as an N-by-N matrix where $N = 2^n$. In situations where we focus exclusively on the case of exact block encoding without additional scaling, characterized by $\alpha = 1$ and $\epsilon = 0$, we refer to the model succinctly as an m-block-encoding. An illustration of block encoding is provided in Fig. 2.1.

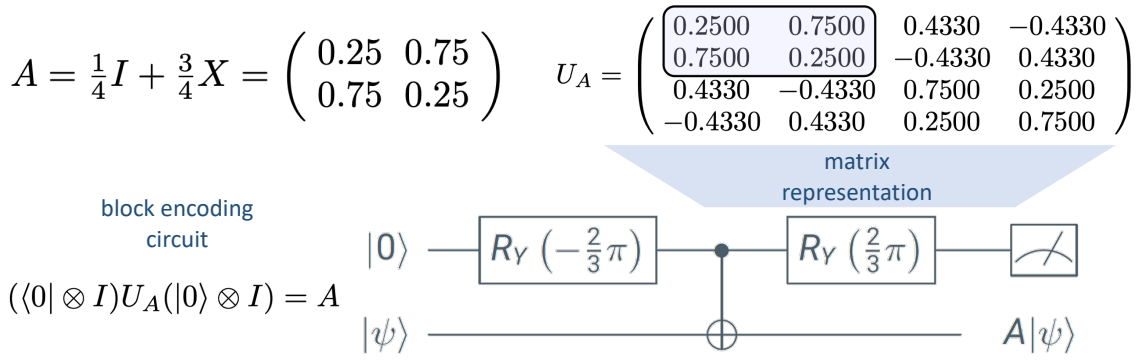


Figure 2.1: An example of block encoding.

Given U_A an m-block-encoding of an n-qubit complex matrix $A = W\Sigma V^\dagger$, Theorem 2.2.2 implies that there exists $W', V' \in \mathbb{C}^{N(M-1)\times N(M-1)}$ so that

$$U_A = \begin{pmatrix} W & 0 \\ 0 & W' \end{pmatrix} \begin{pmatrix} \Sigma & S & 0 \\ -S & \Sigma & 0 \\ 0 & 0 & I_{N(M-2)} \end{pmatrix} \begin{pmatrix} V^\dagger & 0 \\ 0 & V'^\dagger \end{pmatrix}. \tag{2.5}$$

Here, $S = \sqrt{I - \Sigma^2}$ is the supplementary diagonal matrix. For notation brevity, the large unitary matrices are denoted as

$$\widetilde{W} := \begin{pmatrix} W & 0 \\ 0 & W' \end{pmatrix}, \widetilde{V} := \begin{pmatrix} V & 0 \\ 0 & V' \end{pmatrix}.$$

Note that the simplified matrix in the middle exhibits a special structure. By conjugating with a permutation matrix K, this middle matrix can be expressed as a direct sum of 2-by-2 blocks and scalars:

$$\begin{aligned} \begin{pmatrix} \Sigma & S & 0 \\ -S & \Sigma & 0 \\ 0 & 0 & I_{N(M-2)} \end{pmatrix} &= K \bigoplus_{j\in[N]} \begin{pmatrix} \sigma_j & \sqrt{1-\sigma_j^2} \\ -\sqrt{1-\sigma_j^2} & \sigma_j \end{pmatrix} \oplus I_{N(M-2)} K^\dagger \\ &= K \bigoplus_{j\in[N]} e^{-\mathrm{i}\frac{\pi}{4}Z} e^{\mathrm{i}\arccos(\sigma_j)X} e^{\mathrm{i}\frac{\pi}{4}Z} \oplus I_{N(M-2)} K^\dagger. \end{aligned} \tag{2.6}$$

This process of rearrangement is termed *qubitization*. Qubitization offers a systematic approach to decomposing high-dimensional unitary matrices into their submatrices. The dimension of each submatrix is limited to two, and each is uniquely characterized by the singular value of the block-encoded submatrix A. An illustration of qubitization is provided in Fig. 2.2.

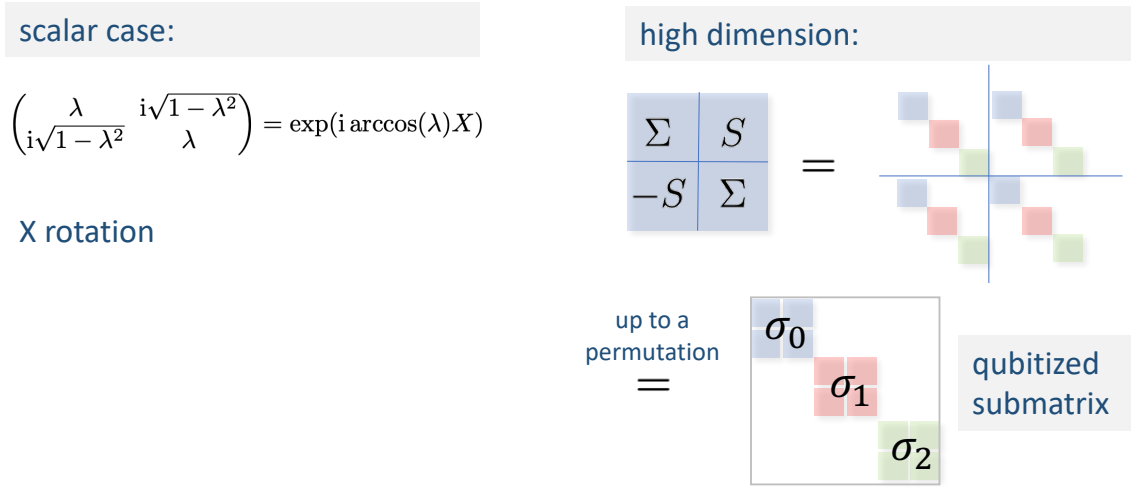


Figure 2.2: An illustration of qubitization.

Quantum Signal Processing

Quantum Signal Processing (QSP) offers a framework for manipulating and transforming information within the block encoding formalism. As observed in the previous subsection, each block encoding U_A allows for a decomposition through the left and right action of unitary matrices $\widetilde{W}K$ and $\widetilde{V}K$. In this decomposition, each submatrix is uniquely determined by a singular value of the matrix A. Therefore, to access and manipulate the singular values of A, we can multiply the block encoding with matrices that commute with these unitary transformations, $\widetilde{W}K$ and $\widetilde{V}K$.

It is noteworthy that a simple diagonal matrix fits within the requirements of this framework. This category of matrices forms a single-angle parametric matrix family, commonly known as "controlled rotation" in the literature. Given an angle parameter φ, the matrix representation under the computational basis is

$$\mathrm{R}_{\mathrm{ctrl}}(\varphi) = \begin{pmatrix} e^{\mathrm{i}\varphi} I_N & 0 \\ 0 & e^{-\mathrm{i}\varphi} I_{N(M-1)} \end{pmatrix}. \tag{2.7}$$

It can be demonstrated that, following the application of the unitary transformation associated with qubitization, the controlled rotation is effectively transformed into a simultaneous single-qubit Z rotation applied to each submatrix block:

$$\widetilde{W}K\mathrm{R}_{\mathrm{ctrl}}(\varphi)K^{\dagger}\widetilde{V}^{\dagger} = \bigoplus_{j\in[N]} e^{\mathrm{i}\varphi Z} \oplus e^{-\mathrm{i}\varphi} I_{N(M-2)}. \tag{2.8}$$

Direct implementation of the controlled rotation presents challenges. Nevertheless, by incorporating an additional ancilla, the controlled rotation matrix can be efficiently implemented using a minimal number of gates, as illustrated in Fig. 2.3 (a). The matrix representation of the indirect implementation shown in Fig. 2.3 (a) is an

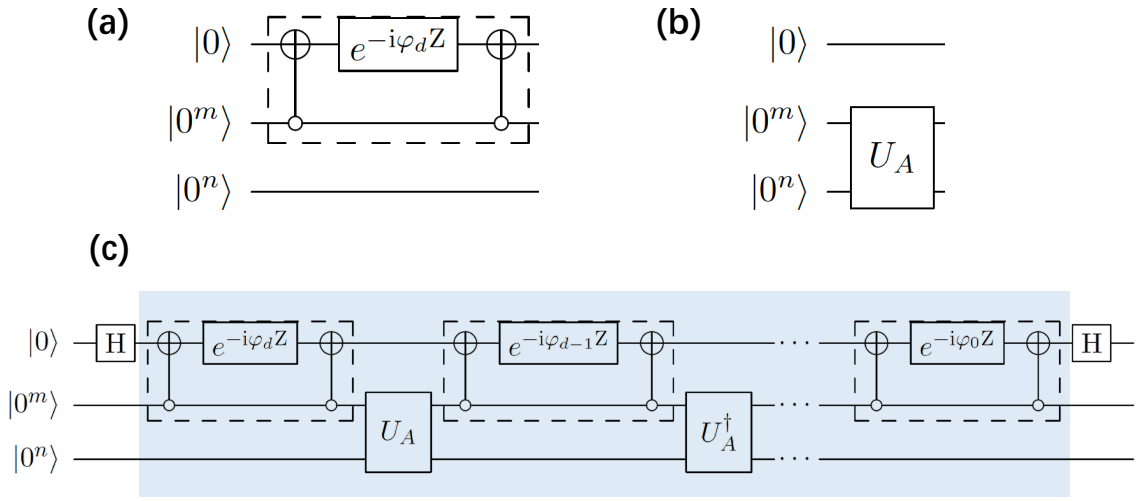


Figure 2.3: Representations of gates and circuits. (a) Implementation of controlled rotation using Toffoli gates and single-qubit Z rotation. (b) Gate representation of block encoding incorporating an idle ancilla qubit. (c) Circuit representation of Quantum Singular Value Transformation (QSVT) circuits, with the shaded area indicating an alternating phase modulation sequence.

augmented matrix of doubled size:

$$\begin{pmatrix} \mathrm{R}_{\mathrm{ctrl}}(\varphi) & 0 \\ 0 & \mathrm{R}_{\mathrm{ctrl}}(-\varphi) \end{pmatrix}.$$

By introducing an additional ancilla qubit, the block encoding is similarly augmented:

$$\begin{pmatrix} U_A & 0 \\ 0 & U_A \end{pmatrix}.$$

Consequently, our focus remains on studying the action of the original controlled rotation $\mathrm{R}_{\mathrm{ctrl}}(\varphi)$. By clearly understanding the structure using original controlled rotations, the formalism using augmented matrices can be derived by supplementing the lower-right block, essentially negating all phase angles of the upper-left block.

Interleaving controlled rotations with block encoding results in a structure known as the Alternating Phase Modulation Sequence (APMS), illustrated in Fig. 2.3 (c). For a given m-block-encoding of an n-qubit complex matrix A, and a set of phase factors $\Psi := (\varphi_0, \cdots, \varphi_d) \in \mathbb{R}^{d+1}$, the APMS is defined as:

$$\mathcal{U}(\Psi, U_A) := \mathrm{R}_{\mathrm{ctrl}}(\varphi_0) \left(\prod_{j=1}^{d} U_A^{(-1)^{d-j}} \mathrm{R}_{\mathrm{ctrl}}(\varphi_j) \right). \tag{2.9}$$

Following our previous analysis, the shaded area in Fig. 2.3 (c) can be represented by a block diagonal matrix:

$$\begin{pmatrix} \mathcal{U}(\Psi, U_A) & 0 \\ 0 & \mathcal{U}(-\Psi, U_A) \end{pmatrix}. \tag{2.10}$$

Due to the interleaving of U_A and $U_A^\dagger$, the subsequent left- and right-unitary conjugations differ, although they share the most crucial component in the middle in terms of the qubitized structure. When d is odd, the APMS is represented as:

$$\mathcal{U}(\Psi, U_A) = \widetilde{W} \left(\mathrm{R}_{\mathrm{ctrl}}(\varphi_0) \left(\prod_{j=1}^{d} \begin{pmatrix} \Sigma & S & 0 \\ -S & \Sigma & 0 \\ 0 & 0 & I_{q-p} \end{pmatrix}^{(-1)^{d-j}} \mathrm{R}_{\mathrm{ctrl}}(\varphi_j) \right) \right) \widetilde{V}^\dagger. \tag{2.11}$$

Conversely, when d is even, the APMS is given by:

$$\mathcal{U}(\Psi, U_A) = \widetilde{V} \left(\mathrm{R}_{\mathrm{ctrl}}(\varphi_0) \left(\prod_{j=1}^{d} \begin{pmatrix} \Sigma & S & 0 \\ -S & \Sigma & 0 \\ 0 & 0 & I_{N(M-2)} \end{pmatrix}^{(-1)^{d-j}} \mathrm{R}_{\mathrm{ctrl}}(\varphi_j) \right) \right) \widetilde{V}^\dagger. \tag{2.12}$$

Despite the difference in the left-most unitary matrix, the core component remains identical:

$$\widetilde{\mathcal{U}}(\Psi, U_A) := \mathrm{R}_{\mathrm{ctrl}}(\varphi_0) \left(\prod_{j=1}^{d} \begin{pmatrix} \Sigma & S & 0 \\ -S & \Sigma & 0 \\ 0 & 0 & I_{N(M-2)} \end{pmatrix}^{(-1)^{d-j}} \mathrm{R}_{\mathrm{ctrl}}(\varphi_j) \right). \tag{2.13}$$

Incorporating the qubitization structure, on each two-dimensional qubitized subspace, the action of $\widetilde{U}(\Psi, U_A)$ is characterized by interleaved single-qubit Z and X rotations. To address the variations introduced by interleaving U_A and $U_A^\dagger$, we define a set of modified phase factors $\Phi := (\phi_0, \cdots, \phi_d)$ for a more streamlined presentation. When $d = 2k$, it is

$$\phi_j = \begin{cases} \varphi_j + \frac{\pi}{4}, & j = 0 \text{ or } 2k, \\ \varphi_j + \frac{\pi}{2}, & j = 2, 4, \cdots, 2k-2, \\ \varphi_j - \frac{\pi}{2}, & j = 1, 3, \cdots, 2k-1. \end{cases} \tag{2.14}$$

Conversely, when $d = 2k+1$, it is

$$\phi_j = \begin{cases} \varphi_j - \frac{\pi}{4}, & j = 0, \\ \varphi_j + \frac{\pi}{4}, & j = 2k+1, \\ \varphi_j - \frac{\pi}{2}, & j = 2, 4, \cdots, 2k, \\ \varphi_j + \frac{\pi}{2}, & j = 1, 3, \cdots, 2k-1. \end{cases} \tag{2.15}$$

Consequently, the formalism is further simplified as follows:

$$\widetilde{\mathcal{U}}(\Psi, U_A) = K \bigoplus_{j\in[N]} \left(e^{\mathrm{i}\phi_0 Z} \prod_{j=1}^{d} e^{\mathrm{i}\arccos(\sigma_j)X} e^{\mathrm{i}\phi_j Z} \right) \oplus e^{-\mathrm{i}\sum_{j=0}^{d}\varphi_j} I_{N(M-2)} K^\dagger. \tag{2.16}$$

Thus, qubitization results in a theoretical framework within SU(2) that parallels the APMS.

Definition 2.3.2 (Quantum signal processing (QSP))**.** *Given a set of phase factors* $\Phi = (\phi_0, \cdots, \phi_d) \in \mathbb{R}^{d+1}$*, QSP defines a map from* $[-1, 1]$ *to* SU(2)

$$\begin{aligned} x \mapsto U(x, \Phi) &:= e^{\mathrm{i}\phi_0 Z} \prod_{j=1}^{d} W(x) e^{\mathrm{i}\phi_j Z} \\ \textit{where } W(x) &:= e^{\mathrm{i}\arccos(x)X} = \begin{pmatrix} x & \mathrm{i}\sqrt{1-x^2} \\ \mathrm{i}\sqrt{1-x^2} & x \end{pmatrix}. \end{aligned} \tag{2.17}$$

Given a set of phase factors, QSP defines a parametric family within SU(2) in terms of the variable x. Intriguingly, these parametric SU(2)-valued matrices exhibit a structural connection to two distinct polynomials. This relationship is elucidated in the following theorem.

Theorem 2.3.3 (Quantum signal processing in SU(2) [66, Theorem 3])**.** *For any* $P, Q \in \mathbb{C}[x]$ *and a positive integer* d *such that*

(1) $\deg(P) \leq d, \deg(Q) \leq d-1$,

(2) P *has parity* $(d \mod 2)$ *and* Q *has parity* $(d-1 \mod 2)$,

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

Then, there exists a set of phase factors $\Phi := (\phi_0, \cdots, \phi_d) \in \mathbb{R}^{d+1}$ *such that*

$$U(x,\Phi) = \begin{pmatrix} P(x) & \mathrm{i}Q(x)\sqrt{1-x^2} \\ \mathrm{i}Q^*(x)\sqrt{1-x^2} & P^*(x) \end{pmatrix}, \tag{2.18}$$

where the complex conjugate of a complex polynomial $P(x) = \sum_j p_j x^j$, $p_j \in \mathbb{C}$ *is defined by taking complex conjugate on its coefficients, i.e.,* $P^*(x) = \sum_j \bar{p}_j x^j$.

This theorem indicates the existence of a pair of complex-valued polynomials $(P,Q) \in \mathbb{C}_d[x] \times \mathbb{C}_{d-1}[x]$, which are determined by the set of phase factors. Consequently, the APMS can be further simplified as follows:

$$K^\dagger \widetilde{\mathcal{U}}(\Psi, U_A) K = \begin{pmatrix} P(\Sigma) & \mathrm{i}\sqrt{I_N - \Sigma^2}Q(\Sigma) & 0 \\ \mathrm{i}\sqrt{I_N - \Sigma^2}Q^*(\Sigma) & P^*(\Sigma) & 0 \\ 0 & 0 & e^{-\mathrm{i}\sum_{j=0}^{d}\varphi_j} I_{N(M-2)} \end{pmatrix}. \tag{2.19}$$

The earlier discussion highlights that the left-most unitary matrix varies depending on the parity of d. Therefore, while the results appear similar in the qubitized subspace, they diverge when extending the analytical results to higher dimensions. To account for this disparity, we introduce what is called the singular value transformation of matrices, as opposed to the standard matrix function, which is applicable only to Hermitian matrices. It is important to note that these two approaches coincide in cases where A is Hermitian.

Definition 2.3.4 (Singular value transformation (SVT) of general matrices)**.** *Let* $A = W\Sigma V^\dagger$ *represent any complex square matrix, with the latter expression being its SVD. For any even function* P*, the SVT of* A *through* P *is defined as* $P_{\mathrm{SV}}(A) = VP(\Sigma)V^\dagger$*. Conversely, when* P *is an odd function, the SVT is defined as* $P_{\mathrm{SV}}(A) = WP(\Sigma)V^\dagger$.

Leveraging the notation of SVT, the matrix representation of the APMS can be written compactly as follows by multiplying the unitary matrices back:

$$\mathcal{U}(\Psi, U_A) = \begin{pmatrix} P_{\mathrm{SV}}(A) & * \\ * & * \end{pmatrix}. \tag{2.20}$$

The upper-left submatrix block of the APMS corresponds to the SVT of the block-encoded matrix A, determined through the polynomial function associated with the selected phase factors. Therefore, the quantum circuit of APMS, particularly the shaded area in Fig. 2.3 (c), is recognized as the Quantum Singular Value Transformation (QSVT) in the literature.

While the focus has been on the upper-left block of the augmented APMS, the lower-right block exhibits a similar form, achieved by inverting all phase factors. Intriguingly, there exists a fundamental connection between the polynomials derived from a set of phase factors and those derived from their negation. This connection is elucidated in the subsequent theorem.

Theorem 2.3.5. *Let Φ be the set of modified phase factors derived from Ψ, obtained through either Eq.* (2.14) *or Eq.* (2.15)*. Additionally, let (P, Q) represent a pair of polynomials induced by Φ as per Definition 2.3.2. Then, the matrix representation of the QSVT circuit shown in Fig. 2.3 (c) is as follows:*

$$\begin{pmatrix} P_{\mathrm{SV}}(A) & * & 0 & 0 \\ * & * & 0 & 0 \\ 0 & 0 & (P^*)_{\mathrm{SV}}(A) & * \\ 0 & 0 & * & * \end{pmatrix}. \tag{2.21}$$

Proof. The upper-left block has already been determined through prior computations. It's important to note that the lower-right block represents the APMS achieved by inverting all phase factors. Thus, it is crucial to explore the resultant effects on the SU(2) model. Let ϕ_j^+ be the modified phase factors of Ψ and ϕ_j^- be those of $-\Psi$. Define the corresponding vectors as $\Phi = (\phi_j^+)$ and $\Phi^- = (\phi_j^-)$. These vectors are linked by the following relationship: when $d = 2k$

$$\phi_j^- = \begin{cases} -\phi_j^+ + \frac{\pi}{2}, & j = 0 \text{ or } 2k, \\ -\phi_j^+ + \pi, & j = 2, 4, \cdots, 2k-2, \\ -\phi_j^+ - \pi, & j = 1, 3, \cdots, 2k-1, \end{cases} \tag{2.22}$$

and when $d = 2k + 1$

$$\phi_j^- = \begin{cases} -\phi_j^+ - \frac{\pi}{2}, & j = 0, \\ -\phi_j^+ + \frac{\pi}{2}, & j = 2k+1, \\ -\phi_j^+ - \pi, & j = 2, 4, \cdots, 2k, \\ -\phi_j^+ + \pi, & j = 1, 3, \cdots, 2k-1. \end{cases} \tag{2.23}$$

Utilizing the characteristic that the Pauli X matrix is a real matrix, we can establish that:

$$\overline{W(x)} = e^{-\mathrm{i}\arccos(x)X} = ZW(x)Z.$$

By applying the complex conjugate to the SU(2) model, we have

$$\overline{U(x,\Phi)} = e^{-\mathrm{i}\phi_0 Z}\prod_{j=1}^{d}\overline{W(x)}e^{-\mathrm{i}\phi_j Z} = Ze^{-\mathrm{i}\phi_0 Z}\prod_{j=1}^{d}W(x)e^{-\mathrm{i}\phi_j Z}Z = ZU(x,-\Phi)Z. \tag{2.24}$$

By applying Theorem 2.3.3, we can derive the following matrix representation:

$$U(x,-\Phi) = Z\overline{U(x,\Phi)}Z = \begin{pmatrix} P^*(x) & \mathrm{i}Q^*(x)\sqrt{1-x^2} \\ \mathrm{i}Q(x)\sqrt{1-x^2} & P(x) \end{pmatrix}. \tag{2.25}$$

To address the differences between $-\Phi$ and Φ^-definitions, we separately evaluate the results for even and odd cases. For the case when $d = 2k$, the result is

$$\begin{aligned} U(x,\Phi^-) &= -e^{\mathrm{i}\frac{\pi}{2}Z}U(x,-\Phi)e^{\mathrm{i}\frac{\pi}{2}Z} = ZU(x,-\Phi)Z = \overline{U(x,\Phi)} \\ &= \begin{pmatrix} P^*(x) & -\mathrm{i}Q^*(x)\sqrt{1-x^2} \\ -\mathrm{i}Q(x)\sqrt{1-x^2} & P(x) \end{pmatrix}. \end{aligned} \tag{2.26}$$

Conversely, when $d = 2k+1$, the result is

$$\begin{aligned} U(x,\Phi^-) &= e^{-\mathrm{i}\frac{\pi}{2}Z}U(x,-\Phi)e^{\mathrm{i}\frac{\pi}{2}Z} = ZU(x,-\Phi)Z = \overline{U(x,\Phi)} \\ &= \begin{pmatrix} P^*(x) & -\mathrm{i}Q^*(x)\sqrt{1-x^2} \\ -\mathrm{i}Q(x)\sqrt{1-x^2} & P(x) \end{pmatrix}. \end{aligned} \tag{2.27}$$

These prove the theorem. □

In most applications, our interest primarily lies in utilizing the real part of P. As a direct consequence of the preceding theorem, the application of the quantum circuit depicted in Fig. 2.3 (c) involves transforming a block encoding U_A into another block encoding. In this transformation, the submatrix undergoes modification via a real polynomial. This relationship is further expounded in the following corollary.

Corollary 2.3.6 (QSVT through real polynomials)**.** *Let $f(x) = \mathrm{Re}[P(x)]$ be the real component of the complex-valued matrix P. By conjugating the QSVT circuit, the matrix representation of Fig. 2.3 (c) is*

$$\begin{pmatrix} f_{\mathrm{SV}}(A) & * & * & * \\ * & * & * & * \\ * & * & f_{\mathrm{SV}}(A) & * \\ * & * & * & * \end{pmatrix}. \tag{2.28}$$

The following corollary is a slight variation of [67, Corollary 5], which states that the condition on the real part of P can be easily satisfied. Due to the relation between the real and imaginary components given in Eq. (4.4), the conditions on the imaginary part of P are the same.

Corollary 2.3.7 (Quantum signal processing with real target polynomials [67, Corollary 5])**.** *Let $f \in \mathbb{R}[x]$ be a degree-d polynomial for some $d \geq 1$ such that*

1. *$f(x)$ has parity $(d \mod 2)$,*
2. *$|f(x)| \leq 1, \forall x \in [-1, 1]$.*

Then there exists some $P, Q \in \mathbb{C}[x]$ satisfying properties (1)-(3) of Theorem 2.3.3 such that $f(x) = \mathrm{Re}[P(x)]$.

These findings highlight the practicality of the QSVT circuits depicted in Fig. 2.3 (c), demonstrating their capability in implementing polynomial transformations of block-encoded matrices on quantum computers. An illustration of QSP is provided in Fig. 2.4.

2.4 Symmetric quantum signal processing

Given a target polynomial $f \in \mathbb{R}[x]$ satisfying (1) $\deg(f) = d$, (2) the parity of f is $d \bmod 2$, (3) $\|f\|_\infty := \max_{x\in[-1,1]} |f(x)| < 1$, the problem of quantum signal processing (QSP) [101] is to find a set of parameters (called phase factors) $\Phi := (\phi_0, \cdots, \phi_d) \in [-\pi, \pi)^{d+1}$ so that

$$f(x) = g(x, \Phi) := \mathrm{Re}[\langle 0|U(x, \Phi)|0\rangle], \quad x \in [-1, 1], \tag{2.29}$$

where $U(x, \Phi)$ is defined in Theorem 2.3.3.

When the phase factors are restricted to be symmetric, *i.e.*,

$$\Phi = (\phi_0, \phi_1, \phi_2, \ldots, \phi_2, \phi_1, \phi_0) \in [-\pi, \pi)^{d+1}, \tag{2.30}$$

this is referred to as the symmetric quantum signal processing. The simplest example is $\Phi = (0, \ldots, 0)$. This gives $U(x, \Phi) = e^{id \arccos(x) X}$ and $g(x, \Phi) = \cos(d \arccos(x)) = T_d(x)$, where T_d is the Chebyshev polynomial of the first kind of degree d.

Due to the parity constraint, the number of degrees of freedom in the target polynomial $f(x)$ is $\widetilde{d} := \lceil \frac{d+1}{2} \rceil$. Hence $f(x)$ is entirely determined by its values on $\widetilde{d}$ distinct points. Throughout the dissertation, we choose these points to be $x_k = \cos\left(\frac{2k-1}{4\widetilde{d}}\pi\right)$, $k = 1, ..., \widetilde{d}$, *i.e.*, positive nodes of the Chebyshev polynomial

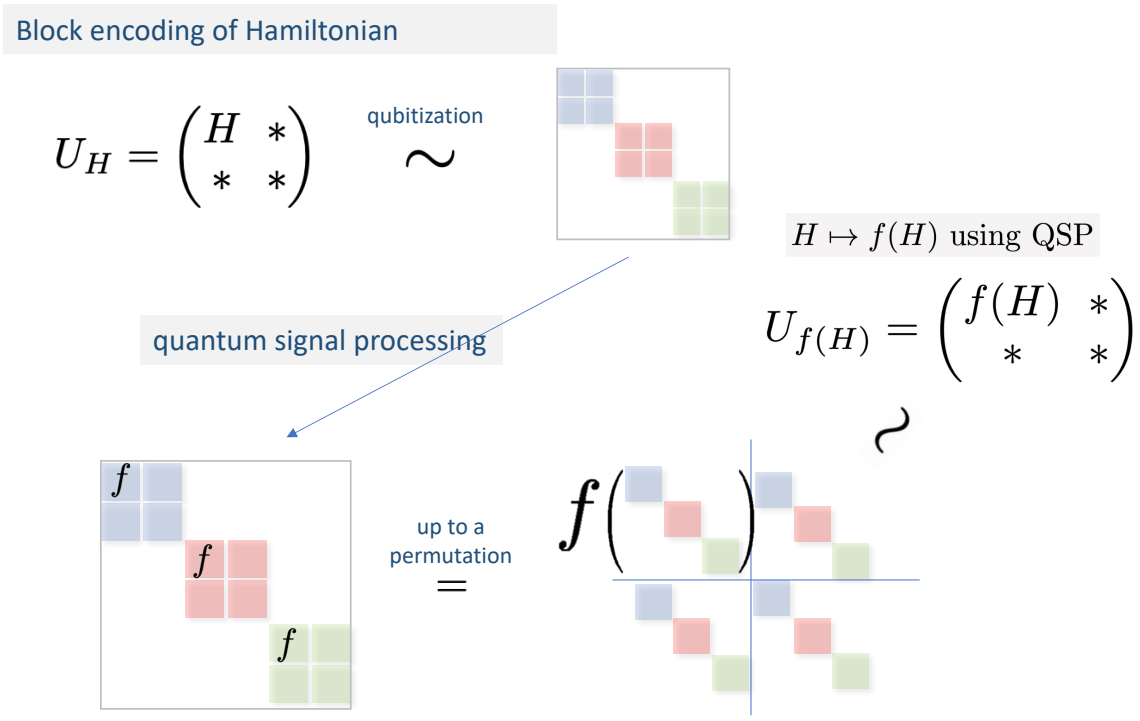


Figure 2.4: An illustration of QSP.

$T_{2\tilde{d}}(x)$. For any target polynomial $f(x)$ defined above, the solution to Eq. (2.29) exists [101, 67], and Ref. [53] suggests that the solution can be restricted to be symmetric. The existence of the solution implies that the problem of symmetric quantum signal processing can be equivalently solved via the following optimization problem

$$\Phi^* = \underset{\substack{\Phi\in[-\pi,\pi)^{d+1},\\ \text{symmetric.}}}{\arg\min}\ L(\Phi),\ L(\Phi) := \frac{1}{\tilde{d}}\sum_{k=1}^{\tilde{d}} |g(x_k,\Phi) - f(x_k)|^2, \tag{2.31}$$

i.e., any solution Φ^* to Eq. (2.29) achieves the global minimum of the cost function with $L(\Phi^*) = 0$, and vice versa.

However, the energy landscape of the cost function $L(\Phi)$ is very complex, and has numerous global as well as local minima (see [153]). This is already the case with the symmetry constraint. (Without the symmetry constraint, the number of variables is larger than the number of equations and there should be an infinite number of global minima.) Starting from a random initial guess, an optimization algorithm can easily

be trapped at a local minima already when d is small[1]. It is therefore surprising that starting from a special symmetric initial guess

$$\Phi^0 = (\pi/4, 0, 0, \ldots, 0, 0, \pi/4), \tag{2.32}$$

which is independent of target function, at least one global minimum can be robustly identified using standard unconstrained optimization algorithms even when d is as large as $10,000$ [53], and the optimization method is free from being trapped by any local minima. Direct calculation shows that $g(x, \Phi^0) = 0$, and therefore Φ^0 does not contain any *a priori* information of the target polynomial $f(x)$!

Let the domain of the symmetric phase factors be

$$D_d = \begin{cases} [-\frac{\pi}{2}, \frac{\pi}{2})^{\frac{d}{2}} \times [-\pi, \pi) \times [-\frac{\pi}{2}, \frac{\pi}{2})^{\frac{d}{2}}, & d \text{ is even}, \\ [-\frac{\pi}{2}, \frac{\pi}{2})^{d+1}, & d \text{ is odd}. \end{cases} \tag{2.33}$$

The following result presents the existence and uniqueness of symmetric phase factors for a class of polynomial matrices in SU(2).

Theorem 2.4.1 (Existence and uniqueness of symmetric phase factors [153, Theorem 1])**.** *Consider any $P \in \mathbb{C}[x]$ and $Q \in \mathbb{R}[x]$ satisfying the following conditions*

(1) $\deg(P) = d$ *and* $\deg(Q) = d - 1$.

(2) P has parity $(d \bmod 2)$ *and Q has parity* $(d - 1 \bmod 2)$.

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

(4) If d is odd, then the leading coefficient of Q is positive.

There exists a unique set of symmetric phase factors $\Phi := (\phi_0, \phi_1, \cdots, \phi_1, \phi_0) \in D_d$ such that

$$U(x, \Phi) = \begin{pmatrix} P(x) & \mathrm{i}Q(x)\sqrt{1 - x^2} \\ \mathrm{i}Q(x)\sqrt{1 - x^2} & P^*(x) \end{pmatrix}. \tag{2.34}$$

Theorem 2.4.1 implies that there is a bijection between the global minimizers of Eq. (2.31) and all possible pairs of $(P(x), Q(x))$ satisfying the assumption in Theorem 2.4.1 and $\mathrm{Re}[P](x) = f(x)$. The conditions (1), (2) for the target polynomial f are compatible with the first two requirements in Theorem 2.4.1. The condition (3) on the maxnorm, *i.e.*, $\|f\|_\infty < 1$ is compatible with the normalization condition, which is itself a natural condition due to the unitarity of $U(x, \Phi)$.

[1] An early attempt showed that even finding Φ with $d > 30$ can be very costly [42, Apendix H.3].

2.5 Quantum eigenvalue transformation for unitary matrices

One key distinction of symmetric QSP, as compared to traditional QSP, is that the resulting polynomial Q is a real polynomial, not a complex-valued one. This distinction gives rise to a theory of polynomial transformations known as QETU, as referenced in [51]. QETU facilitates polynomial transformation using Hamiltonian evolution input $e^{-\mathrm{i}H}$ instead of the block encoding input U_H. Analogous to QSP in SU(2), this section introduces a SU(2) version of QETU, and subsequently extends this framework to higher dimensions, effectively elevating it to the more comprehensive QETU.

Let us revisit the formula introduced earlier

$$W(x) = e^{i \arccos(x) X} = \begin{pmatrix} x & \mathrm{i}\sqrt{1-x^2} \\ \mathrm{i}\sqrt{1-x^2} & x \end{pmatrix}, \quad x \in [-1, 1]. \tag{2.35}$$

We first state the result of quantum signal processing for real polynomials [66, Corollary 10], and specifically the symmetric quantum signal processing [153, Theorem 10] in Theorem 2.5.1.

Theorem 2.5.1 (Symmetric quantum signal processing, W-convention)**.** *Given a real polynomial $F(x) \in \mathbb{R}[x]$, and* $\deg F = d$*, satisfying*

1. *F has parity $d \bmod 2$,*
2. *$|F(x)| \leq 1, \forall x \in [-1, 1]$,*

then there exists polynomials $G(x), Q(x) \in \mathbb{R}[x]$ and a set of symmetric phase factors $\Phi := (\phi_0, \phi_1, \cdots, \phi_1, \phi_0) \in \mathbb{R}^{d+1}$ such that the following QSP representation holds:

$$e^{i\phi_0 Z} \prod_{j=1}^{d} \left[W(x) e^{i\phi_j Z} \right] = \begin{pmatrix} F(x) + iG(x) & iQ(x)\sqrt{1-x^2} \\ iQ(x)\sqrt{1-x^2} & F(x) - iG(x) \end{pmatrix}, \tag{2.36}$$

In order to derive QETU, we define

$$W_z(x) = e^{i \arccos(x) Z} = \begin{pmatrix} e^{i \arccos(x)} & 0 \\ 0 & e^{-i \arccos(x)} \end{pmatrix}, \quad x \in [-1, 1]. \tag{2.37}$$

Then Theorem 2.5.2 is equivalent to Theorem 2.5.1, but uses the variable x is encoded in the W_z matrix instead of the W matrix.

Theorem 2.5.2 (Symmetric quantum signal processing, W_z-convention)**.** *Given a real, even polynomial $F(x) \in \mathbb{R}[x]$, and $\deg F = d$, satisfying $|F(x)| \leq 1, \forall x \in [-1,1]$, then there exists polynomials $G(x), Q(x) \in \mathbb{R}[x]$ and a symmetric phase factors $\Phi_z := (\varphi_0, \varphi_1, \cdots, \varphi_1, \varphi_0) \in \mathbb{R}^{d+1}$ such that the following QSP representation holds:*

$$\begin{aligned} U_{\Phi_z}(x) &= e^{i\varphi_0 X} W_z^*(x) e^{i\varphi_1 X} W_z(x) e^{i\varphi_2 X} \cdots e^{i\varphi_2 X} W_z^*(x) e^{i\varphi_1 X} W_z(x) e^{i\varphi_0 X} \\ &= \begin{pmatrix} F(x) & -Q(x)\sqrt{1-x^2} + iG(x) \\ Q(x)\sqrt{1-x^2} + iG(x) & F(x) \end{pmatrix}. \end{aligned} \tag{2.38}$$

Proof. Using

$$e^{\mathrm{i}\varphi X} = \mathrm{H} e^{i\varphi Z} \mathrm{H}, \tag{2.39}$$

and

$$\mathrm{H} W_z(x) \mathrm{H} = W(x), \quad \mathrm{H} W_z^*(x) \mathrm{H} = -e^{-i\frac{\pi}{2}Z} W(x) e^{-i\frac{\pi}{2}Z}, \tag{2.40}$$

we have

$$\begin{aligned} U_{\Phi_z}(x) =& (-1)^{\frac{d}{2}} \mathrm{H} \Big\{ e^{i(\varphi_0 - \frac{\pi}{2})Z} W(x) e^{i(\varphi_1 - \frac{\pi}{2})Z} W(x) e^{i(\varphi_2 - \frac{\pi}{2})Z} \cdots \\ & e^{i(\varphi_2 - \frac{\pi}{2})Z} W(x) e^{i(\varphi_1 - \frac{\pi}{2})Z} W(x) e^{i\varphi_0 Z} \Big\} \mathrm{H} \\ =& (-1)^{\frac{d}{2}} \mathrm{H} e^{-i\frac{\pi}{4}Z} \Big\{ e^{i(\varphi_0 - \frac{\pi}{4})Z} W(x) e^{i(\varphi_1 - \frac{\pi}{2})Z} W(x) e^{i(\varphi_2 - \frac{\pi}{2})Z} \cdots \\ & e^{i(\varphi_2 - \frac{\pi}{2})Z} W(x) e^{i(\varphi_1 - \frac{\pi}{2})Z} W(x) e^{i(\varphi_0 - \frac{\pi}{4})Z} \Big\} e^{i\frac{\pi}{4}Z} \mathrm{H}. \end{aligned} \tag{2.41}$$

The term in the parenthesis satisfies the condition of Theorem 2.5.1. We may choose a symmetric phase factor $(\phi_0, \phi_1, \ldots, \phi_1, \phi_0)$, so that

$$\begin{aligned} & e^{i\phi_0 Z} W(x) e^{i\phi_1 Z} W(x) e^{i\phi_2 Z} \cdots e^{i\phi_2 Z} W(x) e^{i\phi_1 Z} W(x) e^{i\phi_0 Z} \\ =& (-1)^{\frac{d}{2}} \begin{pmatrix} F(x) + iG(x) & iQ(x)\sqrt{1-x^2} \\ iQ(x)\sqrt{1-x^2} & F(x) - iG(x) \end{pmatrix}. \end{aligned} \tag{2.42}$$

Then define $\varphi_j = \phi_j + (2 - \delta_{j0})\pi/4$ for $j = 0, \ldots, d/2$, direct computation shows

$$U_{\Phi_z}(x) = \begin{pmatrix} F(x) & -Q(x)\sqrt{1-x^2} + iG(x) \\ Q(x)\sqrt{1-x^2} + iG(x) & F(x) \end{pmatrix}, \tag{2.43}$$

which proves the theorem. □

This SU(2) version can be lifted to higher dimension following a similar analysis as that in the qubitization structure of QSP. The quantum circuit share the commonality with Fig. 2.3 (c) which is depicted in Fig. 2.5.

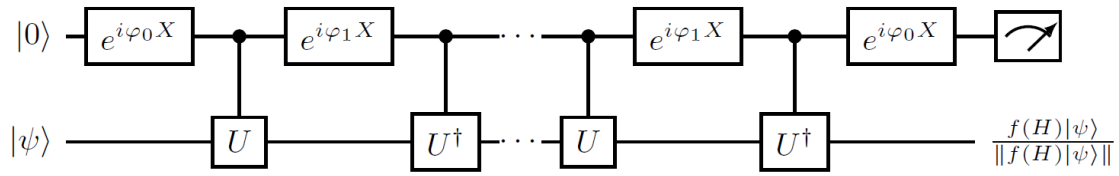


Figure 2.5: Representation of QETU circuits. For a very general class of functions f the circuit (c) can approximately prepare a normalized quantum state $f(H)\,|\psi\rangle\,/\,\|f(H)\,|\psi\rangle\|$ with approximate success probability $p = \|f(H)\,|\psi\rangle\|^2$, by interleaving the forward (U) and backward ($U^\dagger$) time evolution with some properly chosen X-rotations in the ancilla qubit.

Theorem 2.5.3 (QETU)**.** *Let $U = e^{-iH}$ with an n-qubit Hermitian matrix H. For any even real polynomial $F(x)$ of degree d satisfying $|F(x)| \leq 1, \forall x \in [-1, 1]$, we can find a sequence of symmetric phase factors $\Phi_z := (\varphi_0, \varphi_1, \cdots, \varphi_1, \varphi_0) \in \mathbb{R}^{d+1}$, such that the circuit in Fig. 2.5 denoted by $\mathcal{U}$ satisfies $(\langle 0| \otimes I_n)\mathcal{U}(|0\rangle \otimes I_n) = F\left(\cos\frac{H}{2}\right)$.*

Proof. For any eigenstate $|v_j\rangle$ of H with eigenvalue λ_j, note that $\mathrm{span}\{|0\rangle\,|v_j\rangle\,, |1\rangle\,|v_j\rangle\}$ is an invariant subspace of $U, U^\dagger, X \otimes I_n$, and hence of $\mathcal{U}$. Together with the fact that for any phase factors φ, φ',

$$e^{i\varphi X} W_z^*(x) e^{i\varphi' X} W_z(x) = e^{i\varphi X} \begin{pmatrix} 1 & 0 \\ 0 & e^{2i\arccos(x)} \end{pmatrix} e^{i\varphi' X} \begin{pmatrix} 1 & 0 \\ 0 & e^{-2i\arccos(x)} \end{pmatrix}, \tag{2.44}$$

we have

$$\mathcal{U}\,|0\rangle\,|v_j\rangle = (U_{\Phi_z}(\cos(\lambda_j/2))\,|0\rangle)\,|v_j\rangle = F(\cos(\lambda_j/2))\,|0\rangle\,|v_j\rangle + \alpha_j\,|1\rangle\,|v_j\rangle\,. \tag{2.45}$$

Here we have used $x = \cos(\lambda_j/2)$, and

$$\alpha_j = Q(\cos(\lambda_j/2))\sin(\lambda_j/2) + iG(\cos(\lambda_j/2)) \tag{2.46}$$

is an irrelevant constant according to Eq. (2.38).

Since any state $|\psi\rangle$ can be expanded as the linear combination of eigenstates $|v_j\rangle$ as

$$|\psi\rangle = \sum_j c_j\,|v_j\rangle\,, \tag{2.47}$$

we have

$$\begin{aligned} \mathcal{U}\,|0\rangle\,|\psi\rangle &= \sum_j c_j \mathcal{U}\,|0\rangle\,|v_j\rangle = |0\rangle \sum_j c_j F(\cos(\lambda_j/2))\,|v_j\rangle + |1\rangle\,|\perp\rangle \\ &= |0\rangle\,F(\cos(H/2))\,|\psi\rangle + |1\rangle\,|\perp\rangle\,, \end{aligned} \tag{2.48}$$

where $|\perp\rangle$ is some unnormalized quantum state. This proves the theorem. □

Sometimes instead of controlled U, we have direct access to an oracle that simultaneously implements a controlled forward and backward time evolution:

$$V = \begin{pmatrix} e^{iH} & 0 \\ 0 & e^{-iH} \end{pmatrix}. \tag{2.49}$$

This is the case, for instance, in certain implementation of QETU in a control-free setting. Corollary 2.5.4 describes this version of QETU.

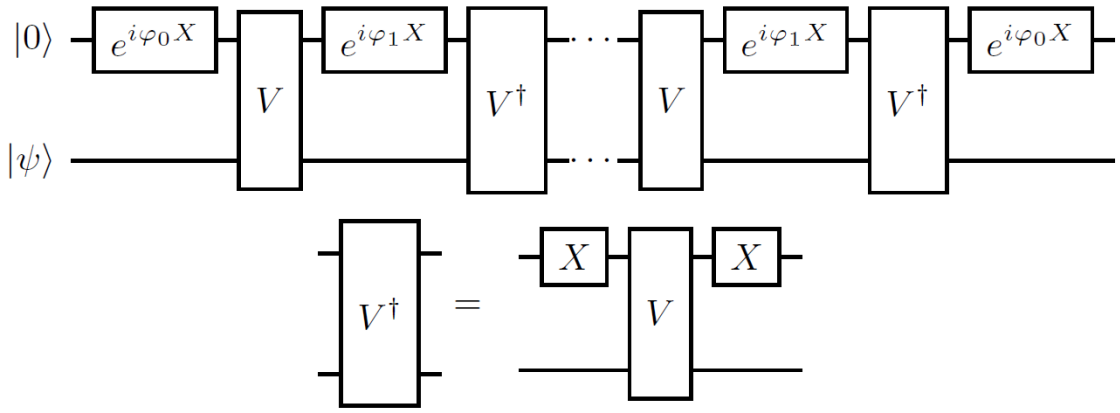


Figure 2.6: Variant of QETU with an oracle implementing the controlled forward and backward time evolution. The implementation of $V^\dagger$ can be carried out by conjugating V with Pauli X gates acting on the first qubit, and the Pauli X gate can be first combined with the phase rotation as $e^{i\varphi X}X = ie^{i(\varphi+\pi/2)X}$.

Corollary 2.5.4 (QETU with forward and backward time evolution)**.** *Let V be the unitary matrix given in Eq.* (2.49) *corresponding to an n-qubit Hermitian matrix H. For any even real polynomial $F(x)$ of degree d satisfying $|F(x)| \leq 1, \forall x \in [-1, 1]$, we can find a sequence of symmetric phase factors $\Phi_z := (\varphi_0, \varphi_1, \cdots, \varphi_1, \varphi_0) \in \mathbb{R}^{d+1}$, such that the circuit in Fig. 2.6 denoted by $\mathcal{U}$ satisfies $\langle 0|\otimes I_n)\mathcal{U}(|0\rangle \otimes I_n) = F(\cos H)$.*

Proof. Let $|v_j\rangle$ be an eigenstate of H with eigenvalue λ_j. For any phase factors φ, φ', using

$$e^{i\varphi X}W_z^*(x)e^{i\varphi' X}W_z(x) = e^{i\varphi X}\begin{pmatrix} e^{-i\arccos(x)} & 0 \\ 0 & e^{i\arccos(x)} \end{pmatrix} e^{i\varphi' X}\begin{pmatrix} e^{i\arccos(x)} & 0 \\ 0 & e^{-i\arccos(x)} \end{pmatrix}, \tag{2.50}$$

we have

$$\mathcal{U}\left|0\right\rangle\left|v_j\right\rangle = \left(U_{\Phi_z}(\cos\lambda_j)\left|0\right\rangle\right)\left|v_j\right\rangle = F(\cos\lambda_j)\left|0\right\rangle\left|v_j\right\rangle + \alpha_j\left|1\right\rangle\left|v_j\right\rangle. \tag{2.51}$$

Here $x = \cos\lambda_j$, and $\alpha_j = Q(\cos\lambda_j)\sin\lambda_j + iG(\cos\lambda_j)$. The rest of the proof follows that of Theorem 2.5.3. □

2.6 Multi-level quantum signal processing

By capitalizing on the capabilities of Hamiltonian evolution access and QETU, particularly when the Hamiltonian is amenable to fast-forwarding as discussed in [72, 136], we can propose an efficient multi-level filter-based algorithm. This algorithm is specifically designed for the preparation of the ground state and the estimation of ground-state energy.

Consider H as a positive semidefinite Hermitian matrix characterized by a large spectral radius $\|H\|$. Additionally, we assume that H can be fast-forwarded. This implies that for any $\epsilon > 0$, there exists a $\tau > 0$, which is independent of $\|H\|$, and a class of t-dependent unitary $O_H(t)$ so that

$$\sup_{t\leq\tau}\left\|O_H(t) - e^{-\mathrm{i}Ht}\right\| \leq \epsilon \tag{2.52}$$

with the implementation cost of $O_H(t)$ remaining independent of t.

Let us consider the eigen pairs of the Hamiltonian H as $\{\lambda_k, |\psi_k\rangle\}$, ordered such that $\lambda_0 < \lambda_1 \leq \lambda_2 \leq \cdots$. Our objective is to prepare the ground state $|\psi_0\rangle$, starting from an initial guess $|\phi\rangle$ that has a nonzero overlap with the ground state, quantified as $|\langle\phi|\psi_0\rangle| \geq \gamma > 0$. Additionally, we assume the existence of a value μ and a predefined spectral gap Δ, satisfying $\lambda_0 \leq \mu - \Delta/2 < \mu + \Delta/2 \leq \lambda_1$. The parameter μ can be determined using a binary search strategy. Consequently, the primary focus of our algorithm design revolves around the construction of a (trigonometric) polynomial filter.

The remainder of this section is structured as follows: Initially, we lay the groundwork for our algorithmic proposal by analyzing the problem's solution through the use of a Linear Combination of Unitaries (LCU) [41]. Subsequently, we introduce a refined method based on QSP to address the same problem. The complexity analysis of this method will be presented, demonstrating that the dependence on the spectral radius $\|H\|$ can be exponentially improved.

LCU-based method

The filter function that implements the ground-state projection can be expressed in the context of LCU as follows:

$$f(x) = \sum_{k=1}^{M} c_k e^{-\mathrm{i}xt_k}, \;\; f(H) = \sum_{k=1}^{M} c_k e^{-\mathrm{i}Ht_k}. \tag{2.53}$$

The function $f(x)$ satisfies the following conditions:

(1) $f(x) \approx 1$ for any $x < \mu - \Delta/2$,

(2) $f(x) \approx 0$ for any $\mu + \Delta/2 < x < \|H\|$,

(3) and $|f(x)| \leq 1$ for any $0 \leq x \leq \|H\|$.

The implementation of this filter can be achieved by setting $\max t_k = \mathcal{O}(1/\Delta)$ and choosing a grid size $h = \mathcal{O}(1/\|H\|)$. For simplicity, the logarithmic dependence on the precision parameter is omitted here. As a result, the number of discretization points required is $M = \mathcal{O}(\|H\|/\Delta)$. The construction of the LCU circuit necessitates the implementation of $e^{-\mathrm{i}H2^\ell h}$ where $\ell \leq \lceil \log_2 M \rceil = \lceil \log_2(\|H\|/\Delta) \rceil$. To utilize the fast-forwarding property of the Hamiltonian evolution, the implementation of $e^{-\mathrm{i}H2^\ell h}$ can be refined as follows:

$$e^{-\mathrm{i}H2^\ell h} = \begin{cases} O_H(2^\ell h) & \text{when } 2^\ell h \leq \tau \\ O_H^r(2^\ell h/r) & \text{otherwise, and } r := \lceil 2^\ell h/\tau \rceil. \end{cases} \tag{2.54}$$

Given the fast-forwarding assumption, the cost of this implementation is

$$\mathcal{O}(r) = \mathcal{O}(1/(\Delta\tau)). \tag{2.55}$$

Significantly, this cost is independent of ℓ. Note that the success probability of this procedure is γ^2 due to the initial overlap. Therefore, the overall cost of the LCU-based method is

$$\mathcal{O}(\gamma^{-2} \log_2(M)/(\Delta\tau)) = \mathcal{O}\left(\frac{1}{\Delta\tau\gamma^2} \log\left(\frac{\|H\|}{\Delta}\right)\right) \tag{2.56}$$

which exhibits only a weak dependence on the spectral radius $\|H\|$ through a logarithmic term. However, a drawback of using LCU is that the "prepare" oracle for the linear combination coefficients $\{c_k\}$ is not immediately apparent. Its implementation demands $\log_2 M$ additional ancilla qubits and multi-qubit control gates, which poses challenges for early fault-tolerant quantum devices.

Multi-level QSP-based method

The computational cost of QSP-based methods, including QETU and QSVT, typically scales linearly with the number of terms. Consequently, a direct application of these techniques, such as in QETU, results in a cost that depends linearly on the spectral radius $\|H\|$. This is partially attributed to the fact that QSP-based methods primarily query $e^{-\mathrm{i}Hh}$, thereby not fully capitalizing on the fast-forwarding assumption. In the following section, we introduce a multi-level QETU approach designed to tackle this limitation. This method significantly enhances efficiency, exponentially reducing the dependence on the spectral radius.

Let us consider a real even polynomial g which satisfies the following conditions:

(1) $g(x) \approx 1$ for any $x > \cos(\pi/8) \approx 0.92$,

(2) $g(x) \approx 0$ for any $0 < x < \cos(\pi/4) \approx 0.71$,

(3) and $|g(x)| \leq 1$ for any $x \in [-1, 1]$.

According to Theorem 2.5.3, by performing a controlled implementation of $e^{-\mathrm{i}H\pi t}$ for d times, a QETU is capable of implementing $g(\cos(H\pi t/2))$. It is noteworthy that the observed difference between functions f and g arises from the intermediate cosine transformation, which reverses their monotonicity.

Let us express the initial guess in the eigenbasis of H as:

$$|\phi\rangle = \sum_k c_k |\psi_k\rangle . \tag{2.57}$$

Then, applying $g(\cos(H\pi t/2))$ yields:

$$g(\cos(H\pi t/2)) |\phi\rangle = \sum_k c_k g(\cos(\lambda_k \pi t/2)) |\psi_k\rangle =: \sum_k c_k^{(1)} |\psi_k\rangle =: \phi^{(1)}. \tag{2.58}$$

Here, the ket notation is not applied to $\phi^{(1)}$ in order to emphasize that it represents an unnormalized state.

Given the properties of g, we observe:

(1) For $0 \leq \lambda_k < 1/(4t)$, the coefficient remains almost unchanged: $c_k^{(1)} \approx c_k$.

(2) For $1/(2t) < \lambda_k \leq 1/t$, the component is nearly eliminated: $c_k^{(1)} \approx 0$.

(3) In other cases, while the magnitude of the component may change, it does not increase: $\left|c_k^{(1)}\right| \leq |c_k|$.

By recurrently applying $g(\cos(H\pi t_\ell/2))$ with varying t_ℓ's, a sequence of unnormalized states

$$\left(\phi^{(1)}, \phi^{(2)}, \cdots, \phi^{(L)}\right)$$

is obtained. This sequential application progressively eliminates certain components. It is justified that the ground state can be prepared by selecting $t_\ell = 2^\ell / \|H\|$, setting the stopping point $L = \lfloor \log_2(\tau \|H\|) \rfloor$, and applying a distinct clean-up filter function to the state $\phi^{(L)}$. Unlike the sequential filters, this final filter is constructed by querying $O_H(\tau)$ for $\mathcal{O}(1/(\Delta\tau))$ times. Note that the success probability of this procedure is γ^2 due to the initial overlap. Consequently, the overall complexity is:

$$\mathcal{O}(1/\gamma^2) \times \mathcal{O}\left(Ld + \frac{1}{\Delta\tau}\right) = \mathcal{O}(1/\gamma^2) \times \mathcal{O}\left(d\log(\tau\|H\|) + \frac{1}{\Delta\tau}\right) = \widetilde{O}(1/(\Delta\tau\gamma^2)) \tag{2.59}$$

where the notation $\widetilde{\mathcal{O}}$ hides all logarithmic dependence. It shows a logarithmic dependence on $\|H\|$ in the overall complexity.

Chapter 3

Streamlined subroutines for enhanced applications of quantum signal processing

Although Quantum Signal Processing (QSP) excels in implementing matrix polynomials on quantum computers, its streamlined application in solving scientific computing problems necessitates an integrated algorithmic design that incorporates other useful subroutines. This chapter delves into several topics that act as streamlined subroutines to bolster the application of QSP. We begin in Section 3.1 by showcasing examples that illustrate how numerical linear algebra problems can be conceptualized as matrix functions. This abstraction lays the foundation for another crucial subroutine: developing efficient algorithms to find the best polynomial approximations for the target function. In Section 3.2, we introduce both a Remez exchange algorithm and a convex optimization-based method, tailored for efficiently numerically solving the best polynomial approximation problems. A limitation of direct QSP-based methods is the requirement for the function to be real, bounded, and of definite parity. In practical scenarios, however, the target function might be a general complex-valued function. While implementation can be approached through a divide-and-conquer strategy and the Linear Combination of Unitaries (LCU) algorithm, such methods face practical challenges, particularly in the costly synthesis of multi-qubit control gates. In Section 3.3, we explore an efficient approach to implement controlled QETU circuits and to execute general matrix polynomials based on this framework.

Please note that Section 3.1 is based on [55] (joint work with Lin Lin, Hongkang Ni, and Jiasu Wang), Section 3.2 is based on [53] (joint work with Xiang Meng, K. Birgitta Whaley, and Lin Lin) and [51] (joint work with Lin Lin and Yu Tong), and

Section 3.3 is based on [51] (joint work with Lin Lin and Yu Tong).

3.1 Conceptualizing applications through matrix function transformations

The core of quantum algorithm design based on QSP lies in the abstraction of the original problem as a matrix function transformation. This transformation allows us to encode the desired function or its polynomial approximation by finding the appropriate phase factors. In order to illustrate this procedure, we present the following examples.

Quantum Hamiltonian simulation

The problem of quantum Hamiltonian simulation involves finding an efficient method for implementing the time evolution matrix of a Hamiltonian matrix, denoted as $H \mapsto \exp(-\mathrm{i}\tau H)$, for a given evolution time τ. In Ref. [101], a near-optimal quantum Hamiltonian simulation algorithm based on QSP is proposed. This algorithm abstracts the problem into a function approximation task, where the target function $f(x) = e^{-\mathrm{i}\tau x}$ is parametrized using QSP. The Chebyshev series expansion, known as the Jacobi-Anger expansion [101], is commonly employed to approximate this target function:

$$e^{-\mathrm{i}\tau x} = J_0(x) + 2\sum_{k \text{ even}} (-1)^{k/2} J_k(\tau) T_k(x) + 2\mathrm{i}\sum_{k \text{ odd}} (-1)^{(k-1)/2} J_k(\tau) T_k(x), \tag{3.1}$$

where J_k's are the Bessel functions of the first kind. As a result, by truncating the Jacobi-Anger series, a polynomial approximation of the target function can be obtained. The real and imaginary parts of the truncated series, which approximate $\cos(\tau x)$ and $\sin(\tau x)$ respectively, serve as the target polynomials for two separate QSP phase-evaluation problems. To ensure that the truncation error is upper-bounded by ϵ_0, it is sufficient to choose the degree of truncation as $d = e|\tau|/2 + \log(1/\epsilon_0)$.

Quantum Gaussian filter

The quantum Gaussian filter is a matrix function parameterized by μ and σ. It is proportional to $\exp(-(H-\mu I)^2/\sigma^2)$, where H is the Hamiltonian matrix. This matrix function is designed to localize around the given "energy level" μ, with the

degree of localization controlled by the bandwidth parameter σ. The function suppresses eigenvalues of H that are far from μ. Ideally, one would choose μ to be close to an eigenvalue of H, allowing the matrix function to approximate the projection onto the corresponding eigenspace.

The quantum Gaussian filter serves as an intermediate subroutine for near-optimal quantum linear system solvers [95]. However, directly decomposing the defining function of the quantum Gaussian filter may result in exponentially large scaling factors due to hyperbolic functions. To address this issue and improve numerical stability, one can shift and rescale the Hamiltonian so that its eigenvalues lie in a smaller subinterval $D_\kappa = [1/\kappa, 1]$ within the positive half-axis. By employing this eigenvalue shifting technique, it is sufficient to approximate the Gaussian density function in the positive half-axis. Thus, the target function is set to $f(x) = e^{-(|x|-\mu)^2/\sigma^2}$ as an even extension.

Heaviside energy filter

Heaviside function is widely used in classical applications such as signal processing and filter design. It also plays a crucial role as a subroutine in quantum algorithms for ground-state energy estimation and ground state preparation [51].

Consider a Hamiltonian matrix that has been shifted and scaled so that its eigenvalues lie in the interval $[0, 1]$. The Heaviside energy filter $f(H)$ attenuates the high-energy components of the Hamiltonian. The function $f(x)$ is defined as follows:

$$f(x) = \begin{cases} 1 & |x| < 0.5 \\ \frac{1}{2} & |x| = 0.5 \\ 0 & |x| > 0.5 \end{cases}. \tag{3.2}$$

To address the singularity at 0.5, we assume that the target function only needs to be approximated within the interval $D_\delta = [0, (1-\delta)/2] \cup [(1+\delta)/2, 1]$. This allows us to focus on the desired energy range and mitigate the effects of the singularity.

Matrix inversion

Matrix inversion is a fundamental topic in numerical linear algebra with wide-ranging applications, including numerical optimization and least squares problems. In the context of function transformation, the equivalent problem is to implement the transformation $H \mapsto f(H) = H^{-1}$. If the matrix has a condition number of $\kappa = \mathrm{cond}(H)$, it suffices to approximate the target function $f(x) = 1/x$ on the interval $D_\kappa = [1/\kappa, 1]$ using an odd function. This allows us to focus on the desired range of the function and effectively approximate the matrix inversion operation.

3.2 Constructing polynomials that approximate a target function

To ensure numerical stability in the phase-factor evaluation method, we approximate the target functions using target polynomials that satisfy the conditions outlined in Theorem 2.4.1. Various methods have been proposed in the literature for constructing these polynomial approximations in a streamlined manner.

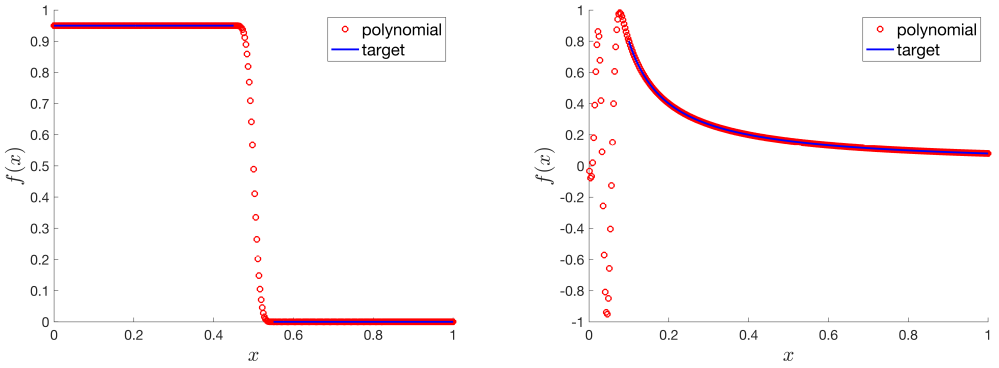
One approach is to directly truncate the Chebyshev series expansion of the target function. This can be efficiently achieved using Fast Fourier Transformation (FFT) applied to the transformed target function $f(\cos(\theta))$. However, when the target function is not defined on the entire interval $[-1, 1]$, the truncated series polynomial may not be bounded by 1 on the entire interval, making it unsuitable for representation using QSP. To address this issue, one approach is to use the Remez exchange algorithm proposed in Ref. [53] to find the best polynomial approximation for the partially defined target function. Another method involves numerically finding the best polynomial approximation using a convex optimization-based approach as described in Ref. [51, Section IV]. We demonstrate the convex optimization-based approach on examples defined in the previous section where the target function is defined on a further subinterval. The resulting target polynomials, obtained using the convex optimization-based method, are visualized in Fig. 3.1.

Remez Method

The objective of identifying the best approximation polynomial in terms of the L^∞ error involves solving the following min-max problem:

$$f^* = \arg\min_{f \in \mathbb{R}_d[x]} \max_{x \in [a,b]} |h(x) - f(x)|. \tag{3.3}$$

Due to the parity constraint stipulated by Theorem 2.4.1, our ansatz is constructed as a linear combination of a general basis of functions $\{g_1(x), \ldots, g_N(x)\}$, instead of the standard monomial basis $\{1, x, \ldots, x^d\}$. To align with the degree of freedom, we set $N = \lceil \frac{d+1}{2} \rceil$. A function set $\{g_1(x), \ldots, g_N(x)\}$ satisfies the Haar condition on a set X if each $g_j(x)$ is continuous, and for any N distinct points $x_1, \ldots, x_N \in X$, the vectors $v_j := (g_1(x_j), \ldots, g_N(x_j))$, $1 \leq j \leq N$, are linearly independent, as defined by Haar [74]. For example, the Haar condition is met if we select $g_j(x) = T_{2j-1}(x)$ (or $T_{2j-2}(x)$) and $X \subset (0, 1]$, providing the best odd (even) approximation polynomial. The imposition of the Haar condition simplifies solving the generalized approximation problem.



(a) (b)

Figure 3.1: Polynomial approximation of the target functions obtained by the convex-optimization-based method. (a) Heaviside energy filter function and its polynomial approximation with $\delta = 0.1$. (b) Matrix inversion function and its polynomial approximation with $\kappa = 10$.

The optimal approximation polynomial f^* using the linear combination of functions $\{g_1(x), \ldots, g_N(x)\}$ can be determined using the Remez exchange method, outlined in Algorithm 3.2.1. This method computes a series of approximation polynomials on discrete sets and the resulting polynomial sequence uniformly converges to the optimal polynomial f^* with a linear convergence rate. For a wide range of functions h, this rate can be improved to quadratic. Further details on the Remez method can be found in [36, Chapter 3].

Algorithm 3.2.1 Remez method for solving the best approximation polynomial

Input: An interval $[a, b] \subset \mathbb{R}$, target function F, a basis $\{g_1, \ldots, g_N\}$ satisfying the Haar condition, $N + 1$ initial points $a \leq x_0 \leq \cdots \leq x_N \leq b$.

1: Set $t = 0$.
2: **while** stopping criterion is not satisfied **do**
3: Set $t = t + 1$.
4: Solve the linear equation for $a_1, \ldots, a_N$ and Δ

$$\sum_{j=1}^{N} a_j g_j(x_k) - h(x_k) = (-1)^k \Delta, \ k = 0, \ldots, N.$$

5: Denote $f_t(x) = \sum_{j=1}^{N} a_j g_j(x)$ and residual $r(x) = h(x) - f_t(x)$.

6: $r(x)$ has a root $z_j \in (x_{j-1}, x_j)$ for $j = 1, \ldots, N$. Set $z_0 = a$ and $z_{N+1} = b$.
7: Let $\sigma_j = \mathrm{sgn}(r(x_j))$. Find $y_j = \arg\max_{y \in [z_j, z_{j+1}]} \sigma_j r(y)$ for each $j = 0, \ldots, N$.
8: **if** $\|r(x)\|_\infty > \max_j |r(y_j)|$ **then**
9: Choose

$$y = \arg\max_{y \in [a,b]} |r(y)|.$$

10: Replace a $y_k \in \{y_0, \ldots, y_N\}$ by y in such a way that the values of $r(y)$ on the resulting ordered set still satisfies

$$r(y_j) r(y_{j+1}) < 0,\ j = 0, \ldots, N-1.$$

11: **end if**
12: Replace $\{x_0, \ldots, x_N\}$ by $\{y_0, \ldots, y_N\}$.
13: **end while**
Output: an approximation to the best approximation polynomial $f_t(x)$

Convex optimization-based approach

To find a min-max polynomial approximation to a general even target function $h(x)$ on a set $\mathcal{I} \subseteq [-1, 1]$ satisfying $|h(x)| \leq \alpha < 1, x \in \mathcal{I}$, we may solve the optimization problem on sampled points $\{x_j\}_{j=1}^M \subset \mathcal{I}$

$$\begin{aligned} \min_{\{c_k\}} \quad & \max_{x_j \in \mathcal{I}} |g(x_j) - h(x_j)| \\ \text{s.t.} \quad & g(x_j) = \sum_k A_{jk} c_k, \quad |g(x_j)| \leq \alpha, \quad \forall j = 0, \ldots, M-1. \end{aligned} \tag{3.4}$$

Here, we define the coefficient matrix, $A_{jk} = T_{2k}(x_j), k = 0, \ldots, d/2$, according to the expansion

$$g(x) = \sum_{k=0}^{d/2} T_{2k}(x) c_k. \tag{3.5}$$

This is a convex optimization problem and can be solved using software packages such as CVX [70]. The norm constraint $|g(x)| \leq 1$ is relaxed to $|g(x_j)| \leq \alpha$ to take into account that the constraint can only be imposed on the sampled points, and the values of $|g(x)|$ may slightly overshoot on $[-1, 1] \backslash \{x_j\}_{j=0}^{M-1}$. The effect of this relaxation is negligible in practice and we can choose c to be sufficiently close to 1 (for instance, α can be 0.999). Since Eq. (3.4) approximately solves the original min-max problem on $\mathcal{I}$, it achieves the near-optimal solution (in the sense of the L^∞ norm) by definition both in the asymptotic and pre-asymptotic regimes.

3.3 Control-free implementation of the controlled QETU and complex-valued polynomial transformation

In this section, we demonstrate that the controlled QETU circuit can be simplified, eliminating the need for accessing controlled Hamiltonian evolution. Utilizing the structure of QETU circuits as illustrated in Fig. 2.6, we can implement the controlled circuit by merely controlling single-qubit X rotations, which significantly reduces implementation costs. The analysis is depicted in Fig. 3.2.

This simplification is possible due to the observation that forward and backward time evolutions in the QETU circuit are interleaved in pairs. Thus, by deactivating all single-qubit X rotations, the forward and backward time evolutions effectively cancel each other out. As a result, an efficient implementation of the controlled QETU is achieved, requiring only the control of single-qubit X rotations.

This approach also significantly streamlines the implementation of matrix functions via complex-valued polynomials. While QSP is a powerful framework, its direct application is subject to certain constraints, as the function must be a bounded real polynomial of a specific parity. To implement a general complex-valued function, a divide-and-conquer strategy is employed. Initially, the function is decomposed into four components:

$$h(x) = h_{\mathrm{Re,even}}(x) + h_{\mathrm{Re,odd}}(x) + \mathrm{i}h_{\mathrm{Im,even}}(x) + \mathrm{i}h_{\mathrm{Im,odd}}(x). \tag{3.6}$$

Defined as:

$$\begin{aligned} h_{\mathrm{Re,even}}(x) &:= \mathrm{Re}\left(h(x) + h(-x)\right)/2, \quad h_{\mathrm{Re,odd}}(x) := \mathrm{Re}\left(h(x) - h(-x)\right)/2, \\ h_{\mathrm{Im,even}}(x) &:= \mathrm{Im}\left(h(x) + h(-x)\right)/2, \quad h_{\mathrm{Im,odd}}(x) := \mathrm{Im}\left(h(x) - h(-x)\right)/2. \end{aligned} \tag{3.7}$$

Each component is a bounded real function with definite parity, parametrized by individual QSP-based circuits. To synthesize the matrix function $h(H)$, an addition operation via LCU is necessary. This operation typically requires access to controlled circuits of QSP-based methods. Direct implementation often demands either controlled block encoding or controlled Hamiltonian evolution, both of which are resource-intensive.

However, these challenges can be mitigated using QETU. The key advantages leveraged are the simplicity of shifting the Hamiltonian in time evolution circuits and a control-free implementation of controlled QETU circuits.

The first insight is that access to scaled and shifted Hamiltonian evolution of $\widetilde{H} := \alpha H + \beta I \succ 0$ can be achieved by adjusting the evolution time and adding a

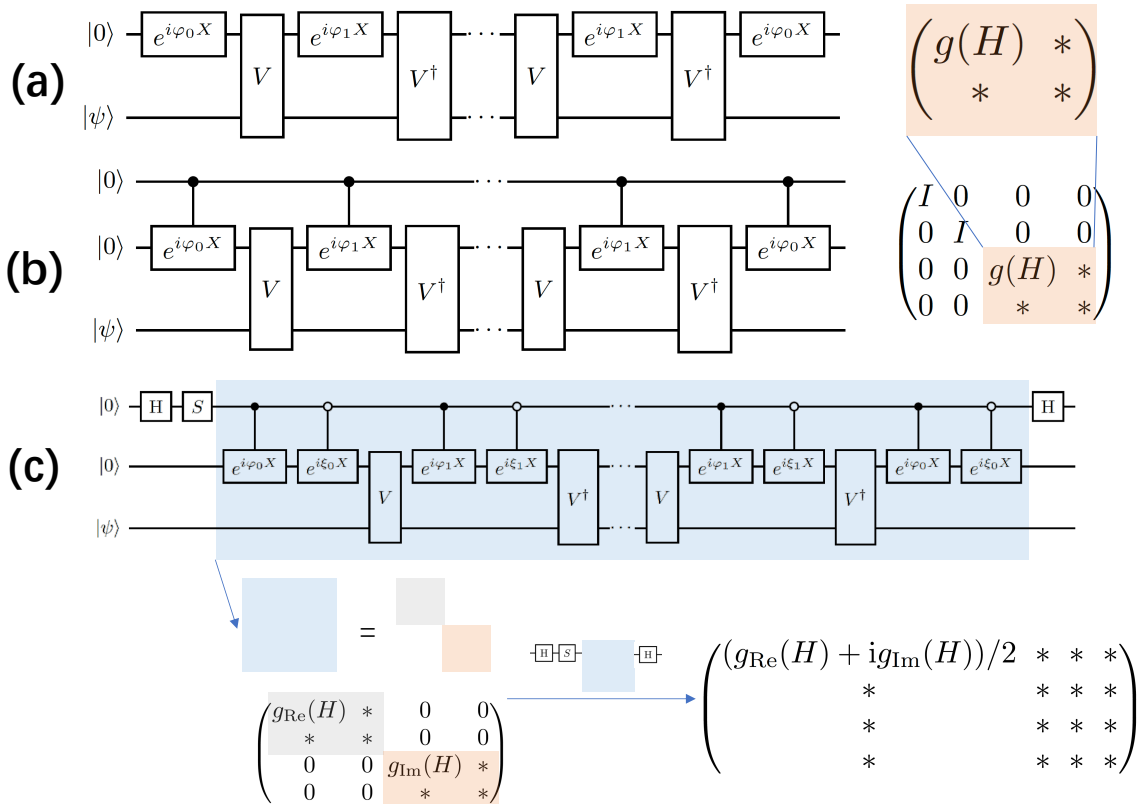


Figure 3.2: Quantum circuits visualizing the controlled QETU circuits. (a) The quantum circuit of QETU and its matrix representation. (b) The control-free implementation of controlled QETU circuits and its matrix presentation. (c) Using LCU and the control-free implementation of controlled QETU to implement general matrix functions efficiently.

Z rotation. Utilizing this capability, we can reposition the function of interest to a subinterval on the positive axis and smoothly extend it to an even function:

$$\widetilde{h}(x) = h((x-\beta)/\alpha) = \widetilde{h}_{\mathrm{Re}}(x) + \mathrm{i}\widetilde{h}_{\mathrm{Im}}(x). \tag{3.8}$$

In this case, both $\widetilde{h}_{\mathrm{Re}}(x)$ and $\widetilde{h}_{\mathrm{Im}}(x)$ are real even functions and can be approximated using QETU circuits. This method effectively reduces the number of function segments from four to two. Consequently, the controlled QETU circuits can be implemented with ease, and $h(H) = \widetilde{h}(\widetilde{H})$ is obtained through a subsequent LCU operation without significantly increasing the cost. The procedure is visualized in Fig. 3.3.

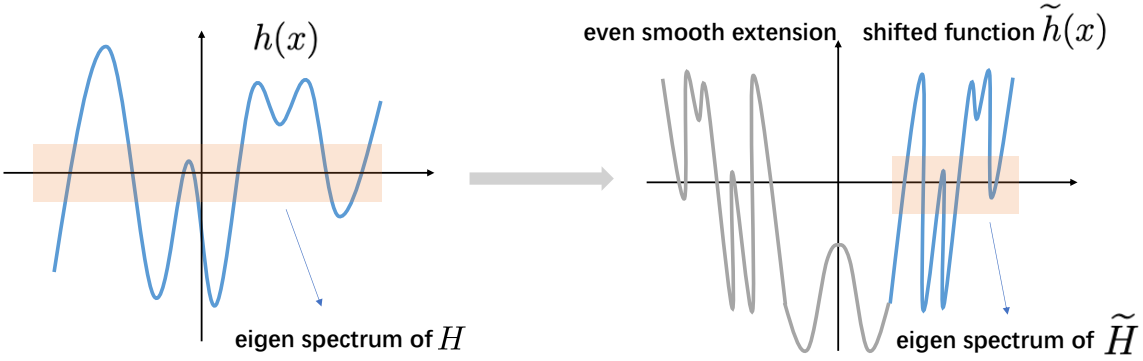


Figure 3.3: Graphical demonstration of the implementation of matrix functions via Eq. (3.8).

Chapter 4

Efficient iterative algorithms for phase-factor evaluation in quantum signal processing

Many scientific computing problems can be viewed as implementing matrix functions $A \mapsto f(A)$. For simplicity we can assume that A is a Hermitian matrix with eigenvalues in $[-1, 1]$, and that $f : \mathbb{R} \to \mathbb{R}$ is a real polynomial. Quantum signal processing (QSP) [101, 99, 67, 105] provides a systematic approach and a compact quantum circuit for implementing a broad class of matrix functions on quantum computers. This leads to efficient algorithms for various quantum applications, including linear system solving, Hamiltonian system simulation, ground-state energy estimation, and quantum benchmarking [101, 67, 95, 105, 51, 50, 52, 108, 49]. Recently, the construction of QSP has been studied and generalized using advanced theoretical tools [130, 131, 132].

Since any continuous function can be approximated using polynomials, the key idea behind QSP is to represent a target polynomial as a product of matrices in the special unitary group SU(2), parameterized by a set of phase factors denoted as Φ. However, due to the constraints of SU(2), the target polynomial must satisfy specific conditions.

Definition 4.0.1 (Target polynomial of QSP)**.** *A polynomial $f \in \mathbb{R}[x]$ is called a* target polynomial *of quantum signal processing if it satisfies (1) $\deg(f) = d$, (2) the parity of f is $(d \bmod 2)$, (3) $\|f\|_\infty := \max_{x\in[-1,1]} |f(x)| \leq 1$. Furthermore, f is called* fully-coherent *if $\|f\|_\infty = 1$.*

The mapping from phase factors $\Psi \in \mathbb{R}^{d+1}$ to the target polynomial of degree d (described by its Chebyshev coefficients denoted by $c \in \mathbb{R}^{d+1}$) can be abstractly

written as

$$F(\Psi) = c. \tag{4.1}$$

The mapping F is highly nonlinear and is not one-to-one. For a given c, and our goal is to find *a* solution to the nonlinear system (4.1).

QSP in the fully-coherent regime (or near fully-coherent regime, where $\|f\|_\infty = 1 - \delta$ for a small $\delta > 0$) finds applications in quantum algorithms for Hamiltonian simulation [104] and time-marching based simulation of non-Hermitian dynamics [58]. As will be discussed below, this problem is particularly challenging in the fully-coherent regime where the Jacobian matrix of F is very ill-conditioned (see the numerical section for an illustration of this phenomenon).

In this chapter, we introduce a series of efficient iterative algorithms designed for evaluating QSP phase factors. We will present these algorithms in detail and outline their convergence analysis. Extensive numerical tests have been conducted to validate the efficiency and robustness of these methods. Remarkably, to date, we have not encountered any instances of failure. Furthermore, these methods have been implemented in the QSPPACK software package [1].

Please note that Section 4.1 is based on [55] (joint work with Lin Lin, Hongkang Ni, and Jiasu Wang), Section 4.3 is based on [53] (joint work with Xiang Meng, K. Birgitta Whaley, and Lin Lin), Section 4.4 is based on [153] (joint work with Jiasu Wang and Lin Lin), Section 4.5 is based on [54] (joint work with Lin Lin, Hongkang Ni, and Jiasu Wang), and Sections 4.6 and 4.7 are based on [55] (joint work with Lin Lin, Hongkang Ni, and Jiasu Wang).

4.1 Preliminaries

Quantum signal processing (QSP) represents a class of polynomials in terms of SU(2) matrices, which is parameterized by phase factors [67, Theorem 4]. The phase factors $\Psi = (\psi_0, \psi_1, \cdots, \psi_d) \in \mathbb{R}^{d+1}$ are symmetric if they satisfy the constraint $\psi_i = \psi_{d-i}$ for any $i = 0, \cdots, d$. Ref. [153] proposes a variant of QSP representation focusing on symmetric phase factors:

Theorem 4.1.1 (Quantum signal processing with symmetric phase factors [153, Theorem 1])**.** *Consider any $P \in \mathbb{C}[x]$ and $Q \in \mathbb{R}[x]$ satisfying the following conditions:*

1. $\deg(P) = d$ *and* $\deg(Q) = d - 1$,

[1] The examples are available on the website `https://qsppack.gitbook.io/qsppack/` and the codes are open-sourced in `https://github.com/qsppack/QSPPACK`.

2. *P has parity $(d \bmod 2)$ and Q has parity $(d-1 \bmod 2)$,*

3. *(Normalization condition) $\forall x \in [-1,1] : |P(x)|^2 + (1-x^2)|Q(x)|^2 = 1$,*

4. *If d is odd, then the leading coefficient of Q is positive.*

There exists a unique set of symmetric phase factors $\Psi := (\psi_0, \psi_1, \cdots, \psi_d) \in D_d$ such that

$$U(x,\Psi) = e^{\mathrm{i}\psi_0 Z} \prod_{j=1}^{d} \left[W(x) e^{\mathrm{i}\psi_j Z} \right] = \begin{pmatrix} P(x) & \mathrm{i}Q(x)\sqrt{1-x^2} \\ \mathrm{i}Q(x)\sqrt{1-x^2} & P^*(x) \end{pmatrix}, \tag{4.2}$$

where

$$D_d = \begin{cases} [-\frac{\pi}{2}, \frac{\pi}{2})^{\frac{d}{2}} \times [-\pi, \pi) \times [-\frac{\pi}{2}, \frac{\pi}{2})^{\frac{d}{2}}, & d \text{ is even,} \\ [-\frac{\pi}{2}, \frac{\pi}{2})^{d+1}, & d \text{ is odd.} \end{cases} \tag{4.3}$$

In the above equations, X and Z denote Pauli matrices, and $P^*(x)$ represents the complex conjugate of the complex polynomial $P(x)$ obtained by conjugating all its coefficients.

In most applications, only either the real or the imaginary part of the complex polynomial $P(x) = \langle 0|U(x,\Psi)|0\rangle$ is relevant. It can be shown that these two parts can be exchanged by conjugating the unitary matrix product with $\pi/4$-Z rotation, that is,

$$\mathrm{Re}[\langle 0|U(x,\Psi)|0\rangle] = \mathrm{Im}\left[\langle 0|e^{\mathrm{i}\frac{\pi}{4}Z} U(x,\Psi) e^{\mathrm{i}\frac{\pi}{4}Z}|0\rangle\right]. \tag{4.4}$$

This is equivalent to shifting the edge phase factors ψ_0, ψ_d by $\pi/4$. For the simplicity of presentation, this chapter assumes that the imaginary part is relevant. Furthermore, we denote it as $g(x,\Psi)$,

$$g(x,\Psi) := \mathrm{Im}[\langle 0|U(x,\Psi)|0\rangle]. \tag{4.5}$$

More details of the conventions of phase factors and their equivalences are given in Section 4.2. Given any symmetric phase factors $\Psi := (\psi_0, \psi_1, \ldots, \psi_d)$ of length $d+1$, we define the *reduced phase factors* Φ as

$$\Phi = (\phi_0, \phi_1, \ldots, \phi_{\widetilde{d}-1}) := \begin{cases} (\frac{1}{2}\psi_{\widetilde{d}-1}, \psi_{\widetilde{d}}, \cdots, \psi_d), & d \text{ is even,} \\ (\psi_{\widetilde{d}}, \psi_{\widetilde{d}-1}, \cdots, \psi_d), & d \text{ is odd,} \end{cases} \tag{4.6}$$

where $\widetilde{d} := \lceil \frac{d+1}{2} \rceil$. For the sake of simplicity, we do not explicitly distinguish the full set of phase factors and the reduced phase factors when they are used as the

argument of some function. For example, $U(x,\Phi)$ and $g(x,\Phi)$ are assumed to represent evaluations with respect to the full set of phase factors, $U(x,\Psi)$ and $g(x,\Psi)$. To be embedded in a SU(2) matrix, it is naturally required that $g(x,\Psi)\leq 1$ for any $x\in[-1,1]$. Hence, the target function $f:\mathbb{R}\to\mathbb{R}$ is also normalized so that its norm is bounded:

$$\|f\|_\infty = \max_{x\in[-1,1]} |f(x)| \leq 1. \tag{4.7}$$

Theorem 4.1.1 implies that if the target polynomial f of definite parity can be represented as symmetric QSP, it admits a Chebyshev polynomial expansion:

$$f(x) = \begin{cases} \sum_{j=0}^{\widetilde{d}-1} c_j T_{2j}(x), & f \text{ is even}, \\ \sum_{j=0}^{\widetilde{d}-1} c_j T_{2j+1}(x), & f \text{ is odd}. \end{cases} \tag{4.8}$$

Let $\mathcal{F}$ denote the linear mapping from a target polynomial to its Chebyshev-coefficient vector

$$c := (c_0, c_1, \cdots, c_{\widetilde{d}-1}) \in \mathbb{R}^{\widetilde{d}}. \tag{4.9}$$

This induces the mapping from the set of reduced phase factors to the Chebyshev-coefficient vector of $g(x,\Phi)$,

$$F:\mathbb{R}^{\widetilde{d}}\to\mathbb{R}^{\widetilde{d}}, \quad F(\Phi) := \mathcal{F}(g(x,\Phi)). \tag{4.10}$$

4.2 Conventions of phase factors and equivalences

Depending on the specific context in which QSP is employed, there are various phase-factor conventions that facilitate a more straightforward presentation of the QSP ansatz. While these conventions aim to achieve the same ultimate goal, the numerical representation of QSP phase factors can differ significantly. For instance, the equivalence shown in Eq. (4.4) illustrates the numerical variation when setting either the real or the imaginary component of $\langle 0|U(x,\Phi|0\rangle$ as the target function. In this section, we will comprehensively review several different phase-factor conventions. This detailed examination aims to clarify the methodologies in integrated applications of QSP-based methods.

Choosing between real or imaginary components as the target

Let us revisit the equivalence relation presented in Eq. (4.4):

$$\mathrm{Re}[\langle 0|U(x,\Psi)|0\rangle] = \mathrm{Im}\left[\langle 0|e^{\mathrm{i}\frac{\pi}{4}Z}U(x,\Psi)e^{\mathrm{i}\frac{\pi}{4}Z}|0\rangle\right]. \tag{4.11}$$

This relation effectively corresponds to a $\pi/4$ shift in the edge phase factors ψ_0, ψ_d. This relation can be shown by direct computation. Demonstrated through direct computation, and utilizing the fact that $e^{\mathrm{i}\frac{\pi}{4}Z}|0\rangle = e^{\mathrm{i}\frac{\pi}{4}}|0\rangle$, we find:

$$\langle 0|e^{\mathrm{i}\frac{\pi}{4}Z}U(x,\Psi)e^{\mathrm{i}\frac{\pi}{4}Z}|0\rangle = e^{\mathrm{i}\frac{\pi}{2}}\langle 0|U(x,\Psi)|0\rangle = \mathrm{i}P(x) \tag{4.12}$$

where $P(x) := \langle 0|U(x,\Phi)|0\rangle$ is a complex-valued polynomial. Consequently, the equivalence relation follows $\mathrm{Re}[P(x)] = \mathrm{Im}[\mathrm{i}P(x)]$.

It is important to note that the initial guess defined in Eq. (2.32) pertains to cases where the target is set to the real component. In contrast, when targeting the imaginary component, this equivalence implies an all-zero vector initial guess, $\Phi^0 = (0, 0, \cdots, 0)$, as utilized in this chapter.

Historically, the real component has been the default target in the literature. Notably, the real-target convention simplified the presentation in the context of QETU as seen in Ref. [51]. However, the imaginary-target convention offers more concise analysis in understanding the structural properties of phase factors. Therefore, we adopt the imaginary-target convention in Chapters 4 and 5, while the real-target convention is used in the remaining chapters.

Selection among lifted phase factors

Recall the transformation rule between the set of phase factors $(\phi_j)_{j=0}^d$ derived SU(2) model and those $(\varphi_j)_{j=0}^d$ used in high-dimensional quantum circuit applications, as presented in Eqs. (2.14) and (2.15). The alternating signs in this rule reflect the interleaved application of the block encoding and its inverse in the quantum circuit. Additionally, literature [53, 52, 51] often employs another set of lifted phase factors $(\widetilde{\varphi}_j)_{j=0}^d$:

$$\widetilde{\varphi}_j = \begin{cases} \phi_0 + \frac{\pi}{4} & \text{, when } j = 0, \\ \phi_j + \frac{\pi}{2} & \text{, when } 1 \le j \le d-1, \\ \phi_d + \frac{\pi}{4} & \text{, when } j = d. \end{cases} \tag{4.13}$$

Notably, this choice does not differentiate between even and odd cases, unlike the rules in Eqs. (2.14) and (2.15). Direct computation reveals that for even degree

d, using Eq. (4.13) in high-dimensional quantum circuits directly implements the desired task. However, for odd d, an additional Z gate after the first Hadamard gate on the ancilla qubit is required to maintain consistency with the results. This small difference also agrees with literature, e.g. Refs. [53, 52, 51].

This verification can be performed by analyzing the induced SU(2) phase factors through the application of the inversion rule in Eqs. (2.14) and (2.15):

$$\text{SU(2) phase factors } (\phi_j)_{j=0}^{d} \xrightarrow{\text{Eq. (4.13)}} \text{lifted } (\widetilde{\varphi}_j)_{j=0}^{d} \xrightarrow{\text{Eqs. (2.14) and (2.15)}} \text{induced } (\widetilde{\phi}_j)_{j=0}^{d}.$$

The consistency of the induced phase factors with the original set is assured if the first transformation follows Eqs. (2.14) and (2.15). Let us consider the polynomial pair defined by Theorem 2.3.3 using the original set of phase factors as (P, Q) and the pair defined by the induced phase factors as $(\widetilde{P}, \widetilde{Q})$.

For the case where $d = 2k$, the transformation yields the following induced phase factors:

$$\widetilde{\phi}_j = \begin{cases} \phi_j + \frac{\pi}{2} & \text{, when } j = 0 \text{ or } 2k, \\ \phi_j + \pi & \text{, when } j = 2, 4, \cdots, 2k-2, \\ \phi_j & \text{, when } j = 1, 3, \cdots, 2k-1. \end{cases} \tag{4.14}$$

Analyzing the impact on the QSP unitary matrix and polynomials:

$$U(x, \widetilde{\Phi}) = (-1)^k Z U(x, \Phi) Z \Rightarrow \widetilde{P} = (-1)^k P, \widetilde{Q} = (-1)^{k-1} Q. \tag{4.15}$$

Consequently, the QSP circuit using these induced phase factors implements the polynomial transformation that aligns with the original set of phase factors, up to a negligible global phase factor $(-1)^k$.

For the case where $d = 2k + 1$, we have:

$$\widetilde{\phi}_j = \begin{cases} \phi_j + \frac{\pi}{2} & \text{, when } j = 2k+1, \\ \phi_j + \pi & \text{, when } j = 1, 3, \cdots, 2k-1, \\ \phi_j & \text{, when } j = 2, 4, \cdots, 2k. \end{cases} \tag{4.16}$$

The corresponding effect on the QSP unitary matrix and polynomials is:

$$U(x, \widetilde{\Phi}) = \mathrm{i}(-1)^k Z U(x, \Phi) Z \Rightarrow \widetilde{P} = \mathrm{i}(-1)^k P, \widetilde{Q} = \mathrm{i}(-1)^{k-1} Q. \tag{4.17}$$

In this scenario, the QSP polynomial is phased with i , interchanging the real and imaginary components of the complex polynomial P. As a result, directly applying the QSP circuit as depicted in Fig. 2.3 (c) would implement P_{Im} instead. To correct this, an additional Z gate can be inserted after the first Hadamard gate acting on the ancilla qubit. This modification ensures that the quantum circuit implements P_{Re} , up to a negligible global phase, as the relation $(\mathrm{i}P - (\mathrm{i}P)^*)/2 = \mathrm{i}P_{\mathrm{Re}}$holds. The negation in the second term arises due to the added Z gate.

4.3 Optimization-based method for finding phase factors

The QSP problem can also be solved using numerical optimization

$$\begin{aligned} \Phi^* &= \arg\min_{\Phi} \|F(\Phi) - c\|_2^2 = \arg\min_{\Phi} \sum_{j=1}^{\widetilde{d}} |g(x_j, \Phi) - f(x_j)|^2 , \\ \text{cost function: } L(\Phi) &:= \frac{1}{\widetilde{d}} \sum_{j=1}^{\widetilde{d}} |g(x_j, \Phi) - f(x_j)|^2 . \end{aligned} \tag{4.18}$$

Here, $x_j = \cos\left(\frac{(2j-1)\pi}{4\widetilde{d}+1}\right)$ is the j-th node of the Chebyshev polynomial $T_{2\widetilde{d}}(x)$. The equality in the optimization problem follows the discrete orthogonality on Chebyshev nodes. In Ref. [53], the optimization problem is first solved using the LBFGS method and the running complexity is numerically studied. The authors also propose the use of an initial guess $\Phi^0 = (0, 0, \cdots, 0)$, from which the convergence of the LBFGS method is numerically observed to be fast and stable. We remark that the initial guess in the original paper is not identical to this form, due to the difference in the definition. The original paper considers the real part of $\langle 0|U(x, \Psi)|0\rangle$ to encode the polynomial of interest, whereas this paper considers the imaginary part, with equivalence established through Eq. (4.4). The choice of the initial guess is justified in the theoretical analysis in Ref. [153], which is credited to a class of optima called the maximal solution. In Ref. [153], the authors analyze the energy landscape of the optimization problem and conclude that when the target function is scaled as $\|f\|_\infty = \mathcal{O}(1/d)$, the optimization-based algorithm converges locally at $\mathcal{O}(d^2 \log(1/\epsilon))$ computational cost.

Before delving into the presentation of the algorithm, we first provide a theoretical result concerning the derivation of the Hessian matrix at the initial guess Φ^0. This theorem reveals that the initial Hessian matrix assumes a notably simple form. It is important to note that the theorem presented here is a modified version of [153, Theorem 24]. The modification stems from the differences in the definition of reduced phase factors, as in Eq. (4.6).

Theorem 4.3.1. *At the initial point* $\Phi^0 := (0, 0, \cdots, 0) \in \mathbb{R}^{\widetilde{d}}$*, the Hessian matrix of the optimization problem Eq.* (4.18) *is*

$$\mathrm{Hess}(\Phi^0) = 4I. \tag{4.19}$$

Proof. The Hessian matrix of the objective function defined in Eq. (4.18) is

$$\mathrm{Hess}(\Phi) = \frac{2}{\widetilde{d}}\sum_{k=1}^{\widetilde{d}} g_i(x_k,\Phi)g_j(x_k,\Phi) - \frac{2}{\widetilde{d}}\sum_{k=1}^{\widetilde{d}}[g(x_k,\Phi)-f(x_k)]g_{ij}(x_k,\Phi). \tag{4.20}$$

Note that

$$U(x,\Phi^0) = W^d(x).$$

The integer power of the matrix $W(x) := e^{\mathrm{i}\arccos(x)X}$ exactly generates the Chebyshev polynomial of the first kind as its upper-left matrix element. Furthermore, its upper-left is real. The claim follows the property of the exponential map of Pauli matrices $W^d(x) = e^{\mathrm{i}d\arccos(x)X} = \cos(d\arccos(x)) + \mathrm{i}\sin(d\arccos(x))X$. Then, noting that $\langle 0|X|0\rangle = 0$, it yields $\langle 0|W^d(x)|0\rangle = \cos(d\arccos(x)) = T_d(x)$.

To proceed, let us consider the monomial

$$M^{d_1,d_2,d_3}(x) := W^{d_1}(x)(\mathrm{i}Z)W^{d_2}(x)(\mathrm{i}Z)W^{d_3}(x)$$

for any $d_1, d_2, d_3 \geq 0$. We will show that the upper-left element $\langle 0|M^{d_1,d_2,d_3}(x)|0\rangle$ is real. Following the anticommutation relation $ZXZ = -X$, a useful equality can be derived $ZW(x)Z = e^{\mathrm{i}\arccos(x)ZXZ} = e^{-\mathrm{i}\arccos(x)X} = W^{-1}(x)$. Consequentially, by moving Z gates to the same site for cancellation, it can be shown that $\langle 0|M^{d_1,d_2,d_3}(x)|0\rangle = -T_{|d_1+d_3-d_2|}(x)$ is real.

Note that taking second-order derivative on the unitary defined in Theorem 4.1.1 at Φ^0 results in a integer-coefficient superposition of monomials of the defined form. To illustrate, given an odd integer $d = 2\widetilde{d}-1$ and $0 \leq i < j < \widetilde{d}$, one can verify that

$$\frac{\partial^2}{\partial\phi_i\partial\phi_j}U(x,\Phi)\bigg|_{\Phi^0} = M^{i,j-i,d-j}(x) + M^{i,d-j-i,j}(x) + M^{d-j,j-i,i}(x) + M^{j,d-j-i,i}(x).$$

Then, by taking the imaginary component, the second-order derivative g_{ij} is vanishing at Φ^0. Thus, the second term in the right-hand side of Equation (4.20) vanishes since $g_{ij}(x,\Phi^0) = 0$.

When $d = 2\widetilde{d}-1$ is odd, $g_i(x,\Phi^0) = 2\mathrm{Im}\left[\langle 0|\mathrm{i}W^i(x)ZW^{d-i}(x)|0\rangle\right] = 2T_{d-2i}(x)$ for $i = 0,\cdots,\widetilde{d}-1$ and $g_{\widetilde{d}-1}(x,\Phi^0) = -1$. According to the discrete orthogonality of Chebyshev nodes, we have

$$\sum_{k=1}^{\widetilde{d}} g_i(x_k,\Phi^0)g_j(x_k,\Phi^0) = 2\widetilde{d}\delta_{ij}.$$

We have similarly when d is even.

Following Eq. (4.20), the Hessian matrix at Φ^0 takes the form in the theorem which completes the proof. □

In numerical optimization, the Broyden–Fletcher–Goldfarb–Shanno (BFGS) algorithm is a quasi-Newton method for solving unconstrained optimization problems [139, Chapter 5]. The BFGS method stores a dense $n \times n$ matrix to approximate the inverse of Hessian matrix. It updates this approximation by performing a rank two update using gradient information along its trajectory. Limited-memory BFGS (L-BFGS) approximates the BFGS method by using a limited amount of computer memory [139, Chapter 5]. In particular, it represents the inverse of Hessian matrix implicitly by only a few vectors. For a comprehensive understanding, we have outlined the procedure of the L-BFGS method in Algorithm 4.3.1. Utilizing this method as a key subroutine, the optimization-based approach for solving QSP phase factors is concisely summarized in Algorithm 4.3.2.

Algorithm 4.3.1 Function: $\phi = \text{L-BFGS}(\phi^0, L, m, B^0)$

Input: Initial point ϕ^0, objective function $L(\phi)$, a non-negative integer m and initial approximation of inverse Hessian B^0.
Set $t = 0$
while stopping criteria does not meet **do**
  Compute $g_t = \nabla L(\phi^t)$, set $q = g_t$
  for $i = t-1, \ldots, t-m$ **do**
    Set $\alpha_i = \rho_i s_i^\top q$
    $q = q - \alpha_i y_i$
  end for
  $r = B_0 q$
  for $i = t-m, \ldots, t-1$ **do**
    $\beta = \rho_i y_i^\top r$
    $r = r + s_i(\alpha_i - \beta)$
  end for
  Set search direction $d_t = -r$.
  Find a step size γ_t using backtracking line search.
  Set

$$\phi^{t+1} = \phi^t + \gamma_t d_t, s_k = \phi^{t+1} - \phi^t,$$
$$y_t = g_{t+1} - g_t, \rho_t = \frac{1}{s_t^\top y_t}.$$

  Set $t = t + 1$.
end while
Return: ϕ^t

Algorithm 4.3.2 Function: $\Phi = \text{QSPBFGS}(\Phi^0, f, \epsilon)$

Input: An initial vector Φ^0, a real polynomial f of degree d and error tolerance ϵ.

1: Choose $\tilde{d} = \lceil \frac{d+1}{2} \rceil$ points $x_j = \cos(\frac{(2j-1)\pi}{4\tilde{d}})$ as the positive roots of Chebyshev polynomial $T_{2\tilde{d}}$.
2: Construct objective function $L(\Phi)$ using Eq. (4.18).
3: Choose the initial approximation of inverse Hessian B_0 using Eq. (4.19).
4: Set $t = 0$
5: **while** $L(\Phi) > \epsilon$ **do**
6: Obtain Φ^{t+1} by updating Φ^t via L-BFGS algorithm.
7: Set $t = t + 1$.
8: **end while**
Return: Φ^t

4.4 Energy landscape and convergence analysis of optimization-based algorithm

In this section, we summarize a theory to characterize the energy landscape of the cost function $L(\Phi)$, and to explain this unusual behavior of numerical optimization [153]. To our knowledge, this leads to the first provable algorithm for solving the QSP problem without referring to extended precision arithmetic operations [73]. We note that the theory presented here is a modified version of Ref. [153] due to the differences in the definition of reduced phase factors, as outlined in Eq. (4.6).

Our first main result is the existence and uniqueness of symmetric phase factors for a class of polynomial matrices in SU(2). The theorem, as stated in Theorem 4.1.1, implies the existence of a bijection between the global minimizers of Eq. (4.18) and all possible pairs of $(P(x), Q(x))$ satisfying the assumption in Theorem 4.1.1 and $\mathrm{Im}[P](x) = f(x)$. The conditions (1), (2) for the target polynomial f are compatible with the first two requirements in Theorem 4.1.1. The condition (3) on the maxnorm, *i.e.*, $\|f\|_\infty < 1$ is compatible with the normalization condition, which is itself a natural condition due to the unitarity of $U(x, \Phi)$.

To simplify the discussion, we introduce the following definition of admissible pair of polynomials associated with a target polynomial. Here $P_{\mathrm{Re}} := \mathrm{Re}[P], P_{\mathrm{Im}} := \mathrm{Im}[P]$.

Definition 4.4.1 (Admissible pair of polynomials)**.** *Let $f \in \mathbb{R}[x]$ be a target polynomial satisfying (1) $\deg(f) = d$, (2) the parity of f is $(d \bmod 2)$, (3) $\|f\|_\infty < 1$. Then (P, Q) is an admissible pair of polynomials associated with f if the conditions (1)-(3) in Theorem 4.1.1 are satisfied, together with*

(4) The leading coefficient of Q is positive,

(5) $P_{\mathrm{Im}}(x) = f(x)$.

The polynomials $P_{\mathrm{Re}}, Q \in \mathbb{R}[x]$ are also called complementary polynomials to f.

Comparing Theorem 4.1.1 and Definition 4.4.1, the main modification is that the leading coefficient of the complementary polynomial Q is always restricted to be positive. This allows us to unify the discussion of odd and even values of d. The bijection relation can then be concisely formulated as follows.

Corollary 4.4.2 (Bijection between global minima and admissible pairs)**.** *If d is odd, there is a bijection between the global minima of Eq.* (4.18) *and all admissible pairs (P, Q).*

If d is even, there is a bijection between the global minima of Eq. (4.18) *and all pairs of polynomials $(P, \pm Q)$, where (P, Q) is an admissible pair.*

The proof of Theorem 4.1.1 is constructive. So given an admissible pair, we have an algorithm to evaluate the symmetric phase factor. Theorem 4.4.3 explicitly constructs all the admissible pairs. Together with Theorem 4.1.1, we have a complete description of all global minima of the cost function in Eq. (4.18).

Theorem 4.4.3 (Construction of admissible pairs)**.** *Given a target polynomial $f(x)$, all admissible pairs (P, Q) must take the following form,*

$$\begin{aligned} P_{\mathrm{Re}}(x) &= \sqrt{\alpha}\frac{e\left(x+\mathrm{i}\sqrt{1-x^2}\right)+e\left(x-\mathrm{i}\sqrt{1-x^2}\right)}{2}, \\ Q(x) &= \sqrt{\alpha}\frac{e\left(x+\mathrm{i}\sqrt{1-x^2}\right)-e\left(x-\mathrm{i}\sqrt{1-x^2}\right)}{2\mathrm{i}\sqrt{1-x^2}}, \end{aligned} \tag{4.21}$$

Here, $e(z) := z^{-d}\prod_{i=1}^{2d}(z - r_i)$, where $\{r_i\}_{i=1}^{2d}$ are roots of function $\mathfrak{F}(z) := 1 - \left[f\left(\frac{z+z^{-1}}{2}\right)\right]^2$. And $\alpha \in \mathbb{C}$ satisfies $\mathfrak{F}(z) = \alpha e(z)e(z^{-1})$.

Note that $\{r_i\}_{i=1}^{2d}$ are only part of roots of the *Laurent polynomial* $\mathfrak{F}(z)$, which has $4d$ roots in total. By properly choosing the roots (see a more detailed description in [153, Remark 17]), we can construct all the admissible pairs. In particular, Theorem 4.4.3 shows that the number of global minima is finite, but grows combinatorially with respect to d. The finiteness of the number of global minima is also a direct consequence of the compactness of the domain D_d.

Unfortunately, the procedure described above for finding phase factors is numerically unstable. It requires extended precision arithmetic operations and is therefore

very expensive when d is large (see detailed discussions in [153, Remark 16]). This is noticeably different from solving the optimization problem in Eq. (4.18), which is numerically stable and can be readily performed using standard double precision arithmetic operations.

Among the myriad of global minima, Theorem 4.4.3 allows us to identify a special global minimum, which is obtained by choosing $\{r_i\}_{i=1}^{2d}$ to be the roots of $\mathfrak{F}(z)$ within the unit disc. The unique symmetric phase factor associated with this admissible pair is referred to as the *maximal solution* (the reason for the naming is technical and is explained in [153, Section 4.2]).

The maximal solution enjoys many desirable properties. For any target polynomial f with $\|f\|_\infty \leq \frac{1}{2}$, the maximal solution lies in the neighborhood of Φ^0.

Theorem 4.4.4 (Distance between the maximal solution and Φ^0)**.** *Let Φ^* be the maximal solution for the target function $f(x)$ in terms of reduced phase factors. If $\|f(x)\|_\infty \leq \frac{1}{2}$, then*

$$\left\|\Phi^* - \Phi^0\right\|_2 \leq \frac{\pi}{\sqrt{3}} \|f(x)\|_\infty . \tag{4.22}$$

In particular, if the target polynomial is $f = 0$, then the maximal solution is the initial guess Φ^0. We then prove that when $\|f\|_\infty$ is sufficiently small, the maximal solution belongs to a neighborhood of Φ^0, on which the cost function $L(\Phi)$ is strongly convex, *i.e.*, the Hessian matrix denoted by $\mathrm{Hess}(\Phi)$ is positive definite.

Theorem 4.4.5 (Local strong convexity)**.** *If the target polynomial satisfies $\|f\|_\infty \leq \frac{\sqrt{3}}{20\pi\widetilde{d}}$, for any symmetric phase factors of length $d+1$ and with reduced phase factors Φ satisfying $\|\Phi - \Phi^0\|_2 \leq \frac{1}{20\widetilde{d}}$, the following estimate holds:*

$$\frac{1}{4} \leq \lambda_{\min}\left(\mathrm{Hess}(\Phi)\right) \leq \lambda_{\max}\left(\mathrm{Hess}(\Phi)\right) \leq \frac{25}{4}. \tag{4.23}$$

The local strong convexity result in Theorem 4.4.5 immediately implies that when $\|f\|_\infty$ is sufficiently small, standard optimization algorithms, such as the projected gradient method [21, 83], can converge in the neighborhood of Φ^0, without being trapped by any local minima.

Corollary 4.4.6 (Convergence of projected gradient method)**.** *If the target polynomial satisfies $\|f\|_\infty \leq \frac{\sqrt{3}}{20\pi\widetilde{d}}$, starting from Φ^0, the projected gradient method with step size $t = \frac{1}{L}$ converges exponentially to the maximal solution Φ^*, i.e., at the ℓ-th iteration*

$$\left\|\Phi^\ell - \Phi^*\right\|_2^2 \leq e^{-\frac{\sigma}{L}\ell} \left\|\Phi^0 - \Phi^*\right\|_2^2. \tag{4.24}$$

Here $\sigma = \frac{1}{4}$ and $L = \frac{25}{4}$.

Therefore when $\|f\|_\infty \leq \frac{\sqrt{3}}{20\pi\widetilde{d}}$, in order to reach precision ϵ, the projected gradient method can be terminated after $\mathcal{O}(\log(1/\epsilon))$ steps independent of the details of f. Since each iteration is numerically stable, the maximal solution can be readily obtained using standard double precision arithmetic operations in practice.

In the remainder of this section, additional numerical results are presented to illustrate key features of the energy landscape associated with the optimization problem for symmetric phase factors, as detailed in Eq. (4.18).

Hessian matrix without symmetry constraint is singular at global optima

When the symmetry constraint is not imposed, the Hessian matrix can be derived from Eq. (4.20). Note that the Jacobian matrix $A \in \mathbb{R}^{\widetilde{d}\times d}$ does not have full column rank because $d > \widetilde{d}$. Then, following that $\ker(A^\top A) = \ker(A)$, we have $\operatorname{rank}(A^\top A) = \operatorname{rank}(A) \leq \widetilde{d} < d$. Note that at the global optima, the Hessian matrix is precisely $\mathrm{Hess}(\Phi^*) = (A^*)^\top A^*$ due to the vanishing residual term $R = 0$. Here, A^* refers to the Jacobian matrix at the optima. Thus, the Hessian matrix at optima is singular. Because the residual term R is small near the optima by continuity, the Hessian matrix is almost singular in a neighborhood of the optima. This behavior is demonstrated by the numerical result displayed in Fig. 4.1 (a). On the other hand, when the symmetry constraint is imposed, Fig. 4.1 (b) shows that the Hessian matrix is positive definite in a neighborhood of the maximal solution and it agrees our theoretical results about the local strong convexity of the Hessian matrix.

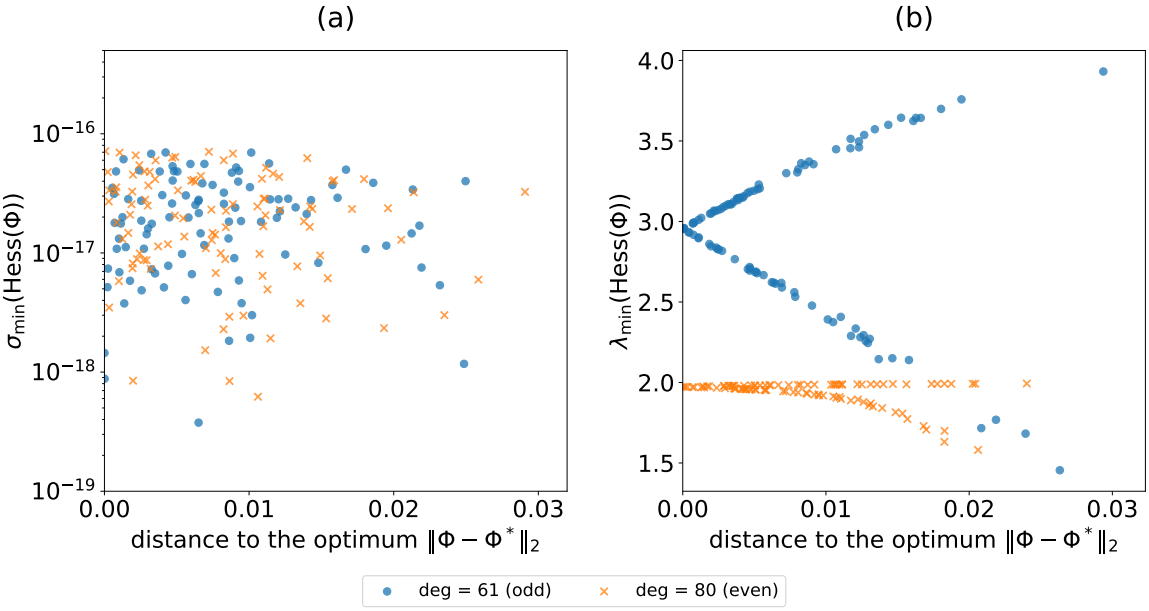


Figure 4.1: The smallest singular value ($\sigma_{\min}$) or eigenvalue ($\lambda_{\min}$) of the Hessian matrix evaluated at 100 randomly sampled Φ near the optimizer Φ^* of a given polynomial. Blue dots correspond to an odd polynomial of degree 61 and orange crosses correspond to an even polynomial of degree 80. (a) The symmetry constraint is not imposed. The plotted data concentrates around the machine precision $\sim 10^{-17}$, which implies a large basin near the optimizer on which the Hessian matrix is singular if the symmetry constraint is not imposed. (b) The symmetry constraint is imposed. The numerical results show that the Hessian matrix is well conditioned near the optimizer, and it is positive definite which agrees our result of local strong convexity.

Existence of local minima

To visualize the full energy landscape of the optimization problem, we first consider the simplest scenario so that only two phase factors suffice to solve the optimization problem exactly. We choose two target polynomials $f(x) = x^2 - \frac{1}{2}$ and $f(x) = \frac{1}{\sqrt{3}}x^3 - \frac{2}{\sqrt{3}}x$. The energy landscape on the irreducible domain is displayed in Fig. 4.2. The global minima derived from the proposed method are annotated in the domain. In this special case, there is not any local minimum in the landscape. However, this is a rare exception rather than the norm. Existence of local minima can be observed by slightly increasing the degree of the target polynomial

$f(x) = 0.2103T_4(x) + 0.1231T_2(x) + 0.1666$. The local minimizer is numerically searched by randomly initiating the stochastic gradient descent algorithm to solve Eq. (4.18). The value of the objective function at Φ^{loc} is 0.0218. The Hessian matrix has eigenvalues $0.1359, 4.5815, 7.7510$, which indicates that Φ^{loc} is indeed a local minimum. To visualize the landscape, we plot a two-dimensional section of the optimization landscape spanned by the affine plane spanned by the eigenvectors of the least two eigenvalues of the Hessian matrix (Fig. 4.2).

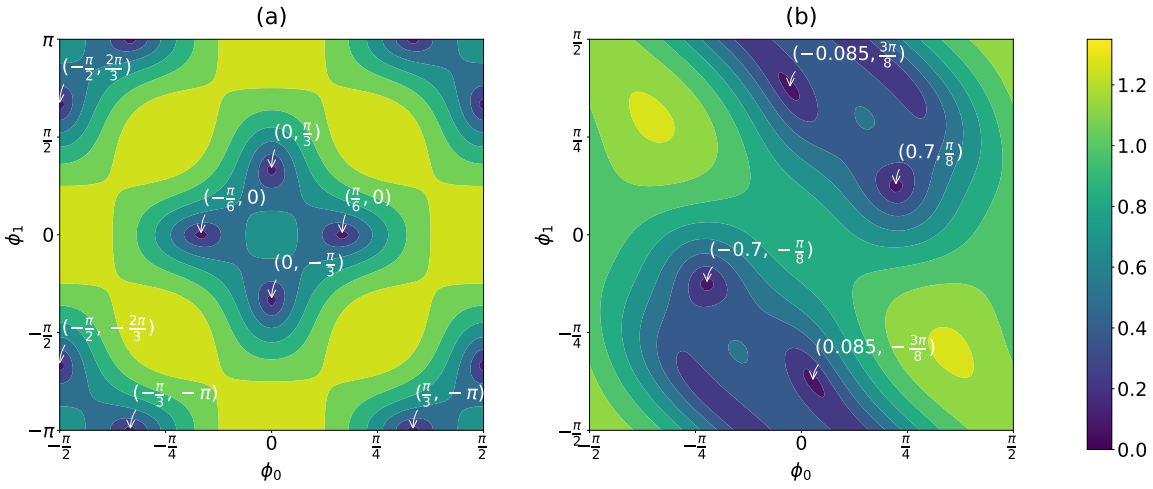


Figure 4.2: The optimization landscape of the modified cost function $F(\widetilde{\Phi})^{1/3}$ on the irreducible domain D_d. Here, the cube root is taken to signify the structure of the landscape near optima. The annotated values are all optima exactly computed from the proposed method. (a) The target function is set to $f(x) = x^2 - \frac{1}{2}$ which yields 8 inequivalent optima. (b) The target function is set to $f(x) = \frac{1}{\sqrt{3}}x^3 - \frac{2}{\sqrt{3}}x$ which yields 4 inequivalent optima.

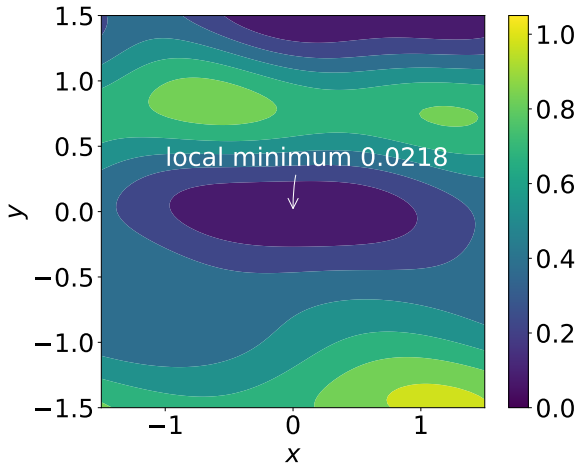


Figure 4.3: A two-dimensional section of the optimization landscape $F(\widetilde{\Phi}^{\mathrm{loc}} + xv_1 + yv_2)$, where Φ^{loc} is a local minimum of a given degree-5 target polynomial, and v_1, v_2 are unit eigenvectors corresponding to the least two eigenvalues of the Hessian matrix evaluated at Φ^{loc}. The objective value of the local minimum is annotated on the figure.

Numerical behavior near the maximal solution

Here we demonstrate that among all global minima, the maximal solution is particularly desirable from the perspective of numerical optimization. Given an arbitrarily chosen polynomial $f(x) = \frac{1}{4}T_6(x) + \frac{5}{4}T_4(x) + \frac{1}{8}T_2(x) - T_0(x)$, we consider a sequence of scaled polynomials $f^{(k)}(x) := 10^{-k} f(x)$, so that $\lim_{k\to\infty} f^{(k)}(x) = 0$. All exact solutions to the optimization problem associated to $f^{(k)}(x)$ can be constructed explicitly via [153, Remark 17]. This procedure yields several sequences of phase factors, which converge to different limiting points as $k \to \infty$. For the specific degree-6 polynomial above, there are four distinct limits which is referred to as Φ_0^{class},

- class 1: $\Phi_0^1 = (0, 0, 0, 0, 0, 0, 0)$,
- class 2: $\Phi_0^2 = \left(0, 0, \frac{\pi}{4}, -\frac{\pi}{2}, \frac{\pi}{4}, 0, 0\right)$,
- class 3: $\Phi_0^3 = \left(0, 0, -\frac{\pi}{4}, \frac{\pi}{2}, -\frac{\pi}{4}, 0, 0\right)$,
- class 4: $\Phi_0^4 = \left(0, \frac{\pi}{4}, 0, -\frac{\pi}{2}, 0, \frac{\pi}{4}, 0\right)$.

Specifically, the first class is associated with the maximal solution. The sequence of the set of phase factors corresponding to the scaled polynomial $f^{(k)}(x)$ and a

given class of construction is referred to as $\Phi_*^{\text{class}}(k)$. Fig. 4.4 shows that the class associated with the maximal solution distinguishes from other classes, in the sense that the convergence rate of $\Phi_*^1(k)$ towards Φ_0^1 is much faster. This suggests that the convergence basin near Φ_0^1 is much flatter, which justifies the choice of using Φ_0^1 as the initial guess of the optimization.

To further test the performance of numerical optimization associated with different initial guesses, we use the limit point Φ_0^{class} as the initial guess and find the optimal phase factors $\Phi_*^{\text{class}}(k)$ generating the scaled polynomial $f^{(k)}(x)$ by running QSPPACK [53]. The trajectory of the optimization process is displayed in Fig. 4.5. It shows that the optimization starting from Φ_0^1 also has the fastest convergence rate.

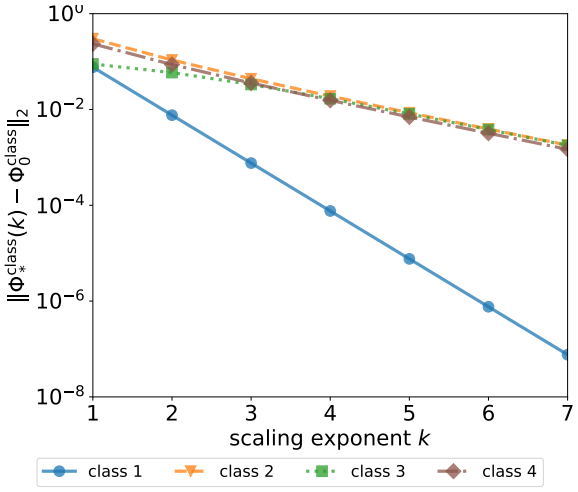


Figure 4.4: The convergence to the limit point of different class as the target polynomial being scaled down. Given a polynomial $f(x)$, the optimal set of phase factors $\Phi_*^{\text{class}}(k)$ parameterizes the scaled polynomial $f^{(k)}(x) := 10^{-k} f(x)$. Here, Φ_0^{class} is the limit point of the corresponding class.

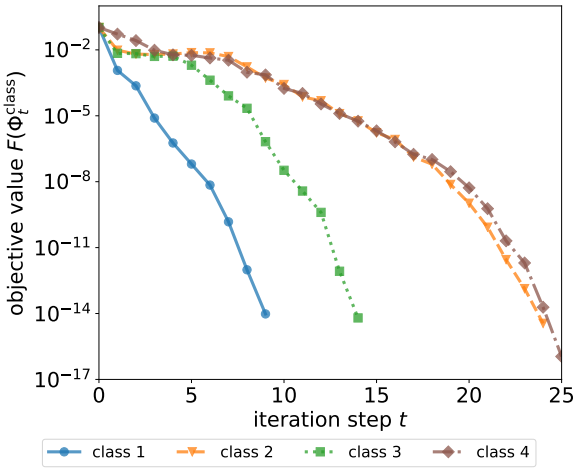


Figure 4.5: The objective value in each iteration step by running QSPPACK. Initiating the optimization from the limit point of a class Φ_0^{class}, Φ_t^{class} is the set of phase factors in the t-th optimization step. Different symbols correspond to the optimization trajectories starting from distinct classes of initial guess.

4.5 Fixed-point iteration method for finding phase factors

As outlined in Eq. (4.1), the QSP problem can be reformulated as solving the nonlinear system $F(\Phi) = c$, where F represents the mapping from reduced phase factors to the Chebyshev-coefficient vector of $g(x, \Phi)$, and c is the Chebyshev-coefficient vector of the target polynomial. Inspired by the analysis of the Jacobian map DF, in Ref. [54], a fixed-point iteration method (FPI) is proposed for solving Eq. (4.1):

$$\Phi^0 = 0 \in \mathbb{R}^{\widetilde{d}}, \quad \Phi^{t+1} = \Phi^t - \frac{1}{2}\left(F\left(\Phi^t\right) - c\right). \tag{4.25}$$

Notably, the initial guess of the FPI method coincides with that used in the LBFGS method. This algorithm can be viewed as finding the fixed point of the mapping $G(x) := x - \frac{1}{2}(F(x) - c)$ by means of a fixed point iteration $\Phi^{t+1} = G(\Phi^t)$. This can also be viewed as an inexact Newton algorithm [115, Chapter 11], as the inverse of the Jacobian matrix at $0 \in \ell^1$ satisfies $[DF(0)]^{-1} = \frac{1}{2}\mathbf{I}$.

Algorithm 4.5.1 Fixed-point iteration algorithm for solving reduced phase factors

Input: Chebyshev-coefficient vector c of a target polynomial, and stopping criteria.

Initiate $\Phi^0 = 0$, $t = 0$;
while stopping criterion is not satisfied **do**
    Update $\Phi^{t+1} = \Phi^t - \frac{1}{2}\left(F\left(\Phi^t\right) - c\right)$;
    Set $t = t + 1$;
end while

Output: Reduced phase factors Φ^{t+1}.

For a given target function, the Chebyshev-coefficient vector can be efficiently evaluated using the fast Fourier transform (FFT). The FPI algorithm in Algorithm 4.5.1 is the simplest algorithm thus far to evaluate phase factors. This algorithm provably converges when $\|c\|_1$ is upper bounded by a constant.

Theorem 4.5.1 (Convergence of the fixed-point iteration algorithm)**.** *There exists a universal constant $\widetilde{r}_c \approx 0.861$ so that when $\|c\|_1 \leq \widetilde{r}_c$,*

(i) Algorithm 4.5.1 converges Q-linearly to $\Phi^\star = \overline{F}^{-1}(c)$ where $\overline{F}$ is an extension of F [54]. Specifically, there exists a constant C and the error satisfies

$$\left\|\Phi^t - \Phi^\star\right\|_1 \leq C\widetilde{\gamma}^t, \quad \widetilde{\gamma} \approx 0.8189, \quad t \geq 1. \tag{4.26}$$

(ii) The overall time complexity is $\mathcal{O}(d^2 \log \frac{1}{\epsilon})$, where d is the degree of target polynomial and ϵ is the desired precision.

A more accurate characterization about the region where Algorithm 4.5.1 converges and the convergence rate is presented in Ref. [54, Section 5.1].

Theorem 4.5.1 and the analysis in Ref. [54] demonstrate that the FPI method exhibits linear convergence to the exact solution when $\|c\|_1 \leq 0.861$. This result is based on the observation that the update rule acts as a contraction mapping in a neighborhood of the initial guess $\Phi^0 = 0$. In [54, Section 6], numerical experiments demonstrate that Algorithm 4.5.1 is an efficient algorithm, and we observe that its convergence radius can be much larger than the theoretical prediction. However, this property does not hold universally across the entire domain. The analysis in Ref. [54] reveals that the contraction property is valid when the Chebyshev coefficient vector of the target function lies within an ℓ^1 ball centered at the origin. In cases where the target function contains significant "high-frequency" components, the increasingly large Chebyshev coefficient vector may hinder the contraction of the update rule in Eq. (4.25). This situation commonly occurs in various applications; for instance, problematic convergence issues can arise when dealing with functions $\sin(\tau x)$ and $\cos(\tau x)$, where $\tau \gg 1$ is large.

4.6 Numerical difficulties of iterative methods near the fully-coherent regime

For the target function near the fully-coherent regime, it is hard to guarantee the convergence of optimization-based methods. The ill-conditioned Hessian matrix around the fully-coherent regime poses a challenge for the optimizer to be convergent. The numerical study in Ref. [53] shows that the condition number of the Hessian matrix at the optimum grows rapidly as the target function gets closer to the fully-coherent regime. Furthermore, the theoretical analysis of the optimization landscape also suggests that the region of convergence shrinks as the target function approaches the fully-coherent regime. Hence, the convergence guarantee of optimization-based methods is compromised near the fully-coherent regime.

On the other hand, the analysis in Ref. [54] demonstrates that the FPI method exhibits linear convergence to the exact solution when $\|c\|_1 \leq 0.861$. This result is based on the observation that the update rule acts as a contraction mapping in a neighborhood of the initial guess $\Phi^0 = 0$. However, this property does not hold universally across the entire domain. The analysis in Ref. [54] reveals that the contraction property is valid when the Chebyshev coefficient vector of the target function lies within an ℓ^1 ball centered at the origin. In cases where the target function contains significant "high-frequency" components, the increasingly large Chebyshev coefficient vector may hinder the contraction of the update rule in Eq. (4.25). This situation commonly occurs in various applications; for instance, problematic convergence issues can arise when dealing with functions $\sin(\tau x)$ and $\cos(\tau x)$, where $\tau \gg 1$ is large.

To conclude the discussion on the challenges faced by iterative methods in the fully-coherent regime, we present a numerical result that substantiates these difficulties. We consider the target function to be a degree-733 polynomial approximating $f(x) = 0.999\cos(500x)$ obtained by truncating the Chebyshev expansion. This function violates the convergence analysis of iterative methods, as discussed earlier. In Fig. 4.6, we plot the residual error at each iteration step. The FPI method does not converge at all, while the LBFGS method eventually reaches the optimum. However, the optimizer becomes trapped after the 100th step and requires over 1000 iterations to converge. In contrast, Newton's method, proposed in Ref. [55], exhibits fast and stable convergence in the numerical results. The residual error decreases super-exponentially, consistent with the expected quadratic convergence described in standard textbooks.

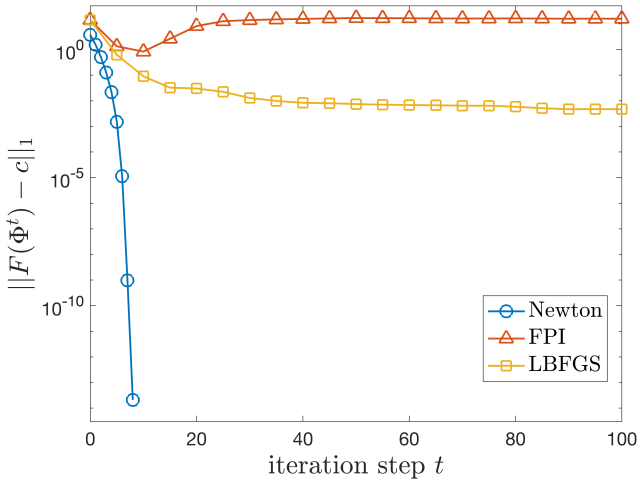


Figure 4.6: The residual error after each iteration using three different methods to determine phase factors for the target function $f(x) = 0.999\cos(500x)$ (up to the first 100 iterations). The stopping criterion is that the residual error reaches below 10^{-13}. Newton's method converges after 9 steps. The LBFGS method can eventually converge but takes over 1000 steps. The FPI method fails to converge.

4.7 Robust iterative method for symmetric quantum signal processing in all parameter regimes

When using iterative algorithms discussed in the previous section to find phase factors, the issue of convergence becomes increasingly significant when the target function is close to the fully-coherent regime. To remedy these difficulties, we propose using Newton's method to solve the nonlinear equation Eq. (4.1) for phase-factor evaluation. In this section, we introduce this method and discuss its implementation. The core techniques to accelerate the algorithm are fast Jacobian evaluation based on the MPS/TT structure and a real-arithmetic formalism of symmetric QSP.

Newton's method can be viewed as an improvement over the FPI method by taking the local landscape into account. It can be verified that the Jacobian of the nonlinear equation in Eq. (4.1) at the origin coincides with a doubled identity matrix, that is, $DF(\mathbf{0}) = 2\mathbf{I}$. Hence, the FPI method is a variant of Newton's method, where the Jacobian is approximated along the iteration by that at the initial point, which

is the origin. To be precise, the update rules $\Phi^{t+1} = T(\Phi^t)$ of both methods are written as

$$\begin{aligned} T_{\text{Newton}}(\Phi) &= \Phi - DF(\Phi)^{-1}(F(\Phi) - c), \\ \text{and } T_{\text{FPI}}(\Phi) &= \Phi - DF(\mathbf{0})^{-1}(F(\Phi) - c), \text{ where } DF(\mathbf{0}) = 2\mathbf{I}. \end{aligned} \tag{4.27}$$

The algorithm based on the first update rule is outlined in Algorithm 4.7.1. In the remainder of this section, we will discuss an accelerated implementation of this algorithm leveraging the structure of the symmetric QSP problem.

Algorithm 4.7.1 Newton's method for finding reduced phase factors

Input: Chebyshev-coefficient vector c of a target polynomial, and stopping criteria.

1: Initiate $t = 0$ and Φ^0 to be zero vector $\mathbf{0}$;
2: **while** stopping criterion is not satisfied **do**
3: Compute $DF(\Phi^t)$;
4: Update $\Phi^{t+1} = \Phi^t - DF(\Phi^t)^{-1}\left(F\left(\Phi^t\right) - c\right)$;
5: Set $t = t + 1$;
6: **end while**

Output: Reduced phase factors Φ^t.

Numerical experiments

We evaluate the performance of Newton's method for finding phase factors in the presented numerical tests (see Section 3.1 for details). All experiments are conducted using Matlab R2020a on a computer with an Intel Core i5 Quad CPU running at 2.11 GHz and 8 GB of RAM.

The performance metrics used to evaluate the performance of Newton's method are the runtime and the residual error. The runtime refers to the amount of time it takes for the method to converge and find the desired phase factors. The residual error measures the discrepancy between the polynomial parametrized by the computed phase factors and the true polynomial which is defined as

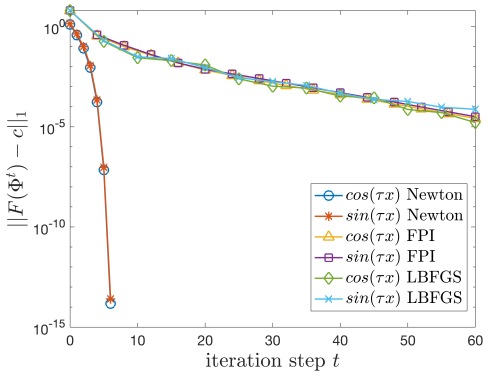
$$\text{residual error} = \left\|F(\Phi) - c\right\|_1. \tag{4.28}$$

The numerical results for the error metric of Newton's method are presented in Fig. 4.7. It is evident from the results that Newton's method exhibits significantly faster convergence compared to other iterative methods for solving phase factors. The error curve aligns well with the expected quadratic convergence of Newton's method in general analysis. Besides, Newton's method exhibits greater stability in terms of

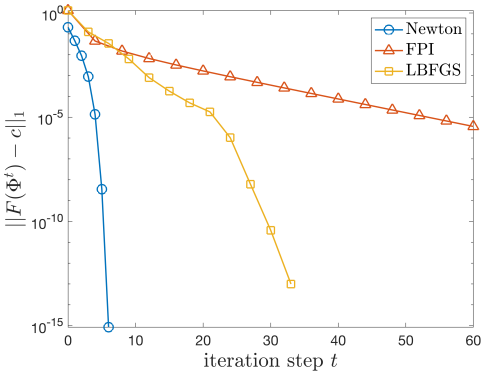
runtime as the target function approaches the fully-coherent regime. Fig. 4.8 depicts the numerical results for the runtime of three iterative methods for determining phase factors. It also clearly illustrates the superior speed of Newton's method compared to the other two iterative methods.

To further analyze the performance of Newton's method near the fully-coherent regime, the runtime and the number of iterations are plotted as a function of the distance to the fully-coherent regime ($1 - \|f\|_\infty$) in Fig. 4.9. It is noteworthy that even when the target function is extremely close to being fully-coherent ($1 - \|f\|_\infty \leq 1 \times 10^{-9}$), Newton's method is capable of locating the optimum within a small number of iterations. This result highlights the robustness of Newton's method for finding phase factors in the nearly fully-coherent regime.

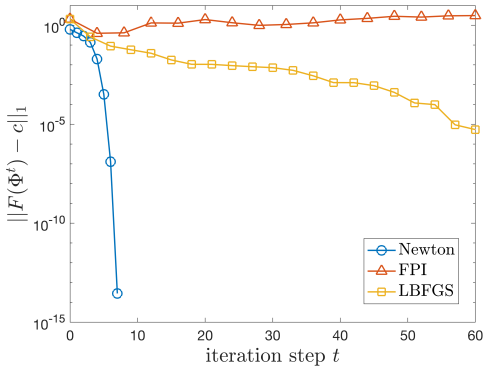
Finally, we investigate the condition number of the Jacobian matrices at the phase factors obtained by Newton's method for different target functions, as presented in Fig. 4.10. The results indicate that as the target function approaches the fully-coherent regime, the condition number of the Jacobian matrix becomes increasingly ill-conditioned. Despite this challenge, Newton's method continues to exhibit remarkable performance in finding phase factors. This emphasizes the effectiveness and reliability of Newton's method in phase factor determination, even in challenging scenarios near the fully-coherent regime.



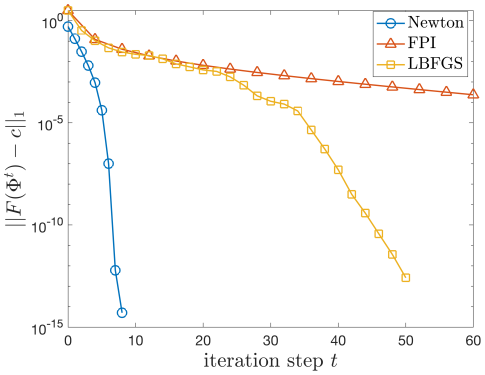
(a) Hamiltonian simulation



(b) Gaussian filter

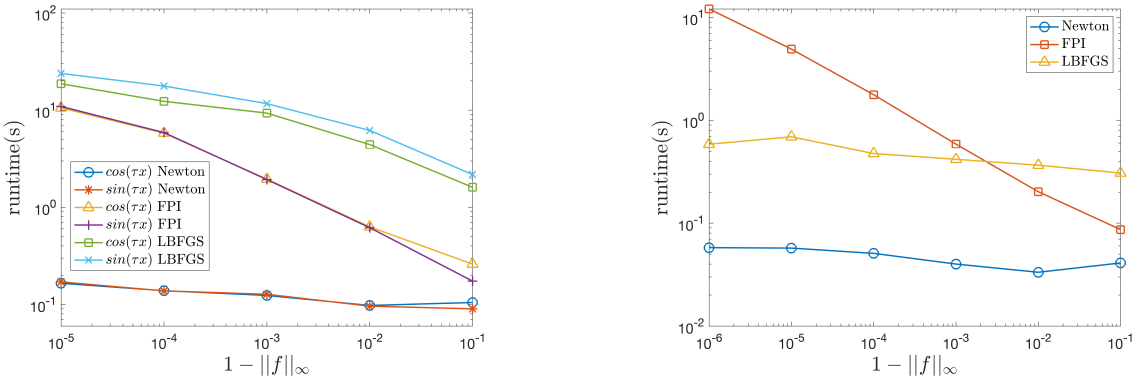


(c) Heaviside energy filter



(d) Matrix inversion

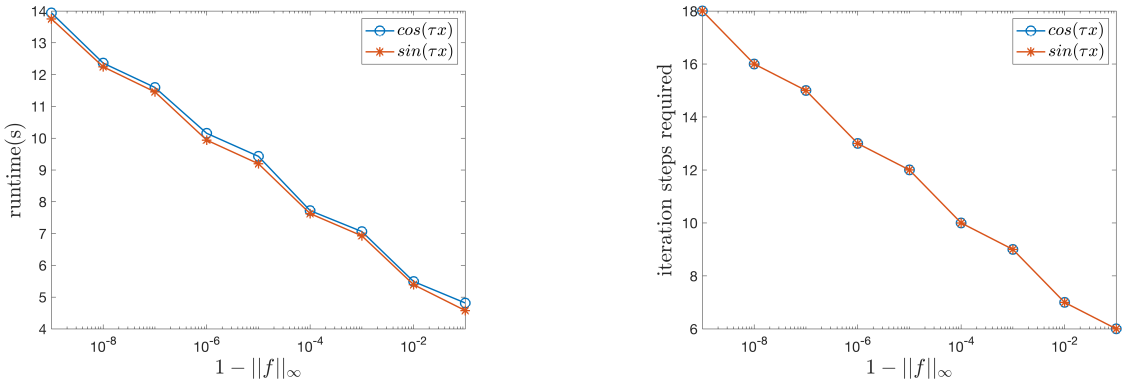
Figure 4.7: Residual error as a function of iteration steps for different numerical examples. The target polynomials in (a)-(d) are chosen to be near the fully-coherent regime, with the maximum absolute value set to 0.99 in (a)-(c) and 0.998 in (d). (a) Quantum Hamiltonian simulation with $\tau = 100$. The target polynomials, obtained by truncating the Jacobi-Anger series with truncation error $\epsilon_0 = 10^{-14}$, have degrees of 1390 and 1391. (b) Quantum Gaussian filter with $\mu = 0.5$ and $\sigma = 0.1$. The target polynomial is derived from the Chebyshev series expansion using FFT, resulting in a degree-100 polynomial. (c) Heaviside energy filter with $\delta = 0.1$. The target polynomial is a degree-250 polynomial obtained from a convex-optimization-based method. (d) Matrix inversion with $\kappa = 10$. The target polynomial is a degree-301 polynomial derived from a convex-optimization-based method.



(a) Hamiltonian simulation

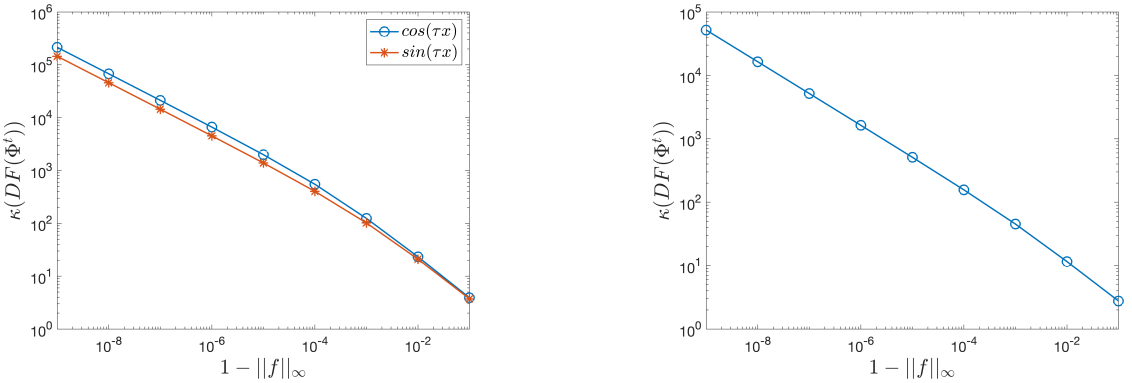
(b) Gaussian filter

Figure 4.8: Runtime analysis of numerical examples for different target functions. The target polynomials in (a) and (b) are scaled by a constant to make the problem increasingly close to the fully-coherent regime $\|f\|_\infty \to 1$. (a) Quantum Hamiltonian simulation with $\tau = 100$ and truncation error $\epsilon_0 = 1 \times 10^{-14}$. The target polynomials have degrees of 167 and 168. (b) Quantum Gaussian filter with $\mu = 0.5$ and $\sigma = 0.01$. The target polynomial is of degree 100.



(a) (b)

Figure 4.9: Convergence analysis of Newton's method near the fully-coherent regime. The target polynomial in each example is scaled by a constant to make the problem increasingly close to the fully-coherent regime $\|f\|_\infty \to 1$. The problem is set to be quantum Hamiltonian simulation with $\tau = 1000$ and truncation error $\epsilon_0 = 1 \times 10^{-14}$. The degrees of the target polynomials are 1390 and 1391. (a) Runtime of Newton's method. (b) Number of iterations before convergence.



(a) Hamiltonian simulation

(b) Gaussian filter

Figure 4.10: Condition number of the Jacobian matrix at the numerical optimum. Each problem is solved by using Newton's method. The target polynomial in each example is scaled by a constant to make the problem increasingly close to the fully-coherent regime. (a) Quantum Hamiltonian simulation with $\tau = 1000$ and truncation error $\epsilon_0 = 1 \times 10^{-14}$. The target polynomials have degrees of 1390 and 1391. (b) Quantum Gaussian filter with $\mu = 0.5$ and $\sigma = 0.01$. The target polynomial is of degree 100.

Chapter 5

Structures of symmetric quantum signal processing problem

Quantum Signal Processing (QSP) utilizes a product of 2×2 unitary matrices to represent a real scalar polynomial of degree d, parameterized by $(d+1)$ real numbers known as phase factors. This innovative approach to polynomial representation finds extensive applications in quantum computation. As a simple yet versatile parametric model within SU(2), QSP exhibits rich structures that not only facilitate the acceleration of numerical solvers for QSP-related problems but also contribute to a deeper theoretical understanding of its mathematical framework. In this chapter, we explore three distinct structures pertinent to the symmetric QSP problem.

The chapter is organized as follows. In Section 5.1, we delve into the matrix product state and its significance in the context of QSP's structure. Section 5.2 explores the representation of QSP using real-number arithmetic, highlighting the resulting acceleration in algorithmic performance. Lastly, Section 5.3 introduces a generalization of the QSP formalism to scenarios involving infinitely long phase factors. This extension broadens the parametrization scope from merely polynomials to a wider range of non-polynomial functions. Moreover, our analysis uncovers an intriguing correlation between the regularity of the target function and the decay characteristics of the phase factors.

Please note that Sections 5.1 and 5.2 are based on [55] (joint work with Lin Lin, Hongkang Ni, and Jiasu Wang) and Section 5.3 is based on [54] (joint work with Lin Lin, Hongkang Ni, and Jiasu Wang).

5.1 Matrix product state structure and efficient evaluation of the Jacobian matrix

In gradient-based iterative methods for finding phase factors, the evaluation of the Jacobian matrix $DF(\Phi)$ constitutes the most important but computationally demanding step. In this section, we study a special structure of the QSP problem, and propose an efficient approach to compute the Jacobian matrix taking advantage of this structure. By doing so, we can significantly reduce the overall computational complexity. As we will illustrate, the computation of different columns of the Jacobian matrix exhibits substantial overlap. This indicates that we can reuse intermediate computational results and avoid redundancy, leading to increased efficiency.

Matrix product state structure of quantum signal processing

The QSP problem, being a well-structured product of SU(2) matrices, possesses inherent properties that allow for a special tensor structure known as a matrix product state (MPS) or tensor train (TT). These properties can be effectively leveraged to accelerate our numerical algorithm. By exploiting this tensor structure, we achieve a significant reduction in computation complexity, enabling the scalability of the solver for larger-scale applications.

In this subsection, we present a concise overview of the theory and construction of MPS/TT. Additionally, we establish its connection with our problem. To be specific, QSP admits an MPS/TT structure with bond dimension 2 due to its SU(2)-product defining equation Eq. (4.2).

Given a field $\mathbb{F} = \mathbb{R}$ or $\mathbb{C}$, an *order-r tensor* is referred to as a multidimensional array $G \in \mathbb{F}^{n_1 \times \cdots \times n_r}$ where the r-tuple $(n_1, \cdots, n_r) \in \mathbb{N}^r$ specifies the size of the tensor. Each entry of the tensor can be accessed with multi-indices, $G(i_1, \cdots, i_r)$, where $1 \leq i_j \leq n_j, \forall j = 1, \cdots, r$. The contraction of two tensors yields a new tensor by summing over the specified indices. For example, if $G_1 \in \mathbb{F}^{n_1 \times n_2 \times n_3}, G_2 \in \mathbb{F}^{m_1 \times m_2 \times n_3}$ are two order-3 tensors, $G_3(i, j, k, l) := \sum_{s=1}^{n_3} G_1(i, j, s) G_2(k, l, s)$ elementwisely defines an order-4 tensor $G_3 \in \mathbb{F}^{n_1 \times n_2 \times m_1 \times m_2}$ by contracting the index s. A graphical illustration is given in Fig. 5.1. Specifically, the contraction of order-2 tensors (i.e. matrices) coincides with matrix multiplication.

A (parametric) tensor $G(\alpha) \in \mathbb{F}^{n_1 \times \cdots \times n_r}$ is called an MPS or TT if each of its entries can be expressed as a product of matrices [120]. To be precise, an MPS/TT can be written as

$$G(i_1, \cdots, i_r; \alpha) = \mathcal{G}_1(i_1; \alpha) \mathcal{G}_2(i_2; \alpha) \cdots \mathcal{G}_r(i_r; \alpha).$$

In the above expression, each $\mathcal{G}_j(\alpha)$ is an order-3 tensor. By fixing an index i_j, $\mathcal{G}_j(i_j;\alpha) = [G_j(k,l,i_j;\alpha)]_{kl} \in \mathbb{F}^{m_{j-1}\times m_j}$ becomes order-2 which is equivalent to a matrix of size (m_{j-1}, m_j). The dangling index i_j is referred to as the mode index (or external index). The indices k, l which are dummy in contraction are referred to as rank core indices. The right-hand side of the defining equation is a shorthand for contracting all non-fixed indices by matrix multiplication. The contracted tensor $G(i_1, \cdots, i_r;\alpha)$ is assumed to be a scalar entry-wisely. Hence, the dangling components $\mathcal{G}_1$ and $\mathcal{G}_r$ are set to order-2 tensors, namely, $m_0 = m_r = 1$. The bond dimension of an MPS/TT is defined to be the maximal dimension in the contraction $\max_{0\le j\le r} m_j$.

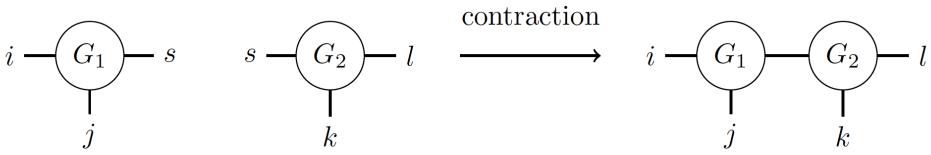


Figure 5.1: A graphical visualization of two order-3 tensors and their contraction.

Using the language of tensors, the upper-left entry of the QSP unitary matrix defined in Eq. (4.2) can be interpreted as an MPS/TT of bond dimension 2. To see this, assuming that x and a full set of phase factors $\Psi := (\psi_0, \cdots, \psi_d)$ are given, the QSP matrix entry of interest is

$$\langle 0|U(x,\Psi)|0\rangle = \mathcal{R}_0(\psi_0)\mathcal{W}(x)\mathcal{R}(\psi_1)\mathcal{W}(x)\cdots\mathcal{W}(x)\mathcal{R}(\psi_{d-1})\mathcal{W}(x)\mathcal{R}_d(\psi_d) \in \mathbb{C}. \quad (5.1)$$

Here, the boundary components $\mathcal{R}_0(\psi_0) := (e^{\mathrm{i}\psi_0}, 0), \mathcal{R}_d(\psi_d) := (e^{\mathrm{i}\psi_d}, 0)^\top$ are two-dimensional complex vectors. Furthermore, $\mathcal{W}(x) := e^{\mathrm{i}\arccos(x)X}$ and $\mathcal{R}(\psi_j) := e^{\mathrm{i}\psi_j Z}$ are 2-by-2 complex matrices. By identifying x, ψ as external indices, the graphical visualization of this interpretation is given in Fig. 5.2.

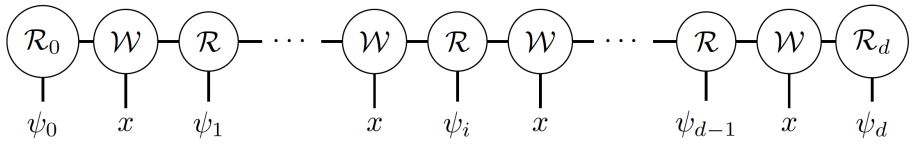


Figure 5.2: A graphical visualization of $\langle 0|U(x,\Psi)|0\rangle$ as a MPS/TT structure (of bond dimension 2).

Jacobian of QSP problem

According to the defining equation Eq. (4.10) of F, the i-th column of $DF(\Phi)$ is

$$\frac{\partial F(\Phi)}{\partial \phi_i} = \mathcal{F}\left(\frac{\partial g(x,\Phi)}{\partial \phi_i}\right) \in \mathbb{R}^{\widetilde{d}},\ i = 0, \cdots, \widetilde{d}-1. \tag{5.2}$$

A straightforward approach for constructing the Jacobian matrix is to compute it column-wise without any optimization. This method involves performing the following procedure independently for each $0 \leq i < \widetilde{d}$: evaluating $\partial g(x_k, \Phi)/\partial \phi_i$ at approximately $\mathcal{O}(d)$ distinct points and then applying a discrete Fourier transformation. Each evaluation of $\partial g(x_k, \Phi)/\partial \phi_i$ requires $\mathcal{O}(d)$ multiplications of SU(2). Consequently, the complexity for computing a column of the Jacobian is $\mathcal{O}(d^2)$. As a result, the overall complexity of this Jacobian evaluation is $\mathcal{O}(d^3)$. It is important to note that this approach does not take into account the structural characteristics of the problem, leaving room for potential optimization strategies.

In the subsequent subsection, we will present an accelerated evaluation method that capitalizes on the MPS/TT structure of the problem. This improved approach leads to a notable reduction in the complexity of Jacobian evaluation, from $\mathcal{O}(d^3)$ to $\mathcal{O}(d^2 \log(d))$.

In the remainder of this subsection, we delve into the structure of the Jacobian matrix columns. For the sake of simplicity, we assume that d is even. As a reminder, in the even case, the full set of phase factors can be represented as $\Psi = (\phi_{\widetilde{d}-1}, \cdots, \phi_1, 2\phi_0, \phi_1, \cdots, \phi_{\widetilde{d}-1})$. This choice does not lose generality, as a similar derivation can be obtained for the odd case.

We first observe that $\frac{\partial g(x,\Phi)}{\partial \phi_i} = \mathrm{Im}\left[\langle 0|\frac{\partial}{\partial \phi_i} U(x,\Psi)|0\rangle\right]$, and that taking derivative on the unitary matrix $U(x,\Psi)$ is equivalent to the insertion of an additional $\mathrm{i}Z = e^{\mathrm{i}\pi Z/2}$ in the matrix product. Due to symmetry, the derivative leads to two matrix products with insertion. Specifically, we have:

$$\frac{\partial}{\partial \phi_i} U(x,\Psi) = U(x, \Psi + \frac{\pi}{2} e_{\widetilde{d}-1-i}) + U(x, \Psi + \frac{\pi}{2} e_{\widetilde{d}-1+i}), \quad \forall 0 \leq i \leq \widetilde{d}-1. \tag{5.3}$$

We remark that $\Psi + \frac{\pi}{2} e_{\widetilde{d}-1+i}$ is the reversed ordering of $\Psi + \frac{\pi}{2} e_{\widetilde{d}-1-i}$, which is helpful for the simplification. Consequently, $U(x, \Psi + \frac{\pi}{2} e_{\widetilde{d}-1+i})$ is identical to the transpose of $U(x, \Psi + \frac{\pi}{2} e_{\widetilde{d}-1-i})$ since the transpose reverses the order due to the symmetry of

matrix Z. Hence

$$\begin{aligned}\frac{\partial g(x,\Phi)}{\partial \phi_i} &= \mathrm{Im}\left[\langle 0|\frac{\partial}{\partial \phi_i}U(x,\Phi)|0\rangle\right] \\ &= \mathrm{Im}[\langle 0|U(x,\Psi+\frac{\pi}{2}e_{\widetilde{d}-1-i})|0\rangle] + \mathrm{Im}[\langle 0|U(x,\Psi+\frac{\pi}{2}e_{\widetilde{d}-1+i})|0\rangle] \\ &= \mathrm{Im}[\langle 0|U(x,\Psi+\frac{\pi}{2}e_{\widetilde{d}-1-i})|0\rangle] + \mathrm{Im}[\langle 0|U(x,\Psi+\frac{\pi}{2}e_{\widetilde{d}-1-i})^\top|0\rangle] \\ &= 2\mathrm{Im}[\langle 0|U(x,\Psi+\frac{\pi}{2}e_{\widetilde{d}-1-i}))|0\rangle] = 2g(x,\Psi+\frac{\pi}{2}e_{\widetilde{d}-1-i}).\end{aligned} \tag{5.4}$$

While $\Psi+\frac{\pi}{2}e_{\widetilde{d}-1-i}$ is not symmetric, the evaluation of the induced polynomial is still well defined. To extract its Chebyshev coefficients, it suffices to sample this polynomial on the Chebyshev nodes $\{x_k=\cos(2\pi k/(2d+1)) : k=0,\cdots,d\}$. Subsequently, the Chebyshev coefficients can be extracted from the sample by performing Fast Fourier Transformation (FFT). The detail is given in Algorithm 5.1.1.

Algorithm 5.1.1 Compute $\mathcal{F}(g(x,\Psi^\sharp))$.

Input: A full set of phase factors $\Psi^\sharp$ of length $d+1$ ($\Psi^\sharp$ is not necessarily symmetric).

1: Initialize $\mathbf{g}=(0,0,\cdots)\in\mathbb{R}^{2d+1}$.
2: Evaluate $\mathbf{g}_j \leftarrow g(x_j,\Psi^\sharp), x_j=\cos\left(\frac{2\pi j}{2d+1}\right), j=0,\cdots,d$.
3: Evaluate $\mathbf{g}_j \leftarrow \mathbf{g}_{2d+1-j}$, $j=d+1,\cdots,2d$.
4: Compute $\mathbf{v}_l \leftarrow \mathrm{Re}\left(\sum_{j=0}^{2d-1}\mathbf{g}_j e^{-\mathrm{i}\frac{2\pi}{2d+1}jl}\right), l=0,\ldots,d$ using FFT.
5: **if** $(d \bmod 2)=0$ **then**
6: $\quad\mathcal{F}(g(x,\Psi^\sharp)) \leftarrow \frac{2}{2d+1}(\frac{\mathbf{v}_0}{2},\mathbf{v}_2,\mathbf{v}_4,\cdots,\mathbf{v}_d)$.
7: **else**
8: $\quad\mathcal{F}(g(x,\Psi^\sharp)) \leftarrow \frac{2}{2d+1}(\mathbf{v}_1,\mathbf{v}_3,\mathbf{v}_5,\cdots,\mathbf{v}_d)$.
9: **end if**

Output: $\mathcal{F}(g(x,\Psi^\sharp))$.

Efficient evaluation of the Jacobian matrix

Without loss of generality, we consider the case where d is even in the derivation. The odd case can be analyzed similarly. Recall that the full set of phase factors is defined as

$$\Psi := (\psi_0,\psi_1,\cdots,\psi_d) = (\phi_{\widetilde{d}-1},\cdots,\phi_1,2\phi_0,\phi_1,\cdots,\phi_{\widetilde{d}-1}), \tag{5.5}$$

where $\Phi=(\phi_0,\cdots,\phi_{\widetilde{d}-1})$ are the reduced phases factors. We observe that each column of the Jacobian matrix is directly associated with taking the derivative of

a QSP without symmetry, which arises from the insertion of the iZ matrix. In the absence of symmetry constraint of phase factors, each phase factor ϕ_i is independent. When calculating the derivative with respect to ϕ_i, we can separate the matrix multiplication into three parts

$$\langle 0|U(x,\Psi)|0\rangle = \mathcal{M}_{\text{left}}^{(i)} e^{\mathrm{i}\phi_i Z} \mathcal{M}_{\text{right}}^{(i)}, \tag{5.6}$$

where

$$\begin{aligned} \mathcal{M}_{\text{left}}^{(i)} &:= \mathcal{R}_0(\phi_{\widetilde{d}-1}) \prod_{j=\widetilde{d}-2}^{i+1} [\mathcal{W}(x)\mathcal{R}(\phi_j)]\, \mathcal{W}(x), \\ \mathcal{M}_{\text{right}}^{(i)} &:= \prod_{j=i-1}^{0} [\mathcal{W}(x)\mathcal{R}(\phi_j)] \prod_{j=0}^{\widetilde{d}-2} [\mathcal{R}(\phi_j)\mathcal{W}(x)]\, \mathcal{R}_d(\phi_{\widetilde{d}-1}). \end{aligned} \tag{5.7}$$

The left and right components are irrelevant to taking the derivative with respect to ϕ_i because

$$\langle 0|\partial_{\phi_i} U(x,\Phi)|0\rangle = 2\langle 0|U(x,\Psi+\frac{\pi}{2}e_{k-i})|0\rangle = 2\mathrm{i}\mathcal{M}_{\text{left}}^{(i)} Z e^{\mathrm{i}\phi_i Z} \mathcal{M}_{\text{right}}^{(i)}. \tag{5.8}$$

Consequently, the intermediate quantities $\mathcal{M}_{\text{left}}^{(i)}$ and $\mathcal{M}_{\text{right}}^{(i)}$ can be stored and maintained in the computation process. Figure 5.3 visually illustrates this idea.

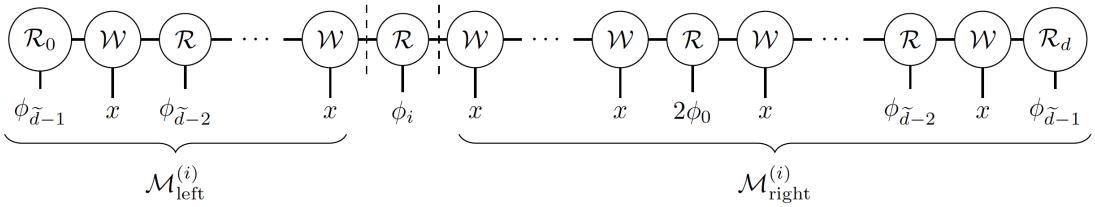


Figure 5.3: A graphical visualization of the isolation and grouping when evaluating the derivative $\langle 0|\partial_{\phi_i} U(x,\Phi)|0\rangle$.

Transiting to the next step, the intermediate quantities are updated through matrix multiplications

$$\mathcal{M}_{\text{left}}^{(i+1)} \leftarrow \mathcal{M}_{\text{left}}^{(i)} \mathcal{W}^{-1}(x) e^{-\mathrm{i}\phi_{i+1} Z}, \quad \text{and} \quad \mathcal{M}_{\text{right}}^{(i+1)} \leftarrow \mathcal{W}(x) e^{\mathrm{i}\phi_i Z} \mathcal{M}_{\text{right}}^{(i)}.$$

By utilizing intermediate quantities, the computation of the derivatives, which are the columns of the Jacobian matrix before FFT, can be performed simultaneously,

resulting in a computational cost of $\mathcal{O}(d^2)$ rather than $\mathcal{O}(d^3)$ in the straightforward method. The overall complexity of computing the Jacobian matrix is $\mathcal{O}(d^2 \log d)$ due to the use of FFT. The detailed procedure is summarized in Algorithm 5.1.2.

Algorithm 5.1.2 Compute Jacobian matrix $DF(\Phi)$ using the MPS structure.

Input: Reduced phase factors Φ of length $\widetilde{d}$ and parity.
Set $d = 2\widetilde{d} - 2 +$ parity and initialize $\mathbf{g}$ as a zero matrix of size $\widetilde{d} \times (2d+1)$.
for $j = 0, \cdots, d$ **do**
  Set $x_j = \cos\left(\frac{2\pi j}{2d+1}\right)$.
  $\mathcal{M}_{\text{left}}(x_j) = (1,0)\prod_{i=\widetilde{d}-1}^{1}\left(e^{\mathrm{i}\phi_i Z}W(x_j)\right)$.
  $\mathcal{M}_{\text{right}}(x_j) = e^{\mathrm{i}\phi_0 Z}\mathcal{M}_{\text{left}}(x_j)^{\top}$.
  if parity is odd **then**
    $\mathcal{M}_{\text{right}}(x_j) = W(x_j)\mathcal{M}_{\text{right}}(x_j)$.
  end if
  $g_{0,j} \leftarrow 2\mathrm{Im}[\mathcal{M}_{\text{left}}\mathrm{i}Z\mathcal{M}_{\text{right}}]$.
  for $i = 1, \cdots, \widetilde{d}-1$ **do**
    $\mathcal{M}_{\text{left}}(x_j) \leftarrow \mathcal{M}_{\text{left}}(x_j)W^{-1}(x_j)e^{-\mathrm{i}\phi_i Z}$.
    $\mathcal{M}_{\text{right}}(x_j) \leftarrow W(x_j)e^{\mathrm{i}\phi_i Z}\mathcal{M}_{\text{right}}(x_j)$.
    $g_{i,j} \leftarrow 2\mathrm{Im}[\mathcal{M}_{\text{left}}\mathrm{i}Z\mathcal{M}_{\text{right}}]$.
  end for
end for
Evaluate $\mathbf{g}_{i,j} \leftarrow \mathbf{g}_{i,2d+1-j}$, $j = d+1, \cdots, 2d$.
Compute $\mathbf{v}_{il} \leftarrow \mathrm{Re}\left(\sum_{j=0}^{2d-1}\mathbf{g}_{i,j}e^{-\mathrm{i}\frac{2\pi}{2d+1}lj}\right), l = 0, \ldots, d$ using FFT.
if parity $= 0$ **then**
  $\frac{\partial F(\Phi)}{\partial \phi_i} \leftarrow \frac{2}{2d+1}\left(\frac{\mathbf{v}_{i0}}{2}, \mathbf{v}_{i2}, \mathbf{v}_{i4}, \cdots, \mathbf{v}_{id}\right)$.
else
  $\frac{\partial F(\Phi)}{\partial \phi_i} \leftarrow \frac{2}{2d+1}\left(\mathbf{v}_{i1}, \mathbf{v}_{i3}, \mathbf{v}_{i5}, \cdots, \mathbf{v}_{id}\right)$.
end if
Output: $DF(\Phi)$.

5.2 Formalism of symmetric QSP in real-number arithmetic operations

In the existing literature, the conventional formalism of QSP is typically presented in terms of the product of SU(2) matrices, which involves complex-number arithmetic operations. This complex-number arithmetic formalism is both necessary and sufficient for general QSP, as the induced polynomials P and Q are complex without any

additional symmetry constraints. However, in the case of symmetric QSP, according to Theorem 4.1.1, the polynomial Q is a real polynomial. This observation raises the question of whether the formalism of QSP can be simplified to accommodate this symmetry.

In this section, we will introduce a formalism for symmetric QSP that utilizes real-number arithmetic operations. This alternative formalism proves to be beneficial for the implementation of algorithms designed to solve symmetric QSP, resulting in a constant improvement in the prefactor of the overall computational complexity.

The core idea is that SU(2) is homeomorphic to $\mathbb{S}^3 \subset \mathbb{R}^4$, which arises from the parametric form of general SU(2) matrices. By imposing the symmetric constraint, the upper-right entry of the consequent SU(2) matrix is purely imaginary. Consequently, we can associate any symmetric QSP matrix with a vector in $\mathbb{S}^2 \subset \mathbb{R}^3$. The identification is

$$\begin{aligned} U(x,\Phi) &= \begin{pmatrix} p(x,\Phi)+\mathrm{i}g(x,\Phi) & \mathrm{i}\sqrt{1-x^2}q(x,\Phi) \\ \mathrm{i}\sqrt{1-x^2}q(x,\Phi) & p(x,\Phi)-\mathrm{i}g(x,\Phi) \end{pmatrix} \in \mathrm{SU}(2) \\ \leftrightarrow V(x,\Phi) &:= \big(p(x,\Phi),\ g(x,\Phi),\ \sqrt{1-x^2}q(x,\Phi)\big)^\top \in \mathbb{S}^2. \end{aligned} \tag{5.9}$$

Under the identification we introduced, the matrix multiplication in symmetric QSP is equivalent to interleaved rotations in SO(3). This relation is quantified by the following recurrence equation:

$$V(x,(\phi_k,\phi_{k-1},\cdots)) = R_z(2\phi_k)R_x(2\arccos(x))V(x,(\phi_{k-1},\cdots)), \tag{5.10}$$

where the SO(3) rotations are

$$R_z(2\phi) = \begin{pmatrix} \cos 2\phi & -\sin 2\phi & \\ \sin 2\phi & \cos 2\phi & \\ & & 1 \end{pmatrix} \text{ and } R_x(2\theta) = \begin{pmatrix} \cos(2\theta) & & -\sin(2\theta) \\ & 1 & \\ \sin(2\theta) & & \cos(2\theta) \end{pmatrix}. \tag{5.11}$$

By leveraging this identification, the QSP polynomials can be derived from the product of real matrices, leading to a faster computation with a constant improvement in the prefactor, compared to evaluating them using the product of complex matrices.

Details about the formalism of symmetric QSP in real-number arithmetic operations

In this subsection, we aim to provide a more comprehensive discussion and present additional details on the real-number arithmetic representation of QSP.

The computation of the QSP matrix boils down to that of a sequence of unitary matrix multiplications in Eq. (4.2). Furthermore, the QSP matrix admits the following decomposition as a consequence of Theorem 4.1.1

$$U(x,\Phi)=\begin{pmatrix} a_{\widetilde{d}-1}(x)+\mathrm{i}d_{\widetilde{d}-1}(x) & \mathrm{i}\alpha_{\widetilde{d}-1}(x) \\ \mathrm{i}\alpha_{\widetilde{d}-1}(x) & a_{\widetilde{d}-1}(x)-\mathrm{i}d_{\widetilde{d}-1}(x)\end{pmatrix}. \tag{5.12}$$

Here, $a_{\widetilde{d}-1}(x)$, $d_{\widetilde{d}-1}(x)$ and $\alpha_{\widetilde{d}-1}(x)/\sqrt{1-x^2}$ are real polynomials in the variable x. According to the presented convention, $d_{\widetilde{d}-1}$ stands for the component of interest. It is also known as $g(x,\Phi)$ in previous chapters which emphasizes the dependence in phase factors Φ. As the goal in this subsection is to derive a simple recipe for computing the QSP matrix with a given set of phase factors Φ, we drop the Φ dependence in this section for the notational simplicity.

Let the entry-wise value of the phase factors be $\Phi=(\phi_0,\phi_1,\cdots,\phi_{\widetilde{d}-1})$. For ease of discussion, we refer to $\Phi^{(k)}=(\phi_0,\phi_1,\cdots,\phi_k)$ as the k-th truncated phase factors for each $k=0,1,\cdots,\widetilde{d}-1$. The corresponding sequence of QSP matrices is denoted entry-wise as

$$U(x,\Phi^{(k)})=\begin{pmatrix} a_k(x)+\mathrm{i}d_k(x) & \mathrm{i}\alpha_k(x) \\ \mathrm{i}\alpha_k(x) & a_k(x)-\mathrm{i}d_k(x)\end{pmatrix}. \tag{5.13}$$

We remark that each truncated set of phase factors also gives a symmetric QSP. Hence, the decomposition in Eq. (5.12) applies, implying that $a_k(x)$, $\alpha_k(x)$ and $d_k(x)$ are well defined. By appending ϕ_k to the $(k-1)$-th truncation $\Phi^{(k-1)}$, the recurrence relation follows

$$\begin{aligned}&\begin{pmatrix} a_k(x)+\mathrm{i}d_k(x) & \mathrm{i}\alpha_k(x) \\ \mathrm{i}\alpha_k(x) & a_k(x)-\mathrm{i}d_k(x)\end{pmatrix} \\ &=e^{\mathrm{i}\phi_k Z}W(x)\begin{pmatrix} a_{k-1}(x)+\mathrm{i}d_{k-1}(x) & \mathrm{i}\alpha_{k-1}(x) \\ \mathrm{i}\alpha_{k-1}(x) & a_{k-1}(x)-\mathrm{i}d_{k-1}(x)\end{pmatrix}W(x)e^{\mathrm{i}\phi_k Z}.\end{aligned} \tag{5.14}$$

It can be verified that the following rearrangement is equivalent to the recurrence relation

$$\begin{pmatrix} a_k(x) \\ d_k(x) \\ \alpha_k(x)\end{pmatrix}=R_z(2\phi_k)R_x(2\arccos(x))\begin{pmatrix} a_{k-1}(x) \\ d_{k-1}(x) \\ \alpha_{k-1}(x)\end{pmatrix}. \tag{5.15}$$

It can also be shown that the base cases of the recurrence are

(1) when d is even

$$\begin{pmatrix} a_0(x) \\ d_0(x) \\ \alpha_0(x)\end{pmatrix}=\begin{pmatrix} \cos(2\phi_0) \\ \sin(2\phi_0) \\ 0\end{pmatrix}, \tag{5.16}$$

and (2) when d is odd

$$\begin{pmatrix} a_0(x) \\ d_0(x) \\ \alpha_0(x) \end{pmatrix} = \begin{pmatrix} \cos(2\phi_0)x \\ \sin(2\phi_0)x \\ \sqrt{1-x^2} \end{pmatrix}. \tag{5.17}$$

Remarkably, the equivalent recurrence relation Eq. (5.15) involves only real quantities, and R_x and R_z only actively act as a rotation on two entries. In contrast to the complex recurrence relation Eq. (5.14), the real recurrence has lower time and space complexity. This improvement is due to the simplified structure of symmetric QSP compared with the original formalism without symmetry.

The MPS/TT structure still holds in the real recurrence relation. We refer $\mathcal{X}$ and $\mathcal{Z}$ to the parametric order-2 tensor standing for the induced SO(3) rotations, namely, $\mathcal{Z}(\phi) = R_z(2\phi)$ and $\mathcal{X}(x) = R_x(2\arccos(x))$. Let $\mathcal{I}$ be the order-1 tensor representing the base of the recurrence in Eqs. (5.16) and (5.17). Furthermore, to extract the component of the computational interest, let $\mathcal{H}$ be the order-1 tensor representing the last operation, which is

$$\mathcal{H}(\phi_{\widetilde{d}-1}) = \begin{pmatrix} 0 & 1 & 0 \end{pmatrix} R_z(2\phi_{\widetilde{d}-1}) = \begin{pmatrix} \sin(2\phi_{\widetilde{d}-1}) & \cos(2\phi_{\widetilde{d}-1}) & 0 \end{pmatrix}. \tag{5.18}$$

Then, the recurrence relation in real-number arithmetic can be visualized graphically in Fig. 5.4. In contrast to the computation in the complex-arithmetic representation, the symmetry constraint of the QSP phase factors is reflected in the doubled argument in the Z tensor of the real-number arithmetic representation. Hence, when computing the derivative, it does not need tricks to arrange the derivatives coming from two symmetric sites. Specifically, the following identity holds

$$\frac{\partial g(x,\Phi)}{\partial \phi_i} = \mathcal{N}^{(i)}_{\text{left}} \frac{\mathrm{d}\mathcal{Z}(\phi_i)}{\mathrm{d}\phi_i} \mathcal{N}^{(i)}_{\text{right}} = 2\mathcal{N}^{(i)}_{\text{left}} \mathcal{Z}(\phi_i + \pi/4) \mathcal{N}^{(i)}_{\text{right}}. \tag{5.19}$$

Here, the left and right parts under the partition are given by

$$\mathcal{N}^{(i)}_{\text{left}} := \mathcal{H} \prod_{j=\widetilde{d}-2}^{i+1} \left(\mathcal{X}(x)\mathcal{Z}(\phi_j)\right) \mathcal{X}(x), \quad \mathcal{N}^{(i)}_{\text{right}} := \prod_{j=i-1}^{1} \mathcal{X}(x)\mathcal{Z}(\phi_j)\mathcal{I}, \tag{5.20}$$

whose graphical visualizations are presented in Fig. 5.4. The update of these quantities in the computational process is

$$\mathcal{N}^{(i+1)}_{\text{left}} \leftarrow \mathcal{N}^{(i)}_{\text{left}} \mathcal{X}^{-1}(x) \mathcal{Z}(-\phi_{i+1}), \quad \text{and} \quad \mathcal{N}^{(i+1)}_{\text{right}} \leftarrow \mathcal{Z}(\phi_i)\mathcal{X}(x)\mathcal{N}^{(i)}_{\text{right}}. \tag{5.21}$$

For completeness, we provide the algorithm for computing the Jacobian matrix using the MPS/TT structure and the real-number arithmetic representation in Algorithm 5.2.1.

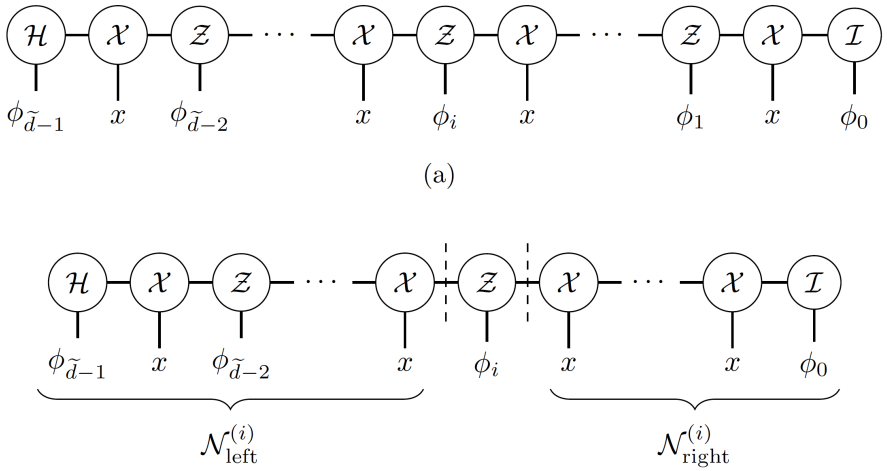


Figure 5.4: A graphical visualization of the MPS/TT structure of the problem in the real-number arithmetic representation. (a) The structure of the recurrence relation. (b) The partition when computing the Jacobian.

Algorithm 5.2.1 computing the Jacobian matrix using the MPS/TT structure and the real-number arithmetic representation.

Input: A set of reduced phase factors Φ, its length $\widetilde{d}$ and its parity $p \in \{0, 1\}$.

Set $d = 2\widetilde{d} - 2 + p$ and initialize $\mathbf{g}$ as a zero matrix of size $\widetilde{d} \times (2d + 1)$.
for $j = 0, \cdots, d$ **do**
  Set $x_j = \cos\left(\frac{2\pi j}{2d+1}\right)$.
  $\mathcal{N}_{\text{left}}(x_j) = \mathcal{H} \prod_{i=\widetilde{d}-2}^{1} \left(\mathcal{X}(x_j)\mathcal{Z}(\phi_i)\right) \mathcal{X}(x_j)$.
  if parity is even $(p = 0)$ **then**
    $\mathcal{N}_{\text{right}}(x_j) = (1, 1, 0)^\top$.
  else
    $\mathcal{N}_{\text{right}}(x_j) = (x, x, \sqrt{1 - x^2})^\top$.
  end if
  $g_{0,j} \leftarrow 2\mathcal{N}_{\text{left}}\mathcal{Z}(\phi_0 + \pi/4)\mathcal{N}_{\text{right}}(x_j)$.
  for $i = 1, \cdots, \widetilde{d} - 1$ **do**
    $\mathcal{N}_{\text{left}}(x_j) \leftarrow \mathcal{N}_{\text{left}}(x_j)\mathcal{X}^{-1}(x_j)\mathcal{Z}(-\phi_i)$.
    $\mathcal{N}_{\text{right}}(x_j) \leftarrow \mathcal{X}(x_j)\mathcal{Z}(\phi_{i-1})\mathcal{N}_{\text{right}}(x_j)$.
    $g_{i,j} \leftarrow 2\mathcal{N}_{\text{left}}(x_j)\mathcal{Z}(\phi_i + \pi/4)\mathcal{N}_{\text{right}}(x_j)$.

end for
end for
Set $\mathbf{g}_{i,j} \leftarrow \mathbf{g}_{i,2d+1-j}$, $j = d+1, \cdots, 2d$.
Compute $\mathbf{v}_{il} \leftarrow \mathrm{Re}\left(\sum_{j=0}^{2d-1} \mathbf{g}_{i,j} e^{-\mathrm{i}\frac{2\pi}{2d+1}lj}\right), l = 0, \ldots, d$ by using FFT.
if parity is even $(p = 0)$ **then**
    $\frac{\partial F(\Phi)}{\partial \phi_i} \leftarrow \frac{2}{2d+1}\left(\frac{\mathbf{v}_{i0}}{2}, \mathbf{v}_{i2}, \mathbf{v}_{i4}, \cdots, \mathbf{v}_{id}\right)$.
else
    $\frac{\partial F(\Phi)}{\partial \phi_i} \leftarrow \frac{2}{2d+1}\left(\mathbf{v}_{i1}, \mathbf{v}_{i3}, \mathbf{v}_{i5}, \cdots, \mathbf{v}_{id}\right)$.
end if
Output: $DF(\Phi)$.

In Fig. 5.5, we numerically demonstrate that using the real-number arithmetic formalism of QSP improves the time complexity of iterative methods by a constant prefactor. Notably, this improvement is not limited to Newton's method but also applies to other iterative methods for finding phase factors.

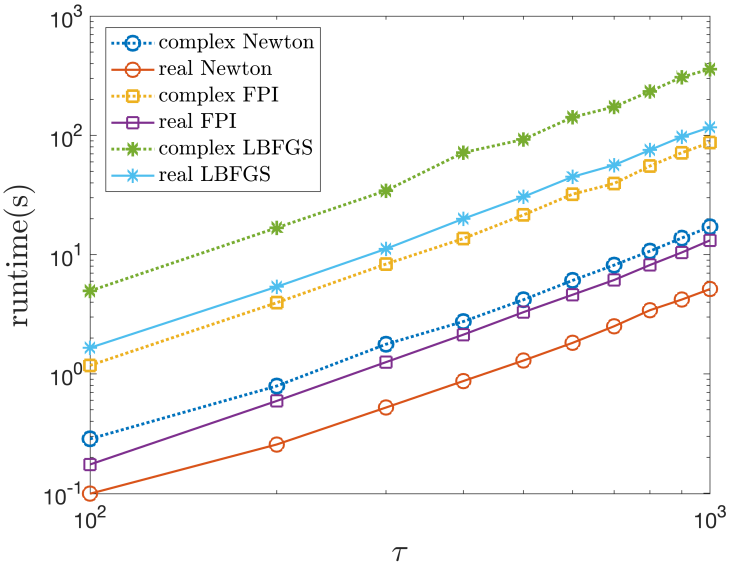


Figure 5.5: Comparing the runtime of iterative methods for finding phase factors using the real-number and complex-number arithmetic. The problem is set to quantum Hamiltonian simulation with variable τ parameters. The target polynomial is derived by truncating the Jacobi-Anger series with truncation error $\epsilon_0 = 1 \times 10^{-14}$. The maximal value of the target polynomial is scaled by a constant so that $\|f\|_\infty = 0.9$.

5.3 Infinite quantum signal processing and tail decay property of reduced phase factors

The study of the representation of polynomials has a long history, with rich applications in a diverse range of fields. It is therefore exciting that a new way of representing polynomials, called quantum signal processing[1] (QSP) [101, 67], has emerged recently in the context of quantum computation. QSP is an innovative way of encoding the information of a polynomial in terms of 2×2 unitary matrices. This construction has found many applications, such as Hamiltonian simulation [101, 67], solving linear system of equations [67, 95, 105], solving eigenvalue problems [94, 51], preparing Gibbs states [67], Petz recovery channel [65], benchmarking quantum systems [50, 52], to name a few. We refer interested readers to Refs. [67, 105].

To implement QSP and its high-dimensional realization QSVT, we need to efficiently calculate the phase factors Ψ corresponding to a target polynomial of degree d. Many of the aforementioned applications are formulated as the evaluation of a matrix function $f(H)$, where $f : [-1, 1] \to \mathbb{R}$ is not a polynomial but a smooth function, which can be expressed as an infinite polynomial series (e.g., the Chebyshev polynomial series). To approximate $f(H)$, we need to first truncate the polynomial series to $f^{(d)}$ with a proper degree d so that the difference between $f^{(d)}$ and f is sufficiently small. Then for each $f^{(d)}$ we can find (at least) one set of phase factors $\Psi^{(d)}$. When d is fixed, there has been significant progresses in computing the phase factors in the past few years [67, 73, 33, 53, 156]. The questions we would like to answer in this section are as follows.

1. As $d \to \infty$, can the phase factors $\{\Psi^{(d)}\}$ be chosen to have a well-defined limit $\Psi^\star$ in a properly chosen space?

2. If f is smooth, its Chebyshev coefficients decay rapidly. Does the tail of $\Psi^\star$ exhibit decay properties? If so, how is it related to the smoothness of f?

Our goal is to conceptually generalize QSP to represent smooth functions with a set of infinitely long phase factors, and we dub the resulting limit *infinite quantum signal processing* (iQSP). This section is summarized from our work in Ref. [55].

[1] The term "signal processing" is due to an analogy to digital filter designs on classical computers.

Setup of the problem

Let us revisit the *reduced phase factors* introduced earlier:

$$\Phi = (\phi_0, \phi_1, \ldots, \phi_{\widetilde{d}-1}) := \begin{cases} (\frac{1}{2}\psi_{\widetilde{d}-1}, \psi_{\widetilde{d}}, \cdots, \psi_d), & d \text{ is even}, \\ (\psi_{\widetilde{d}}, \psi_{\widetilde{d}-1}, \cdots, \psi_d), & d \text{ is odd}, \end{cases} \tag{5.22}$$

where $\Psi \in \mathbb{R}^{d+1}$ is a full set of symmetric phase factors.

The number of reduced phase factors is equal to $\widetilde{d} = \lceil \frac{d+1}{2} \rceil$, and matches the number of degrees of freedom in f. With some abuse of notation, we identify $U(x, \Phi)$ with $U(x, \Psi)$ and $g(x, \Phi)$ with $g(x, \Psi)$, and Φ is always referred to as the *reduced phase factors* of a full set of phase factors Ψ. For a given target polynomial, the existence of the symmetric phase factors is proved in [153, Theorem 1], but the choice is still not unique. However, near the trivial phase factors $\Phi = (0, \ldots, 0)$, there exists a unique and consistent choice of symmetric phase factors called the maximal solution [153].

Let ℓ^1 denote the set of all infinite dimensional vectors with finite 1-norm:

$$\ell^1 := \{v = (v_0, v_1, \cdots) : \|v\|_1 < \infty\}, \quad \|v\|_1 := \sum_{k=0}^{\infty} |v_k|, \quad v = (v_0, v_1, \cdots). \tag{5.23}$$

The vector space ℓ^1 is complete, i.e., every Cauchy sequence of points in ℓ^1 has a limit that is also in ℓ^1. Let $\mathbb{R}^\infty$ be the set of all infinite dimensional vectors with only a finite number of nonzero elements.

Definition 5.3.1 (Target function)**.** *A target function $f : \mathbb{R} \to \mathbb{R}$ is an infinite Chebyshev polynomial series with a definite parity*

$$f(x) = \begin{cases} \sum_{j=0}^{\infty} c_j T_{2j}(x), & f \text{ is even}, \\ \sum_{j=0}^{\infty} c_j T_{2j+1}(x), & f \text{ is odd}, \end{cases} \tag{5.24}$$

The coefficient vector $c = (c_0, c_1, \ldots) \in \ell^1$, and f satisfies the norm constraint

$$\|f\|_\infty = \max_{x \in [-1,1]} |f(x)| \leq 1. \tag{5.25}$$

In other words, the set of even target functions is

$$S_e = \left\{ f : [-1,1] \to [-1,1] : f(x) = \sum_{j=0}^{\infty} c_j T_{2j}(x), \quad \sum_{j=0}^{\infty} |c_j| < \infty \right\}, \tag{5.26}$$

and the set of odd target functions is

$$S_o = \left\{ f : [-1,1] \to [-1,1] : f(x) = \sum_{j=0}^{\infty} c_j T_{2j+1}(x), \quad \sum_{j=0}^{\infty} |c_j| < \infty \right\}. \tag{5.27}$$

If we truncate the Chebyshev coefficients to be $c^{(\widetilde{d})} = (c_0, c_1, \ldots, c_{\widetilde{d}}, 0, \ldots) \in \mathbb{R}^\infty$, the corresponding Chebyshev polynomial $f^{(d)}$ is of degree d (recall that $\widetilde{d} = \lceil \frac{d+1}{2} \rceil$ and hence d is determined by $\widetilde{d}$ and the parity of the function). Furthermore, $c \in \ell^1$ implies that $\lim_{d\to\infty} \left\| f^{(d)} - f \right\|_\infty = 0$. Throughout the paper, $f^{(d)}$ will be referred to as a *target polynomial* approximating the target function f.

In order to compare phase factors of different lengths, an important observation is that if we pad $\Phi = (\phi_0, \phi_1, \ldots, \phi_{\widetilde{d}})$ with an arbitrary number of 0's at the right end and obtain $\widetilde{\Phi} = (\phi_0, \phi_1, \ldots, \phi_{\widetilde{d}}, 0, \ldots, 0)$, we have $g(x, \Phi) = g(x, \widetilde{\Phi})$ (see [55, Lemma 9]). Therefore $g(x, \cdot)$ is a well defined mapping in $\mathbb{R}^\infty$, and we can identify Φ with $\widetilde{\Phi}$. Let $\mathcal{F}$ be the linear mapping from a target polynomial to its Chebyshev-coefficient vector $c \in \mathbb{R}^\infty$ as defined in Eq. (5.24). This induces a mapping

$$F : \mathbb{R}^\infty \to \mathbb{R}^\infty, \quad F(\Phi) := \mathcal{F}(g(x, \Phi)), \tag{5.28}$$

which maps the reduced phase factors $\Phi \in \mathbb{R}^\infty$ to the Chebyshev coefficients of $g(x, \Phi)$.

Note that $\mathbb{R}^\infty$ is dense in ℓ^1, i.e., any point in ℓ^1 is either a point in $\mathbb{R}^\infty$ or a limit point of $\mathbb{R}^\infty$. By exploiting some nice properties of F, we can define $\overline{F} : \ell^1 \to \ell^1$ to be the extension of F, such that $\overline{F}(\Phi)$ agrees with $F(\Phi)$ for any $\Phi \in \mathbb{R}^\infty$. Then the problem of infinite quantum signal processing asks whether the inverse of the mapping $\overline{F}$ exists.

Problem 5.3.2 (Infinite quantum signal processing)**.** *For a target function in Definition 5.3.1 given by its Chebyshev coefficients $c \in \ell^1$, does it exist $\Phi^\star \in \ell^1$ such that $\overline{F}(\Phi^\star) = c$?*

Main results

Theorem 5.3.3 (Invertibility of $\overline{F}$)**.** *There exists a universal constant $r_c \approx 0.902$, so that $\overline{F}$ has an inverse map $\overline{F}^{-1} : B(0, r_c) \subset \ell^1 \to \ell^1$, where $B(a, r) := \{v \in \ell^1 : \|v - a\|_1 < r\}$.*

Theorem 5.3.3 provides a positive answer to Problem 5.3.2 as well as to the first question raised at the beginning of this section, when the 1-norm of the Chebyshev

coefficients is upper bounded by a constant. Note that for a given target function f, we can always multiply it by a constant C, so that the Cf satisfies the condition of Theorem 5.3.3. The main technical tools are a series of vector 1-norm estimates of F, and matrix 1-norm estimates of the Jacobian matrix DF. These estimates do not explicitly depend on the length of phase factors, and can therefore be extended to $\overline{F}$. A more detailed statement of Theorem 5.3.3 is [55, Theorem 22], which will be presented in [55, Section 3.4].

Since $\Phi^\star = (\phi_0, \phi_1, \ldots) \in \ell^1$, the tail of $\Phi^\star$ must exhibit decay properties, i.e., $\lim_{n\to\infty} \sum_{k>n} |\phi_k| = 0$. Fig. 5.6 shows that the tail decay of $\Phi^\star$ closely matches that of the Chebyshev coefficients c. The duality between the smoothness of a function and the decay of its Fourier / Chebyshev coefficients is well studied (see e.g. [144, Chapter 7]). But it is surprising that the tail decay of the reduced phase factors can be directly related to the smoothness of the target function. Such a behavior was first numerically observed in Ref. [53], in which an explanation of the phenomenon was also given in the perturbative regime. Using the tools developed in proving Theorem 5.3.3, we provide a refined and non-perturbative analysis of the tail decay in Theorem 5.3.4.

Theorem 5.3.4 (Decay properties of reduced phase factors)**.** *Given a target function f with $\|c\|_1 < r_c$, and $\Phi^\star := \overline{F}^{-1}(c) = (\phi_0, \phi_1, \ldots) \in \ell^1$, then there exists a constant C such that for any n,*

$$\sum_{k>n} |\phi_k| \leq C \sum_{k>n} |c_k| \,. \tag{5.29}$$

The proof is given in [55, Section 4] with an explicit characterization of the constant C. Assume the target function f is of C^α smoothness for some $\alpha > 0$, then the Chebyshev coefficients decay algebraically in the sense of $\sum_{k>n} |c_k| = \mathcal{O}\left(n^{-\alpha}\right)$. Then, it induces an decay of the corresponding reduced phase factors, namely, $\sum_{k>n} |\phi_k| = \mathcal{O}\left(n^{-\alpha}\right)$. If f is of C^∞ or C^ω smoothness, then the tail of Chebyshev-coefficient vector decays super-algebraically or exponentially respectively, and so does the tail of the reduced factors. These results are also verified numerically in Fig. 5.6. These results provide a positive answer to the second question raised at the beginning of this section.

Theorem 5.3.3 also has algorithmic implications. It has been empirically observed that a quasi-Newton optimization based algorithm is highly effective for finding the phase factors [53]. However, the theoretical justification of the optimization based algorithm has only been shown if the target function satisfies $\|f\|_\infty \leq C_f \widetilde{d}^{-1}$, where $C_f \approx 0.028$ is a universal constant [153, Corollary 7]. Hence as $\widetilde{d}$ increases, existing theoretical results fail to predict the effectiveness of the algorithm, even if the target

function is a fixed polynomial of finite degree (in this case, we pad the Chebyshev coefficients with zeros to increase $\widetilde{d}$).

Numerical justification

We demonstrate the decay of phase factors in Fig. 5.6. In the first example, we truncate the series expansion of $f(x) = 0.8\,|x|^3$ in terms of Chebyshev polynomials of the first kind up to degree $d = 1000$ and use the FPI method in Algorithm 4.5.1 to find the corresponding reduced phase factors. The third order derivative of $f(x)$ is discontinuous. In Fig. 5.6(a), we plot the magnitude of its Chebyshev-coefficient vector, as well as the reduced phase factors obtained by Algorithm 4.5.1. Here, the 1-norm of Chebyshev coefficients is about 0.8149, which is bounded by r_c. Fig. 5.6(a) shows that the reduced phase factors decays away from the center with an algebraic decay rate around 4, which matches the decay rate of Chebyshev coefficients. This also agrees with our theoretical results in Theorem 5.3.4. In the second example, we choose the truncated Jacobi-Anger series of sine function as target polynomial and present the magnitude of both Chebyshev-coefficient vector and the corresponding reduced phase factors in Fig. 5.6(b). The 1-norm of Chebyshev coefficients is around 3.2332, which exceeds the norm condition in Theorem 5.3.4. Nonetheless, the decay of the tail of the phase factors closely match that of the Chebyshev-coefficient vector.

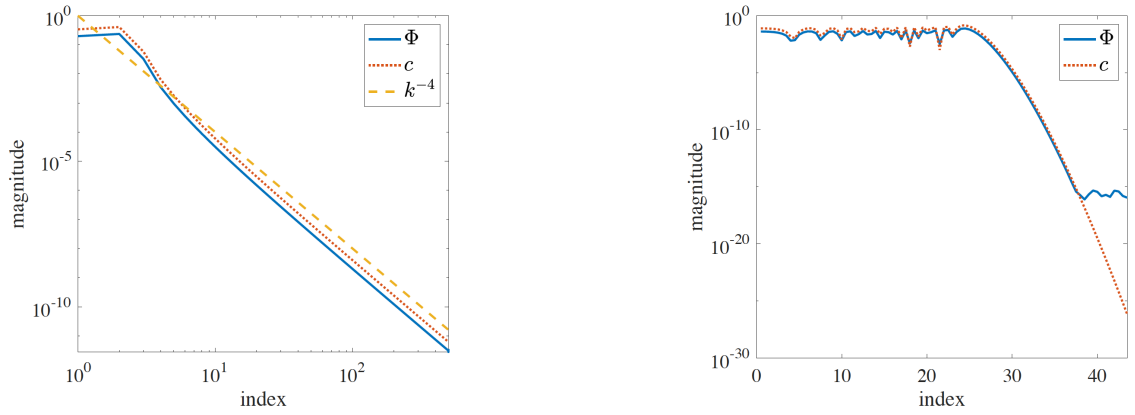


Figure 5.6: Magnitude of the Chebyshev-coefficient vector c and the corresponding reduced phase factors Φ. (a) The target polynomial is $\sum_{k=0}^{1000} c_k T_{2k}(x)$, where c_k is the Chebyshev coefficient of $0.8\,|x|^3$ w.r.t. T_{2k}. The slopes of blue and red curves are about -4, representing $|\phi_k| \approx \text{const} \cdot k^{-4}$ and $|c_k| \approx \text{const} \cdot k^{-4}$. (b) The target polynomial is a degree-173 truncated Jacobi-Anger series of $\sin(100x)$.

Chapter 6

Ground-state preparation and energy estimation on early fault-tolerant quantum computers

Under suitable assumptions, the algorithms in [94] can estimate the ground-state energy and prepare the ground state of a quantum Hamiltonian with near-optimal query complexities. However, this is based on a block encoding input model of the Hamiltonian, whose implementation is known to require a large resource overhead. We develop a tool called quantum eigenvalue transformation of unitary matrices with real polynomials (QETU), which uses a controlled Hamiltonian evolution as the input model, a single ancilla qubit and no multi-qubit control operations, and is thus suitable for early fault-tolerant quantum devices. This leads to a simple quantum algorithm that outperforms all previous algorithms with a comparable circuit structure for estimating the ground-state energy. For a class of quantum spin Hamiltonians, we propose a new method that exploits certain anti-commutation relations and further removes the need of implementing the controlled Hamiltonian evolution. Coupled with a Trotter-based approximation of the Hamiltonian evolution, the resulting algorithm can be very suitable for early fault-tolerant quantum devices. We demonstrate the performance of the algorithm using IBM Qiskit for the transverse field Ising model. If we are further allowed to use multi-qubit Toffoli gates, we can then implement amplitude amplification and a new binary amplitude estimation algorithm, which increases the circuit depth but decreases the total query complexity. The resulting algorithm saturates the near-optimal complexity for ground-state preparation and energy estimating using a constant number of ancilla qubits (no more than 3).

Please note that this chapter is based on [51] (joint work with Lin Lin and Yu

Tong).

6.1 Introduction

Preparing the ground state and estimating the ground-state energy of a quantum Hamiltonian have a wide range of applications in condensed matter physics, quantum chemistry, and quantum information. To solve such problems, quantum computers promise to deliver a new level of computational power that can be significantly beyond the boundaries set by classical computers. Despite exciting early progress on NISQ devices [124], it is widely believed that most scientific advances in quantum sciences require some version of fault-tolerant quantum computers, which are expected to be able to accomplish much more complicated tasks. On the other hand, the fabrication of full-scale fault-tolerant quantum computers remains a formidable technical challenge for the foreseeable future, and it is reasonable to expect that early fault-tolerant quantum computers share the following characteristics: (1) The number of logical qubits is limited. (2) It can be difficult to execute certain controlled operations (e.g., multi-qubit control gates), whose implementation require a large number of non-Clifford gates. Besides these, the maximum circuit depth of early-fault-tolerant quantum computers, which is determined by the maximum coherence time of the devices, may still be limited. Therefore it is still important to reduce the circuit depth, sometimes even at the expense of a larger total runtime (via a larger number of repetitions). Quantum algorithms tailored for early fault-tolerant quantum computers [31, 10, 24, 89, 93, 149, 158, 151] need to properly take these limitations into account, and the resulting algorithmic structure can be different from those designed for fully fault-tolerant quantum computers.

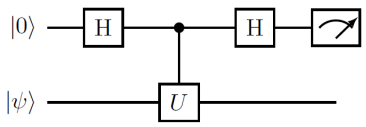
To gain access to the quantum Hamiltonian H, a standard input model is the *block encoding* (BE) model, which directly encodes the matrix H (after proper rescaling) as a submatrix block of a larger unitary matrix U_H [99, 32]. Combined with techniques such as linear combination of unitaries (LCU) [14], quantum signal processing [101] or quantum singular value transformation [67], one can implement a large class of matrix functions of H on a quantum computer. This leads to quantum algorithms for ground-state preparation and ground-state energy estimation with near-optimal query complexities to U_H [94]. The block encoding technique is also very useful in many other tasks such as Hamiltonian simulation, solving linear systems, preparing the Gibbs state, and computing Green's function and the correlation functions [143, 32, 128, 66, 101]. However, the block encoding of a quantum Hamiltonian (e.g., a sparse matrix) often involves a relatively large number of ancilla qubits, as well as multi-qubit controlled operations that lead to a large number of two-qubit gates

and long circuit depths [67], and is therefore not suitable in the early fault-tolerant setting.

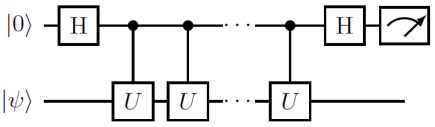
A widely used alternative approach for accessing the information in H is the time evolution operator $U = \exp(-i\tau H)$ for some time τ. This input model will be referred to as the *Hamiltonian evolution* (HE) model. While Hamiltonian simulation can be performed using quantum signal processing for sparse Hamiltonians with optimal query complexity [101], such an algorithm queries a block encoding of H, which defeats the purpose of employing the HE model. On the other hand, when H can be efficiently decomposed into a linear combination of Pauli operators, the time evolution operator can be efficiently implemented using, e.g., the Trotter product formula [96, 40] without using any ancilla qubit. This remarkable feature has inspired quantum algorithms for performing a variety of tasks using controlled time evolution and one ancilla qubit. A textbook example of such an algorithm is the Hadamard test. It uses one ancilla qubit and the controlled Hamiltonian evolution to estimate the average value $\mathrm{Re}\,\langle\psi|U|\psi\rangle$, which is encoded by the probability of measuring the ancilla qubit with outcome 0 (see Fig. 6.1 (a)). The number of repeated measurements of this procedure is $\mathcal{O}(\epsilon^{-2})$, where ϵ is the desired precision. Assume the spectrum of the Hamiltonian H is contained in $[\eta, \pi - \eta]$ for some $\eta > 0$. If $|\psi\rangle$ is the exact ground state of H, we can retrieve the eigenvalue as $\lambda = \arccos(\mathrm{Re}\,\langle\psi|U|\psi\rangle)$. By allowing a series of longer simulation times of $t = d$ for some integer d, this leads to Kitaev's algorithm that uses only $\log \epsilon^{-1}$ measurements, at the expense of increasing the circuit depth to $\mathcal{O}(\epsilon^{-1})$ (see Fig. 6.1 (b)). The total simulation time is therefore $\mathcal{O}(\epsilon^{-1} \log \epsilon^{-1})$, which reaches the Heisenberg limit [69, 68, 160, 159] up to a logarithmic factor.

When the input state $|\phi_0\rangle$ (prepared by an oracle U_I) is different from the exact ground state of H denoted by $|\psi_0\rangle$, there have been multiple quantum algorithms using the circuit of Fig. 6.1 (b) or its variant to estimate the ground-state energy [150, 155, 116, 93]. Let γ be a lower bound of the initial overlap, i.e., $|\langle\phi_0|\psi_0\rangle| \geq \gamma$. It is worth noting that *all* quantum algorithms with provable performance guarantees require *a priori* knowledge that γ is reasonably large (assuming black-box access to the Hamiltonian). Without such an assumption, this problem is QMA-hard [86, 84, 119, 6]. Candidates for such $|\phi_0\rangle$ include the Hartree-Fock state in quantum chemistry [87, 145], and quantum states prepared using the variational quantum eigensolver [121, 109, 118]. Some techniques can be used to boost the overlap using low-depth circuits [151]. Furthermore, algorithms using the circuit of Fig. 6.1 (b) typically cannot be used to prepare the ground state. With the time evolution operator as the input, one can use the LCU algorithm to prepare the ground state [62, 82], thus reducing the number of ancilla qubits needed to implement the block encoding of the Hamiltonian. Note that LCU requires additional ancilla qubits to

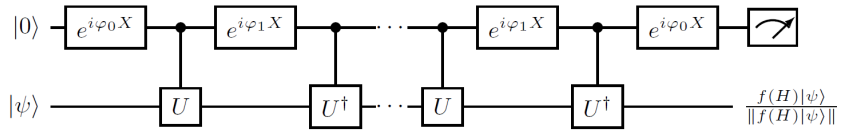
store the coefficients, and as a result cannot be implemented using $\mathcal{O}(1)$ qubits.



(a) Hadamard test, short time evolution



(b) Hadamard test, long time evolution



(c) QETU

Figure 6.1: After a proper rescaling, the n-qubit circuit U implements e^{-iH}, and Hadamard test circuit (a) estimates $\mathrm{Re}\,\langle\psi|e^{-iH}|\psi\rangle$. Repeating the controlled evolution d times, the circuit (b) estimates the average value of a long time evolution $\mathrm{Re}\,\langle\psi|e^{-idH}|\psi\rangle$. For a very general class of functions f the circuit (c) can approximately prepare a normalized quantum state $f(H)\,|\psi\rangle\,/\,\|f(H)\,|\psi\rangle\|$ with approximate success probability $p = \|f(H)\,|\psi\rangle\|^2$, by interleaving the forward (U) and backward ($U^\dagger$) time evolution with some properly chosen X-rotations in the ancilla qubit.

We will show that both the ground-state preparation and energy estimation can be solved by (repeatedly) preparing a quantum state of the form $|\psi_f\rangle \propto f(H)\,|\phi_0\rangle$, where f is a real polynomial approximating a shifted sign function. A main technical tool developed in this this chapter is called *quantum eigenvalue transformation of unitary matrices with real polynomials* (QETU), which allows us to prepare such a state $|\psi_f\rangle$ by querying $U = e^{-iH}$, using only one ancilla qubit, and does not require any multi-qubit control operation (Theorem 6.2.1). The circuit structure (Fig. 6.1 (c)) is only slightly different from that in Fig 6.1 (b). The QETU technique is closely related to concepts such as quantum signal processing, quantum eigenvalue transformation, and quantum singular value transformation. The relations among these techniques are detailed in Section 2.1. The information of the function f of interest is stored in the adjustable parameters $\{\varphi_i\}$ called *phase factors*. To find such parameters, we need to identify a polynomial approximation to the shifted sign function, and then evaluate the phase factors corresponding to the approximate polynomial. Most quantum signal processing (QSP) based applications construct such a polynomial approximation analytically, which can sometimes lead

to cumbersome expressions and suboptimal approximation results. We provide a convex-optimization-based procedure to streamline this process and yield the near-optimal approximation (see Section 6.4). Both the QETU technique and the convex optimization method can be useful in applications beyond ground-state preparation and energy estimation.

The computational cost will be primarily measured in terms of the query complexity, i.e., how many times we need to query U and U_I in total, and we will also analyze the additional one- and two-qubit gates needed, such as the single qubit rotation gates, and the two-qubit gates needed to implement the n-qubit reflection operator. In our algorithms, the number of additional gates has the same scaling as the query complexity, or involves an n factor where n is system size. We measure the circuit depth requirement in terms of *query depth*: the number of times we need to query U in one coherent run of the circuit. Note that this term is not to be confused with the circuit depth for implementing the oracle U, which we will not consider in this work. This metric reflects the circuit depth requirement faithfully because in our algorithm, in one coherent run of the circuit, the number of queries to U_I is also upper bounded by this metric, and the additional circuit depth needed for additional gates is upper bounded by this metric up to a factor of $\mathcal{O}(n)$.

Besides the query depth, we also focus on whether multi-qubit control needs to be implemented. Algorithms such as amplitude amplification and amplitude estimation [25] can be used to reduce the total query complexity, but they also need to use $(n+1)$-bit Toffoli gates (specifically, n-qubit reflection operator with respect to the zero state $|0^n\rangle$), which can be implemented using $\mathcal{O}(n)$ two-qubit gates and one ancilla qubit [11]. These operations will be referred to as "low-level" multi-qubit control gates. Some other quantum algorithms may require more complex multi-qubit control operations as well as more ancilla qubits. For instance, the high-confidence QPE algorithm [88, 123, 113] requires a circuit to carry out the arithmetic operation of taking the median of multiple energy measurement results, which can require poly(n) two-qubit gates and ancilla qubits. Such operations will be referred to as "high-level" multi-qubit control gates.

To solve the ground-state preparation and energy estimation problem, we propose two different types of algorithms, with two different goals in mind. For the first type of algorithms, which we call the *short query depth algorithms*, we only use QETU to prioritize reducing the quantum resources needed. No multi-qubit controlled operation is involved. For the second type of algorithms, which we call the *near-optimal algorithms*, we optimize the total query complexity by using amplitude amplification and a new binary amplitude estimation algorithm (Lemma 6.3.11). Such algorithms only use low-level multi-qubit control operations. Both types of algorithms only use a small number of ancilla qubits (no more than 2 or 3).

For ground-state energy estimation, surprisingly, even though the total query complexity of the short query depth algorithm does not have the optimal asymptotic scaling, it still outperforms *all* previous algorithms with the same ancilla qubit number constraint [78, 15, 135, 93], in terms of total query complexity (see Table 6.1). Most notably, in this setting we achieve a quadratic improvement on the γ dependence, from $\widetilde{\mathcal{O}}(\gamma^{-4})$ to $\widetilde{\mathcal{O}}(\gamma^{-2})$ (the notation $\widetilde{\mathcal{O}}(g)$ means $\mathcal{O}(g \operatorname{poly}\log(g))$ unless otherwise stated). Moreover the circuit depth from the previous state-of-the-art result is preserved in our algorithm. Numerical comparison (see Fig. 6.4) demonstrates that our algorithm outperforms QPE, not only in terms of the asymptotic scaling, but also the exact non-asymptotic number of queries for moderately small values of γ. Our near-optimal algorithm takes this advantage even further, matching the best known query complexity scaling in Ref. [94] (which saturates the query complexity lower bound).

For ground-state preparation, the only other algorithm that can use at most constantly many ancilla qubits is the quantum phase estimation algorithm with semi-classical Fourier transform [71]. Compared to this algorithm, our short query depth algorithm has an exponentially improved precision dependence and a quadratically improved γ dependence, from $\widetilde{\mathcal{O}}(\gamma^{-4})$ to $\widetilde{\mathcal{O}}(\gamma^{-2})$, while maintaining the same circuit depth. The near-optimal algorithm further improves the dependence to $\widetilde{\mathcal{O}}(\gamma^{-1})$. A comparison of the algorithms for ground-state preparation can be found in Table 6.2. We remark that here we consider the case where we know a parameter μ such that $\lambda_0 \leq \mu - \Delta/2 < \mu + \Delta/2 \leq \lambda_1$, as in Theorem 6.3.5. If no such μ is known, we need to first estimate the ground-state energy to precision $\mathcal{O}(\Delta)$, and the resulting algorithm is discussed in Theorem 6.3.12. If the ground-state energy is known *a priori*, then the algorithm in [44] may yield a similar speedup for preparing the ground state, but such knowledge is generally not available.

In the above analysis, specifically in Tables 6.1 and 6.2, we compared with algorithms whose complexity can be rigorously analyzed under the assumptions of a good initial overlap (and spectral gap for ground-state preparation). We did not compare with heuristic algorithms such as the variational quantum eigensolver [121, 109, 118]. There are also algorithms that are designed with different but similar goals in mind, such as the quantum algorithmic cooling technique in Ref. [157], which can estimate an eigenvalue λ_j belonging to a given range $[\lambda_j^L, \lambda_j^R]$ (assuming all other eigenvalues are away from this range). Then the total runtime scaling is $\widetilde{\mathcal{O}}(\gamma^{-4})$ where γ is the overlap between the initial guess and the target eigenstate [157, Theorem 2]. The same technique can also be used to estimate the observable expectation value of the target eigenstate, without coherently preparing the target eigenstate.

For certain Hamiltonians, QETU can be implemented with the standard Hamil-

tonian evolution rather than the controlled version. Note that "control-free" only means that the Hamiltonian evolution is not controlled by one or more qubits, but control gates that are independent of the Hamiltonian can still be used. In Ref. [80], the control-free setting for an n-qubit time evolution is achieved by introducing an n-qubit reference state, on which the time evolution acts trivially. The algorithm also requires the implementation of the controlled n-qubit SWAP gate, and therefore has a relatively large overhead. There are other control-free algorithms proposed in Refs. [102, 117, 93] for energy and phase estimation via the measurement of certain scalar expectation values as the output. In particular, such algorithms cannot coherently implement a controlled time evolution and are therefore not compatible with the implementation of QETU. In this chapter, we exploit certain anti-commutation relations and structures of the Hamiltonian to propose a new control-free implementation. In the context of QETU, the algorithm does not introduce any ancilla qubit and requires a small number of two-qubit gates that scales linearly in n. We demonstrate the optimized circuit implementation of the transverse field Ising model under the control-free setting. To the extent of our knowledge, this circuit is significantly simpler than all previous QSP-type circuits for simulating a physical Hamiltonian. We show the numerical performance of our algorithm for estimating the ground energy estimation in the presence of tunable quantum error using IBM Qiskit.

6.2 Quantum eigenvalue transformation of unitary matrices

Given the Hamiltonian evolution input model $U = e^{-iH}$, we first demonstrate that by slightly modifying the circuit for the Hadamard test in Fig. 6.1 (b), we can approximately prepare a target state $|\psi_f\rangle = f(H)|\psi\rangle / \|f(H)|\psi\rangle\|$ efficiently and with controlled accuracy for a large class of real functions f. Specifically, this requires alternately applying the controlled forward time evolution operator U, a single qubit X rotation in the ancilla qubit, and the controlled backward time evolution operator $U^\dagger$ (see Fig. 6.1 (c)). This circuit does not store the eigenvalues of H either in a classical or a quantum register, and the information of the function f of interest is entirely stored in the adjustable parameters $\varphi_0, \varphi_1, \varphi_2, \ldots, \varphi_{d/2}$. These parameters form a set of symmetric phase factors $(\varphi_0, \varphi_1, \varphi_2, \ldots, \varphi_2, \varphi_1, \varphi_0) \in \mathbb{R}^{d+1}$ used in the circuit Fig. 6.1 (c). The symmetry of the phase factors is the key to attaining the reality of the function f of interest.

Theorem 6.2.1 (QETU)**.** *Let $U = e^{-iH}$ with an n-qubit Hermitian matrix H. For any even real polynomial $F(x)$ of degree d satisfying $|F(x)| \leq 1, \forall x \in [-1, 1]$, we can*

	Query depth	Query complexity	# ancilla qubits	Need MQC?	Input model
This work (Theorem 6.3.3)	$\widetilde{\mathcal{O}}(\epsilon^{-1})$	$\widetilde{\mathcal{O}}(\epsilon^{-1}\gamma^{-2})$	$\mathcal{O}(1)$	No	HE
This work (Theorem 6.3.4)	$\widetilde{\mathcal{O}}(\epsilon^{-1}\gamma^{-1})$	$\widetilde{\mathcal{O}}(\epsilon^{-1}\gamma^{-1})$	$\mathcal{O}(1)$	Low	HE
QPE (high confidence) [88, 123, 113]	$\widetilde{\mathcal{O}}(\epsilon^{-1})$	$\widetilde{\mathcal{O}}(\epsilon^{-1}\gamma^{-2})$	$\mathcal{O}(\mathrm{poly}\log(\gamma^{-1}\epsilon^{-1}))$	High	HE
QPE (semi-classical) [78, 15]	$\widetilde{\mathcal{O}}(\epsilon^{-1}\gamma^{-2})$	$\widetilde{\mathcal{O}}(\epsilon^{-1}\gamma^{-4})$	$\mathcal{O}(1)$	No	HE
QEEA [135, 93]	$\widetilde{\mathcal{O}}(\epsilon^{-1})$	$\widetilde{\mathcal{O}}(\epsilon^{-4}\gamma^{-4})$	$\mathcal{O}(1)$	No	HE
GTC19 (Theorem 4) [62]	$\widetilde{\mathcal{O}}(\epsilon^{-3/2}\gamma^{-1})$	$\widetilde{\mathcal{O}}(\epsilon^{-3/2}\gamma^{-1})$	$\mathcal{O}(\log(\epsilon^{-1}))$	High	HE
LT20 [94]	$\widetilde{\mathcal{O}}(\epsilon^{-1}\gamma^{-1})$	$\widetilde{\mathcal{O}}(\epsilon^{-1}\gamma^{-1})$	$m+\mathcal{O}(\log(\epsilon^{-1}))$	High	BE
LT22 [93]	$\widetilde{\mathcal{O}}(\epsilon^{-1})$	$\widetilde{\mathcal{O}}(\epsilon^{-1}\gamma^{-4})$	$\mathcal{O}(1)$	No	HE

Table 6.1: Comparison of the performance of quantum algorithms for ground-state energy estimation in terms of the query complexity, query depth, number of ancilla qubits, and the level of multi-qubit control (abbreviated as "MQC" in the table) operation is needed. γ is the overlap between the initial guess $|\phi_0\rangle$ and the ground state, and ϵ is the allowed error. "HE" stands for the Hamiltonian evolution model (assuming no ancilla qubits), and "BE" for the block encoding model. The sharper estimate for estimating the ground-state energy using the quantum eigenvalue estimation algorithm (QEEA) is given in [93, Appendix C]. We assume that Ref. [94] uses m ancilla qubits, and the high-level MQC operation is due to the block encoding of H.

find a sequence of symmetric phase factors $\Phi_z := (\varphi_0, \varphi_1, \cdots, \varphi_1, \varphi_0) \in \mathbb{R}^{d+1}$, *such that the circuit in Fig. 6.1 (c) denoted by* $\mathcal{U}$ *satisfies* $(\langle 0|\otimes I_n)\mathcal{U}(|0\rangle\otimes I_n) = F\left(\cos\frac{H}{2}\right)$.

The proof of Theorem 6.2.1 is given in Section 2.5. It is worth mentioning that the concept of "qubitization" [99, 67] appears very straightforwardly in QETU. Let the matrix function of interest be expressed as $f(H) = (f\circ g)(\cos\frac{H}{2})$, where $g(x) = 2\arccos(x)$. Therefore we can find a polynomial approximation $F(x)$ so that

$$\sup_{x\in[\sigma_{\min},\sigma_{\max}]} |(f\circ g)(x) - F(x)| \leq \epsilon. \tag{6.1}$$

Here $\sigma_{\min} = \cos\frac{\lambda_{\max}}{2}, \sigma_{\max} = \cos\frac{\lambda_{\min}}{2}$, respectively (note that $\cos(x/2)$ is a monotonically *decreasing* function on $[0,\pi]$). This ensures that the operator norm error satisfies

$$\|(\langle 0|\otimes I_n)\mathcal{U}(|0\rangle\otimes I_n) - f(H)\| \leq \epsilon.$$

	Query depth	Query complexity	# ancilla qubits	Need MQC?	Input model
This work (Theorem 6.3.5)	$\widetilde{\mathcal{O}}(\Delta^{-1})$	$\widetilde{\mathcal{O}}(\Delta^{-1}\gamma^{-2})$	$\mathcal{O}(1)$	No	HE
This work (Theorem 6.3.10)	$\widetilde{\mathcal{O}}(\Delta^{-1}\gamma^{-1})$	$\widetilde{\mathcal{O}}(\Delta^{-1}\gamma^{-1})$	$\mathcal{O}(1)$	Low	HE
QPE (high confidence) [88, 123, 113]	$\widetilde{\mathcal{O}}(\Delta^{-1})$	$\widetilde{\mathcal{O}}(\Delta^{-1}\gamma^{-2})$	$\mathcal{O}(\mathrm{poly}\log(\Delta^{-1}\gamma^{-1}\epsilon^{-1}))$	High	HE
QPE (semi-classical) [78, 15]	$\widetilde{\mathcal{O}}(\Delta^{-1}\gamma^{-2})$	$\widetilde{\mathcal{O}}(\Delta^{-1}\gamma^{-4})$	$\mathcal{O}(1)$	No	HE
GTC19 (Theorem 1) [62]	$\widetilde{\mathcal{O}}(\Delta^{-1}\gamma^{-1})$	$\widetilde{\mathcal{O}}(\Delta^{-1}\gamma^{-1})$	$\mathcal{O}(\log(\Delta^{-1}) + \log\log(\epsilon^{-1}))$	High	HE
LT20 [94]	$\widetilde{\mathcal{O}}(\Delta^{-1}\gamma^{-1})$	$\widetilde{\mathcal{O}}(\Delta^{-1}\gamma^{-1})$	m	High	BE

Table 6.2: Comparison of the performance of quantum algorithms for ground-state preparation in terms of the query complexity, query depth, number of ancilla qubits, and the level of multi-qubit control (abbreviated as "MQC" in the table) operation is needed. γ is the overlap between the initial guess $|\phi_0\rangle$ and the ground state, Δ is a lower bound of the spectral gap, and $1-\epsilon$ is the target fidelity. "HE" stands for the Hamiltonian evolution model (assuming no ancilla qubits), and "BE" the block encoding model. Here we assume that an upper bound of the ground-state energy is known (μ in Theorem 6.3.5). The algorithm in Ref. [62] (GTC19 in the table) requires precise knowledge of the ground-state energy. We assume that Ref. [94] uses m ancilla qubits, and the high-level MQC operation is due to the block encoding of H.

The implementation of $U = e^{-iH}$ corresponds to a Hamiltonian simulation problem of H at time $t = 1$. In practice, we can use the Trotter decomposition to obtain an approximate implementation of U without ancilla qubits, i.e., we can partition the time interval into r steps with $\tau = r^{-1}$ and use a low order Trotter method to implement an approximation to $U_\tau \approx e^{-iH\tau}$. Then

$$U = e^{-iH} \approx (U_\tau)^r. \tag{6.2}$$

In line with other works in analyzing the performance of quantum algorithms using the HE input model [78, 15, 135, 93], in the discussion below, unless otherwise specified, we assume U is implemented exactly, and the errors are due to other sources such as polynomial approximation, the binary search process etc. We refer readers to Section 6.9 for the complexity analysis of QETU when U is implemented using a p-th order Trotter formula, as well as its implication in the ground-state energy estimation.

6.3 Ground-state energy estimation and ground-state preparation

In this section we discuss how to estimate the ground-state energy and to prepare the ground state within the QETU framework. The setup of the problems is as follows: we assume that the Hamiltonian H can be accessed through its time-evolution operator e^{-iH}. The goal is (1) to estimate the ground-state energy, and (2) to prepare the ground state. For the first task we assume that we have access to a good initial guess $|\phi_0\rangle$ of the ground state, i.e., $|\langle\phi_0|\psi_0\rangle| \geq \gamma$ where $|\psi_0\rangle$ is the ground state. For the second task, we need the additional assumption that the ground-state energy λ_0 is separated from the rest of the spectrum by a gap Δ. These assumptions are stated more formally in the definitions below.

Definition 6.3.1 (ground-state energy estimation)**.** *Suppose we are given a Hamiltonian H on n qubits whose spectrum is contained in $[\eta, \pi - \eta]$ for some $\eta > 0$. The Hamiltonian can be accessed through a unitary $U = e^{-iH}$. Also suppose we have an initial guess $|\phi_0\rangle$ of the ground state $|\psi_0\rangle$ satisfying $|\langle\phi_0|\psi_0\rangle| \geq \gamma$. This initial guess can be prepared by U_I. The oracles U and U_I are provided as black-box oracles. The goal is to estimate the ground-state energy λ_0 to within additive error ϵ.*

Definition 6.3.2 (ground-state preparation)**.** *Under the same assumptions as in Definition 6.3.1, and the additional assumption that there is a spectral gap at least Δ separating the ground-state energy λ_0 from the rest of the spectrum, the goal is to prepare a quantum state $|\widetilde{\psi}_0\rangle$ such that $|\langle\psi_0|\widetilde{\psi}_0\rangle| \geq 1 - \epsilon$.*

We will primarily focus on ground-state energy estimation. This is because once we have the ground-state energy, preparing the ground state can be done by applying an approximate projection, which can be directly performed using QETU. We will consider two settings: the short query depth setting and the near-optimal setting. In the first setting we prioritize lowering the query depth (and hence the circuit depth), and in the second setting we prioritize lowering the query complexity (and hence the total runtime). Our results for the two settings are stated in the following theorems:

Theorem 6.3.3 (ground-state energy estimation using QETU)**.** *Under the assumptions stated in Definition 6.3.1, we can estimate the ground-state energy to within additive error ϵ, with probability at least $1 - \vartheta$, with the following cost:*

1. *$\widetilde{\mathcal{O}}(\epsilon^{-1}\gamma^{-2}\log(\vartheta^{-1}))$ queries to (controlled-) U and $\mathcal{O}(\gamma^{-2}\operatorname{poly}\log(\epsilon^{-1}\vartheta^{-1}))$ queries to U_I.*

2. *One ancilla qubit.*

3. $\widetilde{\mathcal{O}}(\epsilon^{-1}\gamma^{-2}\log(\vartheta^{-1}))$ *additional one-qubit quantum gates.*

4. $\mathcal{O}(\epsilon^{-1}\log(\gamma^{-1}))$ *query depth of* U.

Note that here, using the short query depth algorithm, we do not need to use extra two-qubit gates beyond what is needed in (controlled-) U.

Theorem 6.3.4 (near-optimal ground-state energy estimation with QETU and improved binary amplitude estimation)**.** *Under the assumptions stated in Definition 6.3.1, we can estimate the ground-state energy to within additive error* ϵ*, with probability at least* $1-\vartheta$*, with the following cost:*

1. $\widetilde{\mathcal{O}}(\epsilon^{-1}\gamma^{-1}\log(\vartheta^{-1}))$ *queries to (controlled-)* U *and* $\mathcal{O}(\gamma^{-1}\operatorname{poly}\log(\epsilon^{-1}\vartheta^{-1}))$ *queries to* U_I.

2. *Three ancilla qubits.*

3. $\widetilde{\mathcal{O}}(n\gamma^{-1}\log(\epsilon^{-1}\vartheta^{-1})+\epsilon^{-1}\gamma^{-1}\log(\vartheta^{-1}))$ *additional one- and two-qubit quantum gates.*

4. $\widetilde{\mathcal{O}}(\epsilon^{-1}\gamma^{-1}\log(\vartheta^{-1}))$ *query depth of* U.

To the best of our knowledge, this is also the first algorithm that can estimate the ground-state energy with $\widetilde{\mathcal{O}}(\gamma^{-1}\epsilon^{-1})$ query complexity using only a constant number of ancilla qubits.

We can see from the two theorems stated above that the trade-off between the query depth and the query complexity, which is also shown in Table 6.1: the short query depth algorithm has a $\widetilde{\mathcal{O}}(\gamma^{-2})$ dependence on γ, which is sub-optimal. This is compensated by the fact that the query depth is only logarithmic in γ, which can be significantly smaller than that required in the near-optimal algorithm. Similar trade-off exists for the ground-state preparation algorithms (Theorems 6.3.5 and 6.3.10), as shown in Table 6.2.

We first discuss in Section 6.3 the quantum algorithms to solve these tasks with short query depth. As a side note, when the initial state is indeed an eigenstate of H, Theorem 6.3.3 also directly gives rise to a new algorithm for performing QPE using QETU that achieves the Heisenberg-limited precision scaling (see Section 6.3). Finally, assuming access to $(n+1)$-bit Toffoli gates, Section 6.3 describes the quantum algorithms for solving the ground-state preparation and energy estimation problems with near-optimal complexity.

In Sections 6.3, 6.3, and 6.3, we will mostly describe the algorithms to solve these tasks and state the results as lemmas and theorems along the way. We believe this can help readers better grasp the whole picture. A exception is the proofs of Theorems 6.3.3 and 6.3.4, which are presented as formal proofs.

Algorithms with short query depths

Let us first focus on the ground-state preparation problem. We first consider a simple setting in which we assume knowledge of a parameter μ such that

$$\lambda_0 \leq \mu - \Delta/2 < \mu + \Delta/2 \leq \lambda_1, \tag{6.3}$$

where λ_1 is the first excited state energy. We need to find a polynomial approximation to the shifted sign function

$$\theta(x-\mu) = \begin{cases} 1, & x \leq \mu, \\ 0, & x > \mu, \end{cases}$$

and the polynomial should satisfy the requirement in Theorem 6.2.1. To this end, given a number $0 < c < 1$, we would like to find a real polynomial $f(x)$ satisfying

$$|f(x) - c| \leq \epsilon, \quad \forall x \in [\eta, \mu - \Delta/2]; \quad |f(x)| \leq \epsilon, \quad \forall x \in [\mu + \Delta/2, \pi - \eta]. \tag{6.4}$$

As will be discussed Section 6.4, it is preferable to choose c to be slighter smaller than 1 to avoid numerical overshooting. Compared to $c = 1$, this has a negligible effect in practice and does not affect the asymptotic scaling of the algorithm. Taking the cosine transformation in Theorem 6.2.1 into account, we need to find a real even polynomial satisfying

$$\begin{aligned} &|F(x) - c| \leq \epsilon, \quad x \in [\sigma_+, \sigma_{\max}]; \quad |F(x)| \leq \epsilon, \quad x \in [\sigma_{\min}, \sigma_-]; \\ &|F(x)| \leq 1, \quad x \in [-1, 1], \end{aligned} \tag{6.5}$$

where

$$\sigma_\pm = \cos\frac{\mu \mp \Delta/2}{2}, \quad \sigma_{\min} = \cos\frac{\pi - \eta}{2}, \quad \sigma_{\max} = \cos\frac{\eta}{2}. \tag{6.6}$$

Here we have used the fact that $\cos(\cdot)$ is a monotonically decreasing function on $[0, \pi/2]$.

To find such a polynomial $F(x)$, we may use the result in [100, Corollary 7], which constructs a polynomial of degree $\mathcal{O}(\Delta^{-1} \log \epsilon^{-1})$ for $c = 1$ and any $\mu \in [\eta, \pi - \eta]$. This algorithm first replaces the discontinuous shifted sign function by a continuous approximation using error functions (need to shift both horizontally and vertically, and symmetrize to get an even polynomial), and then truncates a polynomial expansion of the resulting smooth function. The construction is specific to the shifted sign function. Its implementation relies on modified Bessel functions of the first kind, which should be carefully treated to ensure numerical stability especially when Δ is small. In Section 6.4, we introduce a simple convex-optimization-based method for

generating a near-optimal approximation, which does not rely on any analytic computation. The convex optimization procedure can be used not only to approximate the shifted sign function, but also to find polynomial approximations in a wide range of settings. The process of obtaining the phase factors can also be streamlined using QSPPACK [53]. The details of this procedure is described in Section 6.4, and an example of the optimal approximate polynomial is given in Fig. 6.2.

We can then run the QETU circuit to apply $f(H) = F(\cos(H/2))$ to an initial guess $|\phi_0\rangle$. If $|\phi_0\rangle$ has a non-zero component in the direction of the ground state $|\psi_0\rangle$, then $f(H)$ will preserve this component up to a factor $c \approx 1$, but will suppress the orthogonal component by a factor $\epsilon \approx 0$, thus giving us a quantum state close to the ground state. This procedure does not always succeed due to the non-unitary nature of $f(H)$, and consequently we need to repeat it multiple times until we get a success. The number of repetitions needed is $\mathcal{O}(\gamma^{-2}\log(\vartheta^{-1}))$ to guarantee a success probability of at least $1-\vartheta$. The result is summarized in Theorem 6.3.5:

Theorem 6.3.5 (ground-state preparation using QETU)**.** *Under the same assumptions as in Definition 6.3.2, with the additional assumption that we have μ satisfying Eq.* (6.3)*, we can prepare the ground state up to fidelity $1-\epsilon$, with probability at least* $2/3$*, with the following cost:*

1. $\widetilde{\mathcal{O}}(\gamma^{-2}\Delta^{-1}\log(\epsilon^{-1}))$ *queries to (controlled-) U and $\mathcal{O}(\gamma^{-2})$ queries to U_I.*
2. *One ancilla qubit.*
3. $\widetilde{\mathcal{O}}(\gamma^{-2}\Delta^{-1}\log(\epsilon^{-1}))$ *additional one-qubit quantum gates.*
4. $\widetilde{\mathcal{O}}(\Delta^{-1}\log(\epsilon^{-1}\gamma^{-1}))$ *maximal query depth of U.*

Again, using the short query depth algorithm, we do not need to use extra two-qubit gates beyond what is needed in (controlled-) U. We can repeat the procedure multiple times to make the success probability exponentially close to 1. In this algorithm, to be more precise than the $\widetilde{\mathcal{O}}$ notation used in Theorem 6.3.5, we need $\mathcal{O}(\gamma^{-2}\Delta^{-1}\log(\gamma^{-1}\epsilon^{-1}))$ queries to U. There is a logarithmic dependence on γ^{-1} because we need to account for subnormalization that comes from post-selecting measurement results when analyzing the error. Success of the above procedure is flagged by the measurement outcome of the ancilla qubit.

For ground-state energy estimation, our strategy is to adapt the binary search algorithm in [94, Theorem 8] to the current setting. In order to estimate the ground-state energy with increasing precision, we need to repeatedly solve a decision problem:

Definition 6.3.6 (The fuzzy bisection problem)**.** *Under the same assumptions as in Definition 6.3.1, we are asked to solve the following problem: output* 0 *when* $\lambda_0 \leq x - h$*, and output* 1 *when* $\lambda_0 \geq x + h$*.*

Here the fuzziness is in the fact that when $x - h < \lambda_0 < x + h$ we are allowed to output either 0 or 1. This is in fact essential for making this problem efficiently solvable. Solving the fuzzy bisection problem will enable us to find the ground-state energy through binary search. We will discuss the details in the proof of Theorem 6.3.3.

To solve the fuzzy bisection problem, we need a real even polynomial $F(x)$ satisfying the following:

$$\begin{aligned} &c - \epsilon' \leq F(x) \leq c + \epsilon', && x \in [\cos((x-h)/2), 1] \\ &|F(x)| \leq \epsilon', && x \in [0, \cos((x+h)/2)] \end{aligned} \tag{6.7}$$

For asymptotic analysis, we can use the approximate sign function from [100, Corollary 6], and the degree of $F(x)$ is $\mathcal{O}(h^{-1}\log(\epsilon'^{-1}))$. With a choice of $F(x)$ that satisfies the above requirements, if $\lambda_0 \geq x + h$, then $\|F(\cos(H/2))\,|\phi_0\rangle\,\| \leq \epsilon'$; if $\lambda_0 \leq x - h$, then

$$\|F(\cos(H/2))\,|\phi_0\rangle\,\| \geq \|F(\cos(\lambda_0/2))\,|\phi_0\rangle\,\| \geq (c - \epsilon')\gamma.$$

Therefore, after choosing $\epsilon' = \gamma c/(2(\gamma+1))$, to solve the fuzzy bisection problem, we only need to distinguish between the following two cases: $\|F(\cos(H/2))\,|\phi_0\rangle\,\| \leq \epsilon' = \gamma c/(2(\gamma+1))$ or $\|F(\cos(H/2))\,|\phi_0\rangle\,\| \geq (c-\epsilon')\gamma = (\gamma+2)\gamma c/(2(\gamma+1))$. These two cases are well separated, because

$$\frac{(\gamma+2)\gamma c}{2(\gamma+1)} - \frac{\gamma c}{2(\gamma+1)} = \frac{\gamma c}{2}.$$

Hence these two quantities are separated by a gap of order $\Omega(\gamma)$, which enables us to distinguish between them using a modified version of amplitude estimation, as will be discussed later. A block encoding of $F(\cos(H/2))$ can be constructed using QETU, which we denote by U_{proj}:

$$(\langle 0| \otimes I_n) U_{\text{proj}} (|0\rangle \otimes I_n) = F(\cos(H/2)). \tag{6.8}$$

Because of the estimate of the degree of $F(x)$, U_{proj} here uses $\mathcal{O}(h^{-1}\log(\gamma^{-1}))$ queries to $U = e^{-iH}$. All we need to do is to distinguish between the following two cases:

$$\|(\langle 0| \otimes I) U_{\text{proj}} (I \otimes U_I)(|0\rangle\,|0^n\rangle)\| \leq \frac{\gamma c}{2(\gamma+1)}$$

or

$$\|(\langle 0|\otimes I)U_{\text{proj}}(I\otimes U_I)(|0\rangle\,|0^n\rangle)\|\geq\frac{(\gamma+2)\gamma c}{2(\gamma+1)}.$$

This problem can be generalized into the following binary amplitude estimation problem:

Definition 6.3.7 (Binary amplitude estimation)**.** *Let W be a unitary acting on two registers (one with one qubit and the other with n qubits), with the first register indicating success or failure. Let $A=\|(\langle 0|\otimes I_n)W(|0\rangle\,|0^n\rangle)\|$ be the success amplitude. Given $0\leq\gamma_1<\gamma_2$, provided that A is either smaller than γ_1 or greater than γ_2, we want to correctly distinguish between the two cases, i.e. output 0 for the former and 1 for the latter.*

In the context of the fuzzy bisection problem in Definition 6.3.6, we need to choose $W=U_{\text{proj}}(I\otimes U_I)$, $\gamma_1=\gamma c/(2(\gamma+1))$, $\gamma_2=(\gamma+2)\gamma c/(2(\gamma+1))$. Note that $\gamma_2/\gamma_1=\gamma+2\geq 2$, and therefore henceforth we only consider the case where for some constant c' we have $\gamma_2/\gamma_1\geq c'$.

Now we can use Monte Carlo sampling to estimate $A=\|(\langle 0|\otimes I_n)W(|0\rangle\,|0^n\rangle)\|$. We will estimate how many samples are needed to distinguish whether $A\geq\gamma_2$ or $A\leq\gamma_1$. We implement $W\,|0\rangle\,|0^n\rangle$ and measure the first qubit, and the output will be a random variable, taking value in $\{0,1\}$, following the Bernoulli distribution and its expectation value is $1-A^2$. We denote $p_1=1-\gamma_1^2$ and $p_2=1-\gamma_2^2$. We will generate N_s samples and check whether the average is larger than $p_{1/2}=(p_1+p_2)/2$ (in which case we choose to believe that $A\leq\gamma_1$), or the average is smaller than $p_{1/2}$ (in which case we choose to believe that $A\geq\gamma_2$). By the Chernoff–Hoeffding Theorem, the error probability is upper bounded by

$$\max\left\{e^{-D(p_{1/2}||p_1)N_s},e^{-D(p_{1/2}||p_2)N_s}\right\},\tag{6.9}$$

where

$$D(x||y)=x\log(x/y)+(1-x)\log((1-x)/(1-y))$$

is the Kullback–Leibler divergence between Bernoulli distributions. Direct calculation, using the fact that $\gamma_2\geq c'\gamma_1$, shows that $D(p_{1/2}||p_1),D(p_{1/2}||p_2)\geq\Omega(\gamma_1^2)$. Therefore to ensure that the error probability is below ϑ', we only need to choose N_s such that $e^{-\Omega(\gamma_1^2N_s)}\leq\vartheta'$. Thus we get the scaling of $N_s=\mathcal{O}(\gamma_1^{-2}\log(\vartheta'^{-1}))$. From the above analysis, we have the following lemma.

Lemma 6.3.8 (Monte Carlo method for solving binary estimation)**.** *The binary amplitude estimation problem in Definition 6.3.7, with the additional assumption that there exists constant $c'>0$ such that $\gamma_2/\gamma_1\geq c'$, can be solved correctly with*

probability at least $1-\vartheta'$ *by querying* W $\mathcal{O}(\gamma_1^{-2}\log(\vartheta'^{-1}))$ *times, and this procedure does not require additional ancilla qubits besides the ancilla qubits already required in* W*. The maximal query depth of* W *is* $\mathcal{O}(1)$*.*

Lemma 6.3.8 enables us to solve the fuzzy bisection problem stated in Definition 6.3.6. With the tools introduced above, we can now prove Theorem 6.3.3.

Proof of Theorem 6.3.3. We solve the ground-state energy estimation problem by performing a binary search, and at each search step we need to solve a fuzzy bisection problem, which we know can be done from the above discussion. Below we will discuss how the binary search works, i.e., why repeatedly solving the fuzzy bisection problem can help us find the ground-state energy. For simplicity of the discussion we set $\eta = \pi/4$ in Definition 6.3.1, i.e., the spectrum of the Hamiltonian is contained in the interval $[\pi/4, 3\pi/4]$. In each iteration of the binary search we have l and r such that $l \leq \lambda_0 \leq r$. In the first iteration we choose $l = \pi/4$ and $r = 3\pi/4$. With l and r we want to solve the following fuzzy bisection problem: output 0 when $\lambda_0 \leq (2l+r)/3$, and output 1 when $\lambda_0 \geq (l+2r)/3$. In other words we let $x = (l+r)/2$ and $h = (r-l)/6$ in Definition 6.3.6. After solving this fuzzy bisection problem, if the output is 0, then we know $l \leq \lambda_0 \leq (l+2r)/3$, and therefore we can update r to be $(l+2r)/3$. Similarly if the output is 1 we can update l to be $(2l+r)/3$. In this way we get a new pair of l and r such that $l \leq \lambda_0 \leq r$ and $r-l$ shrinks by $2/3$.

The values of l and r will converge to λ_0 from both sides, and therefore when $r-l \leq 2\epsilon$, λ_0 will be within ϵ distance from $(l+r)/2$, thus giving us the ground-state energy estimate we want. This will take $\lceil \log_{3/2}(\pi\epsilon^{-1}/2) \rceil$ iterations because $r-l$ shrinks by a factor $2/3$ in each iteration and the initial value is $\pi/2$.

In our context, we need to perform a binary amplitude estimation at each of the $\mathcal{O}(\log(\epsilon^{-1}))$ steps to solve the fuzzy bisection problem, and therefore to ensure a final success probability of at least ϑ we need to choose $\vartheta' = \Theta(\vartheta/\log(\epsilon^{-1}))$ in Lemma 6.3.8. Since $\gamma_1 = \gamma c/(2(\gamma+1)) = \Omega(\gamma)$, as discussed immediately after we introduced Definition 6.3.7, each time we solve the binary amplitude estimation problem we need to use $W = U_{\text{proj}}(I \otimes U_I)$ for $\mathcal{O}(\gamma^{-2}(\log(\vartheta^{-1}) + \log\log(\epsilon^{-1})))$ times. Note that each U_{proj} requires using $U = e^{-iH}$ for $\mathcal{O}(h^{-1}\log(\gamma^{-1}))$ times. Adding up for h that decreases exponentially until it is of order ϵ, we can get the estimate for the cost of estimating the ground-state energy, as stated in Theorem 6.3.3. □

Quantum phase estimation revisited

As an application of the ground-state energy estimation algorithm using QETU, let us revisit the task of performing quantum phase estimation (QPE). Assuming access

to a unitary $U = e^{-iH}$ and an eigenstate $|\psi\rangle$ such that $U|\psi\rangle = e^{-i\lambda}|\psi\rangle$, the goal of the phase estimation is to estimate λ to precision ϵ. This can be viewed as the ground-state energy estimation problem with an initial overlap $\gamma = 1$. Although λ may not be the ground-state energy of H, other eigenvalues of H do not matter because the initial state has zero overlap with other eigenstates.

In order to estimate λ, we can repeatedly solve the fuzzy bisection problem in Definition 6.3.6, and this gives us an algorithm that is essentially identical to the one described in the proof of Theorem 6.3.3. As a corollary of Theorem 6.3.3, the phase estimation problem can be solved with the following cost that achieves the Heisenberg limit:

Corollary 6.3.9 (Phase estimation)**.** *Suppose we are given $U = e^{-iH}$, a quantum state $|\psi\rangle$ such that $U|\psi\rangle = e^{-i\lambda}|\psi\rangle$, where $\lambda \in [\eta, \pi - \eta]$ for some constant $\eta > 0$. We can estimate λ to precision ϵ with probability at least $1 - \vartheta$ using $\widetilde{\mathcal{O}}(\epsilon^{-1}\log(\vartheta^{-1}))$ applications to (controlled) U and its inverse, a single copy of $|\psi\rangle$, and $\mathcal{O}(\epsilon^{-1}(\log(\vartheta^{-1}) + \log\log(\epsilon^{-1})))$ additional one qubit gates.*

It may require some explanation as to why we only need a single copy of $|\psi\rangle$ rather than repeatedly apply a circuit that prepares $|\psi\rangle$. This is because $|\psi\rangle$ is an eigenstate and consequently will be preserved up to a phase factor in the circuit depicted in Figure 6.1 (c). Therefore it can be reused throughout the algorithm and there is no need to prepare it more than once.

Algorithms with near-optimal query complexities

With a block encoding input model, Theorems 6 and 8 in Ref. [94] are near-optimal algorithms for preparing the ground state and for estimating the ground-state energy, respectively. In this section, we combine QETU with amplitude amplification and a new binary amplitude estimation method to yield quantum algorithms with the same near-optimal query complexities. For amplitude estimation we avoid using quantum Fourier transform as it would require an additional register of qubits. Instead we use a procedure described in Section 6.10 based on QETU. Unlike previous near-term methods for amplitude estimation [150, 152] that typically rely on Bayesian inference techniques, and thus require knowledge of a prior distribution, our method does not require such prior knowledge. One could also adapt the QFT-free approximate counting algorithms in [154, 5] to the amplitude estimation problem, but our approach in Section 6.10 is better tailored for the QETU framework.

We need to use amplitude amplification to quadratically improve the γ dependence in Theorem 6.3.5, and to achieve the near-optimal query complexity for preparing the ground state in Definition 6.3.2 (assuming knowledge of μ). Let us first study

the number of ancilla qubits needed for this task. For amplitude amplification we need to construct a reflection operator around the initial guess $|\phi_0\rangle$. This requires implementing $2|0^n\rangle\langle 0^n| - I$, which is equivalent, using phase kickback, to implementing an $(n+1)$-bit Toffoli gate:

$$|1^n\rangle\langle 1^n| \otimes X + (I - |1^n\rangle\langle 1^n|) \otimes I.$$

This $(n+1)$-bit Toffoli gate can be implemented using $\mathcal{O}(n)$ elementary one- or two-qubit gates, on $n+2$ qubits [11, Corollary 7.4]. Note that this can be a relatively costly operation on early fault-tolerant quantum devices. We need two ancilla qubits to implement the reflection operator, but one of them can be reused for other purposes. The reason is as follows: one ancilla qubit is the one that X acts on conditionally in the $(n+1)$-bit Toffoli gate. This one cannot be reused because it needs to start from $|0\rangle$ and will be returned to $|0\rangle$. The other qubit, however, can start from any state and will be returned to the original state, as discussed in [11, Corollary 7.4], and therefore we can use any qubit in the circuit for this task, except for the $n+1$ qubits already involved in the $(n+1)$-bit Toffoli gate. Note in Theorem 6.3.5 we have one ancilla qubit that is used for QETU. This qubit can therefore serve as the ancilla qubit needed in implementing the $(n+1)$-bit Toffoli gate. Thus we only need two ancilla qubits in the whole procedure.

To summarize the cost, we have the following theorem:

Theorem 6.3.10 (near-optimal ground-state preparation with QETU and amplitude amplification)**.** *Under the same assumptions as in Definition 6.3.2, with the additional assumption that we have μ satisfying* (6.3)*, we can prepare the ground state, with probability* $2/3$*, up to fidelity* $1-\epsilon$ *with the following cost:*

1. $\widetilde{\mathcal{O}}(\gamma^{-1}\Delta^{-1}\log(\epsilon^{-1}))$ *queries to (controlled-) U and* $\mathcal{O}(\gamma^{-1})$ *queries to U_I.*
2. *Two ancilla qubits.*
3. $\widetilde{\mathcal{O}}(n\gamma^{-1}\Delta^{-1}\log(\epsilon^{-1}))$ *additional one- and two-qubit quantum gates.*
4. $\widetilde{\mathcal{O}}(\gamma^{-1}\Delta^{-1}\log(\epsilon^{-1}))$ *query depth for U.*

Note that we can repeat this procedure multiple times to make the success probability exponentially close to 1.

For ground-state energy estimation, in the algorithm described in the proof of Theorem 6.3.3, our short query depth algorithm has a $\widetilde{\mathcal{O}}(\gamma^{-2})$ scaling because of the Monte Carlo sampling in the binary amplitude estimation (Definition 6.3.7) procedure. Here we will improve the scaling to $\widetilde{\mathcal{O}}(\gamma^{-1})$ using the technique developed in

[94, Lemma 7]. The technique in [94, Lemma 7] uses phase estimation, and requires $\mathcal{O}(\log((\gamma_2 - \gamma_1)^{-1})) = \mathcal{O}(\log(\gamma^{-1}))$ ancilla qubits. In Section 6.10, we propose a method to solve the binary amplitude estimation problem using QETU, which reduces the number of additional ancilla qubits down from $\mathcal{O}(\log(\gamma^{-1}))$ to two. The result is summarized here:

Lemma 6.3.11 (Binary amplitude estimation)**.** *The binary amplitude estimation problem in Definition 6.3.7 can be solved correctly with probability at least* $1 - \vartheta'$ *by querying* W $\mathcal{O}((\gamma_2 - \gamma_1)^{-1} \log(\vartheta'^{-1}))$ *times, and this procedure requires one additional ancilla qubit (besides the ancilla qubits already required in* W*).*

The key idea is to treat the walk operator in amplitude estimation as a time evolution operator corresponding to a Hamiltonian, and this allows us to apply QETU to extract information about that Hamiltonian.

As a result of this new method to solve the binary amplitude estimation problem, the ground-state energy estimation problem can now be solved using only three ancilla qubits: one for QETU, and two others for binary amplitude estimation.

With these results we can now analyze the cost of ground-state energy estimation in the near-optimal setting, and thereby prove Theorem 6.3.4.

Proof of Theorem 6.3.4. We adopt the same strategy of performing a binary search to locate the ground-state energy, as used in Theorem 6.3.3. The main difference is that instead of using Monte Carlo sampling to solve the binary amplitude estimation problem, we now use QETU to do so, with the complexity stated in Lemma 6.3.11.

We now count how many times we need to query $U = e^{-iH}$ in this approach. Each time we perform binary amplitude estimation, we need to use U_{proj} for $\mathcal{O}(\gamma^{-1} \log(\vartheta'^{-1}))$ times to have at least $1 - \vartheta'$ success probability each time. We need to perform binary amplitude estimation for each step of the binary search, and there are in total $\mathcal{O}(\log(\epsilon^{-1}))$ steps. Therefore to ensure a final success probability of at least $1 - \vartheta$ we need to choose $\vartheta' = \Theta(\vartheta / \log(\epsilon^{-1}))$. At the k-th binary search step, U_{proj} uses $U = e^{-iH}$ for $\mathcal{O}((3/2)^k \log(\gamma^{-1}))$ times. Therefore in total we need to query U, up to a constant factor

$$\sum_{k=0}^{\lceil \log_{3/2}(\pi\epsilon^{-1}/2) \rceil} (3/2)^k \log(\gamma^{-1}) \gamma^{-1} \log(\vartheta'^{-1}) = \mathcal{O}\Big(\epsilon^{-1}\gamma^{-1} \log(\gamma^{-1}) \big(\log(\vartheta^{-1}) + \log\log(\epsilon^{-1}) \big) \Big)$$

times. This query complexity agrees with that in [94, Theorem 8] up to a logarithmic factor.

When we count the number of additional quantum gates needed, there will be an n dependence, which comes from the fact that we need to implement a reflection operator $2\left|0^{n+1}\right\rangle\left\langle 0^{n+1}\right| - I$ each time we implement U_{proj} in the binary amplitude estimation procedure (see Section 6.10). These reflection operators require $\widetilde{\mathcal{O}}(n\gamma^{-1}\log(\epsilon^{-1}\vartheta^{-1}))$ gates. We also need $\mathcal{O}(\epsilon^{-1}\gamma^{-1}\log(\vartheta^{-1}))$ additional quantum gates that come from implementing QETU. Combining the two numbers we get the number of gates as shown in the theorem. □

When preparing the ground state, the parameter μ in Theorem 6.3.10 is generally not known *a priori*. For exactly solvable models or small quantum systems, despite the ability to simulate them classically, one might still want to prepare the ground state, and in such cases μ is available. In the general case, to prepare the ground state without knowing a parameter μ as in Theorem 6.3.10, we can first estimate the ground-state energy to within additive error $\mathcal{O}(\Delta)$, and then run the algorithm in Theorem 6.3.10. This results in an algorithm with the following costs:

Theorem 6.3.12. *Under the assumptions stated in Definition 6.3.2, we can prepare the ground state to fidelity at least $1-\epsilon$, with probability at least $1-\vartheta$, with the following cost:*

1. $\widetilde{\mathcal{O}}(\Delta^{-1}\gamma^{-1}\operatorname{poly}\log(\epsilon^{-1}\vartheta^{-1}))$ *queries to (controlled-) U and* $\mathcal{O}(\gamma^{-1}\operatorname{poly}\log(\Delta^{-1}\epsilon^{-1}\vartheta^{-1}))$ *queries to U_I.*
2. *Three ancilla qubits.*
3. $\widetilde{\mathcal{O}}(n\gamma^{-1}\log(\Delta^{-1}\epsilon^{-1}\vartheta^{-1}) + \Delta^{-1}\gamma^{-1}\log(\epsilon^{-1}\vartheta^{-1}))$ *additional one- and two-qubit quantum gates.*
4. $\widetilde{\mathcal{O}}(\Delta^{-1}\gamma^{-1}\operatorname{poly}\log(\epsilon^{-1}\vartheta^{-1}))$ *query depth of U.*

6.4 Convex-optimization-based method for constructing approximating polynomials

To approximate an even target function using an even polynomial of degree d, we can express the target polynomial as the linear combination of Chebyshev polynomials with some unknown coefficients $\{c_k\}$:

$$F(x) = \sum_{k=0}^{d/2} T_{2k}(x) c_k. \tag{6.10}$$

To formulate this as a discrete optimization problem, we first discretize $[-1,1]$ using M grid points (e.g., roots of Chebyshev polynomials $\{x_j = -\cos\frac{j\pi}{M-1}\}_{j=0}^{M-1}$). One can also restrict them to $[0,1]$ due to symmetry). We define the coefficient matrix, $A_{jk} = T_{2k}(x_j), \quad k = 0,\ldots,d/2$. Then the coefficients for approximating the shifted sign function can be found by solving the following optimization problem

$$\begin{aligned} \min_{\{c_k\}} \quad & \max\left\{ \max_{x_j\in[\sigma_{\max},\sigma_+]} |F(x_j)-c|\,, \max_{x_j\in[\sigma_{\min},\sigma_-]} |F(x_j)| \right\} \\ \text{s.t.} \quad & F(x_j) = \sum_k A_{jk}c_k, \quad |F(x_j)| \le c, \quad \forall j = 0,\ldots,M-1. \end{aligned} \tag{6.11}$$

This is a convex optimization problem and can be solved using software packages such as CVX [70]. The norm constraint $|F(x)| \le 1$ is relaxed to $|F(x_j)| \le c$ to take into account that the constraint can only be imposed on the sampled points, and the values of $|F(x)|$ may slightly overshoot on $[-1,1]\backslash\{x_j\}_{j=0}^{M-1}$. The effect of this relaxation is negligible in practice and we can choose c to be sufficiently close to 1 (for instance, c can be 0.999). Since Eq. (6.11) approximately solves a min-max problem, it achieves the near-optimal solution (in the sense of the L^∞ norm) by definition both in the asymptotic and pre-asymptotic regimes.

Once the polynomial $F(x)$ is given, the Chebyshev coefficients can be used as the input to find the symmetric phase factors using an optimization-based method. Due to the parity constraint, the number of degrees of freedom in the target polynomial $F(x)$ is $\widetilde{d} := \lceil\frac{d+1}{2}\rceil$. Hence $F(x)$ is entirely determined by its values on $\widetilde{d}$ distinct points. We may choose these points to be $x_k = \cos\left(\frac{2k-1}{4\widetilde{d}}\pi\right)$, $k = 1,...,\widetilde{d}$, which are the positive nodes of the Chebyshev polynomial $T_{2\widetilde{d}}(x)$. The problem of finding the symmetric phase factors can be equivalently solved via the following optimization problem

$$\Phi^* = \underset{\substack{\Phi\in[-\pi,\pi)^{d+1},\\ \text{symmetric.}}}{\arg\min}\ L(\Phi),\ L(\Phi) := \frac{1}{\widetilde{d}}\sum_{k=1}^{\widetilde{d}} |g(x_k,\Phi) - F(x_k)|^2\,, \tag{6.12}$$

where

$$g(x,\Phi) := \mathrm{Re}[\langle 0|e^{\mathrm{i}\phi_0 Z}e^{\mathrm{i}\arccos(x)X}e^{\mathrm{i}\phi_1 Z}e^{\mathrm{i}\arccos(x)X}\cdots e^{\mathrm{i}\phi_{d-1}Z}e^{\mathrm{i}\arccos(x)X}e^{\mathrm{i}\phi_d Z}|0\rangle].$$

The desired phase factor achieves the global minimum of the cost function with $L(\Phi^*) = 0$. It has been found that a quasi-Newton method to solve Eq. (6.12) with a particular symmetric initial guess

$$\Phi^0 = (\pi/4, 0, 0, \ldots, 0, 0, \pi/4), \tag{6.13}$$

can robustly find the symmetric phase factors. Although the optimization problem is highly nonlinear, the success of the optimization based algorithm can be explained in terms of the strongly convex energy landscape near Φ^0 [153]. Numerical results indicate that on a laptop computer, CVX can find the near-optimal polynomials for $d \sim 5000$. Given the target polynomial, the optimization based algorithm can find phase factors for $d \sim 10000$ [53]. This should be more than sufficient for most QSP-based applications on early fault-tolerant quantum computers. The streamlined process of finding near-optimal polynomials and the associated phase factors has been implemented in QSPPACK [1].

As an illustrative example of the numerically optimized min-max polynomials, we set $\eta = 0.1, \mu = 1.0, \Delta = 0.4, M = 400, c = 0.999$. This corresponds to $\sigma_{\min} = 0.0500, \sigma_- = 0.8253, \sigma_+ = 0.9211, \sigma_{\max} = 0.9997$. The resulting polynomial and the pointwise errors with $d = 20$ and $d = 80$ are shown in Fig. 6.2. We remark that the polynomial has been reconstructed using the numerically optimized phase factors using QSPPACK, and hence has taken the error in the entire process into account. We find that the pointwise error of the polynomial approximation satisfies the equioscillation property on each of the intervals $[\sigma_{\min}, \sigma_-]$,$[\sigma_+, \sigma_{\max}]$. This resembles the Chebyshev equioscillation theorem of the best polynomial approximation on a single interval (see e.g. [144, Chapter 10]). Fig. 6.3 shows that the maximum pointwise error on the desired intervals converges exponentially with the increase of the polynomial degree.

More generally, to find a min-max polynomial approximation to a general even target function $h(x)$ on a set $\mathcal{I} \subseteq [-1, 1]$ satisfying $|h(x)| \leq c < 1, x \in \mathcal{I}$ we may solve the optimization problem

$$\begin{aligned} \min_{\{c_k\}} \quad & \max_{x_j \in \mathcal{I}} |F(x_j) - h(x_j)| \\ \text{s.t.} \quad & F(x_j) = \sum_k A_{jk} c_k, \quad |F(x_j)| \leq c, \quad \forall j = 0, \ldots, M-1. \end{aligned} \tag{6.14}$$

We also remark that even though QETU only concerns even polynomials, the same strategy can be applied if the target function h is odd, or does not have a definite parity.

[1] `https://github.com/qsppack/QSPPACK`

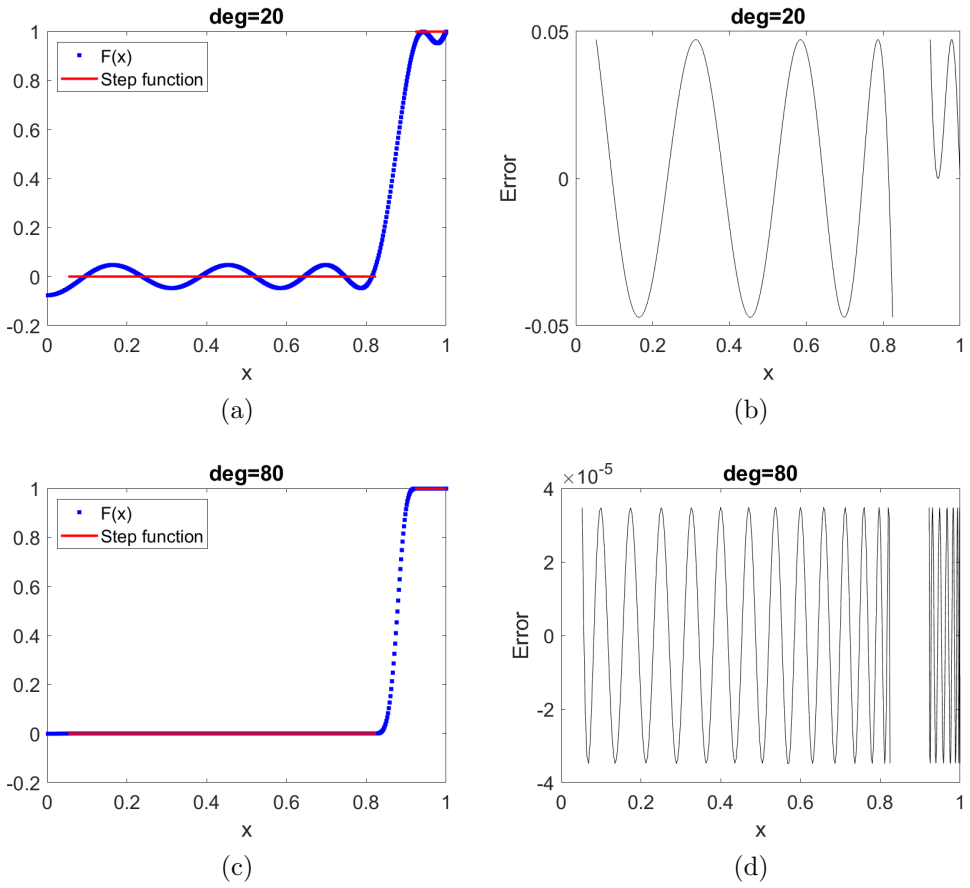


Figure 6.2: The polynomial obtained by convex optimization approximating the shifted sign function and the pointwise error on $[\sigma_{\min}, \sigma_-] \cup [\sigma_+, \sigma_{\max}]$ with $d = 20$ (a,b) and $d = 80$ (c,d).

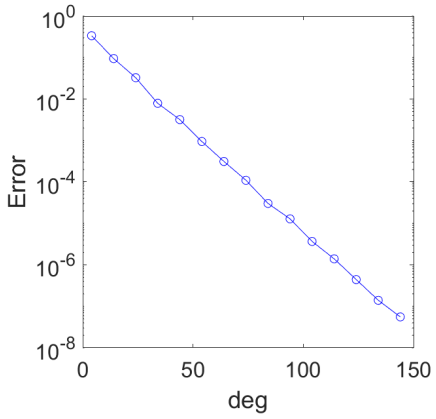
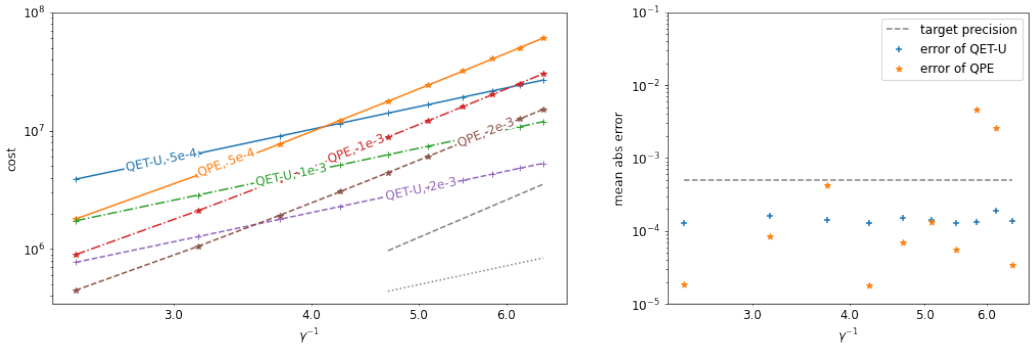


Figure 6.3: Exponential convergence of the maximum pointwise error on $[\sigma_{\min}, \sigma_-] \cup [\sigma_+, \sigma_{\max}]$ with respect to the increase of the polynomial degree obtained by the convex optimization method.

6.5 Numerical comparison with QPE for ground-state energy estimation

In this section we compare the numerical performance of our ground-state energy estimation algorithm in Theorem 6.3.3 with the quantum phase estimation algorithm implemented using semi-classical Fourier transform [71] to save the number of ancilla qubits, as done in Refs. [78, 15]. We will evaluate how many queries to $U = e^{-iH}$ are needed in both algorithms to reach the target accuracy $\epsilon \leq 10^{-3}$. The Hamiltonian H used here has a randomly generated spectrum and is a 200×200 matrix. The initial state $|\phi_0\rangle$ is guaranteed to satisfy $|\langle\phi_0|\psi_0\rangle| \geq \gamma$ with a tunable value of γ. In our algorithm, the number of queries is counted by adding up the degrees of all the polynomials we need to implement using QETU.

Fig. 6.4 shows that to achieve comparable accuracy, our algorithm uses significantly fewer queries than quantum phase estimation, in terms of both the asymptotic scaling (improves from γ^{-4} to γ^{-2}) as well as the actual number of queries for moderately small values of γ. In Fig. 6.4, the error of our method is computed by running the algorithm in Theorem 6.3.3 on a classical computer and comparing the output with the exact ground-state energy, and we show in the figure that the mean of the absolute error in multiple trials. In our method we need to determine the polynomial degree needed for each binary search step (or each time we solve the fuzzy bisection problem in Definition 6.3.6). This polynomial degree is determined by running the



(a) (b)

Figure 6.4: Comparing the performance of the algorithm in Theorem 6.3.3 and the single-ancilla qubit quantum phase estimation using semi-classical Fourier transform [78, 15, 71]. (a) The number of queries of U needed to reach target precision $\epsilon = 5\times10^{-4}, 10^{-3}, 2\times10^{-3}$ for different values of γ. The gray dashed line and dotted line show γ^{-4} scaling and γ^{-2} scaling respectively. Both axes are in logarithmic scale. (b) The mean absolute error achieved by the two algorithms for target accuracy $\epsilon = 5\times10^{-4}$.

algorithm in Section 6.4, and selecting the smallest degree that provides an error below the target accuracy. In our numerical tests we require the approximation error to be below 10^{-3} so that it is much smaller than the squared overlap. The error of QPE is computed by sampling from the exact energy measurement output distribution, which is again simulated on a classical computer, and comparing the output with the exact ground-state energy. We also compute the absolute error for QPE in multiple trials and take the mean in Figure 6.4. The mean absolute errors in Fig. 6.4 (b) show that the advantage of our algorithm does not come from a loose error estimate for QPE, since our algorithm reaches the target precision ($\epsilon = 5\times10^{-4}$ in this case) consistently and QPE does not achieve a higher precision than our algorithm.

6.6 Control-free implementation of quantum spin models

In this section we demonstrate that for certain quantum spin models, the QETU circuit can be simplified without the need of accessing the controlled Hamiltonian

evolution. Consider a Hamiltonian H that is a linear combination of poly(n) terms of Pauli operators. Note that two Pauli operators either commute or anti-commute. Hence for each term in the Hamiltonian, we can easily find another Pauli operator K that anti-commutes with this term. More generally, we assume that H admits a grouping

$$H = \sum_{j=1}^{\ell} H^{(j)}, \quad H^{(j)} = \sum_{s=1}^{d_j} h_s^{(j)}, \tag{6.15}$$

where each $h_s^{(j)}$ is a weighted Pauli operator. For each j, we assume that there exists a single Pauli operator K_j which anti-commutes with $H^{(j)}$, i.e., $K_j H^{(j)} K_j = -H^{(j)}$. The number of groups ℓ is poly(n) in the worst case, but for many Hamiltonians in practice, ℓ may be much smaller. For example, it may be upper bounded by a constant (see examples below).

Conjugating the Pauli string on the time evolution operator flips the sign of the evolution time, i.e., $K_j e^{-i\tau H^{(j)}} K_j = e^{i\tau H^{(j)}}$. Since the time evolution of each Hamiltonian component is the building block of the Trotter splitting algorithm, the time flipping gives a simple implementation of the controlled time evolution without controlling the Hamiltonian components or their time evolution. To implement the controlled time evolution, it suffices to conjugate the circuit implementing the time evolution with the corresponding controlled Pauli string. Suppose $W_j(\tau) \approx U_j(\tau) := e^{-\mathrm{i}\tau H^{(j)}}$ is the quantum circuit approximately implementing the time evolution using Trotter splitting. Then, $K_j W_j(\tau) K_j = W_j(-\tau)$ approximates the reversed time evolution using the same splitting algorithm. This allows us to use Corollary 2.5.4 (see Section 2.5) which simplifies the circuit implementation of QETU.

To illustrate the control-free implementation, let us consider the transverse field Ising model (TFIM) for instance, whose Hamiltonian takes the form

$$H_{\mathrm{TFIM}} = \underbrace{-\sum_{j=1}^{n-1} Z_j Z_{j+1}}_{H_{\mathrm{TFIM}}^{(1)}} \underbrace{-g \sum_{j=1}^{n} X_j}_{H_{\mathrm{TFIM}}^{(2)}}. \tag{6.16}$$

Here $g > 0$ is the coupling constant. Note that a Pauli string

$$K := Y_1 \otimes Z_2 \otimes Y_3 \otimes Z_4 \otimes \cdots \tag{6.17}$$

anti-commutes with both components of the Hamiltonian, namely $K H_{\mathrm{TFIM}}^{(j)} K = -H_{\mathrm{TFIM}}^{(j)}$ for $j = 1$ and 2. Therefore, conjugating the Pauli string K on the time evolution operator flips the sign of the evolution time, i.e., $K e^{-i\tau H_{\mathrm{TFIM}}^{(j)}} K = e^{i\tau H_{\mathrm{TFIM}}^{(j)}}$

for $j = 1$ and 2. In the sense of Eq. (6.15), we have $\ell = 1$. As a consequence, for TFIM, Fig. 6.1(c) is equivalent to the circuit in Fig. 6.5 in which the controlled time evolution is implicitly implemented by inserting controlled Pauli strings.

It should be noted that the controlled Pauli string only requires implementing controlled single-qubit gates, rather than the controlled two-qubit gates of the form $e^{-i\tau Z_j Z_{j+1}}$ (note that the Hamiltonian involves two qubit terms of the form $Z_j Z_{j+1}$). If the quantum circuit conceptually queries the controlled time evolution d times, the simplified circuit only inserts $2d$ controlled Pauli strings in the circuit. In this case, when implementing $W(\frac{1}{2})$ using several Trotter steps, the controlled Pauli strings only need to be inserted before and after each $W(\frac{1}{2})$, but not between the Trotter layers. This simplified implementation gives the quantum circuit in Fig. 6.5(b). Note that in the general case where controlled Pauli strings are inserted in between Trotter layers, the number of controlled Pauli strings required for the implementation is $\mathcal{O}(d\ell r)$ where r is the number of Trotter steps to implement each time evolution operator. The simplified quantum circuit in Fig. 6.5(b) only uses $2d$ controlled Pauli strings. Therefore, this simplified implementation significantly reduces the cost when the number of Trotter step r is large.

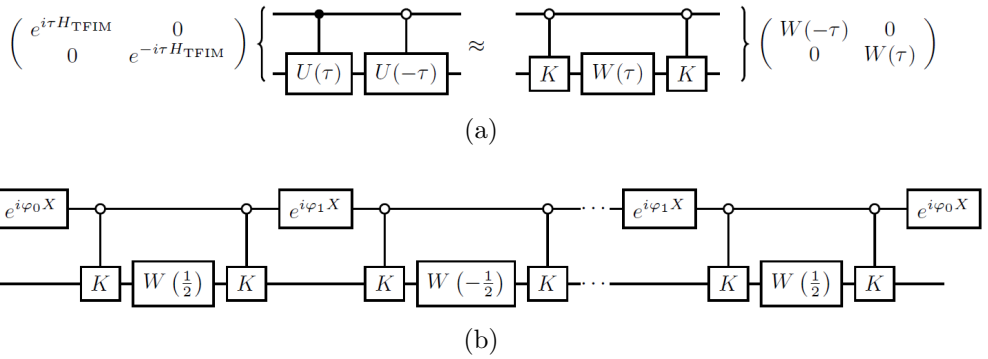


Figure 6.5: Simplified quantum circuit for simulating the TFIM Hamiltonian using QETU. (a) Controlled time evolution of the TFIM Hamiltonian without directly controlling the Hamiltonian. (b) Simplified circuit for implementing QETU. The evolution operator $W\left(-\frac{1}{2}\right)$ can also be implemented as $W\left(\frac{1}{2}\right)$ conjugated by Pauli X operators (see Fig. 2.6).

In contrast to TFIM in which all Hamiltonian components share the same anti-commuting Pauli string, the control-free implementation of general spin Hamiltonian may require more Pauli strings. For example, the Hamiltonian of the Heisenberg

model takes the form

$$H_{\text{Heisenberg}} = \underbrace{-\sum_{j=1}^{n-1} J_x X_j X_{j+1} - \sum_{j=1}^{n-1} J_y Y_j Y_{j+1}}_{H^{(1)}_{\text{Heisenberg}}} \underbrace{-\sum_{j=1}^{n-1} J_z Z_j Z_{j+1}}_{H^{(2)}_{\text{Heisenberg}}}.$$

Let us consider two Pauli strings $K_1 = Z_1 \otimes I_2 \otimes Z_3 \otimes I_4 \otimes \cdots$ and $K_2 = X_1 \otimes I_2 \otimes X_3 \otimes I_4 \otimes \cdots$. Then, we have the anti-commutation relations $K_1 H^{(1)}_{\text{Heisenberg}} K_1 = -H^{(1)}_{\text{Heisenberg}}$ and $K_2 H^{(2)}_{\text{Heisenberg}} K_2 = -H^{(2)}_{\text{Heisenberg}}$. Therefore, conjugating each basic time evolution $e^{-i\tau H^{(j)}_{\text{Heisenberg}}}$ $j = 1, 2$ by controlled K_1 or K_2 respectively, we can implement the controlled time evolution without directly controlling Hamiltonians or the corresponding time evolution operators. This corresponds to $\ell = 2$ in Eq. (6.15). Unlike the implementation for the TFIM, the controlled Pauli strings cannot be canceled between each Trotter layer. Therefore, simulating a Heisenberg model requires additional controlled Pauli gates compared to that in the TFIM simulation.

Other types of quantum Hamiltonians may also be mapped to spin Hamiltonians to perform control-free time evolution using the anti-commutation relation. Consider the 1D Fermi-Hubbard model of interacting fermions in a lattice:

$$H_{\text{FH}} = -\mu \sum_{j=1}^{n} \sum_{\sigma\in\{\uparrow,\downarrow\}} c^\dagger_{j,\sigma} c_{j,\sigma} + u \sum_{j=1}^{n} c^\dagger_{j,\uparrow} c_{j,\uparrow} c^\dagger_{j,\downarrow} c_{j,\downarrow} - t \sum_{j=1}^{n-1} \sum_{\sigma\in\{\uparrow,\downarrow\}} \left(c^\dagger_{j,\sigma} c_{j+1,\sigma} + c^\dagger_{j+1,\sigma} c_{j,\sigma} \right)$$

where $c^\dagger_{j,\sigma}$ and $c_{j,\sigma}$ ($\sigma \in \{\uparrow,\downarrow\} = \{0,1\}$) are creation and annihilation operators for different fermionic mode, μ is the chemical potential, u is the on-site Coulomb repulsion energy, and t is the hopping energy. The equivalent spin Hamiltonian can be derived by applying Jordan-Wigner transformation (see e.g., [129]), which gives (up to a global constant)

$$\begin{aligned} H_{\text{FH,qubits}} = & \underbrace{\frac{1}{2}\left(\frac{1}{2}u - \mu\right) \sum_{j=1}^{n} \sum_{\sigma\in\{0,1\}} Z_{j,\sigma}}_{H^{(1)}_{\text{FH,qubits}}} \\ & + \underbrace{\frac{1}{4} u \sum_{j=1}^{n} Z_{j,0} Z_{j,1}}_{H^{(2)}_{\text{FH,qubits}}} \underbrace{-t \sum_{j=1}^{n-1} \sum_{\sigma\in\{0,1\}} \left(\Sigma^+_{j,\sigma} \Sigma^-_{j+1,\sigma} + \Sigma^+_{j+1,\sigma} \Sigma^-_{j,\sigma} \right)}_{H^{(3)}_{\text{FH,qubits}}} \end{aligned}$$

where the subscript (j,σ) stands for the j-th qubit on the σ-th chain, and $\Sigma^{\pm}_{j,\sigma} := X_{j,\sigma} \pm iY_{j,\sigma}$. Let

$$\begin{aligned} K_1 &= \otimes_{j,\sigma} X_{j,\sigma}, \\ K_2 &= \left(\otimes_{j=1}^{n} X_{j,0}\right) \otimes \left(\otimes_{j=1}^{n} I_{j,1}\right), \\ K_3 &= (\otimes_{j=\text{even}}(Z_{j,0} \otimes Z_{j,1})) \otimes (\otimes_{j=\text{odd}}(I_{j,0} \otimes I_{j,1})) \end{aligned}$$

be three Pauli strings. Then, we have the anti-commutation relations $K_j H^{(j)}_{\text{FH,qubits}} K_j = -H^{(j)}_{\text{FH,qubits}}$ for $j = 1, 2, 3$. Thus, controlling the time evolution of the spin-1/2 Fermi-Hubbard model can be implemented by controlling these Pauli strings. The construction above can also be generalized to 2D Fermi-Hubbard models. Note that direct Trotterization of the 2D Fermi-Hubbard model following the Jordan-Wigner transformation leads to non-optimal complexities, and the complexity can be improved via a fermionic swap networks [87]. The control-free implementation of these more complex instances will be our future work.

For the simplicity of implementation, the energy estimation can be derived from the measurement frequencies of bit-strings using the standard variational quantum eigensolver (VQE) algorithm (see e.g., [109]). We state the algorithm for deriving energy estimation from measurement results in Section 6.11 for completeness. Using the control-free implementation and VQE-type energy estimation, the implementation of the ground-state preparation using QETU can be carried out efficiently on quantum hardware.

6.7 Numerical results for TFIM

Despite the potentially wide range of applications of quantum signal processing (QSP) and quantum singular value transformation (QSVT), their implementation has been limited by the large resource overhead needed to implement the block encoding of the input matrix. To our knowledge, QSP based quantum simulation has only been implemented for matrices encoded by random circuits [50, 52, 47]. Using the QETU and the control-free implementation in the previous section, we show that the short depth version of our algorithm for ground-state preparation and energy estimation can be readily implemented for certain physical Hamiltonians. Our implementation has a very small overhead compared to the Trotter based Hamiltonian simulation, and the circuit uses only one- and two-qubit gate operations. We demonstrate this for the TFIM via IBM Qiskit, and the source code is available in the Github repository [2]. To demonstrate the algorithm, we prepare the ground state of

[2] `https://github.com/qsppack/QETU`

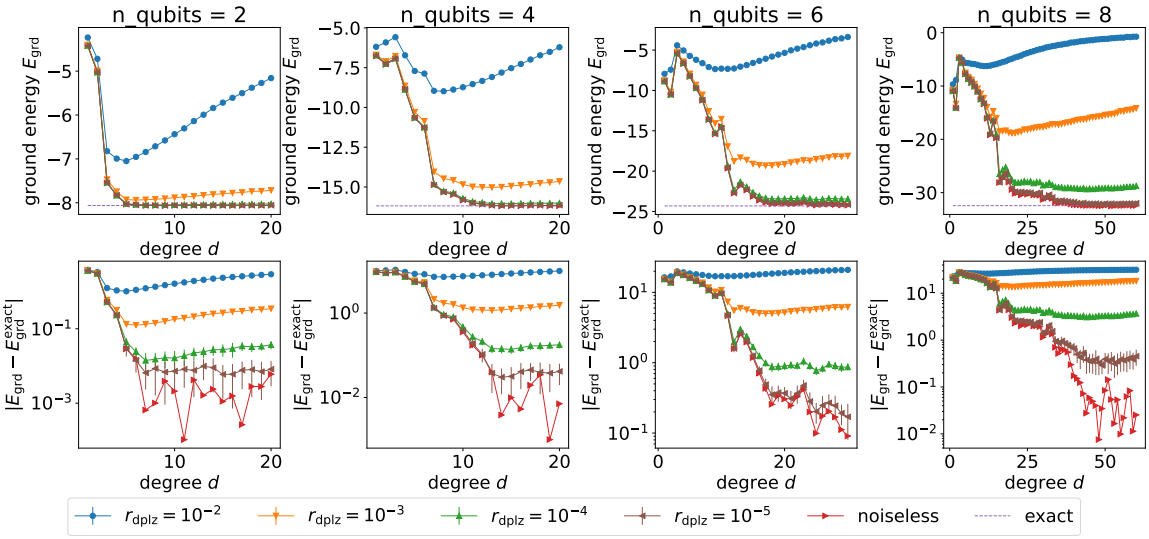


Figure 6.6: Estimating the ground-state energy of the TFIM model using QETU. Each marker labels the data simulated with a given depolarizing error rate r_{dplz}. The dashed line is the exact ground-state energy $E_{\mathrm{grd}}^{\mathrm{exact}}$ of the spin system. The (red) right triangles denote the data computed from the best polynomial approximation by the convex optimization solver, which only include the approximation error and are independent of quantum noise. The error bar stands for the standard deviation estimated from 30 repetitions.

the Ising model with a varying number of qubits n, and coupling strength is set to $g = 4$. In the quantum circuit, we set the initial state to $|0\rangle |\psi_{\mathrm{in}}\rangle$ where $|\psi_{\mathrm{in}}\rangle = |0^n\rangle$ and the additional one qubit is the ancilla qubit on which X rotations are applied. To simplify the numerical test, we compute the value of μ and Δ by explicitly diagonalizing the Hamiltonian. For each Hamiltonian evolution U in the QETU circuit, the number of Trotter steps is set to $r = 3$. We list system-dependent parameters and the initial overlap $\gamma = |\langle\psi_{\mathrm{in}}|\psi_0\rangle|$ for different numbers of system qubits n in Table 6.3. According to the analysis in Section 6.6, it is sufficient to measure two quantum circuits to estimate the ground-state energy of the TFIM. Each quantum circuit in the numerical experiment is measured with 10^5 measurement shots, and we independently repeat the numerical test 30 times to estimate the statistical fluctuation. To emulate the noisy quantum operation, we add a depolarizing error channel to each gate operation in the quantum circuit where the error of single-qubit gate operation and that of the two-qubit gate operation are set to $\frac{r_{\mathrm{dplz}}}{10}$ and r_{dplz} respec-

tively. Assuming the digital error model (DEM), the total effect of the noise can be written as

$$\varrho_{\text{exp}} = \alpha_{\text{DEM}}\varrho_{\text{exact}} + (1 - \alpha_{\text{DEM}})\mathcal{E}(\varrho_{\text{input}}),$$

and the noise channel $\mathcal{E}(\varrho_{\text{input}})$ can be modeled as a global depolarized error channel [23] with circuit fidelity α_{DEM}. Given n and d, the number of single- and two-qubit gates involved in the quantum circuit are

$$n_{g,1} = d(nr + 1) + 1 \quad \text{and} \quad n_{g,2} = d\left((n - 1)r + 2n\right). \tag{6.18}$$

Therefore, the circuit fidelity can be modeled as

$$\alpha_{\text{DEM}} = \left(1 - \frac{r_{\text{dplz}}}{10}\right)^{n_{g,1}} (1 - r_{\text{dplz}})^{n_{g,2}}.$$

The numerical result is presented in Fig. 6.6. The convergence of the noiseless data to the exact ground-state energy suggests that the energy can be computed accurately when d is modest ($10 \sim 30$). The statistical fluctuation, quantified by the standard deviation derived from 30 repetitions, is on the order of 10^{-2} and is not visible in the top panels. When simulating in the presence of the depolarizing noise, numerical results suggest that accurate estimation of the energy requires r_{dplz} to be 10^{-4} or less. This requirement is beyond the noise level that can achieved by current NISQ devices. Therefore we expect that QETU based algorithms are more suited for early fault-tolerant quantum devices. For TFIM, the spectral gap Δ decreases as the number of qubits increases. Therefore the degree of the polynomial also needs to be increased to approximate the shifted sign function and to prepare the ground state to a fixed precision (see Fig. 6.6).

6.8 Discussion

In this work, we develop algorithms for preparing the ground state and for estimating the ground-state energy of a quantum Hamiltonian suitable on early fault-tolerant quantum computers. The early fault-tolerant setting limits the number of qubits, the circuit depth, and the type of multi-qubit control operations that can be employed. While block encoding is an elegant technique for abstractly encoding the information of an input Hamiltonian, existing block encoding strategies (such as those for s-sparse matrices [41, 67]) can lead to a large resource overhead and cannot meet the stringent requirements of early fault-tolerant devices. The resource overhead for approximately implementing a Hamiltonian evolution input model is much lower, and can be a suitable starting point for constructing more complex quantum algorithms.

Many computational tasks can be expressed in the form of applying a matrix function $f(H)$ to a quantum state $|\psi\rangle$. We develop a tool called quantum eigenvalue transformation of unitary matrices with real polynomials (QETU), which performs this task using the controlled Hamiltonian evolution as the input model (similar to that in quantum phase estimation), only one ancilla qubit and no multi-qubit control operations. Combined with a fuzzy bisection procedure, the total query complexity of the resulting algorithm to estimate the ground-state energy scales as $\widetilde{\mathcal{O}}(\epsilon^{-1}\gamma^{-2})$, which saturates the Heisenberg limit with target precision ϵ. The scaling with the initial overlap γ is not optimal, but this result already outperforms all previous quantum algorithms for estimating the ground-state energy using a comparable circuit structure (see Table 6.1).

The QETU technique, and the new convex-optimization-based technique for streamlining the process of finding phase factors, could readily be useful in many other contexts, such as preparing the Gibbs state. It is worth mentioning that other than using shifted sign functions, one can also use the exponential function $e^{-\beta(H-\mu I)}$ (the same as that needed for preparing Gibbs states, with an appropriate choice of β, μ) to approximately prepare the ground state. This gives rise to the imaginary time evolution method. Unlike the quantum imaginary time evolution (QITE) method [112] which performs both real-time evolution and a certain quantum state tomography procedure, QETU only queries the time evolution with performance guarantees and therefore can be significantly more advantageous in the early-fault-tolerant regime.

If we are further allowed to use the $(n+1)$-bit Toffoli gates (which is a relatively low-level multi-qubit operation, as the additional two-qubit operations scale linearly in n), we can develop a new binary amplitude estimation algorithm that is also based on QETU. The total query complexity for estimating the ground-state energy can be improved to the near-optimal scaling of $\widetilde{\mathcal{O}}(\epsilon^{-1}\gamma^{-1})$, at the expense of increasing the circuit depth from $\widetilde{\mathcal{O}}(\epsilon^{-1})$ to $\widetilde{\mathcal{O}}(\epsilon^{-1}\gamma^{-1})$. This matches the results in Ref. [94] with a block encoding input model. This also provides an answer to a question raised in Ref. [93], i.e., whether it is possible to have a quantum algorithm that does not use techniques such as LCU or block encoding, with a short query depth that scales as $\widetilde{\mathcal{O}}(\epsilon^{-1})$, and with a total query complexity that scales better than $\mathcal{O}(\gamma^{-4})$. Our short query depth algorithm shows that it is possible to improve the total query complexity to $\mathcal{O}(\gamma^{-2})$ while satisfying all other constraints. The construction of our near-optimal algorithm (using binary amplitude estimation) indicates that it is unlikely that one can improve the total query complexity to $\mathcal{O}(\gamma^{-1})$ without introducing a factor that scales with γ^{-1} in the circuit depth.

The improvements in circuit depth and query complexity for preparing the ground

state are similar to that of the ground-state energy estimation (see Table 6.2). It is worth mentioning that many previous works using a single ancilla qubit cannot be easily modified to prepare the ground state. It is currently an open question whether the query complexity can be reduced to the near-optimal scaling without using any multi-qubit controlled operation (specifically, whether the additional one- and two-qubit quantum gates can be independent of the system size n).

In practice, the cost of implementing the controlled Hamiltonian evolution can still be high. By exploiting certain anti-commutation relations, we develop a new control-free implementation of QETU for a class of quantum spin Hamiltonians. The results on quantum simulators using IBM Qiskit indicate that relatively accurate estimates to the ground-state energy can be obtained already with a modest polynomial degree ($10 \sim 30$). However, the results of the QETU can be sensitive to quantum noises (such as gate-wise depolarizing noises). On one hand, while the QETU circuit (especially, the control-free variant) may be simple enough to fit on a NISQ device, the error on the NISQ devices may be too large to obtain meaningful results. On the other hand, it may be possible to combine QETU with randomized compilation [148] and/or error mitigation techniques [29] to significantly reduce the impact of the noise, which may then enable us to obtain qualitatively meaningful results on near-term devices [142]. These will be our future works.

6.9 Cost of QETU using Trotter formulas

If the time evolution operator $U = e^{-iH}$ is implemented using Trotter formulas, we can directly analyze the circuit depth and gate complexity of estimating the ground-state energy in the setting of Theorems 6.3.3 and 6.3.4.

We suppose the Hamiltonian H can be decomposed into a sum of terms $H = \sum_{\gamma=1}^{L} H_\gamma$, where each term H_γ can be efficiently exponentiated, i.e. with gate complexity independent of time. In other words we assume each H_γ can be fast-forwarded [9, 72, 136]. We assume that the gate complexity for implementing a single Trotter step is G_{Trotter}, and the circuit depth required is D_{Trotter}. For initial state preparation, we assume we need gate complexity G_{initial} and circuit depth D_{initial}. A p-th order Trotter formula applied to $U = e^{-iH}$ with r Trotter steps gives us a unitary operator U_{HS} with error

$$\|U_{\text{HS}} - U\| \leq C_{\text{Trotter}} r^{-p},$$

where C_{Trotter} is a prefactor, for which the simplest bound is $C_{\text{Trotter}} = \mathcal{O}((\sum_\gamma \|H\|_\gamma)^{p+1})$. Tighter bounds in the form of a sum of commutators are proved in Refs. [40, 137], and there are many works on how to decompose the Hamiltonian to reduce the resource requirement [107, 91, 20, 28]. If the circuit queries $U, U^\dagger$ for d times and the

desired precision is δ, then we can choose

$$d \times C_{\text{Trotter}} r^{-p} = \delta,$$

or equivalently

$$r = \mathcal{O}\left(\max\{d^{1/p} C_{\text{Trotter}}^{1/p} \delta^{-1/p}, 1\}\right). \tag{6.19}$$

As an example, let us now analyze the number of Trotter steps needed in the context of estimating the ground-state energy in Theorem 6.3.3 without amplitude amplification. When replacing the exact U with U_{HS}, we only need to ensure that the resulting error in U_{proj} defined in Eq. (6.8) is $\mathcal{O}(\gamma)$. This will enable us to solve the binary amplitude estimation problem (see Definition 6.3.7) with the same asymptotic query complexity. Since U_{proj} uses U (and therefore U_{HS}) at most $\widetilde{\mathcal{O}}(\epsilon^{-1}\log(\gamma^{-1}))$ times, we only need to ensure

$$\widetilde{\mathcal{O}}(\epsilon^{-1}\log(\gamma^{-1})) \times C_{\text{Trotter}} r^{-p} = \mathcal{O}(\gamma). \tag{6.20}$$

Consequently we need to choose

$$r = \widetilde{\mathcal{O}}\left(\max\{C_{\text{Trotter}}^{1/p} \epsilon^{-1/p} \gamma^{-1/p}, 1\}\right). \tag{6.21}$$

The query depth of U is $\widetilde{\mathcal{O}}(\epsilon^{-1}\log(\gamma^{-1}))$, and therefore the circuit depth is

$$\widetilde{\mathcal{O}}\left(\max\{C_{\text{Trotter}}^{1/p} \epsilon^{-1/p} \gamma^{-1/p}, \log(\gamma^{-1})\} \epsilon^{-1} D_{\text{Trotter}} + D_{\text{initial}}\right).$$

Similarly the total number of queries to U is $\widetilde{\mathcal{O}}(\epsilon^{-1}\gamma^{-2}\log(\vartheta^{-1}))$ times, and the total number of queries to U_I is $\mathcal{O}(\gamma^{-2}\operatorname{poly}\log(\epsilon^{-1}\vartheta^{-1}))$, and the resulting total gate complexity is

$$\widetilde{\mathcal{O}}\left(\max\{C_{\text{Trotter}}^{1/p} \epsilon^{-1/p} \gamma^{-1/p}, 1\} \epsilon^{-1}\gamma^{-2} G_{\text{Trotter}} \log(\vartheta^{-1}) + \gamma^{-2} G_{\text{initial}} \log(\vartheta^{-1})\right).$$

The analysis of the number of Trotter steps in the setting of Theorem 6.3.4 is similar. We still want to ensure (6.20), which results in the same choice of the number of Trotter steps r as in (6.21). Combined with the query complexity in Theorem 6.3.4, the total gate complexity is

$$\widetilde{\mathcal{O}}\left(\max\{C_{\text{Trotter}}^{1/p} \epsilon^{-1/p} \gamma^{-1/p}, 1\} \epsilon^{-1}\gamma^{-1} G_{\text{Trotter}} \log(\vartheta^{-1}) + \gamma^{-1} G_{\text{initial}} \log(\vartheta^{-1})\right). \tag{6.22}$$

6.10 Binary amplitude estimation with a single ancilla qubit and QETU

In this section we discuss how to solve the binary amplitude estimation problem in Definition 6.3.7 using a single ancilla qubit. We restate the problem here.

Definition 6.10.1 (Binary amplitude estimation)**.** *Let W be a unitary acting on two registers, with the first register indicating success or failure. Let $A = \|(\langle 0| \otimes I_n)W(|0\rangle |0^n\rangle)\|$ be the success amplitude. Given $0 \leq \gamma_1 < \gamma_2$, provided that A is either smaller than γ_1 or greater than γ_2, we want to correctly distinguish between the two cases, i.e. output 0 for the former and 1 for the latter.*

In the following we will describe an algorithm to solve this problem and thereby prove Lemma 6.3.11. We can find quantum states $|\Phi\rangle$ and $|\perp\rangle$ such that

$$W(|0\rangle |0^n\rangle) = A |0\rangle |\Phi\rangle + \sqrt{1-A^2}\, |\perp\rangle ,$$

and $(\langle 0| \otimes I) |\perp\rangle = 0$. We also define

$$|\perp'\rangle = -\sqrt{1-A^2}\, |0\rangle |\Phi\rangle + A |\perp\rangle .$$

As in amplitude amplification, we define two reflection operators:

$$R_0 = (2 |0\rangle \langle 0| - I) \otimes I_n, \quad R_1 = W(2 |0^{n+1}\rangle \langle 0^{n+1}| - I_{n+1})W^\dagger.$$

Relative to the basis $\{W(|0\rangle |0^n\rangle), |\perp'\rangle\}$ the two reflection operators can be represented by the matrices

$$\begin{pmatrix} 2A^2-1 & -2A\sqrt{1-A^2} \\ -2A\sqrt{1-A^2} & 1-2A^2 \end{pmatrix}, \quad \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

Therefore we can verify that $|\Psi_\pm\rangle = (W |0\rangle |0^n\rangle \pm i |\perp'\rangle)/\sqrt{2}$ are eigenvectors of R_0R_1:

$$R_0R_1 |\Psi_\pm\rangle = e^{\mp i2 \arccos(A)} |\Psi_\pm\rangle .$$

If we use the usual amplitude estimation algorithm to estimate A we can simply perform phase estimation with R_0R_1 on the quantum state $W(|0\rangle |0^n\rangle)$, which is an equal superposition of $|\Psi_\pm\rangle$:

$$W(|0\rangle |0^n\rangle) = \frac{1}{\sqrt{2}}(|\Psi_+\rangle + |\Psi_-\rangle).$$

However, here we will do something different. We view R_0R_1 has a time-evolution operator corresponding to some Hamiltonian L:

$$R_0R_1 = e^{-iL},$$

where, in the subspace spanned by $\{W(|0\rangle|0^n\rangle), |\perp'\rangle\}$ we have

$$L = 2\arccos(A)\,|\Psi_+\rangle\langle\Psi_+| - 2\arccos(A)\,|\Psi_-\rangle\langle\Psi_-|\,.$$

Then, using QETU in Theorem 6.2.1 we can implement a block encoding, which we denote by $\mathcal{U}$, of $P(\cos(L/2))$ for any suitable polynomial P, and in the same subspace we have

$$P(\cos(L/2)) = P(A)\big(\,|\Psi_+\rangle\langle\Psi_+| + |\Psi_-\rangle\langle\Psi_-|\,\big) = P(A)\big(W|0\rangle|0^n\rangle\langle 0|\langle 0^n|W^\dagger + |\perp'\rangle\langle\perp'|\,\big).$$

Using QETU we can use Monte Carlo sampling to estimate the quantity

$$\|(\langle 0|\otimes I_{n+1})\mathcal{U}\big(\,|0\rangle\otimes(W|0\rangle|0^n\rangle)\big)\|^2 = |P(A)|^2. \tag{6.23}$$

To be more precise, we can start from the state $|0\rangle|0\rangle|0^n\rangle$ on $n+2$ qubits, apply W to the last $n+1$ qubits, then $\mathcal{U}$ to all $n+2$ qubits, and in the end measure the first qubit. The probability of obtaining 0 in the measurement outcome is exactly Eq. (6.23).

Now let us consider an even polynomial $P(x)$ such that $|P(x)| \le 1$ for $x \in [-1,1]$, and

$$P(x) \ge 1-\delta,\ x\in[\gamma_2,1], \quad |P(x)| \le \delta,\ x\in[0,\gamma_1].$$

Such a polynomial of degree $\mathcal{O}((\gamma_2-\gamma_1)^{-1}\log(\delta^{-1}))$ can be constructed using the approximate sign function in [100] (if we take this approach we need to symmetrize the polynomial through $P(x) = (Q(x)+Q(-x))/2$) or the optimization procedure described in Section 6.4. Using this polynomial, we can then use Monte Carlo sampling to distinguish two cases, which will solve the binary amplitude estimation problem:

$$\|(\langle 0|\otimes I_{n+1})\mathcal{U}\big(\,|0\rangle\otimes(W|0\rangle|0^n\rangle)\big)\|^2 \ge (1-\delta)^2$$

or

$$\|(\langle 0|\otimes I_{n+1})\mathcal{U}\big(\,|0\rangle\otimes(W|0\rangle|0^n\rangle)\big)\|^2 \le \delta^2.$$

We can choose $\delta = 1/4$ and it takes running W and $\mathcal{U}$ and measuring the first qubit each $\mathcal{O}(\log(\vartheta^{-1}))$ times to successfully distinguish between the above two cases with probability at least $1-\vartheta$. In this we use the standard majority voting procedure to

boost the success probability. Each single run of $\mathcal{U}$ requires $\mathcal{O}((\gamma_2-\gamma_1)^{-1})$ applications of W, which corresponds to the polynomial degree. Therefore in total we need to apply W $\mathcal{O}((\gamma_2-\gamma_1)^{-1}\log(\vartheta^{-1}))$ times.

In this whole procedure we need one additional ancilla qubit for QETU. Note that the $n+1$ qubits reflection operator in R_1 can be implemented using the $(n+2)$-qubit Toffoli gate and phase kickback. Using [11, Corollary 7.4] we can implement the $(n+2)$-qubit Toffoli gate on $(n+3)$ qubits. As a result another ancilla qubit is needed. We have proved Lemma 6.3.11, which we restate here:

Lemma 6.10.2. *The binary amplitude estimation problem in Definition 6.3.7 can be solved correctly with probability at least $1-\vartheta$ by querying W $\mathcal{O}((\gamma_2-\gamma_1)^{-1}\log(\vartheta^{-1}))$ times, and this procedure requires two additional ancilla qubits (besides the ancilla qubits already required in W).*

6.11 Details of numerical simulation of TFIM

In the numerical simulation for estimating the ground-state energy of TFIM, we explicitly diagonalize the Hamiltonian to obtain the exact ground state $|\psi_0\rangle$, ground energy E_0, the first excited energy E_1, and the highest excited energy E_{n-1}. We then perform an affine transformation to the shifted Hamiltonian

$$H^{\mathrm{sh}} = c_1 H + c_2 I_n, \quad c_1 = \frac{\pi-2\eta}{E_{n-1}-E_0} \text{ and } c_2 = \eta - c_1 E_0. \tag{6.24}$$

Consequently, the eigenvalues of the shifted Hamiltonian are exactly in the interval $[\eta, \pi-\eta]$, i.e., $E_0^{\mathrm{sh}} = \eta$ and $E_{n-1}^{\mathrm{sh}} = \pi-\eta$. The time evolution is then $e^{-\mathrm{i}\tau H^{\mathrm{sh}}} = e^{-\mathrm{i}\tau c_2} e^{-\mathrm{i}\tau c_1 H}$ which means that the evolution time is scaled to $\tau^{\mathrm{sh}} = \tau c_1$ with an additional phase shift $\phi^{\mathrm{sh}} = \tau c_2$. The system dependent parameters are then given by

$$\mu = \frac{1}{2}\left(E_0^{\mathrm{sh}} + E_1^{\mathrm{sh}}\right),\ \Delta = E_1^{\mathrm{sh}} - E_0^{\mathrm{sh}},\ \text{and } \sigma_\pm = \cos\frac{\mu \mp \Delta/2}{2}. \tag{6.25}$$

We set the input quantum state to $|0\rangle|\psi_{\mathrm{in}}\rangle$ where $|\psi_{\mathrm{in}}\rangle = |0^n\rangle$ and the additional one qubit is the ancilla qubit for performing X rotations in QETU. The initial overlap is $\gamma = |\langle\psi_{\mathrm{in}}|\psi_0\rangle|$. We list the system dependent parameters used in the numerical experiments in Fig. 6.6 in Table 6.3.

For completeness, we briefly introduce the algorithm for deriving the energy estimation from the measurement of bit-string frequencies. The energy estimation process can be optimized so that it is sufficient to measure a few quantum circuits to

n	μ	Δ	σ_+	σ_-	c_1	c_2	γ
2	0.7442	1.2884	0.9988	0.7686	0.1824	1.5708	0.5301
4	0.3926	0.5851	0.9988	0.9419	0.0909	1.5708	0.3003
6	0.2887	0.3773	0.9988	0.9717	0.0605	1.5708	0.1703
8	0.2394	0.2788	0.9988	0.9821	0.0453	1.5708	0.0965

Table 6.3: System dependent parameters for different number of system qubits n.

compute the energy. For TFIM, if the ground state is $|\psi_0\rangle$, its ground-state energy is

$$\begin{aligned} E_0 &= \langle\psi_0|H_{\text{TFIM}}|\psi_0\rangle \\ &= -\sum_{j=1}^{n-1}\langle\psi_0|Z_jZ_{j+1}|\psi_0\rangle - g\sum_{j=1}^{n}\langle\psi_0|X_j|\psi_0\rangle =: -\sum_{j=1}^{n-1}\Psi_{j,j+1} - g\sum_{j=1}^{n}\Psi_j^H. \end{aligned}$$

We will show that the energy component $\Psi_{j,j+1}$ and Ψ_j^H can be exactly expressed as the marginal probabilities readable from measurements. Decomposing Z_jZ_{j+1} with respect to eigenvectors, we have

$$\begin{aligned} \Psi_{j,j+1} = \langle\psi_0|Z_jZ_{j+1}|\psi_0\rangle &= \sum_{z_j=0}^{1}\sum_{z_{j+1}=0}^{1}(-1)^{z_j+z_{j+1}}\left|\langle\psi_0|z_j,z_{j+1}\rangle\right|^2 \\ &= \sum_{z_j=0}^{1}\sum_{z_{j+1}=0}^{1}(-1)^{z_j+z_{j+1}}\mathbb{P}\left(z_j,z_{j+1}|\,\psi_0\right). \end{aligned}$$

Here, $\mathbb{P}\left(z_j,z_{j+1}|\psi_0\right)$ is the marginal probability measuring the j-th qubit with z_j and the $(j+1)$-th qubit with z_{j+1} under computational basis when the quantum circuit for preparing the ground state $|\psi_0\rangle$ is given. Similarly, the other quantity involved in the energy is

$$\Psi_j^H = \langle\psi_0|X_j|\psi_0\rangle = \langle\psi_0|H^{\otimes n}Z_jH^{\otimes n}|\psi_0\rangle = \langle\psi_0^H|Z_j|\psi_0^H\rangle = \sum_{z_j=0}^{1}(-1)^{z_j}\mathbb{P}\left(z_j|\psi_0^H\right).$$

Here, $\mathbb{P}\left(z_j|\psi_0^H\right)$ is the marginal probability measuring the j-th qubit with z_j under computational basis when the quantum circuit for preparing the ground state $|\psi_0\rangle$ following a Hadamard transformation, which is denoted as $|\psi_0^H\rangle := H^{\otimes n}|\psi_0\rangle$, is given.

In order to estimate the ground-state energy of TFIM, it suffices to measure all qubits in two circuits: the circuit in Fig. 6.5 (b) and that following a Hadamard transformation on all system qubits. The measurement results estimate the marginal probabilities up to the Monte Carlo measurement error. Furthermore, their linear combination with signs gives the ground-state energy estimate based on the previous analysis.

The procedure for estimating the energy can readily be generalized to other models. Consider a Hamiltonian

$$H = \sum_{k=1}^{L} H_k, \quad H_k = \sum_{j=1}^{v_k} h_{k,j}. \tag{6.26}$$

Here we group the components of the Hamiltonian into L classes, and for a fixed k, the components $h_{k,j}$ can be simultaneously diagonalized by an efficiently implementable unitary V_k. The strategies of Hamiltonian grouping has also been used in e.g., Refs. [81, 146]. We want to estimate the expectation $\langle\psi_0|H|\psi_0\rangle$ where $|\psi_0\rangle$ is the quantum state prepared by some quantum circuit. Then, it suffices to measure L different quantum circuits $\{V_k|\psi_0\rangle : k = 1, \cdots, L\}$ and to compute the expectation from the measurement data by some signed linear combination. For example, to estimate the ground-state energy of the Heisenberg model, we can let $L = 3$ and $V_1 = I^{\otimes n}$, $V_2 = \mathrm{H}^{\otimes n}$ and $V_3 = \left(\mathrm{HS}^\dagger\right)^{\otimes n}$ where H and S are Hadamard gate and phase gate respectively.

Chapter 7

A quantum Hamiltonian simulation benchmark

Hamiltonian simulation is one of the most important problems in quantum computation, and quantum singular value transformation (QSVT) is an efficient way to simulate a general class of Hamiltonians. However, the QSVT circuit typically involves multiple ancilla qubits and multi-qubit control gates. In order to simulate a certain class of n-qubit random Hamiltonians, we propose a drastically simplified quantum circuit that we refer to as the minimal QSVT circuit, which uses only one ancilla qubit and no multi-qubit controlled gates. We formulate a simple metric called the quantum unitary evolution score (QUES), which is a scalable quantum benchmark and can be verified without any need for classical computation. Under the globally depolarized noise model, we demonstrate that QUES is directly related to the circuit fidelity, and the potential classical hardness of an associated quantum circuit sampling problem. Under the same assumption, theoretical analysis suggests there exists an 'optimal' simulation time $t^{\text{opt}} \approx 4.81$, at which even a noisy quantum device may be sufficient to demonstrate the potential classical hardness.

Please note that this chapter is based on [50] (joint work with Lin Lin) and [52] (joint work with K. Birgitta Whaley and Lin Lin).

7.1 Introduction

Recent years have witnessed tremendous progress in quantum hardware and quantum algorithms. As near-term quantum devices become increasingly accessible, the need for holistic benchmarking of such devices is also rapidly growing. Indeed, while most of the frequently used quantum benchmarks, such as randomized benchmarking

[103] and gateset tomography [22], still focus on the performance of one or a few qubits, over the past three years a number of 'whole machine' benchmarks have been proposed that aim at assessing the performance of quantum devices beyond a small number of qubits [23, 48, 57, 8, 125, 47, 50, 126].

While results from such generic benchmarks certainly provide important characteristics of the quantum devices themselves, we are ultimately interested in applying the devices to carry out specific computational tasks. However, the circuit structure of quantum algorithms can be vastly different for different algorithms. Generic quantum benchmarks can miss structural information that is specific to a particular algorithm and which may amplify either quantum errors of certain types or errors amongst a certain group of qubits, and/or reduce errors elsewhere. In this work we address the benchmarking of quantum simulations for time-independent Hamiltonians. Such a simulation can be stated as follows: given an initial state $|\psi_0\rangle$ and a Hamiltonian $\mathfrak{H}$, evaluate the quantum state at time t according to $|\psi(t)\rangle = \exp(-\mathrm{i}t\mathfrak{H})\,|\psi_0\rangle$. Hamiltonian simulation is of immense importance in characterizing quantum dynamics for a diverse range of systems and situations in quantum physics, chemistry and materials science. Simulation of one quantum Hamiltonian by another quantum system was also one of the motivations of Feynman's 1982 proposal for design of quantum computers [59]. Hamiltonian simulation is also used as a subroutine in numerous other quantum algorithms, such as quantum phase estimation [85] and solving linear systems of equations [76].

Following the conceptualization of a universal quantum simulator using a Trotter decomposition of the time evolution operator $e^{-it\mathfrak{H}}$ [96], a number of quantum algorithms for Hamiltonian simulation have been proposed [19, 16, 101, 97, 30]. Detailed assessment of these algorithms, with continued improvement of theoretical error bounds, has since emerged as a very active area of research [17, 18, 14, 42, 38, 98, 39, 40, 34, 133, 7, 138]. In this context, one of the most significant developments in recent years is the quantum signal processing (QSP) method [101], and its generalization, the quantum singular value transformation (QSVT) method [67]. For sparse Hamiltonian simulation, the query complexity of QSVT matches the complexity lower bound with respect to all parameters [101, 67]. The QSVT method also enjoys another advantage, namely that the quantum circuit is relatively simple, and requires very few ancilla qubits. QSVT allows one to use essentially the same parameterized quantum circuit to perform a wide range of useful computational tasks, including Hamiltonian simulation [53], solution of linear systems [67, 95, 143], and finding eigenstates of quantum Hamiltonians [94]. In this sense, it provides a 'grand unification' of a large class of known quantum algorithms [105].

Despite these advantages, QSVT is generally not viewed as a suitable technique for near-term quantum devices today. This is largely because these techniques rely

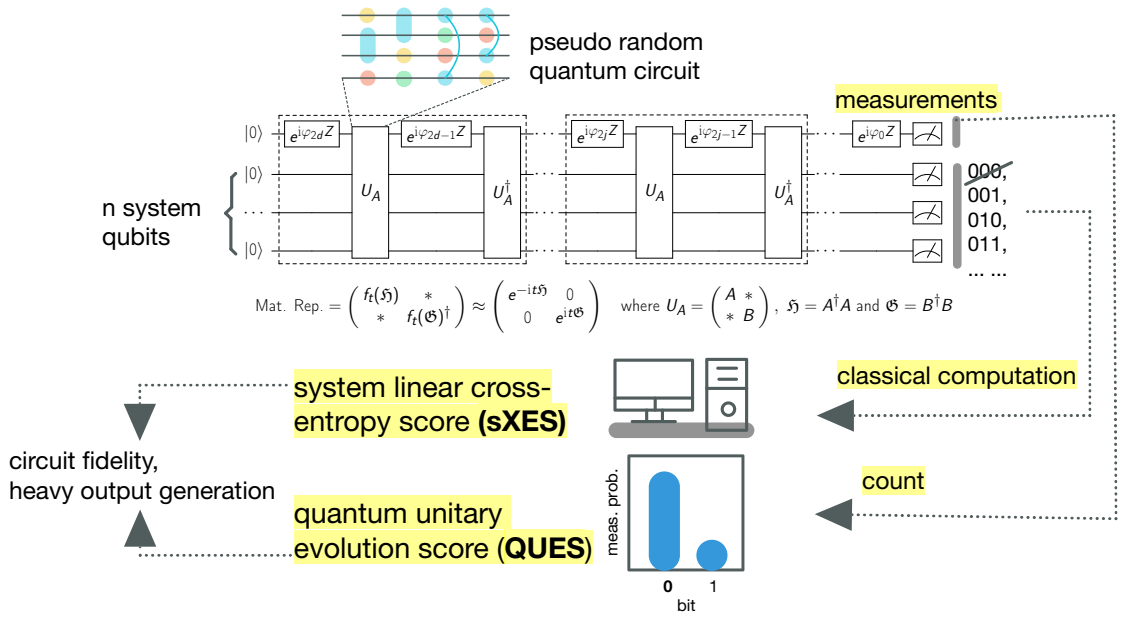


Figure 7.1: Illustration of the minimal quantum singular value transformation (mQSVT) circuit for the Hamiltonian simulation benchmark. The overall circuit implements a complex matrix polynomial $f_t(\mathfrak{H})$ of degree d on the Hamiltonian $\mathfrak{H}$ that is defined in terms of a pseudo random quantum circuit U_A. The circuit acts on $n+1$ qubits, consisting of n system qubits and 1 ancilla qubit. After measuring the top ancilla qubit and post-selecting on the 0 outcome of this, the action on the bottom n system qubits accurately approximates $\exp(-it\mathfrak{H})\,|0^n\rangle$.

on an input model called 'block encoding' which views the Hamiltonian $\mathfrak{H}$ as a submatrix of an enlarged unitary matrix $U_{\mathfrak{H}}$. For Hamiltonians arising from realistic applications (e.g., linear combination of products of Pauli or fermionic operators, and sparse matrices in general), the construction of $U_{\mathfrak{H}}$ often involves multiple ancilla qubits and multi-qubit control gates. Taken together, these requirements can make QSVT very difficult to implement with high fidelity and to date there has been no QSVT based Hamiltonian simulation on realistic devices.

7.2 Preliminaries

Random circuit based block-encoding matrix

To harness the power of the block-encoding model and to avoid its pitfalls, we propose the Random circuit based block-encoding matrix (RACBEM) model as follows. Instead of first identifying A and then finding its block-encoding U_A, we reverse this thought process: we first identify a unitary U_A that is easy to implement on a quantum computer, and then ask which matrix can be block-encoded by U_A.

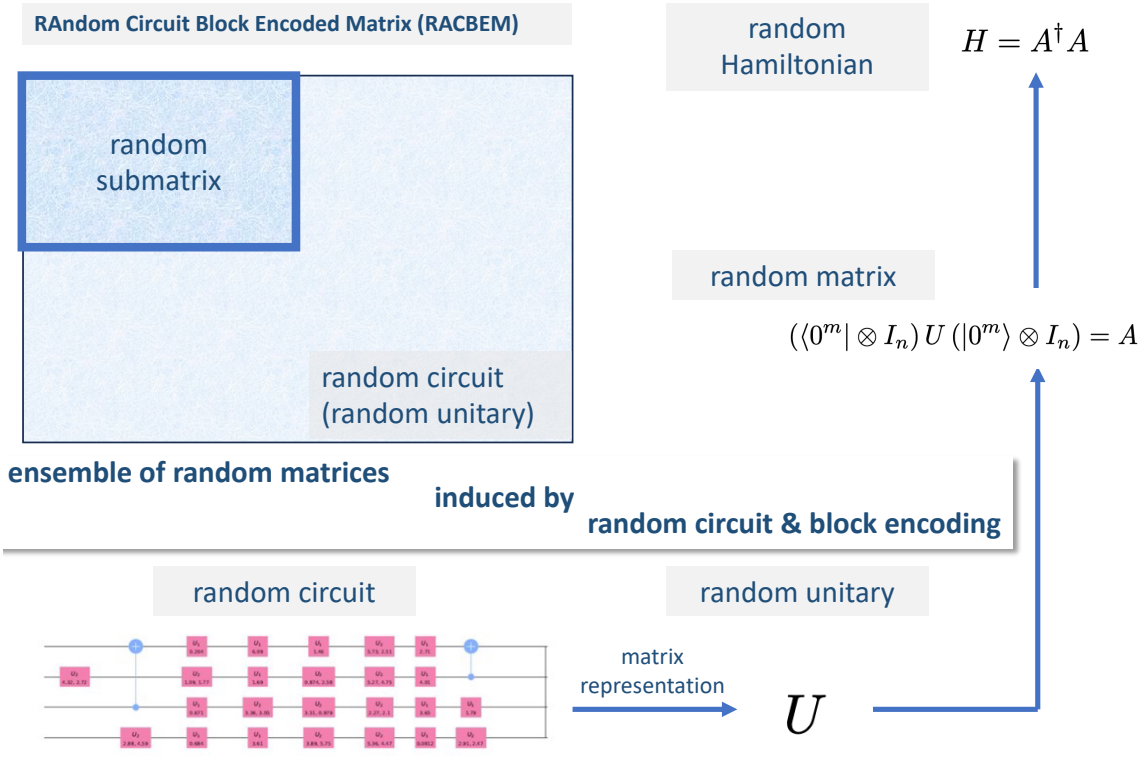


Figure 7.2: An illustration of the main idea and concepts of RACBEM model.

It turns out that any matrix A with $\|A\|_2 \leq 1$ can be given by a $(1, 1, 0)$-block-encoding. Consider any n-qubit matrix with its singular value decomposition (SVD) $A = W\Sigma V^\dagger$, where all singular values in Σ belong to $[0, 1]$. Then we may construct

an $(n+1)$-qubit unitary matrix

$$U_A := \begin{pmatrix} A & W\sqrt{I_n - \Sigma^2} \\ \sqrt{I_n - \Sigma^2}V^\dagger & -\Sigma \end{pmatrix} = \begin{pmatrix} W & 0 \\ 0 & I_n \end{pmatrix} \begin{pmatrix} \Sigma & \sqrt{I_n - \Sigma^2} \\ \sqrt{I_n - \Sigma^2} & -\Sigma \end{pmatrix} \begin{pmatrix} V^\dagger & 0 \\ 0 & I_n \end{pmatrix},$$

which is a $(1, 1, 0)$-block-encoding of A. Since a random circuit with poly(n) depth can approximate an n-qubit Haar measure at least according to the criterion of the 2-design [77], a sufficiently general $(n+1)$-qubit unitary U_A can give access to in principle any n-qubit non-unitary matrices A (up to a scaling factor). Furthermore, such a random circuit U_A can be constructed using only basic one-qubit unitaries and CNOT gates. The matrix A obtained by measuring the first qubit (or in fact, any qubit used as the ancilla) is called a RACBEM. Since the Haar measure is the uniform distribution of unitary matrices, we conclude that RACBEM is a proper generalization of dense matrices on quantum computers suitable for performing linear algebra tasks. The layout of the two-qubit operations can be designed to be compatible with the coupling map of the hardware.

Hermitian RACBEM

In many applications, such as Hamiltonian simulation, thermal state preparation etc, we are only interested in Hermitian matrices. It is possible to find a general circuit U_A that coincidentally block-encodes a Hermitian matrix, but this can become increasingly difficult as n increases. A useful fact is that once a random circuit U_A is given, its Hermitian conjugate $U_A^\dagger$ is easily accessible by conjugating the each gate and reversing the gate sequence. We will show below that this allows us to get access to in principle any n-qubit Hermitian matrix.

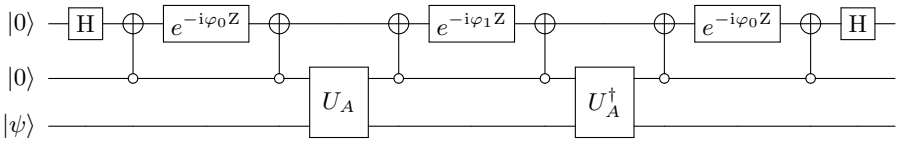


Figure 7.3: Quantum circuit for generating a $(1, 2, 0)$-block-encoding of a H-RACBEM from a $(1, 1, 0)$-block-encoding U_A and its Hermitian conjugate. H is the Hadamard gate, and Z is the Pauli-Z gate.

Consider the quantum circuit in Fig. 7.3 denoted by $U_{\mathfrak{H}}$, where $\varphi_0, \varphi_1 \in [-\pi, \pi)$. In Ref. [50], a direct calculation shows that

$$\mathfrak{H} = (\langle 0^2| \otimes I_n) U_{\mathfrak{H}} (|0^2\rangle \otimes I_n) = [-2\sin(2\varphi_0)\sin\varphi_1]\, A^\dagger A + \cos(2\varphi_0 - \varphi_1). \tag{7.1}$$

Here $\mathfrak{H}$ is a Hermitian matrix. We refer to it as a Hermitian RACBEM (H-RACBEM), and $U_{\mathfrak{H}}$ is its $(1, 2, 0)$-block-encoding. In particular, choosing $\varphi_0 = \pi/8$, $\varphi_1 = -\pi/4$, then $\mathfrak{H} = A^\dagger A$ is Hermitian positive semi-definite. This will be referred to as a canonical H-RACBEM. In other words, a canonical H-RACBEM is constructed from its (non-unique) matrix square root A.

In Fig. 7.3, the CNOT gate controlling on 0 instead of 1 is mainly for notational convenience, and in fact not all CNOT gates are necessary here. For example, in order to implement a canonical H-RACBEM, we only need 1 application of U_A, 1 application of $U_A^\dagger$, 2 H gates, 2 standard CNOT gates, 1 $\mathrm{S}^\dagger$ gate, and 2 T gates (see [50, Appendix A]). Since any matrix with singular values bounded by 1 can be represented as a RACBEM, we immediately have that any Hermitian positive semi-definite matrix with eigenvalues bounded by 1 can be represented as a canonical H-RACBEM, with a sufficiently flexible U_A. As outlined in Section 7.17, the random Hamiltonian matrix $\mathfrak{H} := A^\dagger A$ is uniformly distributed within the unit ball of the space of Hermitian matrices, when considered in terms of the operator norm.

In terms of matrix function, Fig. 7.3 implements a QSVT for a second order polynomial $\mathfrak{H} = h_{\mathrm{SV}}(A)$ with a symmetric choice of phase factors $\Phi = (\varphi_0, \varphi_1, \varphi_0)$, where

$$h(x) = [-2\sin(2\varphi_0)\sin\varphi_1]\, x^2 + \cos(2\varphi_0 - \varphi_1). \tag{7.2}$$

A canonical H-RACBEM is given by $h(x) = x^2$.

Consider any real polynomial $g(x)$ of degree d without parity constraint, satisfying $|g(x)| \leq 1$ for any $x \in [-1, 1]$. Then using the identity

$$g(\mathfrak{H}) = (g \circ h)_{\mathrm{SV}}(A) := f_{\mathrm{SV}}(A), \tag{7.3}$$

any matrix function $g(\mathfrak{H})$ can be expressed as a QSVT with respect to an even polynomial $f = g \circ h$ of degree $2d$. We remark that when g does not have a definite parity, the associated QSVT of A is much more involved. It generally requires using a linear combination of block-encoding of the even and odd parts, which in turn requires implementing controlled U_A [67]. Normally such controlled operations have significant overhead. For instance, if we would like to implement a controlled-RACBEM, generally we need to convert all quantum gates in the circuit of U_A to the controlled version.

Proposal of quantum LINPACK benchmark

The RACBEM, as well as the H-RACBEM model provides a solution to the read-in problem using only basic quantum gates, and we can design them to be optimally adapted to the hardware architecture without resorting to complex quantum compilers. Hence they can be regarded as the proper generalization of "test dense matrices" in the quantum setting.

In this section, we demonstrate that the usage of the H-RACBEM model for solving QLSP. We assume $A = \mathfrak{H}$ is a H-RACBEM, and without loss of generality we may take $|b\rangle = |0^n\rangle$. Let the condition number of $\mathfrak{H}$ be denoted by κ, which is the ratio between the maximum and the minimum of the singular values of $\mathfrak{H}$. It is believed that the computational complexity for solving QLSP cannot generally be better than $\mathcal{O}(\kappa^{1-\delta})$ for any $\delta > 0$ [76], and the cost of using QSVT to solve general linear systems scales as $\widetilde{\mathcal{O}}(\kappa^2 \log(1/\epsilon))$ [67]. So the treatment of ill-conditioned matrices is very difficult especially on near-term devices. To reduce the circuit depth, in the near future it may be more productive to focus on solving well conditioned linear systems.

Note that if A has at least one singular value that is zero (or near zero), a canonical H-RACBEM $\mathfrak{H}$ is not invertible (or very ill-conditioned). Such events can occur more frequently as n becomes large. It can be difficult to diagnose such a problem without first obtaining some spectral information of A, which is perhaps a more difficult task than solving the linear system problem itself.

The H-RACBEM model offers a new and natural way to solve this problem. From Eq. (7.1), assuming $\cos(2\varphi_0 - \varphi_1) > 0$ and $-2\sin(2\varphi_0)\sin\varphi_1 > 0$, and use that $0 \preceq A^\dagger A \preceq 1$, the condition number of $\mathfrak{H}$ can be bounded from above:

$$\kappa(\mathfrak{H}) \leq \frac{\cos(2\varphi_0 + \varphi_1)}{\cos(2\varphi_0 - \varphi_1)}.$$

Therefore the condition number of a H-RACBEM is fully tunable by changing the phase factors φ_0, φ_1. According to Eq. (7.2), this changes the second order polynomial function $h(x)$, so that $\mathfrak{H} := h_{\mathrm{SV}}(A) \succ 0$ has a tunable, bounded condition number.

In order to solve QLSP, we are interested in finding a polynomial $g(x)$ of degree d so that

$$|g(x) - x^{-1}| \leq \epsilon, \quad x \in [\kappa^{-1}, 1].$$

Following [66, Corollary 69], there can be satisfied by an odd polynomial $g(x)$ with degree $d \sim \mathcal{O}(\kappa \log(1/\epsilon))$, which gives an upper bound on d. Numerical results shows that a better approximation to $g(x)$ can be obtained by solving a minimax problem using the Remez algorithm [53], and the polynomial can be chosen to be

either even or odd. In particular, the usage of an even polynomial can further reduce the polynomial degree (see Ref. [50] for details).

We may then construct a degree $2d$ polynomial $f = g \circ h$ as in previous subsection. Once we find the associated phase factors, there is a quantum circuit implementing $g(\mathfrak{H})\,|b\rangle$, which satisfies the error bound $\|g(\mathfrak{H})\,|b\rangle - \kappa^{-1}\mathfrak{H}^{-1}\,|b\rangle\|_2 \leq \epsilon$ for any normalized vector $|b\rangle$. The success probability of measuring the ancilla qubits and obtaining an all 0 output (will be referred to as the success probability for short) is

$$p = \|g(\mathfrak{H})\,|b\rangle\|_2^2 \geq (\kappa^{-1} - \epsilon)^2,$$

which can be of modest size given κ is not too large.

Since measuring the accuracy of all entries of the solution is not a practical goal, our proposal of the quantum LINPACK benchmark is to *measure the success probability* p, i.e. the probability of measuring the 2 ancilla qubits and obtaining $|00\rangle$, and to compare it with the numerically exact probability (denoted by p_{exact}) computed using a classical computer. When κ, ϵ and H-RACBEM are given, the quantum LINPACK benchmark reports the relative error $|p - p_{\text{exact}}|\,/p_{\text{exact}}$ to describe the accuracy of a quantum computer for solving QLSP. In the future we may also take into account the wall clock time, or a quantity analogous to the floating point operations per second (FLOPS) for classical computers to measure the efficiency of a quantum computer. The quantum LINPACK benchmark uses only basic single-qubit gates and CNOT gates, and hence is easy to implement even on near-term devices.

Clearly, the success of the quantum LINPACK benchmark is only a *necessary condition* for the accurate solution of QLSP. However, as will be shown in numerical experiments, this task can already be challenging for near-term devices due to the presence of noise, and therefore the benchmark provides a meaningful and easily implementable criterion for measuring the accuracy of quantum computers. In the subsequent sections, we will present a significant simplification of the quantum circuit. Additionally, a straightforward metric based on Hamiltonian simulation will be proposed to evaluate the performance of quantum computers in addressing scientific computing challenges.

7.3 Results

Overview

In this work we remedy this situation by identifying and demonstrating an application for QSVT on near term quantum devices that allows benchmarking of Hamiltonian

simulation for a class of Hamiltonians that are relevant to recent efforts to demonstrate supremacy of quantum computation over classical computation [8]. This is the class of random Hamiltonians generated from block encoding of random unitary operators that correspond to random unitary circuits. We show that for this class of Hamiltonians it is possible to formulate a simple metric, called the quantum unitary evolution score (QUES), for the success of quantum unitary evolution. This metric is the primary output from the Hamiltonian simulation benchmark, and is directly related to the circuit fidelity. This allows verification of Hamiltonian simulation on near-term quantum devices without any need for classical computation, and the approach can be scaled to a large number of qubits.

The main result of this chapter is a *very simple* quantum circuit (Fig. 7.1), called the minimal QSVT (mQSVT) circuit. With proper parameterization, the mQSVT circuit is able to propagate a certain class of random Hamiltonians $\mathfrak{H}$ to any given target accuracy. In fact, we argue that the mQSVT circuit is not only *the simplest* quantum circuit for carrying out a QSVT based Hamiltonian simulation, but that it is actually the simplest possible circuit for all tasks based on QSVT. Here $\mathfrak{H}$ is not a Hamiltonian corresponding to a given physical system, but a random Hamiltonian generated using a simple random unitary circuit, called a Hermitian random circuit block encoded matrix (H-RACBEM) [50]. However, for the purpose of benchmarking the capability of a quantum device to perform arbitrary Hamiltonian simulations, averaging over a distribution of the underlying arbitrary Hamiltonians is precisely what is required to generate a holistic benchmark protocol that samples from all possible instantiations.

The quantum circuit in Fig. 7.1 consists of two components: an arbitrary random unitary matrix U_A that implicitly defines the Hamiltonian $\mathfrak{H}$, together with its Hermitian conjugate $U_A^\dagger$ and a series of R_z gates with carefully chosen phase factors $\{\varphi_i\}_{i=0}^{2d}$ (see Section 4.3). The mQSVT circuit makes d queries to U_A and $U_A^\dagger$, two of which are shown explicitly in Fig. 7.1. For an n-qubit matrix $\mathfrak{H}$, the total number of qubits needed is always $n+1$, i.e., only 1 ancilla qubit, hereafter referred to as the signal qubit, is required. This is even smaller than the simplest QSVT circuit [101], which requires at least 2 ancilla qubits. However, more important than the reduction of the number of qubits is the fact that Fig. 7.1 removes all two-qubit and multi-qubit gates outside of the unitary U_A. This means that one can choose any convenient entangling two-qubit gate (e.g.. CZ, CNOT, $\sqrt{\text{iSWAP}}$) and any coupling map that is native to a quantum device to construct the random U_A. Combining this with the sequence of single qubit R_z gates then makes the resulting benchmarking quantum circuit of Fig. 7.1 readily executable.

Quantum unitary evolution score (QUES)

Fig. 7.1 implements $f_t(\mathfrak{H})\,|0^n\rangle$ on the system qubits, where $f_t(\mathfrak{H})$ is a matrix polynomial, with approximation error in the operator norm bounded by $\left\| f_t(\mathfrak{H}) - e^{-\mathrm{i}t\mathfrak{H}} \right\|_2 \leq \epsilon$. Therefore in the absence of quantum errors, after applying the circuit to the input state $|0^{n+1}\rangle$, the probability $P_t(U_A) := \|f_t(\mathfrak{H})\,|0^n\rangle\|^2$ of measuring the top ancilla qubit with outcome 0 will be close to 1, indicating that the underlying Hamiltonian evolution is unitary.

From now on, we will primarily consider mQSVT circuits with a fixed set of phase factors $\{\varphi_i\}$ and hence fixed simulation time t. For notational simplicity, we will drop the t-dependence in quantities such as $P_t(U_A)$, unless specified otherwise.

On a real quantum device, the probability $P(U_A)$ should be replaced by $P_{\exp}(U_A)$, which is the experimentally measured probability. We define the quantum unitary evolution score (QUES) by

$$\mathrm{QUES}(n, d) := \mathbb{E}\left(P_{\exp}(U_A)\right), \tag{7.4}$$

where the expectation is taken over the ensemble of random quantum circuit instances U_A. The deviation of QUES from 1 then measures the average performance of the quantum computer under a Hamiltonian simulation task.

There is no unique prescription for constructing random quantum circuits. To fix the choice of U_A, we employ here the model random quantum circuit construction used to analyze the concept of quantum volume in [48]. Here, given a number of qubits n, U_A is constructed to contain n layers, each consisting of a random permutation of the qubit labels followed by random two-qubit gates between the n qubits. Given this construction, the QUES in Eq. (7.4) will then depend only on n and d, and the overall depth of the circuit is approximately $2d$ times the circuit depth of U_A. Note that given the basic quantum gate set of a particular quantum device, alternative constructions of U_A using random choices of specific one- and two-qubit gates are possible.

Fig. 7.4 shows the results of computing the QUES across 8 different IBM Q quantum devices [1], each having 5 qubits and one of three distinct coupling maps (panel b). When the number of qubits $n \leq 3$, the QUES on all devices is relatively high ($\gtrsim 0.7$) but it decreases sharply for $n \geq 4$. In contrast, the QUES decreases only relatively mildly as d increases. This is particularly noticeable for $n = 2$, which may indicate that the quantum circuit transpiler provided by the IBM Q may be particularly effective for this device with very small qubit number. We emphasize that compared to generic benchmark measures such as the quantum volume, the QUES is

[1] https://quantum-computing.ibm.com

specific to the computational task of the Hamiltonian simulation, and any information specific to this is not diluted by additional averaging over output distributions from other computational tasks. In particular, we find that even for quantum devices with relatively small quantum volume (8-QV), the performance in terms of QUES is only mildly worse than for those with a larger quantum volume (32-QV).

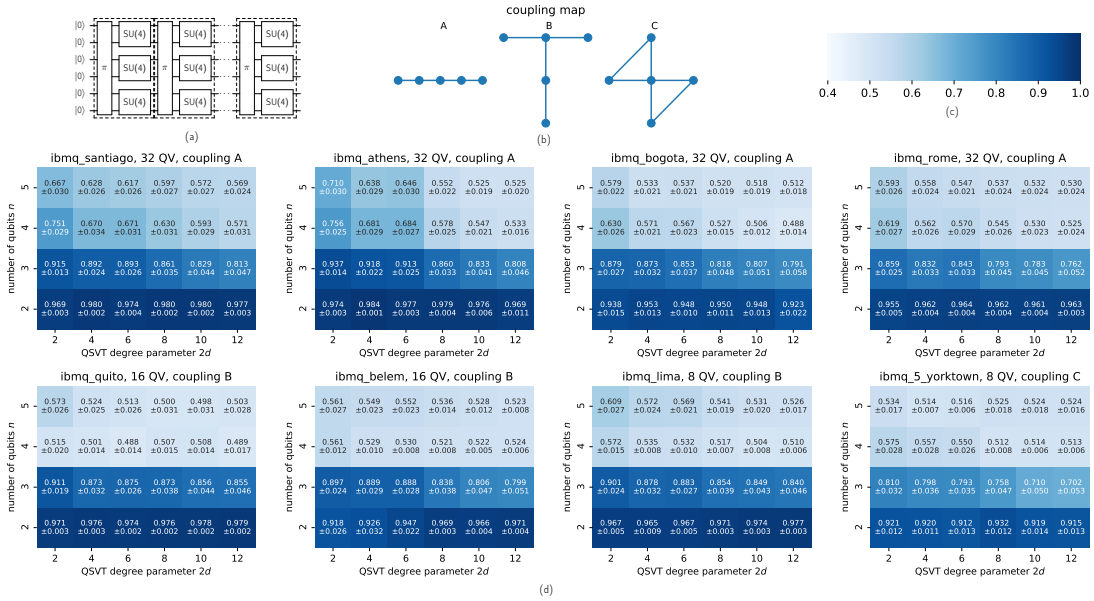


Figure 7.4: Quantum unitary evolution score (QUES) of the 5-qubit quantum devices provided by the IBM Q platform [1]. (a) Visualization of the quantum circuit U_A used in computing QUES. When the number of qubits is n, there are n layers of the dashed boxes consists of the random permutation of the qubits labels followed by random two-qubit gates. After calling the transpiler, the circuit U_A is decomposed with respect to the basic gate set $\Gamma = \{\mathrm{Rz}, \sqrt{\mathrm{X}}, \mathrm{X}, \mathrm{CNOT}\}$ and the coupling map which indicates the available qubit pairs on which CNOT can act. (b) Layouts of coupling maps. (c) Color bar of the heatmap. (d) Each heatmap displays the benchmarking result of a specific quantum device, with the title showing the name of the device, its quantum volume, and its coupling map. Each QUES is estimated from 50 circuit instances. Each circuit instance is measured with $1,000$ measurement shots. The number displayed in each heatmap is the QUES value and its 95% confidence interval.

Circuit fidelity and system linear cross-entropy score (sXES)

The quality of a noisy implementation of a quantum circuit is often characterized by the circuit fidelity. Loosely speaking, the output quantum state of a noisy circuit can be characterized as a convex combination of the correct result and the result obtained under noise, i.e., 'output' $= \alpha \times$ 'correct result' $+ (1-\alpha) \times$ 'noise', where $0 \leq \alpha \leq 1$ is the circuit fidelity. Let $p(U_A, x)$ be the noiseless bitstring probability of measuring the mQSVT circuit with outcome 0 in the ancilla qubit and an n-bit binary string x in the n system qubits. Let $p_{\text{exp}}(U_A, x)$ be the corresponding experimental bitstring probability, which can be estimated from the frequency of occurrence of the bitstring $0x$ in the measurement outcomes. Since the random circuit U_A is approximately drawn from the Haar measure, we make analogous assumptions to those in [23, 8], and assume the following global depolarized error model:

$$p_{\text{exp}}(U_A, x) = \alpha p(U_A, x) + \frac{1-\alpha}{2^{n+1}}. \tag{7.5}$$

We discuss the justification and potential generalization of such an error model in Section 7.9.

Under the global depolarized error model, we now analyze the effect of noise on the circuit and show how measuring the QUES allows the circuit fidelity to be extracted. Prior work has made use of a combination of quantum and classical computation to obtain the circuit fidelity α. Such analysis relies on the possibility of evaluating the noiseless bitstring probability $p(U_A, x)$ classically, given U_A and x, e.g., via tensor network contraction [147]. This enabled the estimation of α from measurements of cross-entropy, referred to as XEB in this setting [23, 8]. We adapt this approach to the Hamiltonian simulation problem by defining a system linear cross-entropy score (sXES):

$$\text{sXES}(U_A) := \sum_{x \neq 0^n} p(U_A, x) p_{\text{exp}}(U_A, x). \tag{7.6}$$

The prefix 'system' is added because the ancilla qubit is fixed to be the $|0\rangle$ state in the definition of $p(U_A, x), p_{\text{exp}}(U_A, x)$, and the $|x\rangle$ state belongs to the system register. In order to connect to the problem of generating heavy weight samples later, our definition of sXES excludes the bitstring 0^n. This is necessary also since the statistical properties of the bitstring 0^n are different from those of the bitstrings in the system register. Taking the expectation with respect to the distribution of U_A, and rearranging Eq. (7.5) then gives an expression for the circuit fidelity:

$$\alpha = \frac{\mathbb{E}\left(\text{sXES}(U_A)\right) - \frac{1}{2^{n+1}} \mathbb{E}\left(\sum_{x \neq 0^n} p(U_A, x)\right)}{\mathbb{E}\left(\sum_{x \neq 0^n} p(U_A, x)^2\right) - \frac{1}{2^{n+1}} \mathbb{E}\left(\sum_{x \neq 0^n} p(U_A, x)\right)}. \tag{7.7}$$

This expression holds for any ensemble of random matrices, and relies only on the assumption that the noise model is depolarizing.

Once the probability distribution of U_A is specified (e.g., the Haar measure [110]), the only term in α that requires a quantum computation is sXES(U_A), and all other terms in Eq. (7.7) can be evaluated classically. However, evaluation of the right-hand side of Eq. (7.7) often requires a significant amount of classical computation when n becomes large [23].

Inferring circuit fidelity from QUES

Based on the discussion so far, it might seem surprising that an alternative, very good approximation to the circuit fidelity can readily be obtained from the QUES metric in Eq. (7.4). This is arrived at by first defining $P_{\text{exp}}(U_A) = \sum_x p_{\text{exp}}(U_A, x)$, i.e., the average over all possible output bit strings x of the probability of measuring a given bit string as outcome of the action of U_A on the input state $|0^{n+1}\rangle$. Then summing both sides of Eq. (7.5) with respect to all bit strings x, further taking the expectation value of both sides over all possible U_A yields a fidelity estimate α_{QUES} that can be obtained directly from the measured QUES value, namely

$$\alpha_{\text{QUES}} = 2 \times \text{QUES} - 1. \tag{7.8}$$

The approximation error ϵ is defined as the maximal error for simulating a bounded Hamiltonian using the mQSVT circuit, namely $\epsilon := \max_{\|\mathfrak{H}\|_2 \leq 1} \left\| f_t(\mathfrak{H}) - e^{-\mathrm{i}t\mathfrak{H}} \right\|_2$. It determines the extent of deviation of α_{QUES} from α. Specifically, under the globally depolarized noise model, we have the following bound (Section 7.11)

$$|\alpha_{\text{QUES}} - \alpha| \leq 16\epsilon + \mathcal{O}(\epsilon^2). \tag{7.9}$$

Here the error bound is derived without including the Monte Carlo measurement error due to the finite number of measurement shots. The analysis of the resulting statistical error is given in Section 7.18.

It is evident that, unlike Eq. (7.7), there is no classical overhead for evaluating α_{QUES} for any n. Since the circuit fidelity α should be non-negative, combining Eq. (7.8) and Eq. (7.9) also indicates that under the assumption of the depolarizing noise model, we have QUES $\geq 0.5 - 8\epsilon + \mathcal{O}(\epsilon^2)$.

To numerically verify the relation between QUES and circuit fidelity, we make use of the digital error model of [23] in which each quantum gate in the circuit is subject to a depolarizing error channel with a certain error rate. We test the resulting noisy quantum circuit with different two-qubit gate error rates r_2 and set the one-qubit gate error rate to $r_1 = r_2/10$. We also discard the rotation gate with phase factor

φ_{2d}, since this just adds a global phase to the exact Hamiltonian simulation. Then, given U_A with a total of g_1 one-qubit gates and g_2 two-qubit gates, the reference value of the circuit fidelity can be set to $\alpha_{\text{ref}} := (1-r_1)^{2d(g_1+1)}(1-r_2)^{2dg_2}$ [8, 23]. We assume U_A is Haar-distributed (numerically verified in Section 7.12) to simplify the computation of classical expectations.

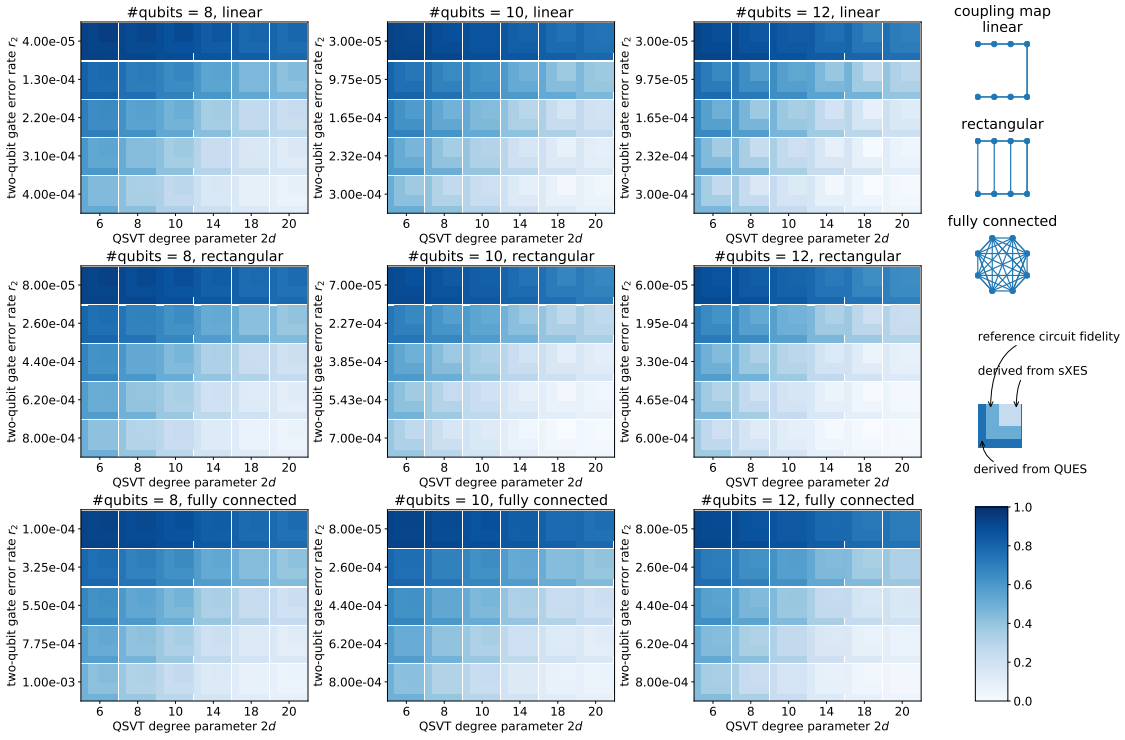


Figure 7.5: Circuit fidelity estimated from the quantum Hamiltonian simulation benchmark. Colored grids represent the circuit fidelity estimated from ~ 100 circuit repetitions. The benchmarking is performed for circuits with a range of number of system qubits, having also variable types of couplings and a range of error parameters. The depth of the random circuit instances is set to the convergent depth deduced from the convergence to Haar measure (see Section 7.12). The right column contains graphical depictions of the coupling maps, the layout of each grid, and the color bar.

Fig. 7.5 summarizes the estimated circuit fidelity for random quantum circuits with different depth parameter d, variable coupling maps, and a range of error parameters. In all cases, we find that the derived circuit fidelity from QUES (α_{QUES}), the circuit fidelity α obtained from sXES, and the reference value α_{ref} are generally

consistent with each other. Numerical results also show that α_{QUES} exhibits a trend that slightly overestimates the value of the fidelity α (see Table 7.5 for numerical values of the fidelities). We also see that for a given set of error rates r_1, r_2, the circuits with highest connectivity show the best performance. This is because random circuits on these architectures converge faster to the Haar measure, which reduces the circuit depth (see Section 7.12).

In the next two subsections we show how to assess and evaluate whether the Hamiltonian simulation with the mQSVT circuit can be a classically hard task. We first define the analog of XHOG for Hamiltonian simulation, which we refer to as sXHOG, and give conditions for the hardness of this. We then show that potential classical hardness can be inferred directly from the value of the circuit fidelity obtained from the QUES, i.e. from α_{QUES}.

Classical hardness and system linear cross-entropy heavy output generation (sXHOG)

The complexity-theoretic foundation of the Google claim of 'quantum supremacy' in [8] is based on a computational task called linear cross-entropy heavy output generation (XHOG) with Haar-distributed unitaries [4, 3, 23, 8]. Specifically, given a number $b > 1$ and a random n-qubit unitary U, the task is to generate k nonzero bitstrings $x_1, x_2, \cdots, x_k \in \{0,1\}^n$ such that $\frac{1}{k}\sum_{j=1}^{k} q(U, x_j) \geq b \times 2^{-n}$, where $q(U,x) = |\langle x|U|0^n\rangle|^2$. Here we use U without the subscript to distinguish the XHOG problem and the sXHOG problem which will be defined later. For k randomly generated bitstrings, we expect that $\frac{1}{k}\sum_{j=1}^{k} q(U, x_j) \approx 2^{-n}$. Therefore any value $b > 1$ will correspond to a 'heavy weight' output. When k is large enough, successful solution of the XHOG problem is considered to be classically hard for every value $b > 1$ [3, 4]. This holds for every circuit fidelity estimate $\alpha > 0$ obtained from the XEB metric, leading to the claim of supremacy in [8] based on extraction of a value $\alpha \approx 0.002$ from the experiments.

For the Hamiltonian simulation benchmark, we can define an analogous linear cross-entropy heavy output generation problem for the n system qubits. Note that the heavy weight samples are now defined only for the system qubits. We shall refer to this heavy output generation problem for Hamiltonian simulation as the sXHOG problem, to emphasize this important feature and the difference from the standard XHOG problem. Specifically, given a number $b > 1$, a Hamiltonian simulation benchmark circuit with sufficiently small approximation error ϵ, and a random $(n+1)$-qubit unitary U_A defining a random Hamiltonian on the n qubits, the task is to generate k nonzero bitstrings $x_1, x_2, \cdots, x_k \in \{0,1\}^n \backslash \{0^n\}$ such that $\frac{1}{k}\sum_{j=1}^{k} p(U_A, x_j) \geq b \times 2^{-n}$.

Now for the case of Hamiltonian simulation, $p(U_A, x) = \mathcal{O}(2^{-n})$ for any $x \neq 0^n$ at all t, but $p(U_A, 0^n)$ can be much larger (for more details see Fig. 7.10(b) in Section 7.16). The state 0^n is then by definition 'heavy' and we must therefore exclude this from the measure in order to avoid a trivial outcome. This is what distinguishes the sXHOG problem from the original XHOG problem.

The potential classical hardness of the XHOG problem is justified by a reduction to a complexity-theoretic conjecture, called linear cross-entropy quantum threshold assumption (XQUATH) [4]. For completeness, we give a similar variant of the reduction of sHOG problem to a conjecture named system linear cross-entropy quantum threshold assumption (sXQUATH) in Theorem 7.14.3 of Section 7.14. The concept of sXQUATH directly parallelizes that of XQUATH, with a similar restriction as above to exclude the output bit string 0^n (for more details see Section 7.14). Similar to the construction in Ref. [4], the classically efficient solution to sXHOG problem yields a violation to sXQUATH, which assumes that $p(U_A, x)$ for $x \neq 0^n$ cannot be efficiently estimated on classical computers to sufficient precision.

Inferring classical hardness from QUES

In order to decide whether a noisy implementation of the Hamiltonian simulation benchmark is potentially in the classically hard regime, we need to establish whether or not the sXHOG problem can be solved for $b > 1$.

Under the assumption that U_A is drawn from the Haar measure, and that the approximation error ϵ of the mQSVT circuit is sufficiently small, we derive the following relation between b and the circuit fidelity α:

$$b = 1 + \frac{\gamma(\alpha - \alpha^*)}{\alpha + 1}. \tag{7.10}$$

Here α^* is a fidelity threshold (not the complex conjugation of α) and γ a constant. Explicit expressions for the threshold value α^* and the constant γ are given in Section 7.15. Both quantities are independent of the circuit fidelity α and depend only on the number of system qubits n and the simulation time t. Eq. (7.10) thus shows that when $\gamma > 0$ and $\alpha > \alpha^*$, we will have $b > 1$ so that the sXHOG problem solved by the mQSVT circuit might be classically hard. This is qualitatively different from the situation for XEB experiments, for which every $\alpha > 0$ implies $b > 1$ [8].

Using the relation between QUES and α in Eqs. (7.8) and (7.9), and assuming that ϵ is sufficiently small, we immediately arrive at the conclusion that when

$$\text{QUES} \geq (1 + \alpha^*)/2, \quad \gamma > 0, \tag{7.11}$$

the corresponding sXHOG problem might be classically hard for a sufficiently large value of n. This is a surprising result, since as noted above, the estimation of QUES does not require intensive classical computation. In fact it is not even necessary to actually generate any heavy weight samples - instead we just need to measure the value of QUES, Eq. (7.4), which is readily done by repeatedly running the circuit in Fig. 7.1 with random circuit parameters as described above. Of course, should one wish to actually solve the sXHOG problem itself, the heavy weight samples would need to be generated using a quantum computer and intensive classical computation for computation of $\frac{1}{k}\sum_{j=1}^{k} p(U_A, x_j)$ would then also be required. But in order to demonstrate the potential regime of classical hardness for Hamiltonian simulation, i.e., the minimal values of n and d to reach this regime, this is not required.

To further investigate the implications of Eq. (7.10), we now explicitly indicate the time dependence of all quantities (i.e., we employ the notation $\gamma \to \gamma_t, \alpha^* \to \alpha_t^*$). In Fig. 7.6 we plot the values of γ_t, α_t^* according to the expressions given in Section 7.15 as a function of the simulation time t, for $n = 4, 8, 12$ qubits. Fig. 7.6 shows that $\gamma_t > 0$ for all t, so then we only need to determine whether it is possible to have fidelity $\alpha \geq \alpha_t^*$. It is evident from Fig. 7.6 that both α_t^* (panel (a)) and γ_t (panel (b)) show oscillatory behavior. We now analyze this behavior to identify an optimal time at which the potential classical hardness of Hamiltonian simulation in this random Hamiltonian setting can be demonstrated for a sufficiently large number of qubits n.

For very short times, i.e., when $t \approx 0$, we have $\alpha_t^* > 1$. This means that we cannot have $b > 1$ for any value of the circuit fidelity $0 \leq \alpha \leq 1$. To see why this is the case, consider the limit $t = 0$. Here $p_t(U_A, 0^n) = 1$, and $p_t(U_A, x) = 0$ for any $x \neq 0^n$. By continuity, when t is very small, the magnitude of $p_t(U_A, x)$ for most bitstrings $x \neq 0^n$ is still very small and cannot reach the heavy output regime. Fig. 7.6(a) also shows that there is a critical simulation time $t^{\text{thr}} \approx 2.26$, for which $\alpha_t^* < 1$ for any $t > t^{\text{thr}}$.

When $t > t^{\text{thr}}$, ideally we would like to have $\alpha_t^* \approx 0$, so that a very low experimental circuit fidelity α is sufficient to reach the heavy output regime. To this end we investigate what happens at the vanishing circuit fidelity, i.e., $\alpha = 0$. Detailed analysis shows that in the large n limit, we have $\gamma_t \alpha_t^* = \mathbb{E}\left(p_t(U_A, 0^n)\right)$, and Eq. (7.10) can be simplified as (see Section 7.15)

$$b|_{\alpha=0} = 1 - \mathbb{E}\left(p_t(U_A, 0^n)\right), \tag{7.12}$$

where the expectation value is taken with respect to the random unitaries U_A as before. Thus when the expectation value is positive, i.e., $\mathbb{E}\left(p_t(U_A, 0^n)\right) > 0$, in the large n limit we have $b|_{\alpha=0} < 1$ and the task should not be classically hard. Moreover, since b is a continuous function of α, even if we now have finite circuit fidelity α, when

this is small enough we can still find $b < 1$. This provides an alternative explanation of Eq. (7.10), namely, that the circuit fidelity α needs to be larger than the finite positive threshold value $\alpha_t^* > 0$ for *most* values of $t > t^{\text{thr}}$.

As a result of these considerations, when n is large enough, it is important to focus on the regimes where the expectation value $\mathbb{E}\left(p_t(U_A, 0^n)\right) \approx 0$, which from Eq. (7.10) implies that the threshold fidelity $\alpha_t^* \approx 0$. The numerical results shown in Fig. 7.6 indicate that this can happen in two different scenarios. The first is when the simulation time $t \to \infty$ (see the analytic justification of this statement in Section 7.19). Of course this requires a very large circuit depth and is a physically 'trivial' limit that is impractical on near-term quantum devices. The second scenario, which is much more relevant in practice, is when α_t^* reaches its first minimum, which defines an optimal time $t = t^{\text{opt}}$. In the large n limit, the value of t^{opt} can be rationalized as the first node of the Bessel function $J_0(t/2)$ (see Section 7.16). Fig. 7.6 (a) shows that for $t^{\text{opt}} \approx 4.81$, we already have $\mathbb{E}\left(p_t(U_A, 0^n)\right) \approx 0$ and $\alpha_t^* \approx 0$. Therefore simulating to the time $t = t^{\text{opt}}$ is highly desirable, since this is a relatively short time at which the Hamiltonian simulation benchmark is nevertheless now guaranteed to solve the sXHOG problem even for a very small circuit fidelity. Our numerical results indicate that the values of t^* and t^{opt} depend only weakly on n, and their values are nearly converged for n as small as 12. Therefore this value of t^{opt} can be used in a future quantum simulation in the heavy output regime.

7.4 Discussion

We have presented a quantum benchmark for Hamiltonian simulation on quantum computers. The Hamiltonian simulation problem is solved using a minimal quantum singular value transformation (mQSVT) circuit. The primary output of the Hamiltonian simulation benchmark is a single number called QUES, which can be verified without any classical computation, even in the regime that is potentially hard for classical computation. Therefore the Hamiltonian simulation benchmark provides a scalable benchmark of the circuit fidelity under the global depolarized error model, and can be executed and verified on future quantum devices with a large number of qubits.

As the current quantum computing technologies advance, the possibility of implementing some error correction is improving [35]. Here the highly structured mQSVT circuit provides useful indications of where best to implement error correction under limited resources for this. Recall that the mQSVT circuit consists of a series of repetitions of a random circuit U_A and its conjugate $U_A^\dagger$, interleaved with single-qubit Z rotation operators characterized by carefully selected phase factors. Thus given

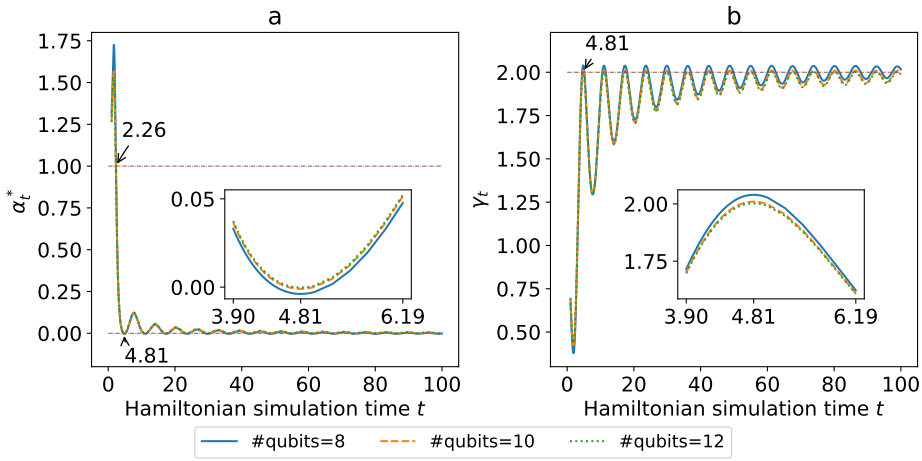


Figure 7.6: Quantities relevant to the system linear cross-entropy heavy output generation (sXHOG) problem, evaluated using the explicit expressions given in Section 7.15. (a) The threshold fidelity α_t^* as a function of Hamiltonian simulation time t. The upper value noted on the plot indicates the time value $t^{\text{thr}} \approx 2.26$ where $\alpha^*(t^{\text{thr}}) = 1$. The lower value noted on the plot indicates the regime at finite time $t^{\text{opt}} \approx 4.81$ with the first minimal value of threshold fidelity. (b) The parameter γ_t as a function of Hamiltonian simulation time t. The value noted on the plot indicates the value $\gamma_t \approx 2$ at the optimal time $t^{\text{opt}} \approx 4.81$. The averages in (a) and (b) are estimated numerically from ~ 100 instances of the mQSVT circuit encoding random Hamiltonians drawn from the Haar measure. Insets in each panel show the behavior of α_t^* and γ_t near the optimal time $t^{\text{opt}} \approx 4.81$.

a specific random Hamiltonian block encoded in U_A, the time dependent evolution operator for this Hamiltonian is defined entirely in terms of the phase angles for the single-qubit Z rotation operators. Since these phases should moreover be precisely determined, this suggests that on near-term quantum devices that may allow for some error correction but have overall limited resources, quantum error correction for these single-qubit rotations should be prioritized.

It is also useful to consider here the applicability of this Hamiltonian simulation approach to general Hamiltonians, i.e., not restricted to random Hamiltonians, on near-term quantum computers. Unfortunately it appears that for current quantum technologies there is potentially a large gap between the feasible simulation of a H-RACBEM given in this work and that of a general Hamiltonian relevant to e.g., molecular or solid-state physics. The main reason is that the block encoding of most

Hamiltonians of practical interest will involve significant numbers of ancilla qubits, as well as multi-qubit control gates, all of which are extremely expensive on near-term quantum devices. In contrast to this general situation, the construction of H-RACBEM uses only whatever one-qubit and two-qubit gates are available for a given quantum device and is thus considerably easier. Nevertheless, it is possible that undertaking Hamiltonian simulation with H-RACBEM may also yield interesting physical applications to the various settings in which quantum chaotic dynamics are relevant. One immediate possibility in this direction is to use H-RACBEM to simulate the dynamics of quantum scrambling or quantum chaos in strongly interacting quantum systems. Scrambling dynamics can be studied by simulating out-of-time-order correlators (OTOCs) for effective Hamiltonians that can be defined implicitly in terms of a random circuit for time t (see e.g., [111]). We note that one can easily perform a Hamiltonian simulation backward in time, merely by reversing the sign of t, so the mQSVT circuit for an OTOC at any time t of a random Hamiltonian encoded in H-RACBEM can be readily constructed by adding local operators between forward and backward implementations of the mQSVT. Evaluation of the circuit at different times t can be implemented either by reevaluating the phase factors (which may required building a longer circuit depending on the accuracy required). The circuits can also be adapted to Hamiltonian simulation at finite temperatures and hence also to scrambling at finite temperatures. From a theoretical perspective it would also be useful to explore to what extent the structure of the H-RACBEM influences the speed of scrambling [27].

Our theoretical analysis of the sensitivity of the Hamiltonian benchmarking scheme in this work was based on a fully depolarized noise model, which is often assumed to be a good model for superconducting qubits [8]. In general the Pauli stochastic noise model on which is based may be biased or non-uniform across qubits. In addition, thermal noise and coherent errors are important for some qubit architectures. It will be useful to extend the current analysis to more general noise models, and some of these aspects are discussed in Section 7.9.

Finally, we note that while this Hamiltonian simulation benchmark is restricted to the specific class of random Hamiltonians, it might also provide information relevant to more general Hamiltonian simulations. Efforts to analyze the complexity of analog Hamiltonian simulations have often focused on the relation of such simulations to classical sampling tasks [2, 26, 75], and are closely related to the cross-entropy analysis for sampling of random quantum circuits of [23, 8]. As noted recently [75], the classical hardness can be shown for certain classes of analog quantum Hamiltonian simulation [60, 13]. Note that the potential classical hardness of the original XHOG problem corresponding to Google's supremacy experiment is justified by a reduction to a complexity-theoretic conjecture called XQUATH [4]. However, a recent paper

[61] that appeared after submission of the current work has provided evidence that can refute XQUATH, at least for some classes of quantum circuits. Therefore it is possible that our sXQUATH assumption can be refuted on the same basis. It could be useful to explore generalizations of other classical sampling tasks to the QSVT setting, as was done here for the cross-entropy heavy output generation, to help guide the search for Hamiltonians whose simulation by QSVT can exhibit quantum advantages. Finally, the current approach of analysis of alternative fidelity measures under Hamiltonian simulation using mQSVT may provide useful for analysis of recent fidelity based experimental studies of analog Hamiltonian simulations that followed the emergent random nature of a projected ensemble of states [43].

7.5 Notations

We first introduce the definition of block encoding. Let $A \in \mathbb{C}^{N\times N}$ be an n-qubit Hermitian matrix ($N = 2^n$). If we can find an $(n+1)$-qubit unitary matrix U_A such that ($*$ stands for a matrix block whose entries are not of interest)

$$U_A = \begin{pmatrix} A & * \\ * & * \end{pmatrix} \tag{7.13}$$

holds, i.e. A is the upper-left matrix block of U_A, then we may get access to the action of A on an n-qubit state $|\psi\rangle$ via the unitary matrix U_A by

$$U_A |0\rangle |\psi\rangle = |0\rangle (A|\psi\rangle) + |\perp\rangle ,$$

where $|\perp\rangle$ is an unnormalized $(n+1)$-qubit state not of interest and satisfies $(|0\rangle\langle 0| \otimes I_n)|\perp\rangle = 0$. Here we follow the row-major order convention. For instance,

$$|0\rangle |\psi\rangle \equiv \begin{pmatrix} \psi \\ 0^n \end{pmatrix}, \quad |1\rangle |\psi\rangle \equiv \begin{pmatrix} 0^n \\ \psi \end{pmatrix},$$

and Eq. (7.13) can also be written as $A = (\langle 0| \otimes I_n) U_A (|0\rangle \otimes I_n)$.

Clearly when the operator norm $\|A\|_2$ is larger than 1, A cannot be encoded by any unitary U_A as in Eq. (7.13). Generally if we can find $\alpha, \epsilon' \in \mathbb{R}_+$, and an $(m+n)$-qubit matrix U_A such that

$$\|A - \alpha (\langle 0^m| \otimes I_n) U_A (|0^m\rangle \otimes I_n)\|_2 \leq \epsilon', \tag{7.14}$$

then U_A is called an (α, m, ϵ')-block-encoding of A. Here m is called the number of ancilla qubits for block encoding. We refer to [67] for more details on block encoding.

When the block encoding is exact with $\epsilon' = 0$, U_A is called an (α, m)-block-encoding of A. The special case of the $(1, 1)$-block-encoding may also be called a 1-block-encoding.

In this chapter, for notational simplicity, we may use U without a subscript to represent a $(n+1)$-qubit quantum circuit drawn at random from a certain probability distribution. Unless otherwise noted, A denotes the upper-left n-qubit submatrix of U, i.e. U is the 1-block-encoding of A. This matrix A is also called a random circuit block encoded matrix (RACBEM), and $\mathfrak{H} = A^\dagger A$ is corresponding Hermitian random circuit block encoded matrix (H-RACBEM) [50].

We use $N = 2^n$ to represent the dimension of the Hilbert space of the system qubits, and I_n to denote the n-qubit identity matrix. For a complex square matrix A with singular value decomposition (SVD) $A = W\Sigma V^\dagger$, its singular value transformation through an even function g is defined as $g_{\mathrm{SV}}(A) = Vg(\Sigma)V^\dagger$. Here, the right triangle in the notation means only the right singular vectors V are kept in the transformation. If we consider $|A| := \sqrt{A^\dagger A} = V\Sigma V^\dagger$, then the singular value transformation of A is equal to the eigenvalue transformation of the Hermitian matrix $|A|$, namely, $g_{\mathrm{SV}}(A) = g(|A|)$. Furthermore, due to the even parity of g, there is a function f so that $g(x) = f(x^2)$ and $g_{\mathrm{SV}}(A) = f(|A|^2) = f(\mathfrak{H})$. In particular, when $g_t(x)$ is an even polynomial approximation to $s_t(x) = e^{-\mathrm{i}tx^2}$, we can define $g_t(x) = f_t(x^2)$. Hence $f_t(x)$ approximates $e^{-\mathrm{i}tx}$, and $(g_t)_{\mathrm{SV}}(A) = f_t(\mathfrak{H})$ approximates the Hamiltonian evolution $e^{-\mathrm{i}t\mathfrak{H}}$.

We use $\mathcal{U}_{f,U}$ to represent the minimal quantum singular value transformation (mQSVT) circuit in Fig. 7.1, which has only a single ancilla qubit, $m = 1$. For any n-qubit input state $|\psi\rangle$, the mQSVT circuit performs the following transformation of the input quantum state,

$$\mathcal{U}_{f,U} |0\rangle \otimes |\psi\rangle = |0\rangle \otimes (g_{\mathrm{SV}}(A) |\psi\rangle) + |1\rangle \otimes |\perp\rangle ,$$

where $|\perp\rangle \in \mathbb{C}^N$ is an unnormalized quantum state. In other words, $\mathcal{U}_{f,U}$ is the 1-block-encoding of $f(\mathfrak{H}) \equiv g_{\mathrm{SV}}(A)$. $\|A\|_2 := \sigma_{\max}(A)$ is the operator norm of a matrix which is equal to its maximal singular value. $\|f\|_\infty := \max_{x\in[-1,1]} |f(x)|$ is the infinity norm of continuous functions on $[-1, 1]$. $\mathbb{E}(\cdot)$ stands for the average over the random matrix ensemble (most commonly, the ensemble of U). Both $\overline{z}$ and z^* stands for the complex conjugate of a complex number z. For a complex polynomial $P(x) = \sum_i c_i x^i \in \mathbb{C}[x]$, its complex conjugate as $P^*(x) = \sum_i c_i^* x^i$. For a matrix A, the transpose, Hermitian conjugate and complex conjugate are denoted by $A^\top$, $A^\dagger$, A^*, respectively. Without otherwise noted, an n-bit binary string $x \in \{0, 1\}^n$ is identified to its decimal representation. Specifically, when an n-bit binary string appears in the subscript of a matrix or a vector, it is identified to be its decimal representation (we use a zero-based indexing). For example, $A_{0^n,1^n} := A_{0,2^n-1}$.

Table 7.1 summarizes the main notations used in this chapter. In the context of Hamiltonian simulation, many quantities depend on the value of the simulation time t. Such a t-dependence is usually added as a subscript such as $p_t(U, x)$. Most of the discussion focuses on the simulation at a fixed time t. Therefore when the context is clear, for simplicity we may drop the t dependence.

Symbol	Definition
$\mathcal{U}_{f,U}$	mQSVT circuit in Fig. 7.1 implementing a 1-block-encoding of $f(\mathfrak{H})$
A	Upper-left n-qubit submatrix of a $(n+1)$-qubit random unitary matrix U
$\mathfrak{H}$	$A^\dagger A$, also called a H-RACBEM
$s_t(x)$	e^{-itx^2}
$g_t(x)$	an even polynomial approximation to $s_t(x)$, also denoted by $P(x, \Phi)$ with phase factor Φ
$f_t(x)$	$g_t(x^2)$, which is a polynomial approximation to e^{-itx}
$\mathbb{P}(\cdot)$	probability density function of random quantum circuits
$\mathbb{P}_{\text{exp}}(\cdot)$	probability density function associated with the noisy implementation of random quantum circuits
p_j	probability associated with the matrix element at the 0-th column and the j-th row of a unitary matrix V, i.e., $p_j := \lvert V_{j0}\rvert^2$
p_{ij}	probability associated with the matrix element at the i-th column and the j-th row of a unitary matrix V, i.e., $p_j := \lvert V_{ij}\rvert^2$
$p(U, x)$	noiseless bitstring probability of measuring $\mathcal{U}_{f,U}$ with outcome 0 in the ancilla qubit and an n-bit binary string x in the n system qubits (dependence on f is omitted)
$P(U)$	noiseless probability of measuring $\mathcal{U}_{f,U}$ with outcome 0 in the ancilla qubit, satisfying $P(U) = \sum_x p(U, x)$ (dependence on f is omitted)
$p_{\text{exp}}(U, x)$	bitstring probability of measuring the noisy implementation of $\mathcal{U}_{f,U}$ with outcome 0 in the ancilla qubit and an n-bit binary string x in the n system qubits (dependence on f is omitted)
$P_{\text{exp}}(U)$	probability of measuring the noisy implementation of $\mathcal{U}_{f,U}$ with outcome 0 with 0 in the ancilla qubit, satisfying $P_{\text{exp}}(U) = \sum_x p_{\text{exp}}(U, x)$ (dependence on f is omitted)

Table 7.1: Summary of notations used in this chapter.

7.6 Equivalence between minimal and standard QSVT circuits

The standard implementation of the QSVT circuit [67] uses one extra ancilla qubit, called the signal ancilla qubit. In this section, we show that when the system matrix A is block encoded with only one ancilla qubit, the signal ancilla qubit is no longer needed. Therefore the only ancilla qubit is due to the block encoding of A, and the circuit is called the minimal QSVT (mQSVT) circuit in Fig. 7.1. Furthermore, Fig. 7.1 removes all two-qubit and multi-qubit gates outside of the unitary U, which greatly simplifies the implementation for a given quantum device. We prove the equivalence between the mQSVT and the standard QSVT circuits in this section for completeness.

For any $(n+1)$-qubit unitary U, let the singular value decomposition of its upper-left n-qubit submatrix A be $A = W_1 \Sigma V_1^\dagger$. Following the cosine-sine decomposition (CSD), there exists n-qubit unitaries W_2, V_2 so that U can be decomposed as follows,

$$U = \begin{pmatrix} A & * \\ * & B \end{pmatrix} = \begin{pmatrix} W_1 & 0 \\ 0 & W_2 \end{pmatrix} \begin{pmatrix} \Sigma & S \\ -S & \Sigma \end{pmatrix} \begin{pmatrix} V_1 & 0 \\ 0 & V_2 \end{pmatrix}^\dagger,$$

where $S = \sqrt{I - \Sigma^2}$. This decomposition also implies any n-qubit non-unitary matrix A, up to a scaling factor, can in principle be block encoded using only one ancilla qubit.

Then, the unitary matrix representation of the quantum circuit in Fig. 7.1 is

$$\begin{aligned} \text{Mat. Rep.} = &\begin{pmatrix} V_1 & 0 \\ 0 & V_2 \end{pmatrix} \begin{pmatrix} e^{\mathrm{i}\varphi_0} I_n & 0 \\ 0 & e^{-\mathrm{i}\varphi_0} I_n \end{pmatrix} \prod_{j=1}^{d} \left[\begin{pmatrix} \Sigma & -S \\ S & \Sigma \end{pmatrix} \right. \\ &\left. \begin{pmatrix} e^{\mathrm{i}\varphi_{2j-1}} I_n & 0 \\ 0 & e^{-\mathrm{i}\varphi_{2j-1}} I_n \end{pmatrix} \begin{pmatrix} \Sigma & S \\ -S & \Sigma \end{pmatrix} \begin{pmatrix} e^{\mathrm{i}\varphi_{2j}} I_n & 0 \\ 0 & e^{-\mathrm{i}\varphi_{2j}} I_n \end{pmatrix} \right] \begin{pmatrix} V_1 & 0 \\ 0 & V_2 \end{pmatrix}^\dagger. \end{aligned}$$

Let K be the permutation matrix permuting the j-th and the $(N+j)$-th columns, and $V = \mathrm{diag}\{V_1,\ V_2\}$. The multiplicand is simplified as a direct sum of N 2-by-2 blocks upon conjugating $\widetilde{V} := VK$, i.e.

$$\widetilde{V}^\dagger \left(\text{Mat. Rep.}\right) \widetilde{V} = \bigoplus_{q=0}^{N-1} e^{\mathrm{i}\varphi_0 Z} \prod_{j=1}^{d} R_q e^{\mathrm{i}\varphi_{2j-1} Z} R_q^\top e^{\mathrm{i}\varphi_{2j} Z},$$

where

$$R_q = \begin{pmatrix} \sigma_q & -\sqrt{1-\sigma_q^2} \\ \sqrt{1-\sigma_q^2} & \sigma_q \end{pmatrix} = e^{\mathrm{i}\frac{\pi}{4} Z} e^{\mathrm{i}\arccos(\sigma_q) X} e^{-\mathrm{i}\frac{\pi}{4} Z}.$$

Let $W(x) := e^{\mathrm{i}\arccos(x) X}$, and

$$\widetilde{\varphi}_i = \begin{cases} \varphi_i + \frac{\pi}{4}, & i = 0 \text{ or } 2d, \\ \varphi_i + \frac{\pi}{2}, & i = 2, 4, \cdots, 2d-2, \\ \varphi_i - \frac{\pi}{2}, & i = 1, 3, \cdots, 2d-1. \end{cases}$$

The matrix representation is then

$$\widetilde{V}^\dagger \left(\text{Mat. Rep.}\right) \widetilde{V} = \bigoplus_{q=0}^{N-1} e^{\mathrm{i}\widetilde{\varphi}_0 Z} \prod_{j=1}^{2d} W(\sigma_q) e^{\mathrm{i}\widetilde{\varphi}_j Z}.$$

It is straightforward to show that the following mapping from $[-1,1]$ to SU(2)

$$x \mapsto e^{\mathrm{i}\widetilde{\varphi}_0 Z}\prod_{j=1}^{2d} W(x)e^{\mathrm{i}\widetilde{\varphi}_j Z} = \begin{pmatrix} P(x) & \mathrm{i}\sqrt{1-x^2}Q(x) \\ \mathrm{i}\sqrt{1-x^2}Q^*(x) & P^*(x) \end{pmatrix}$$

defines an even polynomial $P(x)$ of degree at most $2d$, and an odd polynomial $Q(x)$ of degree at most $2d-1$, so that $|P(x)|^2+(1-x^2)\,|Q(x)|^2=1$ holds for any $x\in[-1,1]$.

Then, the matrix representation of the quantum circuit is

$$\begin{aligned}\text{Mat. Rep.} &= V\begin{pmatrix} P(\Sigma) & \mathrm{i}\sqrt{I_n-\Sigma^2}Q(\Sigma) \\ \mathrm{i}\sqrt{I_n-\Sigma^2}Q^*(\Sigma) & P^*(\Sigma) \end{pmatrix}V^\dagger \\ &= \begin{pmatrix} P_{\mathrm{SV}}(A) & * \\ * & (P_{\mathrm{SV}}(A))^\dagger \end{pmatrix}.\end{aligned}$$

For example, when $g_t(x)$ is an even polynomial approximation to $s_t(x)=e^{-\mathrm{i}tx^2}$, we can define $g_t(x)=f_t(x^2)$, and the diagonal n-qubit submatrices are $(g_t)_{\mathrm{SV}}(A)=f_t(A^\dagger A)$ and $((g_t)_{\mathrm{SV}}(A))^\dagger=(f_t(A^\dagger A))^\dagger$ respectively.

This remarkably simple structure of the QSVT circuit is due to the use of 1-block-encoding. In general, if an n-qubit matrix A is block encoded in an $(n+m)$-qubit unitary U, the standard QSVT circuit has the structure in Fig. 2.3 (c). In particular, even when $m=1$, two CNOT gates are needed to implement each phase rotation. This introduces additional errors and can be practically cumbersome on near term devices the default two-qubit gate is not CNOT (e.g. $\sqrt{\mathrm{iSWAP}}$).

In the mQSVT circuit in Fig. 7.1, the phase factors $(\varphi_0,\cdots,\varphi_{2d})$ are determined by an optimization procedure that provides an even polynomial $g_t(x)$ satisfying $\|g_t(x)-s_t(x)\|_\infty\le\epsilon$ for some given precision parameter ϵ (see Section 4.3), and the evolution time t is encoded in the choice of phase factors. We then measure the top ancilla qubit and post-select on the 0 outcome of this measurement. This then ensures that the action on the lower n system qubits approximates the Hamiltonian evolution $e^{-\mathrm{i}t\mathfrak{H}}\,|0^n\rangle\approx f_t(\mathfrak{H})\,|0^n\rangle$, where $\mathfrak{H}=A^\dagger A$. Here $f_t(\mathfrak{H})$ is a matrix polynomial, and the approximation error in the operator norm is upper bounded by $\left\|f_t(\mathfrak{H})-e^{-\mathrm{i}t\mathfrak{H}}\right\|_2\le\epsilon$.

In the absence of quantum errors the probability of measuring the top ancilla qubit with outcome 0, i.e. the $P_t(U):=\|f_t(\mathfrak{H})\,|\psi\rangle\|^2$, will be close to 1. Specifically, by the triangle inequality, the probability of measuring the top ancilla qubit with

outcome 0 is lower bounded:

$$\begin{aligned} P_t(U) &= \left\| g_t(\Sigma) V^\dagger \left| 0^n \right\rangle \right\|_2^2 = \sum_{j=0}^{N-1} \left| g_t(\sigma_j) \right|^2 p_j \\ &= \left| 1 + \sum_{j=0}^{N-1} \left(\left(g_t(\sigma_j) - s_t(\sigma_j) \right) \overline{g_t(\sigma_j)} + s_t(\sigma_j) \overline{\left(g_t(\sigma_j) - s_t(\sigma_j) \right)} \right) p_j \right| \qquad (7.15) \\ &\geq 1 - 2\epsilon \sum_{j=0}^{N-1} p_j = 1 - 2\epsilon, \end{aligned}$$

where $p_j = |V_{0,j}|^2$.

We also find that the probability $p_t(U,x) = |\langle 0x | \mathcal{U}_{f_t,U} | 00^n \rangle|^2 \approx |\langle x| \exp(-\mathrm{i}t\mathfrak{H}) |0^n\rangle|^2$ will characterize the dynamics of the propagation from 0^n to x for any n-bit string $x \in \{0,1\}^n$. If the simulation time t is short, then $\exp(-\mathrm{i}t\mathfrak{H}) \approx I$, and $p_t(U,0^n)$ can be much larger than $p_t(U,x)$ for any bitstring $x \neq 0^n$. This issue will be particularly important when defining the 'heavy weight samples' in later discussions. Therefore we shall primarily focus on the case when $x \neq 0^n$.

As an illustrative example, Fig. 7.7 shows a quantum circuit implementing a 2-qubit matrix A encoded by a 3-qubit unitary matrix U. The construction uses only the basic gate set $\{\mathrm{U}_1, \mathrm{U}_2, \mathrm{U}_3, \mathrm{CNOT}\}$. In Section 4.3 we describe an optimization based method to obtain the phase factors for a relatively short time t to a small approximation error ϵ. In this example we set $t = 1$. To obtain a theoretical error bound at large t, we can use the phase factor concatenation technique in Section 7.8 to obtain the simulation at t from 2 to 10, and the error bound $\epsilon_t = \epsilon t^2$ is given by Theorem 7.8.1. Using these circuits, we may measure the outcome of the system qubits in the computational basis, and follow the dynamics of the probability $\sum_{x \neq 0^n} p_t(U,x)$, i.e. that of the quantum state moving away from the initial state $|0^n\rangle$. Fig. 7.7(b) shows that this agrees very well with the result using the exact dynamics $\sum_{x \neq 0^n} |\langle x| e^{-\mathrm{i}t\mathfrak{H}} |0^n\rangle|^2$. Furthermore, according to Eq. (7.15), the probability $P_t(U)$ in Fig. 7.7(c) satisfies the theoretical bounds $1 - 2\epsilon_t \leq P_t(U) \leq 1$ and is very close to 1.

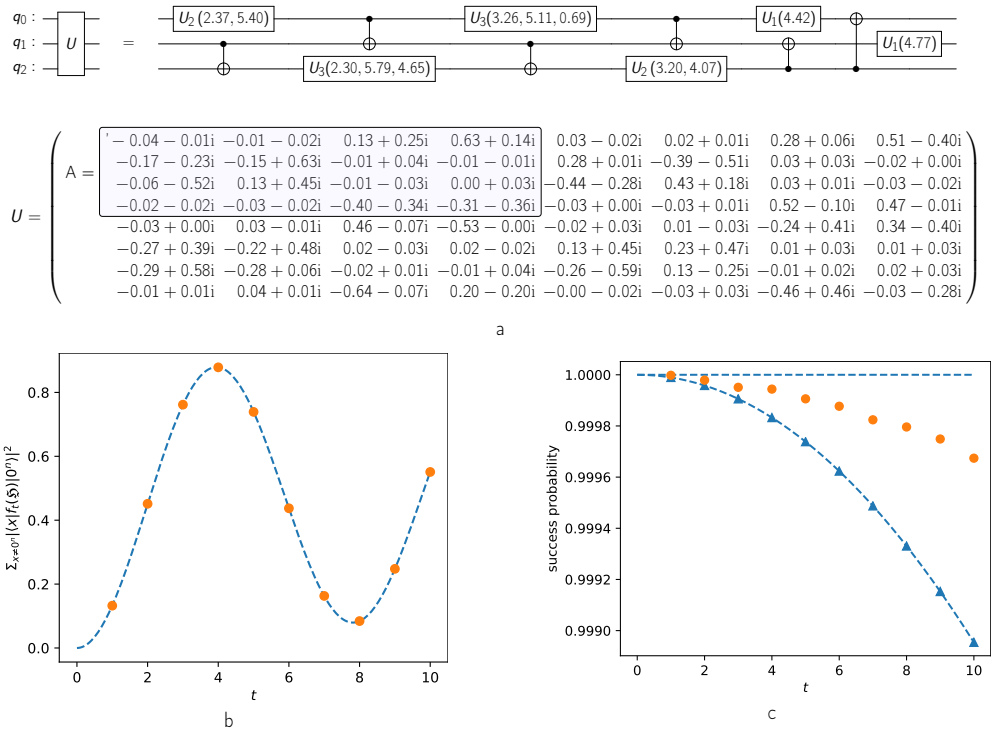


Figure 7.7: An illustrative implementation from the quantum Hamiltonian simulation benchmark. (a) Top: A 3-qubit random quantum circuit constructed from the basic gate set $\{\mathrm{U}_1, \mathrm{U}_2, \mathrm{U}_3, \mathrm{CNOT}\}$. Bottom: The 3-qubit unitary matrix representation U of the quantum circuit and its upper-left 2-qubit submatrix A (in the shaded area). The top qubit q_0 is the ancilla qubit for block encoding. (b) Dynamics of the evolution away from the initial condition $\sum_{x\neq 0^n} p_t(U, x)$ implemented using the mQSVT circuit, compared to the reference solution $\sum_{x\neq 0^n} |\langle x|e^{-it\mathfrak{H}}|0^n\rangle|^2$. (c) The probability $P_t(U)$ obtained from 10^6 noiseless measurements (orange dots) and theoretical bounds (blue dashed).

7.7 Optimization-based method for finding phase factors in the Hamiltonian simulation benchmark

In order to implement the Hamiltonian simulation benchmark at time t, we need to find the phase factors Φ^* that generates an even polynomial $P(x, \Phi^*)$ so that $\|P(x, \Phi^*) - s_t(x)\|_\infty \leq \epsilon$ for a sufficiently small ϵ. The optimization problem can be written as

$$\Phi^* = \underset{\Phi \in \mathbb{R}^{d+1}}{\arg\min} \frac{1}{\widetilde{d}} \sum_{k=1}^{\widetilde{d}} |P(x_k, \Phi) - s_t(x_k)|^2 . \tag{7.16}$$

It's important to note that this problem differs from the formalism introduced in Section 4.3, as the target function here is complex-valued, in contrast to being real-valued. This divergence invalidates the assumptions underlying the theoretical guarantee provided in Theorem 4.1.1, thereby escalating the complexity of the optimization challenge. Nevertheless, this optimization problem can still be effectively addressed numerically using a quasi-Newton method. Table 7.2 describes the approximation error for polynomials measured by $\|P(x, \Phi^*) - s_t(x)\|_\infty$ at simulation time $t = 1$, for polynomial degrees between 6 and 20. When the polynomial degree is 20, the approximation error is as small as 10^{-8}, which demonstrates the effectiveness of the optimization based method.

degree d	approximation error
6	5.543×10^{-03}
8	5.805×10^{-04}
10	5.230×10^{-06}
14	3.332×10^{-06}
18	9.535×10^{-08}
20	1.107×10^{-08}

Table 7.2: Approximation error $\left\| P(x, \Phi^*) - e^{-\mathrm{i}tx^2} \right\|_\infty$ at time $t = 1$ with different polynomial degrees d.

According to Fig. 7.6, there exists an 'optimal' simulation time $t^{\text{opt}} = 4.8096$, for which the threshold fidelity $\alpha^*(t^{\text{opt}}) \approx 0$ (the derivation is in Section 7.16). Table 7.3 describes the phase factor sequences that can be directly used in Fig. 7.1 to perform

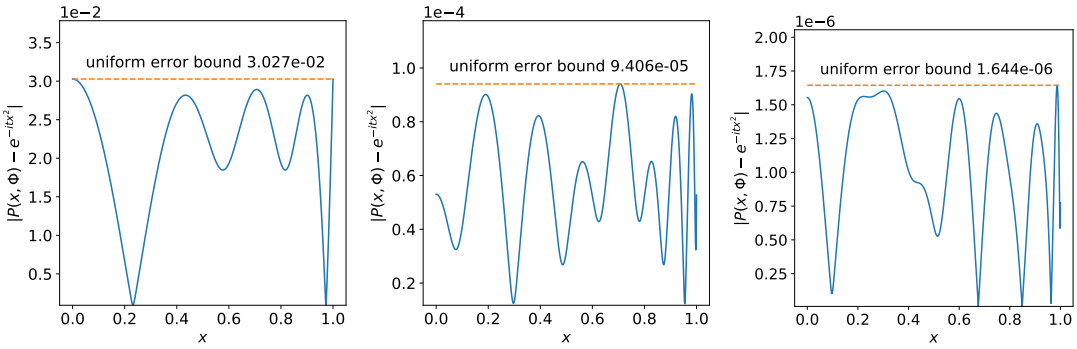
Hamiltonian simulation at time t^{opt}. In order to reach low (3.0×10^{-2}), medium (9.4×10^{-5}), and high (1.6×10^{-6}) accuracy, the degrees of the polynomial found by the optimization procedure are $10, 18, 26$, respectively. Fig. 7.8 further shows the pointwise approximate error on the interval $[0, 1]$ (the error on $[-1, 0]$ is the same due to the even parity). Compared to Table 7.2, in order to reach precision $\epsilon = 3.3 \times 10^{-6}$ at simulation time $t = 1$, the polynomial degree still needs to be 14. So even though t^{opt} is nearly 5 times larger, the polynomial degree only increases by less than twofold to reach similar accuracy. Since t^{opt} is still relatively small, this does not violate the 'no-fast-forwarding' theorem of Hamiltonian simulation [19].

approximation error $\left\|P(x,\Phi) - e^{-itx^2}\right\|_\infty = 3.027 \times 10^{-2}$						
φ_0	φ_1	φ_2	φ_3	φ_4	φ_5	φ_6
-2.7731963	2.7942520	-1.5707963	2.5930970	-1.5707963	-0.6434012	-1.5707963
φ_7	φ_8	φ_9	φ_{10}			
2.5930970	-1.5707963	2.7942520	-2.7731963			

approximation error $\left\|P(x,\Phi) - e^{-itx^2}\right\|_\infty = 9.406 \times 10^{-5}$						
φ_0	φ_1	φ_2	φ_3	φ_4	φ_5	φ_6
-2.7731963	2.8229351	-1.5707963	-2.5716144	-1.5707963	-3.1056796	-1.5707963
φ_7	φ_8	φ_9	φ_{10}	φ_{11}	φ_{12}	φ_{13}
-1.1677625	1.5707963	-0.6437954	1.5707963	-1.1677625	-1.5707963	-3.1056796
φ_{14}	φ_{15}	φ_{16}	φ_{17}	φ_{18}		
-1.5707963	-2.5716144	-1.5707963	2.8229351	-2.7731963		

approximation error $\left\|P(x,\Phi) - e^{-itx^2}\right\|_\infty = 1.644 \times 10^{-6}$						
φ_0	φ_1	φ_2	φ_3	φ_4	φ_5	φ_6
-1.5893341	-0.3207550	2.8668325	-2.9662972	-1.1921175	-0.4528806	1.5270366
φ_7	φ_8	φ_9	φ_{10}	φ_{11}	φ_{12}	φ_{13}
1.6658052	-0.2379487	-2.9130657	0.3245889	0.7863552	-1.3306612	-0.2863103
φ_{14}	φ_{15}	φ_{16}	φ_{17}	φ_{18}	φ_{19}	φ_{20}
-1.3306612	0.7863552	0.3245889	-2.9130657	-0.2379487	1.6658052	1.5270366
φ_{21}	φ_{22}	φ_{23}	φ_{24}	φ_{25}	φ_{26}	
-0.4528806	-1.1921175	-2.9662972	2.8668325	-0.3207550	-1.5893341	

Table 7.3: Phase factors for Hamiltonian simulation at time $t^{\text{opt}} = 4.8096$. The table lists three sets of phase factors with different approximation errors.



(a) (b) (c)

Figure 7.8: Point-wise approximation error of phase factors for Hamiltonian simulation at time $t^{\text{opt}} = 4.8096$. The corresponding sets of phase factors are listed in Table 7.3.

7.8 Concatenating phase factors for long time Hamiltonian simulation

Although simulation at very long time is certainly beyond the regime of near term applications, the unitarity of Hamiltonian simulation provides an alternative method to obtain phase factors. Specifically, given the phase factor sequence at some short time t, the phase factor sequence at a long simulation time rt can be easily constructed for some integer $r > 1$. The procedure, called phase factor concatenation, is given in Eq. (7.17), and the quality of the approximation is describe in Theorem 7.8.1.

Theorem 7.8.1. *Let $\Phi = (\phi_0, \cdots, \phi_d)$ be a set of phase factors so that $\|P(x,\Phi) - s_t(x)\|_\infty \leq \epsilon$. Given an integer $r \geq 1$, define*

$$\Phi^{(r)} := \left(\phi_0, \phi_1, \cdots, \phi_{d-1}, \underbrace{\phi_d + \phi_0, \phi_1, \cdots, \phi_{d-1}}_{\text{repeat } r-1 \text{ times}}, \phi_d \right). \tag{7.17}$$

Then,

$$\left\| P(x,\Phi^{(r)}) - s_{tr}(x) \right\|_\infty \leq r^2 \epsilon. \tag{7.18}$$

Proof. For simplicity, let $P(x) := P(x,\Phi)$ and $P^{(r)}(x) := P(x,\Phi^{(r)})$. According to the normalization condition,

$$(1-x^2)\,|Q(x)|^2 = 1 - P(x)\overline{P(x)} = (s_t(x) - P(x))\,\overline{P(x)} + s_t(x)\overline{(s_t(x) - P(x))}.$$

By the triangle inequality, $\left\|(1-x^2)\left|Q(x)\right|^2\right\|_\infty \leq 2\epsilon$.

Let ϵ_r be the approximation error of $P^{(r)}$, and let $Q^{(r)}$ be the complementing polynomial in as Theorem 2.3.3. Similarly, we have $\left\|(1-x^2)\left|Q^{(r)}(x)\right|^2\right\|_\infty \leq 2\epsilon_r$. By the construction of the phase factor sequence in Eq. (7.17), we have

$$P^{(r)}(x) = P^{(r-1)}(x)P(x) - (1-x^2)Q^{(r-1)}(x)\overline{Q(x)}.$$

This gives

$$\begin{aligned} P^{(r)}(x) - s_t^r(x) = &\left(P^{(r-1)}(x) - s_t^{r-1}(x)\right)P(x) + s_t^{r-1}(x)\left(P(x) - s_t(x)\right) \\ &- (1-x^2)Q^{(r-1)}(x)\overline{Q(x)}. \end{aligned}$$

Note that $s_t^r(x) = s_{tr}(x)$. By the triangle inequality, $\epsilon_r \leq \epsilon_{r-1} + \epsilon + 2\sqrt{\epsilon_{r-1}\epsilon} = (\sqrt{\epsilon_{r-1}} + \sqrt{\epsilon})^2$. Then, $\sqrt{\epsilon_r} \leq \sqrt{\epsilon_{r-1}} + \sqrt{\epsilon} \leq \cdots \leq r\sqrt{\epsilon}$. Therefore

$$\epsilon_r \leq r^2\epsilon, \tag{7.19}$$

which proves the theorem. □

Following the construction of Eq. (7.17), in order to perform Hamiltonian simulation at time rt, the length of the phase factor sequence increases by a factor of r, and the error grows at most quadratically with respect to r according to Theorem 7.8.1. However, from Tables 7.2 and 7.3 we observe that this estimate can be significantly improved if we can construct the phase factor sequence at time rt directly using the optimization based method. Even when Eq. (7.17) is used, numerical results in Fig. 7.7 also indicate that $\epsilon_r \leq r^2\epsilon$ is only an upper bound of the numerical error, which only grows linearly at least until $t = 10$.

7.9 Noise model

In the experimental setting, the density matrix after the application of the quantum circuit $\mathcal{U}_{f,U}$ can be written as

$$\varrho_{\text{exp}} = \alpha\left|\psi_{f,U}\right\rangle\left\langle\psi_{f,U}\right| + (1-\alpha)\chi_{f,U}. \tag{7.20}$$

Here, $\left|\psi_{f,U}\right\rangle := \mathcal{U}_{f,U}\left|0\right\rangle \otimes \left|\psi\right\rangle$ is the ideal quantum state generated by the exact implementation of the quantum circuit, and the operator $\chi_{f,U}$ is due to the noise channel. Under the global depolarizing noise model for the mQSVT circuit, we have $\chi_{f,U} = I/2^{n+1}$ and the diagonal entries of Eq. (7.20) then yield the probabilities of Eq. (7.5).

However, in practice $\chi_{f,U}$ may not be a scaled identity matrix, or even a diagonal matrix. If so, the system linear cross-entropy score (sXES) in Eq. (7.6) should be expressed more generally as

$$\mathrm{sXES}(U) = \sum_{x\neq 0^n} p(U,x)p_{\mathrm{exp}}(U,x) = \alpha \sum_{x\neq 0^n} p^2(U,x) + (1-\alpha)\sum_{x\neq 0^n} p(U,x)\,\langle x|\chi_{f,U}|x\rangle\,. \tag{7.21}$$

Now we can write

$$\sum_{x\neq 0^n} p(U,x)\,\langle x|\chi_{f,U}|x\rangle = \frac{1}{2^{n+1}}\sum_{x\neq 0^n} p(U,x) + \epsilon_\chi, \tag{7.22}$$

where ϵ_χ represents the effects of correlations between noise channels. Under the global depolarized noise channel we have $\epsilon_\chi = 0$. Refs. [23] and [8, Supplementary information IV.B] argue that when the noise $\langle x|\chi_{f,U}|x\rangle$ is uncorrelated with the signal $p(U,x)$, the statistical fluctuation error ϵ_χ can be of higher order, as a result of the concentration of measure phenomenon and Levy's lemma in high dimensional spaces [90].

On the other hand, even though each component U in the mQSVT circuit can be taken to be the same as the random circuit in the supremacy experiment of Ref. [8], the overall mQSVT circuit exhibits additional structure due to the presence of the R_z (phase) gates and the multiple repetitions of each U and $U^\dagger$ pair. Thus the global depolarized error model may not hold in practice. To overcome this difficulty, the randomized compilation method in Ref. [148] can be applied to the mQSVT circuits to convert all noise in the circuit into stochastic Pauli errors. The theory of error randomization can then be used to guarantee that the effect of these Pauli errors can be accurately modeled by global depolarization [23, 127]. Specifically, the method of [148] divides a universal gate set into a set of 'easy' gates and a set of 'hard' gates and then uses randomization applied to the easy gates to twirl the errors on the hard gates. Ref. [148] shows that for a set of elementary gates consisting of Clifford plus T gates, making the easy set equal to one-qubit Clifford operations allows one to tailor general multi-qubit noise into Pauli noise on the hard gates. In the mQSVT circuit, thanks to the full flexibility in generating the random circuit U, we may explicitly construct U so that most gates are taken directly from an easy gate set. This enables the twirling operations to be applied independently within all of the $U, U^\dagger$ blocks and thereby reduces the noise in each layer of gates in an mQSVT circuit into stochastic Pauli channels. Since the U circuits consist of random layers that generate a 2-design, multiple layers of this circuit implement an approximate 2-design. A 2-design twirls any errors into global depolarization (see also [103, 63, 29]), and so the overall error on a single U or $U^\dagger$ can be approximated by a global depolarizing channel [23]. The

repeated structure of U and $U^\dagger$ subroutines in a mQSVT means that an error that is randomized by U is then 'de-randomized' by $U^\dagger$. To mitigate this effect, it is possible to apply the randomized compilation method of Ref. [148] to each $U, U^\dagger$ independently. The impact of this type of effects has been studied in the setting of randomized mirror circuit benchmarks [127, 126, 125], which also use circuits with a $U, U^\dagger$ structure and employ a form of randomized compilation. These studies have explicitly shown that with this approach the overall error can still be modeled by a global depolarizing channel.

7.10 Structure of the probability space of measuring noisy random quantum circuits and sXES

There are two sources of randomness when measuring noisy random quantum circuits. The first is due to the random choice of U with probability density $\mathbb{P}(U)$. The second is due to the Monte Carlo nature of the quantum measurement. Specifically, given the choice of U and a noisy implementation of the quantum circuit, the probability to obtain $0x$ as the measurement outcome is $p_{\text{exp}}(U,x) := \langle 0x|\varrho_{\text{exp}}|0x\rangle$. The joint probability of U and the measurement outcome $0x$ is

$$\mathbb{P}_{\text{exp}}(U,x) = p_{\text{exp}}(U,x)\mathbb{P}(U).$$

Here for simplicity, we only focus on the measurement result whose ancilla qubit is measured with outcome 0. When $\mathbb{P}(U)$ is given by the Haar measure, the noise channel is depolarized, and the noisy (experimental) bitstring probability is

$$p_{\text{exp}}(U,x) = \alpha p(U,x) + \frac{1-\alpha}{2N}, \tag{7.23}$$

which is the convex combination of the exact bitstring probability and the uniform distribution [23].

The information of the circuit fidelity can then be encoded in the experimental average of various quantities. For example, the bitstring probability for nonzero bitstrings is given by

$$\begin{aligned}\mathbb{E}_{\text{exp}}\left(p(U,x); x \neq 0^n\right) &= \mathbb{E}\left(\sum_{x\neq 0^n} p(U,x)p_{\text{exp}}(U,x)\right) \\ &= \sum_{x\neq 0^n} \alpha\mathbb{E}\left(p(U,x)^2\right) + \frac{1-\alpha}{2N}\mathbb{E}\left(p(U,x)\right).\end{aligned} \tag{7.24}$$

Eq. (7.24) connects quantities evaluated from quantum experiments and classical computation on the left hand side and those from classical computations on the right hand side. The left-hand side is given by the system linear cross-entropy score (sXES) in Eq. (7.6), which can be evaluated from multiplying the bitstring frequency $p_{\text{exp}}(U,x)$ obtained from the quantum experiment and the bitstring probability $p(U,x)$ computed classically. The quantities on the right-hand side can be evaluated fully classically. The circuit fidelity α is then the only unknown and can be solved for by substituting the quantum experimental and classically computed quantities into Eq. (7.7) of the main text.

7.11 Estimating circuit fidelity from quantum unitary evolution score

The experimental average of the probability of measuring the ancilla qubit with outcome 0 is

$$\begin{aligned}\mathbb{E}_{\text{exp}}\left(P(U)\right) &= \mathbb{E}\left(\sum_x P(U)p_{\text{exp}}(U,x)\right) = \mathbb{E}\left(P(U)P_{\text{exp}}(U)\right)\\ &= \alpha\mathbb{E}\left(P(U)^2\right) + \frac{1-\alpha}{2}\mathbb{E}\left(P(U)\right).\end{aligned} \tag{7.25}$$

Here $P(U) := \sum_x p(U,x)$, and $P_{\text{exp}}(U)$ is the probability which can be approximately determined by the bit frequency of the measurement outcome in the experiment. Rearranging the terms in Eq. (7.25), the circuit fidelity can be alternatively estimated via

$$\alpha = \frac{\mathbb{E}\left(P(U)P_{\text{exp}}(U)\right) - \frac{1}{2}\mathbb{E}\left(P(U)\right)}{\mathbb{E}\left(P(U)^2\right) - \frac{1}{2}\mathbb{E}\left(P(U)\right)}. \tag{7.26}$$

From Eq. (7.15), we have $P(U) \in [1-2\epsilon, 1]$. Hence a lower and upper bound on the fidelity follows:

$$\underline{\alpha} := \frac{2\left(1-2\epsilon\right)\mathbb{E}\left(P_{\text{exp}}(U)\right) - 1}{1+2\epsilon} \leq \alpha \leq \frac{2\mathbb{E}\left(P_{\text{exp}}(U)\right) - (1-2\epsilon)}{1-8\epsilon} =: \overline{\alpha}. \tag{7.27}$$

The difference between the upper and lower bound is

$$\overline{\alpha} - \underline{\alpha} \leq 16\epsilon + \mathcal{O}(\epsilon^2). \tag{7.28}$$

Therefore, $\lim_{\epsilon\to 0}(\overline{\alpha} - \underline{\alpha}) = 0$ and the derived bounds are tight. Let us choose the form of the estimate as ϵ-independent

$$\alpha_{\text{QUES}} := 2\mathbb{E}\left(P_{\text{exp}}(U)\right) - 1 \in [\underline{\alpha}, \overline{\alpha}]. \tag{7.29}$$

Then, $|\alpha_{\text{QUES}} - \alpha| \leq \overline{\alpha} - \underline{\alpha} \leq 16\epsilon + \mathcal{O}(\epsilon^2)$. Furthermore, the estimate can be determined using only the experimentally measurable quantity $P_{\text{exp}}(U)$ and is independent of the classical computation of $P(U)$, which may be hard to evaluate for large n. This remarkable fact, namely the evaluation of circuit fidelity without any classical computation, arises from the approximate implementation of Hamiltonian simulation of the overall circuit. Since only one ancilla qubit is measured, the QUES defined in Eq. (7.29) cannot entirely capture whether the circuit is implemented correctly. However, when the assumption that the noise channel is depolarized and when the polynomial approximation to $s_t(x)$ is sufficiently accurate, α_{QUES} provides a very good estimate to the circuit fidelity.

7.12 Algorithm for constructing random quantum circuits and numerical convergence to Haar measure

In order to theoretically analyze the circuit fidelity, we need the additional assumption that $\mathbb{P}(U)$ is the Haar measure. This has the advantage that several terms in Eq. (7.7) can be evaluate analytically. Using the Haar measure, the statistics of an ensemble of random Hamiltonians is much simplified and can be computed by the statistics of the truncation matrix of Haar unitaries [161, 122, 106, 45, 46]. Details of the statistics are given in Section 7.17. Furthermore, if U is Haar-distributed, then the noise effect of directly sampling U is well captured by a fully depolarized error channel, due to the nearly maximal entanglement in the output state [23, 8]. The need to choose an appropriate circuit depth ℓ such that the circuit statistics approximate those of Haar unitaries motivates an investigation of the statistics of Haar random quantum circuits of finite number of qubits.

Algorithm 7.12.1 Constructing random quantum circuits

Input: Coupling map $G = \langle V, E\rangle$ where V is the set of n qubits, E is the set of qubit pairs on which CNOT is available, basic gates $\Gamma = \{\text{U1}, \text{U2}, \text{U3}, \text{CNOT}\}$, the number of total one-qubit gates g_1, and the density of one-qubit gates $p_1 \in (0, 1)$.

1: Set the number of two-qubit gates to $g_2 = \lceil \frac{1-p_1}{2p_1} g_1 \rceil$.
2: Set the maximal number of two-qubit gates in each layer to $y_2 = \lceil \frac{1-p_1}{2} n \rceil$.
3: Set $m_1 = m_2 = 0$, initialize an empty quantum circuit $\mathcal{C}$.
4: **while** $m_1 \leq g_1$ **and** $m_2 \leq g_2$ **do**

Draw $x_2 \leq y_2$ pairs of qubits from E so that each pair (q_1, q_2) and its permutation (q_2, q_1) are not selected in the previous layer. The choice of x_2 also satisfies $m_2 + x_2 \leq g_2$.

Draw $x_1 = \min\{n - 2x_2, g_1 - m_1\}$ one-qubit gates uniformly at random from $\Gamma\backslash\{\text{CNOT}\}$ and act them on the rest of qubits in this layer.

Update the numbers of one- and two-qubit gates, $m_1 \leftarrow m_1 + x_1$ and $m_2 \leftarrow m_2 + x_2$.

end while

if $m_1 < g_1$ **then**

Append layers of random $g_1 - m_1$ one-qubit gates sampled uniformly at random from $\Gamma\backslash\{\text{CNOT}\}$.

else if $m_2 < g_2$ **then**

Append layers of $g_2 - m_2$ CNOT gates acting on random operands.

end if

Output: A random quantum circuit $\mathcal{C}$ with g_1 one-qubit gates and g_2 two-qubit gates.

We first construct random quantum circuits by using the algorithm given in Algorithm 7.12.1. It follows a similar recipe in Ref. [50]. We set the basic one-qubit gates to U1, U2 and U3 gates. Up to a global phase factor, the U3 gate is

$$\mathrm{U}_3(\theta, \phi, \lambda) = R_z(\phi + 3\pi)R_x(\pi/2)R_z(\theta + \pi)R_x(\pi/2)R_z(\lambda),$$

which is a generic single-qubit operation parameterized by three Euler angles. The U1 and U2 gates are defined by restricting to one or two Z-rotation angles respectively, i.e.

$$\mathrm{U}_1(\lambda) = R_z(\lambda), \quad \mathrm{U}_2(\phi, \lambda) = R_z(\phi + \pi/2)R_x(\pi/2)R_z(\lambda - \pi/2).$$

Taken together with the CNOT gate, these form a continuously parameterized gate set that is universal.

Although we specify the choice of one-qubit gates and the use of the CNOT gate here, Algorithm 7.12.1 can be directly generalized to an arbitrary basic gate set. The random quantum circuit generated by the algorithm respects the architecture of a quantum computer. In practice, we set the density of one-qubit gates to $p_1 = 0.5$. Then, for an n-qubit random quantum circuit with ℓ layers, the number of one-qubit gates is $g_1 = \frac{\ell n}{2}$ and that of two-qubit gates is $g_2 = \frac{\ell n}{4}$.

To measure the numerical convergence of random circuits to the Haar measure, we first summarize some of the statistical properties of the Haar measure. Given an n-qubit Haar-distributed unitary U, we denote $p_{ij} := |U_{ij}|^2$. As a special case of the more general Theorem 7.17.1 (to be presented in Section 7.17), the p_{ij}'s are identically Beta-distributed.

Theorem 7.12.1. *The probability density of p_{ij} is* Beta$(1, N-1)$*,*

$$\mathbb{P}(p_{ij}) = (N-1)(1-p_{ij})^{N-2}\mathbb{1}_{0\le p_{ij}\le 1}.$$

Proof. Let the submatrix of interest be the upper left 1-by-1 block, namely, a single matrix element $a := U_{00}$. Note that $p_{00} := |a|^2$. Then, Eq. (7.42) indicates that the probability density of a is

$$\mathbb{P}(a) \propto (1-p_{00})^{N-2}\,\mathbb{1}_{0\le p_{00}\le 1}.$$

The polar decomposition of the complex number $a = re^{\mathrm{i}\theta}$ yields the Jacobian $\mathrm{d}a = r\,\mathrm{d}r\,\mathrm{d}\theta \propto \mathrm{d}p_{00}\,\mathrm{d}\theta$. Then, integrating with respect to $\mathrm{d}\theta$, the marginal distribution of p_{00} is

$$\mathbb{P}(p_{00}) = (N-1)\,(1-p_{00})^{N-2}\,\mathbb{1}_{0\le p_{00}\le 1}.$$

This is the Beta$(1, N-1)$ distribution. When $i \neq 0$ or $j \neq 0$, let K_1 be the matrix permuting the i-th row and the 0-th row by left multiplication, and let K_2 be the matrix permuting the j-th column and the 0-th column by right multiplication. Then, $\widetilde{U} := K_1 U K_2$ is Haar distributed by the bi-invariance of the Haar measure. Furthermore, $\widetilde{U}_{00} = U_{ij}$. Therefore, the previous proof shows that p_{ij} is also Beta$(1, N-1)$ distributed. □

Note that in the limit $N \gg 1$, the distribution of p_{ij} is well approximated by the exponential distribution Exp(N), a.k.a. the Porter-Thomas distribution derived in [23]. The statistics follows straightforward computation by integrating with respect to the probability density.

Theorem 7.12.2. *Let $M_k := \sum_{i=0}^{N-1} p_{ij}^k$ be the k-th moment, $S := \sum_{i=0}^{N-1} -p_{ij}\ln(p_{ij})$ be the entropy. Their averages with respect to the Haar distribution take the form*

$$M_k^{\mathrm{Haar}} := \mathbb{E}\,(M_k) = \prod_{i=1}^{k-1}\frac{1+i}{N+i}, \quad S^{\mathrm{Haar}} := \mathbb{E}\,(S) = \sum_{i=2}^{N} i^{-1}.$$

The variance of the k-th moment $V_k^{\mathrm{Haar}} := \sum_{i=0}^{N-1}\mathrm{Var}(p_{ij}^k)$ is

$$V_k^{\mathrm{Haar}} = \left(\frac{1}{N}\binom{2k}{k} + \frac{N-1}{N}\right)\prod_{i=k}^{2k-1}\frac{N-k+i}{N+i} - 1.$$

We remark that in the limit $N \gg 1$, $M_k^{\mathrm{Haar}} \approx \frac{k!}{N^{k-1}}$ and $S^{\mathrm{Haar}} \approx \ln(N) + \gamma - 1$ where γ is Euler's constant. The asymptotic results are the same as those derived in [23]. The variance is asymptotically $V_k^{\mathrm{Haar}} \approx \frac{1}{N}\left(\binom{2k}{k} - 1\right)$. From the variance,

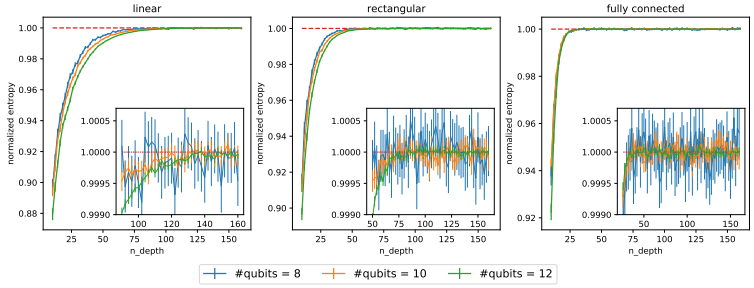
we conclude two important features about the statistics. Given N, the variance (i.e. fluctuation) increases with respect to the order of the moment. For each moment, the statistics becomes concentrated as N increases, namely the variance V_k^{Haar} vanishes as $N \to \infty$. By Taylor expansion, the entropy has the same concentrated behavior which can be numerically observed in Fig. 7.9.

Fig. 7.9 presents the statistics of random quantum circuits for several different structures of the circuit coupling map and shows that for all three coupling maps studied in the main text, the distribution of circuits of sufficient depth converges to the Haar measure. In Fig. 7.9(a), we plot the normalized entropy S/S^{Haar}. The figure shows that the entropy converges to that of Haar measure, i.e., $S/S^{\text{Haar}} \to 1$ after the circuit depth of U increases beyond specific values that depend only weakly on the number of qubits n. Since the random quantum circuit is constructed by combining layers of random one- and two-qubit gates, we test the convergence of random quantum circuits for different coupling maps, thereby varying the qubit pairs on which the two-qubit gates can act. The numerical results in Fig. 7.9(a) show that coupling maps with greater connectivity converge significantly faster to the Haar measure. We attribute this to the larger number of possible allocations of two-qubit gates enhancing state entanglement within the system and thereby leading to faster mixing of information.

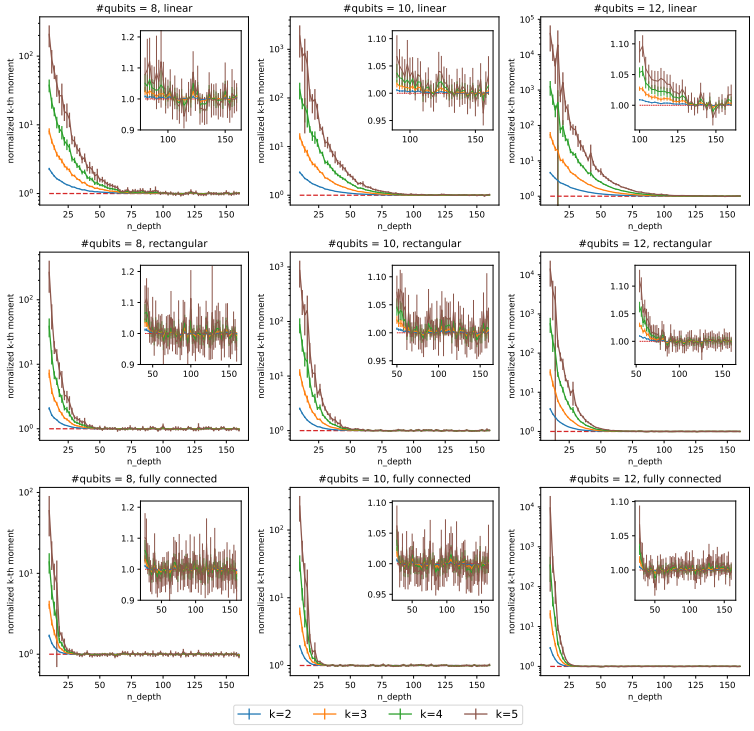
In addition to showing the convergence in terms of circuit entropy, we also quantify the convergence to the Haar measure for the first five moments in Fig. 7.9(b). The minimal depth to achieve approximate Haar random circuits deduced from the convergence in moments is highly consistent with that derived from the convergence in entropy. We list the depth used in the computation of the sXES in Table 7.4.

coupling map	n (number of system qubits)		
	7	9	11
linear	140	160	160
rectangular	76	94	100
fully connected	60	60	60

Table 7.4: Depth for random quantum circuits used in the computation of the system linear cross-entropy score. Each depth is chosen so that both entropy and moments are close to these derived from the Haar measure.



(a)



(b)

Figure 7.9: Convergence to the Haar measure. (a) Convergence in terms of entropy. Normalized entropy S/S^{Haar} as a function of the depth for random quantum circuits with different number of system qubits and coupling map. (b) Convergence in terms of moments. Each panel is the first five normalized moments M_k/M_k^{Haar} as a function of the depth for random quantum circuits with different number of system qubits and coupling map. The convergence of curves to the dashed line at 1 shows that the random quantum circuit with a modest circuit depth can well approximate the Haar measure. Error bars correspond to the 95% confidence interval estimated from ~ 1000 circuit instances.

7.13 Circuit fidelity from the system linear cross-entropy score

The system linear cross-entropy score is based on Hamiltonian simulation. Note that in the circuit fidelity of Eq. (7.7), only the terms involved the system linear cross entropy sXES in the numerator contain quantities that must be experimentally evaluated. All other terms can be simplified by using the statistical property of an ensemble of random matrices inherited from the Haar measure of U in Section 7.17 (in particular Theorem 7.17.7).

The procedure of computing the system linear cross-entropy score can be summarized as follows.

1. Draw quantum circuits U_i's approximately from the Haar measure at random.
2. For each U_i, build the mQSVT circuit for the Hamiltonian simulation benchmark, and measure all qubits to count the bitstring frequencies of $0x$ where $x \in \{0,1\}^n$. The bitstring frequency is an estimate to $p_{\exp}(U_i, x)$. Furthermore, the sum of bitstring frequencies for all x's is an estimate to $P_{\exp}(U_i) = \sum_{x\in\{0,1\}^n} p_{\exp}(U_i, x)$.
3. For each U_i and bitstring $0x$, compute the noiseless bitstring probability $p(U_i, x)$ on classical computers.
4. Compute estimates of the fidelity according to Eqs. (7.7) and (7.8).

We list the circuit fidelity estimated by different methods in Table 7.5. The agreement shows the consistency of the quantum Hamiltonian simulation benchmark. Here, the theoretical reference value is estimated from the depolarization noise model. Given U with a total of g_1 one-qubit gates and g_2 two-qubit gates, the value $\alpha_{\text{ref}} := (1-r_1)^{2d(g_1+1)}(1-r_2)^{2dg_2}$ follows approximately assuming each quantum error fully mixes the quantum state.

7.14 Classical hardness of sXHOG

Definition 7.14.1 (sXHOG, or System Linear Cross-entropy Heavy Output Generation)**.** *Given as input a number $b > 1$, a random $(n+1)$-qubit unitary U, and the mQSVT circuit for the Hamiltonian simulation benchmark with sufficiently small*

two-qubit gate error rate r_2	QSVT degree parameter $2d$					
	6	8	10	14	18	20
4.00×10^{-5}	0.95	0.92	0.90	0.85	0.80	0.79
	0.92	0.90	0.87	0.83	0.79	0.76
	0.93	0.92	0.89	0.86	0.82	0.80
1.30×10^{-4}	0.78	0.71	0.67	0.55	0.45	0.44
	0.77	0.70	0.65	0.54	0.46	0.42
	0.81	0.76	0.69	0.61	0.53	0.48
2.20×10^{-4}	0.65	0.56	0.50	0.36	0.26	0.25
	0.64	0.55	0.48	0.36	0.26	0.23
	0.70	0.62	0.54	0.43	0.34	0.28
3.10×10^{-4}	0.54	0.43	0.38	0.24	0.15	0.15
	0.54	0.43	0.35	0.23	0.15	0.12
	0.60	0.51	0.41	0.30	0.22	0.17
4.00×10^{-4}	0.45	0.34	0.28	0.16	0.087	0.090
	0.45	0.34	0.26	0.15	0.089	0.068
	0.52	0.42	0.32	0.21	0.14	0.10

Table 7.5: Circuit fidelity estimated from quantum Hamiltonian simulation benchmark. The total number of qubits is 8, namely, there are 7 system qubits and 1 ancilla qubit. The coupling map is linear. In each cell of the table, the top data is estimated from sXES, the middle data is the theoretical reference value, and the bottom data is estimated from QUES.

approximation error ϵ*, output nonzero bitstrings* $x_1, x_2, \cdots, x_k \in \{0,1\}^n \backslash \{0^n\}$ *so that*

$$\frac{1}{k}\sum_{j=1}^{k} p(U, x) \geq b \times 2^{-n}. \tag{7.30}$$

The classical hardness of the XEB experiment is justified by reducing the XHOG problem to a complexity assumption referred to as Linear Cross-entropy Quantum Threshold Assumption (XQUATH)[4]. Similarly, the hardness of the sXHOG problem can be reduced to an assumption that we refer to by analogy as sXQUATH.

Definition 7.14.2 (sXQUATH, or System Linear Cross-entropy Quantum Threshold Assumption)**.** *Given a random $(n+1)$-qubit unitary U, and the mQSVT circuit for the Hamiltonian simulation benchmark with sufficiently small approximation error ϵ, for a uniformly random $x \in \{0,1\}^n \backslash \{0^n\}$, there is no polynomial-time classical algorithm that produces an estimate p of $p_x := p(U,x)$ so that*

$$\mathbb{E}\left((p_x - p)^2\right) = \mathbb{E}\left((p_x - 2^{-n})^2\right) - \Omega(2^{-3n}).$$

Here, the expectation is taken over random circuits U, the internal randomness of the algorithm, and the random bitstring x.

The reduction of the XHOG problem is given in the following theorem, which is directly parallel to that in [4, Theorem 1].

Theorem 7.14.3 (Classical hardness of sXHOG)**.** *Assuming sXQUATH, no polynomial-time classical algorithm can solve the XHOG problem in Definition 7.14.1 with probability $s > \frac{1}{2} + \frac{1}{2b}$, and*

$$k \geq \frac{1}{((2s-1)b-1)(b-1)}.$$

Proof. Suppose that A is a classical algorithm solving sXHOG in Definition 7.14.1 with a success probability s as stated in the theorem. Given U and the mQSVT circuit as the input of A, it outputs $\mathsf{S} := \{x_i \neq 0^n : i = 1, \cdots, k\}$. When A successfully solves the sXHOG problem, the set S satisfies Eq. (7.30). Specifically, let $x \in \{0,1\}^n \backslash \{0^n\}$ be a bitstring sampled uniformly at random. We now construct an algorithm to produce an estimate p of $p(U,x)$. Given such a bitstring x, the algorithm outputs an estimate $p = b2^{-n}$ if $x \in \mathsf{S}$ and $p = 2^{-n}$ if $x \notin \mathsf{S}$.

Consider a random variable

$$X(U,x) := \left(p(U,x) - 2^{-n}\right)^2 - \left(p(U,x) - p\right)^2 = \left(2p(U,x) - (p + 2^{-n})\right)\left(p - 2^{-n}\right).$$

Here, the randomness comes from the uniformly random bitstring x, the random unitary U and its corresponding mQSVT circuit, and whether the classical algorithm A succeeds. We write them explicitly as the subscript of the expectation. Furthermore, we denote by $\mathsf{S}_U^{(s)} := \mathsf{S}$ when A succeeds, and $\mathsf{S}_U^{(f)} := \mathsf{S}$ when A fails. Let $\mathbb{1}_E$ be the indicator function which gives 1 if the condition E is satisfied and gives 0 otherwise.

According to the algorithm,

$$\begin{aligned}
&\mathbb{E}_{x,U}\left(X(U,x)\mathbb{1}_{x\in\mathsf{S}_U^{(s)}}|\mathsf{A}\text{ succeeded}\right)\\
&=2\cdot 2^{-n}(b-1)\cdot\mathbb{E}_{x,U}\left(p(U,x)\mathbb{1}_{x\in\mathsf{S}_U^{(s)}}\middle|\mathsf{A}\text{ succeeded}\right)+2^{-2n}(1-b^2)\mathbb{E}_{x,U}\left(\mathbb{1}_{x\in\mathsf{S}_U^{(s)}}\right),\\
&\mathbb{E}_{x,U}\left(X(U,x)\mathbb{1}_{x\in\mathsf{S}_U^{(f)}}\middle|\mathsf{A}\text{ failed}\right)\\
&=2\cdot 2^{-n}(b-1)\cdot\mathbb{E}_{x,U}\left(p(U,x)\mathbb{1}_{x\in\mathsf{S}_U^{(f)}}\middle|\mathsf{A}\text{ failed}\right)+2^{-2n}(1-b^2)\mathbb{E}_{x,U}\left(\mathbb{1}_{x\in\mathsf{S}_U^{(f)}}\right)\\
&\geq 2^{-2n}(1-b^2)\mathbb{E}_{x,U}\left(\mathbb{1}_{x\in\mathsf{S}_U^{(f)}}\right).
\end{aligned}\tag{7.31}$$

Furthermore, $X(U,x)\equiv 0$ if $x\notin\mathsf{S}$, regardless of whether A succeeds or not. By the law of total expectation,

$$\begin{aligned}
&\mathbb{E}_{x,U,\mathsf{A}}\left(X(U,x)\right)=s\cdot\mathbb{E}_{x,U}\left(X(U,x)|\mathsf{A}\text{ succeeded}\right)+(1-s)\cdot\mathbb{E}_{x,U}\left(X(U,x)|\mathsf{A}\text{ failed}\right)\\
=&s\cdot\mathbb{E}_{x,U}\left(X(U,x)\mathbb{1}_{x\in\mathsf{S}_U^{(s)}}\middle|\mathsf{A}\text{ succeeded}\right)+(1-s)\cdot\mathbb{E}_{x,U}\left(X(U,x)\mathbb{1}_{x\in\mathsf{S}_U^{(f)}}\middle|\mathsf{A}\text{ failed}\right)\\
\geq&s\cdot\left(2\cdot 2^{-n}(b-1)\mathbb{E}_{x,U}\left(p(U,x)\mathbb{1}_{x\in\mathsf{S}_U^{(s)}}|\mathsf{A}\text{ succeeded}\right)+2^{-2n}(1-b^2)\mathbb{E}_{x,U}\left(\mathbb{1}_{x\in\mathsf{S}_U^{(s)}}\right)\right)\\
&+(1-s)\cdot 2^{-2n}(1-b^2)\mathbb{E}_{x,U}\left(\mathbb{1}_{x\in\mathsf{S}_U^{(f)}}\right).
\end{aligned}\tag{7.32}$$

Here

$$\mathbb{E}_{x,U}\left(p(U,x)\mathbb{1}_{x\in\mathsf{S}_U^{(s)}}|\mathsf{A}\text{ succeeded}\right)=\frac{k}{2^n-1}\mathbb{E}_U\left(\frac{1}{k}\sum_{x\in\mathsf{S}_U^{(s)}}p(U,x)\right)\geq\frac{bk2^{-n}}{2^n-1}.$$

Note that $\mathsf{S}_U^{(s)}$ and $\mathsf{S}_U^{(f)}$ are sets of k distinct bitstrings. Following that x is uniformly distributed, we have

$$\mathbb{E}_{x,U}\left(\mathbb{1}_{x\in\mathsf{S}_U^{(s)}}\right)=\mathbb{E}_U\left(\mathbb{E}_x\left(\mathbb{1}_{x\in\mathsf{S}_U^{(s)}}\right)\right)=\frac{k}{2^n-1}$$

and

$$\mathbb{E}_{x,U}\left(\mathbb{1}_{x\in\mathsf{S}_U^{(f)}}\right)=\mathbb{E}_U\left(\mathbb{E}_x\left(\mathbb{1}_{x\in\mathsf{S}_U^{(f)}}\right)\right)=\frac{k}{2^n-1}.$$

Then

$$\begin{aligned}\mathbb{E}_{x,U,\mathsf{A}}\left(X(U,x)\right) &\geq \frac{2^{-2n}}{2^n-1}\left(ks\cdot(b-1)^2+k(1-s)\cdot(1-b^2)\right)\\ &\geq 2^{-3n}k\left((2s-1)b-1\right)(b-1)=\Omega(2^{-3n})\end{aligned} \tag{7.33}$$

when $k \geq \frac{1}{((2s-1)b-1)(b-1)}$. This violates sXQUATH and thereby proves the classical hardness of sHOG. □

7.15 Circuit fidelity and sXHOG

In this section we demonstrate that the success of sXHOG can be verified by experimental evaluation of the circuit fidelity. Due to the relation between the circuit fidelity and QUES in Section 7.13, it means that the success of sXHOG can be verified by QUES, which does not involve any classical computation.

First, since we are only interested in the measurement outcome whose ancilla qubit returns 0, we normalize the bitstring probability as a conditional probability

$$p_{\text{exp}}(U,x|\text{ancilla}=0) := p_{\text{exp}}(U,x)/P_{\text{exp}}(U). \tag{7.34}$$

The physical interpretation of the normalization of the bitstring probability is to discard the measurement result whose ancilla is measured with 1. We also remark that when the Hamiltonian simulation benchmark circuit is sufficiently accurate, we have $P(U)\approx 1$, and it is not necessary to normalize the noiseless bitstring probability in Eq. (7.30).

The probability that the experimental measurement on the ancilla qubit outputs 0 is

$$\mathbb{P}_0 := \sum_{x\in\{0,1\}^n}\int p_{\text{exp}}(U,x)\,\mathrm{d}U = \sum_{x\in\{0,1\}^n}\int \alpha p(U,x)+\frac{1-\alpha}{2N}\,\mathrm{d}U = \frac{1+\alpha}{2}\approx P_{\text{exp}}(U).$$

Given the circuit fidelity α, we denote the conditional probability density as

$$\mathbb{Q}_\alpha(U,x) := \frac{1}{\mathbb{P}_0}p_{\text{exp}}(U,x),$$

and the corresponding expectation is denoted as $\mathbb{E}_{\mathbb{Q}_\alpha}(\cdot)$. Then, the conditional average bitstring probability, which is directly related to the parameter b in determining

the sXHOG problem as

$$\begin{aligned} b(\alpha) &:= N\mathbb{E}_{\mathbb{Q}_\alpha}\left(\sum_{x\neq 0^n} p(U,x)\right) = \frac{\mathbb{E}\left(\mathrm{sXES}(U)\right)}{\mathbb{P}_0} \\ &= \left(1+\frac{\alpha(2-5\mathcal{H}_1+4\mathcal{H}_2)-\mathcal{H}_1}{\alpha+1}\right)+\mathcal{O}\left(\frac{1}{N}\right). \end{aligned} \tag{7.35}$$

The last equality is derived using results in Section 7.17. Here

$$\mathcal{H}_1 = \int \mathbb{P}^{(2)}_{\mathrm{eig}}(\lambda_1,\lambda_2)\cos\left(t\left(\lambda_1-\lambda_2\right)\right)\,\mathrm{d}\lambda_1\,\mathrm{d}\lambda_2, \tag{7.36}$$

and

$$\mathcal{H}_2 = \int \mathbb{P}^{(4)}_{\mathrm{eig}}(\lambda_1,\lambda_2,\lambda_3,\lambda_4)\cos\left(t\left(\lambda_1-\lambda_2+\lambda_3-\lambda_4\right)\right)\,\mathrm{d}\lambda_1\,\mathrm{d}\lambda_2\,\mathrm{d}\lambda_3\,\mathrm{d}\lambda_4 \tag{7.37}$$

are cosine transformations of $\mathbb{P}^{(2)}_{\mathrm{eig}}$ and $\mathbb{P}^{(4)}_{\mathrm{eig}}$, which are the 2-marginal and the 4-marginal distribution of eigenvalues corresponding to the ensemble of random Hermitian matrices, respectively. The values of $\mathcal{H}_1$ and $\mathcal{H}_2$ can be evaluated on classical computers according to Theorem 7.17.5 and Algorithm 7.17.1.

Thus for large n, we have

$$b(\alpha) \approx 1+\frac{\alpha(2-5\mathcal{H}_1+4\mathcal{H}_2)-\mathcal{H}_1}{\alpha+1} =: 1+\frac{\gamma(\alpha-\alpha^*)}{\alpha+1}. \tag{7.38}$$

Here $\gamma = 2-5\mathcal{H}_1+4\mathcal{H}_2$ and $\alpha^* = \mathcal{H}_1/\gamma$. When $\mathcal{H}_1, \mathcal{H}_2$ are sufficiently small, $b(\alpha)$ is monotonically increasing.

The hardness of classical spoofing also requires $b(\alpha) \geq 1$ (see Theorem 7.14.3). Thus, we define the threshold $\alpha^* := \frac{\mathcal{H}_1}{2-5\mathcal{H}_1+4\mathcal{H}_2}$ be the fidelity so that $b(\alpha^*) = 1$. To achieve supremacy, the fidelity is required to satisfy $\alpha \geq \alpha^*$. To see the existence of the threshold fidelity, let us consider a fully contaminated noise where $\alpha = 0$. Then, the average bitstring probability is

$$\mathbb{E}_{\mathbb{Q}_0}\left(\sum_{x\neq 0^n} p(U,x)\right) = \frac{1}{N}\left(1-\mathbb{E}\left(p(U,0^n)\right)\right) \Rightarrow b(\alpha)|_{\alpha=0} = 1-\mathbb{E}\left(p(U,0^n)\right) \leq 1. \tag{7.39}$$

By continuity, a threshold fidelity α^* exists to ensure $b(\alpha) > 1$. It also indicates that the threshold is very close to zero when the diagonal elements of the time evolution vanish simultaneously in the ensemble, namely $\mathbb{E}\left(p(U,0^n)\right) \approx 0$. The threshold can

be suppressed by choosing a larger simulation time t since $\alpha^* \to 0$ as $t \to \infty$. Furthermore, when $\alpha^* \ll 1$, the conditional average bitstring probability is

$$\mathbb{E}_{\mathbb{Q}_\alpha}\left(\sum_{x\neq 0^n} p(U,x)\right) = \frac{1}{N}\left(1+\frac{2(\alpha-\alpha^*)}{1+\alpha}\right)+\mathcal{O}\left(\frac{1}{N^2}\right).$$

Note that at $t^{\rm opt}$, the threshold $\alpha^*|_{t^{\rm opt}} \approx \mathcal{H}_1/2$. Eq. (7.55) implies that $\mathcal{H}_1 \geq -\frac{2}{N-1}$. Therefore, when the number of qubits n is not sufficiently large, $\alpha^*|_{t^{\rm opt}}$ can possibly be negative. However, as n increases, $\alpha^*|_{t^{\rm opt}}$ converges to 0 exponentially fast because the lower bound $-\frac{2}{N-1} \to 0$ in the large n-limit. This agrees with the numerical behavior of the threshold α^* shown in Fig. 7.6.

7.16 Analytic estimation of $t^{\rm opt}$ for large n

According to the result in Fig. 7.6, at $t = t^{\rm opt}$, we have

$$\mathbb{E}\left(p_t(U,0^n)\right) = \mathbb{E}|\left\langle 0^n|e^{-{\rm i}\mathfrak{H}t}|0^n\right\rangle|^2 \approx 0.$$

Jensen's inequality gives

$$\left|\mathbb{E}\left\langle 0^n|e^{-{\rm i}\mathfrak{H}t}|0^n\right\rangle\right|^2 \leq \mathbb{E}|\left\langle 0^n|e^{-{\rm i}\mathfrak{H}t}|0^n\right\rangle|^2 \approx 0.$$

If $\mathfrak{H}$ is a H-RACBEM and the corresponding U is drawn from the Haar measure, then

$$\mathbb{E}\left\langle 0^n|e^{-{\rm i}\mathfrak{H}t}|0^n\right\rangle = \int_0^1 e^{-{\rm i}\lambda t}\mathbb{P}^{(1)}_{\rm eig}(\lambda)\,{\rm d}\lambda.$$

Here $\mathbb{P}^{(1)}_{\rm eig}(\lambda)$ is defined in defined in Eq. (7.47) with $\ell = 1$. It is also the 1-marginal (a.k.a. the level density) of the joint probability distribution of all eigenvalues in Eq. (7.44), which is called a β-Jacobi ensemble with $\beta = 2$ [56]. With the block encoding of one ancilla qubit (i.e. $M = 2$), the level density follows the Beta$(0.5, 0.5)$ distribution in the large n-limit [92], i.e. in the sense of weak convergence, we have

$$\lim_{n\to\infty}\mathbb{P}^{(1)}_{\rm eig}(\lambda) = \frac{1}{\pi}\lambda^{-\frac{1}{2}}(1-\lambda)^{-\frac{1}{2}}.$$

Therefore for any t,

$$\begin{aligned}\lim_{n\to\infty}\int_0^1 e^{-\mathrm{i}\lambda t}\mathbb{P}^{(1)}_{\mathrm{eig}}(\lambda)\,\mathrm{d}\lambda &= \int_0^1 e^{-\mathrm{i}\lambda t}\frac{1}{\pi}\lambda^{-\frac{1}{2}}(1-\lambda)^{-\frac{1}{2}}\,\mathrm{d}\lambda \\ &\overset{\lambda=\sin^2\left(\frac{\theta}{2}\right)}{=\!=\!=} \frac{1}{\pi}\int_0^\pi \exp\left(-\mathrm{i}t\sin^2\frac{\theta}{2}\right)\mathrm{d}\theta \\ &= e^{-\frac{\mathrm{i}t}{2}}\frac{1}{2\pi}\int_{-\pi}^{\pi} e^{\mathrm{i}\frac{t}{2}\cos\theta}\,\mathrm{d}\theta \\ &= e^{-\frac{\mathrm{i}t}{2}}J_0(t/2).\end{aligned} \tag{7.40}$$

Here we have used the integral representation of the Bessel function of the first kind

$$J_0(t/2) = \frac{1}{2\pi}\int_{-\pi}^{\pi} e^{\mathrm{i}\frac{t}{2}\sin\theta}\,\mathrm{d}\theta = \frac{1}{2\pi}\int_{-\pi}^{\pi} e^{\mathrm{i}\frac{t}{2}\cos\theta}\,\mathrm{d}\theta.$$

Therefore in the large n limit, $\mathbb{E}\,\langle 0^n|e^{-\mathrm{i}\mathfrak{H}t}|0^n\rangle$ approximately vanishes at the first node of $J_0(t/2)$, which gives

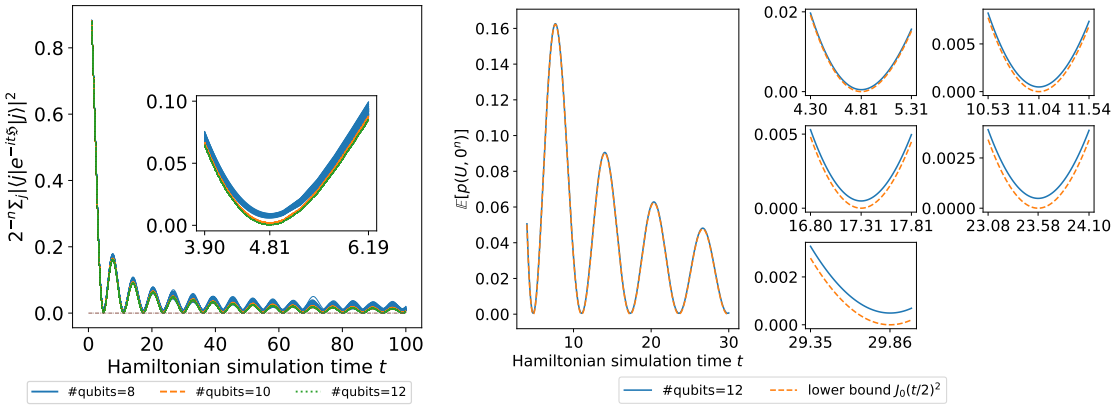
$$t^{\mathrm{opt}} \approx 4.81.$$

This agrees very well with the numerical results in Figs. 7.6 and 7.10.

7.17 Statistical property of the random-matrix ensemble inherited from Haar measure

The solution of the system heavy output generation problem and analytic evaluation of the system linear cross-entropy score require the use of statistical properties of the ensemble of random matrices obtained from the Haar measure. In this section, we derive the statistical properties of the ensemble. We consider a generic block encoding with m extra ancilla qubits, namely, an n-qubit matrix A is a submatrix of an $(n+m)$-qubit unitary U. We use $M = 2^m$ to represent the dimension of the Hilbert space generated by m ancilla qubits. We assume that U is drawn from an $(n+m)$-qubit Haar measure.

Given the identification $\mathbb{C}^{N\times N} \simeq \mathbb{C}^{N^2}$, the uniform measure on the space of complex matrices is identified as the pushforward of the Lebesgue measure on $\mathbb{C}^{N^2}$, for example, by taking the coordinate system as matrix elements. We denote this uniform measure as $\mathrm{d}A$. Assuming that A is an n-qubit submatrix of a Haar-distributed $(n+m)$-qubit unitary U, the first theorem gives a characterization of the induced probability distribution of A.



(a) (b)

Figure 7.10: Numerical justification of t^{opt}. (a) The trajectory of the average diagonal probability $2^{-n}\sum_j \left|\langle j|e^{-it\mathfrak{H}}|j\rangle\right|^2$ as a function of Hamiltonian simulation time t. The broadening is due to plotting ~ 100 instances individually. The subfigure in the box shows the behavior near $t^{\text{opt}} \approx 4.81$. (b) The average probability by measuring all qubits with 0 and the analytical lower bound on it. Zooming into the first five zeros of the Bessel function, the minima of the average probability well agree these zeros.

Theorem 7.17.1 ([45, Theorem 1.3.1])**.** *Let $A \in \mathbb{C}^{N\times N}$ be a submatrix block encoded in a Haar unitary. Then the probability density is*

$$\mathbb{P}(A) = \mathcal{Z}^{-1} \det\left(I - A^\dagger A\right)^{N(M-2)} \mathbb{1}_{\|A\|_2\leq 1}, \tag{7.41}$$

where $\mathcal{Z} := \int_{\|A\|_2\leq 1} \det\left(I - A^\dagger A\right)^{N(M-2)} \mathrm{d}A$ is a normalization constant. Here, $\mathbb{1}$ is an indicator function. It gives 1 when the condition in the subscript is satisfied, and gives 0 otherwise. Generically, let A be an n_1-by-n_2 submatrix of an n-by-n Haar-distributed unitary U, and $n \geq n_1 + n_2$. Then, the probability density is

$$\mathbb{P}(A) \propto \det\left(I - A^\dagger A\right)^{n-n_1-n_2} \mathbb{1}_{\|A\|_2\leq 1}. \tag{7.42}$$

In particular, for 1-block-encoded matrix with $m = 1$, the exponent of the determinant is 0, and A is uniformly distributed in the unit ball $\{A \in \mathbb{C}^{N\times N} : \|A\|_2 \leq 1\}$. Let us consider $A = W\Sigma V^\dagger$ where $W \in \mathrm{U}(N)/\mathrm{U}(1)^N, V \in \mathrm{U}(N)$ and $\operatorname{diag}\Sigma = (\sigma_1, \cdots, \sigma_N)$. The Jacobian of this decomposition is

$$\mathrm{d}A \propto \mathrm{d}V\,\mathrm{d}W \left(\Delta(\sigma_1^2, \cdots, \sigma_N^2)^2 \prod_{j=1}^N \sigma_j\,\mathrm{d}\sigma_j\right)$$

where $\mathrm{d}W$, $\mathrm{d}V$ are the Haar measure on their compact manifolds respectively and $\Delta(x_1, \cdots, x_n) := \prod_{i<j}(x_i - x_j)$ is the Vandermonde determinant. Then, the distribution of V, W, Σ follow immediately the theorem.

Corollary 7.17.2. *Let W, V be Haar-distributed. The joint distribution of all singular values has the density*

$$\mathbb{P}(\sigma_1, \cdots, \sigma_N) \propto \Delta(\sigma_1^2, \cdots, \sigma_N^2)^2 \prod_{i=1}^{N} \sigma_i \left(1 - \sigma_i^2\right)^{N(M-2)} \mathbb{1}_{\sigma_i \in [0,1]}. \tag{7.43}$$

Since $\mathfrak{H} = A^\dagger A$, the eigenvalue λ_j of the random Hermitian matrix $\mathfrak{H}$ and the singular value σ_j of the complex matrix A is related by $\lambda_j = \sigma_j^2$. By a direct change-of-variable, the joint distribution of all eigenvalues has the density

$$\mathbb{P}(\lambda_1, \cdots, \lambda_N) = \mathcal{Z}_{\mathrm{eig}}^{-1} \Delta(\lambda_1, \cdots, \lambda_N)^2 \prod_{i=1}^{N} (1 - \lambda_i)^{N(M-2)} \mathbb{1}_{\lambda_i \in [0,1]}. \tag{7.44}$$

The normalization constant is precisely given by the Selberg's integral [134]

$$\mathcal{Z}_{\mathrm{eig}} = \prod_{j=0}^{N-1} \frac{\Gamma(j+1)\Gamma(j+2)\Gamma(j+N(M-2)+1)}{\Gamma(j+N(M-1)+1)}. \tag{7.45}$$

The distribution is invariant under the relabeling of eigenvalues $(\lambda_1, \cdots, \lambda_N) \mapsto (\lambda_{\pi(1)}, \cdots, \lambda_{\pi(N)})$ for any permutation π. This feature is inherited from the bi-invariance of the Haar measure on the compact Lie group.

In practice, only the marginal distribution involving a few eigenvalues will be used. However, the Vandermonde determinant in the joint distribution couples all eigenvalues together, which makes it hard to compute the marginal distribution analytically. Nonetheless, a semi-analytical representation by orthogonal polynomial expansion can be derived as follows.

Let $w(x) := (1-x)^{N(M-2)}$ be the weight function, $(f, g)_w := \int_0^1 f(x)g(x)w(x)\,\mathrm{d}x$ be the weighted inner product on $[0, 1]$, and $\|f\|_w := \sqrt{(f, f)_w}$ be the weighted norm.

Theorem 7.17.3 ([110, Theorem 5.7.1])**.** *Let $\{C_i(x) : \deg C_i = i,\ i = 0, \cdots, N-1\}$ be a set of linearly independent monic polynomials such that they are orthogonal with respect to $(\cdot, \cdot)_w$. Let $c_i := \|C_i\|_w^2$. Define a bivariate function*

$$K(x, y) := w(x) \sum_{i=0}^{N-1} \frac{1}{c_i} C_i(x) C_i(y). \tag{7.46}$$

Then, the joint distribution for ℓ eigenvalues follows a determinantal process

$$\mathbb{P}_{eig}^{(\ell)}(\lambda_1, \cdots, \lambda_\ell) = \frac{(N-\ell)!}{N!} \det\left[K(\lambda_{j_1}, \lambda_{j_2})\right]_{j_1, j_2 = 1, \cdots, \ell}. \tag{7.47}$$

The orthogonal polynomial can be generated by 3-point recursion formula. Specifically, for 1-block-encoding (i.e. $m = 1$ and $M = 2$), the weight function is $w \equiv 1$, and the orthogonal polynomial is the shifted Legendre polynomial

$$C_i(x) \propto P_i(2x - 1). \tag{7.48}$$

Corollary 7.17.4. *For 1-block-encoding, the joint eigenvalue distribution can be expressed in terms of the bivariate function*

$$K(x, y) = \sum_{i=0}^{N-1} (2i + 1) P_i(2x - 1) P_i(2y - 1), \tag{7.49}$$

where P_i is the i-th Legendre polynomial.

With Corollary 7.17.4, the averages of bitstring probability can be evaluated semi-analytically. For a generic complex even polynomial, we define

$$\mathcal{R}^{q_1, \cdots, q_{\ell_2}}_{k_1, \cdots, k_{\ell_1} | r_1, \cdots, r_{\ell_3}} := \mathbb{E}\left(\prod_{j=1}^{\ell_1} g^{k_j}(\sigma_j) \prod_{j=1}^{\ell_2} \overline{g^{q_j}(\sigma_{\ell_1 + j})} \prod_{j=1}^{\ell_3} |g^{r_j}(\sigma_{\ell_1 + \ell_2 + j})| \right). \tag{7.50}$$

A complex polynomial $f \in \mathbb{C}[x]$ can be determined by setting $f(x^2) = g(x)$. Note that $g(\sigma_j) = f(\lambda_j)$ relates the singular value transformation and the eigenvalue transformation. The expectation can be expresses exactly by the integration with joint distribution,

$$\mathcal{R}^{q_1, \cdots, q_{\ell_2}}_{k_1, \cdots, k_{\ell_1} | r_1, \cdots, r_{\ell_3}} = \int_{[0,1]^{\ell_1 + \ell_2 + \ell_3}} \mathbb{P}_{\text{eig}}^{(\ell_1 + \ell_2 + \ell_3)} \prod_{j=1}^{\ell_1} f^{k_j}(\lambda_j) \prod_{j=1}^{\ell_2} \overline{f^{q_j}(\lambda_{\ell_1 + j})} \prod_{j=1}^{\ell_3} |f^{r_j}(\lambda_{\ell_1 + \ell_2 + j})| \, \mathrm{d}\lambda_1 \cdots \mathrm{d}\lambda_{\ell_1 + \ell_2 + \ell_3}. \tag{7.51}$$

For Hamiltonian simulation, $e^{-\mathrm{i}tx^2}$ has unit absolute value for all $x \in \mathbb{R}$. For simplicity we assume the approximation error is sufficiently small, and $g(x) = s_t(x) = e^{-\mathrm{i}tx^2}$, or equivalently $f(x) = e^{-\mathrm{i}tx}$. This allows us to omit the terms due to $|f^{r_j}(\lambda_{\ell_1 + \ell_2 + j})|$. Furthermore, when the upper and lower indices of $\mathcal{R}$ in Eq. (7.51) are the same

$k_j = q_j = 1$, the relabeling invariance of eigenvalues implies that the defined quantity is reduced to

$$\begin{aligned}\mathcal{H}_\ell(t) &= \mathbb{E}\left(\prod_{j=1}^{\ell} g(\sigma_j)\overline{g(\sigma_{\ell+j})}\right) \\ &= \int_{[0,1]^{2\ell}} \mathbb{P}_{\text{eig}}^{(2\ell)}(\lambda_1, \cdots, \lambda_{2\ell}) \cos\left(t\sum_{j=1}^{\ell}\lambda_j - \lambda_{\ell+j}\right) \mathrm{d}\lambda_1 \cdots \mathrm{d}\lambda_{2\ell},\end{aligned} \tag{7.52}$$

which is directly related to the Hamiltonian simulation. When the time dependence is irrelevant to the analysis, we drop the t dependence in $\mathcal{H}_\ell(t)$ by writing it as $\mathcal{H}_\ell$ for simplicity. We can compute $\mathcal{H}_\ell$ as follows.

Theorem 7.17.5. *Let the degree-d complex polynomial $f(x) = \sum_{q=0}^{d} c_q P_q(2x-1)$ be decomposed in terms of Legendre polynomials. Then,*

$$\mathcal{H}_\ell = \frac{(N-2\ell)!}{N!}\sum_{k_1=0}^{N-1}\cdots\sum_{k_{2\ell}=0}^{N-1}\sum_{\varsigma\in\mathsf{S}_{2\ell}}\mathrm{sgn}(\varsigma)\prod_{j=1}^{2\ell}\left((2k_j+1)\sum_{q=0}^{d} C_q^{(j)} F_{q,k_j,k_{\varsigma^{-1}(j)}}\right). \tag{7.53}$$

Here, $\mathsf{S}_{2\ell}$ is the symmetric group,

$$C_q^{(j)} = \begin{cases} c_q & , \text{ if } j \le \ell, \\ \overline{c_q} & , \text{ otherwise,}\end{cases}$$

and

$$\begin{aligned}F_{i,j,k} &= \frac{1}{2}\int_{-1}^{1} P_i(x)P_j(x)P_k(x)\,\mathrm{d}x \\ &= \begin{cases} \frac{(2s-2i)!(2s-2j)!(2s-2k)!}{(2s+1)!}\left(\frac{s!}{(s-i)!(s-j)!(s-k)!}\right)^2, \\ \quad \text{if } 2s = i+j+k \text{ is even and } |i-j| \le k \le i+j, \\ 0, \text{ otherwise.}\end{cases}\end{aligned} \tag{7.54}$$

Proof. Let $f^{(j)}(x) = \sum_{q=0}^{d} C_q^{(j)} P_q(2x-1)$ so that $f^{(j)}(x) = f(x)$ when $j \le \ell$ and $f^{(j)}(x) = \overline{f(x)}$ when $j > \ell$. Then, directly applying Theorem 7.17.3 and Corollary 7.17.4, the quantity can be evaluated immediately,

$$\begin{aligned}\frac{N!}{(N-2\ell)!}\mathcal{H}_\ell &= \sum_{\varsigma\in\mathsf{S}_{2\ell}}\mathrm{sgn}(\varsigma)\int_{[0,1]^{2\ell}}\prod_{j=1}^{2\ell} f^{(j)}(x_j)K(x_j, x_{\varsigma(j)})\,\mathrm{d}x_j \\ &= \sum_{\varsigma\in\mathsf{S}_{2\ell}}\mathrm{sgn}(\varsigma)\sum_{k_1=0}^{N-1}\cdots\sum_{k_{2\ell}=0}^{N-1}\prod_{j=1}^{2\ell}(2k_j+1)\int_0^1 f^{(j)}(x)P_{k_j}(2x-1)P_{k_{\varsigma^{-1}(j)}}(2x-1)\,\mathrm{d}x.\end{aligned}$$

The conclusion follows. □

The constraint in Eq. (7.54) will be referred to as the triangle rule. Due to the triangle rule, $F_{i,j,k}$ is a sparse tensor. Many terms in the (2ℓ)-fold summation vanishes, which can be used to accelerate the evaluation. Note that Legendre polynomials are bounded by 1 on $[-1,1]$, which implies that $|F_{i,j,k}| \leq 1$. To circumvent the numerical instability arising from factorials, $F_{i,j,k}$ can be evaluated recursively,

$$F_{0,0,0} = 1,\ F_{i,j+1,k+1} = \frac{2s+1-2i}{s+1-i}\frac{s+1}{2s+3}F_{i,j,k},$$
$$F_{i,j,k+2} = \frac{2s+1-2i}{s+1-i}\frac{2s+1-2j}{s+1-j}\frac{s-k}{2s-2k-1}\frac{s+1}{2s+3}F_{i,j,k},$$

where $2s = i+j+k$. Using Algorithm 7.17.1, $F_{i,j,k}$ can be evaluated stably with s recursions.

Algorithm 7.17.1 A stable recursive algorithm for computing $F_{i,j,k}$

Input: A triplet (i,j,k) satisfying the triangle rule.

1: Order and relabel the triplet so that $i \leq j \leq k$.
2: Set $2s = i+j+k$.
3: **if** $k \geq j-i+2$ **then**
4: Recursively call the algorithm to compute $F_{i,j,k-2}$. Note $(i,j,k-2)$ preserves the triangle rule.
5: **Return:** $F_{i,j,k} = \frac{2s-1-2i}{s-i}\frac{2s-1-2j}{s-j}\frac{s-k+1}{2s-2k+1}\frac{s}{2s+1}F_{i,j,k-2}$.
6: **else if** $k < j-i+2$ and $i < j$ **then**
7: Recursively call the algorithm to compute $F_{i,j-1,k-1}$. Note $(i,j-1,k-1)$ preserves the triangle rule.
8: **Return:** $F_{i,j,k} = \frac{2s-1-2i}{s-i}\frac{s}{2s+1}F_{i,j-1,k-1}$.
9: **else** $i = j = k = 0$
10: **Return:** $F_{0,0,0} = 1$.
11: **end if**

The measurement on all qubits gives an $(n+1)$-bit string. We are interested in the bitstring $0x$ which means the outcome of the ancilla qubit is 0 and that of n system qubits is $x \in \{0,1\}^n$. The probability of measuring the bitstring $0x$ is

$$p(U,x) = |\langle 0x|\mathcal{U}_{f,U}|00^n\rangle|^2 = |\langle x|g_{\mathrm{SV}}(A)|0^n\rangle|^2 = \sum_{j,k=0}^{N-1} g(\sigma_j)\overline{g(\sigma_k)}V_{j0}\overline{V_{jx}}\,\overline{V_{k0}}V_{kx}.$$

When $x = 0^n \equiv 0$, $p(U,x) = \sum_{j,k} g(\sigma_j)\overline{g(\sigma_k)}\,|V_{j0}|^2\,|V_{k0}|^2$ involves only one column of a Haar unitary V. Yet when $x \neq 0^n$, the probability involves two columns. For

$x \neq 0^n$ or 1^n, let us consider another unitary $\widetilde{V}$ by permuting the column 1 and column x of V. By the bi-invariance of Haar measure on unitary group, $\widetilde{V}$ and V are identically distributed. Therefore, we conclude the following lemma.

Lemma 7.17.6. *For any nonzero bitstring $0^n \neq x \in \{0,1\}^n$, $p(U,x)$ is identically distributed.*

According to Corollary 7.17.2, the distributions of Σ and V are decoupled. The average over the singular values can be evaluated semi-analytically, and the average over the Haar unitary can be analytically computed by using representation theory. We conclude the relevant average values as follows.

Theorem 7.17.7. *For Hamiltonian simulation benchmark, the average of bitstring probability is*

$$\mathbb{E}\left(p(U,0^n)\right) = \frac{N-1}{N+1}\mathcal{H}_1 + \frac{2}{N+1}, \quad \mathbb{E}\left(\sum_{x\neq 0^n} p(U,x)\right) = \frac{N-1}{N+1}\left(1-\mathcal{H}_1\right). \quad (7.55)$$

Furthermore, the second-order average quantities are

$$\begin{aligned}
&\mathbb{E}\left(p(U,0^n)^2\right) = \frac{12}{(N+2)(N+3)} + \frac{12N(N-1)\mathcal{H}_1}{(N+1)(N+2)(N+3)} \\
&+ \frac{(N-1)(N-2)(N-3)}{(N+1)(N+2)(N+3)}\mathcal{H}_2, \\
&\textit{and} \\
&\mathbb{E}\left(\sum_{x\neq 0^n} p(U,x)^2\right) = \frac{2(N-1)(N^2+3N+6)}{N(N+1)(N+2)(N+3)} - \frac{4(N-1)(N^2-N+6)\mathcal{H}_1}{N(N+1)(N+2)(N+3)} \\
&+ \frac{2(N-1)(N-2)(N-3)}{N(N+1)(N+2)(N+3)}\mathcal{H}_2.
\end{aligned} \quad (7.56)$$

Proof. We first evaluate the first order moments in Eq. (7.55). Let $p_j =: |V_{j0}|^2$. Directly applying Theorem 7.17.1 to the 0-th column of V, the joint probability density of k distinct success probabilities is

$$\mathbb{P}(p_1,\cdots,p_k) = \prod_{j=0}^{k-1}(N-k+j)\left(1-\sum_{j=1}^{k} p_j\right)^{N-k-1} \mathbb{1}_{\sum_{j=1}^{k} p_j<1}.$$

Given an index set $\boldsymbol{\alpha} := (\alpha_1, \cdots, \alpha_k)$ and $|\boldsymbol{\alpha}| := \sum_{j=1}^k \alpha_j$, the average $\mathcal{I}_{\boldsymbol{\alpha}} := \mathbb{E}\left(\prod_{j=1}^k p_j^{\alpha_j}\right) = \left(\prod_{j=1}^k \alpha_j!\right) \prod_{j=0}^{|\boldsymbol{\alpha}|-1} \frac{1}{N+j}$ follows direct computation,

$$\begin{aligned}
&\left(\prod_{j=0}^{k-1} \frac{1}{N-k+j}\right) \mathcal{I}_{\boldsymbol{\alpha}} \\
&= \int_{\sum_{j=1}^{k-1} p_j < 1} \prod_{j=1}^{k-1} p_j^{\alpha_j} \, \mathrm{d}p_j \int_0^{1-\sum_{j=1}^{k-1} p_j} p_k^{\alpha_k} \left(1 - \sum_{j=1}^{k-1} p_j - p_k\right)^{N-k-1} \mathrm{d}p_k \\
&= \alpha_k! \prod_{j=0}^{\alpha_k} \frac{1}{N-k+j} \int_{\sum_{j=1}^{k-1} p_j < 1} \left(1 - \sum_{j=1}^{k-1} p_j\right)^{N-k+\alpha_k} \prod_{j=1}^{k-1} p_j^{\alpha_j} \, \mathrm{d}p_j \\
&= \cdots = \left(\prod_{j=1}^{k} \alpha_j!\right) \prod_{j=0}^{|\boldsymbol{\alpha}|+k-1} \frac{1}{N-k+j}.
\end{aligned}$$

For the second equal sign, we use the identity

$$\int_0^y x^\alpha (y-x)^\beta \, \mathrm{d}x = y^{\alpha+\beta+1} \mathrm{B}(\alpha+1, \beta+1) = y^{\alpha+\beta+1} \frac{\alpha!\beta!}{(\alpha+\beta+1)!}, \tag{7.57}$$

where the Beta function is

$$\mathrm{B}(x,y) := \int_0^1 t^{x-1}(1-t)^{y-1} \, \mathrm{d}t = \frac{\Gamma(x)\Gamma(y)}{\Gamma(x+y)}.$$

By definition, $p(U, 0^n) = \sum_{j,k} g(\sigma_j)\overline{g(\sigma_k)} p_j p_k$. Applying these results for average and using the fact that singular values and singular vectors are independent, the average bitstring probability is

$$\begin{aligned}
\mathbb{E}\left(p(U,0^n)\right) &= N\mathbb{E}\left(|g(\sigma_1)|^2\right) \mathcal{I}_{(2)} + N(N-1)\mathbb{E}\left(\frac{1}{2}\left(g(\sigma_1)\overline{g(\sigma_2)} + \overline{g(\sigma_1)} g(\sigma_2)\right)\right) \mathcal{I}_{(1,1)} \\
&= \frac{N-1}{N+1} \mathcal{H}_1 + \frac{2}{N+1}.
\end{aligned}$$

Then,

$$\mathbb{E}\left(\sum_{x \neq 0^n} p(U,x)\right) = 1 - \mathbb{E}\left(p(U,0^n)\right) = \frac{N-1}{N+1}\left(1 - \mathcal{H}_1\right).$$

Now we evaluate the second order moments in Eq. (7.56). Let $\pi \in \mathsf{S}_4$ be any permutation, and R be any set of constraints on four n-bit binary strings i, j, k, l.

We denote the action of the symmetric group as $\pi \cdot \mathsf{R} := \{(\pi(i), \pi(j), \pi(k), \pi(l)) : (i,j,k,l) \in \mathsf{R}\}$. Due to the relabeling invariance of the joint distribution of singular values, we have

$$\begin{aligned}\sum_{(i,j,k,l)\in\pi\cdot\mathsf{R}} \mathbb{E}\left(g(\sigma_i)\overline{g(\sigma_j)}g(\sigma_k)\overline{g(\sigma_l)}\right) &= \sum_{(i,j,k,l)\in\mathsf{R}} g(\sigma_{\pi(i)})\overline{g(\sigma_{\pi(j)})}g(\sigma_{\pi(k)})\overline{g(\sigma_{\pi(l)})} \\ &= \sum_{(i,j,k,l)\in\mathsf{R}} g(\sigma_i)\overline{g(\sigma_j)}g(\sigma_k)\overline{g(\sigma_l)}.\end{aligned} \tag{7.58}$$

For example, let $\mathsf{R}_1 := \{(i=j)\neq(k=l)\}, \mathsf{R}_2 := \{(i=k)\neq(j=l)\}, \mathsf{R}_3 := \{(i=l)\neq(j=k)\}$, and $\pi_1 := \begin{pmatrix} i & j & k & l \\ i & k & j & l \end{pmatrix}, \pi_2 := \begin{pmatrix} i & j & k & l \\ i & l & j & k \end{pmatrix}$. Then, $\pi_1 \cdot \mathsf{R}_1 = \mathsf{R}_2$ and $\pi_2 \cdot \mathsf{R}_1 = \mathsf{R}_3$ holds. Therefore, for any $\ell = 1, 2, 3$, it holds that

$$\sum_{(i,j,k,l)\in\mathsf{R}_\ell} \mathbb{E}\left(g(\sigma_i)\overline{g(\sigma_j)}g(\sigma_k)\overline{g(\sigma_l)}\right) = N(N-1)\mathbb{E}\left(|g(\sigma_1)|^2 |g(\sigma_2)|^2\right) = N(N-1).$$

For bitstring $x = 0^n$,

$$\mathbb{E}\left(p(U,0^n)^2\right) = \mathbb{E}\left(\sum_{i,j,k,l} g(\sigma_i)\overline{g(\sigma_j)}g(\sigma_k)\overline{g(\sigma_l)}p_i p_j p_k p_l\right)$$

Using Eq. (7.58), the four-fold summation can be categorized into five equivalent classes. We define the partition $\boldsymbol{\alpha}$ be the array with at most 4 entries whose sum is exactly 4.

1. All indices are distinct with partition $\boldsymbol{\alpha} = (1,1,1,1)$: $i \neq j \neq k \neq l$. The contribution is $N(N-1)(N-2)(N-3)\mathcal{H}_2\mathcal{I}_{(1,1,1,1)}$ where $\mathcal{I}_{(1,1,1,1)} = \frac{1}{N(N+1)(N+2)(N+3)}$.

2. Only three indices are distinct with partition $\boldsymbol{\alpha} = (2,1,1)$: $i = j \neq k \neq l$ and its other five permutations. The contribution is $6N(N-1)(N-2)\mathcal{H}_1\mathcal{I}_{(2,1,1)}$ where $\mathcal{I}_{(2,1,1)} = \frac{2}{N(N+1)(N+2)(N+3)}$.

3. Only two indices are distinct with partition $\boldsymbol{\alpha} = (3,1)$: $i = j = k \neq l$ and its other three permutations. The contribution is $4N(N-1)\mathcal{H}_1\mathcal{I}_{(3,1)}$ where $\mathcal{I}_{(3,1)} = \frac{6}{N(N+1)(N+2)(N+3)}$.

4. Only two indices are distinct with partition $\boldsymbol{\alpha} = (2,2)$: $(i = j) \neq (k = l)$ and its other two permutations. The contribution is $3N(N-1)\mathcal{I}_{(2,2)}$ where $\mathcal{I}_{(2,2)} = \frac{4}{N(N+1)(N+2)(N+3)}$.

5. All indices are the same with partition $\boldsymbol{\alpha} = (4)$: $i = j = k = l$. The contribution is $N\mathcal{I}_{(4)}$ where $\mathcal{I}_{(4)} = \frac{24}{N(N+1)(N+2)(N+3)}$.

Collecting all the terms together, we have

$$\begin{aligned}\mathbb{E}\left(p(U,0^n)^2\right) =& \frac{12}{(N+2)(N+3)} + \frac{12N(N-1)\mathcal{H}_1}{(N+1)(N+2)(N+3)} \\ &+ \frac{(N-1)(N-2)(N-3)}{(N+1)(N+2)(N+3)}\mathcal{H}_2.\end{aligned}$$

The evaluation of the second moment of the probability with nonzero bitstrings x follows a similar procedure but is more involved. Using Lemma 7.17.6, it holds that $\mathbb{E}\left(\sum_{x\neq 0^n} p(U,x)^2\right) = (N-1)\mathbb{E}\left(p(U,1^n)^2\right)$. Let

$$H(i,j,k,l) := \mathbb{E}\left(\overline{V_{1^n,i}}V_{0^n,i}V_{1^n,j}\overline{V_{0^n,j}}\,\overline{V_{1^n,k}}V_{0^n,k}V_{1^n,l}\overline{V_{0^n,l}}\right).$$

Then,

$$\mathbb{E}\left(p(U,1^n)^2\right) = \sum_{ijkl}\mathbb{E}\left(g(\sigma_i)\overline{g(\sigma_j)}g(\sigma_k)\overline{g(\sigma_l)}\right)H(i,j,k,l).$$

Let us consider a linear map $\phi : \left(\mathbb{C}^N\right)^{\otimes 4} \to \left(\mathbb{C}^N\right)^{\otimes 4}$,

$$\phi_{ijkl} := \mathbb{E}\left(\bigotimes_{q\in\{i,j,k,l\}} V\left|q\right\rangle\left\langle q\right|V^\dagger\right),$$

Then

$$H(i,j,k,l) = \left\langle 0^n,1^n,0^n,1^n\right|\phi_{ijkl}\left|1^n,0^n,1^n,0^n\right\rangle.$$

Note that the linear map ϕ_{ijkl} commutes with the action of the symmetric group S_4 and that of the unitary group $\mathsf{U}(N)$ on the tensor product space $\left(\mathbb{C}^N\right)^{\otimes 4}$. Then, by Schur's lemma over $\mathbb{C}$, ϕ_{ijkl} is a scalar on each subspace decomposed with respect to Schur-Weyl duality. By taking trace on each subspace, the linear map is determined as the linear combination of projectors. The generic formula was derived in [46] by which $H(i,j,k,l)$ is evaluated. Let the four-fold sum in terms of $ijkl$ be broken into five equivalent classes as follows.

1. All indices are distinct with partition $(1,1,1,1)$: $i \neq j \neq k \neq l$. The contribution is $N(N-1)(N-2)(N-3)\mathcal{H}_2\frac{2}{N^2\,(N-1)(N+1)(N+2)(N+3)}$.

2. Only three indices are distinct with partition $(2,1,1)$: $i = j \neq k \neq l$ and its other five permutations. The contribution is

$$6N(N-1)(N-2)\mathcal{H}_1\left(-\frac{2\ (N-3)}{3\,N^2\,(N-1)(N+1)(N+2)(N+3)}\right).$$

3. Only two indices are distinct with partition $(3,1)$: $i = j = k \neq l$ and its other three permutations. The contribution is $4N(N-1)\mathcal{H}_1\left(-\frac{4}{N(N-1)(N+1)(N+2)(N+3)}\right)$.

4. Only two indices are distinct with partition $(2,2)$: $(i = j) \neq (k = l)$ and its other two permutations. The contribution is $3N(N-1)\frac{2\left(N^2+N+6\right)}{3\,N^2\,(N-1)(N+1)(N+2)(N+3)}$.

5. All indices are the same with partition (4): $i = j = k = l$. The contribution is $N\frac{4}{N\,(N+1)\,(N+2)\,(N+3)}$.

To conclude, the second moment of the probability with nonzero bitstrings is

$$\mathbb{E}\left(\sum_{x\neq 0^n} p(U,x)^2\right) = \frac{2(N-1)(N^2+3N+6)}{N(N+1)(N+2)(N+3)} - \frac{4(N-1)(N^2-N+6)\mathcal{H}_1}{N(N+1)(N+2)(N+3)} + \frac{2(N-1)(N-2)(N-3)}{N(N+1)(N+2)(N+3)}\mathcal{H}_2.$$

□

The asymptotic behavior of the ensemble is discussed in Section 7.19.

7.18 Estimating the number of measurements

In this subsection, we estimate the number of measurement shots needed to achieve given accuracy. In the experimental implementation, the measurement probability is computed by counting the frequency of a bit or bitstring $\mathfrak{b}$. For the i-th measurement, the outcome $\mathfrak{b}_i$ is associated with an indicator $\mathcal{I}_i(\mathfrak{b}) := \delta_{\mathfrak{b}_i,\mathfrak{b}}$ which is 1 if the outcome is b and is 0 otherwise. If the probability of measuring $\mathfrak{b}$ in the experiment is $p_{\exp}(\mathfrak{b})$, then the indicator $\mathcal{I}_i(\mathfrak{b})$'s are i.i.d. Bernoulli distributed, namely

$$\mathcal{I}_i(\mathfrak{b}) \text{ i.i.d. } \sim \text{Bernoulli}\left(p_{\exp}(\mathfrak{b})\right) = \begin{cases} 1 & \text{, with probability } p_{\exp}(\mathfrak{b}), \\ 0 & \text{, with probability } 1 - p_{\exp}(\mathfrak{b}). \end{cases}$$

Then, the frequency becomes an unbiased estimator of the probability $p_{\text{exp}}(\mathfrak{b})$. If M_{meas} measurement shots are used, the bitstring frequency is defined as

$$\hat{p}_{\text{exp}}(\mathfrak{b}) := \frac{1}{M_{\text{meas}}} \sum_{i=1}^{M_{\text{meas}}} \mathcal{I}_i(\mathfrak{b}).$$

Furthermore, using the Bernoulli distribution, the mean and variance of the estimator can be explicitly computed

$$\mathbb{E}\left(\hat{p}_{\text{exp}}(\mathfrak{b})\right) = p_{\text{exp}}(\mathfrak{b}), \quad \text{Var}\left(\hat{p}_{\text{exp}}(\mathfrak{b})\right) = \frac{p_{\text{exp}}(\mathfrak{b})\left(1 - p_{\text{exp}}(\mathfrak{b})\right)}{M_{\text{meas}}}.$$

The central limit theorem suggests that when M_{meas} is sufficiently large,

$$\mathbb{P}\left(|\hat{p}_{\text{exp}}(\mathfrak{b}) - p_{\text{exp}}(\mathfrak{b})| \leq 2\sqrt{2}\sqrt{\text{Var}\left(\hat{p}_{\text{exp}}(\mathfrak{b})\right)}\right) > 0.99.$$

In other words, with confidence level higher than 99%, it suffices to bound the deviation as

$$|\hat{p}_{\text{exp}}(\mathfrak{b}) - p_{\text{exp}}(\mathfrak{b})| \leq 2\sqrt{2}\sqrt{\text{Var}\left(\hat{p}_{\text{exp}}(\mathfrak{b})\right)} \leq \delta$$

where δ is some error control parameter, which gives

$$M_{\text{meas}} \geq \frac{8p_{\text{exp}}(\mathfrak{b})\left(1 - p_{\text{exp}}(\mathfrak{b})\right)}{\delta^2}.$$

Note that any probability is bounded $0 \leq p_{\text{exp}}(\mathfrak{b}) \leq 1$ which further gives

$$p_{\text{exp}}(\mathfrak{b})\left(1 - p_{\text{exp}}(\mathfrak{b})\right) \leq \frac{1}{4}.$$

Hence it suffices to choose

$$M_{\text{meas}} \geq \left\lceil \frac{2}{\delta^2} \right\rceil$$

so that the deviation in the probability is bounded $|\hat{p}_{\text{exp}}(\mathfrak{b}) - p_{\text{exp}}(\mathfrak{b})| \leq \delta$ with high probability. Then, when computing the QUES by measuring each circuit with $M_{\text{meas}} \geq \left\lceil \frac{2}{\delta^2} \right\rceil$ shots, which is referred to as $\widehat{\text{QUES}} := \mathbb{E}\left(\hat{P}_{\text{exp}}(U)\right)$, the statistical error is bounded as

$$\left|\widehat{\text{QUES}} - \text{QUES}\right| \leq \mathbb{E}\left(\left|\hat{P}_{\text{exp}}(U_i) - P_{\text{exp}}(U_i)\right|\right) \leq \delta.$$

The circuit fidelity is estimated by $\hat{\alpha}_{\mathrm{QUES}} := 2 \times \widehat{\mathrm{QUES}} - 1$. Furthermore, the error is bounded as

$$\begin{aligned} |\hat{\alpha}_{\mathrm{QUES}} - \alpha| &\leq |\hat{\alpha}_{\mathrm{QUES}} - \alpha_{\mathrm{QUES}}| + |\alpha_{\mathrm{QUES}} - \alpha| = 2\left|\widehat{\mathrm{QUES}} - \mathrm{QUES}\right| + |\alpha_{\mathrm{QUES}} - \alpha| \\ &\leq 2\delta + 16\epsilon + \mathcal{O}(\epsilon^2) \end{aligned} \tag{7.59}$$

where the last inequality uses Eq. (7.9). The derived error bound is a generalization of Eq. (7.9) by including the Monte Carlo measurement error due to the finite number of measurement shots.

7.19 Asymptotic behavior of the long time Hamiltonian simulation benchmark

The first and second moments of the bitstring probability are directly relevant to the construction of the system linear cross-entropy benchmarking based on Hamiltonian simulation. For simplicity, we define

$$K_1(x,t) := \mathbb{E}\left(p_t(U,x)\right), \text{ and } K_2(x,t) := \mathbb{E}\left(p_t(U,x)^2\right), \tag{7.60}$$

where the dependence on t is encoded in the implementation of the quantum circuit. In this section, we will investigate the behavior of these moments in different regimes in terms of the Hamiltonian simulation time t when N is sufficiently large.

According to Lemma 7.17.6, $K_1(x,t)$ and $K_2(x,t)$ are constant for any nonzero bitstring $x \neq 0^n$. Therefore, by using Theorem 7.17.7, we have

$$K_1(0^n,t) = \mathcal{H}_1(t) + \mathcal{O}\left(\frac{1}{N}\right), \; K_2(0^n,t) = \mathcal{H}_2(t) + \frac{12}{N}\left(\mathcal{H}_1(t) - \mathcal{H}_2(t)\right) + \mathcal{O}\left(\frac{1}{N^2}\right),$$

and for any $x \neq 0^n$,

$$\begin{aligned} K_1(x,t) &= \frac{1}{N}\left(1 - \mathcal{H}_1(t)\right) + \mathcal{O}\left(\frac{1}{N^2}\right), \\ K_2(x,t) &= \frac{1}{N^2}\left(2 - 4\mathcal{H}_1(t) + 2\mathcal{H}_2(t)\right) + \mathcal{O}\left(\frac{1}{N^3}\right). \end{aligned}$$

Note that by definition, $\mathcal{H}_1(t)$ and $\mathcal{H}_2(t)$ are the cosine transformation of the joint eigenvalue distribution $\mathbb{P}^{(2)}_{\mathrm{eig}}$ and $\mathbb{P}^{(4)}_{\mathrm{eig}}$ respectively. According to Theorem 7.17.3, these joint distributions are polynomials. Then both $\mathcal{H}_1(t)$ and $\mathcal{H}_2(t)$ converge to zero as

$t \to \infty$. In particular, there exists t^* such that $\mathcal{H}_1(t), \mathcal{H}_2(t) = \mathcal{O}\left(\frac{1}{N}\right)$ for any $t > t^*$. In this regime, for any nonzero bitstring $x \neq 0^n$,

$$K_1(x,t) = \frac{1}{N} + \mathcal{O}\left(\frac{1}{N^2}\right), \text{ and } K_2(x,t) = \frac{2}{N^2} + \mathcal{O}\left(\frac{1}{N^3}\right).$$

Note that the bitstring probability of a Haar-distributed unitary U, $p_{ij} := |U_{ij}|^2$, has the first and second moment $\mathbb{E}(p_{ij}) = \frac{1}{N}$ and $\mathbb{E}\left(p_{ij}^2\right) = \frac{2}{N^2} + \mathcal{O}\left(\frac{1}{N^3}\right)$. Furthermore, by using Lemma 7.17.6 and Theorem 7.17.7, the cross moment of two different nonzero bitstrings $x \neq y$ is

$$\mathbb{E}(p_t(U,x)p_t(U,y)) = \frac{1 - 2\mathcal{H}_1(t) + \mathcal{H}_2(t)}{N^2} + \mathcal{O}\left(\frac{1}{N^3}\right).$$

Thus, in the regime where $\mathcal{H}_1(t), \mathcal{H}_2(t) = \mathcal{O}\left(\frac{1}{N}\right)$, the cross moment is $\frac{1}{N^2} + \mathcal{O}\left(\frac{1}{N^3}\right)$. Following Theorem 7.12.1, the cross moment of success probabilities of a Haar-distributed unitary is exactly the same up to higher order $\mathbb{E}(p_{ij}p_{kj}) = \frac{1}{N(N+1)} = \frac{1}{N^2} + \mathcal{O}\left(\frac{1}{N^3}\right)$. Remarkably, the correlation between different nonzero bitstrings is small, which is quantified by the covariance $\mathrm{Cov}(p(U,x)p(U,y)) = \mathbb{E}(p(U,x)p(U,y)) - \mathbb{E}(p(U,x))\,\mathbb{E}(p(U,y)) = \mathcal{O}\left(\frac{1}{N^3}\right)$. We conclude that in the defined regime, the ensemble of the time evolution matrix induced by our construction has approximately the same statistics as that of Haar-distributed unitaries up to at least the second moment.

Note that the threshold $\alpha_t^* := \frac{\mathcal{H}_1(t)}{2 - 5\mathcal{H}_1(t) + 4\mathcal{H}_2(t)}$ is directly related to $\mathcal{H}_1(t)$ and $\mathcal{H}_2(t)$. When $t = t^{\mathrm{opt}}$ for small time or $t > t^*$ in the long time regime, we have $\mathcal{H}_1(t), \mathcal{H}_2(t) = \mathcal{O}\left(\frac{1}{N}\right)$ which leads to an exponentially small threshold $\alpha_t^* = \mathcal{O}\left(\frac{1}{N}\right)$. It allows the quantum supremacy to be achieved for a small circuit fidelity $\alpha > \alpha_t^*$. Note t^* can be impractically large for near-term applications. Hence it is crucial that the simulation at $t = t^{\mathrm{opt}} \approx 4.81$ is equally effective and much more tractable.

Chapter 8

Conclusion

This dissertation presents a series of our research efforts undertaken in recent years to expand both the theory and practical applications of Quantum Signal Processing (QSP). The work is divided into two main parts: (1) theories of the QSP algorithm, and (2) the implementation of QSP related tasks on near-term and early fault-tolerant quantum devices.

From a computational perspective, despite the theoretical foundations of QSP being laid out in [101, 64], devising a numerically stable algorithm for computing phase factors was a significant challenge. We demonstrate that an optimization-based approach enables efficient and accurate evaluation of the phase factors necessary for constructing QSP circuits for generating unitary representations of non-unitary operations. This method effectively removes a critical bottleneck in applying QSP to quantum algorithms. We delve into the optimization-based algorithm, analyze its landscape, and underscore the importance of symmetry constraints in phase factors. By establishing local strong convexity, we show the algorithm's convergence within an L^∞-ball with a radius of $\mathcal{O}(1/d)$, where d is the polynomial's degree. To address the limitations encountered with high-degree polynomials, we refine the approach by solving a system of nonlinear equations, leading to a simple fixed-point iteration algorithm with d-independent convergence radius in ℓ^1 space. The formalism also extends to Newton's method, pushing practical applications into the near fully coherent regime.

Theoretically, we explore the infinite quantum processing problem: whether an infinite-length set of phase factors in ℓ^1 can represent target polynomials expressed as infinite series. Theorem 5.3.3 provides a partial affirmative answer, subject to the constraint that the 1-norm of the Chebyshev coefficients c of the target function is bounded by a constant r_c. This constraint, while surmountable by rescaling the target function, is not always naturally met. For instance, in Hamiltonian simulation,

$\|c\|_1 = \mathcal{O}(\tau)$, where τ is the simulation time. Our numerical results suggest that the fixed-point algorithm and decay properties hold even for large $\|c\|_1$, indicating potential for relaxing the $\|c\|_1 \leq r_c$ condition.

On the other hand, the impressive performance of Newton's method, especially in the fully-coherent regime, remains theoretically unexplained. Extensive numerical experiments consistently converge to maximal solutions, aligning with the symmetric phase-factor solutions proposed in Ref. [153]. Further investigation is needed to ascertain whether the mapping F maintains a unique landscape within its injective neighborhood near $\mathbf{0}$.

A practical challenge in implementing QSP-based algorithms lies in the complexity of implementing block encodings, particularly due to the extensive use of multi-qubit Toffoli gates. This complexity poses hurdles for near-term applications and remains challenging for achieving desired accuracies on fault-tolerant devices. To address this, we explore two potential solutions: transitioning to a Hamiltonian evolution input model, exemplified by the QETU algorithm, and adopting a random matrix input model to reduce structural dependence. The QETU algorithm, efficient for ground-state preparation and energy estimation, shows promise for early fault-tolerant devices. The random matrix input model, analyzed for its statistical properties, leads to a quantum Hamiltonian simulation benchmark that provides a straightforward metric to assess quantum computers' performance in scientific computing.

Despite these advancements, the cost of implementing these algorithms remains substantial. We introduce a control-free implementation of QETU for a class of quantum spin Hamiltonians, significantly simplifying circuit complexity. Nonetheless, the sensitivity of QETU results to quantum noise and the benchmark's reliance on globally depolarizing noise assumptions pose challenges. These circuits, while potentially suitable for NISQ devices, may still suffer from substantial errors. Combining these methods with randomized compilation techniques, as suggested in Ref.[148], and/or error mitigation strategies, as proposed in Ref.[29], could significantly reduce noise impact and enable meaningful results on near-term devices.

Bibliography

[1] URL: https://quantum-computing.ibm.com.

[2] Scott Aaronson and Alex Arkhipov. "The computational complexity of linear optics". In: *Proceedings of the forty-third annual ACM symposium on Theory of computing*. 2011, pp. 333–342.

[3] Scott Aaronson and Lijie Chen. "Complexity theoretic foundations of quantum supremacy experiments". In: *arXiv:1612.05903* (2016).

[4] Scott Aaronson and Sam Gunn. "On the classical hardness of spoofing linear cross-entropy benchmarking". In: *arXiv preprint arXiv:1910.12085* (2019).

[5] Scott Aaronson and Patrick Rall. "Quantum approximate counting, simplified". In: *Symposium on Simplicity in Algorithms*. SIAM. 2020, pp. 24–32.

[6] Dorit Aharonov et al. "The power of quantum systems on a line". In: *Comm. Math. Phys.* 287.1 (2009), pp. 41–65. DOI: 10.1007/s00220-008-0710-3.

[7] Dong An, Di Fang, and Lin Lin. "Time-dependent unbounded Hamiltonian simulation with vector norm scaling". In: *Quantum* 5 (2021), p. 459.

[8] Frank Arute et al. "Quantum supremacy using a programmable superconducting processor". In: *Nature* 574.7779 (2019), pp. 505–510.

[9] Yosi Atia and Dorit Aharonov. "Fast-forwarding of Hamiltonians and exponentially precise measurements". In: *Nature Comm.* 8.1 (2017). DOI: 10.1038/s41467-017-01637-7.

[10] Ryan Babbush et al. "Focus beyond quadratic speedups for error-corrected quantum advantage". In: *PRX Quantum* 2.1 (2021), p. 010103.

[11] Adriano Barenco et al. "Elementary gates for quantum computation". In: *Physical review A* 52.5 (1995), p. 3457.

[12] Paul Benioff. "The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines". In: *Journal of statistical physics* 22.5 (1980), pp. 563–591.

[13] Juan Bermejo-Vega et al. "Architectures for quantum simulation showing a quantum speedup". In: *Phys. Rev. X* 8.2 (2018), p. 021010.

[14] D. W. Berry, A. M. Childs, and R. Kothari. "Hamiltonian simulation with nearly optimal dependence on all parameters". In: *Proceedings of the 56th IEEE Symposium on Foundations of Computer Science* (2015), pp. 792–809.

[15] D. W. Berry et al. "How to perform the most accurate possible phase measurements". In: *Phys. Rev. A* 80.5 (2009). DOI: 10.1103/physreva.80.052114.

[16] D. W. Berry et al. "Simulating Hamiltonian dynamics with a truncated Taylor series". In: *Phys. Rev. Lett.* 114 (2015), p. 090502.

[17] Dominic W Berry and Andrew M Childs. "Black-box Hamiltonian simulation and unitary implementation". In: *Quantum Information & Computation* 12.1-2 (2012), pp. 29–62.

[18] Dominic W Berry, Richard Cleve, and Sevag Gharibian. "Gate-efficient discrete simulations of continuous-time quantum query algorithms". In: *Quantum Information and Computation* 14.1-2 (2014), pp. 1–30.

[19] Dominic W Berry et al. "Efficient quantum algorithms for simulating sparse Hamiltonians". In: *Commun. Math. Phys.* 270.2 (2007), pp. 359–371.

[20] Dominic W Berry et al. "Qubitization of arbitrary basis quantum chemistry leveraging sparsity and low rank factorization". In: *Quantum* 3 (2019), p. 208.

[21] Dimitri P Bertsekas. "On the Goldstein-Levitin-Polyak gradient projection method". In: *IEEE Transactions on automatic control* 21.2 (1976), pp. 174–184.

[22] Robin Blume-Kohout et al. "Demonstration of qubit operations below a rigorous fault tolerance threshold with gate set tomography". In: *Nat. Commun.* 8.1 (2017), pp. 1–13.

[23] Sergio Boixo et al. "Characterizing quantum supremacy in near-term devices". In: *Nat. Phys.* 14.6 (2018), pp. 595–600.

[24] Kyle EC Booth et al. "Quantum-accelerated constraint programming". In: *Quantum* 5 (2021), p. 550.

[25] Gilles Brassard et al. "Quantum amplitude amplification and estimation". In: *Contemp. Math.* 305 (2002), pp. 53–74.

[26] Michael J Bremner, Ashley Montanaro, and Dan J Shepherd. "Average-case complexity versus approximate simulation of commuting quantum computations". In: *Phys. Rev. Lett.* 117.8 (2016), p. 080501.

[27] Winton Brown and Omar Fawzi. "Scrambling speed of random quantum circuits". In: *arXiv preprint arXiv:1210.6644* (2012).

[28] Vera von Burg et al. "Quantum computing enhanced computational catalysis". In: *Physical Review Research* 3.3 (2021), p. 033055.

[29] Zhenyu Cai, Xiaosi Xu, and Simon C Benjamin. "Mitigating coherent noise using Pauli conjugation". In: *npj Quantum Inf.* 6.1 (2020), pp. 1–9.

[30] Earl Campbell. "Random compiler for fast Hamiltonian simulation". In: *Phys. Rev. Lett.* 123.7 (2019), p. 070503.

[31] Earl T Campbell. "Early fault-tolerant simulations of the Hubbard model". In: *Quantum Science and Technology* 7.1 (2021), p. 015007.

[32] Shantanav Chakraborty, András Gilyén, and Stacey Jeffery. "The power of block-encoded matrix powers: improved regression techniques via faster Hamiltonian simulation". In: *arXiv:1804.01973* (2018).

[33] Rui Chao et al. "Finding angles for quantum signal processing with machine precision". In: *arXiv preprint arXiv:2003.02831* (2020).

[34] Chi-Fang Chen et al. "Quantum simulation via randomized product formulas: Low gate complexity with accuracy guarantees". In: *arXiv* (2020).

[35] Zijun Chen et al. "Exponential suppression of bit or phase flip errors with repetitive error correction". In: *arXiv preprint arXiv:2102.06132* (2021).

[36] Elliott Ward Cheney. *Introduction to approximation theory*. McGraw-Hill, 1966.

[37] Andrew M Childs. "On the relationship between continuous-and discrete-time quantum walk". In: *Commun. Math. Phys.* 294.2 (2010), pp. 581–603.

[38] Andrew M Childs, Aaron Ostrander, and Yuan Su. "Faster quantum simulation by randomization". In: *Quantum* 3 (2019), p. 182.

[39] Andrew M Childs and Yuan Su. "Nearly optimal lattice simulation by product formulas". In: *Phys. Rev. Lett.* 123.5 (2019), p. 050503.

[40] Andrew M Childs et al. "Theory of trotter error with commutator scaling". In: *Phys. Rev. X* 11.1 (2021), p. 011020.

[41] Andrew M. Childs, Robin Kothari, and Rolando D. Somma. "Quantum Algorithm for Systems of Linear Equations with Exponentially Improved Dependence on Precision". In: *SIAM J. Comput.* 46 (2017), pp. 1920–1950.

[42] Andrew M. Childs et al. "Toward the first quantum simulation with quantum speedup". In: *Proc. Nat. Acad. Sci.* 115 (2018), pp. 9456–9461.

[43] Joonhee Choi et al. "Emergent Randomness and Benchmarking from Many-Body Quantum Chaos". In: *arXiv preprint arXiv:2103.03535* (2021).

[44] Kenneth Choi et al. "Rodeo algorithm for quantum computing". In: *Physical Review Letters* 127.4 (2021), p. 040505.

[45] Benoit Collins. "Intégrales matricielles et probabilités non-commutatives". PhD thesis. 2003.

[46] Benoıt Collins and Piotr Śniady. "Integration with respect to the Haar measure on unitary, orthogonal and symplectic group". In: *Communications in Mathematical Physics* 264.3 (2006), pp. 773–795.

[47] Arjan Cornelissen, Johannes Bausch, and András Gilyén. "Scalable Benchmarks for Gate-Based Quantum Computers". In: *arXiv preprint arXiv:2104.10698* (2021).

[48] Andrew W Cross et al. "Validating quantum computers using randomized model circuits". In: *Physical Review A* 100.3 (2019), p. 032328.

[49] Yulong Dong, Jonathan Gross, and Murphy Yuezhen Niu. "Beyond Heisenberg Limit Quantum Metrology through Quantum Signal Processing". In: *arXiv preprint arXiv:2209.11207* (2022).

[50] Yulong Dong and Lin Lin. "Random circuit block-encoded matrix and a proposal of quantum LINPACK benchmark". In: *Phys. Rev. A* 103.6 (2021), p. 062412.

[51] Yulong Dong, Lin Lin, and Yu Tong. "Ground-State Preparation and Energy Estimation on Early Fault-Tolerant Quantum Computers via Quantum Eigenvalue Transformation of Unitary Matrices". In: *PRX Quantum* 3 (2022), p. 040305.

[52] Yulong Dong, K Birgitta Whaley, and Lin Lin. "A quantum hamiltonian simulation benchmark". In: *npj Quantum Information* 8.1 (2022), p. 131.

[53] Yulong Dong et al. "Efficient phase factor evaluation in quantum signal processing". In: *Phys. Rev. A* 103 (2021), p. 042419.

[54] Yulong Dong et al. "Infinite quantum signal processing". In: *arXiv preprint arXiv:2209.10162* (2022).

[55] Yulong Dong et al. "Robust iterative method for symmetric quantum signal processing in all parameter regimes". In: *arXiv preprint arXiv:2307.12468* (2023).

[56] Ioana Dumitriu and Alan Edelman. "Matrix models for beta ensembles". In: *J. Math. Phys.* 43 (2002), pp. 5830–5847.

[57] Alexander Erhard et al. "Characterizing large-scale quantum computers via cycle benchmarking". In: *Nat. Commun.* 10.1 (2019), p. 5347.

[58] Di Fang, Lin Lin, and Yu Tong. "Time-marching based quantum solvers for time-dependent linear differential equations". In: *Quantum* 7 (2023), p. 955.

[59] Richard P Feynman. "Simulating physics with computers". In: *Int. J. Theor. Phys* 21.6/7 (1982).

[60] Xun Gao, Sheng-Tao Wang, and L-M Duan. "Quantum supremacy for simulating a translation-invariant Ising spin model". In: *Phys. Rev. Letters* 118.4 (2017), p. 040502.

[61] Xun Gao et al. "Limitations of linear cross-entropy as a measure for quantum advantage". In: *arXiv preprint arXiv:2112.01657* (2021).

[62] Yimin Ge, Jordi Tura, and J Ignacio Cirac. "Faster ground state preparation and high-precision ground energy estimation with fewer qubits". In: *J. Math. Phys.* 60.2 (2019), p. 022202. DOI: 10.1063/1.5027484.

[63] Michael R Geller and Zhongyuan Zhou. "Efficient error models for fault-tolerant architectures and the Pauli twirling approximation". In: *Phys. Rev. A* 88.1 (2013), p. 012314.

[64] András Gilyén, Srinivasan Arunachalam, and Nathan Wiebe. "Optimizing quantum optimization algorithms via faster quantum gradient computation". In: *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*. 2019, pp. 1425–1444.

[65] András Gilyén et al. "Quantum algorithm for Petz recovery channels and pretty good measurements". In: *Phys. Rev. Lett.* 128.22 (2022), p. 220502.

[66] András Gilyén et al. "Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics". In: *arXiv:1806.01838* (2018).

[67] 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*. 2019, pp. 193–204.

[68] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. "Advances in quantum metrology". In: *Nature Photon.* 5.4 (2011), p. 222. DOI: 10.1038/nphoton.2011.35.

[69] Vittorio Giovannetti, Seth Lloyd, and Lorenzo Maccone. "Quantum Metrology". In: *Phys. Rev. Lett.* 96.1 (2006). DOI: 10.1103/physrevlett.96.010401.

[70] Michael Grant and Stephen Boyd. *CVX: Matlab Software for Disciplined Convex Programming, version 2.1*. http://cvxr.com/cvx. Mar. 2014.

[71] Robert B Griffiths and Chi-Sheng Niu. "Semiclassical Fourier transform for quantum computation". In: *Phys. Rev. Lett.* 76.17 (1996), p. 3228. DOI: 10.1103/physrevlett.76.3228.

[72] Shouzhen Gu, Rolando D Somma, and Burak Şahinoğlu. "Fast-forwarding quantum evolution". In: *Quantum* 5 (2021), p. 577.

[73] J. Haah. "Product decomposition of periodic functions in quantum signal processing". In: *Quantum* 3 (2019), p. 190.

[74] Alfred Haar. "Die Minkowskische Geometrie und die Annäherung an stetige Funktionen". In: *Mathematische Annalen* 78.1 (1917), pp. 294–311.

[75] Jonas Haferkamp et al. "Closing gaps of a quantum advantage with short-time Hamiltonian dynamics". In: *Phys. Rev. Letters* 125.25 (2020), p. 250501.

[76] Aram W Harrow, Avinatan Hassidim, and Seth Lloyd. "Quantum algorithm for linear systems of equations". In: *Phys. Rev. Lett.* 103 (2009), p. 150502.

[77] Aram W Harrow and Richard A Low. "Random quantum circuits are approximate 2-designs". In: *Commun. Math. Phys.* 291.1 (2009), pp. 257–302.

[78] B. L. Higgins et al. "Entanglement-free Heisenberg-limited phase estimation". In: *Nature* 450.7168 (2007), pp. 393–396. DOI: 10.1038/nature06257.

[79] N. Higham. *Functions of matrices: theory and computation*. Vol. 104. SIAM, 2008.

[80] William J Huggins et al. "A non-orthogonal variational quantum eigensolver". In: *New J. of Phys.* 22.7 (2020), p. 073009. DOI: 10.1088/1367-2630/ab867b.

[81] Artur F Izmaylov et al. "Unitary partitioning approach to the measurement problem in the variational quantum eigensolver method". In: *Journal of chemical theory and computation* 16.1 (2019), pp. 190–195.

[82] Trevor Keen, Eugene Dumitrescu, and Yan Wang. "Quantum Algorithms for Ground-State Preparation and Green's Function Calculation". In: *arXiv preprint arXiv:2112.05731* (2021).

[83] C. T. Kelley. *Iterative methods for optimization*. Vol. 18. SIAM, 1999.

[84] Julia Kempe, Alexei Kitaev, and Oded Regev. "The complexity of the local Hamiltonian problem". In: *SIAM J. Comput.* 35.5 (2006), pp. 1070–1097. DOI: 10.1007/978-3-540-30538-5_31.

[85] A Yu Kitaev. "Quantum measurements and the Abelian stabilizer problem". In: *arXiv preprint quant-ph/9511026* (1995).

[86] Alexei Yu Kitaev, Alexander Shen, and Mikhail N Vyalyi. *Classical and quantum computation*. 47. American Mathematical Soc., 2002.

[87] I.D. Kivlichan et al. "Quantum Simulation of Electronic Structure with Linear Depth and Connectivity". In: *Phys. Rev. Lett.* 120.11 (2018), p. 110501.

[88] Emanuel Knill, Gerardo Ortiz, and Rolando D. Somma. "Optimal quantum measurements of expectation values of observables". In: *Phys. Rev. A* 75.1 (2007). DOI: 10.1103/PhysRevA.75.012328.

[89] David Layden. "First-order trotter error from a second-order perspective". In: *Physical Review Letters* 128.21 (2022), p. 210501.

[90] Michel Ledoux. *The concentration of measure phenomenon*. 89. American Mathematical Soc., 2001.

[91] Joonho Lee et al. "Even more efficient quantum computations of chemistry through tensor hypercontraction". In: *PRX Quantum* 2.3 (2021), p. 030305.

[92] Harvey S Leff. "Class of Ensembles in the Statistical Theory of Energy-Level Spectra". In: *Journal of Mathematical Physics* 5 (1964), pp. 763–768.

[93] Lin Lin and Yu Tong. "Heisenberg-limited ground state energy estimation for early fault-tolerant quantum computers". In: *PRX Quantum* 3 (2022), p. 010318.

[94] Lin Lin and Yu Tong. "Near-optimal ground state preparation". In: *Quantum* 4 (2020), p. 372.

[95] Lin Lin and Yu Tong. "Optimal quantum eigenstate filtering with application to solving quantum linear systems". In: *Quantum* 4 (2020), p. 361.

[96] Seth Lloyd. "Universal quantum simulators". In: *Science* (1996), pp. 1073–1078.

[97] G. H. Low and N. Wiebe. "Hamiltonian Simulation in the Interaction Picture". In: *arXiv:1805.00675* (2019).

[98] Guang Hao Low. "Hamiltonian simulation with nearly optimal dependence on spectral norm". In: *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*. 2019, pp. 491–502.

[99] Guang Hao Low and Isaac L Chuang. "Hamiltonian simulation by qubitization". In: *Quantum* 3 (2019), p. 163.

[100] Guang Hao Low and Isaac L Chuang. "Hamiltonian simulation by uniform spectral amplification". In: *arXiv:1707.05391* (2017).

[101] Guang Hao Low and Isaac L. Chuang. "Optimal Hamiltonian Simulation by Quantum Signal Processing". In: *Phys. Rev. Lett.* 118 (2017), p. 010501.

[102] Sirui Lu, Mari Carmen Banuls, and J Ignacio Cirac. "Algorithms for quantum simulation at finite energies". In: *PRX Quantum* 2.2 (2021), p. 020321.

[103] Easwar Magesan, Jay M Gambetta, and Joseph Emerson. "Scalable and robust randomized benchmarking of quantum processes". In: *Phys. Rev. Lett.* 106.18 (2011), p. 180504.

[104] John M Martyn et al. "Efficient fully-coherent quantum signal processing algorithms for real-time dynamics simulation". In: *J. Chem. Phys.* 158.2 (2023), p. 024106.

[105] John M Martyn et al. "Grand unification of quantum algorithms". In: *PRX Quantum* 2.4 (2021), p. 040203.

[106] Christian Mastrodonato and Roderich Tumulka. "Elementary proof for asymptotics of large Haar-distributed unitary matrices". In: *Letters in Mathematical Physics* 82.1 (2007), pp. 51–59.

[107] Sam McArdle, Earl Campbell, and Yuan Su. "Exploiting fermion number in factorized decompositions of the electronic structure Hamiltonian". In: *Physical Review A* 105.1 (2022), p. 012403.

[108] Sam McArdle, András Gilyén, and Mario Berta. "Quantum state preparation without coherent arithmetic". In: *arXiv preprint arXiv:2210.14892* (2022).

[109] Jarrod R McClean et al. "The theory of variational hybrid quantum-classical algorithms". In: *New J. Phys.* 18.2 (2016), p. 023023.

[110] Madan Lal Mehta. *Random matrices*. Elsevier, 2004.

[111] Xiao Mi et al. "Information scrambling in computationally complex quantum circuits". In: *arXiv:2101.08870* (2021).

[112] Mario Motta et al. "Determining eigenstates and thermal states on a quantum computer using quantum imaginary time evolution". In: *Nat. Phys.* 16.2 (2020), pp. 205–210.

[113] Daniel Nagaj, Pawel Wocjan, and Yong Zhang. "Fast amplification of QMA". In: *Quantum Inf. Comput.* 9.11 (2009), pp. 1053–1068.

[114] Michael A Nielsen and Isaac Chuang. *Quantum computation and quantum information*. 2000.

[115] J. Nocedal and S. J. Wright. *Numerical optimization*. Springer Verlag, 1999.

[116] Thomas E O'Brien, Brian Tarasinski, and Barbara M Terhal. "Quantum phase estimation of multiple eigenvalues for small-scale (noisy) experiments". In: *New J. Phys.* 21.2 (2019), p. 023022.

[117] Thomas E O'Brien et al. "Error mitigation via verified phase estimation". In: *PRX Quantum* 2.2 (2021), p. 020317.

[118] Peter JJ O'Malley et al. "Scalable quantum simulation of molecular energies". In: *Phys. Rev. X* 6.3 (2016), p. 031007. DOI: 10.1103/PhysRevX.6.031007.

[119] Roberto Oliveira and Barbara M Terhal. "The complexity of quantum spin systems on a two-dimensional square lattice". In: *arXiv preprint quant-ph/0504050* (2005).

[120] I. V. Oseledets. "Tensor-train decomposition". In: *SIAM J. Sci. Comput.* 33.5 (2011), pp. 2295–2317.

[121] Alberto Peruzzo et al. "A variational eigenvalue solver on a photonic quantum processor". In: *Nat. Commun.* 5 (2014), p. 4213.

[122] Dénes Petz and Júlia Réffy. "On asymptotics of large Haar distributed unitary matrices". In: *Periodica Mathematica Hungarica* 49.1 (2004), pp. 103–117.

[123] David Poulin and Pawel Wocjan. "Sampling from the Thermal Quantum Gibbs State and Evaluating Partition Functions with a Quantum Computer". In: *Phys. Rev. Lett.* 103.22 (2009). DOI: 10.1103/physrevlett.103.220502.

[124] John Preskill. "Quantum Computing in the NISQ era and beyond". In: *Quantum* 2 (2018), p. 79.

[125] Timothy Proctor et al. "Measuring the capabilities of quantum computers". In: *Nat. Phys.* 18.1 (2022), pp. 75–79.

[126] Timothy Proctor et al. "Scalable randomized benchmarking of quantum computers using mirror circuits". In: *Physical Review Letters* 129.15 (2022), p. 150502.

[127] Timothy J Proctor et al. "Direct randomized benchmarking for multiqubit devices". In: *Phys. Rev. Letters* 123.3 (2019), p. 030503.

[128] Patrick Rall. "Quantum algorithms for estimating physical quantities using block encodings". In: *Physical Review A* 102.2 (2020), p. 022408.

[129] Jan-Michael Reiner et al. "Emulating the one-dimensional Fermi-Hubbard model by a double chain of qubits". In: *Phys. Rev. A* 94 (3 Sept. 2016), p. 032338. DOI: 10.1103/PhysRevA.94.032338. URL: https://link.aps.org/doi/10.1103/PhysRevA.94.032338.

[130] Zane M Rossi and Isaac L Chuang. "Multivariable quantum signal processing (M-QSP): prophecies of the two-headed oracle". In: *Quantum* 6 (2022), p. 811.

[131] Zane M Rossi and Isaac L Chuang. "Semantic embedding for quantum algorithms". In: *arXiv preprint arXiv:2304.14392* (2023).

[132] Zane M Rossi et al. "Quantum signal processing with continuous variables". In: *arXiv preprint arXiv:2304.14383* (2023).

[133] Burak Şahinoğlu and Rolando D Somma. "Hamiltonian simulation in the low-energy subspace". In: *npj Quantum Information* 7.1 (2021), p. 119.

[134] Atlee Selberg. "Berkninger om et multilet integral". In: *Norsk, Mat. Tidsskr.* 26 (1944), pp. 71–78.

[135] Rolando D Somma. "Quantum eigenvalue estimation via time series analysis". In: *New J. Phys.* 21.12 (2019), p. 123025. DOI: 10.1088/1367-2630/ab5c60.

[136] Yuan Su. "Fast-Forwardable Quantum Evolution and Where to Find Them". In: *Quantum Views* 5 (2021), p. 62.

[137] Yuan Su, Hsin-Yuan Huang, and Earl T Campbell. "Nearly tight Trotterization of interacting electrons". In: *Quantum* 5 (2021), p. 495.

[138] Yuan Su et al. "Fault-tolerant quantum simulations of chemistry in first quantization". In: *PRX Quantum* 2.4 (2021), p. 040332.

[139] Wenyu Sun and Ya-Xiang Yuan. *Optimization theory and methods: nonlinear programming*. Vol. 1. Springer Science & Business Media, 2006.

[140] Mario Szegedy. "Quantum speed-up of Markov chain based algorithms". In: *45th Annual IEEE symposium on foundations of computer science*. 2004, pp. 32–41.

[141] Ewin Tang and Kevin Tian. "A CS guide to the quantum singular value transformation". In: *arXiv preprint arXiv:2302.14324* (2023).

[142] Ruslan N Tazhigulov et al. "Simulating challenging correlated molecules and materials on the Sycamore quantum processor". In: *arXiv preprint arXiv:2203.15291* (2022).

[143] Yu Tong et al. "Fast inversion, preconditioned quantum linear system solvers, fast Green's-function computation, and fast evaluation of matrix functions". In: *Physical Review A* 104.3 (2021), p. 032422.

[144] Lloyd N Trefethen. *Approximation theory and approximation practice*. Vol. 164. SIAM, 2019.

[145] Norm M Tubman et al. "Postponing the orthogonality catastrophe: efficient state preparation for electronic structure simulations on quantum devices". In: *arXiv preprint arXiv:1809.05523* (2018).

[146] Vladyslav Verteletskyi, Tzu-Ching Yen, and Artur F Izmaylov. "Measurement optimization in the variational quantum eigensolver using a minimum clique cover". In: *The Journal of chemical physics* 152.12 (2020), p. 124114.

[147] Benjamin Villalonga et al. "Establishing the quantum supremacy frontier with a 281 pflop/s simulation". In: *Quantum Sci. Tech.* 5 (2020), p. 034003.

[148] Joel J Wallman and Joseph Emerson. "Noise tailoring for scalable quantum computation via randomized compiling". In: *Phys. Rev. A* 94.5 (2016), p. 052325.

[149] Kianna Wan, Mario Berta, and Earl T Campbell. "Randomized Quantum Algorithm for Statistical Phase Estimation". In: *Phys. Rev. Lett.* 129.3 (2022), p. 030503.

[150] Daochen Wang, Oscar Higgott, and Stephen Brierley. "Accelerated variational quantum eigensolver". In: *Phys. Rev. Lett.* 122.14 (2019), p. 140504. DOI: 10.1103/physrevlett.122.140504.

[151] Guoming Wang, Sukin Sim, and Peter D Johnson. "State preparation boosters for early fault-tolerant quantum computation". In: *Quantum* 6 (2022), p. 829.

[152] Guoming Wang et al. "Minimizing estimation runtime on noisy quantum computers". In: *PRX Quantum* 2.1 (2021), p. 010346.

[153] Jiasu Wang, Yulong Dong, and Lin Lin. "On the energy landscape of symmetric quantum signal processing". In: *Quantum* 6 (2022), p. 850.

[154] Chu-Ryang Wie. "Simpler quantum counting". In: *arXiv preprint arXiv:1907.08119* (2019).

[155] Nathan Wiebe et al. "Bayesian inference via rejection filtering". In: *arXiv preprint arXiv:1511.06458* (2015).

[156] Lexing Ying. "Stable factorization for phase factors of quantum signal processing". In: *Quantum* 6 (2022), p. 842.

[157] Pei Zeng, Jinzhao Sun, and Xiao Yuan. "Universal quantum algorithmic cooling on a quantum computer". In: *arXiv preprint arXiv:2109.15304* (2021).

[158] Ruizhe Zhang, Guoming Wang, and Peter Johnson. "Computing ground state properties with early fault-tolerant quantum computers". In: *Quantum* 6 (2022), p. 761.

[159] Marcin Zwierz, Carlos A Pérez-Delgado, and Pieter Kok. "Ultimate limits to quantum metrology and the meaning of the Heisenberg limit". In: *Phys. Rev. A* 85.4 (2012), p. 042112. DOI: 10.1103/PhysRevA.85.042112.

[160] Marcin Zwierz, Carlos A. Pérez-Delgado, and Pieter Kok. "General Optimality of the Heisenberg Limit for Quantum Metrology". In: *Phys. Rev. Lett.* 105.18 (2010). DOI: 10.1103/physrevlett.105.180402.

[161] Karol Zyczkowski and Hans-Jürgen Sommers. "Truncations of random unitary matrices". In: *Journal of Physics A: Mathematical and General* 33.10 (2000), p. 2045.