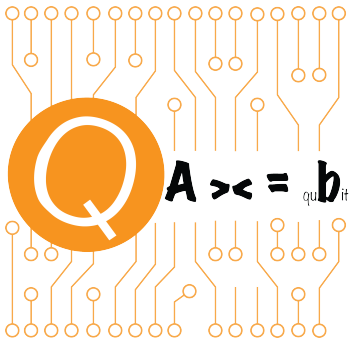
Lecture Notes on Quantum Algorithms for Scientific Computation

Lin Lin

Department of Mathematics, University of California, Berkeley
Challenge Institute of Quantum Computation, University of California, Berkeley
Computational Research Division, Lawrence Berkeley National Laboratory

January 21, 2022



Contents

Preface

With availability of near-term quantum devices and the breakthrough of quantum supremacy experiments, quantum computation has received an increasing amount of attention from a diverse range of scientific disciplines in the past few years. Despite the availability of excellent textbooks as well as lecture notes such as [NC00, KSV02, Nak08, RP11, Aar13, Pre99, DW19, Chi21], these materials often cover *all* aspects of quantum computation, including complexity theory, physical implementations of quantum devices, quantum information theory, quantum error correction, quantum algorithms etc. This leaves little room for introducing how a quantum computer is supposed to *be used* to solve challenging computational problems in scientific and engineering. For instance, after the initial reading of (admittedly, selected chapters of) the classic textbook by Nielsen and Chuang [NC00], I was both amazed by the potential power of a quantum computer, and baffled by its practical range of applicability: are we really trying to build a quantum computer, either to perform a quantum Fourier transform or to perform a quantum search? Is quantum phase estimation the only bridge connecting a quantum computer on one side, and virtually *all* scientific computing problems on the other, such as solving linear systems, eigenvalue problems, least squares problems, differential equations, numerical optimization etc.?

Thanks to the significant progresses in the development of quantum algorithms, it should be by now self-evident that the answer to both questions above is *no*. This is a fast evolving field, and many important progresses have only been developed in the past few years. However, many such developments are theoretically and technically involved, and can be difficult to penetrate for someone with only basic knowledge of quantum computing. I think it is worth delivering some of these exciting results, in a somewhat more accessible way, to a broader community interested in using future fault-tolerant quantum computers to solve scientific problems.

This is a set of lecture notes used in a graduate topic class in applied mathematics called "Quantum Algorithms for Scientific Computation" at the Department of Mathematics, UC Berkeley during the fall semester of 2021. These lecture notes focus only on quantum algorithms closely related to scientific computation, and in particular, matrix computation. In fact, this is only a small class of quantum algorithms viewed from the perspective of the "quantum algorithm zoo"[1]. This means that many important materials are consciously left out, such as quantum complexity theory, applications in number theory and cryptography (notably, Shor's algorithm), applications in algebraic problems (such as the hidden subgroup problems) etc. Readers interested in these topics can consult some of the excellent aforementioned textbooks. Since the materials were designed to fit into the curriculum of one semester, several other topics relevant to scientific computation are not included, notably adiabatic quantum computation (AQC), and variational quantum algorithms (VQA). These materials may be added in future editions of the lecture notes. To my knowledge,

[1] `https://quantumalgorithmzoo.org/`

some of the materials in these lecture notes may be new and have not been presented in the literature. The sections marked by * can be skipped upon first reading without much detriment.

I would like to thank Dong An, Yulong Dong, Di Fang, Fabian M. Faulstich, Cory Hargus, Zhen Huang, Subhayan Roy Moulik, Yu Tong, Jiasu Wang, Mathias Weiden, Jiahao Yao, Lexing Ying for useful discussions and for pointing out typos in the notes. I would like also like to thank Nilin Abrahamsen, Di Fang, Subhayan Roy Moulik, Yu Tong for contributing some of the exercises, and Jiahao Yao for providing the cover image of the notes. For errors / comments / suggestions / general thoughts on the lectures notes, please send me an email: `linlin@math.berkeley.edu`.

CHAPTER 1

Preliminaries of quantum computation

1.1. Postulates of quantum mechanics

We introduce the four main postulates of quantum mechanics related to this course. For more details, we refer readers to [NC00, Section 2.2]. All postulates concern closed quantum systems (i.e., systems isolated from environments) only.

1.1.1. State space postulate. The set of all quantum states of a quantum system forms a complex vector space with inner product structure (i.e., it is a Hilbert space, denoted by $\mathcal{H}$), called the state space. If the state space $\mathcal{H}$ is finite dimensional, it is isomorphic to some $\mathbb{C}^N$, written as $\mathcal{H} \cong \mathbb{C}^N$. Without loss of generality we may simply take $\mathcal{H} = \mathbb{C}^N$. We always assume $N = 2^n$ for some non-negative integer n, often called the number of quantum bits (or qubits). A quantum state $\psi \in \mathbb{C}^N$ can be expressed in terms of its components as

$$\psi = \begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{N-1} \end{pmatrix}. \tag{1.1}$$

Its Hermitian conjugate is

$$\psi^\dagger = \begin{pmatrix} \overline{\psi}_0 & \overline{\psi}_1 & \cdots & \overline{\psi}_{N-1} \end{pmatrix}, \tag{1.2}$$

where $\overline{c}$ is the complex conjugation of $c \in \mathbb{C}$. We also use the Dirac notation, which uses $|\psi\rangle$ to denote a quantum state, $\langle\psi|$ to denote its Hermitian conjugation $\psi^\dagger$, and the inner product

$$\langle\psi|\varphi\rangle := \langle\psi, \varphi\rangle = \sum_{i\in[N]} \overline{\psi}_i \varphi_i. \tag{1.3}$$

Here $[N] = \{\, 0, \ldots, N-1 \,\}$. Let $\{|i\rangle\}$ be the standard basis of $\mathbb{C}^N$. The i-th entry of ψ can be written as an inner product $\psi_i = \langle i|\psi\rangle$. Then $|\psi\rangle\langle\varphi|$ should be interpreted as an outer product, with (i,j)-th matrix element given by

$$\langle i|(|\psi\rangle\langle\varphi|)|j\rangle = \langle i|\psi\rangle\langle\varphi|j\rangle = \psi_i \overline{\varphi}_j. \tag{1.4}$$

Two state vectors $|\psi\rangle$ and $c\,|\psi\rangle$ for some $0 \neq c \in \mathbb{C}$ always refer to the same physical state, i.e., c has no observable effects. Hence without loss of generality we always assume $|\psi\rangle$ is normalized to be a unit vector, i.e., $\langle\psi|\psi\rangle = 1$. Sometimes it is more convenient to write down an unnormalized state, which will be denoted by ψ without the ket notation $|\cdot\rangle$. Restricting to normalized state vectors, the complex number $c = e^{\mathrm{i}\theta}$ for some $\theta \in [0, 2\pi)$, called the global phase factor.

Example 1.1 (Single qubit system)**.** A (single) qubit corresponds to a state space $\mathcal{H} \cong \mathbb{C}^2$. We also define

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}. \tag{1.5}$$

Since the state space of the spin-$\frac{1}{2}$ system is also isomorphic to $\mathbb{C}^2$, this is also called the single spin system, where $|0\rangle, |1\rangle$ are referred to as the spin-up and spin-down state, respectively. A general state vector in $\mathcal{H}$ takes the form

$$|\psi\rangle = a\,|0\rangle + b\,|1\rangle = \begin{pmatrix} a \\ b \end{pmatrix}, \quad a, b \in \mathbb{C}, \tag{1.6}$$

and the normalization condition implies $|a|^2 + |b|^2 = 1$. So we may rewrite $|\psi\rangle$ as

$$|\psi\rangle = e^{\mathrm{i}\gamma}\left(\cos\frac{\theta}{2}\,|0\rangle + e^{\mathrm{i}\varphi}\sin\frac{\theta}{2}\,|1\rangle\right), \quad \theta, \varphi, \gamma \in \mathbb{R}. \tag{1.7}$$

If we ignore the irrelevant global phase γ, the state is effectively

$$|\psi\rangle = \cos\frac{\theta}{2}\,|0\rangle + e^{\mathrm{i}\varphi}\sin\frac{\theta}{2}\,|1\rangle, \quad 0 \le \theta < \pi, 0 \le \varphi < 2\pi. \tag{1.8}$$

So we may identify each single qubit quantum state with a unique point on the unit three-dimensional sphere (called the Bloch sphere) as

$$\mathbf{a} = (\sin\theta\cos\varphi, \sin\theta\sin\varphi, \cos\theta)^\top. \tag{1.9}$$

◇

1.1.2. Quantum operator postulate. The evolution of a quantum state from $|\psi\rangle \to |\psi'\rangle \in \mathbb{C}^N$ is always achieved via a unitary operator $U \in \mathbb{C}^{N\times N}$, i.e.,

$$|\psi'\rangle = U\,|\psi\rangle, \quad U^\dagger U = I_N. \tag{1.10}$$

Here $U^\dagger$ is the Hermitian conjugate of a matrix U, and I_N is the N-dimensional identity matrix. When the dimension is apparent, we may also simply write $I \equiv I_N$. In quantum computation, a unitary matrix is often referred to as a gate.

Example 1.2. For a single qubit, the Pauli matrices are

$$\sigma_x = X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad \sigma_y = Y = \begin{pmatrix} 0 & -\mathrm{i} \\ \mathrm{i} & 0 \end{pmatrix}, \quad \sigma_z = Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{1.11}$$

Together with the two-dimensional identity matrix, they form a basis of all linear operators on $\mathbb{C}^2$. ◇

Some other commonly used single qubit operators include, to name a few:

- Hadamard gate

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{1.12}$$

- Phase gate

$$S = \begin{pmatrix} 1 & 0 \\ 0 & \mathrm{i} \end{pmatrix} \tag{1.13}$$

- T gate:

$$T = \begin{pmatrix} 1 & 0 \\ 0 & e^{\mathrm{i}\pi/4} \end{pmatrix} \tag{1.14}$$

When there are notational conflicts, we will use the roman font such as H, X for these single-qubit gates (one common scenario is to distinguish the Hadamard gate H from a Hamiltonian H). An operator acting on an n-qubit quantum state space is called an n-qubit operator.

Starting from an initial quantum state $|\psi(0)\rangle$, the quantum state can evolve in time, which gives a single parameter family of quantum states denoted by $\{|\psi(t)\rangle\}$. These quantum states are related to each other via a quantum evolution operator U:

$$\psi(t_2) = U(t_2, t_1)\psi(t_1), \tag{1.15}$$

where $U(t_2, t_1)$ is unitary for any given t_1, t_2. Here $t_2 > t_1$ refers to quantum evolution forward in time, $t_2 < t_1$ refers to quantum evolution backward in time, and $U(t_1, t_1) = I$ for any t_1.

The quantum evolution under a time-independent Hamiltonian H satisfies the time-independent Schrödinger equation

$$\mathrm{i}\partial_t |\psi(t)\rangle = H |\psi(t)\rangle . \tag{1.16}$$

Here $H = H^\dagger$ is a Hermitian matrix. The corresponding time evolution operator is

$$U(t_2, t_1) = e^{-\mathrm{i}H(t_2 - t_1)}, \quad \forall t_1, t_2. \tag{1.17}$$

In particular, $U(t_2, t_1) = U(t_2 - t_1, 0)$.

On the other hand, for any unitary matrix U, we can always find a Hermitian matrix H such that $U = e^{\mathrm{i}H}$ (Exercise 1.1).

Example 1.3. Let the Hamiltonian H be the Pauli-X gate. Then

$$U(t, 0) = e^{-\mathrm{i}Xt} = \begin{pmatrix} \cos t & -\mathrm{i}\sin t \\ -\mathrm{i}\sin t & \cos t \end{pmatrix} = (\cos t)I - \mathrm{i}X(\sin t). \tag{1.18}$$

Starting from an initial state $|\psi(0)\rangle = |0\rangle$, after time $t = \pi/2$, the state evolves into $|\psi(\pi/2)\rangle = -\mathrm{i}\,|1\rangle$, i.e., the $|1\rangle$ state (up to a global phase factor). ◇

1.1.3. Quantum measurement postulate. Without loss of generality, we only discuss a special type of quantum measurements called the projective measurement. For more general types of quantum measurements, see [NC00, Section 2.2.3]. All quantum measurements expressed as a positive operator-valued measure (POVM) can be expressed in terms of projective measurements in an enlarged Hilbert space via the Naimark dilation theorem.

In a finite dimensional setting, a quantum observable always corresponds to a Hermitian matrix M, which has the spectral decomposition

$$M = \sum_m \lambda_m P_m. \tag{1.19}$$

Here $\lambda_m \in \mathbb{R}$ are the eigenvalues of M, and P_m is the projection operator onto the eigenspace associated with λ_m, i.e., $P_m^2 = P_m$.

When a quantum state $|\psi\rangle$ is measured by a quantum observable M, the outcome of the measurement is always an eigenvalue λ_m, with probability

$$p_m = \langle\psi|P_m|\psi\rangle . \tag{1.20}$$

After the measurement, the quantum state becomes

$$|\psi\rangle \to \frac{P_m |\psi\rangle}{\sqrt{p_m}} \tag{1.21}$$

Note that this is not a unitary process!

In order to evaluate the expectation value of a quantum observable M, we first use the resolution of identity:

$$\sum_m P_m = I. \tag{1.22}$$

This implies the normalization condition,

$$\sum_m p_m = \sum_m \langle\psi|P_m|\psi\rangle = \langle\psi|\psi\rangle = 1. \tag{1.23}$$

Together with $p_m \geq 0$, we find that $\{p_m\}$ is indeed a probability distribution.

The expectation value of the measurement outcome is

$$\mathbb{E}_\psi(M) = \sum_m \lambda_m p_m = \sum_m \lambda_m \langle\psi|P_m|\psi\rangle = \left\langle \psi \middle| \left(\sum_m \lambda_m P_m\right) \middle| \psi \right\rangle = \langle\psi|M|\psi\rangle . \tag{1.24}$$

Example 1.4. Again let $M = X$. From the spectral decomposition of X:

$$X |\pm\rangle = \lambda_\pm |\pm\rangle , \tag{1.25}$$

where $|\pm\rangle := \frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle)$, $\quad \lambda_\pm = \pm 1$, we obtain the eigendecomposition

$$M = X = |+\rangle\langle+| - |-\rangle\langle-| . \tag{1.26}$$

Consider a quantum state $|\psi\rangle = |0\rangle = \frac{1}{\sqrt{2}}(|+\rangle + |-\rangle)$, then

$$\langle\psi|P_+|\psi\rangle = \langle\psi|P_-|\psi\rangle = \frac{1}{2}. \tag{1.27}$$

Therefore the expectation value of the measurement is $\langle\psi|M|\psi\rangle = 0$. ◇

1.1.4. Tensor product postulate. For a quantum state consists of m components with state spaces $\{\mathcal{H}_i\}_{i=0}^{m-1}$, the state space is their tensor products denoted by $\mathcal{H} = \otimes_{i=0}^{m-1}\mathcal{H}_i$. Let $|\psi_i\rangle$ be a state vector in $\mathcal{H}_i$, then

$$|\psi\rangle = |\psi_0\rangle \otimes \cdots \otimes |\psi_{m-1}\rangle \tag{1.28}$$

in $\mathcal{H}$. However, not all quantum states in $\mathcal{H}$ can be written in the tensor product form above. Let $\{|e_j^{(i)}\rangle\}_{j\in[N_i]}$ be the basis of $\mathcal{H}_i$, then a general state vector in $\mathcal{H}$ takes the form

$$|\psi\rangle = \sum_{j_0\in[N_0],\ldots,j_{m-1}\in[N_{m-1}]} \psi_{j_0\cdots j_{m-1}} |e_{j_0}^{(0)}\rangle \otimes \cdots \otimes |e_{j_{m-1}}^{(m-1)}\rangle . \tag{1.29}$$

Here $\psi_{j_0\cdots j_{m-1}} \in \mathbb{C}$ is an entry of a m-way tensor, and the dimension of $\mathcal{H}$ is therefore $\prod_{i\in[m]} N_i$.

The state space of n-qubits is $\mathcal{H} = (\mathbb{C}^2)^{\otimes n} \cong \mathbb{C}^{2^n}$, rather than $\mathbb{C}^{2n}$. We also use the notation

$$|01\rangle \equiv |0,1\rangle \equiv |0\rangle|1\rangle \equiv |0\rangle \otimes |1\rangle , \quad |0^{\otimes n}\rangle = |0\rangle^{\otimes n} . \tag{1.30}$$

Furthermore, $x \in \{0,1\}^n$ is called a classical bit-string, and $\{|x\rangle \,|\, x \in \{0,1\}^n\}$ is called the computational basis of $\mathbb{C}^{2^n}$.

Example 1.5 (Two qubit system)**.** The state space is $\mathcal{H} = (\mathbb{C}^2)^{\otimes 2} \cong \mathbb{C}^4$. The standard basis is (row-major order, i.e., last index is the fastest changing one)

$$|00\rangle = \begin{pmatrix}1\\0\\0\\0\end{pmatrix}, \quad |01\rangle = \begin{pmatrix}0\\1\\0\\0\end{pmatrix}, \quad |10\rangle = \begin{pmatrix}0\\0\\1\\0\end{pmatrix}, \quad |11\rangle = \begin{pmatrix}0\\0\\0\\1\end{pmatrix}. \tag{1.31}$$

The Bell state (also called the EPR pair) is defined to be

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle) = \frac{1}{\sqrt{2}}\begin{pmatrix}1\\0\\0\\1\end{pmatrix}, \tag{1.32}$$

which cannot be written as any product state $|a\rangle \otimes |b\rangle$ (Exercise 1.4).

There are many important quantum operators on the two-qubit quantum system. One of them is the CNOT gate, with matrix representation

$$\mathrm{CNOT} = \begin{pmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{pmatrix}. \tag{1.33}$$

In other words, when acting on the standard basis, we have

$$\mathrm{CNOT}\begin{cases}|00\rangle &= |00\rangle\\ |01\rangle &= |01\rangle\\ |10\rangle &= |11\rangle\\ |11\rangle &= |10\rangle\end{cases}. \tag{1.34}$$

This can be compactly written as

$$\mathrm{CNOT}\,|a\rangle\,|b\rangle = |a\rangle\,|a \oplus b\rangle\,. \tag{1.35}$$

Here $a \oplus b = (a + b) \mod 2$ is the "exclusive or" (XOR) operation. ◇

Example 1.6 (Multi-qubit Pauli operators)**.** For a n-qubit quantum system, the Pauli operator acting on the i-th qubit is denoted by P_i ($P = X, Y, Z$). For instance

$$X_i := I^{\otimes(i-1)} \otimes X \otimes I^{\otimes(n-i)}. \tag{1.36}$$

◇

1.2. Density operator

So far all quantum states encountered can be described by a single $|\psi\rangle \in \mathcal{H}$, called the pure state. More generally, if a quantum system is in one of a number of states $|\psi_i\rangle$ with respective probabilities p_i, then $\{p_i, |\psi_i\rangle\}$ is an ensemble of pure states. The density operator of the quantum system is

$$\rho := \sum_i p_i\,|\psi_i\rangle\,\langle\psi_i|\,. \tag{1.37}$$

For a pure state $|\psi\rangle$, we have

$$\rho = |\psi\rangle\,\langle\psi| \tag{1.38}$$

is a rank-1 matrix.

Consider a quantum observable in Eq. (1.19) associated with the projectors $\{P_m\}$. For a pure state, it can be verified that the probability result of returning λ_m, and the expectation value of the measurement are respectively,

$$p(m) = \mathrm{Tr}[P_m\rho], \quad \mathbb{E}_\rho[M] = \mathrm{Tr}[M\rho] \tag{1.39}$$

The expression (1.39) also holds for general density operators ρ.

An operator ρ is the density operator associated to some ensemble $\{p_i, |\psi_i\rangle\}$ if and only if (1) $\mathrm{Tr}\,\rho = 1$ (2) $\rho \succeq 0$, i.e., ρ is a positive semidefinite matrix (also called a positive operator). All postulates in Section 1.1 can be stated in terms of density operators (see [NC00, Section 2.4.2]). Note that a pure state satisfies $\rho^2 = \rho$. In general we have $\rho^2 \preceq \rho$. If $\rho^2 \prec \rho$, then ρ is called (the density operator of) a mixed state. Furthermore, an ensemble of admissible density operators is also a density operator.

A quantum operator U that transforms $|\psi\rangle$ to $U|\psi\rangle$ also transforms the density operator according to

$$\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i| \xrightarrow{U} \sum_i p_i U|\psi_i\rangle\langle\psi_i|U^\dagger = U\rho U^\dagger := U[\rho]. \tag{1.40}$$

However, not all quantum operations on density operators need to be unitary! See [NC00, Section 8.2] for more general discussions on quantum operations.

Most of the discussions in this course will be restricted to pure states, and unitary quantum operations. Even in this restricted setting, the density operator formalism can still be convenient, particularly for describing a subsystem of a composite quantum system. Consider a quantum system of $(n+m)$-qubits, partitioned into a subsystem A with n qubits (the state space is $\mathcal{H}_A = \mathbb{C}^{2^n}$) and a subsystem B with m qubits (the state space is $\mathcal{H}_B = \mathbb{C}^{2^m}$) respectively. The quantum state is a pure state $|\psi\rangle \in \mathbb{C}^{2^{n+m}}$ with density operator ρ_{AB}. Let $|a_1\rangle, |a_2\rangle$ be two state vectors in $\mathcal{H}_A$, and $|b_1\rangle, |b_2\rangle$ be two state vectors in $\mathcal{H}_B$. Then the partial trace over system B is defined as

$$\mathrm{Tr}_B[|a_1\rangle\langle a_2| \otimes |b_1\rangle\langle b_2|] = |a_1\rangle\langle a_2|\,\mathrm{Tr}[|b_1\rangle\langle b_2|] = |a_1\rangle\langle a_2|\langle b_2|b_1\rangle. \tag{1.41}$$

Since we can expand the density operator ρ_{AB} in terms of the basis of $\mathcal{H}_A, \mathcal{H}_B$, the definition of (1.41) can be extended to define the reduced density operator for the subsystem A

$$\rho_A = \mathrm{Tr}_B[\rho_{AB}]. \tag{1.42}$$

The reduced density operator for the subsystem B can be similarly defined. The reduced density operators ρ_A, ρ_B are generally mixed states.

Example 1.7 (Reduced density operator of tensor product states)**.** If $\rho_{AB} = \rho_1 \otimes \rho_2$, then

$$\mathrm{Tr}_B[\rho_{AB}] = \rho_1, \quad \mathrm{Tr}_A[\rho_{AB}] = \rho_2. \tag{1.43}$$

◇

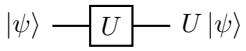
If a quantum observable is defined only on the subsystem A, i.e., $M = M_A \otimes I$ where M_A has the decomposition (1.19), then the success probability of returning λ_m, and the expectation value are respectively

$$p(m) = \mathrm{Tr}[(P_m \otimes I)\rho] = \mathrm{Tr}[P_m\,\mathrm{Tr}_B[\rho]] = \mathrm{Tr}[P_m\rho_A], \quad \mathbb{E}_\rho[M] = \mathrm{Tr}[(M_A \otimes I)\rho] = \mathrm{Tr}[M_A\rho_A]. \tag{1.44}$$

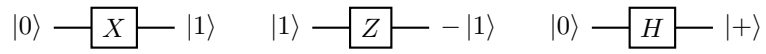
1.3. Quantum circuit

Nearly all quantum algorithms operate on multi-qubit quantum systems. When quantum operators operate on two or more qubits, writing down quantum states in terms of its components as in Eq. (1.29) quickly becomes cumbersome. The quantum circuit language offers a graphical and compact manner for writing down the procedure of applying a sequence of quantum operators to a quantum state. For more details see [NC00, Section 4.2, 4.3].

In the quantum circuit language, time flows from the left to right, i.e., the input quantum state appears on the left, and the quantum operator appears on the right, and each "wire" represents a qubit i.e.,



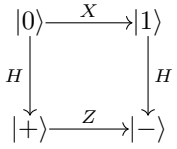
Here are a few examples:



which is a graphical way of writing

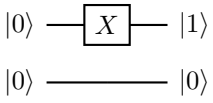
$$X|0\rangle = |1\rangle, \quad Z|1\rangle = -|1\rangle, \quad H|0\rangle = |+\rangle. \tag{1.45}$$

The relation between these states can be expressed in terms of the following diagram



(1.46)

Also verify that

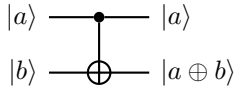


which is a graphical way of writing

$$(X \otimes I)|00\rangle = |10\rangle. \tag{1.47}$$

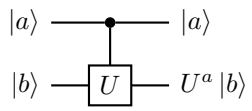
Note that the input state can be general, and in particular does not need to be a product state. For example, if the input is a Bell state (1.32), we just apply the quantum operator to $|00\rangle$ and $|11\rangle$, respectively and multiply the results by $1/\sqrt{2}$ and add together. To distinguish with other symbols, these single qubit gates may be either written as X, Y, Z, H or (using the roman font) $\mathrm{X}, \mathrm{Y}, \mathrm{Z}, \mathrm{H}$.

The quantum circuit for the CNOT gate is



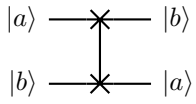
Here the "dot" means that the quantum gate connected to the dot only becomes active if the state of the qubit 0 (called the control qubit) is $a = 1$. This justifies the name of the CNOT gate (controlled NOT).

Similarly,

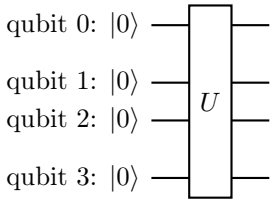


is the controlled U gate for some unitary U. Here $U^a = I$ if $a = 0$. The CNOT gate can be obtained by setting $U = X$.

Another commonly used two-qubit gate is the SWAP gate, which swaps the state in the 0-th and the 1-st qubits.

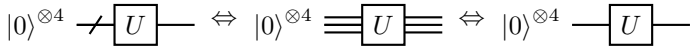


Quantum operators applied to multiple qubits can be written in a similar manner:

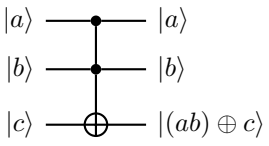


For a multi-qubit quantum circuit, unless stated otherwise, the first qubit will be referred to as the qubit 0, and the second qubit as the qubit 1, etc.

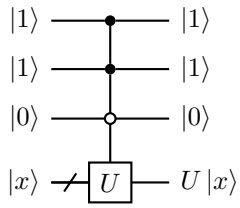
When the context is clear, we may also use a more compact notation for the multi-qubit quantum operators:



One useful multiple qubit gate is the Toffoli gate (or controlled-controlled-NOT, CCNOT gate).



We may also want to apply a n-qubit unitary U only when certain conditions are met



where the empty circle means that the gate being controlled only becomes active when the value of the control qubit is 0. This can be used to write down the quantum "if" statements, i.e., when the qubits $0, 1$ are at the $|1\rangle$ state and the qubit 2 is at the $|0\rangle$ state, then apply U to $|x\rangle$.

A set of qubits is often called a register (or quantum variable). For example, in the picture above, the main quantum state of interest (an n qubit quantum state $|x\rangle$) is called the system register. The first 3 qubits can be called the control register.

1.4. Copy operation and no-cloning theorem

One of the most striking early results of quantum computation is the no-cloning theorem (by Wootters and Zurek, as well as Dieks in 1982), which forbids generic quantum copy operations (see also [NC00, Section 12.1]). The no-deleting theorem is a consequence of linearity of quantum mechanics.

Assume there is a unitary operator U that acts as the copy operations, i.e.,

$$U|x\rangle \otimes |s\rangle = |x\rangle \otimes |x\rangle, \tag{1.48}$$

for any black-box state x, and a chosen target state $|s\rangle$ (e.g. $|0^n\rangle$). Then take two states $|x_1\rangle, |x_2\rangle$, we have

$$U|x_1\rangle \otimes |s\rangle = |x_1\rangle \otimes |x_1\rangle, \quad U|x_2\rangle \otimes |s\rangle = |x_2\rangle \otimes |x_2\rangle. \tag{1.49}$$

Taking the inner product of the two equations, we have

$$\langle x_1|x_2\rangle = \langle x_1|x_2\rangle^2, \tag{1.50}$$

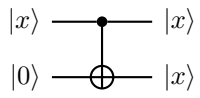
which implies $\langle x_1|x_2\rangle = 0$ or 1. When $\langle x_1|x_2\rangle = 1$, $|x_1\rangle, |x_2\rangle$ refer to the same physical state. Therefore a cloning operator U can at most copy states which are orthogonal to each other, and a general quantum copy operation is impossible.

Given the ubiquity of the copy operation in scientific computing like $y = x$, the no-cloning theorem has profound implications. For instance, all classical iterative algorithms for solving linear systems require storing some intermediate variables. This operation is generally not possible, or at least cannot be efficiently performed.

There are two notable exceptions to the range of applications of the no-cloning theorem. The first is that we know how a quantum state is prepared, i.e., $|x\rangle = U_x|s\rangle$ for a *known* unitary U_x and some $|s\rangle$. Then we can of course copy this specific vector $|x\rangle$ via

$$(I \otimes U_x)|x\rangle \otimes |s\rangle = |x\rangle \otimes |x\rangle. \tag{1.51}$$

The second is the copying of classical information. This is an application of the CNOT gate.



i.e.,

$$\mathrm{CNOT}|x, 0\rangle = |x, x\rangle, \quad x \in \{0, 1\}. \tag{1.52}$$

The same principle applies to copying classical information from multiple qubits. Fig. 1.1 gives an example of copying the classical information stored in 3 bits.

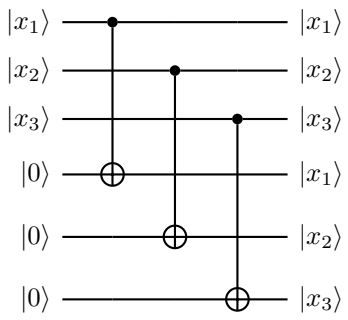


FIGURE 1.1. Copying classical information using multi-qubit CNOT gates.

In general, a multi-qubit CNOT operation can be used to perform the classical copying operation in the computational basis. Note that in the circuit model, this can be implemented with a depth 1 circuit, since they all act on different qubits.

Example 1.8. Let us verify that the CNOT gate does not violate the no-cloning theorem, i.e., it cannot be used to copy a superposition of classical bits $|x\rangle = a|0\rangle + b|1\rangle$. Direct calculation shows

$$\text{CNOT}\,|x\rangle \otimes |0\rangle = a\,|00\rangle + b\,|11\rangle \neq |x\rangle \otimes |x\rangle\,. \tag{1.53}$$

In particular, if $|x\rangle = |+\rangle$, then CNOT creates a Bell state.

◇

The quantum no-cloning theorem implies that there does not exist a unitary U that performs the deleting operation, which resets a black-box state $|x\rangle$ to $|0^n\rangle$. This is because such a deleting unitary can be viewed as a copying operation

$$U\,|0^n\rangle \otimes |x\rangle = |0^n\rangle \otimes |0^n\rangle\,. \tag{1.54}$$

Then take $|x_1\rangle, |x_2\rangle$ that are orthogonal to each other, apply the deleting gate, and compute the inner products, we obtain

$$0 = \langle x_1|x_2\rangle = \langle 0^n|0^n\rangle = 1, \tag{1.55}$$

which is a contradiction.

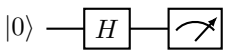
A more common way to express the no-deleting theorem is in terms of the time reversed dual of the no-cloning theorem: in general, given two copies of some arbitrary quantum state, it is impossible to delete one of the copies. More specifically, there is no unitary U performing the following operation using known states $|s\rangle, |s'\rangle$,

$$U\,|x\rangle\,|x\rangle\,|s\rangle = |x\rangle\,|0^n\rangle\,|s'\rangle \tag{1.56}$$

for an arbitrary unknown state $|x\rangle$ (Exercise 1.7).

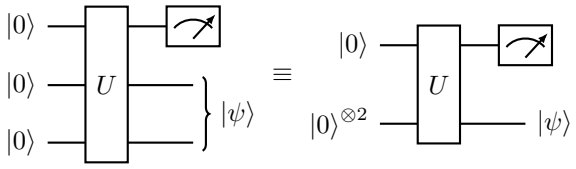
1.5. Measurement

The quantum measurement applied to any qubit, by default, measures the outcome in the computational basis. For example,



outputs 0 or 1 each w.p. 1/2. We may also measure some of the qubits in a multi-qubit system.

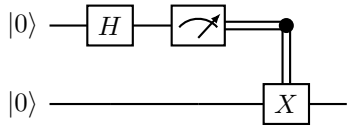
(1.57)



There are two important principles related to quantum measurements: the principle of deferred measurement, and the principle of implicit measurement. At a first glance, both principles may seem to be counterintuitive.

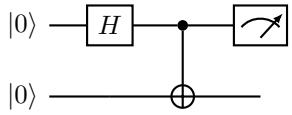
The principle of deferred measurement states that measurement operations can always be moved from an intermediate stage of a quantum circuit to the end of the circuit. This is because even if a measurement is performed as an intermediate step in a quantum circuit, and the result of the measurement is used to conditionally control subsequent quantum gates, such classical controls can always be replaced by quantum controls, and the result of the quantum measurement is postponed to later.

Example 1.9 (Deferring quantum measurements)**.** Consider the circuit



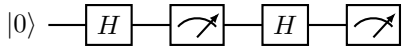
Here the double line denotes the classical control operation. The outcome is that qubit 0 has probability 1/2 of outputting 0, and the qubit 1 is at state $|0\rangle$. Qubit 0 also has probability 1/2 of outputting 1, and the qubit 1 is at state $|1\rangle$.

However, we may replace the classical control operation after the measurement by a quantum controlled X (which is CNOT), and measure the qubit 0 afterwards:

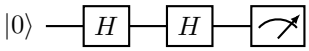


It can be verified that the result is the same. Note that CNOT acts as the classical copying operation. So qubit 1 really stores the classical information (i.e., in the computational basis) of qubit 0. ◇

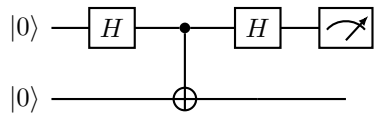
Example 1.10 (Deferred measurement requires extra qubits)**.** The procedure of deferring quantum measurements using CNOTs is general, and important. Consider the following circuit:



The probability of obtaining $0, 1$ is $1/2$, respectively. However, if we simply "defer" the measurement to the end by removing the intermediate measurement, we obtain



The result of the measurement is deterministically 0! The correct way of deferring the intermediate quantum measurement is to introduce another qubit



Measuring the qubit 0, we obtain 0 or 1 w.p. 1/2, respectively. Hence when deferring quantum measurements, it is necessary to store the intermediate information in extra (ancilla) qubits, even if such information is not used afterwards. ◇

The principle of implicit measurements states that at the end of a quantum circuit, any unmeasured qubit may be assumed to be measured. More specifically, assume the quantum system consists of two subsystems A and B. If qubits A are to be measured at the end of the circuits, the results of the measurements does not depend on whether the qubits B are measured or not. Recall from Eq. (1.44) that a measurement on the subsystem A only depends on the reduced density matrix ρ_A. So we only need to show that ρ_A does not depend on the measurement in B. To see why this is the case, let $\{P_i\}$ be the projectors onto the computational basis of B. Before the measurement, the density operator is ρ. If we measure the subsystem B, the resulting density operator is transformed into

$$\rho' = \sum_i (I \otimes P_i)\rho(I \otimes P_i). \tag{1.58}$$

Then it can be verified that

$$\rho'_A = \mathrm{Tr}_B[\rho'] = \mathrm{Tr}_B\left[\rho \sum_i (I \otimes P_i)\right] = \mathrm{Tr}_B[\rho] = \rho_A. \tag{1.59}$$

This proves the principle of implicit measurements.

By definition, the output of all quantum algorithms must be obtained through measurements, and hence the measurement outcome is probabilistic in general. If the goal is to compute the expectation value of a quantum observable M_A acting on a subsystem A, then its variance is

$$\mathrm{Var}_\rho[M_A] = \mathrm{Tr}[M_A^2\rho_A] - (\mathrm{Tr}[M_A\rho_A])^2. \tag{1.60}$$

The number of samples $\mathcal{N}$ needed to estimate $\mathrm{Tr}[M_A\rho_A]$ to *additive* precision ϵ satisfies

$$\sqrt{\frac{\mathrm{Var}_\rho[M_A]}{\mathcal{N}}} \le \epsilon \quad \Rightarrow \quad \mathcal{N} \ge \frac{\mathrm{Var}_\rho[M_A]}{\epsilon^2}, \tag{1.61}$$

which only depends on ρ_A.

Example 1.11 (Estimating success probability on one qubit)**.** Let A be the single qubit to be measured in the computational basis, and we are interested in the accuracy in estimating the success probability of obtaining 1, i.e., p. This can be realized as an expectation value with $M_A = |1\rangle\langle 1|$, and $p = \mathrm{Tr}[M_A\rho_A]$. Note that $M_A^2 = M_A$, then

$$\mathrm{Var}_\rho[M_A] = p - p^2 = p(1-p). \tag{1.62}$$

Hence to estimate p to *additive* error ϵ, the number of samples needed satisfies

$$\mathcal{N} \ge \frac{p(1-p)}{\epsilon^2}. \tag{1.63}$$

Note that if p is close to 0 or 1, the number of samples needed is also very small: indeed, the outcome of the measurement becomes increasing deterministic in this case!

If we are interested in estimating p to *multiplicative* accuracy ϵ, then the number of samples is

$$\mathcal{N} \geq \frac{p(1-p)}{p^2\epsilon^2} = \frac{1-p}{p\epsilon^2}, \tag{1.64}$$

and the task becomes increasingly more difficult when p approaches 0. ◇

1.6. Linear error growth and Duhamel's principle

If a quantum algorithm denoted by a unitary U can be decomposed into a series of simpler unitaries as $U = U_K \cdots U_1$, and if we can implement each U_i to precision ϵ, then what is the global error? We now introduce a simple technique connecting the local error with the global error. In the context of quantum computation, this is often referred to as the "hybrid argument".

Proposition 1.12 (Hybrid argument)**.** *Given unitaries $U_1, \widetilde{U}_1, \ldots, U_K, \widetilde{U}_K \in \mathbb{C}^{N\times N}$ satisfying*

$$\left\| U_i - \widetilde{U}_i \right\| \leq \epsilon, \quad \forall i = 1, \ldots, K, \tag{1.65}$$

we have

$$\left\| U_K \cdots U_1 - \widetilde{U}_K \cdots \widetilde{U}_1 \right\| \leq K\epsilon. \tag{1.66}$$

Proof. Use a telescoping series

$$\begin{aligned} & U_K \cdots U_1 - \widetilde{U}_K \cdots \widetilde{U}_1 \\ =& (U_K \cdots U_2 U_1 - U_K \cdots U_2 \widetilde{U}_1) + (U_K \cdots U_3 U_2 \widetilde{U}_1 - U_K \cdots U_3 \widetilde{U}_2 \widetilde{U}_1) + \cdots \\ & + (U_K \widetilde{U}_{K-1} \cdots \widetilde{U}_1 - \widetilde{U}_K \widetilde{U}_K \cdots \widetilde{U}_1) \\ =& U_K \cdots U_2 (U_1 - \widetilde{U}_1) + U_K \cdots U_3 (U_2 - \widetilde{U}_2) + \cdots + (U_K - \widetilde{U}_K) \widetilde{U}_{K-1} \cdots \widetilde{U}_1. \end{aligned} \tag{1.67}$$

Since all $U_i, \widetilde{U}_i$ are unitary matrices, we readily have

$$\left\| U_K \cdots U_1 - \widetilde{U}_K \cdots \widetilde{U}_1 \right\| \leq \sum_{i=1}^{K} \left\| U_i - \widetilde{U}_i \right\| \leq K\epsilon. \tag{1.68}$$

□

In other words, if we can implement each local unitary to precision ϵ, the global error grows at most *linearly* with respect to the number of gates and is bounded by $K\epsilon$. The telescoping series Eq. (1.67), as well as the hybrid argument can also be seen as a discrete analogue of the variation of constants method (also called Duhamel's principle).

Proposition 1.13 (Duhamel's principle for Hamiltonian simulation)**.** *Let $U(t), \widetilde{U}(t) \in \mathbb{C}^{N\times N}$ satisfy*

$$\mathrm{i}\partial_t U(t) = HU(t), \quad \mathrm{i}\partial_t \widetilde{U}(t) = H\widetilde{U}(t) + B(t), \quad U(0) = \widetilde{U}(0) = I, \tag{1.69}$$

where $H \in \mathbb{C}^{N\times N}$ is a Hermitian matrix, and $B(t) \in \mathbb{C}^{N\times N}$ is an arbitrary matrix. Then

$$\widetilde{U}(t) = U(t) - \mathrm{i} \int_0^t U(t-s) B(s) \, \mathrm{d}s, \tag{1.70}$$

and

$$\left\|\widetilde{U}(t) - U(t)\right\| \leq \int_0^t \|B(s)\| \, \mathrm{d}s. \tag{1.71}$$

PROOF. Directly verify that Eq. (1.70) is the solution to the differential equation. □

As a special case, consider $B(t) = E(t)\widetilde{U}(t)$, then Eq. (1.70) becomes

$$\widetilde{U}(t) = U(t) - \mathrm{i} \int_0^t U(t-s)E(s)\widetilde{U}(s) \, \mathrm{d}s, \tag{1.72}$$

and

$$\left\|\widetilde{U}(t) - U(t)\right\| \leq \int_0^t \|E(s)\| \, \mathrm{d}s. \tag{1.73}$$

This is a direct analogue of the hybrid argument in the continuous setting.

1.7. Universal gate sets and reversible computation

In classical computation, there are many universal gate sets, in the sense that any classical gate can be represented as a combination of gates from the set. For example, the NAND gate ("Not AND") alone forms a universal gate set [NC00, Section 3.1.2]. The NOR gate ("Not OR") is also a universal gate set.

In the quantum setting, any unitary operator on n qubits can be implemented using 1- and 2-qubit gates[NC00, Section 4.5]. It is desirable to come up with a set of discrete universal gates, but this means that we need to give up the notion that the unitary U can be exactly represented. Instead, a set of quantum gates $\mathcal{S}$ is universal if given any unitary operator U and desired precision ϵ, we can find $U_1, \ldots, U_m \in \mathcal{S}$ such that

$$\|U - U_m U_{m-1} \cdots U_1\| \leq \epsilon. \tag{1.74}$$

Here $\|A\| = \sup_{\langle\psi|\psi\rangle=1} \|A|\psi\rangle\|$ is the operator norm (also called the spectral norm) of A, and $\||\psi\rangle\| = \sqrt{\langle\psi|\psi\rangle}$ is the vector 2-norm). There are many possible choices of universal gate sets, e.g. $\{H, T, \mathrm{CNOT}\}$. Another universal gate set is $\{H, \mathrm{Toffoli}\}$, which only involves real numbers.

Are some universal gate sets better than others? The Solovay-Kitaev theorem states that all choices of universal gate sets are asymptotically equivalent (see e.g. [Chi21, Chapter 2]):

THEOREM 1.14 (Solovay-Kitaev). *Let $\mathcal{S}, \mathcal{T}$ be two universal gate sets that are closed under inverses. Then any m-gate circuit using the gate set $\mathcal{S}$ can be implemented to precision ϵ using a circuit of $\mathcal{O}(m \cdot \mathrm{polylog}(m/\epsilon))$ gates from the gate set $\mathcal{T}$, and there is a classical algorithm for finding this circuit in time $\mathcal{O}(m \cdot \mathrm{polylog}(m/\epsilon))$.*

Another natural question is about the computational power of quantum computers. Perhaps surprisingly, it is very difficult to prove that quantum computer is *more powerful than* classical computer. But is quantum computer *at least as powerful as* classical computers? The answer is yes! More specifically, any classical circuit can also be asymptotically efficiently implemented using a quantum circuit.

The proof rests on that the classical universal gate can be efficiently simulated using quantum circuits. Note that this is not a straightforward process: NAND, and other classical gates (such as AND, OR etc.) are not reversible gates! Hence the first step is to perform classical computation

with *reversible* gates. More specifically, any irreversible classical gate $x \mapsto f(x)$ can be made into a reversible classical gate

$$(x, y) \mapsto (x, y \oplus f(x)). \tag{1.75}$$

In particular, we have $(x, 0) \mapsto (x, f(x))$ computed in a reversible way. The key idea is to store all intermediate steps of the computation (see [NC00, Section 3.2.5] for more details).

On the quantum computer, storing all intermediate computational steps indefinitely creates two problems: (1) tremendous waste of quantum resources (2) the intermediate results stored in some extra qubits are still entangled to the quantum state of interest. So if the environments interfere with intermediate results, the quantum state of interest is also affected.

Fortunately, both problems can be solved by a step called "uncomputation". In order to implement a Boolean function $f : \{0, 1\}^n \to \{0, 1\}^m$, we assume there is an oracle

$$|0^m\rangle |x\rangle \mapsto |f(x)\rangle |x\rangle , \tag{1.76}$$

where $|0^m\rangle$ comes from a m-qubit output register. The oracle is often further implemented with the help of a working register (a.k.a. "garbage" register) such that

$$U_f : |0^w\rangle |0^m\rangle |x\rangle \mapsto |g(x)\rangle |f(x)\rangle |x\rangle . \tag{1.77}$$

From the no-deleting theorem, there is no generic unitary operator that can set a black-box state to $|0^w\rangle$. In order to set the working register back to $|0^w\rangle$ while keeping the input and output state, we introduce yet another m-qubit ancilla register initialized at $|0^m\rangle$. Then we can use an n-qubit CNOT controlled on the output register and obtain

$$|0^m\rangle |g(x)\rangle |f(x)\rangle |x\rangle \mapsto \underbrace{|f(x)\rangle}_{\text{ancilla}} \underbrace{|g(x)\rangle}_{\text{working}} \underbrace{|f(x)\rangle}_{\text{output}} \underbrace{|x\rangle}_{\text{input}} . \tag{1.78}$$

It is important to remember that in the operation above, the multi-qubit CNOT gate only performs the classical copying operation in the computational basis, and does not violate the no-cloning theorem.

Recall that $U_f^{-1} = U_f^\dagger$, we have

$$(I_m \otimes U_f^\dagger) |f(x)\rangle |g(x)\rangle |f(x)\rangle |x\rangle = |f(x)\rangle (U_f^\dagger |g(x)\rangle |f(x)\rangle |x\rangle) = |f(x)\rangle |0^w\rangle |0^m\rangle |x\rangle . \tag{1.79}$$

Finally we apply an n-qubit SWAP operator on the ancilla and output registers to obtain

$$|f(x)\rangle |0^w\rangle |0^m\rangle |x\rangle \mapsto |0^m\rangle |0^w\rangle |f(x)\rangle |x\rangle . \tag{1.80}$$

After this procedure, both the ancilla and the working register are set to the initial state. They are no longer entangled to the input or output register, and can be reused for other purposes. This procedure is called *uncomputation*. The circuit is shown in Fig. 1.2.

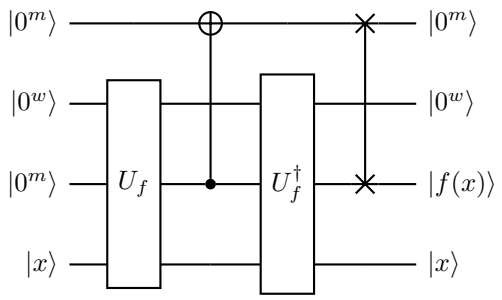


FIGURE 1.2. Circuit for uncomputation. The CNOT and SWAP operators indicate the multi-qubit copy and swap operations, respectively.

Remark 1.15 (Discarding working registers)**.** After the uncomputation as shown in Fig. 1.2, the first two registers are unchanged before and after the application of the circuit (though they are changed during the intermediate steps). Therefore Fig. 1.2 effectively implements a unitary

$$(I_{m+w} \otimes V_f) |0^m\rangle |0^w\rangle |0^m\rangle |x\rangle = |0^m\rangle |0^w\rangle |f(x)\rangle |x\rangle \tag{1.81}$$

or equivalently

$$V_f |0^m\rangle |x\rangle = |f(x)\rangle |x\rangle . \tag{1.82}$$

In the definition of V_f, all working registers have been discarded (on paper). This allows us to simplify the notation and focus on the essence of the quantum algorithms under study. ◇

Using the technique of uncomputation, if the map $x \mapsto f(x)$ can be efficiently implemented on a classical computer, then we can implement this map efficiently on a quantum computer as well. To do this, we first turn it into a reversible map (1.75). All reversible single-bit and two-bit classical gates can be implemented using single-qubit and two-qubit quantum gates. So the reversible map can be made into a unitary operator

$$U_f : |x, y\rangle \mapsto |x, y \oplus f(x)\rangle \tag{1.83}$$

on a quantum computer. This proves that a quantum computer is at least as powerful as classical computers.

The unitary transformation U_f in (1.83) can be applied to any superposition of states in the computational basis, e.g.

$$U_f : \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x, 0^m\rangle \mapsto \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x, f(x)\rangle . \tag{1.84}$$

This does not necessarily mean that we can efficient implement the map $|x\rangle \mapsto |f(x)\rangle$. However, if f is a bijection, and we have access to the inverse of the reversible circuit for computing f^{-1}, then we may use the technique of uncomputation to implement such a map (Exercise 1.9).

1.8. Fixed point number representation and classical arithmetic operations

Let $[N] = \{0, 1, \ldots, N-1\}$. Any integer $k \in [N]$ where $N = 2^n$ can be expressed as an n-bit string as $k = k_{n-1} \cdots k_0$ with $k_i \in \{0, 1\}$. This is called the binary representation of the integer k. It should be interpreted as

$$k = \sum_{i \in [n]} k_i 2^i. \tag{1.85}$$

The number k divided by 2^m ($0 \leq m \leq n$) can be written as (note that the decimal is shifted to be after k_m):

$$a = \frac{k}{2^m} = \sum_{i \in [n]} k_i 2^{i-m} =: (k_{n-1} \cdots k_m . k_{m-1} \cdots k_0). \tag{1.86}$$

The most common case is $m = n$, where

$$a = \frac{k}{2^n} = \sum_{i \in [n]} k_i 2^{i-n} =: (0 . k_{n-1} \cdots k_0). \tag{1.87}$$

Sometimes we may also write $a = 0.k_1 \cdots k_n$, so that k_i is the i-th decimal of a in the binary representation. For a given floating number $0 \leq a < 1$ written as

$$a = (0 . k_1 \cdots k_n k_{n+1} \cdots), \tag{1.88}$$

the number $(0.k_1 \cdots k_n)$ is called the n-bit fixed point representation of a. Therefore to represent a to additive precision ϵ, we will need $n = \lceil \log_2 \epsilon \rceil$ qubits. If the sign of a is also important, we may reserve one extra bit to indicate its sign.

Together with the reversible computational model, we can perform classical arithmetic operations, such as $(x, y) \mapsto x + y$, $(x, y) \mapsto xy$, $x \mapsto x^\alpha$, $x \mapsto \cos(x)$ etc. using reversible quantum circuits. The number of ancilla qubits, and the number of elementary gates needed for implementing such quantum circuits is $\mathcal{O}(\text{poly}(n))$ (see [RP11, Chapter 6] for more details).

It is worth commenting that while quantum computer is *theoretically* as powerful as classical computers, there is a *very significant* overhead in implementing reversible classical circuits on quantum devices, both in terms of the number of ancilla qubits and the circuit depth.

1.9. Fault tolerant computation

All previous discussions assume that quantum operations can be perfectly performed. Due to the immense technical difficulty for realizing quantum computers, both quantum gates and quantum measurements may involve (significant) errors, particularly on near-term quantum devices. However, the threshold theorem states that if the noise in individual quantum gates is below a certain constant threshold (around 10^{-4} or above), it is possible to efficiently perform an arbitrarily large quantum computation with any desired precision (see [NC00, Section 10.6]). This procedure requires quantum error correction protocols.

This course will not discuss any details on quantum error corrections. We always assume fault-tolerant protocols have been implemented, and all errors come from either approximation errors at the mathematical level, or Monte Carlo errors in the readout process due to the probabilistic nature of the measurement process.

1.10. Complexity of quantum algorithms

Let n be the number of qubits needed to represent the input. A quantum algorithm is efficient if the number of gates in the quantum circuit is $\mathcal{O}(\text{poly}(n))$. Due to the probabilistic nature of the measurement outcome, we are typically satisfied if a quantum algorithm can produce the correct answer with sufficiently high probability p. For a decision problem that asks for a binary answer 0 or 1, we require $p > 2/3$ (or at least $p > 1/2 + 1/\text{poly}(n)$). For other problems that we have an efficient procedure to check the correctness of the answer, we require $p = \Omega(1)$. Repeating this process many times and apply the Chernoff bound [NC00, Box 3.4], we can make the probability of outputting an incorrect answer vanishingly small.

In quantum algorithms, the computational cost is often measured in terms of the *query complexity*. Assume that we have access to black-box unitary operator U_f (e.g. the one used in the reversible computation), which is often called a quantum *oracle*. Our goal is to perform a given task using as few queries as possible to U_f.

Example 1.16 (Query access to a boolean function)**.** Let $f : \{0,1\}^n \to \{0,1\}$ be a boolean function, which can be queried via the following unitary

$$U_f |x\rangle = (-1)^{f(x)} |x\rangle . \tag{1.89}$$

This is called a *phase kickback*, i.e., the value of $f(x)$ is returned as a phase factor. The phase kickback is an important tool in many quantum algorithms, e.g. Grover's algorithm. Note that 1) U_f can be applied to a superposition of states in the computational basis, and 2) Having query access to $f(x)$ does not mean that we know everything about $f(x)$, e.g. finding the set $\{ x \mid f(x) = 0 \}$ can still be a difficult task. ◇

Example 1.17 (Partially specified quantum oracles)**.** When designing quantum algorithms, it is common that we are not interested in the behavior of the entire unitary matrix U_f, but only U_f applied to certain vectors. For instance, for a $(n+1)$-qubit state space, we are only interested in

$$U_f |0\rangle |x\rangle = |0\rangle (A |x\rangle) + |1\rangle (B |x\rangle). \tag{1.90}$$

This means that we have only defined the first block-column of U_f as (remember that the row-major order is used)

$$U_f = \begin{pmatrix} A & * \\ B & * \end{pmatrix} \tag{1.91}$$

Here A, B are $N \times N$ matrices, and $*$ stands for an arbitrary $N \times N$ matrix so that U_f is unitary. Of course in order to implement U_f into quantum gates, we still need to specify the content of $*$. However, at the conceptual level, the partially specified unitary (1.90) simplifies the design process of quantum oracles. ◇

The concept of query complexity hides the implementation details of U_f, and in some cases we can prove lower bounds on the number of queries to solve a certain problem, e.g. in the case of Grover's search (proving a lower bound of the number of gates among all quantum algorithms can be much harder). Furthermore, once we have access to the number of elementary gates needed to implement U_f, we obtain immediately the *gate complexity* of the total algorithm. However, some queries can be (provably) difficult to implement, and then there can be a large gap between the query complexity and gate complexity. In order to obtain a meaningful query complexity analysis, one should also make sure that other components of the quantum algorithm will not end up dominating the total gate complexity, when all factors are taken into account.

Another important measure of the complexity is the *circuit depth*, i.e., the maximum number of gates along any path from an input to an output. Since quantum gates can be performed in parallel, the circuit depth is approximately equivalent to the concept of "wall-clock time" in classical computation, i.e., the real time needed for a quantum computer to carry out a certain task. Since quantum states can only be preserved for a short period of time (called the *coherence time*), the circuit depth also provides an approximate measure of whether the quantum algorithm exceeds the coherence limit of a given quantum computer. In many scenarios, the maximum coherence time is the most severe limiting factor. When possible, it is often desirable to reduce the circuit depth, even if it means that the quantum circuit needs to be carried out many more times.

Let us summarize the basic components of a typical quantum algorithm: the set of qubits can be separated into system registers (storing quantum states of interest) and ancilla registers (auxiliary registers needed to implement the unitary operation acting on system registers). Starting from an initial state, apply a series of one-/two-qubit gates, and perform measurements. Uncomputation should be performed whenever possible. Within the ancilla registers, if a register can be "freed" after the uncomputation, it is called a working register. Since working registers can be reused for other purposes, the cost of working registers is often not (explicitly) factored into the asymptotic cost analysis in the literature.

1.11. Notation

We use $\|\cdot\|$ to denote vector or matrix 2-norm: when v is a vector we denote by $\|v\|$ its 2-norm, and when A is matrix we denote by $\|A\|$ its operator norm. Other matrix and vector norms will be introduced when needed. Unless otherwise specified, a vector $v \in \mathbb{C}^N$ is an unnormalized vector, and a normalized vector (stored as a quantum state) is denoted by $|v\rangle = v/\|v\|$. A vector v can be expressed in terms of j-th component as $v = (v_j)$ or $(v)_j = v_j$. We use a 0-based indexing, i.e., $j = 0, \ldots, N-1$ or $j \in [N]$. When 1-based indexing is used, we will explicitly write $j = 1, \ldots, N$. We use the following asymptotic notations besides the usual $\mathcal{O}$ (or "big-O") notation: we write $f = \Omega(g)$ if $g = \mathcal{O}(f)$; $f = \Theta(g)$ if $f = \mathcal{O}(g)$ and $g = \mathcal{O}(f)$; $f = \widetilde{\mathcal{O}}(g)$ if $f = \mathcal{O}(g \operatorname{polylog}(g))$.

Exercise 1.1. Prove that any unitary matrix $U \in \mathbb{C}^{N\times N}$ can be written as $U = e^{\mathrm{i}H}$, where H is an Hermitian matrix.

Exercise 1.2. Prove Eq. (1.26).

Exercise 1.3. Write down the matrix representation of the SWAP gate, as well as the $\sqrt{\text{SWAP}}$ and $\sqrt{\text{iSWAP}}$ gates.

Exercise 1.4. Prove that the Bell state Eq. (1.32) cannot be written as any product state $|a\rangle \otimes |b\rangle$.

Exercise 1.5. Prove Eq. (1.39) holds for a general mixed state ρ.

Exercise 1.6. Prove that an ensemble of admissible density operators is also a density operator.

Exercise 1.7. Prove the no-deleting theorem in Eq. (1.56).

Exercise 1.8. Work out the circuit for implementing Eq. (1.78) and Eq. (1.80).

Exercise 1.9. Prove that if $f : \{0,1\}^n \to \{0,1\}^n$ is a bijection, and we have access to the inverse mapping f^{-1}, then the mapping $U_f : |x\rangle \mapsto |f(x)\rangle$ can be implemented on a quantum computer.

Exercise 1.10. Prove Eq. (1.58) and Eq. (1.59).

CHAPTER 2

Grover's algorithm

Now we will introduce a few basic quantum algorithms, of which the ideas and variants are present in numerous other quantum algorithms. Hence they are called "quantum primitives". There is no official ruling on which quantum algorithms qualify for being included in the set of quantum primitives, but the membership of Grover's algorithm, quantum Fourier transform, quantum phase estimation, and Trotter based Hamiltonian simulation should not be controversial. We first introduce Deutsch's algorithm, which is arguably one of the simplest quantum algorithms carrying out a well-defined task.

2.1. The first quantum algorithm: Deutsch's algorithm

Assume we have two boxes, each of them may contain either an apple or an orange. We would like to answer: whether the two boxes contain the same type of fruit (but do not need to answer whether it is apple or orange).

This seems to be a weird question. If the content of a box can only be checked by opening it, then to answer the question we would need to open *two* boxes and check what is inside. It is impossible to answer whether the fruit types are the same without knowing the types! Nevertheless, this is precisely the question to be addressed by Deutsch's algorithm.

Box 0 Box 1

Mathematically, consider a boolean function $f : \{0,1\} \to \{0,1\}$. The question is whether $f(0) = f(1)$ or $f(0) \neq f(1)$? A quantum fruit-checker assumes the access to f via the following quantum oracle:

$$U_f \left|x, y\right\rangle = \left|x, y \oplus f(x)\right\rangle, \quad x, y \in \{0,1\}. \tag{2.1}$$

A classical fruit-checker can only query U_f in the computational basis as

$$U_f \left|0,0\right\rangle = \left|0, f(0)\right\rangle, \quad U_f \left|1,0\right\rangle = \left|1, f(1)\right\rangle. \tag{2.2}$$

After these *two* queries, we can measure qubit 1 with a deterministic outcome, and answer whether $f(0) = f(1)$. However, a quantum fruit-checker can apply U_f to a linear combination of states in the computational basis.

Let us first check that U_f is unitary:

$$\begin{aligned}\langle x', y'|U_f^\dagger U_f|x, y\rangle =& \langle x', y' \oplus f(x')|x, y \oplus f(x)\rangle \\ =& \langle x'|x\rangle \langle y' \oplus f(x')|y \oplus f(x)\rangle \\ =& \delta_{x,x'}\delta_{y,y'}\end{aligned} \tag{2.3}$$

which gives $U_f^\dagger U_f = I$.

The idea behind Deutsch's algorithm is to convert the oracle (2.1) into a phase kickback in Eq. (1.89). Take $|y\rangle = |-\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$. Then

$$U_f |x, y\rangle = \frac{1}{\sqrt{2}} (|x, f(x)\rangle - |x, 1 \oplus f(x)\rangle) = (-1)^{f(x)} |x, y\rangle . \tag{2.4}$$

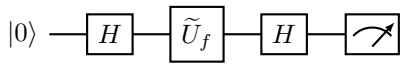
Note that $|y\rangle = HX|0\rangle$, Eq. (2.4) can also be interpreted as

$$(I \otimes XH)U_f(I \otimes HX)|x, 0\rangle = (-1)^{f(x)} |x, 0\rangle . \tag{2.5}$$

The application of XH can be viewed as the step of uncomputation. Neglecting the qubit 1 which does not change before and after the application, we can focus on the first qubit only, which effectively defines a unitary

$$\widetilde{U}_f |x\rangle = (-1)^{f(x)} |x\rangle . \tag{2.6}$$

Hence the information of $f(x)$ is stored as a phase factor (0 or π). Recall that using a Hadamard gate $H|0\rangle = |+\rangle, H|1\rangle = |-\rangle$, the quantum circuit of Deutsch's algorithm is



or in the commonly seen form in Fig. 2.1.

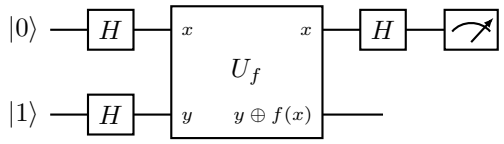


FIGURE 2.1. Quantum circuit for Deutsch's algorithm.

The answer is embedded in the measurement outcome of qubit 0. To verify this:

$$\begin{aligned}|0, 1\rangle \xrightarrow{H\otimes H} |+, -\rangle =& \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes |-\rangle \\ \xrightarrow{U_f}& \frac{1}{\sqrt{2}} \left((-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle\right) \otimes |-\rangle \\ \xrightarrow{H\otimes I}& \frac{1}{2} \left((-1)^{f(0)} + (-1)^{f(1)}\right) |0, -\rangle \\ &+ \frac{1}{2} \left((-1)^{f(0)} - (-1)^{f(1)}\right) |1, -\rangle .\end{aligned} \tag{2.7}$$

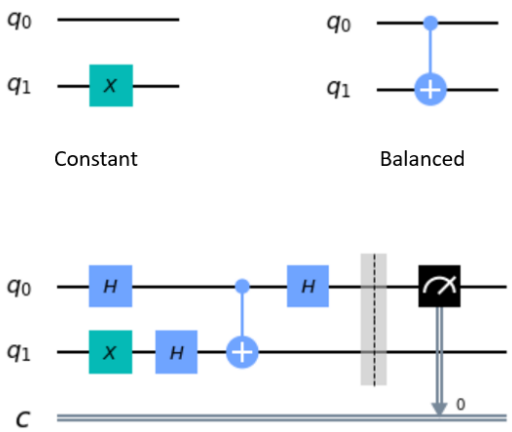
So if $f(0) = f(1)$, the final state is $\pm|0, -\rangle$. Measuring qubit 0 returns 0 deterministically (the globally phase factor is irrelevant). Similarly if $f(0) \neq f(1)$, the final state is $\pm|1, -\rangle$. Measuring qubit 0 returns 1 deterministically. In summary, only *one* query to U_f is sufficient to answer

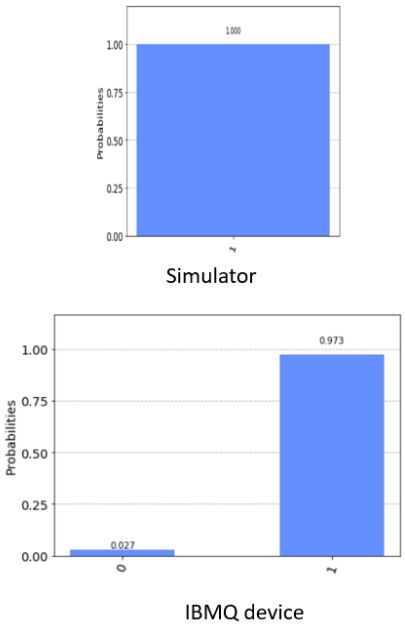
whether the two boxes contain the same type of fruit. The procedure is equally counterintuitive. Note that a classical fruit checker as implemented in Eq. (2.2) naturally receives the information by measuring qubit 1. On the other hand, Deutsch's algorithm only uses qubit 1 as a signal qubit, and all the information is retrieved by measuring qubit 0, which, at least from the classical perspective of Eq. (2.2), seems to contain no information at all!

Now we have seen that in terms of the *query complexity*, a quantum fruit-checker is clearly more efficient. However, it is a fair question how to implement the oracle U_f, especially in a way that somehow does not already reveal the values of $f(0), f(1)$. We give the implementation of some cases of U_f in Example 2.2. In general, proving the query complexity alone may not be convincing enough that quantum computers are better than classical computers, and the gate complexity matters. This will not be the last time we hide away such "implementation details" of quantum oracles.

Remark 2.1 (Deutsch–Jozsa algorithm)**.** The single-qubit version of the Deutsch algorithm can be naturally generalized to the n-qubit version, called the Deutsch–Jozsa algorithm. Given $N = 2^n$ boxes with an apple or an orange in each box, and the promise that either 1) all boxes contain the same type of fruit or 2) exactly half of the boxes contain apples and the other half contain oranges, we would like to distinguish the two cases. Mathematically, given the promise that a boolean function $f : \{0,1\}^n \to \{0,1\}$ is either a constant function (i.e., $|\{\, x \mid f(x) = 0 \,\}| = 0$ or 2^n) or a balanced function (i.e., $|\{\, x \mid f(x) = 0 \,\}| = 2^{n-1}$), we would like to decide to which type f belongs. We refer to [NC00, Section 1.4.4] for more details. ◇

Example 2.2 (Qiskit example of Deutsch's algorithm)**.** For $f(0) = f(1) = 1$ (constant case), we can use $U_f = I \otimes X$. For $f(0) = 0, f(1) = 1$ (balanced case), we can use $U_f = \text{CNOT}$.





◇

2.2. Unstructured search problem

Assume we have $N = 2^n$ boxes, and we are given the promise that only one of the boxes contains an orange, and each of the remaining boxes contains an apple. The goal is to find the box that contains the orange.

Mathematically, given a boolean function $f : \{0,1\}^n \to \{0,1\}$ and the promise that there exists a unique marked state x_0 that $f(x_0) = 1$, we would like to find x_0. This is called an unstructured search problem. Classically, there is no simpler methods than opening $(N-1)$ boxes in the worst case to determine x_0.

The quantum algorithm below, called Grover's algorithm, relies on access to an oracle

$$U_f |x, y\rangle = |x, y \oplus f(x)\rangle , \quad x \in \{0,1\}^n, y \in \{0,1\}, \tag{2.8}$$

and can find x_0 using $\mathcal{O}(\sqrt{N})$ queries. A classical computer again can only query U_f in the computational basis, and Grover's algorithm achieves a *quadratic speedup* in terms of the query complexity.

The origin of the quadratic speedup can be summarized as follows: while classical probabilistic algorithms work with probability densities, quantum algorithms work with wavefunction amplitudes, of which the square gives the probability densities. More specifically, we start from a uniform superposition of all states as the initial state

$$|\psi_0\rangle = \frac{1}{\sqrt{N}} \sum_{x \in [N]} |x\rangle . \tag{2.9}$$

This state can be prepared using Hadamard gates as

$$|\psi_0\rangle = H^{\otimes n} |0^n\rangle . \tag{2.10}$$

We would like to *amplify* the desired amplitude corresponding to $|x_0\rangle$ from $1/\sqrt{N}$ to $\sqrt{p} = \Omega(1)$. We demonstrate below that this requires $\mathcal{O}(\sqrt{N})$ queries to U_f. After this procedure, by measuring the final state in the computational basis, we obtain some output state $|x\rangle$. We can check whether $x = x_0$ by applying another query of U_f according to $U_f |x, 0\rangle = |x, f(x)\rangle$. The probability of obtaining $f(x) = 1$ is p. If $f(x) \neq 1$, we repeat the process. Then after $\mathcal{O}(1/p)$ times of repetition, we can obtain x_0 with high probability.

The first step of Grover's algorithm is to turn the oracle (2.8) into a phase kickback. For this we take $|y\rangle = |-\rangle$, and for any $x \in \{0,1\}^n$,

$$U_f |x, -\rangle = \frac{1}{\sqrt{2}} (|x, f(x)\rangle - |x, 1 \oplus f(x)\rangle) = (-1)^{f(x)} |x, -\rangle . \tag{2.11}$$

Any quantum state $|\psi\rangle$ can be decomposed as

$$|\psi\rangle = \alpha |x_0\rangle + \beta |\psi^\perp\rangle , \tag{2.12}$$

where $|\psi^\perp\rangle$ is the component of $|\psi\rangle$ orthogonal to $|x_0\rangle$, i.e., $\langle\psi^\perp|x_0\rangle = 0$. We have

$$U_f |\psi\rangle \otimes |-\rangle = (-\alpha |x_0\rangle + \beta |\psi^\perp\rangle) \otimes |-\rangle . \tag{2.13}$$

Here the minus sign is gained through the phase kickback. Discarding the $|-\rangle$ which is unchanged by applying U_f, we obtain an n-qubit unitary

$$R_{x_0}(\alpha |x_0\rangle + \beta |\psi^\perp\rangle) = -\alpha |x_0\rangle + \beta |\psi^\perp\rangle . \tag{2.14}$$

Therefore R_{x_0} is a *reflection operator* across the hyperplane orthogonal to $|x_0\rangle$, i.e., the Householder reflector

$$R_{x_0} = I - 2\,|x_0\rangle\,\langle x_0|\,. \tag{2.15}$$

Let us write

$$|\psi_0\rangle = \sin(\theta/2)\,|x_0\rangle + \cos(\theta/2)\,|\psi_0^\perp\rangle\,, \tag{2.16}$$

with $\theta = 2\arcsin\frac{1}{\sqrt{N}} \approx \frac{2}{\sqrt{N}}$, and $|\psi_0^\perp\rangle = \frac{1}{\sqrt{N-1}}\sum_{x\neq x_0}|x\rangle$ is a normalized state orthogonal to $|x_0\rangle$. Then

$$R_{x_0}\,|\psi_0\rangle = -\sin(\theta/2)\,|x_0\rangle + \cos(\theta/2)\,|\psi_0^\perp\rangle\,. \tag{2.17}$$

So $\mathrm{span}\{|x_0\rangle\,, |\psi_0^\perp\rangle\}$ is an invariant subspace of R_{x_0}.

The next key step is to consider another Householder reflector with respect to $|\psi_0\rangle$. For later convenience we add a global phase factor -1 (which is irrelevant to the physical outcome):

$$R_{\psi_0} = -(I - 2\,|\psi_0\rangle\,\langle\psi_0|). \tag{2.18}$$

Direct computation shows

$$\begin{aligned} R_{\psi_0}R_{x_0}\,|\psi_0\rangle =& R_{\psi_0}(|\psi_0\rangle - 2\sin(\theta/2)\,|x_0\rangle) \\ =& (|\psi_0\rangle - 4\sin^2(\theta/2)\,|\psi_0\rangle) + 2\sin(\theta/2)\,|x_0\rangle \\ =& \sin(\theta/2)(3 - 4\sin^2(\theta/2))\,|x_0\rangle + \cos(\theta/2)(1 - 4\sin^2(\theta/2))\,|\psi_0^\perp\rangle \\ =& \sin(3\theta/2)\,|x_0\rangle + \cos(3\theta/2)\,|\psi_0^\perp\rangle\,. \end{aligned} \tag{2.19}$$

So define $G = R_{\psi_0}R_{x_0}$ as the product of the two reflection operators (called the Grover operator), then it amplifies the angle from $\theta/2$ to $3\theta/2$. The geometric picture is in fact even clearer in Fig. 2.2 and the conclusion can be observed without explicit computation.

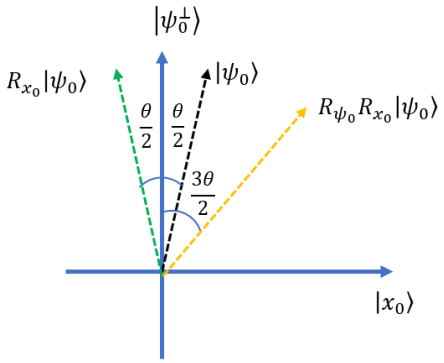


FIGURE 2.2. Geometric interpretation of one Grover iteration.

Applying the Grover operator k times, we obtain

$$G^k\,|\psi_0\rangle = \sin((2k+1)\theta/2)\,|x_0\rangle + \cos((2k+1)\theta/2)\,|\psi_0^\perp\rangle\,. \tag{2.20}$$

So for $\sin((2k+1)\theta/2) \approx 1$, we need $k \approx \frac{\pi}{2\theta} - \frac{1}{2} \approx \frac{\sqrt{N}\pi}{4}$. This proves that Grover's algorithm can solve the unstructured search problem with $\mathcal{O}(\sqrt{N})$ queries to U_f.

Another derivation of the Grover method is to focus on the operator, instead of the initial vector $|\psi_0\rangle$ at each step of the calculation. In the orthonormal basis $\mathcal{B} = \{|x_0\rangle, |\psi_0^\perp\rangle\}$, the matrix representation of the reflector $R_{x_0} = I - 2|x_0\rangle\langle x_0|$ is

$$[R_{x_0}]_{\mathcal{B}} = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{2.21}$$

The matrix representation for the Grover diffusion operator $R_{\psi_0} = 2|\psi_0\rangle\langle\psi_0| - I$ is

$$[R_{\psi_0}]_{\mathcal{B}} = \begin{pmatrix} 2a^2 - 1 & 2a\sqrt{1-a^2} \\ 2a\sqrt{1-a^2} & 1 - 2a^2 \end{pmatrix} = \begin{pmatrix} -\cos\theta & \sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}. \tag{2.22}$$

Here $\sin(\theta/2) = a = 1/\sqrt{N}$. Therefore for the matrix representation of the Grover iterate $G = R_{\psi_0} R_{x_0}$ is

$$[G]_{\mathcal{B}} = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}, \tag{2.23}$$

i.e., G is a rotation matrix restricted to the two-dimensional space $\mathcal{H} = \operatorname{span}\mathcal{B}$.

The initial vector satisfies

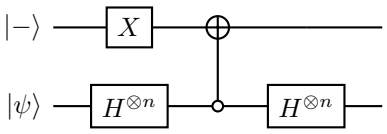
$$[|\psi_0\rangle]_{\mathcal{B}} = \begin{pmatrix} \sin(\theta/2) \\ \cos(\theta/2) \end{pmatrix}, \tag{2.24}$$

so Grover's search can be applied as before, via G^k for $k \approx \frac{\pi\sqrt{N}}{4}$ times.

To draw the quantum circuit of Grover's algorithm, we need an implementation of R_{ψ_0}. Note that

$$R_{\psi_0} = H^{\otimes n}(2|0^n\rangle\langle 0^n| - I)H^{\otimes n}. \tag{2.25}$$

This can be implemented via the following circuit using one ancilla qubit:



Here the controlled-NOT gate is an n-qubit controlled-X gate, and is only active if the system qubits are in the 0^n state. Discarding the signal qubit, we obtain and implementation of R_{ψ_0}. Since the signal qubit $|-\rangle$ only changes up to a sign, it can be reused for both R_{ψ_0} and R_{x_0}.

The reflector R_{ψ_0} can also be implemented without using the ancilla qubit as (use a 3-qubit system as an example)

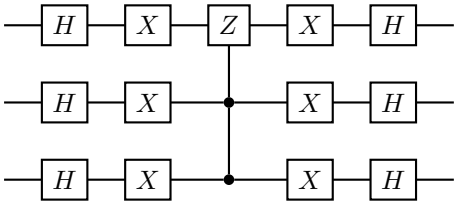


FIGURE 2.3. Implementing R_{ψ_0} for a three qubit system.

Remark 2.3 (Multiple marked states)**.** The Grover search algorithm can be naturally generalized to the case when there are $M > 1$ marked states. The query complexity is $\mathcal{O}(\sqrt{N/M})$. ◇

Example 2.4 (Qiskit for Grover's algorithm)**.** `https://qiskit.org/textbook/ch-algorithms/grover.html` ◇

2.3. Amplitude amplification

Grover's algorithm is not restricted to the problem of unstructured search. One immediate application is called amplitude amplification (AA), which is used ubiquitously as a subroutine to achieve quadratic speedups.

Let $|\psi_0\rangle$ be prepared by an oracle U_{ψ_0}, i.e., $U_{\psi_0}|0^n\rangle = |\psi_0\rangle$. We have the knowledge that

$$|\psi_0\rangle = \sqrt{p_0}\,|\psi_{\text{good}}\rangle + \sqrt{1-p_0}\,|\psi_{\text{bad}}\rangle\,, \tag{2.26}$$

and $p_0 \ll 1$. Here $|\psi_{\text{bad}}\rangle$ is an orthogonal state to the desired state $|\psi_{\text{good}}\rangle$. We cannot get access to $|\psi_{\text{good}}\rangle$ directly, but would like to obtain a state that has a large overlap with $|\psi_{\text{good}}\rangle$, i.e., amplify the amplitude of $|\psi_{\text{good}}\rangle$.

In the problem of unstructured search, $|\psi_{\text{good}}\rangle = |x_0\rangle$, and $p_0 = 1/N$. Although we do not have access to the answer $|x_0\rangle$, we assume access to the reflection oracle R_{x_0}. Here we also assume access to the reflection oracle

$$R_{\text{good}} = 1 - 2\,|\psi_{\text{good}}\rangle\langle\psi_{\text{good}}|\,. \tag{2.27}$$

From U_{ψ_0}, we can construct the reflection with respect to the initial state

$$R_{\psi_0} = 2\,|\psi_0\rangle\langle\psi_0| - I = U_{\psi_0}(2\,|0^n\rangle\langle 0^n| - I)U_{\psi_0}^\dagger \tag{2.28}$$

via the n-qubit controlled-X gate. So following exactly the same procedure as the unstructured search problem, we can construct the Grover iterate

$$G = R_{\psi_0}R_{\text{good}}. \tag{2.29}$$

Applying G^k to $|\psi_0\rangle$ for some $k = \mathcal{O}(1/\sqrt{p_0})$, we obtain a state that has $\Omega(1)$ overlap with $|\psi_{\text{good}}\rangle$.

Example 2.5 (Reflection with respect to signal qubits)**.** One common scenario is that the implementation of U_{ψ_0} requires m ancilla qubits (also called signal qubits), i.e.,

$$U_{\psi_0}|0^m\rangle|0^n\rangle = \sqrt{p_0}\,|0^m\rangle|\psi_0\rangle + \sqrt{1-p_0}\,|\perp\rangle\,, \tag{2.30}$$

where $|\perp\rangle$ is some orthogonal state satisfying

$$(\Pi \otimes I_n)|\perp\rangle = 0, \quad \Pi = |0^m\rangle\langle 0^m|\,. \tag{2.31}$$

Therefore

$$|\psi_{\text{good}}\rangle = |0^m\rangle|\psi_0\rangle\,, \quad |\psi_{\text{bad}}\rangle = |\perp\rangle\,. \tag{2.32}$$

This setting is special since the "good" state can be verified by measuring the ancilla qubits after applying U_{ψ_0} in Eq. (2.30), and post-select the outcome 0^m. In particular, the expected number of measurements needed to obtain $|\psi_{\text{good}}\rangle$ is $1/p_0$.

In order to employ the AA procedure, we first note that the reflection operator can be simplified as

$$R_{\text{good}} = (1 - 2\,|0^m\rangle\langle 0^m|) \otimes I_n = (1 - 2\Pi) \otimes I_n. \tag{2.33}$$

This is because $|\psi_{\text{good}}\rangle$ can be entirely identified by measuring the ancilla qubits. Meanwhile

$$R_{\psi_0} = U_{\psi_0}(2\,|0^{m+n}\rangle\langle 0^{m+n}| - I)U_{\psi_0}^\dagger. \tag{2.34}$$

Let $G = R_{\psi_0}R_{\text{good}}$, and applying G^k to $|\psi_0\rangle$ for some $k = \mathcal{O}(1/\sqrt{p_0})$ times, we obtain a state that has $\Omega(1)$ overlap with $|\psi_{\text{good}}\rangle$. This achieves the desired quadratic speedup. ◇

Example 2.6 (Amplitude damping)**.** Assuming access to an oracle in Eq. (2.30), where p_0 is large, we can easily dampen the amplitude to any number $\alpha \le \sqrt{p_0}$.

We introduce an additional signal qubit. Then Eq. (2.30) becomes

$$(I \otimes U_{\psi_0})\,|0\rangle\,|0\rangle\,|0^n\rangle = |0\rangle\sqrt{p_0}\,|0\rangle\,|\psi_0\rangle + \sqrt{1-p_0}\,|\perp\rangle\,. \tag{2.35}$$

Define a single qubit rotation operation as

$$R_\theta\,|0\rangle = \cos\theta\,|0\rangle + \sin\theta\,|1\rangle\,, \tag{2.36}$$

and we have

$$\begin{aligned}
&(R_\theta \otimes I_{m+n})(I \otimes U_{\psi_0})\,|0\rangle\,|0^m\rangle\,|0^n\rangle \\
=&\cos\theta\,|0\rangle\,(\sqrt{p_0}\,|0^m\rangle\,|\psi_0\rangle + \sqrt{1-p_0}\,|\perp'\rangle) + \sin\theta\,|1\rangle\,(\sqrt{p_0}\,|0^m\rangle\,|\psi_0\rangle + \sqrt{1-p_0}\,|\perp'\rangle) \\
:=&\sqrt{p_0}\cos\theta\,|0\rangle\,|0^m\rangle\,|\psi_0\rangle + \sqrt{1-p_0\cos^2\theta}\,|\perp'\rangle\,.
\end{aligned} \tag{2.37}$$

Here $(|0^{m+1}\rangle\langle 0^{m+1}| \otimes I_n)\,|\perp'\rangle = 0$. We only need to choose $\sqrt{p_0}\cos\theta = \alpha$.

◇

2.4. Lower bound of query complexity*

Recall that the unstructured search problem tries to find a marked state $x_0 \in [N]$, using a reflection oracle

$$R_{x_0} = I - 2\,|x_0\rangle\langle x_0|\,. \tag{2.38}$$

Grover's algorithm can find x_0 with constant probability (e.g. at least 1/2) by making $\mathcal{O}(\sqrt{N})$ times querying R_{x_0}. It turns out that this is *asymptotically optimal*, i.e., no quantum algorithm can perform this task using fewer than $\Omega(\sqrt{N})$ access to R_{x_0}.

Any quantum search algorithm that starts from a universal initial state $|\psi_0\rangle$ and queries R_{x_0} for k steps can be written in the following form:

$$|\psi_k^{x_0}\rangle = \mathcal{U}_k^{x_0}\,|\psi_0\rangle = U_k R_{x_0}\cdots U_2 R_{x_0} U_1 R_{x_0}\,|\psi_0\rangle\,, \tag{2.39}$$

for some unitaries $U_1, \ldots, U_k$. For simplicity we assume no ancilla qubits are used, and the proof can be generalized to the case in the presence of ancilla qubits. The superscript x_0 indicates that the state depends on the marked state x_0. Specifically, by "solving" the search problem, it means that there exists a k so that for each marked state x_0,

$$|\,\langle\psi_k^{x_0}|x_0\rangle\,|^2 \ge \frac{1}{2}. \tag{2.40}$$

In other words, measuring $|\psi_k^{x_0}\rangle$ in the computational basis, the probability of obtaining $|x_0\rangle$ is at least 1/2.

To prove the lower bound, we compare the action of $\mathcal{U}_k^{x_0}$ with another "fake algorithm" $\mathcal{U}_k$, defined as

$$|\psi_k\rangle = \mathcal{U}_k\,|\psi_0\rangle = U_k\cdots U_2 U_1\,|\psi_0\rangle\,. \tag{2.41}$$

In particular, $|\psi_k\rangle$ does not involve any information of the solution x_0 and therefore cannot possibly solve the search problem.

For a set of vectors $\{f^{x_0}\}_{x_0\in[N]}$, and each $f^{x_0}\in\mathbb{C}^N$, we will extensively use the following discrete ℓ_2-norm:

$$\|f\|_{\ell_2} := \left(\sum_{x_0\in[N]} \|f^{x_0}\|\right)^{\frac{1}{2}}. \tag{2.42}$$

In particular, we have the following triangle inequality

$$\|f\|_{\ell_2} - \|g\|_{\ell_2} \le \|f+g\|_{\ell_2} \le \|f\|_{\ell_2} + \|g\|_{\ell_2}. \tag{2.43}$$

The proof contains two steps. First, we show that the true solution and the fake solution differs significantly, in the sense that

$$D_k := \sum_{x_0\in[N]} \||\psi_k^{x_0}\rangle - |\psi_k\rangle\|^2 = \Omega(N). \tag{2.44}$$

Second, we prove that (define $D_0 = 0$):

$$D_k \le 4k^2, \quad k \ge 0. \tag{2.45}$$

Therefore to satisfy Eq. (2.44), we must have $k = \Omega(\sqrt{N})$.

In the first step, since multiplying a phase factor $e^{\mathrm{i}\theta}$ to $|\psi_k^{x_0}\rangle$ does not have any physical consequences, we may choose a particular phase θ so that Eq. (2.40) becomes

$$\langle\psi_k^{x_0}|x_0\rangle \ge \frac{1}{\sqrt{2}}. \tag{2.46}$$

Therefore

$$\||\psi_k^{x_0}\rangle - |x_0\rangle\|^2 = 2 - 2\langle\psi_k^{x_0}|x_0\rangle \le 2-\sqrt{2}. \tag{2.47}$$

This means that

$$\sum_{x_0\in[N]} \||\psi_k^{x_0}\rangle - |x_0\rangle\|^2 \le 2N - \sqrt{2}N. \tag{2.48}$$

On the other hand, for the "fake algorithm", using the Cauchy-Schwarz inequality,

$$\sum_{x_0\in[N]} \||\psi_k\rangle - |x_0\rangle\|^2 \ge 2N - 2\sum_{x_0\in[N]} |\langle x_0|\psi\rangle| \ge 2N - 2\sqrt{N}\sum_{x_0\in[N]} |\langle x_0|\psi\rangle|^2 = 2N - 2\sqrt{N}. \tag{2.49}$$

This violates the bound in Eq. (2.48), and the fake algorithm cannot solve the search problem for arbitrarily large k.

So from Eqs. (2.48) and (2.49), and the triangle inequality, we have

$$\begin{aligned} D_k = \sum_{x_0\in[N]} \| |\psi_k^{x_0}\rangle - |\psi_k\rangle \|^2 &= \sum_{x_0\in[N]} \| (|\psi_k^{x_0}\rangle - |x_0\rangle) - (|\psi_k\rangle - |x_0\rangle) \|^2 \\ &\geq \left(\sqrt{\sum_{x_0\in[N]} \| |\psi_k\rangle - |x_0\rangle \|^2} - \sqrt{\sum_{x_0\in[N]} \| |\psi_k^{x_0}\rangle - |x_0\rangle \|^2} \right)^2 \\ &\geq \left(\sqrt{2N - 2\sqrt{N}} - \sqrt{2N - \sqrt{2}N} \right)^2 \\ &= \Omega(N). \end{aligned} \tag{2.50}$$

This proves Eq. (2.44). In other words, the true solution and the fake solution must be well separated in ℓ_2-norm.

In the second step, we prove Eq. (2.45) inductively. Clearly Eq. (2.45) is true for $k = 0$. Assume this is true, then

$$\begin{aligned} D_{k+1} &= \sum_{x_0\in[N]} \left\| |\psi_{k+1}^{x_0}\rangle - |\psi_{k+1}\rangle \right\|^2 \\ &= \sum_{x_0\in[N]} \| U_{k+1} R_{x_0} |\psi_k^{x_0}\rangle - U_{k+1} |\psi_k\rangle \|^2 \\ &= \sum_{x_0\in[N]} \| R_{x_0} |\psi_k^{x_0}\rangle - |\psi_k\rangle \|^2 \\ &= \sum_{x_0\in[N]} \| R_{x_0}(|\psi_k^{x_0}\rangle - |\psi_k\rangle) + (R_{x_0} - I) |\psi_k\rangle \|^2 \\ &\leq \left(\sqrt{\sum_{x_0\in[N]} \| R_{x_0}(|\psi_k^{x_0}\rangle - |\psi_k\rangle) \|^2} + \sqrt{\sum_{x_0\in[N]} \| (R_{x_0} - I) |\psi_k\rangle \|^2} \right)^2. \end{aligned} \tag{2.51}$$

The last inequality uses the triangle inequality of the discrete ℓ_2-norm. Note that

$$\sqrt{\sum_{x_0\in[N]} \| (R_{x_0} - I) |\psi_k\rangle \|^2} = \sqrt{4 \sum_{x_0\in[N]} |\langle x_0|\psi_k\rangle|^2} = 2, \tag{2.52}$$

and

$$\sqrt{\sum_{x_0\in[N]} \| R_{x_0}(|\psi_k^{x_0}\rangle - |\psi_k\rangle) \|^2} = \sqrt{\sum_{x_0\in[N]} \| |\psi_k^{x_0}\rangle - |\psi_k\rangle \|^2} = \sqrt{D_k}, \tag{2.53}$$

we have

$$\sqrt{D_{k+1}} \leq \sqrt{D_k} + 2 \leq 2(k+1), \tag{2.54}$$

which finishes the induction.

Finally, combining the lower bound Eqs. (2.44) and (2.45), we find that the necessary condition to solve the unstructured search problem is $4k^2 = \Omega(N)$, or $k = \Omega(\sqrt{N})$.

Remark 2.7 (Implication for amplitude amplification)**.** Due to the close relation between unstructred search and amplitude amplification, it means that given a state $|\psi\rangle$ of which the amplitude of

the "good" component is $\alpha \ll 1$, no quantum algorithms can amplify the amplitude to $\Omega(1)$ using $o(\alpha^{-\frac{1}{2}})$ queries to the reflection operators. ◇

Exercise 2.1. In Deutsch's algorithm, demonstrate why not assuming access to an oracle $V_f : |x\rangle \mapsto |f(x)\rangle$.

Exercise 2.2. For all possible mappings $f : \{0,1\} \to \{0,1\}$, draw the corresponding quantum circuit to implement $U_f : |x,0\rangle \mapsto |x, f(x)\rangle$.

Exercise 2.3. Prove Eq. (2.20).

Exercise 2.4. Draw the quantum circuit for Eq. (2.28).

Exercise 2.5. Prove that when ancilla qubits are used, the complexity of the unstructured search problem is still $\Omega(\sqrt{N})$.

CHAPTER 3

Quantum phase estimation

The setup of the phase estimation problem is as follows. Let U be a unitary, and $|\psi\rangle$ is an eigenvector, i.e.,

$$U|\psi\rangle = e^{\mathrm{i}2\pi\varphi}|\psi\rangle, \quad \varphi \in [0,1). \tag{3.1}$$

The goal is to find φ up to certain precision. This is a quantum primitive with numerous applications: prime factorization (Shor's algorithm), linear system (HHL), eigenvalue problem, amplitude estimation, quantum counting, quantum walk, etc.

Using a classical computer, we can estimate φ using $U|\psi\rangle \oslash |\psi\rangle$, where $\oslash$ stands for the element-wise division operation. Specifically, if $|\psi\rangle$ is indeed an eigenvector and $\langle j|\psi\rangle \neq 0$ for any j in the computational basis, then we can extract the phase from

$$\langle j|U|\psi\rangle \,/\, \langle j|\psi\rangle = e^{\mathrm{i}2\pi\varphi}. \tag{3.2}$$

Unfortunately, such a element-wise division operation cannot be efficiently implemented on quantum computers, and alternative methods are needed.

Quantum phase estimation has numerous variants, and still receives intensive research attention till today. This chapter only introduces some of the simplest variants.

3.1. Hadamard test

We first introduce the Hadamard test, which is a useful tool for computing the expectation value of an unitary operator with respect to a state, i.e., $\langle\psi|U|\psi\rangle$. Since U is generally not Hermitian, this does not correspond to the measurement of a physical observable. Instead the real and imaginary part of the expectation value need to be measured separately.

The (real) Hadamard test is the quantum circuit in Fig. 3.1 for estimating the real part of $\langle\psi|U|\psi\rangle$.

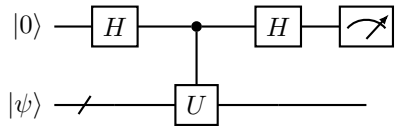


FIGURE 3.1. Hadamard test for $\mathrm{Re}\,\langle\psi|U|\psi\rangle$.

To verify this, we find that the circuit transforms $|0\rangle|\psi\rangle$ as

$$\begin{aligned}|0\rangle|\psi\rangle &\xrightarrow{\mathrm{H}\otimes I} \frac{1}{\sqrt{2}}(|0\rangle+|1\rangle)|\psi\rangle \\ &\xrightarrow{c\text{-}U} \frac{1}{\sqrt{2}}(|0\rangle|\psi\rangle+|1\rangle U|\psi\rangle) \\ &\xrightarrow{\mathrm{H}\otimes I} \frac{1}{2}|0\rangle(|\psi\rangle+U|\psi\rangle)+\frac{1}{2}|1\rangle(|\psi\rangle-U|\psi\rangle).\end{aligned}$$

The probability of measuring the qubit 0 to be in state $|0\rangle$ is

$$p(0)=\frac{1}{2}(1+\operatorname{Re}\langle\psi|U|\psi\rangle). \tag{3.3}$$

This is well defined since $-1\le \operatorname{Re}\langle\psi|U|\psi\rangle\le 1$.

To obtain the imaginary part, we can use the circuit in Fig. 3.2 called the (imaginary) Hadamard test, where

$$S=\begin{pmatrix}1 & 0\\ 0 & \mathrm{i}\end{pmatrix} \tag{3.4}$$

is called the phase gate.

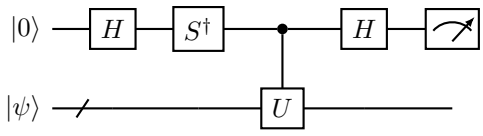


FIGURE 3.2. Hadamard test for $\operatorname{Im}\langle\psi|U|\psi\rangle$.

Similar calculation shows the circuit transforms $|0\rangle|\psi\rangle$ to the state

$$\frac{1}{2}|0\rangle(|\psi\rangle-\mathrm{i}U|\psi\rangle)+\frac{1}{2}|1\rangle(|\psi\rangle+\mathrm{i}U|\psi\rangle). \tag{3.5}$$

Therefore the probability of measuring the qubit 0 to be in state $|0\rangle$ is

$$p(0)=\frac{1}{2}(1+\operatorname{Im}\langle\psi|U|\psi\rangle). \tag{3.6}$$

Combining the results from the two circuits, we obtain the estimate to $\langle\psi|U|\psi\rangle$.

Example 3.1 (Overlap estimate using swap test)**.** An application of the Hadamard test is called the swap test, which is used to estimate the overlap of two quantum states $|\langle\varphi|\psi\rangle|$. The quantum circuit for the swap test is

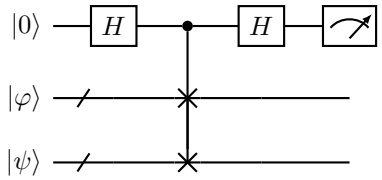


FIGURE 3.3. Circuit for the SWAP test.

Note that this is exactly the Hadamard test with U being the n-qubit swap gate. Direct calculation shows that the probability of measuring the qubit 0 to be in state $|0\rangle$ is

$$p(0) = \frac{1}{2}(1 + \mathrm{Re}\,\langle\varphi, \psi|\psi, \varphi\rangle) = \frac{1}{2}(1 + |\langle\varphi|\psi\rangle|^2). \tag{3.7}$$

◇

Example 3.2 (Overlap estimate with relative phase information)**.** In the swap test, the quantum states $|\varphi\rangle, |\psi\rangle$ can be black-box states, and in such a scenario obtaining an estimate to $|\langle\varphi|\psi\rangle|$ is the best one can do. In order to retrieve the relative phase information and to obtain $\langle\varphi|\psi\rangle$, we need to have access to the unitary circuit preparing $|\varphi\rangle, |\psi\rangle$, i.e.,

$$U_\varphi |0^n\rangle = |\varphi\rangle, \quad U_\psi |0^n\rangle = |\psi\rangle. \tag{3.8}$$

Then we have $\langle\varphi|\psi\rangle = \langle 0^n|U_\varphi^\dagger U_\psi|0^n\rangle$. ◇

Example 3.3 (Single qubit phase estimation)**.** The Hadamard test can also be used to derive the simplest version of the phase estimate based on success probabilities. Apply the Hadamard test in Fig. 3.1 with U, ψ satisfying Eq. (3.1). Then the probability of measuring the qubit 0 to be in state $|1\rangle$ is

$$p(1) = \frac{1}{2}(1 - \mathrm{Re}\,\langle\psi|U|\psi\rangle) = \frac{1}{2}(1 - \cos(2\pi\varphi)). \tag{3.9}$$

Therefore

$$\varphi = \pm\frac{\arccos(1 - 2p(1))}{2\pi} \pmod 1. \tag{3.10}$$

In order to quantify the efficiency of the procedure, recall from Example 1.11 that if $p(1)$ is far away from 0 or 1, i.e., $(2\varphi \mod 1)$ is far away from 0, in order to approximate $p(1)$ (and hence φ) to additive precision ϵ, the number of samples needed is $\mathcal{O}(1/\epsilon^2)$.

Now assume φ is very close to 0 and we would like to estimate φ to additive precision ϵ. Note that

$$p(1) \approx (2\pi\varphi)^2 = \mathcal{O}(\epsilon^2). \tag{3.11}$$

Then $p(1)$ needs to be estimated to precision $\mathcal{O}(\epsilon^2)$, and again the number of samples needed is $\mathcal{O}(1/\epsilon^2)$. The case when φ is close to $1/2$ or 1 is similar.

Note that the circuit in Fig. 3.1 cannot distinguish the sign of φ (or whether $\varphi \geq 1/2$ when restricted to the interval $[0, 1)$). To this end we need Fig. 3.2, but replace $\mathrm{S}^\dagger$ by S, so that the success probability of measuring 1 in the computational basis is

$$p(1) = \frac{1}{2}(1 + \sin(2\pi\varphi)). \tag{3.12}$$

This gives

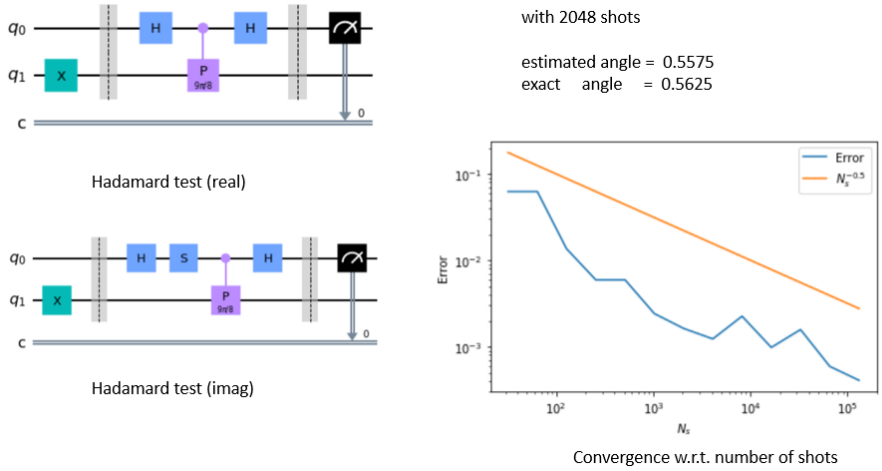
$$\varphi \begin{cases} \in [0, 1/2), & p(1) \geq \frac{1}{2}, \\ \in (1/2, 1), & p(1) < \frac{1}{2}. \end{cases} \tag{3.13}$$

Unlike the previous estimate, in order to correctly estimate the sign, we only require $\mathcal{O}(1)$ accuracy, and run Fig. 3.2 for a constant number of times (unless φ is very close to 0 or π). ◇

Example 3.4 (Qiskit example for phase estimation using the Hadamard test)**.** Here is a qiskit example of the simple phase estimation for

$$R|1\rangle = \begin{pmatrix} 1 & 0 \\ 0 & e^{i2\pi\varphi} \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = e^{\mathrm{i}2\pi\varphi}|1\rangle,$$

with $\varphi = 0.5 + \frac{1}{2^d} \equiv 0.10\cdots01$ (d bits in total). In this example, $d = 4$ and the exact value is $\varphi = 0.5625$.



◇

3.2. Quantum phase estimation (Kitaev's method)*

In Example 3.3, the number of measurements needed to estimate φ to precision ϵ is $\mathcal{O}(1/\epsilon^2)$. A quadratic improvement in precision (i.e., $\mathcal{O}(1/\epsilon)$) can be achieved by means of the quantum phase estimation. One such procedure is called Kitaev's method [KSV02, Section 13.5].

In the fixed point representation in Section 1.8, for simplicity we assume that the eigenvalue can be exactly represented using d bits, i.e.,

$$\varphi = (.\varphi_{d-1}\cdots\varphi_0). \tag{3.14}$$

In the simplest scenario, we assume $d = 1$, and $\varphi = .\varphi_0, \varphi_0 \in \{0, 1\}$. Then $e^{\mathrm{i}2\pi\varphi} = e^{\mathrm{i}\pi\varphi_0}$.

Performing the real Hadamard test in Example 3.3, we have $p(1) = 0$ if $\varphi_0 = 0$, and $p(1) = 1$ if $\varphi_0 = 1$. In either case, the result is *deterministic*, and one measurement is sufficient to determine the value of φ_0.

Next, consider $\varphi = .\underbrace{0\cdots 0}_{d \text{ bits}}\varphi_0$. To determine the value of φ_0, we need to reach precision $\epsilon < 2^{-d}$. The method in Example 3.3 requires $\mathcal{O}(1/\epsilon^2) = \mathcal{O}(2^{2d})$ repeated measurements, or number of queries to U. The observation from Kitaev's method is that if we can have access to U^j for a suitable power j, then the number of queries to U can be reduced. More specifically, if we can query $U^{2^{d-1}}$, then the circuit in Fig. 3.4 with $j = d-1$ gives

$$p(1) = \frac{1}{2}(1 - \cos(2\pi.\varphi_0)) = \begin{cases} 0, & \varphi_0 = 0, \\ 1, & \varphi_0 = 1. \end{cases} \tag{3.15}$$

The result is again deterministic. Therefore the total number of queries to U becomes $\mathcal{O}(2^{-d}) = \mathcal{O}(\epsilon^{-1})$.

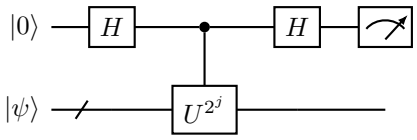


FIGURE 3.4. Circuit used in Kitaev's method. Another one with a phase gate to determine the sign of $2^j\varphi$ may also be used.

This is the basic idea behind Kitaev's method: use a more complex quantum circuit (and in particular, with a larger circuit depth) to reduce the *total* number of queries. As a general strategy, instead of estimating φ from a single number, we assume access to U^{2^j}, and estimate φ bit-by-bit. In particular, changing $U \to U^{2^j}$ in the Hadamard test allows us to estimate

$$2^j\varphi = \varphi_{d-1}\cdots\varphi_{d-j}.\varphi_{d-j-1}\cdots\varphi_0 = .\varphi_{d-j-1}\cdots\varphi_0 \pmod 1 \tag{3.16}$$

One immediate difficulty of the bit-by-bit estimation is that we need to tell $0.0111\ldots$ apart from $0.1000\ldots$, and the two numbers can be arbitrarily close to each other (though the two numbers can also differ at some number of digits), and some careful work is needed. We will first describe the algorithm, and then analyze its performance. The algorithm works for any φ, and then the goal is to estimate its d bits. For simplicity of the analysis, we assume φ is exactly represented by d bits. We will use extensively the distance

$$|x|_1 \equiv |x|_{\bmod 1} := \min\{(x \bmod 1), 1 - (x \bmod 1)\}, \tag{3.17}$$

which is the distance on the unit circle.

First, by applying the circuit in Fig. 3.4 (and the corresponding circuit to determine the sign) with $j = 0, 1, \ldots, d-3$, for each j we can estimate $p(0)$, so that the error in $2^j\varphi$ is less than $1/16$ for all j (this can happen with a sufficiently high success probability. For simplicity, let us assume that this happens with certainty). The measured result is denoted by α_j. This means that any perturbation must be due to the 5th digit in the binary representation. For example, if $2^j\varphi = 0.11100$, then $\alpha_j = 0.11011$ is an acceptable result with an error $0.00001 = 1/32$, but $\alpha_j = 0.11110$ is not acceptable since the error is $0.0001 = 1/16$. We then *round* $\alpha_j \pmod 1$ by its closest 3-bit estimate denoted by β_j, i.e., β_j is taken from the set $\{\,0.000, 0.001, 0.010, 0.011, 0.100, 0.101, 0.110, 0.111\,\}$. Consider the example of $2^j\varphi = 0.11110$, if $\alpha_j = 0.11101$, then $\beta_j = 0.111$. But if $\alpha_j = 0.11111$, then $\beta_j = 0.000$. Another example is $2^j\varphi = 0.11101$, if $\alpha_j = 0.11110$, then both $\beta_j = 0.111$ (rounded

down) and $\beta_j = 0.000$ (rounded up) are acceptable. We can pick one of them at random. We will show later that the uncertainty in α_j, β_j is not detrimental to the success of the algorithm.

Second, we perform some post-processing. Start from $j = d-3$, we can estimate $.\varphi_2\varphi_1\varphi_0$ to accuracy $1/16$, which recovers these three bits exactly. The values of these three bits will be taken from β_{d-3} directly. Then we proceed with the iteration: for $j = d-4, \ldots, 0$, we assign

$$\varphi_{d-j-1} = \begin{cases} 0, & |.0\varphi_{d-j-2}\varphi_{d-j-3} - \beta_j|_{\bmod 1} < 1/4, \\ 1, & |.1\varphi_{d-j-2}\varphi_{d-j-3} - \beta_j|_{\bmod 1} < 1/4. \end{cases} \tag{3.18}$$

Here $|\cdot|_{\bmod 1}$ is the periodic distance on $[0,1)$ and its value is always $\le 1/2$. Since the two possibilities are separated by $1/2$, for each j, there will be at most one case that is satisfied. We will also show that in all circumstances, there is always one case that is satisfied, regardless of the ambiguity of the choice of β_j above.

After running the algorithm above, we recover $\varphi = .\varphi_{d-1}\cdots\varphi_0$ exactly. The total cost of Kitaev's method measured by the number of queries to U is $\mathcal{O}\left(\sum_{j=0}^{d-3} 2^j\right) = \mathcal{O}(\epsilon^{-1})$.

If φ is exactly represented by d bits, we will obtain an estimate

$$|.\varphi_{d-1}\cdots\varphi_0 - \varphi|_{\bmod 1} < 2^{-d} = \epsilon. \tag{3.19}$$

Example 3.5. Consider $\varphi = 0.\varphi_4\varphi_3\varphi_2\varphi_1\varphi_0 = 0.11111$ and $d = 5$. Running Kitaev's algorithm with $j = 0, 1, 2$ gives the following possible choices of β_j:

j	$2^j\varphi$	possible β_j
0	0.11111	$\{0.111, 0.000\}$
1	0.1111	$\{0.111, 0.000\}$
2	0.111	$\{0.111\}$

Start with $j = 2$. We have only one choice of β_j, and can recover $0.\varphi_2\varphi_1\varphi_0 = 0.111$. Then for $j = 1$, we need to use Eq. (3.18) to decide φ_3. If we choose $\beta_j = 0.111$, we have $\varphi_3 = 1$. But if we choose $\beta_j = 0.000$, we still need to choose $\varphi_3 = 1$, since $|.011 - 0.000|_{\bmod 1} = 0.100 = 1/2 > 1/4$, and $|.111 - 0.000|_{\bmod 1} = 0.001 = 1/8 < 1/4$. Similar analysis shows that for $j = 0$ we have $\varphi_4 = 1$. This recovers φ exactly. ◇

Example 3.6 (A variant of Kitaev's algorithm that does not work)**.** Let us modify Kitaev's algorithm as follows: for each $2^j\varphi$ is determined to precision $1/8$, and round the result to $\beta_j \in \{0.00, 0.01, 0.10, 0.11\}$. Start from $j = d-2$, we estimate $.\varphi_1\varphi_0$ exactly. Then for $j = d-3, \ldots, 0$, we assign

$$\varphi_{d-j-1} = \begin{cases} 0, & |.0\varphi_{d-j-2} - \beta_j|_{\bmod 1} < 1/2, \\ 1, & |.1\varphi_{d-j-2} - \beta_j|_{\bmod 1} < 1/2. \end{cases} \tag{3.20}$$

Now that the inequality $< 1/2$ above can be equivalently written as $\le 1/4$.

Let us run the algorithm above for $\varphi = 0.\varphi_4\varphi_3\varphi_2\varphi_1\varphi_0 = 0.1111$ and $d = 4$. This gives:

j	$2^j\varphi$	possible β_j
0	0.1111	$\{0.11, 0.00\}$
1	0.111	$\{0.11, 0.00\}$
2	0.11	$\{0.11\}$

Start with $j = 2$. We have only one choice of β_j, and can recover $0.\varphi_1\varphi_0 = 0.11$. Then for $j = 1$, if we choose $\beta_j = 0.11$, we have $\varphi_2 = 1$. But if we choose $\beta_j = 0.00$, then $|.01 - 0.00|_{\bmod 1} = 0.0.01 = 1/4$,

and $|.11 - 0.000|_{\mod 1} = 0.01 = 1/4$. So the algorithm cannot distinguish the two possibilities and fails. ◇

Let us now inductively show why Kitaev's algorithm works. Again assume φ is exactly represented by d bits. For $j = d-3$, we know that $\varphi_2\varphi_1\varphi_0$ can be recovered exactly. Then assume $\varphi_{d-j-2}\cdots\varphi_0$ have all been exactly computed, at step j we would like to determine the value of φ_{d-j-1}. From

$$\left|\alpha_j - 2^j\varphi\right|_{\mod 1} < 1/16, \quad |\alpha_j - \beta_j|_{\mod 1} \leq 1/16, \tag{3.21}$$

we know

$$\left|2^j\varphi - \beta_j\right|_{\mod 1} < 1/8. \tag{3.22}$$

Then

$$\begin{aligned} &|.\varphi_{d-j-1}\varphi_{d-j-2}\varphi_{d-j-3} - \beta_j|_{\mod 1} \\ \leq &\left|.\varphi_{d-j-1}\varphi_{d-j-2}\varphi_{d-j-3} - 2^j\varphi\right|_{\mod 1} + \left|2^j\varphi - \beta_j\right|_{\mod 1} \\ \leq &1/16 + 1/8 < 1/4. \end{aligned} \tag{3.23}$$

The wrong choice of φ_{d-j-1} denoted by $\widetilde{\varphi}_{d-j-1}$ then satisfies

$$\begin{aligned} &|.\widetilde{\varphi}_{d-j-1}\varphi_{d-j-2}\varphi_{d-j-3} - \beta_j|_{\mod 1} \\ \geq &\left|.\widetilde{\varphi}_{d-j-1}\varphi_{d-j-2}\varphi_{d-j-3} - .\varphi_{d-j-1}\varphi_{d-j-2}\varphi_{d-j-3}\right|_{\mod 1} - |.\varphi_{d-j-1}\varphi_{d-j-2}\varphi_{d-j-3} - \beta_j|_{\mod 1} \\ >&1/2 - 1/4 = 1/4. \end{aligned} \tag{3.24}$$

This proves the validity of Eq. (3.18), and hence that of Kitaev's algorithm.

3.3. Quantum Fourier transform

Fourier transform is used ubiquitously in scientific computing, and fast Fourier transform (FFT) is the backbone for many fast algorithms in classical computation. Similarly, the quantum Fourier transform is also an important component in many quantum algorithms, such as phase estimation, Shor's algorithm, and inspires other fast algorithms such as fast Fermionic Fourier transform (FFFT) etc.

For any j in the computational basis, the (discrete) forward Fourier transform is defined as

$$U_{\mathrm{FT}}\left|j\right\rangle = \frac{1}{\sqrt{N}}\sum_{k\in[N]} e^{\mathrm{i}2\pi\frac{kj}{N}}\left|k\right\rangle. \tag{3.25}$$

In particular

$$U_{\mathrm{FT}}\left|0^n\right\rangle = \frac{1}{\sqrt{N}}\sum_{k\in[N]}\left|k\right\rangle = H^{\otimes n}\left|0^n\right\rangle. \tag{3.26}$$

The (discrete) inverse Fourier transform is

$$U_{\mathrm{FT}}^\dagger\left|j\right\rangle = \frac{1}{\sqrt{N}}\sum_{k\in[N]} e^{-\mathrm{i}2\pi\frac{kj}{N}}\left|k\right\rangle. \tag{3.27}$$

Using the binary representation of integers

$$k = (k_{n-1}\cdots k_0.), \quad j = (j_{n-1}\cdots j_0.) \tag{3.28}$$

we have

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

Therefore the exponential can be written as

$$e^{\mathrm{i}2\pi\frac{kj}{N}} = e^{\mathrm{i}2\pi k_0(.j_{n-1}\cdots j_0)}e^{\mathrm{i}2\pi k_1(.j_{n-2}\cdots j_0)}\cdots e^{\mathrm{i}2\pi k_{n-1}(.j_0)}. \tag{3.29}$$

The most important step of QFT is the following direct calculation, which requires some patience with the manipulation of indices:

(3.30)

$$\begin{aligned}U_{\mathrm{FT}}\left|j_{n-1}\cdots j_0\right\rangle =& \frac{1}{\sqrt{2^n}}\sum_{k_{n-1},\ldots,k_0} e^{\mathrm{i}2\pi k_0(.j_{n-1}\cdots j_0)}e^{\mathrm{i}2\pi k_1(.j_{n-2}\cdots j_0)}\cdots e^{\mathrm{i}2\pi k_{n-1}(.j_0)}\left|k_{n-1}\cdots k_0\right\rangle \\ =& \frac{1}{\sqrt{2^n}}\left(\sum_{k_{n-1}} e^{\mathrm{i}2\pi k_{n-1}(.j_0)}\left|k_{n-1}\right\rangle\right)\otimes\left(\sum_{k_{n-2}} e^{\mathrm{i}2\pi k_{n-2}(.j_1j_0)}\left|k_{n-1}\right\rangle\right) \\ & \otimes\cdots\otimes\left(\sum_{k_0} e^{\mathrm{i}2\pi k_0(.j_{n-1}\cdots j_0)}\left|k_0\right\rangle\right) \\ =& \frac{1}{\sqrt{2^n}}\left(\left|0\right\rangle + e^{\mathrm{i}2\pi(.j_0)}\left|1\right\rangle\right)\otimes\left(\left|0\right\rangle + e^{\mathrm{i}2\pi(.j_1j_0)}\left|1\right\rangle\right)\otimes\cdots\otimes\left(\left|0\right\rangle + e^{\mathrm{i}2\pi(.j_{n-1}\cdots j_0)}\left|1\right\rangle\right).\end{aligned}$$

Eq. (3.30) involves a series of controlled rotations of the form

$$\left|0\right\rangle \to \frac{1}{\sqrt{2}}\left(\left|0\right\rangle + e^{\mathrm{i}2\pi(.j_{n-1}\cdots j_0)}\left|1\right\rangle\right). \tag{3.31}$$

Hence before discussing the quantum circuit for QFT, let us first work out the circuit for implementing this controlled rotation. We use the relation

$$e^{\mathrm{i}2\pi(.j_{n-1}\cdots j_0)} = e^{\mathrm{i}2\pi(.j_{n-1})}e^{\mathrm{i}2\pi(.0j_{n-2})}\cdots e^{\mathrm{i}2\pi(.0\cdots 0j_0)}. \tag{3.32}$$

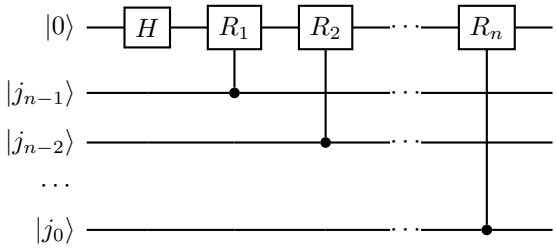
Example 3.7 (Implementation of controlled rotation)**.** Consider the implementation of

$$\left|0\right\rangle\left|j\right\rangle \to \frac{1}{\sqrt{2}}\left(\left|0\right\rangle + e^{\mathrm{i}2\pi(.j_{n-1}\cdots j_0)}\left|1\right\rangle\right)\left|j\right\rangle, \tag{3.33}$$

Let

$$R_z(\theta) = \begin{pmatrix}1 & 0\\ 0 & e^{\mathrm{i}\theta}\end{pmatrix}, \tag{3.34}$$

and $R_j = R_z(\pi/2^{j-1})$. In particular, $R_1 = Z$. The quantum circuit is



◇

The implementation of QFT follows the same principle, but *does not* require the signal qubit to store the phase information. Let us see a few examples.

When $n = 1$, we need to implement

$$|j_0\rangle \to \frac{1}{\sqrt{2}}\left(|0\rangle + e^{\mathrm{i}2\pi(.j_0)}\,|1\rangle\right). \tag{3.35}$$

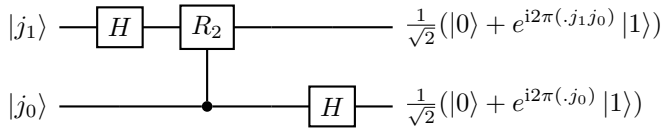
This is the Hadamard gate:

$$|j_0\rangle \to H\,|j_0\rangle\,. \tag{3.36}$$

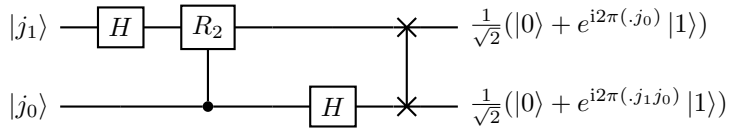
When $n = 2$, we need to implement

$$|j\rangle \to \frac{1}{\sqrt{2^2}}\left(|0\rangle + e^{\mathrm{i}2\pi(.j_0)}\,|1\rangle\right) \otimes \left(|0\rangle + e^{\mathrm{i}2\pi(.j_1 j_0)}\,|1\rangle\right). \tag{3.37}$$

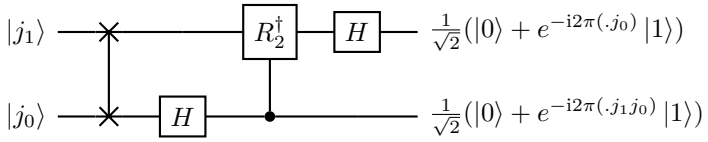
This can be implemented using the following circuit:



Comparing the result with that in Eq. (3.37), we find that the ordering of the qubits is reversed. To recover the desired result in QFT, we can apply a SWAP gate to the outcome, i.e.,



In order to implement the inverse Fourier transform, we only need to apply the Hermitian conjugate as



Similarly one can construct the circuit for U_{FT} and its inverse for $n = 3$.

In general, the QFT circuit is given by Fig. 3.5. Compare the circuit in Fig. 3.5 with Eq. (3.30), we find again that the ordering is reversed in the output. To restore the correct order as defined in QFT, we can use $\mathcal{O}(n/2)$ swaps operations. The total gate complexity of QFT is $\mathcal{O}(n^2)$.

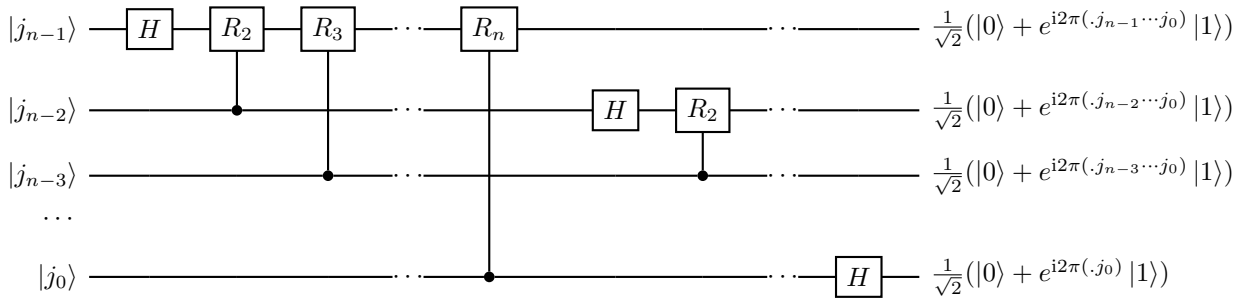


FIGURE 3.5. Quantum circuit for quantum Fourier transform (before applying swap operations).

Example 3.8 (Qiskit example for QFT)**.** https://qiskit.org/textbook/ch-algorithms/quantum-fourier-transform.html ◇

3.4. Quantum phase estimation using quantum Fourier transform

In Kitaev's method, we use 1 ancilla qubit but d *different* circuits of various circuit depths to perform phase estimation. In this section, we introduce the (standard) quantum phase estimation (QPE), which uses one signal quantum circuit based on QFT, but requires d-ancilla qubits to store the phase information in the quantum computer. From now, we assume $\varphi = .\varphi_{d-1}\cdots\varphi_0$ is exact. From the availability of U^j we can define a controlled unitary operation

$$\mathcal{U} = \sum_{j\in[2^d]} |j\rangle\langle j| \otimes U^j. \tag{3.38}$$

When $d = 1$, $\mathcal{U}$ is simply the controlled U operation. For a general d, it seems that we need to implement all 2^d different U^j. However, this is not necessary. Using the binary representation of integers $j = (j_{d-1}\cdots j_0.) = \sum_{i=0}^{d-1} j_i 2^i$, we have $U^j = U^{\sum_{i=0}^{d-1} j_i 2^i} = \prod_{i=0}^{d-1} U^{j_i 2^i}$. Therefore similar to the operations in QFT,

$$\begin{aligned}
\mathcal{U} &= \sum_{j\in[2^d]} |j\rangle\langle j| \otimes U^j \\
&= \sum_{j_{d-1},\cdots,j_0} (|j_{d-1}\rangle\langle j_{d-1}|) \otimes \cdots \otimes (|j_0\rangle\langle j_0|) \otimes \prod_{i=0}^{d-1} U^{j_i 2^i} \\
&= \prod_{i=0}^{d-1}{}' \left(\sum_{j_i} |j_i\rangle\langle j_i| \otimes U^{j_i 2^i} \right) \\
&= \prod_{i=0}^{d-1}{}' \left(|0\rangle\langle 0| \otimes I_n + |1\rangle\langle 1| \otimes U^{2^i} \right).
\end{aligned} \tag{3.39}$$

Here the primed product $\prod'$ is a slightly awkward notation, which means the tensor product for the first register, and the regular matrix product for the second register. It is in fact much clearer to observe the structure in the quantum circuit in Fig. 3.6.

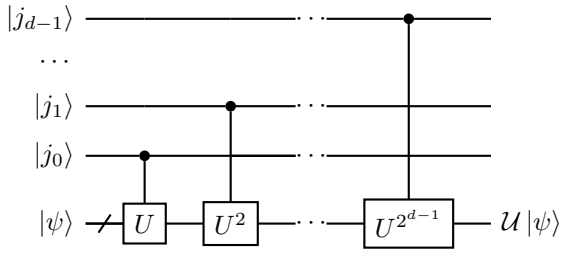


FIGURE 3.6. Circuit for controlled matrix power of U.

Remark 3.9. At first glance, the saving due to the usage of the circuit in Fig. 3.6 may not seem to be large, since we still need to implement matrix powers as high as $U^{2^{d-1}}$. However, the alternative would be to implement $\sum_{j\in[2^d]} |j\rangle\langle j|$, which requires very complicated multi-qubit control operations. Another scenario when significant advantage can be gained is when U can be *fast-forwarded*, i.e., U^j can be implemented at a cost that is independent of j. This is for instance, if $U = R_z(\theta)$ is a single-qubit rotation. Then the circuit Fig. 3.6 is exponentially better than the direct implementation of $\mathcal{U}$. ◇

Now let the initial state in the ancilla qubits be $|0^n\rangle$. Use QFT and $\mathcal{U}$, we transform the initial states according to

$$
\begin{aligned}
|0^d\rangle|\psi_0\rangle &\xrightarrow{U_{\mathrm{FT}}\otimes I} \frac{1}{\sqrt{2^d}}\sum_{j\in[2^d]}|j\rangle|\psi_0\rangle \\
&\xrightarrow{\mathcal{U}} \frac{1}{\sqrt{2^d}}\sum_{j\in[2^d]}|j\rangle U^j|\psi_0\rangle = \frac{1}{\sqrt{2^d}}\sum_{j\in[2^d]}|j\rangle e^{\mathrm{i}2\pi\varphi j}|\psi_0\rangle \\
&\xrightarrow{U_{\mathrm{FT}}^\dagger\otimes I} \sum_{k'\in[2^d]}\left(\frac{1}{2^d}\sum_{j\in[2^d]} e^{\mathrm{i}2\pi j\left(\varphi-\frac{k'}{2^d}\right)}\right)|k'\rangle|\psi_0\rangle\,.
\end{aligned}
\tag{3.40}
$$

Since we have $\varphi = \frac{k}{2^d}$ for *some* $k \in [2^d]$, measuring the ancilla qubits, and we will obtain the state $|k\rangle|\psi_0\rangle$ with certainty, and we obtain the phase information. Therefore the quantum circuit for QFT based QPE is given by Fig. 3.7. Here we have used Eq. (3.26). We should note that $U_{\mathrm{FT}}^\dagger$ includes the swapping operations. (Exercise: 1. what if the swap operation is not implemented? 2. Is it possible to modify the circuit and remove the need of implementing the swap operations?)

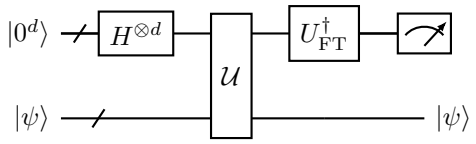


FIGURE 3.7. Quantum circuit for quantum phase estimation using quantum Fourier transform.

Example 3.10 (Hadamard test viewed as QPE)**.** When $d = 1$, note that $U^\dagger = U_{\mathrm{FT}}^\dagger = H$, the QFT based QPE in Fig. 3.7 is exactly the Hadamard test in Fig. 3.1. Note that φ does not need to be exactly represented by a one bit number! ◇

Example 3.11 (Qiskit example for QPE)**.** https://qiskit.org/textbook/ch-algorithms/quantum-phase-estimation.html ◇

3.5. Analysis of quantum phase estimation

In order to apply QPE (standard or Kitaev), we have assumed that

(1) $|\psi_0\rangle$ is an eigenstate.
(2) φ_0 has an d-bit binary representation.

In general practical calculations, neither condition can be *exactly* satisfied, and we need to analyze the effect on the error of the QPE. Recall the discussion in Section 1.9, we assume the only sources of errors are at the mathematical level (instead of noisy implementation of quantum devices). In this context, the error can be due to an inexact eigenstate $|\psi\rangle$, or Monte Carlo errors in the readout process due to the probabilistic nature of the measurement process. In this section, we relax these constraints and study what happens when the conditions are not exactly met. We assume U has the eigendecomposition.

$$U|\psi_j\rangle = e^{\mathrm{i}2\pi\varphi_j}|\psi_j\rangle\,. \tag{3.41}$$

Without loss of generality we assume $0 \le \varphi_0 \le \varphi_1 \cdots \le \varphi_{N-1} < 1$. We are interested in using QPE to find the value of φ_0.

We first relax the condition (1), i.e., assume *all* φ_i's have an exact d-bit binary representation, but the quantum state is given by a linear combination

$$|\phi\rangle = \sum_{k\in[N]} c_k|\psi_k\rangle\,. \tag{3.42}$$

Here the overlap $p_0 = |\langle\phi|\psi_0\rangle|^2 = |c_0|^2 < 1$.

Applying the QPE circuit in Fig. 3.7 to $|0^t\rangle|\phi\rangle$ with $t = d$, and measure the ancilla qubits, we obtain the binary representation of φ_0 with probability p_0. Furthermore, the system register returns the eigenstate $|\psi_0\rangle$ with probability p_0. Of course, in order to recognize that φ_0 is the desired phase, we need some *a priori* knowledge of the location of φ_0, e.g. $\varphi_0 \in (a, b)$ and $\varphi_i > b$ for all $i \neq 0$. It would be desirable to relax both conditions (1) and (2). However, the analysis can be rather involved and additional assumptions are needed. We will discuss some of the implications in the context of estimating the ground state energy in Section 4.1.

For now, to simplify the analysis, we focus on the case that only the condition (2) is violated, i.e., φ_0 cannot be exactly represented by a d-bit number, and we need to apply the QPE circuit to

an initial state $|0^t\rangle |\phi\rangle$ with $t > d$. The exact relation between the t and the desired accuracy d will be determined later. Similar to Eq. (3.40), we obtain the state

$$\begin{aligned} |0^t\rangle |\psi_0\rangle &\to \sum_{k' \in [T]} \left(\frac{1}{T} \sum_{j \in [T]} e^{\mathrm{i}2\pi j \left(\varphi_0 - \frac{k'}{T}\right)} \right) |k'\rangle |\psi_0\rangle \\ &= \sum_{k'} \gamma_{0,k'} |k'\rangle |\psi_0\rangle . \end{aligned} \tag{3.43}$$

Here

$$\gamma_{0,k'} = \frac{1}{T} \sum_{j \in [T]} e^{\mathrm{i}2\pi j \left(\varphi_0 - \frac{k'}{T}\right)} = \frac{1}{T} \frac{1 - e^{\mathrm{i}2\pi T(\varphi_0 - \widetilde{\varphi}_{k'})}}{1 - e^{\mathrm{i}2\pi(\varphi_0 - \widetilde{\varphi}_{k'})}}, \quad \widetilde{\varphi}_{k'} = \frac{k'}{T}. \tag{3.44}$$

Therefore if φ_0 has an exact d-bit representation, i.e., $\varphi_0 = \widetilde{\varphi}_{k'_0}$ for some k'_0, then $\gamma_{0,k'} = \delta_{k',k'_0}$. We recover the previous result that one run of the QPE circuit gives the value φ_0 deterministically.

Now assume that $\varphi_0 \neq \widetilde{\varphi}_{k'}$ for any k'. Note that $e^{\mathrm{i}2\pi x}$ is a periodic function with period 1, we can only determine the value of x mod 1. Therefore we use the periodic distance Eq. (3.17). In terms of the phase, we would like to find k'_0 such that

$$\left|\varphi_0 - \widetilde{\varphi}_{k'_0}\right|_1 < \epsilon. \tag{3.45}$$

Here $\epsilon = 2^{-d} = 2^{t-d}/T$ is the precision parameter. In particular, for any k' we have

$$|\varphi_0 - \widetilde{\varphi}_{k'}|_1 \leq \frac{1}{2}. \tag{3.46}$$

Using the relation that for any $\theta \in [-\pi, \pi]$,

$$\left|1 - e^{\mathrm{i}\theta}\right| = \sqrt{2(1 - \cos\theta)} = 2\,|\sin(\theta/2)| \geq \frac{2}{\pi} |\theta| , \tag{3.47}$$

we obtain

$$|\gamma_{0,k'}| \leq \frac{2}{T \frac{2}{\pi} 2\pi \,|\varphi_0 - \widetilde{\varphi}_{k'}|_1} = \frac{1}{2T \,|\varphi_0 - \widetilde{\varphi}_{k'}|_1}. \tag{3.48}$$

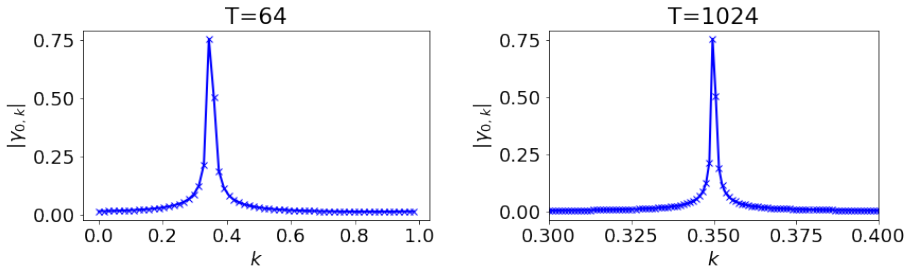


FIGURE 3.8. For $\varphi_0 = 0.35$, the shape of $|\gamma_{0,k}|$ with $T = 64$ and $T = 1024$.

Let k_0' be the measurement outcome, which can be viewed as a random variable. The probability of obtaining some $\widetilde{\varphi}_{k_0'}$ that is at least ϵ distance away from φ_0 is

$$\begin{aligned} P(\left|\varphi_0 - \widetilde{\varphi}_{k_0'}\right|_1 \geq \epsilon) = & \sum_{|\varphi_0 - \widetilde{\varphi}_{k'}|_1 \geq \epsilon} |\gamma_{0,k'}|^2 \\ \leq & \sum_{|\varphi_0 - \widetilde{\varphi}_{k'}|_1 \geq \epsilon} \frac{1}{4T^2 \left|\varphi_0 - \widetilde{\varphi}_{k'}\right|_1^2} \\ \leq & \frac{2}{4T} \int_{\epsilon}^{\infty} \frac{1}{x^2} \, \mathrm{d}x + \frac{2}{4T^2\epsilon^2} = \frac{1}{2T\epsilon} + \frac{1}{2(T\epsilon)^2}. \end{aligned} \tag{3.49}$$

Set $t - d = \lceil \log_2 \delta^{-1} \rceil$, then $T\epsilon = 2^{t-d} \geq \delta^{-1}$. Hence for any $0 < \delta < 1$, the failure probability

$$P(\left|\varphi_0 - \widetilde{\varphi}_{k_0'}\right|_1 \geq \epsilon) \leq \frac{\delta + \delta^2}{2} \leq \delta. \tag{3.50}$$

In other words, in order to obtain the phase φ_0 to accuracy $\epsilon = 2^{-d}$ with a success probability at least $1 - \delta$, we need $d + \lceil \log_2 \delta^{-1} \rceil$ ancilla qubits to store the value of the phase. On top of that, the simulation time needs to be $T = (\epsilon\delta)^{-1}$.

Remark 3.12 (Quantum median method)**.** One problem with QPE is that in order to obtain a success probability $1 - \delta$, we must use $\log_2 \delta^{-1}$ ancilla qubits, and the maximal simulation time also needs to be increased by a factor δ^{-1}. The increase of the maximal simulation time is particularly undesirable since it increases the circuit depth and hence the required coherence time of the quantum device. When $|\psi\rangle$ is an exact eigenstate, this can be improved by the median method, which uses $\log \delta^{-1}$ copies of the result from QPE without using ancilla qubits or increasing the circuit depth. When $|\psi\rangle$ is a linear combination of eigenstates, the problem of the aliasing effect becomes more difficult to handle. One possibility is to generalize the median method into the quantum median method [NWZ09], which uses classical arithmetics to evaluate the median using a quantum circuit. To reach success probability $1 - \delta$, we still need $\log_2 \delta^{-1}$ ancilla qubits, but the maximal simulation time does not need to be increased. ◇

Exercise 3.1. Write down the quantum circuit for the overlap estimate in Example 3.2.

Exercise 3.2. For $\varphi = 0.111111$, we run Kitaev's algorithm to estimate its first 4 bits. Check that the outcome satisfies Eq. (3.19). Note that 0.0000 and 0.1111 are both acceptable answers. Prove the validity of Kitaev's algorithm in general in the sense of Eq. (3.19).

Exercise 3.3. For a 3 qubit system, explicitly construct the circuit for U_{FT} and its inverse.

Exercise 3.4. For a n-qubit system, write down the quantum circuit for the swap operation used in QFT.

Exercise 3.5. Similar to the Hadamard test in Example 3.10, develop an algorithm to perform QPE using the circuit in Fig. 3.7 with only $d = 2$, while the phase φ can be any number in $[0, 1/2)$.

CHAPTER 4

Applications of quantum phase estimation

4.1. Ground state energy estimation

As an application, we use QPE to solve the problem of estimating the ground state energy of a Hamiltonian. Let H be a Hermitian matrix with eigendecomposition

$$H\left|\psi_j\right\rangle = \lambda_j \left|\psi_j\right\rangle. \tag{4.1}$$

Below are two examples of Hamiltonians commonly encountered in quantum many-body physics.

Example 4.1 (Transverse field Ising model)**.** The Hamiltonian for the one dimensional transverse field Ising model (TFIM) with nearest neighbor interaction of length n is

$$H = -\sum_{i=1}^{n-1} Z_i Z_{i+1} - g\sum_{i=1}^{n} X_i. \tag{4.2}$$

The dimension of the Hamiltonian matrix H is 2^n. ◇

Example 4.2 (Fermionic system in second quantization)**.** For a fermionic system (such as electrons), the Hamiltonian can be expressed in terms of the creation and annihilation operators as

$$H = \sum_{ij=1}^{n} T_{ij}\hat{a}_i^\dagger \hat{a}_j + \sum_{ijkl=1}^{n} V_{ijkl}\hat{a}_i^\dagger \hat{a}_j^\dagger \hat{a}_l \hat{a}_k. \tag{4.3}$$

The creation and annihilation operators $\hat{a}_i^\dagger, \hat{a}_i$ can be converted into Pauli operators via e.g. the Jordan-Wigner transform as

$$\hat{a}_i = Z^{\otimes(i-1)} \otimes \frac{1}{2}(X + \mathrm{i}Y) \otimes I^{\otimes(N-i)}, \quad \hat{a}_i^\dagger = Z^{\otimes(i-1)} \otimes \frac{1}{2}(X - \mathrm{i}Y) \otimes I^{\otimes(N-i)}. \tag{4.4}$$

Here X, Y, Z, I are single-qubit Pauli-matrices. The dimension of the Hamiltonian matrix $\hat{H}$ is thus 2^n.

The number operator takes the form

$$\hat{n}_i := \hat{a}_i^\dagger \hat{a}_i = \frac{1}{2}(I - Z_i). \tag{4.5}$$

For a given state $|\Psi\rangle$, the total number of particles is

$$N_e = \left\langle \Psi \middle| \sum_{i=1}^{n} \hat{n}_i \middle| \Psi \right\rangle. \tag{4.6}$$

The Hamiltonian H preserves the total number of particles N_e. ◇

Without loss of generality we assume $0 < \lambda_0 \leq \lambda_1 \leq \cdots < \lambda_{N-1} < \frac{1}{2}$. Note that for the purpose of estimating the ground state energy, we do not necessarily require a positive energy gap. For simplicity of the presentation, we still assume that the ground state is non-degenerate, i.e., $\lambda_0 < \lambda_1$. We are also provided an approximate eigenstate

$$|\phi\rangle = \sum_{k\in[N]} c_k |\psi_k\rangle , \tag{4.7}$$

of which the overlap with the ground state is $p_0 = |\langle\phi|\psi_0\rangle|^2$. Our goal is to estimate λ_0 to precision $\epsilon = 2^{-d}$. We assume $\epsilon < \lambda_0$. This appears in many problems in quantum many-body physics, quantum chemistry, optimization etc.

In order to use QPE (based on QFT), we assume access to the unitary evolution operator $U = e^{\mathrm{i}2\pi H}$. This is called a Hamiltonian simulation problem, which will be discussed in detail in later chapters. For now we assume U can be implemented exactly. Then

$$U |\psi_0\rangle = e^{\mathrm{i}2\pi\lambda_0} |\psi_0\rangle .$$

This becomes a phase estimation problem, where the input vector is not an exact eigenstate. Following the discussion in Section 3.5, if *all* eigenvalues λ_j can be exactly represented by d-bit numbers, we obtain both the ground state and the ground state energy with probability p_0. Therefore repeating the process for $\mathcal{O}(p_0^{-1})$ times we obtain the ground state energy.

Now we relax both conditions (1) and (2) in Section 3.5, and apply the QPE circuit in Fig. 3.7 to an initial state $|0^t\rangle |\phi\rangle$ for some $t > d$. Similar to Eq. (3.40), we have

$$\begin{aligned}
|0^t\rangle |\phi\rangle &\xrightarrow{U_{\mathrm{FT}}\otimes I} \sum_k c_k \frac{1}{\sqrt{T}} \sum_{j\in[T]} |j\rangle |\psi_k\rangle \\
&\xrightarrow{\mathcal{U}} \sum_k c_k \frac{1}{\sqrt{T}} \sum_{j\in[T]} |j\rangle U^j |\psi_k\rangle = \sum_k c_k \frac{1}{\sqrt{T}} \sum_{j\in[T]} |j\rangle e^{\mathrm{i}2\pi\lambda_k j} |\psi_k\rangle \\
&\xrightarrow{U_{\mathrm{FT}}^\dagger\otimes I} \sum_k c_k \sum_{k'\in[T]} \left(\frac{1}{T} \sum_{j\in[T]} e^{\mathrm{i}2\pi j\left(\lambda_k - \frac{k'}{T}\right)} \right) |k'\rangle |\psi_k\rangle \\
&= \sum_k \sum_{k'\in[T]} c_k \gamma_{k,k'} |k'\rangle |\psi_k\rangle .
\end{aligned} \tag{4.8}$$

Here

$$\gamma_{k,k'} = \frac{1}{T} \sum_{j\in[T]} e^{\mathrm{i}2\pi j\left(\lambda_k - \frac{k'}{T}\right)} = \frac{1}{T} \frac{1 - e^{\mathrm{i}2\pi T(\lambda_k - \widetilde{\varphi}_{k'})}}{1 - e^{\mathrm{i}2\pi(\lambda_k - \widetilde{\varphi}_{k'})}}, \quad \widetilde{\varphi}_{k'} = \frac{k'}{T}. \tag{4.9}$$

Therefore the definition in Eq. (3.44) is a special case.

Our algorithm is simple: we would like to run QPE M times, and denote the output of the ℓ-th run by $\widetilde{\varphi}_{k'}^{(\ell)}$. Then we take the minimum of the measured output $\min_\ell \widetilde{\varphi}_{k'}^{(\ell)}$ as the estimate to the ground state energy. The hope is to obtain the ground state energy to accuracy ϵ with success probability at least $1 - \delta$ for any $\delta > 0$. Let us now analyze this algorithm.

If φ_0 has an exact d-bit representation, i.e., $\lambda_0 = \widetilde{\varphi}_{k_0'}$ for some integer k_0', then $\gamma_{k_0',k'} = \delta_{k_0',k'}$. It may seem that this implies with probability p_0, we obtain the exact estimate of φ_0, and correspondingly the eigenstate $|\psi_0\rangle$ is stored in the system register. This is much better than the previous assumption that *all* eigenvalues λ_j need to be represented by a d-bit number.

Unfortunately, this analysis is not correct. In fact, for any λ_k that does not have an exact t-bit representation (note that $t > d$), we may have $\gamma_{k,k''} \neq 0$ and $\widetilde{\varphi}_{k''} < \lambda_0$, i.e., we obtain an energy estimate that is lower than the ground state energy! Therefore the probability of ending up in the state $|k''\rangle |\psi_k\rangle$ is $|c_k \gamma_{k,k''}|^2$, i.e., it is still possible to obtain a wrong ground state energy. This is called the *aliasing effect*.

We demonstrate below that if T is large enough, we can control the probability of underestimating the ground state energy. Since λ_0 is the ground state energy, and all eigenvalues are in $(0, 1/2)$, when $\widetilde{\varphi}_{k'} \leq \lambda_0 - \epsilon$, we have

$$|\lambda_k - \widetilde{\varphi}_{k'}|_1 \geq |\lambda_0 - \widetilde{\varphi}_{k'}|_1 = \lambda_0 - \widetilde{\varphi}_{k'} \geq \epsilon, \quad \forall k \in [N]. \tag{4.10}$$

Then the probability of under estimating the ground state energy by ϵ is

$$\begin{aligned} P(\widetilde{\varphi}_{k'} \leq \lambda_0 - \epsilon) =& \sum_k \sum_{\lambda_0 - \widetilde{\varphi}_{k'} \geq \epsilon} |c_k \gamma_{k,k'}|^2 \\ \leq& \sum_k \sum_{\lambda_0 - \widetilde{\varphi}_{k'} \geq \epsilon} p_k \frac{1}{4T^2 |\lambda_k - \widetilde{\varphi}_{k'}|_1^2} \\ \leq& \sum_k \sum_{\lambda_0 - \widetilde{\varphi}_{k'} \geq \epsilon} p_k \frac{1}{4T^2 |\lambda_0 - \widetilde{\varphi}_{k'}|^2} \\ =& \sum_{\lambda_0 - \widetilde{\varphi}_{k'} \geq \epsilon} \frac{1}{4T^2 |\lambda_0 - \widetilde{\varphi}_{k'}|^2} \\ \leq& \frac{1}{4T\epsilon} + \frac{1}{4(T\epsilon)^2}. \end{aligned} \tag{4.11}$$

Let $T\epsilon = \delta'^{-1}$ with $\delta' < 1/2$, we have

$$P(\widetilde{\varphi}_{k'} \leq \lambda_0 - \epsilon) \leq \frac{\delta'}{2}. \tag{4.12}$$

Therefore after M repetitions, we have

$$P\left(\min_\ell \widetilde{\varphi}_{k'}^{(\ell)} \leq \lambda_0 - \epsilon\right) \leq \frac{M\delta'}{2}. \tag{4.13}$$

In order to obtain $P\left(\min_\ell \widetilde{\varphi}_{k'}^{(\ell)} \leq \lambda_0 - \epsilon\right) < \frac{\delta}{2}$, we need to set $\delta' = M^{-1}\delta$.

On the other hand, we also would like to have bound $P\left(\min_\ell \widetilde{\varphi}_{k'}^{(\ell)} \geq \lambda_0 + \epsilon\right)$. To this end, we first note that when $\widetilde{\varphi}_{k'} \geq \lambda_0 + \epsilon$, we have $|\widetilde{\varphi}_{k'} - \lambda_0|_1 \geq \epsilon$. Moreover,

$$\begin{aligned} P(|\widetilde{\varphi}_{k'} - \lambda_0|_1 < \epsilon) =& \sum_k \sum_{|\widetilde{\varphi}_{k'} - \lambda_0|_1 < \epsilon} |c_k \gamma_{k,k'}|^2 \\ \geq& p_0 \sum_{|\widetilde{\varphi}_{k'} - \lambda_0|_1 < \epsilon} |\gamma_{0,k'}|^2 \\ =& p_0 \left(1 - \sum_{|\widetilde{\varphi}_{k'} - \lambda_0|_1 \geq \epsilon} |\gamma_{0,k'}|^2\right) \\ \geq& p_0 \left(1 - \frac{1}{2T\epsilon} - \frac{1}{2(T\epsilon)^2}\right) \\ \geq& p_0(1 - \delta'). \end{aligned} \tag{4.14}$$

Here we have used the normalization condition that

$$\sum_{k'} |\gamma_{k,k'}|^2 = 1, \quad \forall k. \tag{4.15}$$

Therefore

$$P(|\widetilde{\varphi}_{k'} - \lambda_0|_1 \geq \epsilon) = 1 - P(|\widetilde{\varphi}_{k'} - \lambda_0|_1 < \epsilon) \leq 1 - p_0(1 - \delta') \leq 1 - \frac{p_0}{2}. \tag{4.16}$$

This means that

$$P(\min_\ell \widetilde{\varphi}_{k'}^{(\ell)} \geq \lambda_0 + \epsilon) \leq P(|\widetilde{\varphi}_{k'} - \lambda_0|_1 \geq \epsilon)^M = (1 - p_0/2)^M \leq e^{-\frac{p_0 M}{2}}. \tag{4.17}$$

We can then take $M = \lceil \frac{2}{p_0} \log \frac{2}{\delta} \rceil$ so that

$$P(\min_\ell \widetilde{\varphi}_{k'}^{(\ell)} \geq \lambda_0 + \epsilon) \leq \frac{\delta}{2}. \tag{4.18}$$

To summarize, according to the relation $\delta' = M^{-1}\delta$, in order to estimate the ground state energy to precision $\epsilon = 2^{-d}$ with success probability $1 - \delta$, we need

$$t = d + \lceil \log \delta'^{-1} \rceil = d + \mathcal{O}(\log p_0^{-1} + \log\log \delta^{-1}) \tag{4.19}$$

ancilla qubits in QPE. The circuit depth is

$$T = \mathcal{O}\left((\epsilon\delta p_0)^{-1} \log \delta^{-1}\right). \tag{4.20}$$

Taking the number of repetitions M into account, the total cost of the method is

$$\mathcal{O}\left(\epsilon^{-1}\delta^{-1} p_0^{-2} (\log \delta^{-1})^2\right). \tag{4.21}$$

Remark 4.3 (Dependence on the initial overlap p_0)**.** The analysis above shows that QPE has a nontrivial dependence on the initial overlap p_0, which has a rather indirect source. First, in order to reduce the possibility of overshooting, the number of repetitions M needs to be large enough and is $\mathcal{O}(p_0^{-1})$. However, this also increases the probability of undershooting the eigenvalue, and hence δ' needs to be chosen to be $\mathcal{O}(M^{-1}) = \mathcal{O}(p_0)$. This means that the circuit depth should be $\mathcal{O}(\delta') = \mathcal{O}(p_0^{-1})$. The total complexity is thus $TM = \mathcal{O}(p_0^{-2})$. Therefore, when the initial overlap p_0 is small, using QPE to find the ground state energy can be very costly. On the other hand, using different techniques, the dependence on p_0 can be drastically improved to $\mathcal{O}(p_0^{-\frac{1}{2}})$ [LT20b]. See also the discussions in Section 7.8.

◇

Remark 4.4 (QPE for preparing the ground state)**.** The estimate of the ground state energy does not necessarily require the energy gap $\lambda_g := \lambda_1 - \lambda_0$ to be positive. However, if our goal is to prepare the ground state $|\psi_0\rangle$ from an initial state $|\phi\rangle$ using QPE, then we need stronger assumptions. In particular, we cannot afford to obtain $|k'\rangle|\psi_k\rangle$, where $|\widetilde{\varphi}_{k'} - \lambda_0| < \epsilon$ but $k \neq 0$. This at least requires $\epsilon = 2^{-d} < \lambda_g$, and introduces a natural dependence on the inverse of the gap. ◇

Through the analysis above, we see that although the analysis of QPE is very clean when 1) all eigenvalues (properly scaled to be represented as phase factors) are exactly given by d-bit numbers 2) the input vector is an eigenstate, the analysis can become rather complicated and tedious when such conditions are relaxed. Such difficulty does not merely show up at the theoretical level, but can seriously impact the robust performance of QPE in practical applications. To simplify the discussion of the applications below, we will be much more cavalier about the usage of QPE and assume all eigenvalues are exactly represented by d-bit numbers whenever necessary. But we should keep such caveats in mind. Furthermore, when we move beyond QPE, the issue of having exact d-bit numbers will become much less severe in techniques based on quantum signal processing, i.e., quantum eigenvalue transformation (QET) and quantum singular value transformation (QSVT).

4.2. Amplitude estimation

Let $|\psi_0\rangle$ be prepared by an oracle U_{ψ_0}, i.e., $U_{\psi_0}|0^n\rangle = |\psi_0\rangle$. We have the knowledge that

$$|\psi_0\rangle = \sqrt{p_0}\,|\psi_{\text{good}}\rangle + \sqrt{1-p_0}\,|\psi_{\text{bad}}\rangle\,, \tag{4.22}$$

and $p_0 \ll 1$. Here $|\psi_{\text{bad}}\rangle$ is an orthogonal state to the desired state $|\psi_{\text{good}}\rangle$, and $\sqrt{p_0} = \sin\frac{\theta}{2}$. The problem of amplitude estimation is to estimate p_0 to ϵ-precision. If p_0 is away from 0 or 1 and is to be estimated directly from the Monte Carlo method, the number of samples needed is $\mathcal{N} = \mathcal{O}(\epsilon^{-2})$.

Let $G = R_{\psi_0} R_{\text{good}}$ be the Grover operator as in Section 2.3. Then in the basis $\mathcal{B} = \{|\psi_{\text{good}}\rangle, |\psi_{\text{bad}}\rangle\}$, the subspace $\mathcal{H} = \operatorname{span} B$ is an invariant subspace of G. Recall the computation of Eq. (2.23), the matrix representation is

$$[G]_{\mathcal{B}} = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}. \tag{4.23}$$

Its two eigenstates are

$$|\psi_\pm\rangle = \frac{1}{\sqrt{2}}\left(|\psi_{\text{good}}\rangle \pm \mathrm{i}\,|\psi_{\text{bad}}\rangle\right), \tag{4.24}$$

with eigenvalues $e^{\pm \mathrm{i}\theta}$, respectively.

Therefore the problem of estimating θ can be solved with phase estimation with an imperfect initial state. Note that

$$|\langle\psi_0|\psi_+\rangle|^2 = \frac{1}{2}\left|\sin\frac{\theta}{2} + \mathrm{i}\cos\frac{\theta}{2}\right|^2 = \frac{1}{2} = |\langle\psi_0|\psi_-\rangle|^2\,. \tag{4.25}$$

Consider a QPE circuit with t ancilla qubits and querying G for $T = 2^t$ times. Then each execution with the system register will be in $|\psi_+\rangle$ or $|\psi_-\rangle$ states each with probability 0.5. Let

$$t = d + \lceil \log \delta^{-1} \rceil \tag{4.26}$$

be the number of ancilla qubits with $\epsilon' = 2^{-d}$. Then QPE obtains an estimate denoted by $\widetilde{\theta}$, which approximates θ to precision ϵ' with success probability $1-\delta$. Note that $p_0 = \sin^2\frac{\theta}{2}$, and

$$\begin{aligned}
&\sin^2\frac{\widetilde{\theta}}{2} - \sin^2\frac{\theta}{2} \\
=&\sin^2\frac{\widetilde{\theta}-\theta}{2}\cos^2\frac{\theta}{2} + \cos^2\frac{\widetilde{\theta}-\theta}{2}\sin^2\frac{\theta}{2} + 2\sin\frac{\theta}{2}\cos\frac{\theta}{2}\sin\frac{\widetilde{\theta}-\theta}{2}\cos\frac{\widetilde{\theta}-\theta}{2} - \sin^2\frac{\theta}{2} \\
=&\sin\frac{\theta}{2}\cos\frac{\theta}{2}\sin(\widetilde{\theta}-\theta) + \left(1-2\sin^2\frac{\theta}{2}\right)\sin^2\frac{\widetilde{\theta}-\theta}{2}.
\end{aligned} \tag{4.27}$$

Using the fact that $\left|\sin(\widetilde{\theta}-\theta)\right| \le \left|\widetilde{\theta}-\theta\right| \le \epsilon'$, we have

$$|\widetilde{p}-p| \le \sqrt{p_0(1-p_0)}\epsilon' + (1-2p_0)\frac{\epsilon'^2}{4}. \tag{4.28}$$

Let ϵ' be sufficiently small. Now if $p_0(1-p_0) = \Omega(1)$, we can choose $\epsilon' = \mathcal{O}(\epsilon)$, and the total complexity of QPE is $\mathcal{O}(\epsilon^{-1})$.

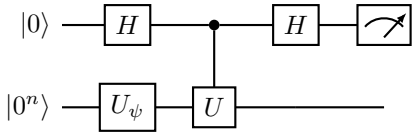
If p_0 is small, then we should estimate p_0 to multiplicative accuracy ϵ instead. Use

$$|\widetilde{p}-p| \approx \sqrt{p_0}\epsilon' < p_0\epsilon, \tag{4.29}$$

we have $\epsilon' = \sqrt{p_0}\epsilon$. Therefore the runtime of QPE is $\mathcal{O}(p_0^{-\frac{1}{2}}\epsilon^{-1})$. If p_0 is to be estimated to precision ϵ' using the Monte Carlo method, the number of samples would be $\mathcal{N} = \mathcal{O}(p_0^{-1}\epsilon^{-2})$.

So, the amplitude estimation method achieves quadratic speedup in the total complexity, but the circuit depth is increased to $\mathcal{O}(\epsilon'^{-1}\delta^{-1})$.

Example 4.5 (Amplitude estimation to accelerate Hadamard test)**.** Consider the circuit for the Hadamard test in Fig. 3.1 to estimate $\mathrm{Re}\,\langle\psi|U|\psi\rangle$. Let the initial state $|\psi\rangle$ be prepared by a unitary U_ψ, then the following combined circuit



maps $|0\rangle|0^n\rangle$ to

$$|\psi_0\rangle = \frac{1}{2}|0\rangle(|\psi\rangle + U|\psi\rangle) + \frac{1}{2}|1\rangle(|\psi\rangle - U|\psi\rangle) := \sqrt{p_0}\,|\psi_{\mathrm{good}}\rangle + \sqrt{1-p_0}\,|\psi_{\mathrm{bad}}\rangle, \tag{4.30}$$

and the goal is to estimate p_0. This also gives the implementation of the reflector R_{ψ_0}.

Note that R_{good} can be implemented by simply reflecting against the signal qubit, i.e.,

$$R_{\mathrm{good}} = (I - 2|0\rangle\langle 0|)\otimes I_n = -Z\otimes I_n. \tag{4.31}$$

Then we can run QPE to the Grover unitary $G = R_{\psi_0}R_{\mathrm{good}}$ to estimate $p(0)$, and the circuit depth is $\mathcal{O}(\epsilon^{-1})$. ◇

4.3. HHL algorithm for solving linear systems of equations

In this section, we consider the solution of linear systems of equations

$$Ax = b, \tag{4.32}$$

where $A \in \mathbb{C}^{N\times N}$ is a non-singular matrix. Without loss of generality we assume A is Hermitian. Otherwise, we can solve a dilated Hermitian matrix, which enlarges the matrix dimension by a factor of 2 (i.e., use one ancilla qubit)

$$\widetilde{A} = \begin{bmatrix} 0 & A^\dagger \\ A & 0 \end{bmatrix} = |1\rangle\langle 0| \otimes A + |0\rangle\langle 1| \otimes A^\dagger, \tag{4.33}$$

and solve the enlarged problem

$$\widetilde{A}\,|0, x\rangle = |1, b\rangle\,. \tag{4.34}$$

We assume $b \in \mathbb{C}^N$ is a normalized vector and hence can be stored in a quantum state. More specifically, we have a unitary U_b such that (may require some work registers)

$$|b\rangle = U_b\,|0^n\rangle\,. \tag{4.35}$$

On classical computers, the solution is simply $x = A^{-1}b$. However, the solution vector is in general not a normalized vector, and hence cannot be directly stored as a quantum state. Therefore the goal of solving the quantum linear system problem (QLSP) is to find a quantum state $|\widetilde{x}\rangle$ so that

$$\||\widetilde{x}\rangle - |x\rangle\| \le \epsilon, \quad |x\rangle = \frac{A^{-1}\,|b\rangle}{\|A^{-1}\,|b\rangle\|}. \tag{4.36}$$

The normalization constant $\left\|A^{-1}\,|b\rangle\right\|$ should be recovered separately.

One useful application of QLSP solvers is to evaluate the many-body Green's function [NO88], based on the quantum many-body Hamiltonian in Example 4.2. We will omit the details here.

4.3.1. Algorithmic description. The HHL algorithm [HHL09], based on QPE, is the first quantum algorithm for solving QLSP. The algorithm can be summarized as follows. Let A have the following eigendecomposition

$$A\,|v_j\rangle = \lambda_j\,|v_j\rangle\,. \tag{4.37}$$

To simplify the analysis, we assume $0 < \lambda_0 \le \lambda_1 \le \ldots \le \lambda_{N-1} < 1$ and all eigenvalues have an exact d-bit representation. The analysis can also be generalized to the case when A has negative eigenvalues, but the interpretation of the result from QPE needs to be modified accordingly.

The matrix A can be queried using Hamiltonian simulation as $U = e^{\mathrm{i}2\pi A}$. Then if the input state is already one of the eigenvectors, i.e., $|b\rangle = |v_j\rangle$, then QPE can be applied to implement the mapping

$$U_{\mathrm{QPE}}\,|0^d\rangle\,|v_j\rangle = |\lambda_j\rangle\,|v_j\rangle\,. \tag{4.38}$$

In general, let the input state $|b\rangle = \sum_j \beta_j\,|v_j\rangle$ be expanded using the eigendecomposition of A. Then by linearity,

$$U_{\mathrm{QPE}}\,|0^d\rangle\,|b\rangle = \sum_j \beta_j\,|\lambda_j\rangle\,|v_j\rangle\,. \tag{4.39}$$

Note that the unnormalized solution satisfies

$$A^{-1}\,|b\rangle = \sum_j \frac{\beta_j}{\lambda_j}\,|v_j\rangle\,, \tag{4.40}$$

so all we need to do is to use the information of the eigenvalue $|\lambda_j\rangle$ stored in the ancilla register, and perform a controlled rotation to multiply the factor λ_j^{-1} to *each* β_j. To this end, we see that it is crucial to store all eigenvalues in the quantum computer coherently, as achieved by QPE. We would like to implement the following controlled rotation unitary (see Section 4.3.2)

$$U_{\mathrm{CR}} |0\rangle |\lambda_j\rangle = \left(\sqrt{1 - \frac{C^2}{\widetilde{\lambda}_j^2}} |0\rangle + \frac{C}{\widetilde{\lambda}_j} |1\rangle \right) |\lambda_j\rangle . \tag{4.41}$$

where each $\widetilde{\lambda}_j$ approximates λ_j.

Finally, perform uncomputation by applying $U_{\mathrm{QPE}}^\dagger$, we convert the information from the ancilla register $|\lambda_j\rangle$ back to $|0^d\rangle$. Therefore the quantum circuit for the HHL algorithm is in Fig. 4.1.

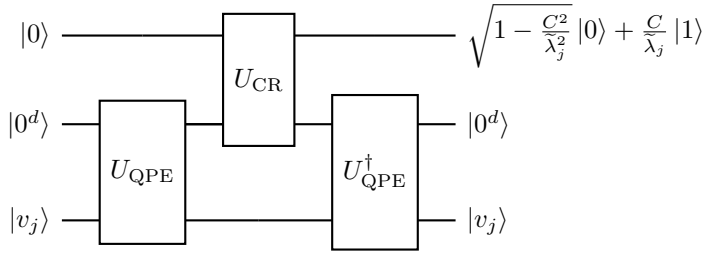


FIGURE 4.1. Circuit for the HHL algorithm.

Note that through the uncomputation, the d ancilla qubits for storing the eigenvalues also becomes a working register. Discarding all working registers, the resulting unitary denoted by U_{HHL} satisfies

$$U_{\mathrm{HHL}} |0\rangle |b\rangle = \sum_j \left(\sqrt{1 - \frac{C^2}{\widetilde{\lambda}_j^2}} |0\rangle + \frac{C}{\widetilde{\lambda}_j} |1\rangle \right) \beta_j |v_j\rangle . \tag{4.42}$$

Finally, measuring the signal qubit (the only ancilla qubit left), if the outcome is 1, we obtain the (unnormalized) vector

$$\widetilde{x} = \sum_j \frac{C\beta_j}{\widetilde{\lambda}_j} |v_j\rangle \tag{4.43}$$

stored as a normalized state in the system register is

$$|\widetilde{x}\rangle = \frac{\widetilde{x}}{\|\widetilde{x}\|} \approx |x\rangle , \tag{4.44}$$

which is the desired approximate solution to QLSP. In particular, the constant C does not appear in the solution.

Remark 4.6 (Recovering the norm of the solution)**.** The HHL algorithm returns the solution to QLSP in the form of a normalized state $|x\rangle$ stored in the quantum computer. In order to recover

the magnitude of the unnormalized solution $\|\widetilde{x}\|$, we note that the success probability of measuring the signal qubit in Eq. (4.42) is

$$p(1) = \left\| \sum_j \frac{C\beta_j}{\widetilde{\lambda}_j} |v_j\rangle \right\|^2 = \|\widetilde{x}\|^2 . \tag{4.45}$$

Therefore sufficient repetitions of running the circuit in Eq. (4.42) and estimate $p(1)$, we can obtain an estimate of $\|\widetilde{x}\|$. ◇

More general discussion of the readout problem of the HHL algorithm will be given in Remark 4.10.

4.3.2. Implementation of controlled rotation.

Proposition 4.7 (Controlled rotation given rotation angles)**.** *Let $0 \le \theta < 1$ has exact d-bit fixed point representation $\theta = .\theta_{d-1}\cdots\theta_0$ be its d-bit fixed point representation. Then there is a $(d+1)$-qubit unitary U_θ such that*

$$U_\theta : |0\rangle|\theta\rangle \mapsto (\cos(\pi\theta)|0\rangle + \sin(\pi\theta)|1\rangle)|\theta\rangle. \tag{4.46}$$

Proof. First (by e.g. Taylor expansion)

$$\exp(-\mathrm{i}\tau\sigma_y) = \begin{pmatrix} \cos(\tau) & -\sin(\tau) \\ \sin(\tau) & \cos(\tau) \end{pmatrix} =: R_y(2\tau). \tag{4.47}$$

Here $R_y(\cdot)$ perform a single-qubit rotation around the y-axis. For any $j \in [2^d]$ with its binary representation $j = j_{d-1}\cdots j_0$, we have

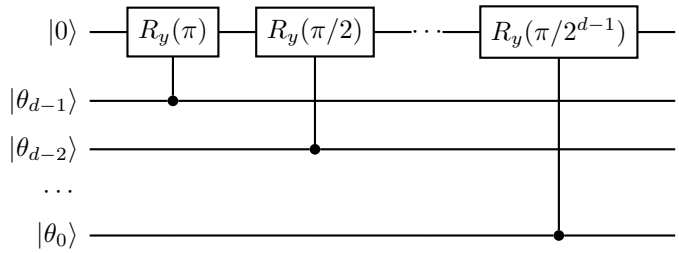
$$j/2^d = (.j_{d-1}\cdots j_0). \tag{4.48}$$

So choose $\tau = \pi(.j_{d-1}\cdots j_0)$, and define

$$U_\theta = \sum_{j\in[2^d]} \exp(-\mathrm{i}\pi(.j_{d-1}\cdots j_0)\sigma_y) \otimes |j\rangle\langle j| . \tag{4.49}$$

Applying U_θ to $|0\rangle|\theta\rangle$ gives the desired results. □

The quantum circuit for the controlled rotation circuit is



This is a sequence of single-qubit rotations on the signal qubit, each controlled by a single qubit.

In order to use the controlled rotation operation, we need to store the information of λ_j in term of an angle θ_j. Let $C > 0$ be a lower bound to λ_0, so that $0 < C/\lambda_j < 1$ for all j. Define

$$\theta_j = \frac{1}{\pi}\arcsin(C/\lambda_j), \tag{4.50}$$

and $\widetilde{\theta}_j$ be its d'-bit representation. Then

$$\sin \pi\widetilde{\theta}_j \equiv \frac{C}{\widetilde{\lambda}_j} \approx \frac{C}{\lambda_j}. \tag{4.51}$$

Again for simplicity we assume d' is large enough so that the error of the fixed point representation is negligible in this step. The mapping

$$U_{\text{angle}} |0^{d'-d}\rangle |\lambda_j\rangle = |\widetilde{\theta}_j\rangle \tag{4.52}$$

can be implemented using *classical arithmetics circuits* in Section 1.8, which may require $\text{poly}(d')$ gates and an additional working register of $\text{poly}(d')$ qubits, which are not displayed here. Therefore the entire controlled rotation operation needed for the HHL algorithm is given by the circuit in Fig. 4.2.

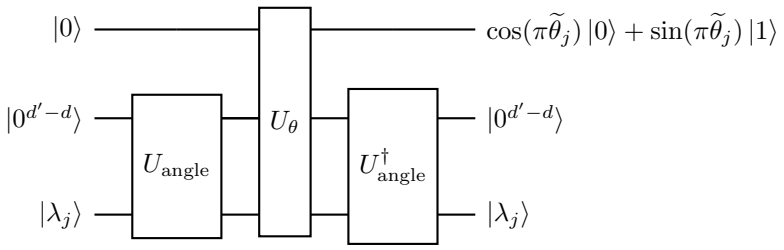


FIGURE 4.2. Circuit for the controlled rotation step used by the HHL algorithm (not including additional working register for classical arithmetic operations).

Therefore through the uncomputation $U^\dagger_{\text{angle}}$, the $d'-d$ ancilla qubits also become a working register. Discard the working register, and we obtain a unitary U_{CR} satisfying

$$U_{\text{CR}} |0\rangle |\lambda_j\rangle = \left(\cos(\pi\widetilde{\theta}_j)|0\rangle + \sin(\pi\widetilde{\theta}_j)|1\rangle\right)|\lambda_j\rangle = \left(\sqrt{1-\frac{C^2}{\widetilde{\lambda}_j^2}}|0\rangle + \frac{C}{\widetilde{\lambda}_j}|1\rangle\right)|\lambda_j\rangle. \tag{4.53}$$

This is the unitary used in the HHL circuit in Fig. 4.1.

4.3.3. Complexity analysis of the HHL algorithm. Although the choice of the constant C does not appear in the normalized quantum state, it does directly affect the success probability. From Eq. (4.42) we immediately obtain the success probability for measuring the signal qubit with outcome 1 is the square of the norm of the unnormalized solution

$$p(1) = \|\widetilde{x}\|^2 \approx C^2 \left\|A^{-1}|b\rangle\right\|^2. \tag{4.54}$$

Therefore the success probability is determined by

(1) the choice of the normalization constant C,
(2) the norm of the true solution $\|x\| = \left\|A^{-1}|b\rangle\right\|$.

To maximize the success probability, C should be chosen to be as large as possible (without exceeding λ_0). So assuming the exact knowledge of λ_0, we can choose $C = \lambda_0$. For a Hermitian

positive definite matrix A, $\|A\| = \lambda_{N-1}$, and $\left\|A^{-1}\right\| = \lambda_0^{-1}$. For simplicity, assume the largest eigenvalue of A is $\lambda_{N-1} = 1$. Then the condition number of A is

$$\kappa := \|A\| \left\|A^{-1}\right\| = \frac{\lambda_{N-1}}{\lambda_0} = C^{-1}. \tag{4.55}$$

Furthermore,

$$\left\|A^{-1} |b\rangle\right\| \geq \frac{1}{\|A\|} \||b\rangle\| = 1. \tag{4.56}$$

Therefore

$$p(1) = \Omega(\kappa^{-2}). \tag{4.57}$$

In other words, in the worst case, we need to repeatedly run the HHL algorithm for $\mathcal{O}(\kappa^2)$ times to obtain the outcome 1 in the signal qubit.

Assuming the number of system qubits n is large, the circuit depth and the gate complexity of U_{HHL} is mainly determined by those of U_{QPE}. Therefore we can measure the complexity of the HHL algorithm in terms of the number of queries to $U = e^{\mathrm{i}2\pi A}$. In order to solve QLSP to precision ϵ, we need to estimate the eigenvalues to *multiplicative accuracy* ϵ instead of the standard additive accuracy.

To see why this is the case, assume $\widetilde{\lambda}_j = \lambda_j(1 + e_j)$ and $|e_j| \leq \frac{\epsilon}{4} \leq \frac{1}{2}$. Then the unnormalized solution satisfies

$$\|\widetilde{x} - x\| = \left\| \sum_j \beta_j \left(\frac{1}{\widetilde{\lambda}_j} - \frac{1}{\lambda_j} \right) |v_j\rangle \right\| \leq \left\| \sum_j \frac{\beta_j}{\lambda_j} \left(\frac{-e_j}{1 + e_j} \right) |v_j\rangle \right\| \leq \frac{\epsilon}{2} \|x\| . \tag{4.58}$$

Hence

$$\left| 1 - \frac{\|\widetilde{x}\|}{\|x\|} \right| \leq \frac{\epsilon}{2}. \tag{4.59}$$

Then the normalized solution satisfies

$$\||\widetilde{x}\rangle - |x\rangle\| = \left\| \frac{\widetilde{x}}{\|\widetilde{x}\|} - \frac{x}{\|x\|} \right\| \leq \left| 1 - \frac{\|\widetilde{x}\|}{\|x\|} \right| + \frac{\|\widetilde{x} - x\|}{\|x\|} \leq \epsilon. \tag{4.60}$$

The discussion requires the QPE to be run to additive precision $\epsilon' = \lambda_0 \epsilon = \epsilon/\kappa$. Therefore the query complexity of QPE is $\mathcal{O}(\kappa/\epsilon)$. Counting the number of times needed to repeat the HHL circuit, the worst case query complexity of the HHL algorithm is $\mathcal{O}(\kappa^3/\epsilon)$.

The above analysis the worst case analysis, because we assume $p(1)$ attains the lower bound $\Omega(\kappa^{-2})$. In practical applications, the result may not be so pessimistic. For instance, if β_j concentrates around the smallest eigenvalues of A, then we may have $\|x\| \sim \Theta(\lambda_0^{-1}) = \Theta(\kappa^{-1})$. Then $p(1) = \Theta(1)$. In such a case, we only need to repeat the HHL algorithm for a constant number of times to yield outcome 1 in the ancilla qubit. This does not reduce the query complexity of each run of the algorithm. Then in this *best* case, the query complexity is $\mathcal{O}(\kappa/\epsilon)$.

4.3.4. Additional considerations. Below we discuss a few more aspects of the HHL algorithm. The first observation is that the asymptotic worst-case complexity of the HHL algorithm can be generally improved using amplitude amplification.

Remark 4.8 (HHL with amplitude amplification)**.** We may view Eq. (4.42) as

$$U_{\mathrm{HHL}} |0\rangle |b\rangle = \sqrt{p(1)} |1\rangle |\psi_{\mathrm{good}}\rangle + \sqrt{1-p(1)} |0\rangle |\psi_{\mathrm{bad}}\rangle , \quad |\psi_{\mathrm{good}}\rangle = |\widetilde{x}\rangle . \tag{4.61}$$

Since $|\psi_{\mathrm{good}}\rangle$ is marked by a single signal qubit, we may use Example 2.5 to construct a reflection operator with respect to the signal qubit. This is simply given by

$$R_{\mathrm{good}} = Z \otimes I_n. \tag{4.62}$$

The reflection with respect to the initial vector is

$$R_{\psi_0} = U_{\psi_0}(2 |0^{1+n}\rangle \langle 0^{1+n}| - I)U_{\psi_0}^\dagger, \tag{4.63}$$

where $U_{\psi_0} = U_{\mathrm{HHL}}(I_1 \otimes U_b)$. Let $G = R_{\psi_0} R_{\mathrm{good}}$ be the Grover iterate. Then amplitude amplification allows us to apply G for $\Theta(p(1)^{-\frac{1}{2}})$ times to boost the success probability of obtaining $|\psi_{\mathrm{good}}\rangle$ with constant success probability. Therefore in the worst case when $p(1) = \Theta(\kappa^{-2})$, the number of repetitions is reduced to $\mathcal{O}(\kappa)$, and the total runtime is $\mathcal{O}(\kappa^2/\epsilon)$. This query complexity is the commonly referred query complexity for the HHL algorithm. Note that as usual, amplitude amplification increases the circuit depth. However, the tradeoff is that the circuit depth *increases* from $\mathcal{O}(\kappa/\epsilon)$ to $\mathcal{O}(\kappa^2/\epsilon)$. ◇

So far our analysis, especially that based on QPE relies on the assumption that all λ_j all eigenvalues have an exact d-bit representation. From the discussion in Section 3.5, we know that such an assumption is unrealistic and causes theoretical and practical difficulties. The full analysis of the HHL algorithm is thus more involved. We refer to e.g. [DHM+18] for more details.

Remark 4.9 (Comparison with classical iterative linear system solvers)**.** Let us now compare the cost of the HHL algorithm to that of classical iterative algorithms. If A is n-qubit Hermitian positive definite with condition number κ, and is d-sparse (i.e., each row/column of A has at most d nonzero entries), then each matrix vector multiplication Ax costs $\mathcal{O}(dN)$ floating point operations. The number of iterations for the steepest descent (SD) algorithm is $\mathcal{O}(\kappa \log \epsilon^{-1})$, and the this number can be significantly reduced to $\mathcal{O}(\sqrt{\kappa} \log \epsilon^{-1})$ by the renowned conjugate gradient (CG) method. Therefore the total cost (or wall clock time) of SD and CG is $\mathcal{O}(dN\kappa \log \epsilon^{-1})$ and $\mathcal{O}(dN\sqrt{\kappa} \log \epsilon^{-1})$, respectively.

On the other hand, the query complexity of the HHL algorithm, even after using the AA algorithm, is still $\mathcal{O}(\kappa^2/\epsilon)$. Such a performance is terrible in terms of both κ and ϵ. Hence the power of the HHL algorithm, and other QLSP solvers is based on that each application of A (in this case, using the unitary U) is *much* faster. In particular, if U can be implemented with $\mathrm{poly}(n)$ gate complexity (also can be measured by the wall clock time), then the total gate complexity of the HHL algorithm (with AA) is $\mathcal{O}(\mathrm{poly}(n)\kappa^2/\epsilon)$. When n is large enough, we expect that $\mathrm{poly}(n) \ll N = 2^n$ and the HHL algorithm would eventually yield an advantage. Nonetheless, for a realistic problem, the assumption that U can be implemented with $\mathrm{poly}(n)$ cost, and *no* classical algorithm can implement Ax with $\mathrm{poly}(n)$ cost should be taken with a grain of salt and carefully examined. ◇

Remark 4.10 (Readout problem of QLSP)**.** By solving the QLSP, the solution is stored as a quantum state in the quantum computer. Sometimes the QLSP is only a subproblem of a larger application, so it is sufficient to treat the HHL algorithm (or other QLSP solvers) as a "quantum subroutine", and leave $|x\rangle$ in the quantum computer. However, in many applications (such as the solution of Poisson's equation in Section 4.4, the goal is to solve the lienar system. Then the information in $|x\rangle$ must be converted to a measurable classical output.

The most common case is to compute the expectation of some observable $\langle O\rangle = \langle x|O|x\rangle \approx \langle \widetilde{x}|O|\widetilde{x}\rangle$. Assuming $\langle O\rangle = \Theta(1)$. Then to reach additive precision ϵ of the observable, the number of samples needed is $\mathcal{O}(\epsilon^{-2})$. On the other hand, in order to reach precision ϵ, the solution vector $|\widetilde{x}\rangle$ must be solved to precision ϵ. Assuming the worst case analysis for the HHL algorithm, the total query complexity needed is

$$\mathcal{O}(\kappa^2/\epsilon) \times \mathcal{O}(\epsilon^{-2}) = \mathcal{O}(\kappa^2/\epsilon^3). \tag{4.64}$$

◇

Remark 4.11 (Query complexity lower bound)**.** The cost of a quantum algorithm for solving a generic QLSP scales at least as $\Omega(\kappa(A))$, where $\kappa(A) := \|A\| \left\|A^{-1}\right\|$ is the condition number of A. The proof is based on converting the QLSP into a Hamiltonian simulation problem, and the lower bound with respect to κ is proved via the "no-fast-forwarding" theorem for simulating generic Hamiltonians [HHL09]. Nonetheless, for specific classes of Hamiltonians, it may still be possible to develop fast algorithms to overcome this lower bound. ◇

4.4. Example: Solve Poisson's equation

As an application of the HHL algorithm, let us consider a toy problem of solving the Poisson's equation in one-dimension with Dirichlet boundary conditions

$$-u''(r) = b(r), \quad r \in \Omega = [0,1], \quad u(0) = u(1) = 0. \tag{4.65}$$

We use the central finite difference formula to discretize the Laplacian operator on a uniform grid $r_i = (i+1)h, i \in [N]$ and $h = 1/(N+1)$. Let $u_i = u(r_i), b_i = b(r_i)$, then Poisson's equation is discretized into a linear system of equations $Au = b$, with a tridiagonal matrix

$$A = \frac{1}{h^2}\begin{pmatrix} 2 & -1 & 0 & \cdots & 0 & 0 & 0 \\ -1 & 2 & -1 & \cdots & 0 & 0 & 0 \\ \vdots & & & & \vdots & & \\ 0 & 0 & 0 & \cdots & -1 & 2 & -1 \\ 0 & 0 & 0 & \cdots & 0 & -1 & 2 \end{pmatrix}. \tag{4.66}$$

Our goal here is not to address the quality of the spatial discretization, but to study the cost for solving the linear system. To this end we need to compute the condition number of A. We also assume the right hand vector b is already normalized so that $|b\rangle = b$.

Proposition 4.12 (Diagonalization of tridiagonal matrices)**.** *A Hermitian Toeplitz tridiagonal matrix*

$$A = \begin{pmatrix} a & \overline{b} & 0 & \cdots & 0 & 0 & 0 \\ b & a & \overline{b} & \cdots & 0 & 0 & 0 \\ \vdots & & & & \vdots & & \\ 0 & 0 & 0 & \cdots & b & a & \overline{b} \\ 0 & 0 & 0 & \cdots & 0 & b & a \end{pmatrix} \in \mathbb{C}^{N\times N} \tag{4.67}$$

can be analytically diagonalized as

$$Av_k = \lambda_k v_k, \quad k = 1, \ldots, N. \tag{4.68}$$

where $(v_k)_j = v_{j,k}, j = 1, \ldots, N$, $b = |b|\, e^{\mathrm{i}\theta}$, *and*

$$\lambda_k = a + 2\,|b| \cos\frac{k\pi}{N+1}, \quad v_{j,k} = \sin\frac{jk\pi}{N+1} e^{\mathrm{i}j\theta}. \tag{4.69}$$

PROOF. Note that formally $v_{0,k} = v_{N+1,k} = 0$. Then direct matrix vector multiplication shows that for any $j = 1, \ldots, N$,

$$(Av_k)_j = a \sin \frac{jk\pi}{N+1} e^{\mathrm{i}j\theta} + 2\,|b| \sin \frac{jk\pi}{N+1} \cos \frac{k\pi}{N+1} e^{\mathrm{i}j\theta} = \lambda_k (v_k)_j. \tag{4.70}$$

□

Using Proposition 4.12 with $a = 2/h^2, b = -1/h^2$, the largest eigenvalue of A is $\lambda_{\max} = \|A\| \approx 4/h^2$, and the smallest eigenvalue $\lambda_{\min} = \left\|A^{-1}\right\|^{-1} \approx \pi^2$. So

$$\kappa(A) \approx \frac{4}{h^2\pi^2} = \mathcal{O}(N^2). \tag{4.71}$$

The circuit depth of the HHL algorithm is $\mathcal{O}(N^2/\epsilon)$, and the worst case query complexity (using AA) is $\mathcal{O}(N^4/\epsilon)$. So when N is large, there is *little benefit* in employing the quantum computer to solve this problem.

Let us now consider solving a d-dimensional Poisson's equation with Dirichlet boundary conditions

$$-\Delta u(\mathbf{r}) = b(\mathbf{r}), \quad \mathbf{r} \in \Omega = [0,1]^d, \quad u|_{\partial\Omega} = 0. \tag{4.72}$$

The grid is the Cartesian product of the uniform grid in 1D with N grid points per dimension and $h = 1/(N+1)$. The total number of grid points is $\mathcal{N} = N^d$. After discretization, we obtain a linear system $\mathcal{A}u = b$, where

$$\mathcal{A} = A \otimes I \otimes \cdots I + \cdots + I \otimes \cdots I \otimes A, \tag{4.73}$$

where I is an identity matrix of size N. Since $\mathcal{A}$ is Hermitian and positive definite, we have $\|\mathcal{A}\| \approx 4d/h^2$, and $\left\|\mathcal{A}^{-1}\right\|^{-1} \approx d\pi^2$. So $\kappa(\mathcal{A}) \approx 4/(h^2\pi^2) \approx \kappa(A)$. Therefore the condition number is independent of the spatial dimension d.

The worst case query complexity of the HHL algorithm is $\mathcal{O}(N^2/\epsilon)$. So when the number of grid points per dimension N is fixed and the spatial dimension d increases, and if $U = e^{\mathrm{i}\mathcal{A}\tau}$ can be implemented efficiently for some $\tau\,\|\mathcal{A}\| < 1$ with $\mathrm{poly}(d)$ cost, then the HHL algorithm will have an advantage over classical solvers, for which each matrix-vector multiplication scales linearly with respect to $\mathcal{N}$ and is therefore exponential in d.

4.5. Solve linear differential equations*

Consider the solution of a time-dependent linear differential equation

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t} x(t) &= A(t)x(t) + b(t), \quad t \in [0,T], \\ x(0) &= x_0. \end{aligned} \tag{4.74}$$

Here $b(t), x(t) \in \mathbb{C}^d$ and $A(t) \in \mathbb{C}^{d\times d}$. Eq. (4.74) can be used to represent a very large class of ordinary differential equations (ODEs) and spatially discretized partial differential equations (PDEs). For instance, if $A(t) = -\mathrm{i}H(t)$ for some Hermitian matrix $H(t)$ and $b(t) = 0$, this is the time-dependent Schrödinger equation

$$\mathrm{i}\frac{\mathrm{d}}{\mathrm{d}t} x(t) = H(t)x(t). \tag{4.75}$$

A special case when $H(t) \equiv H$ is often called the Hamiltonian simulation problem, and the solution can be written as

$$x(T) = e^{-\mathrm{i}HT}x(0), \tag{4.76}$$

which can be viewed as the problem of evaluating the matrix function $e^{-\mathrm{i}HT}$. This will be discussed separately in later chapters.

In this section we consider the general case of Eq. (4.74), and for simplicity discretize the equation in time using the forward Euler method with a uniform grid $t_k = k\Delta t$ where $\Delta t = T/N, k = 0, \ldots, N$. Let $A_k = A(t_k), b_k = b(t_k)$. The resulting discretized system becomes

$$x_{k+1} - x_k = \Delta t(A_k x_k + b_k), \quad k = 1, \ldots, N, \tag{4.77}$$

which can be rewritten as

$$\begin{pmatrix} I & 0 & 0 & \cdots & 0 & 0 \\ -(I+\Delta t A_1) & I & 0 & \cdots & 0 & 0 \\ 0 & -(I+\Delta t A_2) & I & \cdots & 0 & 0 \\ \vdots & & & & & \vdots \\ 0 & 0 & 0 & \cdots & -(I+\Delta t A_{N-1}) & I \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_N \end{pmatrix} = \begin{pmatrix} (I+\Delta t A_0)x_0 + \Delta t b_0 \\ \Delta t b_1 \\ \Delta t b_2 \\ \vdots \\ \Delta t b_{N-1} \end{pmatrix}, \tag{4.78}$$

or more compactly as a linear systems of equations

$$\mathcal{A}\mathbf{x} = \mathbf{b}. \tag{4.79}$$

Here I is the identity matrix of size d, and $\mathbf{x} \in \mathbb{R}^{Nd}$ encodes the entire history of the states.

To solve Eq. (4.78) as a QLSP, the right hand side $\mathbf{b}$ needs to be a normalized vector. This means we need to properly normalize x_0, b_k so that

$$\|\mathbf{b}\|^2 = \|(I+\Delta t A_0)x_0 + \Delta t b_0\|^2 + \sum_{k\in[N-1]} (\Delta t)^2 \|b_k\|^2 = 1. \tag{4.80}$$

In the limit $\Delta t \to 0$, this can be written as

$$\|\mathbf{b}\|^2 = \|x_0\|^2 + \int_0^T \|b(t)\|^2 \,\mathrm{d}t = 1, \tag{4.81}$$

which is not difficult to satisfy as long as $x_0, b(t)$ can be prepared efficiently using unitary circuits and $\|x_0\|, \|b(t)\| = \Theta(1)$.

To solve Eq. (4.78) using the HHL algorithm, we need to estimate the condition number of $\mathcal{A}$. Note that $\mathcal{A}$ is a block-bidiagonal matrix and in particular is not Hermitian. So we need to use the dilation method in Eq. (4.33) and solve the corresponding Hermitian problem.

4.5.1. Scalar case. In order to estimate the condition number, for simplicity we first assume $d = 1$ (i.e., this is a scalar ODE problem), and $A(t) \equiv a \in \mathbb{C}$ is a constant. Then

$$\mathcal{A} = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ -(1+\Delta t a) & 1 & 0 & \cdots & 0 & 0 \\ 0 & -(1+\Delta t a) & 1 & \cdots & 0 & 0 \\ \vdots & & & & & \vdots \\ 0 & 0 & 0 & \cdots & -(1+\Delta t a) & 1 \end{pmatrix} \in \mathbb{C}^{N\times N}. \tag{4.82}$$

Let $\xi = 1 + \Delta t a$. When $\operatorname{Re} a \leq 0$, the absolute stability condition of the forward Euler method requires $|1 + \Delta t a| = |\xi| < 1$. In general we are interested in the regime $\Delta t\,|a| \ll 1$, and in particular $|1 + \Delta t a| = |\xi| < 2$.

Proposition 4.13. *For any $A \in \mathbb{C}^{N\times N}$, $\|A\|^2, (1/\left\|A^{-1}\right\|)^2$ are given by the largest and smallest eigenvalue of $A^\dagger A$, respectively.*

From Proposition 4.13, we need to first compute

$$\mathcal{A}^\dagger\mathcal{A} = \begin{pmatrix} 1+|\xi|^2 & -\overline{\xi} & \cdots & 0 & 0 & \\ -\xi & 1+|\xi|^2 & -\overline{\xi} & \cdots & 0 & 0 \\ 0 & -\xi & 1+|\xi|^2 & \cdots & 0 & 0 \\ \vdots & & & & & \vdots \\ 0 & 0 & 0 & \cdots & 1+|\xi|^2 & -\overline{\xi} \\ 0 & 0 & 0 & \cdots & -\xi & 1 \end{pmatrix}. \tag{4.83}$$

Theorem 4.14 (Gershgorin circle theorem, see e.g. [GVL13, Theorem 7.2.1]). *Let $A \in \mathbb{C}^{N\times N}$ with entries a_{ij}. For each $i = 1, \ldots, N$, define*

$$R_i = \sum_{j\neq i} |a_{ij}|\,. \tag{4.84}$$

Let $D(a_{ii}, R_i) \subseteq \mathbb{C}$ be a closed disc centered at a_{ii} with radius R_i, which is called a Gershgorin disc. Then every eigenvalue of A lies within at least one of the Gershgorin discs $D(a_{ii}, R_i)$

Since $\mathcal{A}^\dagger\mathcal{A}$ is Hermitian, we can restrict the Gershgorin discs to the real line so that $D(a_{ii}, R_i) \subseteq \mathbb{R}$. Then Gershgorin discs of the matrix $\mathcal{A}^\dagger\mathcal{A}$ satisfy the bound

$$\begin{aligned} D(a_{11}, R_1) &\subseteq \left[1+|\xi|^2 - |\xi|\,, 1+|\xi|^2 + |\xi|\right], \\ D(a_{ii}, R_i) &\subseteq \left[1+|\xi|^2 - 2\,|\xi|\,, 1+|\xi|^2 + 2\,|\xi|\right], \quad i = 2, \ldots, N-1 \\ D(a_{NN}, R_N) &\subseteq [1-|\xi|\,, 1+|\xi|]\,. \end{aligned} \tag{4.85}$$

Applying Theorem 4.14 we have

$$\lambda_{\max}(\mathcal{A}^\dagger\mathcal{A}) \leq (1+|\xi|)^2 < 9. \tag{4.86}$$

for all values of a such that $|\xi| < 2$.

To obtain a meaningful lower bound of $\lambda_{\min}(\mathcal{A}^\dagger\mathcal{A})$, we need $\operatorname{Re} a < 0$ and hence $|\xi| < 1$. Use the inequality

$$\sqrt{1+x} \leq 1 + \frac{1}{2}x, \quad x > -1, \tag{4.87}$$

we have

$$1 - |\xi| = 1 - \sqrt{(1+\Delta t \operatorname{Re} a)^2 + \Delta t^2 (\operatorname{Im} a)^2} \geq -\Delta t \operatorname{Re} a - \frac{(\Delta t\,|a|)^2}{2} \geq -\frac{\Delta t}{2}\operatorname{Re} a, \tag{4.88}$$

when

$$\Delta t\,|a| < (-\operatorname{Re} a)/\,|a| \tag{4.89}$$

is satisfied. Then we have

$$1 > 1 - |\xi| \geq -\frac{\Delta t}{2}\operatorname{Re} a. \tag{4.90}$$

Then according to Theorem 4.14, under the condition Eq. (4.89),

$$\lambda_{\min}(\mathcal{A}^\dagger \mathcal{A}) \geq (1 - |\xi|)^2 \geq \frac{(\Delta t \operatorname{Re} a)^2}{4}. \tag{4.91}$$

Therefore

$$\|\mathcal{A}\| = \sqrt{\lambda_{\max}(\mathcal{A}^\dagger \mathcal{A})} \leq \sqrt{1 + |\xi|^2 + 2|\xi|} < 3, \tag{4.92}$$

and

$$\left\|\mathcal{A}^{-1}\right\|^{-1} = \sqrt{\lambda_{\min}(\mathcal{A}^\dagger \mathcal{A})} \geq \frac{\Delta t(-\operatorname{Re} a)}{2}. \tag{4.93}$$

As a result, when $(-\operatorname{Re} a) = \Theta(1)$, the condition number satisfies

$$\kappa(\mathcal{A}) = \|\mathcal{A}\| \left\|\mathcal{A}^{-1}\right\| = \mathcal{O}\left(1/\Delta t\right). \tag{4.94}$$

In summary, for the scalar problem $d = 1$ and $\operatorname{Re} a < 0$, the query complexity of the HHL algorithm is $\mathcal{O}((\Delta t)^{-2}\epsilon^{-2})$.

Example 4.15 (Growth of the condition number when $\operatorname{Re} a \geq 0$)**.** The Gershgorin circle theorem does not provide a meaningful bound of the condition number when $\operatorname{Re} a > 0$ and $1 < |\xi| < 2$. This is a correct behavior. To see this, just consider $a = 1, b = 0$, and the solution should grow exponentially as $x(T) = e^T$. If $\kappa(\mathcal{A}) = \mathcal{O}\left((\Delta t)^{-1}\right)$ holds and in particular is independent of the final T, then the norm of the solution can only grow polynomially in T, which is a contradiction. See Fig. 4.3 for an illustration. Note that when $a = -1.0$, $\Delta t = 0.1$, the condition number is less than 20, which is smaller than the upper bound above, i.e., $3 \times \frac{2}{\Delta t(-a)} = 60$.

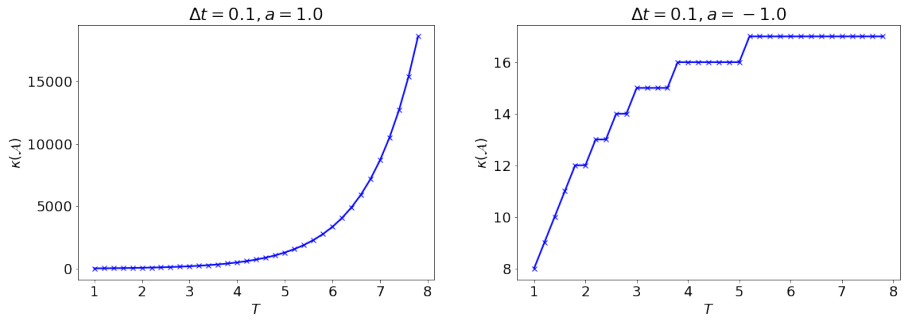


FIGURE 4.3. Growth of the condition number $\kappa(\mathcal{A})$ with respect to T with a fixed step size $\Delta t = 0.1$ for $a = 1.0$ and $a = -1.0$.

◇

Remark 4.16. It may be tempting to modify the (N,N)-th entry of $\mathcal{A}$ to be $1+|\xi|^2$ to obtain

$$\mathcal{G} = \begin{pmatrix} 1+|\xi|^2 & -\overline{\xi} & \cdots & 0 & 0 & \\ -\xi & 1+|\xi|^2 & -\overline{\xi} & \cdots & 0 & 0 \\ 0 & -\xi & 1+|\xi|^2 & \cdots & 0 & 0 \\ \vdots & & & & & \vdots \\ 0 & 0 & 0 & \cdots & 1+|\xi|^2 & -\overline{\xi} \\ 0 & 0 & 0 & \cdots & -\xi & 1+|\xi|^2 \end{pmatrix}. \tag{4.95}$$

Here $\mathcal{G}$ is a Toeplitz tridiagonal matrix satisfying the requirement of Proposition 4.12. The eigenvalues of $\mathcal{G}$ take the form

$$\lambda_k = 1+|\xi|^2 + 2|\xi| \cos\frac{k\pi}{N+1}, \quad k=1,\dots,N. \tag{4.96}$$

If we take the approximation $\lambda_{\min}(\mathcal{A}^\dagger\mathcal{A}) \approx \lambda_{\min}(\mathcal{G})$, we would find that Eq. (4.94) holds even when $\operatorname{Re} a > 0$. This behavior is however incorrect, despite that the matrices $\mathcal{A}^\dagger\mathcal{A}$ and $\mathcal{G}$ only differ by a single entry! ◇

4.5.2. Vector case. Here we consider a general $d>1$, but for simplicity assume $A(t)\equiv A \in \mathbb{C}^{d\times d}$ is a constant matrix. We also assume A is diagonalizable with eigenvalue decomposition

$$A = V\Lambda V^{-1}, \tag{4.97}$$

and $\Lambda = \operatorname{diag}(\lambda_1,\dots,\lambda_N)$. We only consider the case $\operatorname{Re}\lambda_k < 0$ for all k.

Proposition 4.17. *For any diagonalizable $A\in\mathbb{C}^{N\times N}$ with eigenvalue decomposition $Av_k=\lambda_k v_k$, we have*

$$\left\|A^{-1}\right\|^{-1} \le \min_k |\lambda_k| \le \max_k |\lambda_k| \le \|A\|. \tag{4.98}$$

Proof. Use the Schur form $A=QTQ^\dagger$, where Q is an unitary matrix and T is an upper triangular matrix (see e.g. [GVL13, Theorem 7.13]). The diagonal entries of T encodes all eigenvalues of A, and the eigenvalues can appear in any order along the diagonal of T. The proposition follows by arranging the eigenvalue of A with the smallest and largest absolute values to the (N,N)-th entry, respectively. □

The absolute stability condition of the forward Euler method requires $\Delta t\|A\|<1$, and we are interested in the regime $\Delta t\|A\| \ll 1$. Therefore $\Delta t|\lambda_k| \ll 1$ for all k.

Let I be an identity matrix of size d, and denote by $B=-(I+\Delta t A)$, then

$$\mathcal{A}^\dagger\mathcal{A} = \begin{pmatrix} I+B^\dagger B & B^\dagger & \cdots & 0 & 0 & \\ B & I+B^\dagger B & B^\dagger & \cdots & 0 & 0 \\ 0 & B & I+B^\dagger B & \cdots & 0 & 0 \\ \vdots & & & & & \vdots \\ 0 & 0 & 0 & \cdots & I+B^\dagger B & B^\dagger \\ 0 & 0 & 0 & \cdots & B & I \end{pmatrix}. \tag{4.99}$$

Note that

$$\|B\| \le \left\|I+B^\dagger B\right\| \le 1+(1+\Delta t\|A\|)^2 \le 5. \tag{4.100}$$

For any $\mathbf{x} \in \mathbb{C}^{Nd}$,

$$\begin{aligned}\left\|\mathcal{A}^\dagger \mathcal{A}\mathbf{x}\right\|^2 \le \left\|I + B^\dagger B\right\| \Big[&(\|x_1\|^2 + \|x_2\|^2) + (\|x_1\|^2 + \|x_2\|^2 + \|x_3\|^2) + \cdots \\ &+ (\|x_{N-1}\|^2 + \|x_N\|^2)\Big] \le 15 \|\mathbf{x}\|^2 .\end{aligned} \tag{4.101}$$

So $\lambda_{\max}(\mathcal{A}^\dagger \mathcal{A}) \le 15$, and

$$\|\mathcal{A}\| = \sqrt{\lambda_{\max}(\mathcal{A}^\dagger \mathcal{A})} \le \sqrt{15} = \mathcal{O}(1). \tag{4.102}$$

To bound $\left\|\mathcal{A}^{-1}\right\|$, we first note that from the eigenvalue decomposition of A, we have

(4.103)

$$\mathcal{A} = \begin{pmatrix} V & & & \\ & V & & \\ & & \ddots & \\ & & & V \end{pmatrix} \begin{pmatrix} I & 0 & 0 & \cdots & 0 & 0 \\ -(I+\Delta t\Lambda) & I & 0 & \cdots & 0 & 0 \\ 0 & -(I+\Delta t\Lambda) & I & \cdots & 0 & 0 \\ \vdots & & & & & \vdots \\ 0 & 0 & 0 & \cdots & -(I+\Delta t\Lambda) & I \end{pmatrix} \begin{pmatrix} V^{-1} & & & \\ & V^{-1} & & \\ & & \ddots & \\ & & & V^{-1} \end{pmatrix}.$$

Hence

$$\left\|\mathcal{A}^{-1}\right\| \le \|V\| \left\|V^{-1}\right\| \max_k \left\|\mathcal{A}_k^{-1}\right\| = \kappa(V) \left\|\mathcal{A}_k^{-1}\right\|. \tag{4.104}$$

Here $\kappa(V) = \|V\| \left\|V^{-1}\right\|$ is the condition number of the eigenvector matrix

$$\mathcal{A}_k = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 0 \\ -(1+\Delta t\lambda_k) & 1 & 0 & \cdots & 0 & 0 \\ 0 & -(1+\Delta t\lambda_k) & 1 & \cdots & 0 & 0 \\ \vdots & & & & & \vdots \\ 0 & 0 & 0 & \cdots & -(1+\Delta t\lambda_k) & 1 \end{pmatrix}. \tag{4.105}$$

This reduces the problem to the scalar case. Let $\xi_k = 1 + \Delta t\lambda_k$, and assume

$$\operatorname{Re}\lambda_k < 0, \quad \Delta t\,|\lambda_k| < (-\operatorname{Re}\lambda_k)/|\lambda_k|, \quad \forall k. \tag{4.106}$$

Then from Eq. (4.93) we have

$$\min_k \left\|\mathcal{A}_k^{-1}\right\|^{-1} \ge \frac{\Delta t \min_k(-\operatorname{Re}\lambda_k)}{2}. \tag{4.107}$$

So if $\min_k(-\operatorname{Re}\lambda_k) = \Theta(1)$, we have

$$\kappa(\mathcal{A}) = \mathcal{O}(\kappa(V)/\Delta t), \tag{4.108}$$

and the cost of the HHL algorithm is $\mathcal{O}((\Delta t)^{-2}\epsilon^{-2}\kappa(V))$. Hence compared to the scalar case, the condition number of the eigenvector matrix V can play an important role.

4.5.3. Computing observables. The solution of Eq. (4.78) means that the normalized state $|\mathbf{x}\rangle$ is computed to precision ϵ and stored in the quantum computer. In order to evaluate observables at the final time T, i.e., $\langle x(T)|O|x(T)\rangle$, we find that by the normalization condition, $\|x(T)\|$ is on average $\mathcal{O}(N^{-\frac{1}{2}}) = \mathcal{O}((\Delta t)^{\frac{1}{2}})$, and $\langle x(T)|O|x(T)\rangle = \mathcal{O}(\Delta t)$. Therefore instead of reaching accuracy ϵ, the Monte Carlo procedure must reach precision $\mathcal{O}(\epsilon\Delta t)$. This increases the number of samples by another factor of $\mathcal{O}((\Delta t)^{-2})$.

There is however a simple way to overcome this problem. Instead of solving Eq. (4.78), we can redefine $\mathbf{x}$ by artificially padding the vector with N copies of the final state x_N. This can be written as and can be reinterpreted as

$$\mathbf{X} = |0\rangle \otimes \mathbf{x} + |1\rangle \otimes \mathbf{y}, \tag{4.109}$$

with the unnormalized vector

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{pmatrix}, \quad \mathbf{y} = \begin{pmatrix} x_{N+1} \\ x_{N+2} \\ \vdots \\ x_{2N} \end{pmatrix} = \begin{pmatrix} x_N \\ x_N \\ \vdots \\ x_N \end{pmatrix}, \tag{4.110}$$

and the corresponding linear systems of equation becomes

(4.111)

$$\begin{pmatrix} I & & & & & & & \\ -(I+\Delta t A_1) & I & & & & & & \\ & \ddots & & & & & & \\ & -(I+\Delta t A_{N-1}) & I & & & & & \\ & & -I & I & & & & \\ & & & -I & I & & & \\ & & & & & \ddots & & \\ & & & & & & -I & I \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \\ x_{N+1} \\ x_{N+2} \\ \vdots \\ x_{2N} \end{pmatrix} = \begin{pmatrix} (I+\Delta t A_0)x_0 + \Delta t b_0 \\ \Delta t b_1 \\ \vdots \\ \Delta t b_{N-1} \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

Note that the solution vector only requires one ancilla qubit, after solving the equation The condition number of this modified equation is still $\kappa = \mathcal{O}((\Delta t)^{-1})$, so the total query complexity is $\mathcal{O}((\Delta t)^{-2}\epsilon^{-2})$. By solving the modified equation Eq. (4.111) to precision ϵ, we can estimate $\langle x(T)|O|x(T)\rangle$ by

$$\langle \mathbf{y}|I \otimes O|\mathbf{y}\rangle \tag{4.112}$$

of which the magnitude does not scale with Δt. Here I is the identity matrix of size N, and $\mathbf{y}$ can be obtained by measuring the ancilla qubit and obtain 1. If the norm of $\|x(t)\|$ is comparable for all $t \in [0, T]$, then the success probability will be $\Theta(1)$ after $|\mathbf{X}\rangle$ is obtained.

4.6. Example: Solve the heat equation*

As an application of the differential equation solver in 4.5, let us consider a toy problem of solving the heat equation in one-dimension with Dirichlet boundary conditions

$$\partial_t u(r,t) = u''(r), \quad r \in \Omega = [0,1], \quad u(0,t) = u(1,t) = 0. \tag{4.113}$$

After spatial discretization using the central finite difference method with d grid points, this becomes a linear ODE system

$$\partial_t u = -Au, \tag{4.114}$$

where $A \in \mathbb{R}^{N\times N}$ is a tridiagonal matrix given by Eq. (4.66). After applying the forward Euler method and discretize the simulation time T into L intervals with $\Delta t = T/L$, we obtain a linear system of size NL. The eigenvalues of $-A$, denoted by λ_k, are all negative and satisfy

$$-\frac{4}{h^2} \approx -\|A\| \leq \lambda_k \leq -\left\|A^{-1}\right\|^{-1} \approx -\frac{1}{\pi^2}. \tag{4.115}$$

The absolute stability condition requires $|1 + \Delta t \lambda_k| < 1$, or $\Delta t < h^2/4 = \mathcal{O}(N^{-2})$, which implies $L = \mathcal{O}(N^2)$. Since A is Hermitian, we have $\kappa(V) = 1$. So for $T = \mathcal{O}(1)$, the query complexity for solving the heat equation is $\mathcal{O}(N^2\epsilon^{-2})$, which is the same as solving Poisson's equation.

Again, the potential advantage of the quantum solver only appears when solving the d-dimensional heat equation

$$\partial_t u(\mathbf{r}, t) = \Delta u(r), \quad r \in \Omega = [0,1]^d, \quad u(\cdot, t)|_{\partial\Omega} = 0. \tag{4.116}$$

This can be written as a linear system of equations

$$\partial_t u = -\mathcal{A}u, \tag{4.117}$$

where $\mathcal{A}$ is given in Eq. (4.73). The eigenvalues of $\mathcal{A}$ are all negative. Note that $\|\mathcal{A}\| = \Theta(dN^2)$, then $h = \mathcal{O}(d^{-1}N^{-2})$, and the query complexity of the HHL solver is $\mathcal{O}(dN^2\epsilon^{-2})$. This could potentially have an exponential advantage over classical solvers.

Exercise 4.1 (Quantum counting)**.** Given query access to a function $f : \{0,1\}^N \to \{0,1\}$ design a quantum algorithm that computes the size of its kernel, i.e.,, total number of x's that satisfy $f(x) = 1$.

Exercise 4.2. Consider the initial value problem of the linear differential equation Eq. (4.74).

(1) Construct the linear system of equations

$$\mathcal{A}\mathbf{x} = \mathbf{b}$$

like Eq. (4.78) using the backward Euler method.

(2) In the scalar case when $A(t) \equiv a \in \mathbb{C}$ is a constant satisfying $\mathrm{Re}(a) \leq 0$, estimate the query complexity of the HHL algorithm applying to the linear system constructed in (1).

CHAPTER 5

Trotter based Hamiltonian simulation

The Hamiltonian simulation problem with a time-independent Hamiltonian, or the Hamiltonian simulation problem for short is the following problem: given an initial state $|\psi_0\rangle$ and a Hamiltonian H, evaluate the quantum state at time t according to $|\psi(t)\rangle = e^{-\mathrm{i}tH}|\psi_0\rangle$. 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 [Fey82]. We have also seen that Hamiltonian simulation appears as a quantum subroutine in numerous other quantum algorithms, such as QPE and its various applications.

The Hamiltonian simulation problem can also be viewed as a linear ODE:

$$\partial_t \psi(t) = -\mathrm{i}H\psi(t), \quad \psi(0) = \psi_0. \tag{5.1}$$

However, thanks to the unitarity of the operator $e^{-\mathrm{i}tH}$ for any t, we do not need to store the full history of the quantum states as in Section 4.5, and can instead focus on the quantum state at time t of interest.

Following the conceptualization of a universal quantum simulator using a Trotter decomposition of the time evolution operator $e^{-\mathrm{i}tH}$ [Llo96], many new quantum algorithms for Hamiltonian simulation have been proposed. We will discuss some more advanced methods in later chapters. This chapter focuses on the Trotter based Hamiltonian simulation method (also called the product formula).

5.1. Trotter splitting

Consider the Hamiltonian simulation problem for $H = H_1 + H_2$, where $e^{-\mathrm{i}H_1\Delta t}$ and $e^{-\mathrm{i}H_2\Delta t}$ can be efficiently computed at least for some Δt. In general $[H_1, H_2] \neq 0$, and the splitting of the evolution of H_1, H_2 needs to be implemented via the Lie product formula

$$e^{-\mathrm{i}tH} = \lim_{L\to\infty} \left(e^{-\mathrm{i}\frac{t}{L}H_1} e^{-\mathrm{i}\frac{t}{L}H_2}\right)^L. \tag{5.2}$$

When taking L to be a finite number, and let $\Delta t = t/L$, this gives the simplest first order Trotter method with

$$\left\|e^{-\mathrm{i}\Delta tH} - e^{-\mathrm{i}\Delta tH_1} e^{-\mathrm{i}\Delta tH_2}\right\| = \mathcal{O}(\Delta t^2), \tag{5.3}$$

Therefore to perform Hamiltonian simulation to time t, the error is

$$\left\|e^{-\mathrm{i}tH} - \left(e^{-\mathrm{i}\frac{t}{L}H_1} e^{-\mathrm{i}\frac{t}{L}H_2}\right)^L\right\| = \mathcal{O}(\Delta t^2 L) = \mathcal{O}\left(\frac{t^2}{L}\right). \tag{5.4}$$

So to reach precision ϵ in the operator norm, we need

$$L = \mathcal{O}(t^2\epsilon^{-1}). \tag{5.5}$$

This can be improved to the second order Trotter method (also called the symmetric Trotter splitting, or Strang splitting)

$$\left\| e^{-\mathrm{i}\Delta t H} - e^{-\mathrm{i}\Delta t/2 H_2} e^{-\mathrm{i}\Delta t H_1} e^{-\mathrm{i}\Delta t/2 H_2} \right\| = \mathcal{O}(\Delta t^3). \tag{5.6}$$

Following a similar analysis to the first order method, we find that to reach precision ϵ we need

$$L = \mathcal{O}(t^{3/2}\epsilon^{-1/2}). \tag{5.7}$$

Higher order Trotter methods are also available, such as the p-th order Suzuki formula. The local truncation error is $(\Delta t)^{p+1}$. Therefore to reach precision ϵ, we need

$$L = \mathcal{O}(t^{\frac{p+1}{p}}\epsilon^{-1/p}). \tag{5.8}$$

This is often written as $L = \mathcal{O}(t^{1+o(1)}\epsilon^{-o(1)})$ as $p \to \infty$.

Example 5.1 (Simulating transverse field Ising model)**.** For the one dimensional transverse field Ising model (TFIM) with nearest neighbor interaction in Eq. (4.2), wince all Pauli-Z_i operators commute, we have

$$e^{-\mathrm{i}tH_1} := e^{\mathrm{i}t\sum_{i=1}^{n-1} Z_i Z_{i+1}} = \prod_{i=1}^{n-1} e^{\mathrm{i}tZ_iZ_{i+1}}. \tag{5.9}$$

Each $e^{\mathrm{i}tZ_iZ_{i+1}}$ is a rotation involving only the qubits i, j, and the splitting has no error. Similarly

$$e^{-\mathrm{i}tH_2} := e^{g\sum_i X_i} = \prod_i e^{\mathrm{i}tgX_i}, \tag{5.10}$$

and each $e^{\mathrm{i}tgX_i}$ can be implemented independently without error. ◇

Example 5.2 (Particle in a potential)**.** Let $H = -\Delta_{\mathbf{r}} + V(\mathbf{r}) = H_1 + H_2$ be the Hamiltonian of a particle in a potential field $V(\mathbf{r})$, where $\mathbf{r} \in \Omega = [0,1]^d$ with periodic boundary conditions. After discretization using Fourier modes, $e^{\mathrm{i}H_1 t}$ can be efficiently performed by diagonalizing H_1 in the Fourier space, and $e^{\mathrm{i}H_2 t}$ can be efficiently performed since $V(\mathbf{r})$ is diagonal in the real space. ◇

5.2. Commutator type error bound

In this section we try to refine the error bounds in Eq. (5.3) by evaluating the preconstant explicitly. For simplicity we only focus on the first order Trotter formula. The Trotter propagator $\widetilde{U}(t) = e^{-\mathrm{i}tH_1}e^{-\mathrm{i}tH_2}$ satisfies the equation

$$\begin{aligned} \mathrm{i}\partial_t \widetilde{U}(t) &= H_1 e^{-\mathrm{i}tH_1}e^{-\mathrm{i}tH_2} + e^{-\mathrm{i}tH_1}H_2 e^{-\mathrm{i}tH_2} \\ &= (H_1+H_2)e^{-\mathrm{i}tH_1}e^{-\mathrm{i}tH_2} + e^{-\mathrm{i}tH_1}H_2 e^{-\mathrm{i}tH_2} - H_2 e^{-\mathrm{i}tH_1}e^{-\mathrm{i}tH_2} \\ &= H\widetilde{U}(t) + [e^{-\mathrm{i}tH_1}, H_2]e^{-\mathrm{i}tH_2}, \end{aligned} \tag{5.11}$$

with initial condition $\widetilde{U}(0) = I$. By Duhamel's principle, and let $U(t) = e^{-\mathrm{i}tH}$, we have

$$\widetilde{U}(t) = U(t) - \mathrm{i}\int_0^t e^{-\mathrm{i}H(t-s)}[e^{-\mathrm{i}sH_1}, H_2]e^{-\mathrm{i}sH_2}\,\mathrm{d}s. \tag{5.12}$$

So we have

$$\left\| \widetilde{U}(t) - U(t) \right\| \le \int_0^t \left\| [e^{-\mathrm{i}sH_1}, H_2] \right\| \mathrm{d}s. \tag{5.13}$$

Now consider $G(t) = [e^{-\mathrm{i}tH_1}, H_2]e^{\mathrm{i}tH_1} = e^{-\mathrm{i}tH_1}H_2e^{\mathrm{i}tH_1} - H_2$, which satisfies $G(0) = 0$ and

$$\mathrm{i}\partial_t G(t) = e^{-\mathrm{i}tH_1}[H_1, H_2]e^{+\mathrm{i}tH_1}. \tag{5.14}$$

Hence

$$\left\|[e^{-\mathrm{i}tH_1}, H_2]\right\| = \|G(t)\| \leq t \left\|[H_1, H_2]\right\|. \tag{5.15}$$

Plugging this back to Eq. (5.13), we have

$$\left\|\widetilde{U}(t) - U(t)\right\| \leq \int_0^t s \left\|[H_1, H_2]\right\| \mathrm{d}s \leq \frac{t^2}{2} \left\|[H_1, H_2]\right\| \leq t^2\nu^2. \tag{5.16}$$

In the last equality, we have used the relation $\|[H_1, H_2]\| \leq 2\nu^2$ with $\nu = \max\{\|H_1\|, \|H_2\|\}$. Therefore Eq. (5.3) can be replaced by a sharper inequality

$$\left\|e^{-\mathrm{i}\Delta tH} - e^{-\mathrm{i}\Delta tH_1}e^{-\mathrm{i}\Delta tH_2}\right\| \leq \frac{\Delta t^2}{2} \left\|[H_1, H_2]\right\| \leq (\Delta t)^2\nu^2. \tag{5.17}$$

Here the first inequality is called the commutator norm error estimate, and the second inequality the operator norm error estimate.

For the transverse field Ising model with nearest neighbor interaction, we have $\|H_1\|, \|H_2\| = \mathcal{O}(n)$, and hence $\nu^2 = \mathcal{O}(n^2)$. On the other hand, since $[Z_iZ_j, X_k] \neq 0$ only if $k = i$ or $k = j$, the commutator bound satisfies $\|[H_1, H_2]\| = \mathcal{O}(n)$. Therefore to reach precision ϵ, the scaling of the total number of time steps L with respect to the system size is $\mathcal{O}(n^2/\epsilon)$ according to the estimate based on the operator norm, but is only $\mathcal{O}(n/\epsilon)$ according to that based on the commutator norm.

For the particle in a potential, for simplicity consider $d = 1$ and the domain $\Omega = [0, 1]$ is discretized using a uniform grid of size N. For smooth and bounded potential, we have $\|H_1\| = \mathcal{O}(N^2)$, and $\|V\| = \mathcal{O}(1)$. Therefore the operator norm bound gives $\nu^2 = \mathcal{O}(N^4)$. This is too pessimistic. Reexamining the second inequality of Eq. (5.17) shows that in this case, the error bound should be $\mathcal{O}((\Delta t)^2\nu)$ instead of $(\Delta t)^2\nu^2$. So according to the operator norm error estimate, we have $L = \mathcal{O}(N^2/\epsilon)$. On the other hand, in the continuous space, for any smooth function $\psi(r)$, we have

$$[H_1, H_2]\psi = \left[-\frac{\mathrm{d}^2}{\mathrm{d}r^2}, V\right]\psi = -V''\psi - V'\psi'. \tag{5.18}$$

So

$$\|[H_1, H_2]\psi\| \leq \|V''\| + \|V'\| \|\psi'\| = \mathcal{O}(N). \tag{5.19}$$

Here we have used that $\|V'\| = \|V''\| = \mathcal{O}(1)$, and $\|\psi'\| = \mathcal{O}(N)$ in the worst case scenario. Therefore $\|[H_1, H_2]\| = \mathcal{O}(N)$, and we obtain a significantly improved estimate $L = \mathcal{O}(N/\epsilon)$ according to the commutator norm.

The commutator scaling of the Trotter error is an important feature of the method. We refer readers to [JL00] for analysis of the second order Trotter method, and [Tha08, CST+21] for the analysis of the commutator scaling of high order Trotter methods.

Remark 5.3 (Vector norm bound)**.** The Hamiltonian simulation problem of interest in practice often concerns the solution with particular types of initial conditions, instead of arbitrary initial conditions. Therefore the operator norm bound in Eq. (5.17) can still be too loose. Taking the initial condition into account, we readily obtain

$$\left\|e^{-\mathrm{i}\Delta tH}\psi(0) - e^{-\mathrm{i}\Delta tH_1}e^{-\mathrm{i}\Delta tH_2}\psi(0)\right\| \leq \frac{\Delta t^2}{2} \max_{0\leq s\leq \Delta t} \left\|[H_1, H_2]\psi(s)\right\|. \tag{5.20}$$

For the example of the particle in a potential, we have

$$\max_{0\le s\le \Delta t} \|[H_1, H_2]\psi(s)\| \le \|V''\| + \|V'\| \max_{0\le s\le \Delta t} \|\psi'(s)\|\,. \tag{5.21}$$

Therefore if we are given the *a priori* knowledge that $\max_{0\le s\le t} \|\psi'(s)\| = \mathcal{O}(1)$, we may even have $L = \mathcal{O}(\epsilon^{-1})$, i.e., the number of time steps is independent of N. ◇

Exercise 5.1. Consider the Hamiltonian simulation problem for $H = H_1 + H_2 + H_3$. Show that the first order Trotter formula

$$\widetilde{U}(t) = e^{-\mathrm{i}tH_1} e^{-\mathrm{i}tH_2} e^{-\mathrm{i}tH_3}$$

has a commutator type error bound.

Exercise 5.2. Consider the time-dependent Hamiltonian simulation problem for the following controlled Hamiltonian

$$H(t) = a(t)H_1 + b(t)H_2,$$

where $a(t)$ and $b(t)$ are smooth functions bounded together with all derivatives. We focus on the following Trotter type splitting, defined as

$$\widetilde{U}(t) := \widetilde{U}(t_n, t_{n-1}) \cdots \widetilde{U}(t_1, t_0), \quad \widetilde{U}(t_{j+1}, t_j) = e^{-\mathrm{i}\Delta t a(t_j) H_1} e^{-\mathrm{i}\Delta t b(t_j) H_2},$$

where the intervals $[t_j, t_{j+1}]$ are equidistant and of length Δt on the interval $[0, t]$ with $t_n = t$. Show that this method has first-order accuracy, but does *not* exhibit a commutator type error bound in general.

CHAPTER 6

Block encoding

In order to perform matrix computations, we must first address the problem of the *input model*: how to get access to information in a matrix $A \in \mathbb{C}^{N\times N}$ ($N = 2^n$) which is generally a non-unitary matrix, into the quantum computer? One possible input model is given via the unitary $e^{\mathrm{i}\tau A}$ (if A is not Hermitian, in some scenarios we can consider its Hermitian version via the dilation method). This is particularly useful when $e^{\mathrm{i}\tau A}$ can be constructed using simple circuits, e.g. Trotter splitting.

A more general input model, as will be discussed in this chapter, is called "block encoding". Of course, if A is a dense matrix without obvious structures, any input model will be very expensive (e.g. exponential in n) to implement. Therefore a commonly assumed input model is s-sparse, i.e., there are at most s nonzero entries in each row / column of the matrix. Furthermore, we have an efficient procedure to get access to the location, as well as the value of the nonzero entries. This in general can again be a difficult task given that the number of nonzero entries can still be exponential in n for a sparse matrix. Some dense matrices may also be efficiently block encoded on quantum computers. This chapter will illustrate the block encoding procedure for via a number of detailed examples.

6.1. Query model for matrix entries

The query model for sparse matrices is based on certain quantum oracles. In some scenarios, these quantum oracles can be implemented all the way to the elementary gate level.

Throughout the discussion we assume A is an n-qubit, square matrix, and

$$\|A\|_{\max} := \max_{ij} |A_{ij}| < 1. \tag{6.1}$$

If the $\|A\|_{\max} \geq 1$, we can simply consider the rescaled matrix $\widetilde{A}/\alpha$ for some $\alpha > \|A\|_{\max}$.

To query the entries of a matrix, the desired oracle takes the following general form

$$O_A |0\rangle |i\rangle |j\rangle = \left(A_{ij} |0\rangle + \sqrt{1 - |A_{ij}|^2} |1\rangle \right) |i\rangle |j\rangle . \tag{6.2}$$

In other words, given $i, j \in [N]$ and a signal qubit 0, O_A performs a controlled rotation (controlling on i, j) of the signal qubit, which encodes the information in terms of amplitude of $|0\rangle$.

However, the classical information in A is usually not stored natively in terms of such an oracle O_A. Sometimes it is more natural to assume that there is an oracle

$$\widetilde{O}_A |0^{d'}\rangle |i\rangle |j\rangle = |\widetilde{A}_{ij}\rangle |i\rangle |j\rangle , \tag{6.3}$$

where $\widetilde{A}_{ij}$ is a d'-bit fixed point representation of A_{ij}, and the value of $\widetilde{A}_{ij}$ is either computed on-the-fly with a quantum computer, or obtained through an external database. In either case, the implementation of $\widetilde{O}_A$ may be challenging, and we will only consider the query complexity with respect to this oracle.

Using *classical arithmetic operations*, we can convert this oracle into an oracle

$$O'_A |0^d\rangle |i\rangle |j\rangle = |\widetilde{\theta}_{ij}\rangle |i\rangle |j\rangle , \tag{6.4}$$

where $0 \leq \widetilde{\theta}_{ij} < 1$, and $\widetilde{\theta}_{ij}$ is a d-bit representation of $\theta_{ij} = \arccos(A_{ij})/\pi$. This step may require some additional work registers not shown here.

Now using the controlled rotation in Proposition 4.7, the information of $\widetilde{A}_{ij}$, $\widetilde{\theta}_{ij}$ has now been transferred to the phase of the signal qubit. We should then perform uncomputation and free the work register storing such intermediate information $\widetilde{A}_{ij}$, $\widetilde{\theta}_{ij}$. The procedure is as follows

$$\begin{aligned} |0\rangle \underbrace{|0^d\rangle}_{\text{work register}} |i\rangle |j\rangle &\xrightarrow{O'_A} |0\rangle |\widetilde{\theta}_{ij}\rangle |i\rangle |j\rangle \\ &\xrightarrow{\text{CR}} \left(A_{ij} |0\rangle + \sqrt{1 - |A_{ij}|^2} |1\rangle \right) |\widetilde{\theta}_{ij}\rangle |i\rangle |j\rangle \\ &\xrightarrow{(O'_A)^{-1}} \left(A_{ij} |0\rangle + \sqrt{1 - |A_{ij}|^2} |1\rangle \right) |0^d\rangle |i\rangle |j\rangle \end{aligned} \tag{6.5}$$

From now on, we will always assume that the matrix entries of A can be queried using the phase oracle O_A or its variants.

6.2. Block encoding

The simplest example of block encoding is the following: assume we can find a $(n+1)$-qubit unitary matrix U (i.e., $U \in \mathbb{C}^{2N \times 2N}$) such that

$$U_A = \begin{pmatrix} A & * \\ * & * \end{pmatrix}$$

where $*$ means that the corresponding matrix entries are irrelevant, then for any n-qubit quantum state $|b\rangle$, we can consider the state

$$|0, b\rangle = |0\rangle |b\rangle = \begin{pmatrix} b \\ 0 \end{pmatrix}, \tag{6.6}$$

and

$$U_A |0, b\rangle = \begin{pmatrix} Ab \\ * \end{pmatrix} =: |0\rangle A |b\rangle + |\perp\rangle . \tag{6.7}$$

Here the (unnormalized) state $|\perp\rangle$ can be written as $|1\rangle |\psi\rangle$ for some (unnormalized) state $|\psi\rangle$, that is irrelevant to the computation of $A |b\rangle$. In particular, it satisfies the orthogonality relation.

$$(\langle 0| \otimes I_n) |\perp\rangle = 0. \tag{6.8}$$

In order to obtain $A |b\rangle$, we need to *measure* the qubit 0 and only keep the state if it returns 0. This can be summarized into the following quantum circuit:

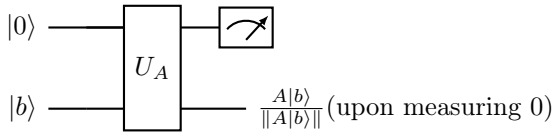


FIGURE 6.1. Circuit for block encoding of A using one ancilla qubit.

Note that the output state is normalized after the measurement takes place. The success probability of obtaining 0 from the measurement can be computed as

$$p(0) = \|A\,|b\rangle\|^2 = \langle b|A^\dagger A|b\rangle\,. \tag{6.9}$$

So the missing information of norm $\|A\,|b\rangle\|$ can be recovered via the success probability $p(0)$ if needed. We find that the success probability is only determined by $A, |b\rangle$, and is independent of other irrelevant components of U_A.

Note that we may not need to restrict the matrix U_A to be a $(n+1)$-qubit matrix. If we can find any $(n+m)$-qubit matrix U_A so that

$$U_A = \begin{pmatrix} A & * & \cdots & * \\ * & * & \cdots & * \\ \vdots & & \vdots & \\ * & * & \cdots & * \end{pmatrix} \tag{6.10}$$

Here each $*$ stands for an n-qubit matrix, and there are 2^m block rows / columns in U_A. The relation above can be written compactly using the braket notation as

$$A = (\langle 0^m| \otimes I_n)\, U_A\, (|0^m\rangle \otimes I_n) \tag{6.11}$$

A necessary condition for the existence of U_A is that $\|A\| \leq 1$. (Note: $\|A\|_{\max} \leq 1$ does not guarantee that $\|A\| \leq 1$, see Exercise 6.2). However, if we can find sufficiently large α and U_A so that

$$A/\alpha = (\langle 0^m| \otimes I_n)\, U_A\, (|0^m\rangle \otimes I_n)\,. \tag{6.12}$$

Measuring the m ancilla qubits and all m-qubits return 0, we still obtain the normalized state $\frac{A|b\rangle}{\||A|b\rangle\|}$. The number α is hidden in the success probability:

$$p(0^m) = \frac{1}{\alpha^2}\,\|A\,|b\rangle\|^2 = \frac{1}{\alpha^2}\,\langle b|A^\dagger A|b\rangle\,. \tag{6.13}$$

So if α is chosen to be too large, the probability of obtaining all 0's from the measurement can be vanishingly small.

Finally, it can be difficult to find U_A to block encode A exactly. This is not a problem, since it is sufficient if we can find U_A to block encode A up to some error ϵ. We are now ready to give the definition of block encoding in Definition 6.1.

Definition 6.1 (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*

$$\|A - \alpha\,(\langle 0^m| \otimes I_n)\, U_A\, (|0^m\rangle \otimes I_n)\,\| \leq \epsilon, \tag{6.14}$$

then U_A is called an (α, m, ϵ)-block-encoding of A. When the block encoding is exact with $\epsilon = 0$, U_A is called an (α, m)-block-encoding of A. The set of all (α, m, ϵ)-block-encoding of A is denoted by $\mathrm{BE}_{\alpha,m}(A, \epsilon)$, and we define $\mathrm{BE}_{\alpha,m}(A) = \mathrm{BE}(A, 0)$.

Assume we know each matrix element of the n-qubit matrix A_{ij}, and we are given an $(n+m)$-qubit unitary U_A. In order to verify that $U_A \in \mathrm{BE}_{1,m}(A)$, we only need to verify that

$$\langle 0^m, i|U_A|0^m, j\rangle = A_{ij}, \tag{6.15}$$

and U_A applied to any vector $|0^m, b\rangle$ can be obtained via the superposition principle.

Therefore we may first evaluate the state $U_A|0^m, j\rangle$, and perform inner product with $|0^m, i\rangle$ and verify the resulting the inner product is A_{ij}. We will also use the following technique frequently. Assume $U_A = U_B U_C$, and then

$$\langle 0^m, i|U_A|0^m, j\rangle = \langle 0^m, i|U_B U_C|0^m, j\rangle = (U_B^\dagger|0^m, i\rangle)^\dagger (U_C|0^m, j\rangle). \tag{6.16}$$

So we can evaluate the states $U_B^\dagger|0^m, i\rangle, U_C|0^m, j\rangle$ independently, and then verify the inner product is A_{ij}. Such a calculation amounts to running the circuit Fig. 6.2, and if the ancilla qubits are measured to be 0^m, the system qubits return the normalized state $\sum_i A_{ij}|i\rangle / \|\sum_i A_{ij}|i\rangle\|$.

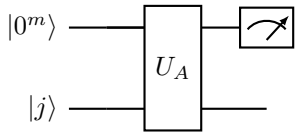


FIGURE 6.2. Circuit for general block encoding of A.

Example 6.2 ($(1,1)$-block-encoding is general)**.** For any n-qubit matrix A with $\|A\|_2 \le 1$, the singular value decomposition (SVD) of A is denoted by $W\Sigma V^\dagger$, where all singular values in the diagonal matrix Σ belong to $[0,1]$. Then we may construct an $(n+1)$-qubit unitary matrix

$$\begin{aligned} U_A &:= \begin{pmatrix} W & 0 \\ 0 & I_n \end{pmatrix} \begin{pmatrix} \Sigma & \sqrt{I_n - \Sigma^2} \\ \sqrt{I_n - \Sigma^2} & -\Sigma \end{pmatrix} \begin{pmatrix} V^\dagger & 0 \\ 0 & I_n \end{pmatrix} \\ &= \begin{pmatrix} A & W\sqrt{I_n - \Sigma^2} \\ \sqrt{I_n - \Sigma^2}V^\dagger & -\Sigma \end{pmatrix} \end{aligned} \tag{6.17}$$

which is a $(1,1)$-block-encoding of A. ◇

Example 6.3 (Random circuit block encoded matrix)**.** In some scenarios, we may want to construct a pseudo-random non-unitary matrix on quantum computers. Note that it would be highly inefficient if we first generate a dense pseudo-random matrix A classically and then feed it into the quantum computer using e.g. quantum random-access memory (QRAM). Instead we would like to work with matrices that are *inherently easy* to generate on quantum computers. This inspires the random circuit based block encoding matrix (RACBEM) model [DL21]. 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.

Example 6.2 shows that in principle, any matrix A with $\|A\|_2 \le 1$ can be accessed via a $(1,1,0)$-block-encoding. In other words, A can be block encoded by an $(n+1)$-qubit random unitary U_A,

and U_A can be constructed using only basic one-qubit unitaries and CNOT gates. The layout of the two-qubit operations can be designed to be compatible with the coupling map of the hardware. A cartoon is shown in Example 6.3, and an example is given in Fig. 6.4.

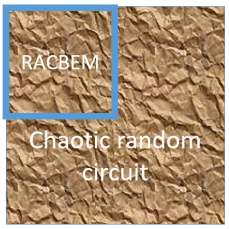
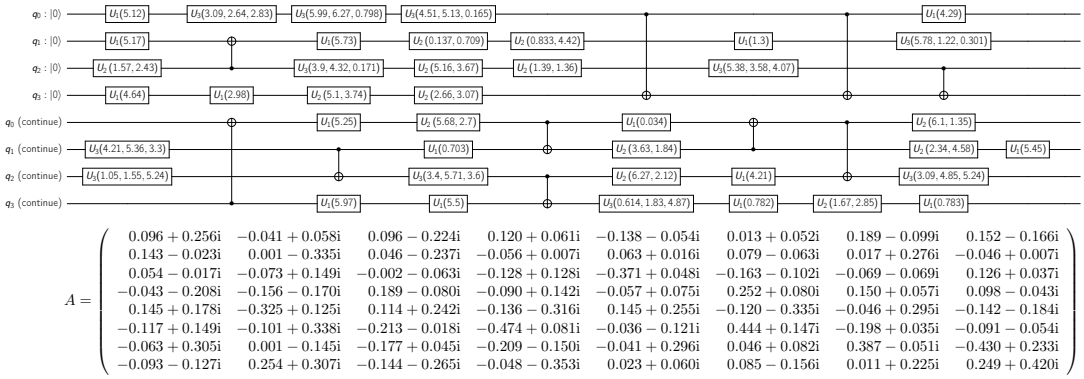


FIGURE 6.3. A cartoon illustration of the RACBEM model.



$$A = \begin{pmatrix} 0.096+0.256i & -0.041+0.058i & 0.096-0.224i & 0.120+0.061i & -0.138-0.054i & 0.013+0.052i & 0.189-0.099i & 0.152-0.166i \\ 0.143-0.023i & 0.001-0.335i & 0.046-0.237i & -0.056+0.007i & 0.063+0.016i & 0.079-0.063i & 0.017+0.276i & -0.046+0.007i \\ 0.054-0.017i & -0.073+0.149i & -0.002-0.063i & -0.128+0.128i & -0.371+0.048i & -0.163-0.102i & -0.069-0.069i & 0.126+0.037i \\ -0.043-0.208i & -0.156-0.170i & 0.189-0.080i & -0.090+0.142i & -0.057+0.075i & 0.252+0.080i & 0.150+0.057i & 0.098-0.043i \\ 0.145+0.178i & -0.325+0.125i & 0.114+0.242i & -0.136-0.316i & 0.145+0.255i & -0.120-0.335i & -0.046+0.295i & -0.142-0.184i \\ -0.117+0.149i & -0.101+0.338i & -0.213-0.018i & -0.474+0.081i & -0.036-0.121i & 0.444+0.147i & -0.198+0.035i & -0.091-0.054i \\ -0.063+0.305i & 0.001-0.145i & -0.177+0.045i & -0.209-0.150i & -0.041+0.296i & 0.046+0.082i & 0.387-0.051i & -0.430+0.233i \\ -0.093-0.127i & 0.254+0.307i & -0.144-0.265i & -0.048-0.353i & 0.023+0.060i & 0.085-0.156i & 0.011+0.225i & 0.249+0.420i \end{pmatrix}$$

FIGURE 6.4. A RACBEM circuit constructed using the basic gate set $\{\mathrm{U}_1, \mathrm{U}_2, \mathrm{U}_3, \mathrm{CNOT}\}$. The circuit at the bottom is a continuation of the top circuit. A is the 3-qubit matrix block encoded as the upper-left block, namely, identifying q_0 as the block encoding qubit.

◇

Example 6.4 (Block encoding of a diagonal matrix)**.** As a special case, let us consider the block encoding of a diagonal matrix. Since the row and column indices are the same, we may simplify the oracle Eq. (6.2) into

$$O_A |0\rangle |i\rangle = \left(A_{ii} |0\rangle + \sqrt{1 - |A_{ii}|^2} |1\rangle \right) |i\rangle . \tag{6.18}$$

In the case when the oracle $\widetilde{O}_A$ is used, we may assume accordingly

$$\widetilde{O}_A |0^{d'}\rangle |i\rangle = |A_{ii}\rangle |i\rangle . \tag{6.19}$$

Let $U_A = O_A$. Direct calculation shows that for any $i, j \in [N]$,

$$\langle 0| \langle i| U_A |0\rangle |j\rangle = A_{ii}\delta_{ij}. \tag{6.20}$$

This proves that $U_A \in \mathrm{BE}_{1,1}(A)$, i.e., U_A is a $(1,1)$-block-encoding of the diagonal matrix A. ◇

6.3. Block encoding of s-sparse matrices

We now give a few examples of block encodings of more general sparse matrices. We start from a 1-sparse matrix, i.e., there is only one nonzero entry in each row or column of the matrix. This means that for each $j \in [N]$, there is a unique $c(j) \in [N]$ such that $A_{c(j),j} \neq 0$, and the mapping c is a permutation. Then there exists a unitary O_c such that

$$O_c |j\rangle = |c(j)\rangle . \tag{6.21}$$

The implementation of O_c may require the usage of some work registers that are omitted here. We also have

$$O_c^\dagger |c(j)\rangle = |j\rangle . \tag{6.22}$$

We assume the matrix entry $A_{c(j),j}$ can be queried via

$$O_A |0\rangle |j\rangle = \left(A_{c(j),j} |0\rangle + \sqrt{1 - \left|A_{c(j),j}\right|^2} |1\rangle \right) |j\rangle . \tag{6.23}$$

Now we construct $U_A = (I \otimes O_c)O_A$, and compute

$$\langle 0| \langle i| U_A |0\rangle |j\rangle = \langle 0| \langle i| \left(A_{c(j),j} |0\rangle + \sqrt{1 - \left|A_{c(j),j}\right|^2} |1\rangle \right) |c(j)\rangle = A_{c(j),j}\delta_{i,c(j)}. \tag{6.24}$$

This proves that $U_A \in \mathrm{BE}_{1,1}(A)$.

For a more general s-sparse matrix, WLOG we assume each row and column has exactly s nonzero entries (otherwise we can always treat some zero entries as nonzeros). For each column j, the row index for the ℓ-th nonzero entry is denoted by $c(j,\ell) \equiv c_{j,\ell}$. For simplicity, we assume that there exists a unitary O_c such that

$$O_c |\ell\rangle |j\rangle = |\ell\rangle |c(j,\ell)\rangle . \tag{6.25}$$

Here we assume $s = 2^{\mathfrak{s}}$ and the first register is an $\mathfrak{s}$-qubit register. A necessary condition for this query model is that O_c is reversible, i.e., we can have $O_c^\dagger |\ell\rangle |c(j,\ell)\rangle = |\ell\rangle |j\rangle$. This means that for each row index $i = c(j,\ell)$, we can recover the column index j given the value of ℓ. This can be satisfied e.g.

$$c(j,\ell) = j + \ell - \ell_0 \pmod N, \tag{6.26}$$

where ℓ_0 is a fixed number. This corresponds to a banded matrix. This assumption is of course somewhat restrictive. We shall discuss more general query models in Section 6.5.

Corresponding to Eq. (6.25), the matrix entries can be queried via

$$O_A |0\rangle |\ell\rangle |j\rangle = \left(A_{c(j,\ell),j} |0\rangle + \sqrt{1 - \left|A_{c(j,\ell),j}\right|^2} |1\rangle \right) |\ell\rangle |j\rangle . \tag{6.27}$$

In order to construct a unitary that encodes all row indices at the same time, we define $D = H^{\otimes \mathfrak{s}}$ (sometimes called a diffusion operator, which is a term originated from Grover's search) satisfying

$$D |0^{\mathfrak{s}}\rangle = \frac{1}{\sqrt{s}} \sum_{\ell \in [s]} |\ell\rangle . \tag{6.28}$$

Consider U_A given by the circuit in Fig. 6.5. The measurement means that to obtain $A|b\rangle$, the ancilla register should all return the value 0.

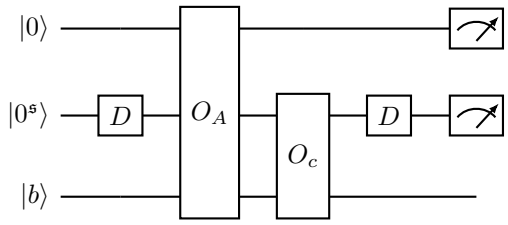


FIGURE 6.5. Quantum circuit for block encoding an s-sparse matrix.

Proposition 6.5. *The circuit in Fig. 6.5 defines* $U_A \in \mathrm{BE}_{s,\mathfrak{s}+1}(A)$.

PROOF. We call $|0\rangle |0^{\mathfrak{s}}\rangle |j\rangle$ the source state, and $|0\rangle |0^{\mathfrak{s}}\rangle |i\rangle$ the target state. In order to compute the inner product $\langle 0| \langle 0^{\mathfrak{s}}| \langle i| U_A |0\rangle |0^{\mathfrak{s}}\rangle |j\rangle$, we apply D, O_A, O_c to the source state accordingly as

$$\begin{aligned} |0\rangle |0^{\mathfrak{s}}\rangle |j\rangle &\xrightarrow{D} \frac{1}{\sqrt{s}} \sum_{\ell\in[s]} |0\rangle |\ell\rangle |j\rangle \\ &\xrightarrow{O_A} \frac{1}{\sqrt{s}} \sum_{\ell\in[s]} \left(A_{c(j,\ell),j} |0\rangle + \sqrt{1 - \left|A_{c(j,\ell),j}\right|^2} |1\rangle \right) |\ell\rangle |j\rangle \\ &\xrightarrow{O_c} \frac{1}{\sqrt{s}} \sum_{\ell\in[s]} \left(A_{c(j,\ell),j} |0\rangle + \sqrt{1 - \left|A_{c(j,\ell),j}\right|^2} |1\rangle \right) |\ell\rangle |c(j,\ell)\rangle . \end{aligned} \tag{6.29}$$

Since we are only interested in the final state when all ancilla qubits are the 0 state, we may apply D to target state $|0\rangle |0^{\mathfrak{s}}\rangle |i\rangle$ as (note that D is Hermitian)

$$|0\rangle |0^{\mathfrak{s}}\rangle |i\rangle \xrightarrow{D} \frac{1}{\sqrt{s}} \sum_{\ell'\in[s]} |0\rangle |\ell'\rangle |i\rangle . \tag{6.30}$$

Hence the inner product

$$\langle 0| \langle 0^{\mathfrak{s}}| \langle i| U_A |0\rangle |0^{\mathfrak{s}}\rangle |j\rangle = \frac{1}{s} \sum_{\ell} A_{c(j,\ell),j} \delta_{i,c(j,\ell)} = \frac{1}{s} A_{ij}. \tag{6.31}$$

□

6.4. Hermitian block encoding

So far we have considered general s-sparse matrices. Note that if A is a Hermitian matrix, its (α, m, ϵ)-block-encoding U_A does not need to be Hermitian. Even if $\epsilon = 0$, we only have that the upper-left n-qubit block of U_A is Hermitian. For instance, even the block encoding of a Hermitian, diagonal matrix in Example 6.4 may not be Hermitian (exercise). On the other hand, there are indeed cases when $U_A = U_A^\dagger$ is indeed a Hermitian matrix, and hence the definition:

Definition 6.6 (Hermitian block encoding)**.** *Let U_A be an (α, m, ϵ)-block-encoding of A. If U_A is also Hermitian, then it is called an (α, m, ϵ)-Hermitian-block-encoding of A. When $\epsilon = 0$, it is called an (α, m)-Hermitian-block-encoding. The set of all (α, m, ϵ)-Hermitian-block-encoding of A is denoted by $\mathrm{HBE}_{\alpha,m}(A, \epsilon)$, and we define $\mathrm{HBE}_{\alpha,m}(A) = \mathrm{HBE}(A, 0)$.*

The Hermitian block encoding provides the simplest scenario of the qubitization process in Section 7.1.

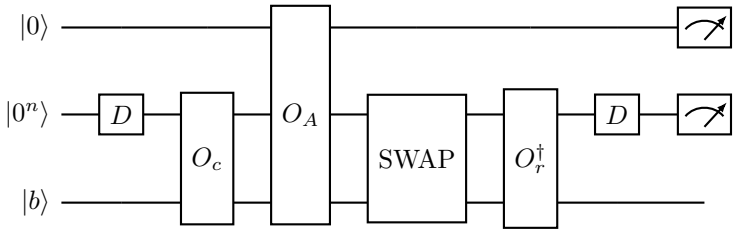


FIGURE 6.6. Quantum circuit for block encoding of general sparse matrices.

6.5. Query models for general sparse matrices*

If we query the oracle (6.25), the assumption that for each ℓ the value of $c(j,\ell)$ is unique for all j seems unnatural for constructing general sparse matrices. So we consider an altnerative method for construct the block encoding of a general sparse matrix as below.

Again WLOG we assume that each row / column has at most $s = 2^{\mathfrak{s}}$ nonzero entries, and that we have access to the following two $(2n)$-qubit oracles

$$\begin{aligned} O_r |\ell\rangle |i\rangle &= |r(i,\ell)\rangle |i\rangle , \\ O_c |\ell\rangle |j\rangle &= |c(j,\ell)\rangle |j\rangle . \end{aligned} \tag{6.32}$$

Here $r(i,\ell), c(j,\ell)$ gives the ℓ-th nonzero entry in the i-th row and j-th column, respectively. It should be noted that although the index $\ell \in [s]$, we should expand it into an n-qubit state (e.g. let ℓ take the last $\mathfrak{s}$ qubits of the n-qubit register following the binary representation of integers). The reason for such an expansion, and that we need two oracles O_r, O_c will be seen shortly.

Similar to the discussion before, we need a diffusion operator satisfying

$$D |0^n\rangle = \frac{1}{\sqrt{s}} \sum_{\ell \in [s]} |\ell\rangle . \tag{6.33}$$

This can be implemented using Hadamard gates as

$$D = I_{n-\mathfrak{s}} \otimes H^{\otimes \mathfrak{s}}. \tag{6.34}$$

We assume that the matrix entries are queried using the following oracle using controlled rotations

$$O_A |0\rangle |i\rangle |j\rangle = \left(A_{ij} |0\rangle + \sqrt{1 - |A_{ij}|^2} \right) |i\rangle |j\rangle , \tag{6.35}$$

where the rotation is controlled by both row and column indices. However, if $A_{ij} = 0$ for some i, j, the rotation can be arbitrary, as there will be no contribution due to the usage of O_r, O_c.

Proposition 6.7. *Fig. 6.6 defines* $U_A \in \mathrm{BE}_{s,n+1}(A)$.

PROOF. We apply the first four gate sets to the source state

$$\begin{aligned}
&|0\rangle|0^n\rangle|j\rangle \xrightarrow{D}\xrightarrow{O_c} \\
&\xrightarrow{O_A}\frac{1}{\sqrt{s}}\sum_{\ell\in[s]}\left(A_{c(j,\ell),j}|0\rangle+\sqrt{1-\left|A_{c(j,\ell),j}\right|^2}|1\rangle\right)|c(j,\ell)\rangle|j\rangle \\
&\xrightarrow{\text{SWAP}}\frac{1}{\sqrt{s}}\sum_{\ell\in[s]}\left(A_{c(j,\ell),j}|0\rangle+\sqrt{1-\left|A_{c(j,\ell),j}\right|^2}|1\rangle\right)|j\rangle|c(j,\ell)\rangle .
\end{aligned} \tag{6.36}$$

We then apply D and O_r to the target state

$$|0\rangle|0^n\rangle|i\rangle \xrightarrow{D}\xrightarrow{O_r}\frac{1}{\sqrt{s}}\sum_{\ell'\in[s]}|0\rangle|r(i,\ell')\rangle|i\rangle . \tag{6.37}$$

Then the inner product gives

$$\begin{aligned}
\langle 0|\langle 0^n|\langle i|U_A|0\rangle|0^n\rangle|j\rangle =&\frac{1}{s}\sum_{\ell,\ell'}A_{c(j,\ell),j}\delta_{i,c(j,\ell)}\delta_{r(j,\ell'),j} \\
=&\frac{1}{s}\sum_{\ell}A_{c(j,\ell),j}\delta_{i,c(j,\ell)}=\frac{1}{s}A_{ij}.
\end{aligned} \tag{6.38}$$

Here we have used that there exists a unique ℓ such that $i = c(j,\ell)$, and a unique ℓ' such that $j = r(i,\ell')$. □

We remark that the quantum circuit in Fig. 6.6 is essentially the construction in [GSLW18, Lemma 48], which gives a $(s, n+3)$-block-encoding. The construction above slightly simplifies the procedure and saves two extra qubits (used to mark whether $\ell \geq s$).

Next we consider the Hermitian block encoding of a s-sparse Hermitian matrix. Since A is Hermitian, we only need one oracle to query the location of the nonzero entries

$$O_c|\ell\rangle|j\rangle = |c(j,\ell)\rangle|j\rangle . \tag{6.39}$$

Here $c(j,\ell)$ gives the ℓ-th nonzero entry in the j-th column. It can also be interpreted as the ℓ-th nonzero entry in the i-th column. Again the first register needs to be interpreted as an n-qubit register. The diffusion operator is the same as in Eq. (6.34).

Unlike all discussions before, we introduce *two* signal qubits, and a quantum state in the computational basis takes the form $|a\rangle|i\rangle|b\rangle|j\rangle$, where $a, b \in \{0,1\}, i, j \in [N]$. In other words, we may view $|a\rangle|i\rangle$ as the first register, and $|b\rangle|j\rangle$ as the second register. The $(n+1)$-qubit SWAP gate is defined as

$$\text{SWAP}|a\rangle|i\rangle|b\rangle|j\rangle = |b\rangle|j\rangle|a\rangle|i\rangle . \tag{6.40}$$

To query matrix entries, we need access to the square root of A_{ij} as (note that act on the second single-qubit register)

$$O_A|i\rangle|0\rangle|j\rangle = |i\rangle\left(\sqrt{A_{ij}}|0\rangle+\sqrt{1-|A_{ij}|}|1\rangle\right)|j\rangle . \tag{6.41}$$

The square root operation is well defined if $A_{ij} \geq 0$ for all entries. If A has negative (or complex) entries, we first write $A_{ij} = |A_{ij}|e^{\mathrm{i}\theta_{ij}}, \theta_{ij} \in [0, 2\pi)$, and the square root is uniquely defined as $\sqrt{A_{ij}} = \sqrt{|A_{ij}|}e^{\mathrm{i}\theta_{ij}/2}$.

Proposition 6.8. *Fig. 6.7 defines* $U_A \in \mathrm{HBE}_{s,n+2}(A)$.

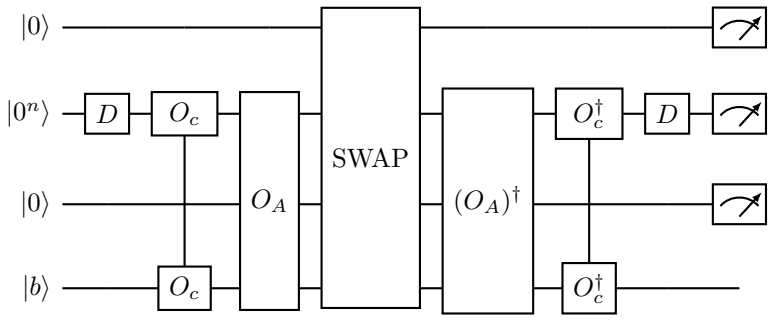


FIGURE 6.7. Quantum circuit for Hermitian block encoding of a general Hermitian matrix

PROOF. Apply the first four gate sets to the source state gives

$$\begin{aligned}
&|0\rangle|0^n\rangle|0\rangle|j\rangle \xrightarrow{D}\xrightarrow{O_c}\\
&\xrightarrow{O_A}\frac{1}{\sqrt{s}}\sum_{\ell\in[s]}|0\rangle|c(j,\ell)\rangle\left(\sqrt{A_{c(j,\ell),j}}|0\rangle+\sqrt{1-\left|A_{c(j,\ell),j}\right|}|1\rangle\right)|j\rangle\\
&\xrightarrow{\text{SWAP}}\frac{1}{\sqrt{s}}\sum_{\ell\in[s]}\left(\sqrt{A_{c(j,\ell),j}}|0\rangle+\sqrt{1-\left|A_{c(j,\ell),j}\right|}|1\rangle\right)|j\rangle|0\rangle|c(j,\ell)\rangle
\end{aligned}\tag{6.42}$$

Apply the last three gate sets to the target state

$$\begin{aligned}
&|0\rangle|0^n\rangle|0\rangle|i\rangle \xrightarrow{D}\xrightarrow{O_c}\\
&\xrightarrow{O_A}\frac{1}{\sqrt{s}}\sum_{\ell'\in[s]}|0\rangle|c(i,\ell')\rangle\left(\sqrt{A_{c(i,\ell'),i}}|0\rangle+\sqrt{1-\left|A_{c(i,\ell'),i}\right|}|1\rangle\right)|i\rangle
\end{aligned}\tag{6.43}$$

Finally, take the inner product as

$$\begin{aligned}
&\langle 0|\langle 0^n|\langle 0|\langle i|U_A|0\rangle|0^n\rangle|0\rangle|j\rangle\\
=&\frac{1}{s}\sum_{\ell,\ell'}\sqrt{A_{c(j,\ell),j}}\sqrt{A^*_{c(i,\ell'),i}}\delta_{i,c(j,\ell)}\delta_{c(i,\ell'),j}\\
=&\frac{1}{s}\sqrt{A_{ij}A^*_{ji}}\sum_{\ell,\ell'}\delta_{i,c(j,\ell)}\delta_{c(i,\ell'),j}=\frac{1}{s}A_{ij}.
\end{aligned}\tag{6.44}$$

In this equality, we have used that A is Hermitian: $A_{ij}=A^*_{ji}$, and there exists a unique ℓ such that $i=c(j,\ell)$, as well as a unique ℓ' such that $j=c(i,\ell')$. □

The quantum circuit in Fig. 6.7 is essentially the construction in [CKS17]. The relation with quantum walks will be further discussed in Section 7.2.

Exercise 6.1. Construct a query oracle O_A similar to that in Eq. (6.5), when $A_{ij}\in\mathbb{C}$ with $|A_{ij}|<1$.

Exercise 6.2. Let $A \in \mathbb{C}^{N\times N}$ be a s-sparse matrix. Prove that $\|A\| \leq s\|A\|_{\max}$. For every $1 \leq s \leq N$, provide an example that the equality can reached.

Exercise 6.3. Construct an s-sparse matrix so that the oracle in Eq. (6.25) does not exist.

Exercise 6.4. Let $A \in \mathbb{C}^{N\times N}$ ($N = 2^n$) be a Hermitian matrix with entries on the complex unit circle $A_{ij} = z_{ij}$, $|z_{ij}| = 1$.

(1) Construct a $2n$ qubit block-diagonal unitary $V \in \mathbb{C}^{N^2\times N^2}$ such that

$$V|0\rangle|j\rangle = \frac{1}{\sqrt{N}} \sum_{i\in[N]} \sqrt{\bar{z}_{ij}}\,|i\rangle|j\rangle\,, \quad j \in [N].$$

Here, block-diagonal means $(\langle x| \otimes I)V(|y\rangle \otimes I) = 0^{N\times N}$ for $x \neq y$.

(2) Draw a circuit which uses V to implement a block encoding U of A with n ancilla qubits . What is the prefactor α for the block encoding?

(3) Give an explicit expression for the entries of the block encoding U.

CHAPTER 7

Matrix functions of Hermitian matrices

Let A be an n-qubit Hermitian matrix. Then A has the eigenvalue decomposition

$$A = V\Lambda V^\dagger. \tag{7.1}$$

Here $\Lambda = \mathrm{diag}(\{\lambda_i\})$ is a diagonal matrix, and $\lambda_0 \le \cdots \le \lambda_{N-1}$. Let the scalar function f be well defined on all λ_i's. Then the matrix function $f(A)$ can be defined in terms of the eigendecomposition:

Definition 7.1 (Matrix function of Hermitian matrices)**.** *Let $A \in \mathbb{C}^{N\times N}$ be a Hermitian matrix with eigenvalue decomposition Eq. (7.1). Let $f : \mathbb{R} \to \mathbb{C}$ be a scalar function such that $f(\lambda_i)$ is defined for all $i \in [N]$. The matrix function is defined as*

$$f(A) := Vf(\Lambda)V^\dagger, \tag{7.2}$$

where

$$f(\Lambda) = \mathrm{diag}\left(f(\lambda_0), f(\lambda_1), \ldots, f(\lambda_{N-1})\right). \tag{7.3}$$

This chapter introduces techniques to construct an efficient quantum circuit to compute $f(A)|b\rangle$ for any state $|b\rangle$. Throughout the discussion we assume A is queried in the block encoding model denoted by U_A. For simplicity we assume that there is no error in the block encoding, i.e., $U_A \in \mathrm{BE}_{\alpha,m}(A)$, and WLOG we can take $\alpha = 1$.

Many tasks in scientific computation can be expressed in terms of matrix functions. Here are a few examples:

- Hamiltonian simulation: $f(A) = e^{\mathrm{i}At}$.
- Gibbs state preparation $f(A) = e^{-\beta A}$.
- Solving linear systems of equation $f(A) = A^{-1}$.
- Eigenstate filtering $f(A) = \mathbb{1}_{(-\infty,0)}(A - \mu I)$.

A key technique for representing matrix functions is called the qubitization.

7.1. Qubitization of Hermitian matrices with Hermitian block encoding

We first introduce some heuristic idea behind qubitization. For any $-1 < \lambda \le 1$, we can consider a 2×2 rotation matrix,

$$O(\lambda) = \begin{pmatrix} \lambda & -\sqrt{1-\lambda^2} \\ \sqrt{1-\lambda^2} & \lambda \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}. \tag{7.4}$$

where we have performed the change of variable $\lambda = \cos\theta$ with $0 \le \theta < \pi$.

Now direct computation shows

$$O^k(\lambda) = \begin{pmatrix} \cos(k\theta) & -\sin(k\theta) \\ \sin(k\theta) & \cos(k\theta) \end{pmatrix}. \tag{7.5}$$

Using the definition of Chebyshev polynomials (of first and second kinds, respectively)

$$T_k(\lambda) = \cos(k\theta) = \cos(k \arccos \lambda), \quad U_{k-1}(\lambda) = \frac{\sin(k\theta)}{\sin\theta} = \frac{\sin(k \arccos \lambda)}{\sqrt{1-\lambda^2}}, \tag{7.6}$$

we have

$$O^k(\lambda) = \begin{pmatrix} T_k(\lambda) & -\sqrt{1-\lambda^2}U_{k-1}(\lambda) \\ \sqrt{1-\lambda^2}U_{k-1}(\lambda) & T_k(\lambda) \end{pmatrix}. \tag{7.7}$$

Note that if we can somehow replace λ by A, we immediately obtain a $(1,1)$-block-encoding for the Chebyshev polynomial $T_k(A)$! This is precisely what qubitization aims at achieving, though there are some small twists.

In the simplest scenario, we assume that $U_A \in \mathrm{HBE}_{1,m}(A)$. Start from the spectral decomposition

$$A = \sum_i \lambda_i |v_i\rangle \langle v_i|, \tag{7.8}$$

we have that for each eigenstate $|v_i\rangle$,

$$U_A |0^m\rangle |v_i\rangle = |0^m\rangle A |v_i\rangle + |\widetilde{\perp}_i\rangle = \lambda_i |0^m\rangle |v_i\rangle + |\widetilde{\perp}_i\rangle. \tag{7.9}$$

Here $|\widetilde{\perp}_i\rangle$ is an unnormalized state that is orthogonal to all states of the form $|0^m\rangle |\psi\rangle$, i.e.,

$$\Pi |\widetilde{\perp}_i\rangle = 0. \tag{7.10}$$

where

$$\Pi = |0^m\rangle \langle 0^m| \otimes I_n \tag{7.11}$$

is a projection operator.

Since the right hand side of Eq. (7.9) is a normalized state, we may also write

$$|\widetilde{\perp}_i\rangle = \sqrt{1-\lambda_i^2} |\perp_i\rangle, \tag{7.12}$$

where $|\perp_i\rangle$ is a normalized state.

Now if $\lambda_i = \pm 1$, then $\mathcal{H}_i = \mathrm{span}\{|0^m\rangle |v_i\rangle\}$ is already an invariant subspace of U_A, and $|\perp_i\rangle$ can be any state. Otherwise, use the fact that $U_A = U_A^\dagger$, we can apply U_A again to both sides of Eq. (7.9) and obtain

$$U_A |\perp_i\rangle = \sqrt{1-\lambda_i^2} |0^m\rangle |v_i\rangle - \lambda_i |\perp_i\rangle. \tag{7.13}$$

Therefore $\mathcal{H}_i = \mathrm{span}\{|0^m\rangle |v_i\rangle, |\perp_i\rangle\}$ is an invariant subspace of U_A. Furthermore, the matrix representation of U_A with respect to the basis $\mathcal{B}_i = \{|0^m\rangle |v_i\rangle, |\perp_i\rangle\}$ is

$$[U_A]_{\mathcal{B}_i} = \begin{pmatrix} \lambda_i & \sqrt{1-\lambda_i^2} \\ \sqrt{1-\lambda_i^2} & -\lambda_i \end{pmatrix}, \tag{7.14}$$

i.e., U_A restricted to $\mathcal{H}_i$ is a reflection operator. This also leads to the name "qubitization", which means that each eigenvector $|v_i\rangle$ is "qubitized" into a two-dimensional space $\mathcal{H}_i$.

In order to construct a block encoding for $T_k(A)$, we need to turn U_A into a rotation. For this note that $\mathcal{H}_i$ is also an invariant subspace for the projection operator Π:

$$[\Pi]_{\mathcal{B}_i} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}. \tag{7.15}$$

Similarly define $Z_\Pi = 2\Pi - 1$, since

$$[Z_\Pi]_{\mathcal{B}_i} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \tag{7.16}$$

Z_Π acts as a reflection operator restricted to each subspace $\mathcal{H}_i$. Then $\mathcal{H}_i$ is the invariant subspace for the *iterate*

$$O = U_A Z_\Pi \tag{7.17}$$

and

$$[O]_{\mathcal{B}_i} = \begin{pmatrix} \lambda_i & -\sqrt{1-\lambda_i^2} \\ \sqrt{1-\lambda_i^2} & \lambda_i \end{pmatrix} \tag{7.18}$$

is the desired rotation matrix. Therefore

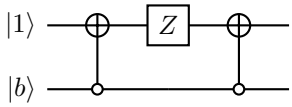
$$[O^k]_{\mathcal{B}_i} = [(U_A Z_\Pi)^k]_{\mathcal{B}_i} = \begin{pmatrix} T_k(\lambda_i) & -\sqrt{1-\lambda_i^2} U_{k-1}(\lambda_i) \\ \sqrt{1-\lambda_i^2} U_{k-1}(\lambda_i) & T_k(\lambda_i) \end{pmatrix}. \tag{7.19}$$

Since $\{|0^m\rangle |v_i\rangle\}$ spans the range of Π, we have

$$O^k = \begin{pmatrix} T_k(A) & * \\ * & * \end{pmatrix} \tag{7.20}$$

i.e., $O^k = (U_A Z_\Pi)^k$ is a $(1, m)$-block-encoding of the Chebyshev polynomial $T_k(A)$.

In order to implement Z_Π, note that if $m = 1$, then Z_Π is just the Pauli Z gate. When $m > 1$, the circuit



returns $|1\rangle |0^m\rangle$ if $b = 0^m$, and $-|1\rangle |b\rangle$ if $b \neq 0^m$. So this precisely implements Z_Π where the signal qubit $|1\rangle$ is used as a work register. We may also discard the signal qubit, and resulting unitary is denoted by Z_Π.

In other words, the circuit in Fig. 7.1 implements the operator O. Repeating the circuit k times gives the $(1, m+1)$-block-encoding of $T_k(A)$.

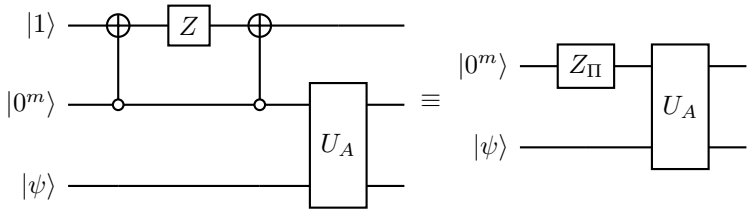


FIGURE 7.1. Circuit implementing one step of qubitization with a Hermitian block encoding of a Hermitian matrix. Here $U_A \in \mathrm{HBE}_{1,m}(A)$.

Remark 7.2 (Alternative perspectives of qubitization)**.** The fact that an arbitrarily large block encoding matrix U_A can be partially block diagonalized into N subblocks of size 2×2 may seem a rather peculiar algebraic structure. In fact there are other alternative perspectives and derivations

of the qubitization result. Some noticeable ones include the use of Jordan's Lemma, and the use of the cosine-sine (CS) decomposition. Throughout this chapter and the next chapter, we will adopt the more "elementary" derivations used above. ◇

7.2. Application: Szegedy's quantum walk*

Quantum walk is one of the major topics in quantum algorithms. Roughly speaking, there are two versions of quantum walks. The continuous time quantum walk is the closely analogous to its classical counterpart, i.e., continuous time random walk. The other version, the discrete time quantum walk, or Szegedy's quantum walk [Sze04] is not so obviously connected to the classical random walks. We will not introduce the motivations behind the continuous and discrete time random walks, and refer readers to [Chi21, Chapter 16,17] for detailed discussions.

7.2.1. Basics of Markov chain. Let $G = (V, E)$ be a graph of size N. A Markov chain (or a random walk) is given by a transition matrix P, with its entry P_{ij} denoting the probability of the transition from vertex i to vertex j. The matrix P is a stochastic matrix satisfying

$$P_{ij} \geq 0, \quad \sum_j P_{ij} = 1. \tag{7.21}$$

Let π be the stationary state, which is a left eigenvector of P with eigenvalue 1:

$$\sum_i \pi_i P_{ij} = \pi_j, \quad \pi_i \geq 0, \quad \sum_i \pi_i = 1. \tag{7.22}$$

A Markov chain is *irreducible* if any state can be reached from any other state in a finite number of steps. An irreducible Markov chain is *aperiodic* if there exists no integer greater than one that divides the length of every directed cycle of the graph. A Markov chain is *ergodic* if it is both irreducible and aperiodic. By the Perron–Frobenius Theorem, any ergodic Markov chain P has a unique stationary state π, and $\pi_i > 0$ for all i. A Markov chain is *reversible* if the following detailed balance condition is satisfied

$$\pi_i P_{ij} = \pi_j P_{ji}. \tag{7.23}$$

Now we define the *discriminant matrix* associated with a Markov chain as

$$D_{ij} = \sqrt{P_{ij} P_{ji}}, \tag{7.24}$$

which is real symmetric and hence Hermitian. For a reversible Markov chain, the stationary state can be encoded as an eigenvector of D (the proof is left as an exercise).

Proposition 7.3 (Reversible Markov chain)**.** *If a Markov chain is reversible, then the coherent version of the stationary state*

$$|\pi\rangle = \sum_i \sqrt{\pi_i}\, |i\rangle \tag{7.25}$$

is a normalized eigenvector of the discriminant matrix D satisfying

$$D\,|\pi\rangle = |\pi\rangle\,. \tag{7.26}$$

Furthermore, when $\pi_i > 0$ for all i, we have

$$D = \operatorname{diag}(\sqrt{\pi}) P \operatorname{diag}(\sqrt{\pi})^{-1}. \tag{7.27}$$

Therefore the set of (left) eigenvalues of P and the set of the eigenvalues of D are the same.

7.2.2. Block encoding of the discriminant matrix. Our first goal is to construct a Hermitian block encoding of D. Assume that we have access to an oracle O_P satisfying

$$O_P |0^n\rangle |j\rangle = \sum_k \sqrt{P_{jk}} |k\rangle |j\rangle . \tag{7.28}$$

Thanks to the stochasticity of P, the right hand side is already a normalized vector, and no additional signal qubit is needed.

We also introduce the n-qubit SWAP operator Swap operator:

$$\mathrm{SWAP} |i\rangle |j\rangle = |j\rangle |i\rangle , \tag{7.29}$$

which swaps the value of the two registers in the computational basis, and can be directly implemented using n two-qubit SWAP gates.

We claim that the following circuit gives $U_D \in \mathrm{HBE}_{1,n}(D)$.

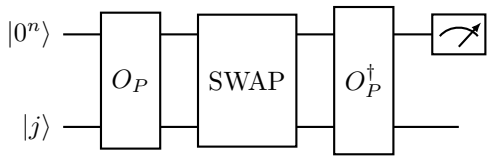


FIGURE 7.2. Circuit for the Hermitian block encoding of a discriminant matrix.

Proposition 7.4. *Fig. 7.2 defines* $U_D \in \mathrm{HBE}_{1,n}(D)$.

PROOF. Clearly U_D is unitary and Hermitian. Now we compute as before

$$|0^n\rangle |j\rangle \xrightarrow{O_P} \sum_k \sqrt{P_{jk}} |k\rangle |j\rangle \xrightarrow{\mathrm{SWAP}} \sum_k \sqrt{P_{jk}} |j\rangle |k\rangle . \tag{7.30}$$

Meanwhile

$$|0^n\rangle |i\rangle \xrightarrow{O_P} \sum_{k'} \sqrt{P_{ik'}} |k'\rangle |i\rangle . \tag{7.31}$$

So the inner product gives

$$\langle 0^n| \langle i| U_D |0^n\rangle |j\rangle = \sum_{k,k'} \sqrt{P_{ik'}P_{jk}}\delta_{j,k'}\delta_{i,k} = \sqrt{P_{ij}P_{ji}} = D_{ij}. \tag{7.32}$$

This proves the claim. □

7.2.3. Szegedy's quantum walk. For a Markov chain defined on a graph $G = (V, E)$, Szegedy's quantum walk implements a qubitization of the discriminant matrix D in Eq. (7.24). Let U_D be the Hermitian block encoding defined by the circuit in Fig. 7.2, we may readily plug it into Fig. 7.1, and obtain the circuit in Fig. 7.3 denoted by O_Z.

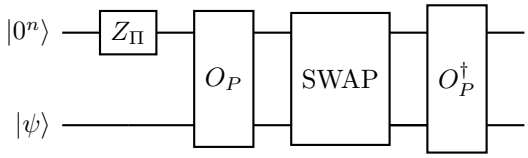


FIGURE 7.3. Circuit implementing one step of Szegedy's quantum walk operator.

Let the eigendecomposition of D be denoted by

$$D\left|v_i\right\rangle = \lambda_i \left|v_i\right\rangle. \tag{7.33}$$

For each $|v_i\rangle$, the associated basis in the 2-dimensional subspace is $\mathcal{B}_i = \{|0^m\rangle\,|v_i\rangle\,, |\perp_i\rangle\}$. Then the qubitization procedure gives

$$[O_Z]_{\mathcal{B}_i} = \begin{pmatrix} \lambda_i & -\sqrt{1-\lambda_i^2} \\ \sqrt{1-\lambda_i^2} & \lambda_i \end{pmatrix}. \tag{7.34}$$

The eigenvalues of O_Z in the 2×2 matrix block are

$$e^{\pm \mathrm{i} \arccos(\lambda_i)}. \tag{7.35}$$

This relation is important for the following reasons. By Proposition 7.3, if a Markov chain is reversible and ergodic, the eigenvalues of D and P are the same. In particular, the largest eigenvalue of D is unique and is equal to 1, and the second largest eigenvalue of D is $1-\delta$, where $\delta > 0$ is called the spectral gap. Since $\arccos(1) = 0$, and $\arccos(1-\delta) \approx \sqrt{2\delta}$, we find that the spectral gap of O_Z on the unit circle is in fact $\mathcal{O}(\sqrt{\delta})$ instead of $\mathcal{O}(\delta)$. This is called the spectral gap amplification, which leads to e.g. the quadratic quantum speedup of the hitting time.

Example 7.5 (Determining whether there is a marked vertex in a complete graph)**.** Let $G = (V, E)$ be a complete, graph of $N = 2^n$ vertices. We would like to distinguish the following two scenarios:

(1) All vertices are the same, and the random walk is given by the transition matrix

$$P = \frac{1}{N} e e^\top, \quad e = (1, \ldots, 1)^\top. \tag{7.36}$$

(2) There is one *marked* vertex. Without loss of generality we may assume this is the 0-th vertex (of course we do not have access to this information). In this case, the transition matrix is

$$\widetilde{P}_{ij} = \begin{cases} \delta_{ij}, & i = 0, \\ P_{ij}, & i > 0. \end{cases} \tag{7.37}$$

In other words, in the case (2), the random walk will stop at the marked index. The transition matrix can also be written in the block partitioned form as

$$\widetilde{P} = \begin{pmatrix} 1 & 0 \\ \frac{1}{N}\widetilde{e} & \frac{1}{N}\widetilde{e}\widetilde{e}^\top \end{pmatrix}. \tag{7.38}$$

Here $\widetilde{e}$ is an all 1 vector of length $N-1$.

For the random walk defined by P, the stationary state is $\pi = \frac{1}{N}e$, and the spectral gap is 1. For the random walk defined by $\widetilde{P}$, the stationary state is $\widetilde{\pi} = (1, 0, \ldots, 0)^\top$, and the spectral gap of is $\delta = N^{-1}$. Starting from the uniform state π, the probability distribution after k steps of

random walk is $\pi^\top \widetilde{P}^k$. This converges to the stationary state of $\widetilde{P}$, and hence reach the marked vertex after $\mathcal{O}(N)$ steps of walks (exercise).

These properties are also inherited by the discriminant matrices, with $D = P$ and

$$\widetilde{D} = \begin{pmatrix} 1 & 0 \\ 0 & \frac{1}{N}\widetilde{e}\widetilde{e}^\top \end{pmatrix}. \tag{7.39}$$

To distinguish the two cases, we are given a Szegedy quantum walk operator called O, which can be either O_Z or $\widetilde{O}_Z$, which is associated with $D, \widetilde{D}$, respectively. The initial state is

$$|\psi_0\rangle = |0^n\rangle \left(H^{\otimes n} |0^n\rangle\right). \tag{7.40}$$

Our strategy is to measure the expectation

$$m_k = \langle \psi_0 | O^k | \psi_0 \rangle , \tag{7.41}$$

which can be obtained via Hadamard's test.

Before determining the value of k, first notice that if $O = O_Z$, then $O_Z |\psi_0\rangle = |\psi_0\rangle$. Hence $m_k = 1$ for all values of k.

On the other hand, if $O = \widetilde{O}_Z$, we use the fact that $\widetilde{D}$ only has two nonzero eigenvalues 1 and $(N-1)/N = 1 - \delta$, with associated eigenvectors denoted by $|\widetilde{\pi}\rangle$ and $|\widetilde{v}\rangle = \frac{1}{\sqrt{N-1}}(0, 1, 1 \ldots, 1)^\top$, respectively. Furthermore,

$$|\psi_0\rangle = \frac{1}{\sqrt{N}} |0^n\rangle |\widetilde{\pi}\rangle + \sqrt{\frac{N-1}{N}} |0^n\rangle |\widetilde{v}\rangle . \tag{7.42}$$

Due to qubitization, we have

$$\widetilde{O}_Z^k |\psi_0\rangle = \frac{1}{\sqrt{N}} |0^n\rangle T_k(1) |\widetilde{\pi}\rangle + \sqrt{\frac{N-1}{N}} |0^n\rangle T_k(1-\delta) |\widetilde{v}\rangle + |\perp\rangle , \tag{7.43}$$

where $|\perp\rangle$ is an unnormalized state satisfying $(|0^n\rangle \langle 0^n|) \otimes I_n |\perp\rangle = 0$. Then using $T_k(1) = 1$ for all k, we have

$$m_k = \frac{1}{N} + \left(1 - \frac{1}{N}\right) T_k(1-\delta). \tag{7.44}$$

Use the fact that $T_k(1-\delta) = \cos(k \arccos(1-\delta))$, in order to have $T_k(1-\delta) \approx 0$, the smallest k satisfies

$$k \approx \frac{\pi}{2 \arccos(1-\delta)} \approx \frac{\pi}{2\sqrt{2\delta}} = \frac{\pi\sqrt{N}}{2\sqrt{2}}. \tag{7.45}$$

Therefore taking $k = \lceil \frac{\pi\sqrt{N}}{2\sqrt{2}} \rceil$, we have $m_k \approx 1/N$. Running Hadamard's test to constant accuracy allows us to distinguish the two scenarios. ◇

Remark 7.6 (Without using the Hadamard test)**.** Alternatively, we may evaluate the success probability of obtaining 0^n in the ancilla qubits, i.e.,

$$p(0^n) = \left\| (|0^n\rangle \langle 0^n| \otimes I_n) O^k |\psi_0\rangle \right\|^2 . \tag{7.46}$$

When $O = O_Z$, we have $p(0^n) = 1$ with certainty. When $O = \widetilde{O}_Z$, according to Eq. (7.43),

$$p(0^n) = \frac{1}{N} + \left(1 - \frac{1}{N}\right) T_k^2(1-\delta). \tag{7.47}$$

So running the problem with $k = \lceil \frac{\pi\sqrt{N}}{2\sqrt{2}} \rceil$, we can distinguish between the two cases. ◇

Remark 7.7 (Comparison with Grover's search)**.** It is natural to draw comparisons between Szegedy's quantum walk and Grover's search. The two algorithms make queries to different oracles, and both yield quadratic speedup compared to the classical algorithms. The quantum walk is slightly weaker, since it only tells whether there is one marked vertex or not. On the other hand, Grover's search also finds the location of the marked vertex. Both algorithms consist of repeated usage of the product of two reflectors. The number of iterations need to be carefully controlled. Indeed, choosing a polynomial degree four times as large as Eq. (7.45) would result in $m_k \approx 1$ for the case with a marked vertex. ◇

Remark 7.8 (Comparison with QPE)**.** Another possible solution of the problem of finding the marked vertex is to perform QPE on the Szegedy walk operator O (which can be O_Z or $\widetilde{O}_Z$). The effectiveness of the method rests on the spectral gap amplification discussed above. We refer to [Chi21, Chapter 17] for more details. ◇

7.2.4. Comparison with the original version of Szegedy's quantum walk. The quantum walk procedure can also be presented as follows. Using the O_P oracle and the multi-qubit SWAP gate, we can define two set of quantum states

$$\begin{aligned} |\psi_j^1\rangle &= O_P |0^n\rangle |j\rangle = \sum_k \sqrt{P_{jk}} |k\rangle |j\rangle , \\ |\psi_j^2\rangle &= \mathrm{SWAP}(O_P |0^n\rangle |j\rangle) = \sum_k \sqrt{P_{jk}} |j\rangle |k\rangle . \end{aligned} \tag{7.48}$$

This defines two projection operators

$$\Pi_l = \sum_{j\in[N]} |\psi_j^l\rangle \langle\psi_j^l| , \quad l = 1, 2, \tag{7.49}$$

from which we can define two $2n$-qubit reflection operators $R_{\Pi_l} = 2\Pi_l - I_{2n}$. Let us write down the reflection operators more explicitly. Using the resolution of identity,

$$R_{\Pi_1} = O_P((2 |0^n\rangle \langle 0^n| - I) \otimes I_n) O_P^\dagger = O_P(Z_\Pi \otimes I_n) O_P^\dagger. \tag{7.50}$$

Similarly

$$R_{\Pi_2} = \mathrm{SWAP}\, O_P(Z_\Pi \otimes I_n) O_P^\dagger \,\mathrm{SWAP}\, . \tag{7.51}$$

Then Szegedy's quantum walk operator takes the form

$$\mathcal{U}_Z = R_{\Pi_2} R_{\Pi_1} , \tag{7.52}$$

which is a rotation operator that resembles Grover's algorithm. Note that

$$\mathcal{U}_Z = \mathrm{SWAP}\, O_P(Z_\Pi \otimes I_n) O_P^\dagger \,\mathrm{SWAP}\, O_P(Z_\Pi \otimes I_n) O_P^\dagger, \tag{7.53}$$

so

$$O_P^\dagger \mathcal{U}_Z (O_P^\dagger)^{-1} = O_Z^2, \tag{7.54}$$

so the walk operator is the same as a block encoding of $T_2(D)$ using qubitization, up to a matrix similarity transformation, and the eigenvalues are the same. In particular, consider the matrix power O_Z^k, which provides a block encoding of the Chebyshev matrix polynomial $T_k(D)$. Then the difference between O_Z^{2k} and $\mathcal{U}_Z^k$ appears only at the beginning and end of the circuit.

7.3. Linear combination of unitaries

In practical applications, we are often not interested in constructing the Chebyshev polynomial of A, but a linear combination of Chebyshev polynomials. For instance, the matrix inversion problem can be solved by expanding $f(x) = x^{-1}$ using a linear combination of Chebyshev polynomials, on the interval $[-1, -\kappa^{-1}] \cup [\kappa^{-1}, 1]$. Here we assume $\|A\| = 1$ and $\kappa = \|A\| \left\|A^{-1}\right\|$ is the condition number. One way to implement this is via a quantum primitive called the linear combination of unitaries (LCU).

Let $T = \sum_{i=0}^{K-1} \alpha_i U_i$ be the linear combination of unitary matrices U_i. For simplicity let $K = 2^a$, and $\alpha_i > 0$ (WLOG we can absorb the phase of α_i into the unitary U_i). Then

$$U := \sum_{i \in [K]} |i\rangle \langle i| \otimes U_i, \tag{7.55}$$

implements the selection of U_i conditioned on the value of the a-qubit ancilla states (also called the control register). U is called a *select oracle*.

Let V be a unitary operation satisfying

$$V |0^a\rangle = \frac{1}{\sqrt{\|\alpha\|_1}} \sum_{i \in [K]} \sqrt{\alpha_i} |i\rangle , \tag{7.56}$$

and V is called the *prepare oracle*. The 1-norm of the coefficients is given by

$$\|\alpha\|_1 = \sum_i |\alpha_i| . \tag{7.57}$$

In the matrix form

$$V = \frac{1}{\sqrt{\|\alpha\|_1}} \begin{pmatrix} \sqrt{\alpha_0} & * & \cdots & * \\ \vdots & * & \ddots & \vdots \\ \sqrt{\alpha_{K-1}} & * & \cdots & * \end{pmatrix} . \tag{7.58}$$

where the first basis is $|0^m\rangle$, and all other basis functions are orthogonal to it. Then

$$V^\dagger = \frac{1}{\sqrt{\|\alpha\|_1}} \begin{pmatrix} \sqrt{\alpha_0} & \cdots & \sqrt{\alpha_{K-1}} \\ * & \cdots & * \\ \vdots & \ddots & \vdots \\ * & \cdots & * \end{pmatrix} . \tag{7.59}$$

Then T can be implemented using the unitary given in Lemma 7.9 (called the LCU lemma).

Lemma 7.9 (LCU)**.** *Define $W = (V^\dagger \otimes I_n) U (V \otimes I_n)$, then for any $|\psi\rangle$,*

$$W |0^a\rangle |\psi\rangle = \frac{1}{\|\alpha\|_1} |0^a\rangle T |\psi\rangle + |\widetilde{\perp}\rangle , \tag{7.60}$$

where $|\widetilde{\perp}\rangle$ is an unnormalized state satisfying

$$(|0^a\rangle \langle 0^a| \otimes I_n) |\widetilde{\perp}\rangle = 0. \tag{7.61}$$

In other words, $W \in \mathrm{BE}_{\|\alpha\|_1, a}(T)$.

PROOF. First

$$U(V \otimes I_n) |0^a\rangle |\psi\rangle = U \frac{1}{\sqrt{\|\alpha\|_1}} \sum_i \sqrt{\alpha_i} |i\rangle |\psi\rangle = \frac{1}{\sqrt{\|\alpha\|_1}} \sum_i \sqrt{\alpha_i} |i\rangle U_i |\psi\rangle . \tag{7.62}$$

Then using the matrix representation (7.59), and let the state $|\widetilde{\perp}\rangle$ collect all the states marked by $*$ orthogonal to $|0^m\rangle$,

$$(V^\dagger \otimes I_n) U (V \otimes I_n) |0^a\rangle |\psi\rangle = \frac{1}{\|\alpha\|_1} |0^a\rangle \sum_i \alpha_i U_i |\psi\rangle + |\widetilde{\perp}\rangle = \frac{1}{\|\alpha\|_1} |0^a\rangle T |\psi\rangle + |\widetilde{\perp}\rangle . \tag{7.63}$$

□

The LCU Lemma is a useful quantum primitive, as it states that the number of ancilla qubits needed only depends logarithmically on K, the number of terms in the linear combination. Hence it is possible to implement the linear combination of a very large number of terms efficiently. From a practical perspective, the select and prepare oracles uses multi-qubit controls, and can be difficult to implement. If implemented directly, the number of multi-qubit controls again depends linearly on K and is not desirable. Therefore an efficient implementation using LCU (in terms of the gate complexity) also requires additional structures in the prepare and select oracles.

If we apply W to $|0^a\rangle |\psi\rangle$ and measure the ancilla qubits, then the probability of obtaining the outcome 0^a in the ancilla qubits (and therefore obtaining the state $T |\psi\rangle / \|T |\psi\rangle\|$ in the system register) is $(\|T |\psi\rangle\| / \|\alpha\|_1)^2$. The expected number of repetition needed to succeed is $(\|\alpha\|_1 / \|T |\psi\rangle\|)^2$. Now we demonstrate that using amplitude amplification (AA) in Section 2.3, this number can be reduced to $\mathcal{O}(\|\alpha\|_1 / \|T |\psi\rangle\|)$.

Remark 7.10 (Alternative construction of the prepare oracle)**.** In some applications it may not be convenient to absorb the phase of α_i into the select oracle. In such a case, we may modify the prepare oracle instead. If $\alpha_i = r_i e^{\mathrm{i}\theta_i}$ with $r_i > 0, \theta_i \in [0, 2\pi)$, we can define $\sqrt{\alpha_i} = \sqrt{r_i} e^{\mathrm{i}\theta_i/2}$, and V is defined as in Eq. (7.56). However, instead of $V^\dagger$, we need to introduce

$$\widetilde{V} = \frac{1}{\sqrt{\|\alpha\|_1}} \begin{pmatrix} \sqrt{\alpha_0} & \cdots & \sqrt{\alpha_{K-1}} \\ * & \cdots & * \\ \vdots & \ddots & \vdots \\ * & \cdots & * \end{pmatrix} . \tag{7.64}$$

Then following the same proof as Lemma 7.9, we find that $W = (\widetilde{V} \otimes I_n) U (V \otimes I_n) \in \mathrm{BE}_{\|\alpha\|_1, a}(T)$. ◇

Remark 7.11 (Linear combination of non-unitaries)**.** Using the block encoding technique, we may immediately obtain linear combination of general matrices that are not unitaries. However, with some abuse of notation, the term "LCU" will be used whether the terms to be combined are unitaries or not. In other words, the term "linear combination of unitaries" should be loosely interpreted as "linear combination of things" (LCT) in many contexts. ◇

Example 7.12 (Linear combination of two matrices)**.** Let U_A, U_B be two n-qubit unitaries, and we would like to construct a block encoding of $T = U_A + U_B$.

There are two terms in total, so one ancilla qubit is needed. The prepare oracle needs to implement

$$V|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle), \tag{7.65}$$

so this is the Hadamard gate. The circuit is given by Fig. 7.4, which constructs $W \in \mathrm{BE}_{\sqrt{2},1}(T)$.

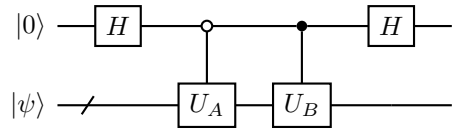


FIGURE 7.4. Circuit for linear combination of two unitaries.

A special case is the linear combination of two block encoded matrices. Given two n-qubit matrices A, B, for simplicity let $U_A \in \mathrm{BE}_{1,m}(A), U_B \in \mathrm{BE}_{1,m}(B)$. We would like to construct a block encoding of $T = A + B$. The circuit is given by Fig. 7.5, which constructs $W \in \mathrm{BE}_{\sqrt{2},1+m}(T)$. This is also an example of a linear combination of non-unitary matrices.

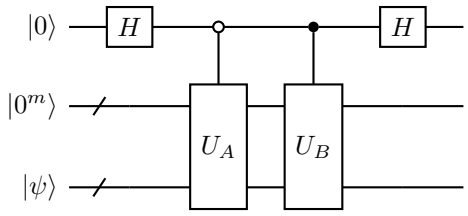


FIGURE 7.5. Circuit for linear combination of two block encoded matrices.

◇

Example 7.13 (Transverse field Ising model)**.** Consider the following TFIM model with periodic boundary conditions ($Z_n = Z_0$), and $n = 2^{\mathfrak{n}}$,

$$\hat{H} = -\sum_{i\in[n]} Z_i Z_{i+1} - \sum_{i\in[n]} X_i. \tag{7.66}$$

In order to use LCU, we need $(\mathfrak{n}+1)$ ancilla qubits. The prepare oracle can be simply constructed from the Hadamard gate

$$V = H^{\otimes(\mathfrak{n}+1)}, \tag{7.67}$$

and the select oracle implements

$$U = \sum_{i\in[n]} |i\rangle\langle i| \otimes (-Z_i Z_{i+1}) + \sum_{i\in[n]} |i+n\rangle\langle i+n| \otimes (-X_i). \tag{7.68}$$

The corresponding $W \in \mathrm{BE}_{\sqrt{2n},\mathfrak{n}+1}(\hat{H})$. ◇

Example 7.14 (Block encoding of a matrix polynomial)**.** Let us use the LCU lemma to construct the block encoding for an arbitrary matrix polynomial for a Hermitian matrix A in Section 7.1.

$$f(A) = \sum_{k\in[K]} \alpha_k T_k(A), \tag{7.69}$$

with $\|\alpha\|_1 = \sum_{k\in[K]} |\alpha_k|$ and we set $K = 2^a$. For simplicity assume $\alpha_k \geq 0$.

We have constructed $U_k := (U_A Z_\Pi)^k$ as the $(1, m)$-block-encoding of $T_k(A)$. From each U_k we can implement the select oracle

$$U := \sum_{k\in[K]} |k\rangle \langle k| \otimes U_k \tag{7.70}$$

via multi-qubit controls. Also given the availability of the prepare oracle

$$V |0^a\rangle = \frac{1}{\sqrt{\|\alpha\|_1}} \sum_{k\in[K]} \sqrt{\alpha_k} |k\rangle, \tag{7.71}$$

we obtain a $(\|\alpha\|_1, m + a)$-block-encoding of $f(A)$.

The need of using a ancilla qubits, and even more importantly the need to implement the prepare and select oracles is undesirable. We will see later that the quantum signal processing (QSP) and quantum singular value transformation (QSVT) can drastically reduce both sources of difficulties. ◇

Example 7.15 (Matrix functions given by a matrix Fourier series)**.** Instead of block encoding, LCU can also utilize a different query model based on Hamiltonian simulation. Let A be an n-qubit Hermitian matrix. Consider $f(x) \in \mathbb{R}$ given by its Fourier expansion (up to a normalization factor)

$$f(x) = \int \hat{f}(k) e^{\mathrm{i}kx} \,\mathrm{d}k, \tag{7.72}$$

and we are interested in computing the matrix function via numerical quadrature

$$f(A) = \int \hat{f}(k) e^{\mathrm{i}kA} \,\mathrm{d}k \approx \Delta k \sum_{k\in\mathcal{K}} \hat{f}(k) e^{\mathrm{i}kA}. \tag{7.73}$$

Here $\mathcal{K}$ is a uniform grid discretizing the interval $[-L, L]$ using $|\mathcal{K}| = 2^{\mathfrak{k}}$ grid points, and the grid spacing is $\Delta k = 2L/|\mathcal{K}|$. The prepare oracle is given by the coefficients $c_k = \Delta k \hat{f}(k)$, and the corresponding subnormalization factor is

$$\|c\|_1 = \sum_{k\in\mathcal{K}} \Delta k \left|\hat{f}(k)\right| \approx \int \left|\hat{f}(k)\right| \,\mathrm{d}k. \tag{7.74}$$

The select oracle is

$$U = \sum_{k\in\mathcal{K}} |k\rangle \langle k| \otimes e^{\mathrm{i}kA}. \tag{7.75}$$

This can be efficiently implemented using the controlled matrix powers as in Fig. 3.6, where the basic unit is the short time Hamiltonian simulation $e^{\mathrm{i}\Delta kA}$. This can be used to block encode a large class of matrix functions. ◇

7.4. Qubitization of Hermitian matrices with general block encoding

In Section 7.1 we assume that $U_A = U_A^\dagger$ to block encode a Hermitian matrix A. For instance, s-sparse Hermitian matrices, such Hermitian block encodings can be constructed following the recipe in Section 6.5. However, this can come at the expense of requiring additional structures and oracles. In general, the block encoding of a Hermitian matrix may not be Hermitian itself. In this section we demonstrate that the strategy of qubitization can be modified to accommodate general block encodings.

Again start from the eigendecomposition Eq. (7.8), we apply U_A to $|0^m\rangle |v_i\rangle$ and obtain

$$U_A |0^m\rangle |v_i\rangle = \lambda_i |0^m\rangle |v_i\rangle + \sqrt{1-\lambda_i^2}\, |\perp_i'\rangle , \tag{7.76}$$

where $|\perp_i'\rangle$ is a normalized state satisfying $\Pi |\perp_i'\rangle = 0$.

Since U_A block-encodes a Hermitian matrix A, we have

$$U_A^\dagger = \begin{pmatrix} A & * \\ * & * \end{pmatrix}, \tag{7.77}$$

which implies that there exists another normalized state $|\perp_i\rangle$ satisfying $\Pi |\perp_i\rangle = 0$ and

$$U_A^\dagger |0^m\rangle |v_i\rangle = \lambda_i |0^m\rangle |v_i\rangle + \sqrt{1-\lambda_i^2}\, |\perp_i\rangle . \tag{7.78}$$

Now apply U_A to both sides of Eq. (7.78), we obtain

$$|0^m\rangle |v_i\rangle = \lambda_i^2 |0^m\rangle |v_i\rangle + \lambda_i \sqrt{1-\lambda_i^2}\, |\perp_i'\rangle + \sqrt{1-\lambda_i^2} U_A |\perp_i\rangle , \tag{7.79}$$

which gives

$$U_A |\perp_i\rangle = \sqrt{1-\lambda_i^2}\, |0^m\rangle |v_i\rangle - \lambda_i |\perp_i'\rangle . \tag{7.80}$$

Define

$$\mathcal{B}_i = \{|0^m\rangle |v_i\rangle , |\perp_i\rangle\}, \quad \mathcal{B}_i' = \{|0^m\rangle |v_i\rangle , |\perp_i'\rangle\}, \tag{7.81}$$

and the associated two-dimensional subspaces $\mathcal{H}_i = \operatorname{span} B_i, \mathcal{H}_i' = \operatorname{span} B_i'$, we find that U_A maps $\mathcal{H}_i$ to $\mathcal{H}_i'$. Correspondingly $U_A^\dagger$ maps $\mathcal{H}_i'$ to $\mathcal{H}_i$.

Then Eqs. (7.76) and (7.80) give the matrix representation

$$[U_A]_{\mathcal{B}_i}^{\mathcal{B}_i'} = \begin{pmatrix} \lambda_i & \sqrt{1-\lambda_i^2} \\ \sqrt{1-\lambda_i^2} & -\lambda_i \end{pmatrix}. \tag{7.82}$$

Similar calculation shows that

$$[U_A^\dagger]_{\mathcal{B}_i'}^{\mathcal{B}_i} = \begin{pmatrix} \lambda_i & \sqrt{1-\lambda_i^2} \\ \sqrt{1-\lambda_i^2} & -\lambda_i \end{pmatrix}. \tag{7.83}$$

Meanwhile both $\mathcal{H}_i$ and $\mathcal{H}_i'$ are the invariant subspaces of the projector Π, with matrix representation

$$[\Pi]_{\mathcal{B}_i} = [\Pi]_{\mathcal{B}_i'} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}. \tag{7.84}$$

Therefore

$$[Z_\Pi]_{\mathcal{B}_i} = [Z_\Pi]_{\mathcal{B}_i'} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{7.85}$$

Hence $\mathcal{H}_i$ is an invariant subspace of $\widetilde{O} = U_A^\dagger Z_\Pi U_A Z_\Pi$, with matrix representation

$$[\widetilde{O}]_{\mathcal{B}_i} = \begin{pmatrix} \lambda_i & -\sqrt{1-\lambda_i^2} \\ \sqrt{1-\lambda_i^2} & \lambda_i \end{pmatrix}^2. \tag{7.86}$$

Repeating k times, we have

$$\begin{aligned} [\widetilde{O}^k]_{\mathcal{B}_i} =& (U_A^\dagger Z_\Pi U_A Z_\Pi)^k = \begin{pmatrix} \lambda_i & -\sqrt{1-\lambda_i^2} \\ \sqrt{1-\lambda_i^2} & \lambda_i \end{pmatrix}^{2k} \\ =& \begin{pmatrix} T_{2k}(\lambda_i) & -\sqrt{1-\lambda_i^2} U_{2k-1}(\lambda_i) \\ \sqrt{1-\lambda_i^2} U_{2k-1}(\lambda_i) & T_{2k}(\lambda_i) \end{pmatrix}. \end{aligned} \tag{7.87}$$

Since any vector $|0^m\rangle |\psi\rangle$ can be expanded in terms of the eigenvectors $|0^m\rangle |v_i\rangle$, we have

$$(U_A^\dagger Z_\Pi U_A Z_\Pi)^k = \begin{pmatrix} T_{2k}(A) & * \\ * & * \end{pmatrix}. \tag{7.88}$$

Therefore if we would like to construct an *even* order Chebyshev polynomial $T_{2k}(A)$, the circuit $(U_A^\dagger Z_\Pi U_A Z_\Pi)^k$ straightforwardly gives a $(1, m)$-block-encoding.

In order to construct the block-encoding of an *odd* polynomial $T_{2k+1}(A)$, we note that

$$[U_A Z_\Pi (U_A^\dagger Z_\Pi U_A Z_\Pi)^k]_{\mathcal{B}_i}^{\mathcal{B}_i'} = \begin{pmatrix} T_{2k+1}(\lambda_i) & -\sqrt{1-\lambda_i^2} U_{2k}(\lambda_i) \\ \sqrt{1-\lambda_i^2} U_{2k}(\lambda_i) & T_{2k+1}(\lambda_i) \end{pmatrix}. \tag{7.89}$$

Using the fact that $\mathcal{B}_i, \mathcal{B}_i'$ share the common basis $|0^m\rangle |v_i\rangle$, we still have the block-encoding

$$U_A Z_\Pi (U_A^\dagger Z_\Pi U_A Z_\Pi)^k = \begin{pmatrix} T_{2k+1}(A) & * \\ * & * \end{pmatrix}. \tag{7.90}$$

Therefore $U_A Z_\Pi (U_A^\dagger Z_\Pi U_A Z_\Pi)^k$ is a $(1, m)$-block-encoding of $T_{2k+1}(A)$.

In summary, the block-encoding of $T_l(A)$ is given by applying $U_A Z_\Pi$ and $U_A^\dagger Z_\Pi$ alternately. If $l = 2k$, then there are exactly k such pairs. The quantum circuit for each pair $\widetilde{O}$ is

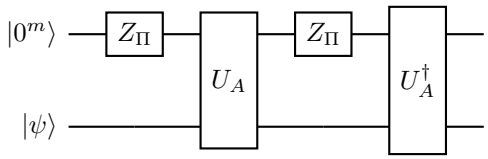


FIGURE 7.6. Circuit implementing one step of qubitization with a general block encoding of a Hermitian matrix. This block encodes $T_2(A)$. Here $U_A \in \mathrm{BE}_{1,m}(A)$.

Otherwise if $l = 2k + 1$, then there is an extra $U_A Z_\Pi$. The effect is to map each eigenvector $|0^m\rangle |v_i\rangle$ back and forth between the two-dimensional subspaces $\mathcal{H}_i$ and $\mathcal{H}_i'$. In Section 7.6, we shall see that the separate treatment even/odd polynomials will play a more prominent role.

Now that we have obtained $O_k \in \mathrm{BE}_{1,m}(T_k(A))$ for all k, we can use the LCU lemma again to construct block encodings for linear combination of Chebyshev polynomials. We omit the details here.

7.5. Quantum eigenvalue transformation

Let us briefly recap what we have done so far: (1) construct a block encoding of an Hermitian matrix A (the block encoding matrix itself can be non-Hermitian); (2) use qubitization to block encode $T_k(A)$; (3) use LCU to block encode an arbitrary polynomial function of A (up to a subnormalization factor). This framework is conceptually appealing, but the practical implementation of the select and prepare oracles are by no means straightforward, and can come at significant costs.

7.5.1. Hermitian block encoding. Note that $\mathrm{i}Z = e^{\mathrm{i}\frac{\pi}{2}Z}$, if A is given by a Hermitian block encoding U_A, the block encoding of the Chebyshev polynomial in Section 7.1 can be written as

$$O^d = (-\mathrm{i})^d \prod_{j=1}^{d} (U_A e^{\mathrm{i}\frac{\pi}{2}Z_\Pi}). \tag{7.91}$$

This is a special case of the following representation. Note that $(-\mathrm{i})^d$ is an irrelevant phase factor and can be discarded.

$$U_{\widetilde{\Phi}} = e^{\mathrm{i}\widetilde{\phi}_0 Z_\Pi} \prod_{j=1}^{d} (U_A e^{\mathrm{i}\widetilde{\phi}_j Z_\Pi}). \tag{7.92}$$

The representation Eq. (7.92) is called a quantum eigenvalue transformation (QET).

Due to qubitization, $U_{\widetilde{\Phi}}$ should block encode some matrix polynomial of A. We first state the following theorem without proof.

THEOREM 7.16 (Quantum signal processing, a simplified version). *Consider*

$$U_A = \begin{pmatrix} x & \sqrt{1-x^2} \\ \sqrt{1-x^2} & -x \end{pmatrix}. \tag{7.93}$$

For any $\widetilde{\Phi} := (\widetilde{\phi}_0, \cdots, \widetilde{\phi}_d) \in \mathbb{R}^{d+1}$,

$$U_{\widetilde{\Phi}} := e^{\mathrm{i}\widetilde{\phi}_0 Z} \prod_{j=1}^{d} (U_A e^{\mathrm{i}\widetilde{\phi}_j Z}) = \begin{pmatrix} P(x) & * \\ * & * \end{pmatrix}, \tag{7.94}$$

where $P \in \mathbb{C}[x]$ *satisfy*

(1) $\deg(P) \le d$,
(2) P *has parity* ($d \bmod 2$),
(3) $|P(x)| \le 1, \forall x \in [-1, 1]$.

Also define $-\widetilde{\Phi} := (-\widetilde{\phi}_0, \cdots, -\widetilde{\phi}_d) \in \mathbb{R}^{d+1}$, *then*

$$U_{-\widetilde{\Phi}} := e^{-\mathrm{i}\widetilde{\phi}_0 Z} \prod_{j=1}^{d} (U_A e^{-\mathrm{i}\widetilde{\phi}_j Z}) = U_{\widetilde{\Phi}}^* = \begin{pmatrix} P^*(x) & * \\ * & * \end{pmatrix}. \tag{7.95}$$

Here $P^*(x)$ *is the complex conjugation of* $P(x)$, *and* $U_{\widetilde{\Phi}}^*$ *is the complex conjugation (without transpose) of* $U_{\widetilde{\Phi}}$.

Remark 7.17. This theorem can be proved inductively. However, this is a special case of the quantum signal processing in Theorem 7.20, so we will omit the proof here. In fact, Theorem 7.20 will also state the converse of the result, which describes precisely the class of matrix polynomials that can be described by such phase factor modulations. In Theorem 7.16, the condition (1) states that the polynomial degree is upper bounded by the number of U_A's, and the condition (3) is simply

a consequence of that $U_{\widetilde{\Phi}}$ is a unitary matrix. The condition (2) is less obvious, but should not come at a surprise, since we have seen the need of treating even and odd polynomials separately in the case of qubitization with a general block encoding. Eq. (7.95) can be proved directly by taking the complex conjugation of $U_{\widetilde{\Phi}}$. ◇

Following the qubitization procedure, we immediately have Theorem 7.18.

THEOREM 7.18 (Quantum eigenvalue transformation with Hermitian block encoding). *Let* $U_A \in \mathrm{HBE}_{1,m}(A)$. *Then for any* $\widetilde{\Phi} := (\widetilde{\phi}_0, \cdots, \widetilde{\phi}_d) \in \mathbb{R}^{d+1}$,

$$U_{\widetilde{\Phi}} = e^{\mathrm{i}\widetilde{\phi}_0 Z_\Pi} \prod_{j=1}^{d} (U_A e^{\mathrm{i}\widetilde{\phi}_j Z_\Pi}) = \begin{pmatrix} P(A) & * \\ * & * \end{pmatrix} \in \mathrm{BE}_{1,m}(P(A)), \tag{7.96}$$

where $P \in \mathbb{C}[x]$ *satisfies the requirements in Theorem 7.16.*

Using Theorem 7.18, we may construct the block encoding of a matrix polynomial without invoking LCU. The cost is essentially the same as block encoding a Chebyshev polynomial.

In order to implement $e^{\mathrm{i}\phi Z_\Pi}$, we note that the quantum circuit denoted by CR_ϕ is in Fig. 7.7 returns $e^{\mathrm{i}\phi} |0\rangle |0^m\rangle$ if $b = 0^m$, and $e^{-\mathrm{i}\phi} |0\rangle |b\rangle$ if $b \neq 0^m$. So omitting the signal qubit, this is precisely $e^{\mathrm{i}\phi Z_\Pi}$.

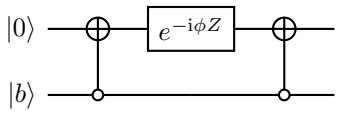


FIGURE 7.7. Implementing the controlled rotation circuit for quantum eigenvalue transformation.

Therefore, if A is given by a Hermitian block encoding U_A, we can follow the argument in Section 7.1 and construct the following unitary The corresponding quantum circuit is in Fig. 7.8, which uses one extra ancilla qubit. When measuring the $(m+1)$ ancilla qubits and obtain $|0\rangle |0^m\rangle$, the corresponding (unnormalized) state in the system register is $P(A) |\psi\rangle$.

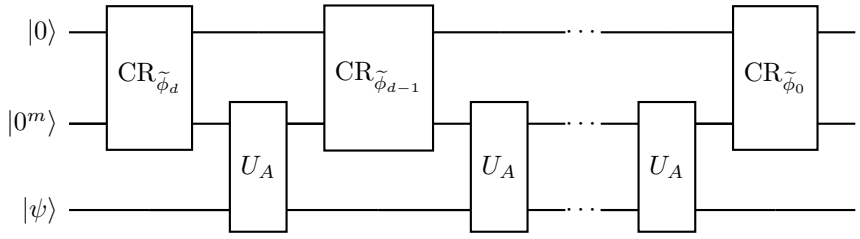


FIGURE 7.8. Circuit of quantum eigenvalue transformation to construct $U_{P(A)} \in \mathrm{BE}_{1,m+1}(P(A))$, using $U_A \in \mathrm{HBE}_{1,m}(A)$.

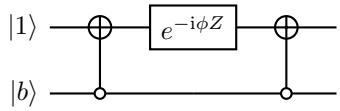
The QET described by the circuit in Fig. 7.8 generally constructs a block encoding of $P(A)$ for some complex polynomial P. In practical applications (such as those later in this chapter),

we would like to construct a block encoding of $P_{\mathrm{Re}}(A) \equiv (\mathrm{Re}\, P)(A) = \frac{1}{2}(P(A) + P^*(A))$ instead. Below we demonstrate that a simple modification of Fig. 7.8 allows us to achieve this goal.

To this end, we use Eq. (7.95). Qubitization allows us to construct

$$U_{-\widetilde{\Phi}} = e^{-\mathrm{i}\widetilde{\phi}_0 Z_\Pi} \prod_{j=1}^{d} (U_A e^{-\mathrm{i}\widetilde{\phi}_j Z_\Pi}) = \begin{pmatrix} P^*(A) & * \\ * & * \end{pmatrix}. \tag{7.97}$$

So all we need is to negate all phase factors in $\widetilde{\Phi}$. In order to implement $\mathrm{CR}_{-\phi}$, we do not actually need to implement a new circuit. Instead we may simply change the signal qubit from $|0\rangle$ to $|1\rangle$:



which returns $e^{-\mathrm{i}\phi} |1\rangle |0^m\rangle$ if $b = 0^m$, and $e^{\mathrm{i}\phi} |1\rangle |b\rangle$ if $b \neq 0^m$. In other words, the circuit for $U_{P^*(A)}$ and $U_{P(A)}$ are *exactly the same* except that the input signal qubit is changed from $|0\rangle$ to $|1\rangle$.

Now we claim the circuit in Fig. 7.9 implements a block encoding $U_{P_{\mathrm{Re}}(A)} \in \mathrm{BE}_{1,m+1}(P_{\mathrm{Re}}(A))$. This circuit can be viewed as an implementation of the linear combination of unitaries $\frac{1}{2}(U_{P^*(A)} + U_{P(A)})$.

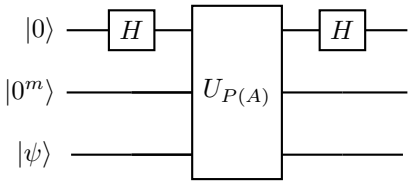


FIGURE 7.9. Circuit of quantum eigenvalue transformation for constructing a $(1, m+1)$-block-encoding of $P_{\mathrm{Re}}(A)$.

To verify this, we may evaluate

$$\begin{aligned} |0\rangle |0^m\rangle |\psi\rangle &\xrightarrow{H\otimes I_{m+n}} \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) |0^m\rangle |\psi\rangle \\ &\xrightarrow{U_{P(A)}} \frac{1}{\sqrt{2}} |0\rangle (|0^m\rangle P(A) |\psi\rangle + |\perp\rangle) + \frac{1}{\sqrt{2}} |1\rangle (|0^m\rangle P^*(A) |\psi\rangle + |\perp'\rangle) \\ &\xrightarrow{H\otimes I_{m+n}} |0\rangle \left(|0^m\rangle \frac{P(A) + P^*(A)}{2} |\psi\rangle \right) + |\widetilde{\perp}\rangle \\ &= |0\rangle |0^m\rangle P_{\mathrm{Re}}(A) |\psi\rangle + |\widetilde{\perp}\rangle \end{aligned} \tag{7.98}$$

Here $|\perp\rangle, |\perp'\rangle$ are two $(m+n)$-qubit state orthogonal to any state $|0^m\rangle |x\rangle$, while $|\widetilde{\perp}\rangle$ is a $(m+n+1)$-qubit state orthogonal to any state $|0\rangle |0^m\rangle |x\rangle$. In other words, by measuring all $(m+1)$ ancilla qubits and obtain 0^{m+1}, the corresponding (unnormalized) state in the system register is $P_{\mathrm{Re}}(A) |\psi\rangle$.

7.5.2. General block encoding. If A is given by a general block encoding U_A, the quantum eigenvalue transformation should consist of an alternating sequence of $U_A, U_A^\dagger$ gates. The circuit is given by Fig. 7.10, and the corresponding block encoding is described in Theorem 7.19. Note that the Hermitian block encoding becomes a special case with $U_A = U_A^\dagger$.

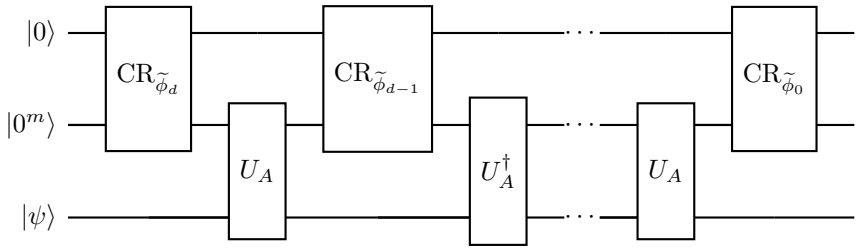


FIGURE 7.10. Circuit of quantum eigenvalue transformation to construct $U_{P(A)} \in \mathrm{BE}_{1,m+1}(P(A))$, using $U_A \in \mathrm{BE}_{1,m}(A)$. Here $U_A, U_A^\dagger$ should be applied alternately. When d is even, the last U_A gate should be replaced $U_A^\dagger$.

THEOREM 7.19 (Quantum eigenvalue transformation with general block encoding). *Let $U_A \in \mathrm{BE}_{1,m}(A)$. Then for any $\widetilde{\Phi} := (\widetilde{\phi}_0, \cdots, \widetilde{\phi}_d) \in \mathbb{R}^{d+1}$, let*

$$U_{\widetilde{\Phi}} = e^{\mathrm{i}\widetilde{\phi}_0 Z_\Pi} \prod_{j=1}^{d/2} \left[U_A^\dagger e^{\mathrm{i}\widetilde{\phi}_{2j-1} Z_\Pi} U_A e^{\mathrm{i}\widetilde{\phi}_{2j} Z_\Pi} \right] \tag{7.99}$$

when d is even, and

$$U_{\widetilde{\Phi}} = (-\mathrm{i})^d e^{\mathrm{i}\widetilde{\phi}_0 Z_\Pi} (U_A e^{\mathrm{i}\widetilde{\phi}_1 Z_\Pi}) \prod_{j=1}^{(d-1)/2} \left[U_A^\dagger e^{\mathrm{i}\widetilde{\phi}_{2j} Z_\Pi} U_A e^{\mathrm{i}\widetilde{\phi}_{2j+1} Z_\Pi} \right] \tag{7.100}$$

when d is odd. Then

$$U_{\widetilde{\Phi}} = \begin{pmatrix} P(A) & * \\ * & * \end{pmatrix} \in \mathrm{BE}_{1,m}(P(A)), \tag{7.101}$$

where $P \in \mathbb{C}[x]$ satisfy the conditions in Theorem 7.16.

Following exactly the same procedure, we find that the circuit in Fig. 7.9, with $U_{P(A)}$ given by Fig. 7.10 implements a $U_{P_{\mathrm{Re}}(A)} \in \mathrm{BE}_{1,m+1}(P_{\mathrm{Re}}(A))$. This is left as an exercise.

7.5.3. General matrix polynomials. In practical applications, we may be interested in matrix polynomials $f(A)$, where $f(x) \in \mathbb{R}[x]$ does not have a definite parity. This violates the parity requirement of Theorem 7.16. This can be solved by using the LCU technique.

Note that

$$f(x) = f_{\mathrm{even}}(x) + f_{\mathrm{odd}}(x), \tag{7.102}$$

where $f_{\mathrm{even}}(x) = \frac{1}{2}(f(x) + f(-x)), f_{\mathrm{odd}}(x) = \frac{1}{2}(f(x) - f(-x))$. If $|f(x)| \le 1$ on $[-1, 1]$, then $|f_{\mathrm{even}}(x)|, |f_{\mathrm{odd}}(x)| \le 1$ on $[-1, 1]$, and $f_{\mathrm{even}}(x)$, $f_{\mathrm{odd}}(x)$ can be each constructed using the circuit in Fig. 7.9. Introducing another ancilla qubit and using the LCU technique, we obtain a $(1, m+2)$-block-encoding of $(f_{\mathrm{even}}(A) + f_{\mathrm{odd}}(A))/2$. In other words, we obtain a circuit $U_f \in \mathrm{BE}_{2,m+2}(f(A))$.

Note that unlike the case of the block encoding of $P_{\mathrm{Re}}(A)$, we lose a subnormalization factor of 2 here.

Following the same principle, if $f(x) = g(x) + \mathrm{i}h(x) \in \mathbb{C}[x]$ is a given complex polynomial, and $g, h \in \mathbb{R}[x]$ do not have a definite parity, we can construct $U_{g(A)} \in \mathrm{BE}_{2,m+2}(g(A)), U_{h(A)} \in \mathrm{BE}_{2,m+2}(h(A))$. Then applying another layer of LCU, we obtain $U_{f(A)} \in \mathrm{BE}_{4,m+3}(f(A))$.

On the other hand, if the real and imaginary parts g, h have definite parity, then $U_{g(A)} \in \mathrm{BE}_{1,m+1}(g(A)), U_{h(A)} \in \mathrm{BE}_{1,m+1}(h(A))$. Applying LCU, we obtain $U_{f(A)} \in \mathrm{BE}_{2,m+2}(f(A))$.

The construction circuits in the cases above is left as an exercise.

7.6. Quantum signal processing

In terms of implementing matrix polynomials of Hermitian matrices, quantum eigenvalue transform provides a much simpler circuit than the method based on LCU and qubitization (i.e., linear combination of Chebyshev polynomials). The simplification is clear both in terms of the number of ancilla qubits and of the circuit architecture. However, it is not clear so far for which polynomials (either a complex polynomial $P \in \mathbb{C}[x]$ or a real polynomial $P_{\mathrm{Re}} \in \mathbb{R}[x]$) we can apply the QET technique, and how to obtain the phase factors. Quantum signal processing (QSP) provides a complete answer to this question.

Due to qubitization, all these questions can be answered in the context of SU(2) matrices. QSP is the theory of QET for SU(2) matrices, or the *unitary representation* of a scalar (real or complex) polynomial $P(x)$. Let $A = x \in [-1, 1]$ be a scalar with a one-qubit Hermitian block encoding

$$U_A(x) = \begin{pmatrix} x & \sqrt{1-x^2} \\ \sqrt{1-x^2} & -x \end{pmatrix}. \tag{7.103}$$

Then

$$O(x) = U_A(x)Z = \begin{pmatrix} x & -\sqrt{1-x^2} \\ \sqrt{1-x^2} & x \end{pmatrix} \tag{7.104}$$

is a rotation matrix.

Similar to Eq. (7.92), the QSP representation takes the following form

$$U(x) = e^{\mathrm{i}\phi_0 Z} O(x) e^{\mathrm{i}\phi_1 Z} O(x) \cdots e^{\mathrm{i}\phi_{d-1} Z} O(x) e^{\mathrm{i}\phi_d Z}. \tag{7.105}$$

By setting $\phi_0 = \cdots = \phi_d = 0$, we immediately obtain the block encoding of the Chebyshev polynomial $T_d(x)$. The representation power of this formulation is characterized by Theorem 7.20, which is based on slight modification of [GSLW19, Theorem 4]. In the following discussion, even functions have parity 0 and odd functions have parity 1.

THEOREM 7.20 (Quantum signal processing). *There exists a set of phase factors* $\Phi := (\phi_0, \cdots, \phi_d) \in \mathbb{R}^{d+1}$ *such that*

$$U_\Phi(x) = e^{\mathrm{i}\phi_0 Z} \prod_{j=1}^{d} \left[O(x) e^{\mathrm{i}\phi_j Z} \right] = \begin{pmatrix} P(x) & -Q(x)\sqrt{1-x^2} \\ Q^*(x)\sqrt{1-x^2} & P^*(x) \end{pmatrix} \tag{7.106}$$

if and only if $P, Q \in \mathbb{C}[x]$ *satisfy*

(1) $\deg(P) \le d, \deg(Q) \le d-1$,
(2) P *has parity* $d \bmod 2$ *and* Q *has parity* $d-1 \bmod 2$, *and*
(3) $|P(x)|^2 + (1-x^2)|Q(x)|^2 = 1, \forall x \in [-1, 1]$.

Here $\deg Q = -1$ *means* $Q = 0$.

PROOF. $\Rightarrow$:

Since both $e^{\mathrm{i}\phi Z}$ and $O(x)$ are unitary, the matrix $U_\Phi(x)$ is always a unitary matrix, which immediately implies the condition (3). Below we only need to verify the conditions (1), (2).

When $d=0$, $U_\Phi(x)=e^{\mathrm{i}\phi_0 Z}$, which gives $P(x)=e^{\mathrm{i}\phi_0}$ and $Q=0$, satisfying all three conditions. For induction, suppose $U_{(\phi_0,\cdots,\phi_{d-1})}(x)$ takes the form in Eq. (7.106) with degree $d-1$, then for any $\phi\in\mathbb{R}$, we have

(7.107)
$$\begin{aligned}U_{(\phi_0,\cdots,\phi_{d-1})}(x) &= \begin{pmatrix} P(x) & -Q(x)\sqrt{1-x^2} \\ Q^*(x)\sqrt{1-x^2} & P^*(x)\end{pmatrix}\begin{pmatrix} x & -\sqrt{1-x^2} \\ \sqrt{1-x^2} & x\end{pmatrix}\begin{pmatrix} e^{\mathrm{i}\phi} & 0 \\ 0 & e^{-\mathrm{i}\phi}\end{pmatrix}\\ &= \begin{pmatrix} xP(x)-(1-x^2)Q(x) & -\sqrt{1-x^2}(P(x)+xQ(x)) \\ -\sqrt{1-x^2}(P^*(x)+xQ^*(x)) & xP^*(x)-(1-x^2)Q^*(x)\end{pmatrix}\begin{pmatrix} e^{\mathrm{i}\phi} & 0 \\ 0 & e^{-\mathrm{i}\phi}\end{pmatrix}\\ &= \begin{pmatrix} e^{\mathrm{i}\phi}(xP(x)-(1-x^2)Q(x)) & e^{-\mathrm{i}\phi}(-\sqrt{1-x^2}(P(x)+xQ(x))) \\ e^{\mathrm{i}\phi}((-\sqrt{1-x^2}(P^*(x)+xQ^*(x))) & e^{\mathrm{i}\phi}(xP^*(x)-(1-x^2)Q^*(x))\end{pmatrix}.\end{aligned}$$

Therefore $U_{(\phi_0,\cdots,\phi_{d-1},\phi)}(x)$ satisfies conditions (1),(2).

$\Leftarrow$:

When $d=0$, the only possibility is $P(x)=e^{\mathrm{i}\phi_0}$ and $Q=0$, which satisfies Eq. (7.106).

For $d>0$, when d is even we may still have $\deg P=0$, i.e., $P(x)=e^{\mathrm{i}\phi_0}$ and $Q=0$. In this case, note that

$$O^{-1}(x)=O^\dagger(x)=\begin{pmatrix} x & \sqrt{1-x^2} \\ -\sqrt{1-x^2} & x\end{pmatrix}=e^{-\mathrm{i}\frac{\pi}{2}Z}O(x)e^{+\mathrm{i}\frac{\pi}{2}Z}, \tag{7.108}$$

we may set $\phi_j=(-1)^j\frac{\pi}{2}, j=1,\ldots,d$, and

$$e^{\mathrm{i}\phi_0 Z}\prod_{j=1}^{d}\left[O(x)e^{\mathrm{i}\phi_j Z}\right]=e^{\mathrm{i}\phi_0 Z}(O^\dagger(x)O(x))^{\frac{d}{2}}=e^{\mathrm{i}\phi_0 Z}. \tag{7.109}$$

Thus the statement holds.

Now given P,Q satisfying conditions (1)–(3), with $\deg P=\ell>0$, and $\ell\equiv d \pmod 2$. Then $\deg(|P(x)|^2)=2\ell>0$, and according to the condition (3) we must have $\deg(Q)=\ell-1$. Let P,Q be expanded as

$$P(x)=\sum_{k=0}^{\ell}\alpha_k x^k,\quad Q(x)=\sum_{k=0}^{\ell-1}\beta_k x^k, \tag{7.110}$$

then the leading term of $|P(x)|^2+(1-x^2)|Q(x)|^2$ is

$$|\alpha_\ell|^2 x^{2\ell}-x^2|\beta_{\ell-1}|^2 x^{2\ell-2}=(|\alpha_\ell|^2-|\beta_{\ell-1}|^2)x^{2\ell}=0, \tag{7.111}$$

which implies $|\alpha_\ell|=|\beta_{\ell-1}|$.

For any $\phi \in \mathbb{R}$, we have

$$\begin{aligned}
&\begin{pmatrix} P(x) & -Q(x)\sqrt{1-x^2} \\ Q^*(x)\sqrt{1-x^2} & P^*(x) \end{pmatrix} e^{-\mathrm{i}\phi Z} O^\dagger(x) \\
=&\begin{pmatrix} P(x) & -Q(x)\sqrt{1-x^2} \\ Q^*(x)\sqrt{1-x^2} & P^*(x) \end{pmatrix} \begin{pmatrix} e^{-\mathrm{i}\phi} & 0 \\ 0 & e^{\mathrm{i}\phi} \end{pmatrix} \begin{pmatrix} x & \sqrt{1-x^2} \\ -\sqrt{1-x^2} & x \end{pmatrix} \\
=&\begin{pmatrix} e^{-\mathrm{i}\phi}xP(x) + (1-x^2)Q(x)e^{\mathrm{i}\phi} & -\sqrt{1-x^2}(-e^{-\mathrm{i}\phi}P(x) + xQ(x)e^{\mathrm{i}\phi}) \\ \sqrt{1-x^2}(-e^{\mathrm{i}\phi}P^*(x) + xQ^*(x)e^{-\mathrm{i}\phi}) & e^{\mathrm{i}\phi}xP^*(x) + (1-x^2)Q^*(x)e^{-\mathrm{i}\phi} \end{pmatrix} \\
=:&\begin{pmatrix} \widetilde{P}(x) & -\widetilde{Q}(x)\sqrt{1-x^2} \\ \widetilde{Q}^*(x)\sqrt{1-x^2} & \widetilde{P}^*(x) \end{pmatrix}.
\end{aligned} \tag{7.112}$$

It may appear that $\deg \widetilde{P} = \ell + 1$. However, by properly choosing ϕ we may obtain $\deg \widetilde{P} = \ell - 1$. Let $e^{2\mathrm{i}\phi} = \alpha_\ell / \beta_{\ell-1}$. Then the coefficient of the $x^{\ell+1}$ term in $\widetilde{P}$ is

$$e^{-\mathrm{i}\phi}\alpha_\ell - e^{\mathrm{i}\phi}\beta_{\ell-1} = 0. \tag{7.113}$$

Similarly, the coefficient of the x^ℓ term in $\widetilde{Q}$ is

$$-e^{-\mathrm{i}\phi}\alpha_\ell + e^{\mathrm{i}\phi}\beta_{\ell-1} = 0. \tag{7.114}$$

The coefficient of the x^ℓ term in $\widetilde{P}$, and the coefficient of the $x^{\ell-1}$ term in $\widetilde{Q}$ are both 0 by the parity condition. So we have

(1) $\deg(\widetilde{P}) \le \ell - 1 \le d - 1, \deg(Q) \le \ell - 2 \le d - 2$,
(2) $\widetilde{P}$ has parity $d - 1 \bmod 2$ and $\widetilde{Q}$ has parity $d - 2 \bmod 2$, and
(3) $|\widetilde{P}(x)|^2 + (1-x^2)|\widetilde{Q}(x)|^2 = 1, \forall x \in [-1, 1]$.

Here the condition (3) is automatically satisfied due to unitarity. The induction follows until $\ell = 0$, and apply the argument in Eq. (7.109) to represent the remaining constant phase factor if needed. □

Remark 7.21 (W-convention of QSP)**.** [GSLW19, Theorem 4] is stated slightly differently as

$$U_{\Phi W}(x) = e^{\mathrm{i}\phi_0^W Z} \prod_{j=1}^{d} \left[W(x) e^{\mathrm{i}\phi_j^W Z} \right] = \begin{pmatrix} P(x) & \mathrm{i}Q(x)\sqrt{1-x^2} \\ \mathrm{i}Q^*(x)\sqrt{1-x^2} & P^*(x) \end{pmatrix}, \tag{7.115}$$

where

$$W(x) = e^{\mathrm{i}\arccos(x)X} = \begin{pmatrix} x & \mathrm{i}\sqrt{1-x^2} \\ \mathrm{i}\sqrt{1-x^2} & x \end{pmatrix}. \tag{7.116}$$

This will be referred to as the W-convention. Correspondingly Eq. (7.105) will be referred to as the O-convention. The two conventions can be easily converted into one another, due to the relation

$$W(x) = e^{-\mathrm{i}\frac{\pi}{4}Z} O(x) e^{+\mathrm{i}\frac{\pi}{4}Z}. \tag{7.117}$$

Correspondingly the relation between the phase angles using the O and W representations are related according to

$$\phi_j = \begin{cases} \phi_0^W - \frac{\pi}{4}, & j = 0, \\ \phi_j^W, & j = 1, \ldots, d-1, \\ \phi_d^W + \frac{\pi}{4}, & j = d, \end{cases} \tag{7.118}$$

On the other hand, note that for any $\theta \in \mathbb{R}$, $U_\Phi(x)$ and $e^{\mathrm{i}\theta Z}U_\Phi(x)e^{-\mathrm{i}\theta Z}$ both block encodes $P(x)$. Therefore WLOG we may as well take

$$\Phi = \Phi^W. \tag{7.119}$$

In many applications, we are only interested in $P \in \mathbb{C}[x]$, and $Q \in \mathbb{C}[x]$ is not provided *a priori*. [GSLW18, Theorem 4] states that under certain conditions P, the polynomial Q can always be constructed. We omit the details here. ◇

7.6.1. QSP for real polynomials. Note that the normalization condition (3) in Theorem 7.20 imposes very strong constraints on the coefficients of $P, Q \in \mathbb{C}[x]$. If we are only interested in QSP for real polynomials, the conditions can be significantly relaxed.

THEOREM 7.22 (Quantum signal processing for real polynomials). *Given a real polynomial* $P_{\mathrm{Re}}(x) \in \mathbb{R}[x]$, *and* $\deg P_{\mathrm{Re}} = d > 0$, *satisfying*

(1) P_{Re} *has parity* $d \bmod 2$,
(2) $|P_{\mathrm{Re}}(x)| \leq 1, \forall x \in [-1, 1]$,

then there exists polynomials $P(x), Q(x) \in \mathbb{C}[x]$ *with* $\mathrm{Re}\, P = P_{\mathrm{Re}}$ *and a set of phase factors* $\Phi := (\phi_0, \cdots, \phi_d) \in \mathbb{R}^{d+1}$ *such that the QSP representation Eq.* (7.106) *holds.*

Compared to Theorem 7.20, the conditions in Theorem 7.22 is much easier to satisfy: given any polynomial $f(x) \in \mathbb{R}[x]$ satisfying condition (1) on parity, we can always scale f to satisfy the condition (2) on its magnitude. Again the presentation is slightly modified compared to [GSLW19, Corollary 5]. We can now summarize the result of QET with real polynomials as follows.

Corollary 7.23 (Quantum eigenvalue transformation with real polynomials)**.** *Let* $A \in \mathbb{C}^{N\times N}$ *be encoded by its* $(1, m)$*-block-encoding* U_A. *Given a polynomial* $P_{\mathrm{Re}}(x) \in \mathbb{R}[x]$ *of degree* d *satisfying the conditions in Theorem 7.22, we can find a sequence of phase factors* $\Phi \in \mathbb{R}^{d+1}$, *so that the circuit in Fig. 7.9 denoted by* U_Φ *implements a* $(1, m+1)$*-block-encoding of* $P_{\mathrm{Re}}(A)$. U_Φ *uses* $U_A, U_A^\dagger$, m*-qubit controlled NOT, and single qubit rotation gates for* $\mathcal{O}(d)$ *times.*

Remark 7.24 (Relation between QSP representation and QET circuit)**.** Although $O(x) = U_A(x)Z$, we do not actually need to implement Z separately in QET. Note that $\mathrm{i}Z = e^{\mathrm{i}\frac{\pi}{2}Z}$, i.e., $Ze^{\mathrm{i}\phi Z} = (-\mathrm{i})e^{\mathrm{i}(\frac{\pi}{2}+\phi)Z}$, we obtain

$$U_\Phi(x) = e^{\mathrm{i}\phi_0 Z}\prod_{j=1}^{d}\left[O(x)e^{\mathrm{i}\phi_j Z}\right] = (-\mathrm{i})^d e^{\mathrm{i}\widetilde{\phi}_0 Z}\prod_{j=1}^{d}\left[U_A(x)e^{\mathrm{i}\widetilde{\phi}_j Z}\right], \tag{7.120}$$

where $\widetilde{\phi}_0 = \phi_0, \widetilde{\phi}_j = \phi_j + \pi/2, j = 1, \ldots, d$. For the purpose of block encoding $P(x)$, another equivalent, and more symmetric choice is

$$\widetilde{\phi}_j = \begin{cases} \phi_0 + \frac{\pi}{4}, & j = 0, \\ \phi_j + \frac{\pi}{2}, & j = 1, \ldots, d-1, \\ \phi_d + \frac{\pi}{4}, & j = d. \end{cases} \tag{7.121}$$

When the phase factors are given in the W-convention, since we can perform a similarity transformation and take $\Phi = \Phi^W$, we can directly convert Φ^W to $\widetilde{\Phi}$ according to Eq. (7.121), which is used in the QET circuit in Fig. 7.10. ◇

Example 7.25 (QSP for Chebyshev polynomial revisited)**.** In order to block encode the Chebyshev polynomial, we have $\phi_j = 0, j = 0, \ldots, d$. This gives $\widetilde{\phi}_0 = 0$, $\widetilde{\phi}_j = \pi/2, j = 1, \ldots, d$, and

$$U_\Phi(x) = [O(x)]^d = \begin{pmatrix} T_d(x) & -\sqrt{1-x^2}U_{d-1}(x) \\ \sqrt{1-x^2}U_{d-1}(x) & T_d(x) \end{pmatrix} = (-\mathrm{i})^d \prod_{j=1}^{d} \left[U_A(x) e^{\mathrm{i}\frac{\pi}{2}Z}\right]. \tag{7.122}$$

According to Eq. (7.121), an equivalent symmetric choice for block encoding $T_d(x)$ is

$$\widetilde{\phi}_j = \begin{cases} \frac{\pi}{4}, & j = 0, \\ \frac{\pi}{2}, & j = 1, \ldots, d-1, \\ \frac{\pi}{4}, & j = d. \end{cases} \tag{7.123}$$

◇

7.6.2. Optimization based method for finding phase factors. QSP for real polynomials is the most useful version for many problems in scientific computation. Let us now summarize the problem of finding phase factors following the W-convention and identify $\Phi = \Phi^W$.

Given a target polynomial $f = P_{\mathrm{Re}} \in \mathbb{R}[x]$ satisfying (1) $\deg(f) = d$, (2) the parity of f is $d \bmod 2$, (3) $\|f\|_\infty := \max_{x\in[-1,1]} |f(x)| < 1$, we would like to find phase factors $\Phi := (\phi_0, \cdots, \phi_d) \in [-\pi, \pi)^{d+1}$ so that

$$f(x) = g(x, \Phi) := \mathrm{Re}[U(x, \Phi)_{11}], \quad x \in [-1, 1], \tag{7.124}$$

with

$$U(x, \Phi) := e^{\mathrm{i}\phi_0 Z} W(x) e^{\mathrm{i}\phi_1 Z} W(x) \cdots e^{\mathrm{i}\phi_{d-1} Z} W(x) e^{\mathrm{i}\phi_d Z}. \tag{7.125}$$

Theorem 7.22 shows the existence of the phase factors. 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 paper, 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 $T_{2\widetilde{d}}(x)$. The QSP problem can be equivalently solved via the following optimization problem

$$\Phi^* = \operatorname*{arg\,min}_{\Phi\in[-\pi,\pi)^{d+1}} F(\Phi), \; F(\Phi) := \frac{1}{\widetilde{d}} \sum_{k=1}^{\widetilde{d}} |g(x_k, \Phi) - f(x_k)|^2, \tag{7.126}$$

i.e., any solution Φ^* to Eq. (7.124) achieves the global minimum of the cost function with $F(\Phi^*) = 0$, and vice versa.

However, note that the number of variables is larger than the number of equations and there should be an infinite number of global minima. [DMWL21, Theorem 2] shows that the existence of symmetric phase factors

$$\Phi = (\phi_0, \phi_1, \phi_2, \ldots, \phi_2, \phi_1, \phi_0) \in [-\pi, \pi)^{d+1}, \tag{7.127}$$

Then the optimization problem is changed to

$$\Phi^* = \operatorname*{arg\,min}_{\substack{\Phi\in[-\pi,\pi)^{d+1}, \\ \text{symmetric.}}} F(\Phi), \; F(\Phi) := \frac{1}{\widetilde{d}} \sum_{k=1}^{\widetilde{d}} |g(x_k, \Phi) - f(x_k)|^2, \tag{7.128}$$

This corresponds to choosing complementary polynomial $Q(x) \in \mathbb{R}[x]$. With the symmetric constraint taken into account, the number of variables matches the number of constraints.

Unfortunately, the energy landscape of the cost function $F(\Phi)$ is very complex, and has numerous global as well as local minima. Starting from a random initial guess, an optimization algorithm can easily be trapped at a local minima already when d is small. It is therefore surprising that starting from a special symmetric initial guess

$$\Phi^0 = (\pi/4, 0, 0, \ldots, 0, 0, \pi/4), \tag{7.129}$$

at least one global minimum can be robustly identified using standard unconstrained optimization algorithms even when d is as large as $10,000$ using standard double precision arithmetic operations [DMWL21], and the optimization method is observed to be 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 $f(x)$.

This optimization based method is implemented in QSPPACK[1].

Remark 7.26 (Other methods for treating QSP with real polynomials)**.** The proof of [GSLW19, Corollary 5] also gives a constructive algorithm for solving the QSP problem for real polynomials. Since $P_{\mathrm{Re}} = f \in \mathbb{R}[x]$ is given, the idea is to first find *complementary polynomials* $P_{\mathrm{Im}}, Q \in \mathbb{R}[x]$, so that the resulting $P(x) = P_{\mathrm{Re}}(x) + \mathrm{i}P_{\mathrm{Im}}(x)$ and $Q(x)$ satisfy the requirement in Theorem 7.20. Then the phase factors can be constructed following the recursion relation shown in the proof of Theorem 7.20. We will not describe the details of the procedure here. It is worth noting that the method is not numerically stable. This is made more precise by [Haa19] that these algorithms require $\mathcal{O}(d \log(d/\epsilon))$ bits of precision, where d is the degree of $f(x)$ and ϵ is the target accuracy. It is worth mentioning that the extended precision needed in these algorithms is not an artifact of the proof technique. For instance, for $d \approx 500$, the number of bits needed to represent each floating point number can be as large as $1000 \sim 2000$. In particular, such a task cannot be reliably performed using standard double precision arithmetic operations which only has 64 bits. ◇

7.6.3. A typical workflow preparing the circuit of QET. Let us now use $f(x) = \cos(xt)$ as in the Hamiltonian simulation to demonstrate a typical workflow of QSP. This function should be an even or odd function to satisfy the parity constraint (1) in Theorem 7.22.

(1) Expand $f(x), x \in [-1, 1]$ using a polynomial expansion (in this case, the Jacob–Anger expansion in Eq. (7.131)), and truncate it to some finite order.
(2) Scale the truncated polynomial by a suitable constant so that the resulting real polynomial $P_{\mathrm{Re}}(x)$ satisfies the maxnorm constraint (2) in Theorem 7.22.
(3) Use the optimization based method to find phase factors $\Phi^W = \Phi$, and convert the result to $\widetilde{\Phi}$ according to the relation in Eq. (7.121). $\widetilde{\Phi}$ can be directly used in the QET circuit in Figs. 7.9 and 7.10.

Remark 7.27. When the function $f(x)$ of interest has singularity on $[-1, 1]$, the function should first be mollified on a proper subinterval of interest, and then approximated by polynomials. A more streamlined method is to use the Remez exchange algorithm with parity constraint to directly approximate $f(x)$ on the subinterval. We refer readers to [DMWL21, Appendix E] for more details. ◇

7.7. Application: Time-independent Hamiltonian simulation

Using QET, let us now revisit the problem of time-independent Hamiltonian simulation problem $\mathcal{U} = e^{\mathrm{i}Ht}$. Instead of Trotter splitting, we assume that we are given access to a block encoding

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

$U_H \in \mathrm{BE}_{\alpha,m}(H)$. Since $e^{\mathrm{i}Ht} = e^{\mathrm{i}(H/\alpha)(\alpha t)}$, the subnormalization factor α can be factored into the simulation time t. So WLOG we assume $U_H \in \mathrm{BE}_{1,m}(H)$. Since

$$\mathcal{U} = \cos(Ht) + \mathrm{i}\sin(Ht), \tag{7.130}$$

and that $\cos(xt), \sin(xt)$ are even and odd functions, respectively, we can construct a block encoding for the real and imaginary part directly using the circuit for real matrix polynomials in Fig. 7.9.

More specifically, we first use the Fourier–Chebyshev series of the trigonometric functions given by the Jacobi-Anger expansion $[-1, 1]$:

$$\begin{aligned} \cos(tx) =& J_0(t) + 2\sum_{k=1}^{\infty}(-1)^k J_{2k}(t)T_{2k}(x), \\ \sin(tx) =& 2\sum_{k=0}^{\infty}(-1)^k J_{2k+1}(t)T_{2k+1}(x). \end{aligned} \tag{7.131}$$

Here $J_\nu(t)$ denotes Bessel functions of the first kind.

This series converges very rapidly. With

$$r = \Theta\left(t + \frac{\log(1/\epsilon)}{\log(e + \log(1/\epsilon)/t)}\right) \tag{7.132}$$

terms, the truncated Jacobi–Anger expansion with degree up to $d = 2r + 1$ can approximate $\cos(tx), \sin(tx)$ to precision $\epsilon/\sqrt{2}$, respectively [GSLW19, Corollary 32]. Such a scaling also matches the complexity lower bound for Hamiltonian simulation [BACS07, LC17b]. Define

$$C_d(x) = \frac{1}{\beta}J_0(t) + \frac{2}{\beta}\sum_{k=1}^{r}(-1)^k J_{2k}(t)T_{2k}(x), \quad S_d(x) = \frac{2}{\beta}\sum_{k=0}^{r}(-1)^k J_{2k+1}(t)T_{2k+1}(x), \tag{7.133}$$

where $\beta > 1$ is chosen so that $|C_d|, |S_d| \le 1$ on $[-1, 1]$, and β can be chosen to be as small as $1 + \epsilon$. Also let $f_d(x) = C_d(x) + \mathrm{i}S_d(x)$, then

$$\max_{x\in[-1,1]} \left|\beta f_d(x) - e^{\mathrm{i}tx}\right| \le \epsilon. \tag{7.134}$$

Theorem 7.22 guarantees the existence of phase factors $\widetilde{\Phi}_C, \widetilde{\Phi}_S$, using which we can construct $\mathcal{U}_C \in \mathrm{BE}_{1,m+1}(C_d(H))$ and $\mathcal{U}_s \in \mathrm{BE}_{1,m+1}(S_d(H))$. Finally, we can use one more ancilla qubit and LCU in Section 7.5.3 to construct a block encoding $\mathcal{U}_d \in \mathrm{BE}_{2,m+2}(f_d(H))$, or $\mathcal{U}_d \in \mathrm{BE}_{2\beta,m+2}(\mathcal{U}, \epsilon)$. The circuit depth is $\mathcal{O}(t + \log(1/\epsilon))$.

As an example, Fig. 7.11 shows the QSP representation of the quality of approximating $\cos(tx)$ using $P_{\mathrm{Re}}(x) = C_d(x)$ with $t = 4\pi, \beta = 1.001, d = 24$. The quality of the approximation can be significantly improved with a larger degree $d = 50$ (see Fig. 7.12). The phase factors are obtained via QSPPACK.

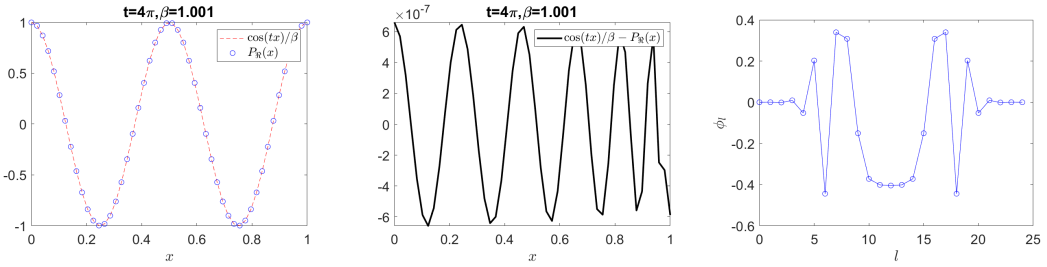


FIGURE 7.11. QSP representation of $\cos(tx)/\beta$ with $t = 4\pi, \beta = 1.001, d = 24$. The phase factors plotted removes a factor of $\pi/4$ on both ends (see Eq. (7.129)).

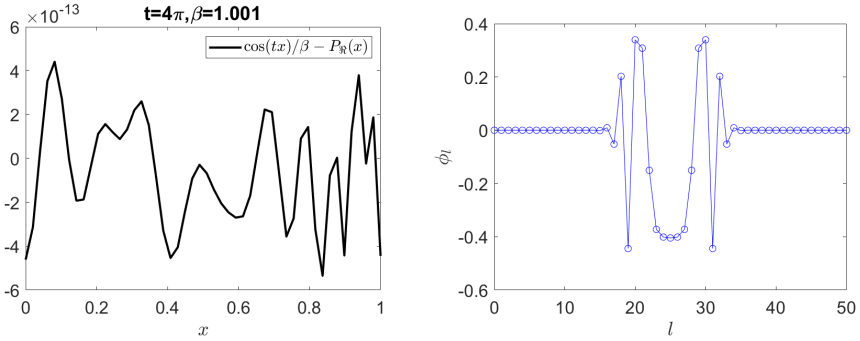


FIGURE 7.12. Error of the QSP representation of $\cos(tx)/\beta$ with $t = 4\pi, \beta = 1.001, d = 50$. The phase factors plotted removes a factor of $\pi/4$ on both ends (see Eq. (7.129)).

7.8. Application: Ground state preparation

Given a block encoding $U_H \in \mathrm{BE}_{1,m}(H)$, and WLOG assume $0 \preceq H \preceq 1$. We assume that we are provided an initial state $|\varphi\rangle$ so that the initial overlap $p_0 = |\langle\varphi|\psi_0\rangle|^2$ is not small. To simplify the problem we also assume that ground and the first excited state energies E_0, E_1 are known with a positive gap $\Delta := E_1 - E_0 > 0$. Our goal is to use QET to prepare an approximate quantum state $|\psi\rangle \approx |\psi_0\rangle$.

To this end, we can construct a threshold polynomial approximating the sign function on $[0, 1]$. Due to the assumption that $0 \preceq H \preceq 1$, this function can be chosen to be an even function. Since the sign function is a discontinuous, the polynomial should only aim at approximating the sign function outside $(\mu - \Delta/2, \mu + \Delta/2)$, where $\mu = (E_0 + E_1)/2$. The polynomial also needs to satisfy the conditions in Theorem 7.22. We need to find an even polynomial $P_{\mathrm{Re}}(x)$ satisfying

$$|P_{\mathrm{Re}}(x) - 1| \le \epsilon, \quad \forall x \in [0, \mu - \Delta/2]; \quad |P_{\mathrm{Re}}(x)| \le \epsilon, \quad \forall x \in [\mu + \Delta/2, 1]. \tag{7.135}$$

We can achieve this by approximating the sign function, with $\deg(P_{\mathrm{Re}}) = \mathcal{O}(\log(1/\epsilon)\Delta^{-1})$. (see e.g. [LC17a, Corollary 7] and [GSLW19, Corollary 16]). This construction is based on an approximation to the erf function. Fig. 7.13 gives a concrete construction of the even polynomial obtained via numerical optimization, and the phase factors are obtained via QSPPACK.

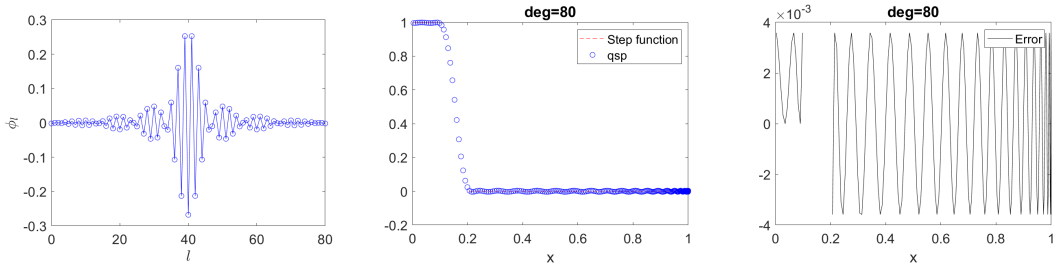


FIGURE 7.13. QSP representation for approximating a step function using an even polynomial on $[0, 0.1] \cup [0.2, 1]$. The phase factors plotted removes a factor of $\pi/4$ on both ends (see Eq. (7.129)).

Applying the circuit U_Φ to the initial state $|\varphi\rangle$, we have

$$U_\Phi |0^m\rangle |\varphi\rangle = \sqrt{p_0} |0^m\rangle P_{\mathrm{Re}}(H) |\psi_0\rangle + \sqrt{1-p_0} |0^m\rangle P_{\mathrm{Re}}(H) |\psi_\perp\rangle + |\perp\rangle . \tag{7.136}$$

Here $|\psi_\perp\rangle$ is a state in the system register orthogonal to $|\psi_0\rangle$, while $|\perp\rangle$ is orthogonal to all states $|0^m\rangle |\psi\rangle$. Note that

$$\|P_{\mathrm{Re}}(H) |\psi_\perp\rangle\| \leq \epsilon, \quad \|P_{\mathrm{Re}}(H) |\psi_0\rangle\| \geq 1 - \epsilon, \tag{7.137}$$

Therefore if we measure the ancilla qubits, the success probability of obtaining 0^m in the ancilla qubits, and the ground state $|\psi_0\rangle$ in the system register is at least $p_0(1-\epsilon)$. So the total number of queries to to U_A and $U_A^\dagger$ is $\mathcal{O}(\Delta^{-1} p_0^{-1} \log(1/\epsilon))$.

Using amplitude amplification, the number of repetitions can be reduced to $\mathcal{O}(\gamma^{-1})$, and the total number of queries to to U_A and $U_A^\dagger$ becomes $\mathcal{O}(\Delta^{-1} p_0^{-\frac{1}{2}} \log(1/\epsilon))$. This also matches the lower bound [LT20a].

Once the ground state is prepared, we can estimate the ground state energy by measuring the expectation value $\langle\psi_0|H|\psi_0\rangle$. The number of samples needed is $\mathcal{O}(1/\epsilon^2)$, which can be reduced to $\mathcal{O}(1/\epsilon)$ using amplitude amplification. In summary, the best complexity for estimating the ground state energy E_0 to accuracy ϵ is $\mathcal{O}(\Delta^{-1} p_0^{-\frac{1}{2}} \epsilon^{-1} \log(1/\epsilon))$. Note that the cost of estimating the ground state energy depends on the gap Δ. This is because the algorithm first prepares the ground state and then estimates the ground state energy. If we are only interested in estimating E_0 to precision ϵ, the gap dependence is not necessary (see Section 4.1 as well as [LT20a]).

Exercise 7.1. Let A, B be two n-qubit matrices. Construct a circuit to block encode $C = A + B$ with $U_A \in \mathrm{BE}_{\alpha_A, m}(A), U_B \in \mathrm{BE}_{\alpha_B, m}(B)$.

Exercise 7.2. Use LCU to construct a block encoding of the TFIM model with periodic boundary conditions in Eq. (4.2), with $g \neq 1$.

Exercise 7.3. Prove Proposition 7.3.

Exercise 7.4. Let A be an n-qubit Hermitian matrix. Write down the circuit for $U_{P_{\mathrm{Re}}(A)} \in \mathrm{BE}_{1,m+1}(P_{\mathrm{Re}}(A))$ with a block encoding $U_A \in \mathrm{BE}_{1,m}(A)$, where P is characterized by the phase sequence $\widetilde{\Phi}$ specified in Theorem 7.16.

Exercise 7.5. Write down the circuit for LCU of Hamiltonian simulation.

Exercise 7.6. Using QET to prepare the Gibbs state.

CHAPTER 8

Quantum singular value transformation

In Chapter 7, we have found that using qubitization, we can effectively block encode the Chebyshev matrix polynomial $T_k(A)$ for a Hermitian matrix A. Combined with LCU, we can construct a block encoding of any matrix polynomial of A. The process is greatly simplified using QSP and QET, which allows the implementation a general class of matrix functions for Hermitian matrices.

In this section, we generalize the results of qubitization and QET to general non-Hermitian matrices. This is called the quantum singular value transformation (QSVT). Throughout the chapter we assume $A \in \mathbb{C}^{N\times N}$ is a square matrix. QSVT is applicable to non-square matrices as well, and we will omit the discussions here.

8.1. Generalized matrix functions

For a square matrix $A \in \mathbb{C}^{N\times N}$, where for simplicity we assume $N = 2^n$ for some positive integer n, the singular value decomposition (SVD) of the normalized matrix A can be written as

$$A = W\Sigma V^\dagger, \tag{8.1}$$

or equivalently

$$A\left|v_i\right\rangle = \sigma_i \left|w_i\right\rangle, \quad A^\dagger \left|w_i\right\rangle = \sigma_i \left|v_i\right\rangle, \quad i \in [N]. \tag{8.2}$$

We may apply a function $f(\cdot)$ on its singular values and define the generalized matrix function [HBI73, ABF16] as below.

Definition 8.1 (Generalized matrix function [ABF16, Definition 4])**.** *Given $A \in \mathbb{C}^{N\times N}$ with singular value decomposition Eq. (8.1), and let $f : \mathbb{R} \to \mathbb{C}$ be a scalar function such that $f(\sigma_i)$ is defined for all $i \in [N]$. The generalized matrix function is defined as*

$$f^\diamond(A) := W f(\Sigma) V^\dagger, \tag{8.3}$$

where

$$f(\Sigma) = \operatorname{diag}\left(f(\sigma_0), f(\sigma_1), \ldots, f(\sigma_{N-1})\right).$$

Given the form in Eq. (8.2), we also define two other types of generalized matrix functions.

Definition 8.2. *Under the conditions in Definition 8.1, the left and right generalized matrix function are defined respectively as*

$$f^\triangleleft(A) := W f(\Sigma) W^\dagger, \quad f^\triangleright(A) := V f(\Sigma) V^\dagger. \tag{8.4}$$

Here the left and right pointing triangles reflects that the transformation only keeps the left and right singular vectors, respectively. For a given A, somewhat confusingly, in the discussion below, the transformation $f^{\triangleright}(A), f^{\triangleleft}(A), f^{\diamond}(A)$ will all be referred to as *singular value transformations*. In particular, QSVT mainly concerns $f^{\triangleright}(A), f^{\diamond}(A)$.

Proposition 8.3. *The following relations hold:*

$$f^{\diamond}(A^\dagger) = (f^{\diamond}(A))^\dagger, \quad f^{\triangleright}(A) = f^{\triangleleft}(A^\dagger), \tag{8.5}$$

and

$$f^{\triangleright}(A) = f^{\diamond}(\sqrt{A^\dagger A}) = f(\sqrt{A^\dagger A}), \quad f^{\triangleleft}(A) = f^{\diamond}(\sqrt{AA^\dagger}) = f(\sqrt{AA^\dagger}). \tag{8.6}$$

Proof. Just note that $A^\dagger A = V\Sigma^2 V^\dagger$, we have $\sqrt{A^\dagger A} = V\Sigma V^\dagger$. So the eigenvalue and singular value decomposition coincide for both $\sqrt{A^\dagger A}$ and $\sqrt{AA^\dagger}$. □

WLOG we assume access to $U_A \in \mathrm{BE}_{1,m}(A)$, so that the singular values of A are in $[0, 1]$, i.e.,

$$0 \le \sigma_i \le 1, \quad i \in [N]. \tag{8.7}$$

8.2. Qubitization of general matrices

In Section 7.4 we have observed that when A is a Hermitian matrix, the qubitization procedure introduces two different subspaces $\mathcal{H}_i$ and $\mathcal{H}_i'$ associated with each eigenvector $|v_i\rangle$. In particular, U_A maps $\mathcal{H}_i$ to $\mathcal{H}_i'$, and $U_A^\dagger$ maps $\mathcal{H}_i'$ to $\mathcal{H}_i$. Furthermore, both $\mathcal{H}_i$ and $\mathcal{H}_i'$ are the invariant subspaces of the projection operator Π. Therefore $\mathcal{H}_i$ is an invariant subspace of $U_A^\dagger f(\Pi) U_A$ for any function f. Much of the same structure can be carried out to the quantum singular value transformation. The only difference is that the qubitization is now defined with respect to the singular vectors. The procedure below almost entirely parallelizes that of Section 7.4, except that we need to work with the singular value decomposition instead of the eigenvalue decomposition.

Start from the SVD in Eq. (8.2), we apply U_A to $|0^m\rangle |v_i\rangle$ and obtain

$$U_A |0^m\rangle |v_i\rangle = \sigma_i |0^m\rangle |w_i\rangle + \sqrt{1-\sigma_i^2}\, |\perp_i'\rangle , \tag{8.8}$$

where $|\perp_i'\rangle$ is a normalized state satisfying $\Pi |\perp_i'\rangle = 0$.

Since U_A block encodes a matrix A, we have

$$U_A^\dagger = \begin{pmatrix} A^\dagger & * \\ * & * \end{pmatrix}, \tag{8.9}$$

which implies that there exists another normalized state $|\perp_i\rangle$ satisfying $\Pi |\perp_i\rangle = 0$ and

$$U_A^\dagger |0^m\rangle |w_i\rangle = \sigma_i |0^m\rangle |v_i\rangle + \sqrt{1-\sigma_i^2}\, |\perp_i\rangle . \tag{8.10}$$

Now apply U_A to both sides of Eq. (8.10), we obtain

$$|0^m\rangle |w_i\rangle = \sigma_i^2 |0^m\rangle |w_i\rangle + \sigma_i \sqrt{1-\sigma_i^2}\, |\perp_i'\rangle + \sqrt{1-\sigma_i^2} U_A |\perp_i\rangle , \tag{8.11}$$

which gives

$$U_A |\perp_i\rangle = \sqrt{1-\sigma_i^2}\, |0^m\rangle |w_i\rangle - \sigma_i |\perp_i'\rangle . \tag{8.12}$$

Define

$$\mathcal{B}_i = \{|0^m\rangle |v_i\rangle , |\perp_i\rangle\}, \quad \mathcal{B}_i' = \{|0^m\rangle |w_i\rangle , |\perp_i'\rangle\}, \tag{8.13}$$

and the associated two-dimensional subspaces $\mathcal{H}_i = \operatorname{span} B_i, \mathcal{H}'_i = \operatorname{span} B'_i$, we find that U_A maps $\mathcal{H}_i$ to $\mathcal{H}'_i$. Correspondingly $U_A^\dagger$ maps $\mathcal{H}'_i$ to $\mathcal{H}_i$.

Then Eqs. (8.8) and (8.12) give the matrix representation

$$[U_A]_{\mathcal{B}_i}^{\mathcal{B}'_i} = \begin{pmatrix} \sigma_i & \sqrt{1-\sigma_i^2} \\ \sqrt{1-\sigma_i^2} & -\sigma_i \end{pmatrix}. \tag{8.14}$$

Similar calculation shows that

$$[U_A^\dagger]_{\mathcal{B}'_i}^{\mathcal{B}_i} = \begin{pmatrix} \sigma_i & \sqrt{1-\sigma_i^2} \\ \sqrt{1-\sigma_i^2} & -\sigma_i \end{pmatrix}. \tag{8.15}$$

Meanwhile both $\mathcal{H}_i$ and $\mathcal{H}'_i$ are the invariant subspaces of the projector Π, with matrix representation

$$[\Pi]_{\mathcal{B}_i} = [\Pi]_{\mathcal{B}'_i} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}. \tag{8.16}$$

Therefore

$$[Z_\Pi]_{\mathcal{B}_i} = [Z_\Pi]_{\mathcal{B}'_i} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{8.17}$$

Hence $\mathcal{H}_i$ is an invariant subspace of $\widetilde{O} = U_A^\dagger Z_\Pi U_A Z_\Pi$, with matrix representation

$$[\widetilde{O}]_{\mathcal{B}_i} = \begin{pmatrix} \sigma_i & -\sqrt{1-\sigma_i^2} \\ \sqrt{1-\sigma_i^2} & \sigma_i \end{pmatrix}^2. \tag{8.18}$$

Repeating k times, we have

$$\begin{aligned} [\widetilde{O}^k]_{\mathcal{B}_i} =& (U_A^\dagger Z_\Pi U_A Z_\Pi)^k = \begin{pmatrix} \sigma_i & -\sqrt{1-\sigma_i^2} \\ \sqrt{1-\sigma_i^2} & \sigma_i \end{pmatrix}^{2k} \\ =& \begin{pmatrix} T_{2k}(\sigma_i) & -\sqrt{1-\sigma_i^2} U_{2k-1}(\sigma_i) \\ \sqrt{1-\sigma_i^2} U_{2k-1}(\sigma_i) & T_{2k}(\sigma_i) \end{pmatrix}. \end{aligned} \tag{8.19}$$

In other words,

$$\widetilde{O}^k = \begin{pmatrix} \sum_i v_i T_{2k}(\sigma_i) v_i^\dagger & * \\ * & * \end{pmatrix} = \begin{pmatrix} T_{2k}^{\triangleright}(A) & * \\ * & * \end{pmatrix}. \tag{8.20}$$

Therefore the circuit $(U_A^\dagger Z_\Pi U_A Z_\Pi)^k$ gives $(1, m)$-block-encoding of $T_{2k}^{\triangleright}(A)$.

Similarly,

$$[U_A Z_\Pi (U_A^\dagger Z_\Pi U_A Z_\Pi)^k]_{\mathcal{B}_i}^{\mathcal{B}'_i} = \begin{pmatrix} T_{2k+1}(\sigma_i) & -\sqrt{1-\sigma_i^2} U_{2k}(\sigma_i) \\ \sqrt{1-\sigma_i^2} U_{2k}(\sigma_i) & T_{2k+1}(\sigma_i) \end{pmatrix}. \tag{8.21}$$

In other words,

$$U_A Z_\Pi (U_A^\dagger Z_\Pi U_A Z_\Pi)^k = \begin{pmatrix} \sum_i w_i T_{2k+1}(\sigma_i) v_i^\dagger & * \\ * & * \end{pmatrix} = \begin{pmatrix} T_{2k+1}^{\diamond}(A) & * \\ * & * \end{pmatrix}. \tag{8.22}$$

Therefore the circuit $U_A Z_\Pi (U_A^\dagger Z_\Pi U_A Z_\Pi)^k$ gives $(1, m)$-block-encoding of $T_{2k+1}^{\diamond}(A)$.

Remark 8.4. By approximating any continuous function f using polynomials, and using the LCU lemma, we can approximately evaluate $f^{\diamond}(A)$ for any odd function f, and $f^{\triangleright}(A)$ for any even function f. This may seem somewhat restrictive. However, note that all singular values are non-negative. Hence when performing the polynomial approximation, if we are interested in $f^{\diamond}(A)$, we can always use first perform a polynomial approximation of an *odd extension* of f, i.e.,

$$g(x) = \begin{cases} f(x), & x > 0, \\ 0, & x = 0, \\ -f(-x), & x < 0, \end{cases} \tag{8.23}$$

and then evaluate $g^{\diamond}(A)$. Similarly, if we are interested in $f^{\triangleright}(A)$ for a general f, we can perform polynomial approximation to its *even extension*

$$g(x) = \begin{cases} f(x), & x \geq 0, \\ f(-x), & x < 0, \end{cases} \tag{8.24}$$

and evaluate $g^{\triangleright}(A)$. ◇

8.3. Quantum singular value transformation

8.3.1. Quantum circuit. Due to the close relation between eigenvalue and singular value transformation in terms of Chebyshev polynomials and qubitization in Section 8.2, we can obtain the QSVT circuit easily following the discussion in Section 7.6.

First, there are no changes to the scalar case of QSP (in terms of SU(2) matrices), and in particular Theorem 7.20 and Theorem 7.22.

For the matrix case, when d is even,

$$U_{\Phi} = (-\mathrm{i})^d e^{\mathrm{i}\widetilde{\phi}_0 Z_{\Pi}} \prod_{j=1}^{d/2} \left[U_A^{\dagger} e^{\mathrm{i}\widetilde{\phi}_{2j-1} Z_{\Pi}} U_A e^{\mathrm{i}\widetilde{\phi}_{2j} Z_{\Pi}} \right] \tag{8.25}$$

gives a $(1, m+1)$-block-encoding of $P^{\triangleright}(A)$ for some even polynomial $P \in \mathbb{C}[x]$.

When d is odd,

$$U_{\Phi} = (-\mathrm{i})^d e^{\mathrm{i}\widetilde{\phi}_0 Z_{\Pi}} (U_A e^{\mathrm{i}\widetilde{\phi}_1 Z_{\Pi}}) \prod_{j=1}^{(d-1)/2} \left[U_A^{\dagger} e^{\mathrm{i}\widetilde{\phi}_{2j} Z_{\Pi}} U_A e^{\mathrm{i}\widetilde{\phi}_{2j+1} Z_{\Pi}} \right] \tag{8.26}$$

gives a $(1, m+1)$-block-encoding of $P^{\diamond}(A)$ for some odd polynomial $P \in \mathbb{C}[x]$.

The quantum circuit is exactly the same as that in Fig. 7.10. The phase factors can be adjusted so that all polynomials P satisfying the conditions in Theorem 7.20 can be exactly represented. If we are only interested in some real polynomial $P_{\mathrm{Re}} \in \mathbb{R}[x]$ and $P_{\mathrm{Re}}^{\diamond}(A)$ (odd) and $P^{\triangleright}(A)$ (even), we can use Theorem 7.22 and the circuit in Fig. 8.1 (which is simply a combination of Figs. 7.9 and 7.10) to implement its $(1, m+1)$-block-encoding. We have the following theorem. Since the conditions of QSP representation for real polynomials is simple to satisfy and is also most useful in practice, we only state the case with real polynomials.

THEOREM 8.5 (Quantum singular value transformation with real polynomials). *Let $A \in \mathbb{C}^{N \times N}$ be encoded by its $(1, m)$-block-encoding U_A. Given a polynomial $P_{\mathrm{Re}}(x) \in \mathbb{R}[x]$ of degree d satisfying the conditions in Theorem 7.22, we can find a sequence of phase factors $\Phi \in \mathbb{R}^{d+1}$, so that the circuit in Fig. 8.1 denoted by U_{Φ} implements a $(1, m+1)$-block-encoding of $P_{\mathrm{Re}}^{\diamond}(A)$ if d is odd, and*

of $P^{\triangleright}_{\mathrm{Re}}(A)$ if d is even. U_Φ uses $U_A, U_A^\dagger$, m-qubit controlled NOT, and single qubit rotation gates for $\mathcal{O}(d)$ times.

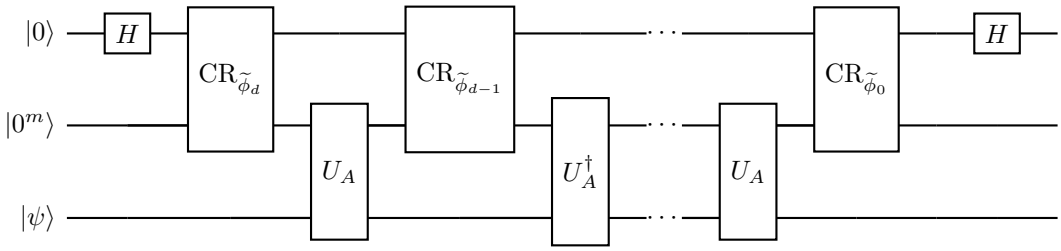


FIGURE 8.1. Circuit of quantum singular value transformation to construct $U_{P^{\diamond}_{\mathrm{Re}}} \in \mathrm{BE}_{1,m+1}(P^{\diamond}_{\mathrm{Re}}(A))$, using $U_A \in \mathrm{BE}_{1,m}(A)$. Here $U_A, U_A^\dagger$ should be applied alternately. When d is even, the last U_A gate should be replaced $U_A^\dagger$, and the circuit constructs $U_{P^{\triangleright}_{\mathrm{Re}}} \in \mathrm{BE}_{1,m+1}(P^{\triangleright}_{\mathrm{Re}}(A))$. This is simply a combination of Figs. 7.9 and 7.10.

8.3.2. QSVT applied to Hermitian matrices. When A is a Hermitian matrix, the quantum circuit for QET and QSVT are the same. This means that the eigenvalue transformation and the singular value transformation are merely two different perspectives of the same object.

For a Hermitian matrix A, the eigenvalue decomposition and the singular value decomposition are connected as

$$A = \sum_i |v_i\rangle \lambda_i \langle v_i| = \sum_i |\mathrm{sgn}(\lambda_i) v_i\rangle |\lambda_i| \langle v_i| := \sum_i |w_i\rangle \sigma_i \langle v_i| . \tag{8.27}$$

Here

$$|w_i\rangle = |\mathrm{sgn}(\lambda_i) v_i\rangle , \quad \sigma_i = |\lambda_i| . \tag{8.28}$$

So if $P \in \mathbb{C}[x]$ is an even polynomial,

$$P(A) = \sum_i |v_i\rangle P(\lambda_i) \langle v_i| = \sum_i |v_i\rangle P(|\lambda_i|) \langle v_i| = P^{\triangleright}(A). \tag{8.29}$$

Similarly, if $P \in \mathbb{C}[x]$ is an odd polynomial, then

$$P(A) = \sum_i |v_i\rangle P(\lambda_i) \langle v_i| = \sum_i |\mathrm{sgn}(\lambda_i) v_i\rangle P(|\lambda_i|) \langle v_i| = P^{\diamond}(A). \tag{8.30}$$

These relations indeed verify that the eigenvalue decomposition and singular value decomposition are indeed the same when P has a definite parity. When the parity of P is indefinite, the two objects are in general not the same, and in particular cannot be directly implemented using the QET circuit.

8.3.3. QSVT and matrix dilation. For general matrices, we have seen in the context of solving linear equations in Section 4.3 that the matrix dilation method in Eq. (4.33) can be used to convert the non-Hermitian problem to a Hermitian problem. Here we study the relation between QSP applied to the dilated Hermitian matrix, and QSVT for the general matrix.

Recall the definition of the dilated Hermitian matrix

$$\widetilde{A} = \begin{bmatrix} 0 & A^\dagger \\ A & 0 \end{bmatrix}. \tag{8.31}$$

When A is given by its block encoding $U_A \in \mathrm{BE}_{1,m}(A)$, the dilated Hermitian matrix $\widetilde{A}$ can be obtained with one ancilla qubit through $U_{\widetilde{A}} = |0\rangle\langle 1| \otimes U_A + |1\rangle\langle 0| \otimes U_A^\dagger$, i.e., $U_{\widetilde{A}} \in \mathrm{BE}_{1,m+1}(\widetilde{A})$. Note that this requires the controlled version of $U_A, U_A^\dagger$.

From the SVD in Eq. (8.2), we can construct

$$|z_i^\pm\rangle = \frac{1}{\sqrt{2}}(|0\rangle|v_i\rangle \pm |1\rangle|w_i\rangle). \tag{8.32}$$

Direct calculation shows

$$\widetilde{A}|z_i^\pm\rangle = \pm\sigma_i|z_i^\pm\rangle, \tag{8.33}$$

i.e., $\{|z_i^\pm\rangle\}$ are all the eigenvectors of $\widetilde{A}$.

For an arbitrary polynomial $f \in \mathbb{C}[x]$, the matrix function applied to $\widetilde{A}$ can be computed as

$$\begin{aligned} f(\widetilde{A}) &= \sum_i |z_i^+\rangle f(\sigma_i)\langle z_i^+| + |z_i^-\rangle f(-\sigma_i)\langle z_i^-| \\ &= \sum_i \begin{pmatrix} |v_i\rangle f_{\mathrm{even}}(\sigma_k)\langle v_i| & |v_i\rangle f_{\mathrm{odd}}(\sigma_k)\langle w_i| \\ |w_i\rangle f_{\mathrm{odd}}(\sigma_k)\langle v_i| & |w_i\rangle f_{\mathrm{even}}(\sigma_k)\langle w_i| \end{pmatrix} \\ &= \begin{pmatrix} f_{\mathrm{even}}^{\triangleright}(A) & f_{\mathrm{odd}}^{\diamond}(A^\dagger) \\ f_{\mathrm{odd}}^{\diamond}(A) & f_{\mathrm{even}}^{\triangleleft}(A) \end{pmatrix}. \end{aligned} \tag{8.34}$$

Here

$$f_{\mathrm{even}}(x) = \frac{1}{2}(f(x) + f(-x)), \quad f_{\mathrm{odd}}(x) = \frac{1}{2}(f(x) - f(-x)). \tag{8.35}$$

Therefore applying the QSP to the dilated matrix $\widetilde{A}$ automatically implements QSVT of A using polynomials of even and odd parities.

In particular, if f is an even function, then

$$f(\widetilde{A})|0\rangle|\psi\rangle = |0\rangle f^{\triangleright}(A)|\psi\rangle, \tag{8.36}$$

i.e., measuring the signal qubit we obtain 0 with certainty, and the system register is $f_{\mathrm{even}}^{\triangleright}(A)|\psi\rangle$. Similarly, if f is odd, then

$$f(\widetilde{A})|0\rangle|\psi\rangle = |1\rangle f^{\diamond}(A)|\psi\rangle, \tag{8.37}$$

i.e., measuring the signal qubit we obtain 1 with certainty.

8.4. Application: Solve linear systems of equations

In this section, we revisit the problem of solving linear systems of equations $Ax = b$. With QSVT we can solve QLSP for general matrices without the need of dilating the matrix into a Hermitian matrix. Assume $A = W\Sigma V^\dagger$ is invertible, i.e., $\forall i, \Sigma_{ii} > 0$, then

$$A^{-1} = V\Sigma^{-1}W^\dagger = f^{\diamond}(A^\dagger), \tag{8.38}$$

where $f(x) = x^{-1}$ is an odd function.

WLOG assume A can be accessed by $U_A \in \mathrm{BE}_{1,m}(A)$. For simplicity we also assume $\|A\| = 1$ (though in general $\|A\| \le \alpha = 1$, and we may not always be able to set $\|A\| = 1$). Let κ be

the condition number of A, then the singular values of A are contained in the interval $[\delta, 1]$, with $\delta = \kappa^{-1}$.

Note that $f(\cdot)$ is not bounded by 1 and in particular singular at $x = 0$. Therefore instead of approximating f on the whole interval $[-1, 1]$ we consider an odd polynomial $p(x)$ such that

$$\left| p(x) - \frac{\delta}{\beta x} \right| \leq \epsilon', \quad \forall x \in [-1, -\delta] \cup [\delta, 1]. \tag{8.39}$$

The β factor is chosen arbitrarily so that $|p(x)| \leq 1$ for all $x \in [-1, 1]$ to satisfy the requirement of the condition (2) in Theorem 7.22. For instance, we may choose $\beta = 4/3$. The precision parameter ϵ' will be chosen later. The degree of the odd polynomial can be chosen to be $\mathcal{O}(\frac{1}{\delta} \log(\frac{1}{\epsilon'}))$ is guaranteed by e.g. [GSLW18, Corollary 69]. This construction is not explicit (see an explicit construction in [CKS17]). Fig. 8.2 gives a concrete construction of an odd polynomial obtained via numerical optimization, and the phase factors are obtained via QSPPACK.

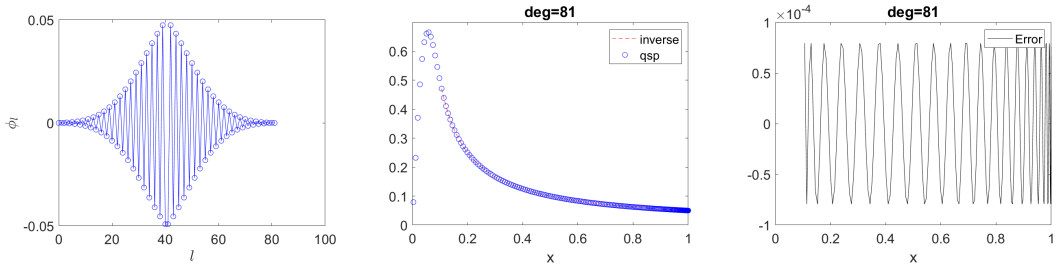


FIGURE 8.2. QSP representation of an odd polynomial approximating the inverse function on $[\kappa^{-1}, 1]$ with $\kappa = 10$. The phase factors plotted removes a factor of $\pi/4$ on both ends (see Eq. (7.129)).

Then Fig. 8.1 implements a $(1, m+1)$-block-encoding of $p^\diamond(A^\dagger) = Vp(\Sigma)W^\dagger$ denoted by U_Φ. We have

$$\|(p^\diamond(A^\dagger) - (\delta/\beta)A^{-1}\| = \|p(\Sigma) - (\delta/\beta)\Sigma^{-1}\| \leq \epsilon'. \tag{8.40}$$

The total number of queries to to U_A and $U_A^\dagger$ is

$$\mathcal{O}\left(\frac{1}{\delta} \log\left(\frac{1}{\epsilon'}\right)\right) = \mathcal{O}\left(\kappa \log\left(\frac{1}{\epsilon'}\right)\right). \tag{8.41}$$

To solve QLSP, we assume the right hand side vector $|b\rangle$ can be accessed through the oracle U_b such that

$$U_b |0^n\rangle = |b\rangle. \tag{8.42}$$

We introduce the parameter

$$\xi = \left\| A^{-1} |b\rangle \right\|, \tag{8.43}$$

which plays an important part in the success probability of the procedure. Let

$$x = (\delta/\beta)A^{-1} |b\rangle \tag{8.44}$$

be the unnormalized true solution, and the normalized solution state is $|x\rangle = x/\|x\|$. Now denote $\widetilde{x} = p^{\diamond}(A^{\dagger})|b\rangle$, and $|\widetilde{x}\rangle = \widetilde{x}/\|\widetilde{x}\|$. Then the unnormalized solution satisfies $\|x - \widetilde{x}\| \leq \epsilon'$. For the normalized state $|y\rangle$, this error is scaled accordingly. When $\epsilon' \ll \| |\widetilde{x}\rangle \|$, we have

$$\| |x\rangle - |x\rangle \| \approx \frac{\|\widetilde{x} - x\|}{\|\widetilde{x}\|} \leq \frac{\epsilon'}{\|\widetilde{x}\|}. \tag{8.45}$$

Also we have

$$\|\widetilde{x}\| = \left\| \frac{\delta}{\beta} A^{-1} |b\rangle \right\| = \frac{\delta\xi}{\beta} = \frac{\xi}{\beta\kappa}. \tag{8.46}$$

Therefore in order for the normalized output quantum state to be ϵ-close to the normalized solution state $|x\rangle$, we need to set $\epsilon' = \mathcal{O}(\epsilon\xi/\kappa)$. This is similar to the case of the HHL algorithm in Section 4.3, where QPE needs to achieve ϵ multiplicative accuracy, which means that the additive accuracy parameter ϵ' should be set to $\mathcal{O}(\epsilon/\kappa)$.

The success probability of the above procedure is $\Omega(\|\widetilde{x}\|^2) = \Omega(\xi^2/\kappa^2)$. With amplitude amplification we can boost the success probability to be greater than $1/2$ with one qubit serving as a witness, i.e., if measuring this qubit we get an outcome 0 it means the procedure has succeeded, and if 1 it means the procedure has failed. It takes $\mathcal{O}(\kappa/\xi)$ rounds of amplitude amplification, i.e., using $U_{\Phi}^{\dagger}$, U_{Φ}, U_b, and $U_b^{\dagger}$ for $\mathcal{O}(\kappa/\xi)$ times. A single U_{Φ} uses U_A and its inverse

$$\mathcal{O}\left(\frac{1}{\delta}\log\left(\frac{1}{\epsilon'}\right)\right) = \mathcal{O}\left(\kappa\log\left(\frac{\kappa}{\epsilon\xi}\right)\right) \tag{8.47}$$

times. Therefore the total number of queries to U_A and its inverse is

$$\mathcal{O}\left(\frac{\kappa^2}{\xi}\log\left(\frac{\kappa}{\xi\epsilon}\right)\right). \tag{8.48}$$

The number of queries to the U_b and its inverse is $\mathcal{O}(\kappa/\xi)$. We consider the following two cases for the magnitude of ξ.

(1) In general if no further promise is given, then $\xi \geq 1$. The total query complexity of U_A is therefore $\mathcal{O}(\kappa^2\log(\kappa/\epsilon))$. This is the typical complexity referred to in the literature.
(2) If $|b\rangle$ has a $\Omega(1)$ overlap with the left-singular vector of A with the smallest singular value, then $\xi = \Omega(\kappa)$. This is the best case scenario, and the total query complexity of U_A is $\mathcal{O}(\kappa\log(1/\epsilon))$, and the number of queries to the right hand side vector $|b\rangle$ is $\mathcal{O}(1)$, which is independent of the condition number.

8.5. Quantum singular value transformation with basis transformation*

So far we have assumed that we have a matrix A in mind, and the access to A is provided via the block encoding matrix U_A. The QSVT circuit takes the form

$$U_{\Phi} = \mathrm{CR}_{\widetilde{\phi}_0} \cdots \mathrm{CR}_{\widetilde{\phi}_{d-2}} U_A^{\dagger} \mathrm{CR}_{\widetilde{\phi}_{d-1}} U_A \mathrm{CR}_{\widetilde{\phi}_d}. \tag{8.49}$$

If we are further given two unitary matrices P, Q, we can equivalently rewrite the QSVT transformation as

$$U_{\Phi} = \cdots (Q^{\dagger}U_A^{\dagger}P)^{\dagger}(Q\,\mathrm{CR}_{\widetilde{\phi}_{d-1}}\,Q^{\dagger})(QU_AP^{\dagger})(P\,\mathrm{CR}_{\widetilde{\phi}_d}\,P^{\dagger})P. \tag{8.50}$$

The beginning of the equation ends with P or Q depending on the parity of d. The insertion of P, Q amounts to a *basis transformation*. Assume that we have access to $\widetilde{U}_A = QU_AP^{\dagger}$, then U_A is the matrix representation of $\widetilde{U}_A$ with respect to the bases given by P, Q, respectively. What is

different is that the controlled rotations before U_A and $U_A^\dagger$ are now expressed with respect to two different basis sets, i.e., $P\,\mathrm{CR}_{\widetilde{\phi}_d}\,P^\dagger$, and $Q\,\mathrm{CR}_{\widetilde{\phi}_{d-1}}\,Q^\dagger$, respectively. This can be useful for certain applications. Let us now express these ideas more formally.

Assume that we are given an $(n+m)$-qubit unitary $\widetilde{U}_A$, and two $(n+m)$-qubit projectors Π, Π'. For simplicity we assume $\mathrm{rank}(\Pi) = \mathrm{rank}(\Pi') = N$. Define an orthonormal basis set

$$\mathcal{B} = \{|\varphi_0\rangle, \ldots, |\varphi_{N-1}\rangle, |v_N\rangle, \ldots, |v_{NM-1}\rangle\}, \tag{8.51}$$

where the vectors $|\varphi_0\rangle, \ldots, |\varphi_{N-1}\rangle$ span the range of Π, and all states $|v_i\rangle$ are orthogonal to $|\varphi_j\rangle$. Similarly define an orthonormal basis set

$$\mathcal{B}' = \{|\psi_0\rangle, \ldots, |\psi_{N-1}\rangle, |w_N\rangle, \ldots, |w_{NM-1}\rangle\} \tag{8.52}$$

where the vectors $|\psi_0\rangle, \ldots, |\psi_{N-1}\rangle$ span the range of Π', and all states $|w_i\rangle$ are orthogonal to $|\psi_j\rangle$. We can think that the columns of $\mathcal{B}, \mathcal{B}'$ form the basis transformation matrix P, Q, respectively.

Then the matrix A is defined implicitly in terms of its matrix representation

$$[\widetilde{U}_A]_{\mathcal{B}}^{\mathcal{B}'} = U_A = \begin{pmatrix} A & * \\ * & * \end{pmatrix}. \tag{8.53}$$

Note that

$$[\Pi]_{\mathcal{B}}^{\mathcal{B}} = [\Pi']_{\mathcal{B}'}^{\mathcal{B}'} = \begin{pmatrix} I_n & 0 \\ 0 & 0 \end{pmatrix} = |0^m\rangle\langle 0^m| \otimes I_n, \tag{8.54}$$

we find that

$$\Pi' \widetilde{U}_A \Pi = \sum_{i,j\in[N]} |\psi_i\rangle\, A_{ij}\, \langle\varphi_j|\,. \tag{8.55}$$

Therefore Theorem 8.5 can be viewed as the singular value transformation of A, which is a submatrix of the *matrix representation* of U_A with respect to bases $\mathcal{B}, \mathcal{B}'$.

The implementation of the controlled rotation $P\,\mathrm{CR}_{\widetilde{\phi}}\,P^\dagger$ relies on the implementation of $\mathrm{C}_\Pi\,\mathrm{NOT}$. The projectors Π, Π' can be accessed directly, and WLOG we focus on one projector Π. Motivated from Grover's search, we may assume access to a reflection operator

$$R_\Pi = I_m - 2\Pi. \tag{8.56}$$

via the controlled NOT gates $\mathrm{C}_\Pi\,\mathrm{NOT}, \mathrm{C}_{\Pi'}\,\mathrm{NOT}$ respectively. We can then define an m-qubit controlled NOT gate as

$$\mathrm{C}_\Pi\,\mathrm{NOT} := X \otimes \Pi + I \otimes (I_m - \Pi), \tag{8.57}$$

which can be constructed using R_Π as

$$\begin{aligned} \mathrm{C}_\Pi\,\mathrm{NOT} &= X \otimes \frac{I_m - R_\Pi}{2} + I \otimes \frac{I_m + R_\Pi}{2} \\ &= \frac{I+X}{2} \otimes I_m + \frac{I-X}{2} \otimes R_\Pi \\ &= |+\rangle\langle +| \otimes I_m + |-\rangle\langle -| \otimes R_\Pi \\ &= (H \otimes I_m)(|0\rangle\langle 0| \otimes I_m + |1\rangle\langle 1| \otimes R_\Pi)(H \otimes I_m). \end{aligned} \tag{8.58}$$

Therefore assuming access to R_Π, the $\mathrm{C}_\Pi\,\mathrm{NOT}$ gate can be implemented using the circuit in Fig. 8.3.

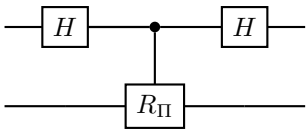


FIGURE 8.3. Circuit for implementing $\mathrm{C}_\Pi\,\mathrm{NOT}$ using a reflector R_Π.

Then according to Theorem 8.5, we can implement the QSVT using $\widetilde{U}_A, \widetilde{U}_A^\dagger$, $\mathrm{C}_{|0^m\rangle\langle 0^m|}\,\mathrm{NOT}$ and single qubit rotation gates, where $|0^m\rangle\langle 0^m| \otimes I_n$ is a rank-2^n projector. In this generalized setting, $\mathrm{C}_{|0^m\rangle\langle 0^m|\otimes I_n}\,\mathrm{NOT} = \mathrm{C}_{|0^m\rangle\langle 0^m|}\,\mathrm{NOT}\otimes I_n$ in the $\mathcal{B},\mathcal{B}'$ basis should be implemented using $\mathrm{C}_\Pi\,\mathrm{NOT}, \mathrm{C}_{\Pi'}\,\mathrm{NOT}$, respectively. We arrive at the following theorem:

THEOREM 8.6 (Quantum singular value transformation with real polynomials and projectors). *Let $\widetilde{U}_A$ be a $(n+m)$-qubit unitary, and Π, Π' be two $(n+m)$-qubit projectors of rank 2^n. Define the basis $\mathcal{B},\mathcal{B}'$ according to Eqs.* (8.51) *and* (8.52)*, and let $A \in \mathbb{C}^{N\times N}$ be defined in terms of the matrix representation in Eq.* (8.53)*. Given a polynomial $P_{\mathrm{Re}}(x) \in \mathbb{R}[x]$ of degree d satisfying the conditions in Theorem 7.22, we can find a sequence of phase factors $\Phi \in \mathbb{R}^{d+1}$ to define a unitary U_Φ satisfying*

$$U_\Phi\,|\varphi_j\rangle = \sum_{i\in[N]} |\psi_i\rangle\,[P^\diamond_{\mathrm{Re}}(A)]_{ij} + |\perp'_j\rangle\,, \tag{8.59}$$

if d is odd, and

$$U_\Phi\,|\varphi_j\rangle = \sum_{i\in[N]} |\varphi_i\rangle\,[P^\triangleright_{\mathrm{Re}}(A)]_{ij} + |\perp_j\rangle\,. \tag{8.60}$$

if d is even. Here $\Pi'\,|\perp'_j\rangle = 0$, $\Pi\,|\perp_j\rangle = 0$, and U_Φ uses $\widetilde{U}_A, \widetilde{U}_A^\dagger$, $\mathrm{C}_\Pi\,\mathrm{NOT}, \mathrm{C}_{\Pi'}\,\mathrm{NOT}$, and single qubit rotation gates for $\mathcal{O}(d)$ times.

8.6. Application: Grover's search revisited, and fixed-point amplitude amplification*

As an application, let us revisit the Grover search problem from the perspective of QSVT. Again let $|x_0\rangle$ be the desired marked state, and $|\psi_0\rangle$ be the uniform superposition of states. Now let us perform a basis transformation. We define an orthonormal basis set $\mathcal{B} = \{|\psi_0\rangle, |v_1\rangle, \ldots, |v_{N-1}\rangle\}$, where all states $|v_i\rangle$ are orthogonal to $|\psi_0\rangle$. Similarly define an orthonormal basis set $\mathcal{B}' = \{|x_0\rangle, |w_1\rangle, \ldots, |w_{N-1}\rangle\}$, where all states $|w_i\rangle$ are orthogonal to $|x_0\rangle$. Then the matrix of reflection operator R_{ψ_0} with respect to $\mathcal{B},\mathcal{B}'$ is (let $a = 1/\sqrt{N}$)

$$[R_{\psi_0}]^{\mathcal{B}'}_{\mathcal{B}} = \begin{pmatrix} a & * \\ * & * \end{pmatrix}, \tag{8.61}$$

Let $A = a$ be a 1×1 matrix, and then $R_{\psi_0} \in \mathrm{BE}_{1,n}(A)$. Furthermore, the projectors $\Pi = |\psi_0\rangle\langle\psi_0|$ and $\Pi' = |x_0\rangle\langle x_0|$ can be accessed via the reflection operator R_{ψ_0}, R_{x_0}, respectively. According to Eq. (8.58), this defines $\mathrm{C}_\Pi\,\mathrm{NOT}, \mathrm{C}_{\Pi'}\,\mathrm{NOT}$. Let $\widetilde{U}_A = R_{\psi_0}$, we have

$$\Pi'\widetilde{U}_A\Pi = a\,|x_0\rangle\langle\psi_0|\,, \tag{8.62}$$

and we would like to use Theorem 8.6 to find U_Φ that block encodes

$$|x_0\rangle\,P^\diamond_{\mathrm{Re}}(a)\,\langle\psi_0| \approx |x_0\rangle\langle\psi_0|\,. \tag{8.63}$$

To this end, we need to find an *odd*, real polynomial $P_{\mathrm{Re}}(x)$ satisfying $P_{\mathrm{Re}}(a) \approx 1$. More specifically, we need to find $P_{\mathrm{Re}}(x)$ satisfying

$$|P_{\mathrm{Re}}(x) - 1| \leq \epsilon^2, \quad \forall x \in [a, 1]. \tag{8.64}$$

We can achieve this by approximating the sign function, with $\deg(P_{\mathrm{Re}}) = \mathcal{O}(\log(1/\epsilon^2)a^{-1}) = \mathcal{O}(\log(1/\epsilon)\sqrt{N})$ (see e.g. [LC17a, Corollary 6]). This construction is also based on an approximation to the erf function. Fig. 8.4 gives a concrete construction of an odd polynomial obtained via numerical optimization, and the phase factors are obtained via QSPPACK.

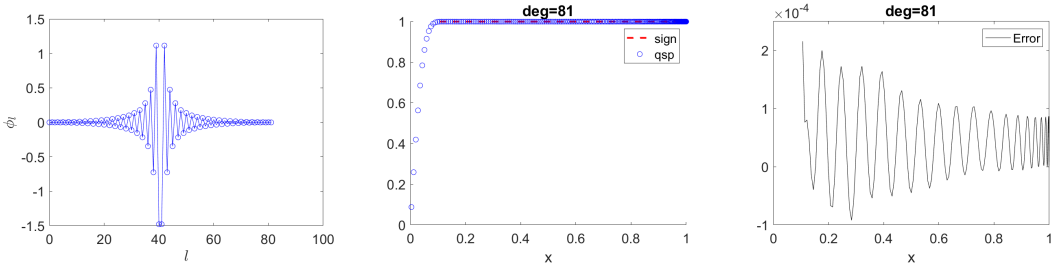


FIGURE 8.4. QSP representation of an odd polynomial approximating the sign function on $[0.1, 1]$. The phase factors plotted removes a factor of $\pi/4$ on both ends (see Eq. (7.129)).

Then

$$U_\Phi |\psi_0\rangle \approx |x_0\rangle . \tag{8.65}$$

More specifically, note that

$$U_\Phi |\psi_0\rangle - |x_0\rangle = (P_{\mathrm{Re}}(a) - 1) |\psi_0\rangle + |\widetilde{\perp}\rangle \tag{8.66}$$

for some unnormalized state $|\widetilde{\perp}\rangle$. Moreover

$$\left\| |\widetilde{\perp}\rangle \right\|^2 + P_{\mathrm{Re}}^2(a) = 1, \tag{8.67}$$

which gives

$$\left\| |\widetilde{\perp}\rangle \right\| = \sqrt{1 - P_{\mathrm{Re}}^2(a)} \leq \sqrt{1 - (1 - \epsilon^2)^2} \leq \sqrt{2\epsilon^2}. \tag{8.68}$$

So

$$\| U_\Phi |\psi_0\rangle - |x_0\rangle \| \leq \epsilon^2 + \sqrt{2}\epsilon = \mathcal{O}(\epsilon). \tag{8.69}$$

Therefore we can measure the system register to find x_0, and we achieve the same Grover type speedup.

Note that the approximation can be arbitrarily accurate, and there is no overshooting problem (though this is a small problem) as in the standard Grover search. While Grover's search does not require the output quantum state to be exactly $|x_0\rangle$, this could be desirable when it is used as a quantum subroutine, such as amplitude amplification.

The immediate generalization of the procedure above is called the fixed point amplitude amplification.

Proposition 8.7 (Fixed-point amplitude amplification)**.** *Let $\widetilde{U}_A$ be an n-qubit unitary and Π' be an n-qubit orthogonal projector such that*

$$\Pi' \widetilde{U}_A |\varphi_0\rangle = a |\psi\rangle, \quad a > \delta > 0. \tag{8.70}$$

Then there is a $(n+1)$-qubit unitary circuit U_Φ such that

$$\| |0\rangle |\psi\rangle - U_\Phi |0\rangle |\varphi_0\rangle \| \leq \epsilon. \tag{8.71}$$

here U_Φ uses the gates $\widetilde{U}_A, \widetilde{U}_A^\dagger, \mathrm{C}_{\Pi'}\,\mathrm{NOT}, \mathrm{C}_{|\varphi_0\rangle\langle\varphi_0|}\,\mathrm{NOT}$ and single qubit rotation gates for $\mathcal{O}(\log(1/\epsilon)\delta^{-1})$ times.

Proof. The procedure is very similar to Grover's search. We only prove the case when Π is of rank 1, though the statement is also correct when the rank of Π is larger than 1. We can construct $\mathcal{B} = \{|\varphi_0\rangle, |v_1\rangle, \ldots, |v_{N-1}\rangle\}$, where all states $|v_i\rangle$ are orthogonal to $|\varphi_0\rangle$. Similarly define an orthonormal basis set $\mathcal{B}' = \{|\psi\rangle, |w_1\rangle, \ldots, |w_{N-1}\rangle\}$, where all states $|w_i\rangle$ are orthogonal to the target state $|\psi\rangle$. Since the target state $|\psi\rangle$ belongs to the range of Π',

$$\langle\psi|\widetilde{U}_A|\varphi_0\rangle = \langle\psi|\Pi'\widetilde{U}_A|\varphi_0\rangle = a, \tag{8.72}$$

i.e.,

$$[\widetilde{U}_A]_{\mathcal{B}}^{\mathcal{B}'} = \begin{pmatrix} a & * \\ * & * \end{pmatrix}. \tag{8.73}$$

Now let $\Pi = |\varphi_0\rangle\langle\varphi_0|$, we can use the same choice of $P_{\mathrm{Re}}(x)$ as in Grover's search so that $|P_{\mathrm{Re}}(x) - 1| = \mathcal{O}(\epsilon^2)$ for any $x \geq \delta$, and $\deg(P_{\mathrm{Re}}) = \mathcal{O}(\log(1/\epsilon)\delta^{-1})$. The corresponding U_Φ uses one ancilla qubit to block encode

$$|\psi\rangle P_{\mathrm{Re}}(\delta) \langle\varphi_0| \approx |\psi\rangle\langle\varphi_0|. \tag{8.74}$$

□

Note that the ranks of Π', Π are different. This does not affect the proof.

Exercise 8.1 (Robust oblivious amplitude amplification)**.** Consider a quantum circuit consisting of two registers denoted by a and s. Suppose we have a block encoding V of A: $A = (\langle 0|_a \otimes I_s) V (|0\rangle_a \otimes I_s)$. Let $W = -V(\mathrm{REF}\otimes I_s)V^\dagger(\mathrm{REF}\otimes I_s)V$, where $\mathrm{REF} = I_a - 2|0\rangle_a\langle 0|_a$. (1) Within the framework of QSVT, what is the polynomial associated with the singular value transformation implemented by W? (2) Suppose $A = U/2$ for some unitary U. What is $(\langle 0|_a \otimes I_s) W (|0\rangle_a \otimes I_s)$? (3) Explain the construction of W in terms of a singular value transformation $f^\circ(A)$ with $f(x) = 3x - 4x^3$. Draw the picture of $f(x)$ and mark its values at $x = 0, \frac{1}{2}, 1$.

Exercise 8.2 (Logarithm of unitaries)**.** Given access to a unitary $U = e^{\mathrm{i}H}$ where $\|H\| \leq \pi/2$. Use QSVT to design a quantum algorithm to approximately implement a block encoding of H, using controlled U and its inverses, as well as elementary quantum gates.

Bibliography

[Aar13] Scott Aaronson. *Quantum computing since Democritus*. Cambridge Univ. Pr., 2013.

[ABF16] F. Arrigo, M. Benzi, and C. Fenu. Computation of generalized matrix functions. *SIAM J. Matrix Anal. Appl.*, 37:836–860, 2016.

[BACS07] Dominic W Berry, Graeme Ahokas, Richard Cleve, and Barry C Sanders. Efficient quantum algorithms for simulating sparse Hamiltonians. *Commun. Math. Phys.*, 270(2):359–371, 2007.

[Chi21] Andrew Childs. Lecture notes on quantum algorithms, 2021.

[CKS17] Andrew M. Childs, Robin Kothari, and Rolando D. Somma. Quantum algorithm for systems of linear equations with exponentially improved dependence on precision. *SIAM J. Comput.*, 46:1920–1950, 2017.

[CST+21] Andrew M Childs, Yuan Su, Minh C Tran, Nathan Wiebe, and Shuchen Zhu. Theory of trotter error with commutator scaling. *Phys. Rev. X*, 11(1):011020, 2021.

[DHM+18] Danial Dervovic, Mark Herbster, Peter Mountney, Simone Severini, Naïri Usher, and Leonard Wossnig. Quantum linear systems algorithms: a primer. *arXiv:1802.08227*, 2018.

[DL21] Yulong Dong and Lin Lin. Random circuit block-encoded matrix and a proposal of quantum linpack benchmark. *Phys. Rev. A*, 103(6):062412, 2021.

[DMWL21] Yulong Dong, Xiang Meng, K Birgitta Whaley, and Lin Lin. Efficient phase factor evaluation in quantum signal processing. *Phys. Rev. A*, 103:042419, 2021.

[DW19] Ronald De Wolf. Quantum computing: Lecture notes. *arXiv:1907.09415*, 2019.

[Fey82] Richard P Feynman. Simulating physics with computers. *Int. J. Theor. Phys*, 21(6/7), 1982.

[GSLW18] András Gilyén, Yuan Su, Guang Hao Low, and Nathan Wiebe. Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics. *arXiv:1806.01838*, 2018.

[GSLW19] András Gilyén, Yuan Su, Guang Hao Low, and Nathan Wiebe. 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*, pages 193–204, 2019.

[GVL13] G. H. Golub and C. F. Van Loan. *Matrix computations*. Johns Hopkins Univ. Press, Baltimore, fourth edition, 2013.

[Haa19] J. Haah. Product decomposition of periodic functions in quantum signal processing. *Quantum*, 3:190, 2019.

[HBI73] J. B. Hawkins and A. Ben-Israel. On generalized matrix functions. *Linear and Multilinear Algebra*, 1:163–171, 1973.

[HHL09] Aram W Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.*, 103:150502, 2009.

[JL00] Tobias Jahnke and Christian Lubich. Error bounds for exponential operator splittings. *BIT*, 40(4):735–744, 2000.

[KSV02] Alexei Yu Kitaev, Alexander Shen, and Mikhail N Vyalyi. *Classical and quantum computation*. Number 47. American Mathematical Soc., 2002.

[LC17a] Guang Hao Low and Isaac L Chuang. Hamiltonian simulation by uniform spectral amplification. *arXiv:1707.05391*, 2017.

[LC17b] Guang Hao Low and Isaac L. Chuang. Optimal hamiltonian simulation by quantum signal processing. *Phys. Rev. Lett.*, 118:010501, 2017.

[Llo96] Seth Lloyd. Universal quantum simulators. *Science*, pages 1073–1078, 1996.

[LT20a] Lin Lin and Yu Tong. Near-optimal ground state preparation. *Quantum*, 4:372, 2020.

[LT20b] Lin Lin and Yu Tong. Optimal quantum eigenstate filtering with application to solving quantum linear systems. *Quantum*, 4:361, 2020.

[Nak08] Mikio Nakahara. *Quantum computing: from linear algebra to physical realizations*. CRC press, 2008.

[NC00] Michael A Nielsen and Isaac Chuang. Quantum computation and quantum information, 2000.

[NO88] J. W. Negele and H. Orland. *Quantum many-particle systems*. Westview, 1988.

[NWZ09] Daniel Nagaj, Pawel Wocjan, and Yong Zhang. Fast amplification of QMA. *Quantum Inf. Comput.*, 9(11):1053–1068, 2009.

[Pre99] John Preskill. Lecture notes for physics 219: Quantum computation. *Caltech Lecture Notes*, 1999.

[RP11] Eleanor G Rieffel and Wolfgang H Polak. *Quantum computing: A gentle introduction*. MIT Press, 2011.

[Sze04] Mario Szegedy. Quantum speed-up of markov chain based algorithms. In *45th Annual IEEE symposium on foundations of computer science*, pages 32–41, 2004.

[Tha08] Mechthild Thalhammer. High-order exponential operator splitting methods for time-dependent Schrödinger equations. *SIAM J. Numer. Anal.*, 46(4):2022–2038, 2008.