Tutorial

Grand Unification of Quantum Algorithms

John M. Martyn[1,2,*] Zane M. Rossi,[2] Andrew K. Tan,[3] and Isaac L. Chuang[3,4]

[1]*Center for Theoretical Physics, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139, USA*
[2]*Department of Physics, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139, USA*
[3]*Department of Physics, Co-Design Center for Quantum Advantage, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139, USA*
[4]*Center for Ultracold Atoms, and Research Laboratory of Electronics, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139, USA*

Quantum algorithms offer significant speed-ups over their classical counterparts for a variety of problems. The strongest arguments for this advantage are borne by algorithms for quantum search, quantum phase estimation, and Hamiltonian simulation, which appear as subroutines for large families of composite quantum algorithms. A number of these quantum algorithms have recently been tied together by a novel technique known as the quantum singular value transformation (QSVT), which enables one to perform a polynomial transformation of the singular values of a linear operator embedded in a unitary matrix. In the seminal GSLW'19 paper on the QSVT [Gilyén *et al.*, ACM STOC 2019], many algorithms are encompassed, including amplitude amplification, methods for the quantum linear systems problem, and quantum simulation. Here, we provide a pedagogical tutorial through these developments, first illustrating how quantum signal processing may be generalized to the quantum eigenvalue transform, from which the QSVT naturally emerges. Paralleling GSLW'19, we then employ the QSVT to construct intuitive quantum algorithms for search, phase estimation, and Hamiltonian simulation, and also showcase algorithms for the eigenvalue threshold problem and matrix inversion. This overview illustrates how the QSVT is a single framework comprising the three major quantum algorithms, suggesting a *grand unification* of quantum algorithms.

CONTENTS

---

*jmmartyn@mit.edu

I. INTRODUCTION

Algorithms solve problems by presenting a process or set of rules to be followed, utilizing a basic set of building blocks provided. Computer science traditionally employs Boolean circuit components as the basic blocks, from which standard arithmetic operations may be composed, as Boolean functions. Quantum computation employs a different set of basic blocks, typically unitary operations on one or more two-state systems (qubits), to realize quantum circuits. A fundamental challenge arises, however, when seeking to unite the world of quantum circuits with that of Boolean functions: in general, Boolean functions need not be reversible, whereas quantum circuits are manifestly unitary transforms and must thus be invertible.

Early in the history of quantum computation, this barrier was transcended by seminal work [1] showing that all Boolean functions can be made reversible, with only a small overhead in space and time. Toffoli and Fredkin famously illustrated this idea by showing how the ideal Newtonian dynamics of finite-radii spheres (billiard balls) can be used to simulate reversible Boolean circuits, via their collisions [2]. Following this concept, simulation of arbitrary Boolean functions can also be accomplished using quantum circuits, by first embedding the desired function into a reversible Boolean circuit and then constructing a quantum circuit realizing this invertible transform. Such an embedding is a core part of Shor's quantum factoring algorithm [3], for example, as used in the modular exponentiation of an input number.

Intriguingly, however, the two other major "primordial" quantum algorithms, Grover's quantum search algorithm [4] and the Hamiltonian simulation algorithm [5,6], do not employ an embedding of a reversible Boolean function. In fact, a key part of the quantum factoring algorithm is its use of the quantum Fourier transform, which has no direct classical analog, in the sense that it is not at all like a quantum embedding of a reversible Boolean function for the Fourier transform. And yet, all three of these algorithms provide solutions to problems with clear classical counterparts, and attain known speed-ups over the comparable classical "Boolean function" approaches. So wherein lies the ability of quantum algorithms to address and speed up the solution to a classically specified problem?

As illustrated in this tutorial, a key idea in uniting quantum and classical computation is not to first make classical computation reversible. Instead, observe that the dynamical behavior of a *subsystem* of a quantum system can be *nonunitary* and thus can directly realize irreversible nonlinear functions. An extreme case of this is projective measurement: the billiard-ball model can realize noninvertible gates simply by discarding balls but this would be inefficient. More constructively, the recently developed framework of *quantum signal processing* (QSP) [7,8] provides a systematic method to make a quantum subsystem transform under nearly arbitrary polynomial functions of degree d, using $\mathcal{O}(d)$ elementary unitary quantum operations. Crucially, the polynomial describes not the output of the full quantum system but only a very specific and clearly identified subsystem. And, remarkably, the essential ideas behind QSP originate from the early days of practical control of two-level quantum systems, with nuclear magnetic resonance [9–11].

With this framework, we present in this tutorial a pedagogical overview of the modern approach to quantum search, factoring, and simulation, focusing on how all three of these central quantum algorithms may be unified as instances of the recently developed *quantum singular value transform (QSVT) algorithm* [12]. The QSVT algorithm generalizes QSP and efficiently applies a polynomial transformation to the singular values of a linear operator (governing a particular subsystem) embedded in a larger unitary operator. And more recently, such a singular value transformation has been generalized to apply to an operator embedded in a block of a Hamiltonian [13].

Singular values naturally arise in this context from the fact that the input and output spaces of the embedded linear operator may be of different sizes. The polynomial transformation of the singular values is achieved by applying a specific sequence of SU(2) rotations to the embedded subspace, where each rotation is parametrized by an angle $\phi_k \in \mathbb{R}$. The QSVT algorithm is parametric in that the polynomial transformation is completely characterized by the choice of phase angles $\{\phi_k\}$. Moreover, given the desired polynomial transformation, the QSP phase angles that generate it may be classically efficiently and stably computed [7,12,14].

This seemingly simple parametrization endows the QSVT with immense flexibility and power. Using the QSVT as a subroutine, we present quantum algorithms for the search problem and phase estimation and give

simplified arguments for optimal Hamiltonian simulation by the QSVT and matrix inversion by QSVT. We also present a QSVT-based algorithm for *the eigenvalue threshold problem*, wherein one wishes to know if a (normal) matrix has an eigenvalue above some threshold, and use this to simplify the presentation of the use of the QSVT for phase estimation. Each algorithm is realized simply as an instance of the QSVT with a natural oracle, adaptively repeated and interspersed in a common pattern with simple quantum gates and measurements.

This common pattern is particularly fascinating. At face value, the algorithms for the search problem, factoring (aka phase estimation), and Hamiltonian simulation appear to share no similar structures, owing their quantum speed-ups to different sources, and yet they can all be derived from a single algorithmic primitive and interpolated between by a simple change of parameters. In addition, these three central algorithms form the foundation for most quantum algorithms currently known. For instance, the Harrow-Hassidim-Lloyd (HHL) algorithm for linear systems [15] incorporates Hamiltonian simulation and phase estimation in order to invert a Hermitian matrix and, similarly, the quantum counting algorithm integrates quantum search with phase estimation to count the number of marked elements in an unstructured set [16]. As shown in this tutorial, by simply adjusting the parameters of the QSVT, one can construct nearly all known quantum algorithms. It is in this sense that the QSVT provides a *grand unification* of quantum algorithms.

While some of these applications have been covered in recent works on the QSVT [12,17], here we aim to present these constructions as pedagogical as possible, providing detailed procedures and intuition for each and including explicit examples to support abstract ideas where appropriate. We also supply performance bounds and resource requirements for each of the algorithms presented here, which we anticipate will be helpful. It is our hope that this presentation makes the QSVT and QSP more accessible, catalyzing future developments in quantum algorithms.

Throughout this tutorial, we assume familiarity with basic concepts in quantum computing, such as unitary dynamics and measurement, as well as the conventional quantum algorithms for search, phase estimation, and Hamiltonian simulation. For a comprehensive review of these subjects, see Refs. [18–20]. Further, we aim to present this work in a manner accessible to readers without prior knowledge of QSP and the QSVT, but if more information on these topics is needed, Refs. [7,12,21,22] may serve as helpful references.

A. Road map

We share the story of this grand unification of quantum algorithms by first surveying the development of the QSVT in Sec. II, beginning with quantum signal

processing and then demonstrating how this technology leads to quantum eigenvalue transforms and, ultimately, the quantum singular value transformation. Thereafter, we detail a statement of grand unification for currently known quantum algorithms, presenting QSVT-based algorithms for the search problem and the eigenvalue threshold problem in Secs. III and IV, respectively. Further fulfilling the unification promise, we introduce an algorithm for phase estimation by QSVT in Sec. V and show how the quantum Fourier transform emerges from cascading QSVT sequences. We then highlight in Sec. VI how the QSVT can yield intuitive algorithms for Hamiltonian simulation and matrix inversion. Lastly, in Sec. VII, we explore the implications of these results and discuss areas of future research in the utility of the QSVT.

II. FROM QSP TO THE QSVT

In this section, we develop the framework of the QSVT, starting with quantum signal processing and then providing a concrete application of this technology to amplitude amplification. Building off this example, we establish quantum eigenvalue transforms and finally quantum singular value transforms. Overall, this section is meant to be pedagogical and easily accessible to readers with a background in basic quantum circuits and linear algebra, as the constructions of this section underlie the remainder of the work. For more details on QSP and the QSVT, including alternative introductions to the topics, rigorous proofs, and applications not covered here, see Refs. [7,12,21,22].

A. Quantum signal processing

QSP generalizes the results of composite-pulse sequences [7,8,23] and is built on the idea of interleaving two kinds of single-qubit rotations: a *signal* rotation operator W and a *signal processing* rotation operator S. These rotation operations are about different axes through the Bloch sphere, e.g., commonly W is an x rotation, while S is a z rotation. Moreover, the signal rotation always rotates through the same angle θ, whereas the signal-processing rotation rotates through a variable angle according to some predetermined sequence.

For example, let the signal-rotation operator be

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

which is an x rotation by angle $\theta = -2\cos^{-1} a$. And let the signal-processing rotation operator be

$$S(\phi) = e^{i\phi Z}, \quad (2)$$

which is a z rotation by angle -2ϕ. For a tuple of phases $\vec{\phi} = (\phi_0, \phi_1, \ldots \phi_d) \in \mathbb{R}^{d+1}$ and using these conventions
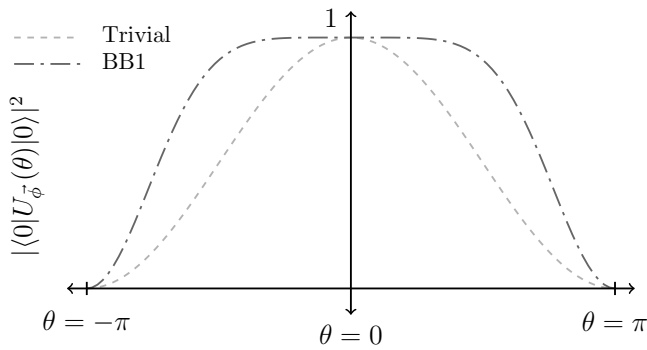
FIG. 1. The transition probabilities for $|0\rangle \rightarrow |0\rangle$ after the application of $\exp(i\theta X/2)$ (uniform dashed) and the QSP sequence for the BB1 protocol (dotted dashed). The latter transition probability is seen to be near one for a wider range of θ.

[24], the QSP operation sequence $U_{\vec{\phi}}$ is defined as

$$U_{\vec{\phi}} = e^{i\phi_0 Z} \prod_{k=1}^{d} W(a)e^{i\phi_k Z}. \qquad (3)$$

What is interesting is how QSP can modify the incoming signal. Suppose that $\vec{\phi} = (0,0)$, i.e., there is no processing, such that $U_{\vec{\phi}} = W(a)$ is just the unchanged signal. If we plot the probability of a $|0\rangle$ qubit input staying unchanged under this operation, i.e., $p = |\langle 0|U_{\vec{\phi}}|0\rangle|^2$, as a function of θ, we find a nice cosinusoid (the lower plot in Fig. 1), because for this case $p = \cos^2(\theta/2)$. Now, if we do some signal processing, by letting $\vec{\phi} = (\pi/2, -\eta, 2\eta, 0, -2\eta, \eta)$, where $\eta = \frac{1}{2}\cos^{-1}(-1/4)$, then for the new $U_{\vec{\phi}}$ using these phases, we find the new probability $p = |\langle 0|U_{\vec{\phi}}|0\rangle|^2 = 1/8\cos^2(\theta/2)\left[3\cos^8(\theta/2) - 15\cos^6(\theta/2) + 35\cos^4(\theta/2) - 45\cos^2(\theta/2) + 30\right]$ (the upper dotted-dashed line in Fig. 1), which for small θ is approximately $1 - 5/8(\theta/2)^6$. This has the nice property that the qubit remains unflipped for a wide range of signal angles but then a sharp transition happens around $\theta \approx 2\pi/3$. This increases sensitivity to specific values of the signal.

Thus, sequences like this are widely employed in magnetic resonance imaging to increase image contrast. This particular sequence is famous in the field of NMR and is known as the "BB1" pulse sequence [11]. Many such "composite-pulse" sequences are known [9,25,26], with a wide range of variations, and in experimental quantum computation they serve to suppress specific kinds of errors and enhance sensitivity to specific kinds of signals, across implementations ranging from quantum dots [27] and nitrogen-vacancy centers in diamonds [28], to trapped ions [29–31] and superconducting qubits [32].

That QSP sequences can have such signal transformation properties is well known, because in general the matrix element $P(a) = \langle 0|U_{\vec{\phi}}|0\rangle$ becomes a polynomial in a, with the order of the polynomial being at most the length of the sequence of QSP phases $\vec{\phi}$. Specifically, for example, for $\vec{\phi} = (0,0)$, $P(a) = a$; for $\vec{\phi} = (0,0,0)$, $P(a) = 2a^2 - 1$; and for $\vec{\phi} = (0,0,0,0)$, $P(a) = 4a^3 - 3a$. These are the Chebyshev polynomials of the first kind, $T_d(a)$.

Perhaps the most remarkable property of the QSP sequence of Eq. (3), however, is the reverse of this statement: it turns out that for a given polynomial $P(a)$ (subject to some reasonable constraints), there exists a set of QSP phase angles $\vec{\phi}$ such that $P(a) = \langle 0|U_{\vec{\phi}}|0\rangle$. Specifically [7]:

Theorem 1 (Quantum Signal Processing): *The QSP sequence $U_{\vec{\phi}}$ produces a matrix that may be expressed as polynomial function of a:*

$$e^{i\phi_0 Z} \prod_{k=1}^{d} W(a)e^{i\phi_k Z}$$
$$= \begin{bmatrix} P(a) & iQ(a)\sqrt{1-a^2} \\ iQ^*(a)\sqrt{1-a^2} & P^*(a) \end{bmatrix}, \qquad (4)$$

for $a \in [-1,1]$, and a $\vec{\phi}$ exists for any polynomials P, Q in a such that:

(i) $deg(P) \le d$, $deg(Q) \le d-1$
(ii) P has parity $d \bmod 2$ and Q has parity $(d-1) \bmod 2$
(iii) $|P|^2 + (1-a^2)|Q|^2 = 1$

The forward part of this theorem is easily proven by induction, starting from the $d = 0$ case, for which $P = e^{i\phi_0}$ and $Q = 0$. The reverse direction of this theorem is more involved and can be proven in a number of ways, including an elegant interpretation involving Laurent polynomial algebras [33,34].

Often, however, we are interested not in the unitaries that can be constructed with QSP but, rather, the achievable polynomial transformations of the input, Poly(a), in a subsystem. If, as above, we choose Poly(a) $= \langle 0|U_{\vec{\phi}}|0\rangle = P(a)$, we are limited to P for which there exists a polynomial Q satisfying the conditions of Theorem 1. This can be quite a limiting condition for some applications. For example, for $a = \pm 1$, $W(\pm 1)$ is proportional to the identity matrix and the entire sequence QSP sequence collapses to a single z rotation, limiting us to polynomials Poly(a) such that $|\text{Poly}(\pm 1)| = 1$. This limitation can be overcome by instead defining Poly(a) $= \langle +|U_{\vec{\phi}}|+\rangle = \text{Re}[P(a)] + i\text{Re}[Q(a)]\sqrt{1-a^2}$. In this case, it can be shown that we can accurately approximate any real polynomial with parity $d \bmod 2$ such that $\deg(\text{Poly}) \le d$ and $|\text{Poly}(a)| \le 1 \ \forall \ a \in [-1,1]$. This can be achieved by selecting an

appropriate P the real part of which approximates the desired function and a Q with a small real component. With this convention, the set of achievable polynomials is sufficiently expressive for all of the applications described in this tutorial. In general, the basis employed for the desired polynomial may be called the *signal basis* and, unless otherwise specified, below we take this basis to be $|+\rangle, |-\rangle$. Appendix A elaborates on this and discusses the different conventional statements of the QSP theorem 1.

While Theorem 1 guarantees the existence of such a $\vec{\phi}$, it does not provide a method for determining it. Fortunately, Remez-type exchange algorithms can efficiently compute a $\vec{\phi}$ that produces a good approximation to any feasible polynomials P and Q [7]. Further, more efficient and numerically stable algorithms have also been found [33–35] and novel optimization techniques are currently under development [36]. Appendix D gives explicit examples of $\vec{\phi}$ for a wide family of polynomials used in realizing quantum algorithms and illustrates an open-source code package accompanying this tutorial for generating QSP phase angle coefficients.

B. An application to amplitude amplification and search

Toward motivating the QSVT from QSP, we introduce an illustrative example of how multiqubit problems may be simplified by identifying qubitlike subsystems, to which the ideas of QSP may be applied. The concept demonstrated in this subsection is essentially that of *qubitization* [8], which forms a core tenet of the QSVT.

Specifically, we discuss the problem of amplitude amplification; a similar construction is discussed in Sec. III employing the major theorems of the QSVT but here we begin from basic principles. It is hoped that the geometric intuition behind the argument that follows and the expediency of the fully developed construction in Sec. III will complement each other.

The utility of Theorem 1 is nicely demonstrated in solving the following problem. Suppose that you are given a unitary U (which may act on some Hilbert space of large dimension; i.e., larger than just a qubit) and its inverse, $U^\dagger$, as well as two operators A_ϕ and B_ϕ, each of which rotates the phases of a specific privileged state, namely:

$$A_\phi = e^{i\phi|A_0\rangle\langle A_0|} \tag{5}$$

$$B_\phi = e^{i\phi|B_0\rangle\langle B_0|}. \tag{6}$$

The challenge is to construct a circuit Q using U, $U^\dagger$, A_ϕ, and B_ϕ such that

$$|\langle A_0|Q|B_0\rangle| \longrightarrow 1 \tag{7}$$

in the limit of a sufficiently large circuit and assuming that the original matrix element $\langle A_0|U|B_0\rangle$ of U is nonzero.

This problem is known as *amplitude amplification* and, remarkably, it can be solved without knowing the specific initial value $\langle A_0|U|B_0\rangle$, by using the *oblivious fixed-point amplitude amplification* quantum algorithm. We show that such an algorithm arises only from Theorem 1, even in the multiple-qubit setting, by recognizing that there are two concentric Bloch spheres (qubitlike spaces) in this problem.

Specifically, one can recognize that $U|B_0\rangle$ is a quantum state that has a nonzero component along $|A_0\rangle$ and another component perpendicular to $|A_0\rangle$. So we define

$$|A_\perp\rangle = \frac{1}{\mathcal{N}}\Big(I - |A_0\rangle\langle A_0|\Big)U|B_0\rangle, \tag{8}$$

where $\mathcal{N}$ is the normalization factor needed to make $|A_\perp\rangle$ a unit vector. Then

$$U|B_0\rangle = a|A_0\rangle + \sqrt{1 - a^2}|A_\perp\rangle \tag{9}$$

for $a = \langle A_0|U|B_0\rangle$ (we may assume that a is real because a possible phase may be absorbed into $|B_0\rangle$). Similarly, we may define some $|B_\perp\rangle$ such that

$$U|B_\perp\rangle = -a|A_\perp\rangle + \sqrt{1 - a^2}|A_0\rangle. \tag{10}$$

These ideas are illustrated in the diagram of Fig. 2, using the familiar Bloch sphere.
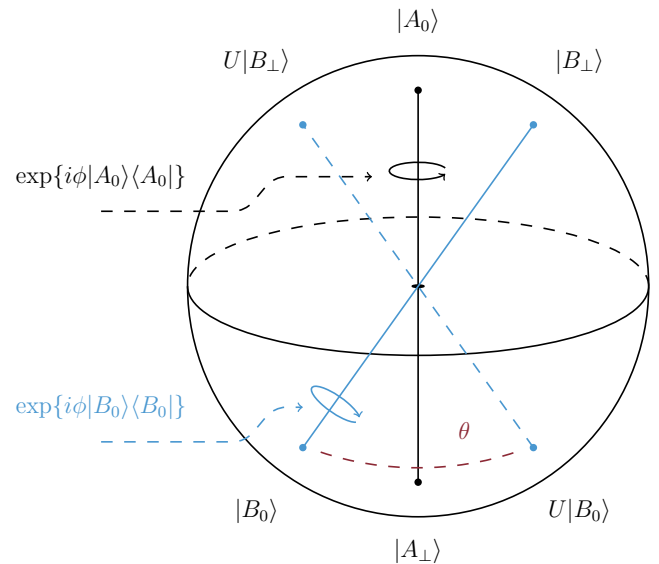
FIG. 2. An illustration of amplitude amplification, where one desires to prepare the state $|A_0\rangle$ (here, the north pole of some Bloch sphere) obliviously to this state, given only $|B_0\rangle$, an operator U the $\langle A_0|U|B_0\rangle = a$ component of which is nonzero, and the ability to rotate about the states $|B_0\rangle$ (blue) and $|A_0\rangle$ (black) through chosen angles. The standard Grover iterate can be recovered in this model if one desires, by producing a simple rotation by $\theta = 2\cos^{-1}\left(\sqrt{1 - a^2}\right) = 2\sin^{-1} a$ (red) toward $|A_0\rangle$.

Thus, on the two-dimensional Hilbert space spanned by $\{|A_0\rangle, |A_\perp\rangle\}$, the action of U is that of a 2×2 unitary:

$$
\begin{aligned}
U = &a\,(|A_0\rangle\langle B_0| - |A_\perp\rangle\langle B_\perp|) \\
&+ \sqrt{1-a^2}\,(|A_\perp\rangle\langle B_0| + |A_0\rangle\langle B_\perp|),
\end{aligned}
\tag{11}
$$

which is convenient to represent in matrix form, as

$$
U = \begin{array}{c} \\ |A_0\rangle \\ |A_\perp\rangle \end{array} \overset{\displaystyle |B_0\rangle \qquad\quad |B_\perp\rangle}{\left[\begin{array}{cc} a & \sqrt{1-a^2} \\ \sqrt{1-a^2} & -a \end{array}\right]},
\tag{12}
$$

where the labels on the matrix indicate that columns act on $|B_0\rangle, |B_\perp\rangle$ (from left to right) and the rows act on $|A_0\rangle, |A_\perp\rangle$ (from top to bottom), such that U brings a state in the B basis into a state in the A basis. These two bases are the two Bloch spheres (or qubit bases) encoded in the problem. Moreover, U is a reflection operation, which we may represent in this qubitlike abstraction as $R(a) = X R_y(\theta)$, with $\theta = 2\cos^{-1}\left(\sqrt{1-a^2}\right) = 2\sin^{-1} a$.

This two-Bloch sphere picture provides the intuitive basis for the following theorem:

Theorem 2 (Amplitude Amplification)**:** *Given unitary U, its inverse $U^\dagger$, and operators $A_\phi = e^{i\phi|A_0\rangle\langle A_0|}$, $B_\phi = e^{i\phi|B_0\rangle\langle B_0|}$,*

$$
\langle A_0| \left[\prod_{k=1}^{d/2} U B_{\phi_{2k-1}} U^\dagger A_{\phi_{2k}}\right] U|B_0\rangle = Poly(a),
\tag{13}
$$

where Poly(a) is a polynomial in $a = \langle A_0|U|B_0\rangle$ of degree at most d and obeys the conditions on P from Theorem 1.

Why does this work? Note first that $U|B_0\rangle$ lives in the A-qubit space, spanned by $|A_0\rangle$ and $|A_\perp\rangle$. This qubit then gets rotated around its "z" axis by A_ϕ. The $U^\dagger$ then rotates this around the y axis (and also does a reflection around Z but we can disregard that for this intuition). $U^\dagger$ also maps the state from the A-qubit space back into the B-qubit space, spanned by $|B_0\rangle$ and $|B_\perp\rangle$. Next, B_ϕ rotates the state around the z axis of the B-qubit space. Then the leftmost U does another y rotation (and reflection) and maps the state back into the A-qubit space. The sequence in the square brackets maps the state back and forth between the two qubit bases, sandwiching y-axis rotations with z-axis rotations. This sandwich of rotations is thus just doing quantum signal processing as in Sec. II A; we understand this behavior well.

The formal proof of this theorem begins by noting that on the subspaces defined by the two Bloch spheres defined above, $U = U^\dagger$. Moreover, note that $R(a)$, the 2×2 unitary representation of U (in the qubit bases relevant to our

problem), is related to the $W(a)$ of Eq. (1) by

$$
R(a) = -i e^{i\frac{\pi}{4}Z} W(a) e^{i\frac{\pi}{4}Z}.
\tag{14}
$$

Substituting this into Eq. (13) and recognizing that A_ϕ and B_ϕ simply become z-axis rotations, we obtain

$$
\langle A_0| \left(e^{i\phi_0' Z}\left[\prod_{k=1}^{d} W(a) e^{i\phi_k' Z}\right]\right) U|B_0\rangle = \mathrm{Poly}(a),
\tag{15}
$$

where $\{\phi_k'\}$ are linear combinations of the original phases $\{\phi_k\}$. By Theorem 1, the term in parentheses is a matrix of related polynomials evaluated at a, completing the proof of Theorem 2.

Theorem 2 takes on the meaning of performing "oblivious amplitude amplification" when the phases $\{\phi_k\}$ are chosen such that the polynomial constructed approximates the sign function for small values of a. The technical aspects of generating the proper polynomials are further discussed in Sec. III and Appendix D 1.

Grover's celebrated quantum search algorithm [4], which is similar in character to the above, can easily be derived from the construction of amplitude amplification. In the search problem, some computational basis state $|A_0\rangle$ is unknown but an oracle is given that implements

$$
A_\pi = e^{i\pi|A_0\rangle\langle A_0|}
\tag{16}
$$

and the goal is to create a quantum state close to $|A_0\rangle$ using as few queries to the oracle as possible. The search algorithm solves this problem by choosing $U = H^{\otimes n}$ (Hadamard gates on all the qubits in the search space) and starting with $|B_0\rangle = |0\rangle$. Note that since $|A_0\rangle$ is a computational basis state, $\langle A_0|U|B_0\rangle = 2^{-n/2} = 1/\sqrt{N}$, because $U|B_0\rangle = |\psi_0\rangle$ is an equal superposition over all N basis states in the search space. This means that the amplitude amplification algorithm can be applied, with the specific case of $a = 1/\sqrt{N}$ being known, meaning we choose all the $\phi_k = \pi$ for all k. This choice implies that

$$
U B_\phi U^\dagger = e^{i\pi H^{\otimes n}|0\rangle\langle 0|H^{\otimes n}} = I - 2|\psi_0\rangle\langle\psi_0|,
\tag{17}
$$

which can be recognized as Grover's "inversion about mean" iterate. This choice also produces an oscillating polynomial that monotonically increases the matrix element up to $d \approx \lceil \pi/(2\sin^{-1} a)\rceil$ steps [37]. This may be seen by noting that each application of the signal-rotation operator rotates through an angle $2\sin^{-1} a$ and the target state is at most an angle π away, similar to the argument used to derive the query complexity of Grover's algorithm in Ref. [18]. The amplitude amplification algorithm thus gives Grover's quantum search algorithm in this limit and the number of oracle calls required is $d \approx \pi/(2\sin^{-1} a) = \pi/[2\sin^{-1}(1/\sqrt{N})] = O(\sqrt{N})$, the known performance of

Grover's algorithm. Once more, a similar result is discussed in Sec. III, where the flow from qubitization to the QSVT to fixed-point amplitude amplification is applied more generally.

C. Quantum eigenvalue transforms

Theorem 2 is known as amplitude amplification because it accomplishes a polynomial transform of one specific amplitude, namely the matrix element of U at $|A_0\rangle\langle B_0|$. However, this polynomial transform can actually be performed over *an entire vector space*, not just a one-dimensional matrix element. In particular, we now show that this technology can be used to polynomially transform all the *eigenvalues* of a Hamiltonian H that has been embedded into a block of a unitary matrix U.

Specifically, suppose that we have the unitary

$$U = \begin{array}{c} 0 \\ 1 \end{array}\!\!\begin{array}{c}\,^0\quad\,^1\\ \left[\begin{array}{cc} \mathcal{H} & \cdot \\ \cdot & \cdot \end{array}\right]\end{array}, \tag{18}$$

where $\mathcal{H}$ is some $N \times N$ (possibly very large) Hamiltonian operator, located in the upper-left block of U, labeled by an index qubit being in the state $|0\rangle$ (and it is said that $\mathcal{H}$ has thus been "qubitized"). We include the indices 0 and 1 adjacent to the matrix representation of U to schematically indicate how $\mathcal{H}$ is encoded in U. At the cost of some generality, let us make some assumptions for simplicity of exposition. In particular, assume that the operator norm $\|\mathcal{H}\|$ is sufficiently small that this block embedding can be achieved, i.e., $\|\mathcal{H}\| \leq 1$ (if not, then one can instead embed some rescaled version of the Hamiltonian, $\mathcal{H}/\alpha$, but for now we neglect this case for the sake of expository clarity). In particular, suppose that the eigenvectors and eigenvalues of $\mathcal{H}$ are given as

$$\mathcal{H} = \sum_\lambda \lambda |\lambda\rangle\langle\lambda|. \tag{19}$$

Then, specializing to a specific block encoding for clarity, the missing blocks of U may be completed as

$$U = \begin{bmatrix} \mathcal{H} & \sqrt{I - \mathcal{H}^2} \\ \sqrt{I - \mathcal{H}^2} & -\mathcal{H} \end{bmatrix}, \tag{20}$$

where

$$\sqrt{I - \mathcal{H}^2} = \sum_\lambda \sqrt{1 - \lambda^2}|\lambda\rangle\langle\lambda|, \tag{21}$$

and it can be seen by inspection that $U^\dagger U = I$ as required for the unitarity of U, as long as the eigenvalues λ are of reasonable scale. While a general block encoding need not take this specialized form, this choice of encoding is sufficient for our illustrative purposes. Moreover, the treatment

of general block encodings is presented in Refs. [8,12], wherein it is shown that a general block encoding takes a form similar to Eq. (20) in a special basis related to the eigenbasis of $\mathcal{H}$.

Our specialized choice of block encoding means that U may be expressed as a sum of two tensor products,

$$U = Z \otimes \mathcal{H} + X \otimes \sqrt{I - \mathcal{H}^2}, \tag{22}$$

and therefore acts as

$$U|0\rangle|\lambda\rangle = \lambda|0\rangle|\lambda\rangle + \sqrt{1 - \lambda^2}|1\rangle|\lambda\rangle \tag{23}$$

$$U|1\rangle|\lambda\rangle = -\lambda|1\rangle|\lambda\rangle + \sqrt{1 - \lambda^2}|0\rangle|\lambda\rangle, \tag{24}$$

which indicates that U contains a Bloch sphere (i.e., a qubit basis) for each eigenspace [38] of $\mathcal{H}$ corresponding to a certain eigenvalue. In particular, U may be expressed as a direct sum over N separate Bloch spheres:

$$\begin{aligned} U &= \bigoplus_\lambda \begin{bmatrix} \lambda & \sqrt{1 - \lambda^2} \\ \sqrt{1 - \lambda^2} & -\lambda \end{bmatrix} \otimes |\lambda\rangle\langle\lambda| \\ &= \bigoplus_\lambda \left[\sqrt{1 - \lambda^2}X + \lambda Z\right] \otimes |\lambda\rangle\langle\lambda| \\ &=: \bigoplus_\lambda R(\lambda) \otimes |\lambda\rangle\langle\lambda|, \end{aligned} \tag{25}$$

where $R(\lambda)$ is defined as the operator in square brackets in the penultimate line above and may be interpreted as a reflection and rotation about the y axis of the Bloch sphere, exactly the same as we find for the $R(a)$ operator that appears in the amplitude amplification construction as per Eq. (12). We thus have a form for U which parallels that of the amplitude amplification scenario. However, unlike amplitude amplification, we do not have one-dimensional phase-rotation operators A_ϕ and B_ϕ, because we have N Bloch spheres, and not just two.

Nevertheless, there are still distinct vector spaces in which the input and output of $\mathcal{H}$ exist: these are, respectively, the column space and row space of the matrix $\mathcal{H}$, within U. In the scenario of Eq. (18), these vector spaces are defined by the projector $\Pi := |0\rangle\langle 0|$ acting on the auxiliary qubit. Generalizing the way in which amplitude amplification employs the phase shift A_ϕ of Eq. (5) acting on a single element $|A_0\rangle\langle A_0|$, we may now define a projector-controlled phase-shift operation Π_ϕ:

$$\Pi_\phi := e^{i2\phi\Pi}, \tag{26}$$

which imparts a phase of $e^{i2\phi}$ to the *entire subspace* determined by the projector Π. Note that if we want to be more precise, we may instead define this operation as

$$\Pi_\phi := e^{i\phi(2\Pi - I)}, \tag{27}$$

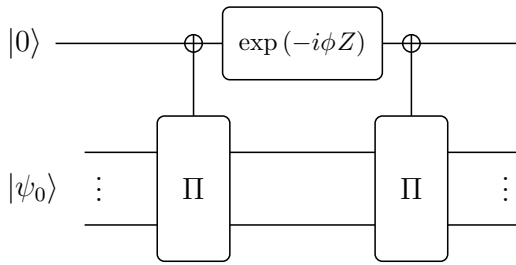which is a proper unitary transform and acts as a z rota-

FIG. 3. The circuit used to realize the projector-controlled phase shift Π_ϕ, also termed the "phased iterate," in analogy to the bare $\exp(i\phi Z)$ operation used in QSP, used as the signal-processing rotation operator.

tion, like $S(\phi)$ from Eq. (2), but these two definitions differ only by a global phase that may be neglected. From a quantum circuit standpoint, Π_ϕ may be realized by employing two instances of projector-controlled-NOT gates (which we refer to as Π-controlled-NOT, or C_ΠNOT for short) around a single-qubit z rotation by angle ϕ, as in Fig. 3:

The main relevant observation is that on the subspace of each of the N Bloch spheres of Eq. (25), Π_ϕ acts as a z-axis rotation,

$$\Pi_\phi = \bigoplus_\lambda e^{i\phi Z} |\lambda\rangle\langle\lambda|, \tag{28}$$

with $|\lambda\rangle$ as an eigenvector.

This picture of N separate Bloch spheres evolving under z and y rotations provides the intuitive basis for the following theorem:

Theorem 3 (Eigenvalue Transformation): *Given a block encoding of Hamiltonian* $\mathcal{H} = \sum_\lambda \lambda |\lambda\rangle\langle\lambda|$ *in a unitary matrix U,*

$$U = \Pi \begin{bmatrix} H & \cdot \\ \cdot & \cdot \end{bmatrix}, \tag{29}$$

with the location of $\mathcal{H}$ *determined by projector* Π *and given the ability to perform* Π-*controlled*-NOT *operations to realize projector-controlled phase-shift operations* Π_ϕ, *then, for even d,*

$$U_{\vec{\phi}} = \left[\prod_{k=1}^{d/2} \Pi_{\phi_{2k-1}} U^\dagger \Pi_{\phi_{2k}} U \right] = \Pi \begin{bmatrix} \mathrm{Poly}(\mathcal{H}) & \cdot \\ \cdot & \cdot \end{bmatrix}, \tag{30}$$

where

$$\mathrm{Poly}(\mathcal{H}) = \sum_\lambda \mathrm{Poly}(\lambda)|\lambda\rangle\langle\lambda| \tag{31}$$

is a polynomial transform of the eigenvalues of $\mathcal{H}$. *The polynomial is of degree at most d and obeys the conditions on P from Theorem 1.*

Similarly, for odd d,

$$U_{\vec{\phi}} = \Pi_{\phi_1} U \left[\prod_{k=1}^{(d-1)/2} \Pi_{\phi_{2k}} U^\dagger \Pi_{\phi_{2k+1}} U \right]$$

$$= \Pi \begin{bmatrix} \mathrm{Poly}(\mathcal{H}) & \cdot \\ \cdot & \cdot \end{bmatrix}, \tag{32}$$

where $\mathrm{Poly}(\mathcal{H})$ *has an analogous interpretation.*

Why does this work? For the same reason that Theorem 2 does. To see this, let us use the specific encoding of Eq. (20) to rewrite Eq. (30) as an action on N separate Bloch spheres [the odd-d case in Eq. (32) being analogous]:

$$U_{\vec{\phi}} = \bigoplus_\lambda \left[\prod_{k=1}^{d/2} e^{i\phi_{2k-1}Z} R(\lambda) e^{i\phi_{2k}Z} R(\lambda) \right], \tag{33}$$

recognizing that we are choosing conventions for $R(\lambda)$ in Eq. (25) such that $R^\dagger(\lambda) = R(\lambda)$. This allows us to relabel the sum and the phases to put Eq. (33) into a standard form of quantum signal processing [similar to how we obtain Eq. (15)]:

$$U_{\vec{\phi}} = \bigoplus_\lambda \left[e^{i\phi_0'Z} \prod_{k=1}^{d} W(\lambda) e^{i\phi_k'Z} \right]. \tag{34}$$

By Theorem 1, the term in brackets is a matrix of polynomials in λ, verifying Theorem 3. Finally, although we are specializing to the specific block encoding of Eq. (20), the proof for more general block encodings is contained in Refs. [8,12].

One immediately apparent utility of this eigenvalue-transformation theorem is its usefulness in filtering eigenvalues. For example, we see in Sec. II A that the BB1 sequence of phases can be used to increase sensitivity to the signal. In this eigenvalue-transformation scenario, the signal is the eigenvalue λ and a sequence of QSP phases can be used to selectively filter a range of eigenvalues, e.g., those below a threshold of $\lambda \approx 2\pi/3$ in the case of BB1 (for now, this is an imprecise statement but a similar question is a major concern of Sec. IV). Measuring the projector Π will show that the QSP sequence flips the index qubit with higher probability if the Hamiltonian has an eigenvalue larger than this threshold, versus when all the eigenvalues of $\mathcal{H}$ are below the threshold. This result also assumes the ability to prepare a state that has some overlap with the relevant eigenstates of $\mathcal{H}$, as input into U,

but the specific amplitudes of the overlap are not crucial, because amplitude amplification can be employed to boost the signal.

D. Quantum singular value transforms

Theorem 3 is known as the eigenvalue transformation because it accomplishes a polynomial transform of the eigenvalues of a matrix embedded within a larger unitary matrix. However, in general the embedded matrix need not have a well-defined set of eigenvalues; for example, instead of being a square matrix, it could be rectangular. Remarkably, the same idea of transforming the embedded matrix using quantum signal processing can still apply. Specifically, we now show that QSP sequences can be used to polynomially transform all the *singular values* of a (possibly nonsquare) matrix A that has been encoded into a block of a unitary matrix U.

Such a general matrix A can be decomposed as

$$A = W_\Sigma \Sigma V_\Sigma^\dagger, \tag{35}$$

where W_Σ and V_Σ are unitary and Σ is diagonal and contains along its diagonal the set of non-negative real numbers $\{\sigma_k\}$, known as the singular values of A, of which there are $r = \text{rank}(A)$ nonzero values. All matrices have such a singular value decomposition (SVD).

As W_Σ and V_Σ are unitary, their columns form orthonormal bases, which we denote by $\{|w_k\rangle\}$ and $\{|v_k\rangle\}$, respectively. $\{|w_k\rangle\}$ are the left singular vectors, which span the left singular vector space, and $\{|v_k\rangle\}$ are the right singular vectors, which span the right singular vector space. Using this notation, we may conveniently rewrite the singular value decomposition of A in a form analogous to the eigenvalue decomposition:

$$A = \sum_{k=1}^{r} \sigma_k |w_k\rangle\langle v_k|. \tag{36}$$

Now, suppose that we are given a unitary matrix U such that A is encoded in a block of U, i.e.,

$$U = \begin{array}{c} \tilde\Pi \\ \end{array} \begin{bmatrix} \overset{\Pi}{A} & \cdot \\ \cdot & \cdot \end{bmatrix}, \tag{37}$$

where $\tilde\Pi := \sum_k |w_k\rangle\langle w_k|$ and $\Pi := \sum_k |v_k\rangle\langle v_k|$ are projectors that locate A within U, such that $A = \tilde\Pi U \Pi$. We again include Π and $\tilde\Pi$ indices adjacent to the matrix representation of U to schematically indicate how A is encoded in U: Π selects the columns and $\tilde\Pi$ the rows in which A is encoded. Moreover, the projectors Π and $\tilde\Pi$ also identify the left and right singular vector spaces, respectively.

For pedagogical simplicity, let us assume for now that A is a square matrix (this assumption is dropped in the next

section). Again specializing to a specific block encoding, we may complete the missing blocks of the unitary U by writing

$$U = \begin{array}{c} 0 \\ 1 \end{array} \begin{bmatrix} \overset{0}{A} & \overset{1}{\sqrt{I - A^2}} \\ \sqrt{I - A^2} & -A \end{bmatrix}, \tag{38}$$

where the 0 and 1 are index qubit labels for the block matrices within U and where $\sqrt{I - A^2}$ is formally defined in terms of the SVD of A, as

$$\sqrt{I - A^2} := \sum_k \sqrt{1 - \sigma_k^2} |w_k\rangle\langle v_k|. \tag{39}$$

It can then be verified straightforwardly that $U^\dagger U = I$, as long as the singular values $\{\sigma_k\}$ are less than or equal to 1 (which may always be achieved by rescaling A to some A/α). Again, while a general block encoding need not take this specialized form, this choice is sufficient for our illustrative purposes. And, moreover, the treatment of general block encodings is presented in Ref. [12], wherein it is shown that a general block encoding takes a form similar to Eq. (38) in a special basis related to the left and right singular vectors of A.

Just as with the analysis of the block-encoded Hamiltonian $\mathcal{H}$, there are multiple Bloch spheres in U. Specifically, note that

$$U|0\rangle|v_k\rangle = \sigma_k|0\rangle|w_k\rangle + \sqrt{1 - \sigma_k^2}|1\rangle|w_k\rangle \tag{40}$$

$$U|1\rangle|v_k\rangle = -\sigma_k|1\rangle|w_k\rangle + \sqrt{1 - \sigma_k^2}|0\rangle|w_k\rangle, \tag{41}$$

so that U may be expressed as a direct sum over a number of separate Bloch spheres equal to the rank of A:

$$
\begin{aligned}
U &= \bigoplus_k \begin{bmatrix} \sigma_k & \sqrt{1 - \sigma_k^2} \\ \sqrt{1 - \sigma_k^2} & -\sigma_k \end{bmatrix} \otimes |w_k\rangle\langle v_k| \\
&= \bigoplus_k \left[\sqrt{1 - \sigma_k^2} X + \sigma_k Z \right] \otimes |w_k\rangle\langle v_k| \\
&=: \bigoplus_k R(\sigma_k) \otimes |w_k\rangle\langle v_k|,
\end{aligned} \tag{42}
$$

where we define $R(\sigma_k)$ analogous to $R(\lambda)$ in Eq. (25). We now have a form for U that directly parallels that of the eigenvalue transform scenario and, for exactly the same reasons, the following theorem holds:

Theorem 4 (Singular Value Transformation)**:** *Given a block encoding of a matrix $A = \sum_k \sigma_k |w_k\rangle\langle v_k|$ in a unitary*

matrix U,

$$
U = \begin{array}{c} \Pi \\ \tilde{\Pi} \end{array} \begin{bmatrix} A & \cdot \\ \cdot & \cdot \end{bmatrix}, \tag{43}
$$

with the location of A determined by projectors Π *and* $\tilde{\Pi}$ *and given the ability to perform* Π- *and* $\tilde{\Pi}$-*controlled-* NOT *operations to realize projector-controlled phase-shift operations* Π_ϕ *and* $\tilde{\Pi}_\phi$ *(defined as in Sec. II C), then, for odd d,*

$$
U_{\vec{\phi}} = \tilde{\Pi}_{\phi_1} U \left[\prod_{k=1}^{(d-1)/2} \Pi_{\phi_{2k}} U^\dagger \tilde{\Pi}_{\phi_{2k+1}} U \right]
$$

$$
= \begin{array}{c} \Pi \\ \tilde{n} \end{array} \begin{bmatrix} \mathrm{Poly}^{(\mathrm{SV})}(A) & \cdot \\ \cdot & \cdot \end{bmatrix}, \tag{44}
$$

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

$$
\mathrm{Poly}^{(\mathrm{SV})}(A) := \sum_k \mathrm{Poly}(\sigma_k) |w_k\rangle\langle v_k|, \tag{45}
$$

which applies a polynomial transform to the singular values of A. The polynomial is of degree at most d and obeys the conditions of P from Theorem 1.

Similarly, for d even,

$$
U_{\vec{\phi}} = \left[\prod_{k=1}^{d/2} \Pi_{\phi_{2k-1}} U^\dagger \tilde{\Pi}_{\phi_{2k}} U \right]
$$

$$
= \begin{array}{c} \Pi \\ \Pi \end{array} \begin{bmatrix} \mathrm{Poly}^{(\mathrm{SV})}(A) & \cdot \\ \cdot & \cdot \end{bmatrix}, \tag{46}
$$

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

$$
\mathrm{Poly}^{(\mathrm{SV})}(A) := \sum_k \mathrm{Poly}(\sigma_k) |v_k\rangle\langle v_k|, \tag{47}
$$

which is also a polynomial transform of the singular values of A but with the modification that the input and output spaces are both the right singular vector space, spanned by $\{|v_k\rangle\}$. *Analogously, the polynomial is of degree at most d and obeys the conditions of P from Theorem 1.*

The main difference between this theorem and the eigenvalue transform of Theorem 3 is that here, the Bloch sphere transformations of *U* also switch between the $\{|v_k\rangle\}$ *and the* $\{|w_k\rangle\}$ *bases*, similar to how the sequence in Theorem 2 [Eq. (13)] flips between bases. Thus, we should now carefully keep track of which vector space the system is in at each stage of operation. In particular, in each of the terms

in square brackets in Eqs. (46) and (44), the *U* on the right moves the system from the $\{|v_k\rangle\}$ basis into the $\{|w_k\rangle\}$ basis and $\tilde{\Pi}_\phi$ then rotates the system around the *z* axis of the Bloch sphere of each left singular vector space. Similarly, the $U^\dagger$ on the left then moves the system back from the $\{|w_k\rangle\}$ basis into the $\{|v_k\rangle\}$ basis and, finally, $\tilde{\Pi}_\phi$ rotates the system around the *z* axis of the Bloch sphere of each right singular vector space. Therefore, the odd-*d* sequence starts in the right singular vector space and ends in the left singular vector space, such that $\mathrm{Poly}^{(\mathrm{SV})}(A)$ is accessed with Π and $\tilde{\Pi}$, whereas the even-*d* sequence starts and ends in the right singular vector space, such that $\mathrm{Poly}^{(\mathrm{SV})}(A)$ is accessed with just Π.

With this crucial modification, Theorem 4 may be verified by following the same logic as that of Theorem 3, while the proof for general block encodings is presented in Ref. [12]. Theorem 4 is the essence of the QSVT algorithm and we elaborate in the next sections upon what can be accomplished with it.

E. Block encodings

A key idea behind the generalization of QSP from single-qubit dynamics to multiqubit transforms is the use of a block-encoded operator, as we have just seen in the last two subsections (and as is discussed in the following sections). While the embedding of one matrix inside a larger one is well known in mathematics as a *dilation* [39], as used here, block-encoded operators are a quantum concept and their construction comes with some caveats. We elaborate on these constructions in this subsection.

The starting point for many applications of QSP and the QSVT is the availability of the desired signal as a linear operator, a matrix *A*, encoded as a block inside a larger unitary matrix *U*, as described in Eq. (37). What is such a block-encoded operator, physically? Some limited intuition comes from the case when *A* is a unitary matrix, in which case *U* is simply a controlled-*A* operator, as depicted in Fig. 4.

When *A* is a large operator, it can be challenging to envision how a controlled-*A* operation may be physically
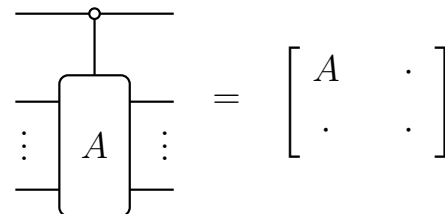

FIG. 4. A simple block-encoding quantum circuit, showing how such a block encoding of a unitary *A* is notated. Observe that the empty circle on the control qubit (top line) indicates a $|0\rangle$ control, whereas a filled-in circle would correspond to a $|1\rangle$ control.
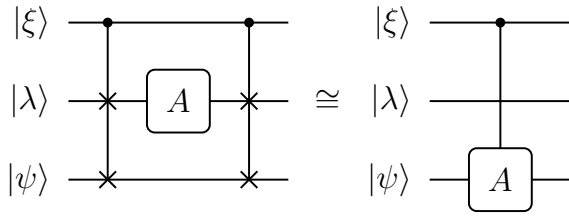
FIG. 5.   The construction of $|1\rangle$-controlled-A (for A a single-qubit operator) given access to $|\lambda\rangle$ a $(+1)$ eigenvector of A, $|\xi\rangle$ the control qubit, and $|\psi\rangle$ the target state. Note that A may in general act on a large Hilbert space, in which case the controlled-swap operator is suitably generalized. The swaps can be made $|0\rangle$ controlled to emulate Fig. 4. We note that this is a similar construction to the circuits employed in Ref. [40] for matrix element measurement.

realized. In particular, given A, how to construct a controlled-A operation is generally not possible without further information. For example, if a $+1$ eigenstate $|\lambda\rangle$ of A is provided, then a controlled-A quantum circuit can be constructed using $|\lambda\rangle$ together with two controlled-SWAP gates (see Fig. 5). However, in general, much more work has to be done in achieving a block encoding, generally in ways specific to the relevant physical system [12,40].

Related to the issue of the block encoding is accessing the desired polynomial transform of the encoded block. As described in Sec. II A, after performing a QSP-QSVT sequence, a measurement may be done in the signal basis to determine if the desired polynomial transform is realized. In particular, in many of the algorithms, we wish to apply the transformed operator, $\mathrm{Poly}^{(SV)}(A)$, to some state $|\psi\rangle$, which may be done using the block encoding, $U_{\vec{\phi}}$. However, this computation will only be successful if we apply to $|\psi\rangle$ the correct block [the one that encodes $\mathrm{Poly}^{(SV)}(A)$], so we must project the final state into being in this block by performing a projective measurement with the projectors Π and $\tilde{\Pi}$, which are assumed accessible. Consequently, the relevant projector for the single-auxiliary-qubit QSVT is often $|+\rangle\langle+| \otimes \Pi$, which both (1) isolates the real part of the transforming polynomial P discussed in Sec. II A and (2) determines if we have applied the desired block of the overall unitary. Since this projective measurement is a one-time projection that is done at the end of the algorithm, its cost is not large and, moreover, the probability of the projective measurement yielding the desired state can be amplified using amplitude amplification or classical repetition. Depending on the construction, this probability can also be either the desired output or an algorithmically useful output. For more about the signal basis and QSP conventions, see Appendix A.

III. SEARCH BY QSVT

Now that we have an overall perspective on the basic theorems of quantum signal processing and quantum singular value transformations, let us return to explicitly constructing the three "primordial" quantum algorithms using the QSVT. Specifically, for each algorithm, we indicate (a) the signal-rotation operator and (b) the appropriate QSP polynomial. This is not just a problem of reconstruction but, in certain cases, it leads to slight improvements in algorithmic performance. We begin with the search algorithm.

A ubiquitous problem that admits a quantum speed-up is unstructured search, the goal of which is to determine a single marked element $m \in \{0, 1, \ldots, N-1\} =: [N]$ from an unstructured database of size N. While the search problem can be solved in $\mathcal{O}(N)$ time classically, Grover's quantum algorithm requires only $\mathcal{O}(\sqrt{N})$ time, providing a provably optimal quadratic speed-up. However, Grover's algorithm in its traditionally stated form suffers from various shortcomings, including divergence from the proper solution if its underlying iterate is repeated too many times. Fortunately, by the parallels between this problem and amplitude amplification as discussed in Sec. II B, it is straightforward to recover an improved version of unstructured search using the QSVT.

In quantum search, one is given access to a unitary operation U the application of which flags some marked element by a phase flip, i.e.,

$$ U|j\rangle = (-1)^{\delta_{jm}}|j\rangle = \begin{cases} |j\rangle & |j\rangle \neq |m\rangle \\ -|j\rangle & |j\rangle = |m\rangle \end{cases}, \quad (48) $$

where $m \in \{0, 1, \ldots, N-1\}$ is the (single) marked element.

As is conventional in quantum search, we take our initial state to be the uniform superposition over all N states, which may be expressed as

$$ |\psi_0\rangle = \frac{1}{\sqrt{N}} \sum_j^N |j\rangle = \sqrt{\frac{N-1}{N}} |m^\perp\rangle + \frac{1}{\sqrt{N}} |m\rangle \quad (49) $$

and is easily prepared by applying n Hadamards to $|0\rangle^{\otimes n}$. With this choice of initialization, our goal is to map $|\psi_0\rangle$ to the marked state $|m\rangle$.

In this context, how might we solve the search problem with the QSVT? The standard method, which includes the construction of the Grover iterate, is not necessary. Instead, we rephrase the problem. In the most general setting, we are given access to some initial state $|\psi_0\rangle$ such that its projection onto the desired final state $|m\rangle$ is nonzero. Denoting this projection by Π, we may express this condition as

$$ \Pi \, V|\psi_0\rangle = a|m\rangle, \quad (50) $$

for some known unitary V and $|a| > 0$. In our case, as illustrated in Eq. (49), we can simply take V as the identity and $a = 1/\sqrt{N}$ by our choice of initial state. In general for amplitude amplification problems, V need not be trivial and is often viewed as a preparation oracle.

The search problem as framed above is precisely the statement of amplitude amplification as in Sec. II B, as we desire a circuit U' (which will depend on U) such that

$$\Big\| \, |m\rangle\langle m|U'|\psi_0\rangle\langle\psi_0| - |m\rangle\langle\psi_0| \, \Big\| < \epsilon, \qquad (51)$$

for some small $\epsilon > 0$. For notational convenience, we refer to $|\psi_0\rangle\langle\psi_0|$ as Π'. Note that this induces a block encoding of the scalar a, that is

$$\Pi \, V \, \Pi' = \Pi \, \Pi' = a|m\rangle\langle\psi_0|. \qquad (52)$$

We now have a problem goal stated in Eq. (51) and a block encoding of a single singular value $a = 1/\sqrt{N}$ stated in Eq. (52). What remains to solve unstructured search with the QSVT is a choice of QSVT polynomial and an analysis of the minimum required QSVT sequence length.

Fortunately, given that there is only one relevant singular value, a, and one relevant left (and right) singular vector, the entire problem is reduced to mapping a as close to 1 in magnitude as possible, as under this condition the circuit will induce, with high probability, the desired block-encoded map $|m\rangle\langle\psi_0|$. Luckily, there already exists a theorem that defines the conditions under which such transformations are possible: the major theorem of the QSVT.

The requirements to perform the QSVT as desired include the application of V, $V^\dagger$, C_ΠNOT, $C_{\Pi'}$NOT, and $e^{-i\phi Z}$ gates. Note that here we are allowed to choose $V = I$, as all Grover-oracle information is encoded in the projector-controlled-NOTs. Moreover, these projector-controlled operators are easy to construct, $C_{\Pi'}$NOT because Π' depends on a fixed known state. Similarly, C_ΠNOT may be crafted by first creating a controlled-U, which is easily achieved, and conjugating the control qubit by Hadamard gates, the result of which is a C_ΠNOT, where the control qubit now becomes the target of the projector-controlled-NOT.

Having now achieved a suitable block encoding and the necessary conditions for the QSVT, we need only choose an appropriate QSVT polynomial. As our goal is to map a to a value close to 1, a sensible choice for such a polynomial can be derived if we start with the simple sign function $\Theta(x - c)$:

$$\Theta(x - c) = \begin{cases} -1, & x < c, \\ 0, & x = c, \\ 1, & x > c. \end{cases} \qquad (53)$$

As discussed in Refs. [12,21], one can estimate the sign function with arbitrary precision by finding a polynomial approximation to $\mathrm{erf}(k[x - c])$ for large enough k. In particular, one can efficiently compute a degree $\mathcal{O}\left[(1/\Delta)\log(1/\epsilon)\right]$ odd polynomial $P^\Theta_{\epsilon,\Delta}(x - c)$, where $\epsilon \in \left(0, \sqrt{2/e\pi}\,\right]$, such that:

(1) $|P^\Theta_{\epsilon,\Delta}(x - c)| \le 1$ for $x \in [-1, 1]$.
(2) $|\Theta(x - c) - P^\Theta_{\epsilon,\Delta}(x - c)| \le \epsilon$
    for $x \in \left[-1, 1\right] \backslash (c - \Delta/2, \, c + \Delta/2)$.

That is, $P^\Theta_{\epsilon,\Delta}(x - c)$ is bounded in magnitude by 1 and ϵ approximates the sign function outside of the region $(c - \Delta/2, \, c + \Delta/2)$. On the other hand, within the range $(c - \Delta/2, \, c + \Delta/2)$, where $\Theta(x - c)$ suffers a discontinuity, $P^\Theta_{\epsilon,\Delta}(x - c)$ is not in general a good approximation to the sign function. In this region, $P^\Theta_{\epsilon,\Delta}(x - c)$ is actually an ϵ approximation to $\mathrm{erf}(k[x - c])$ where $k = (\sqrt{2}/\Delta)\log^{1/2}\left(2/[\pi\epsilon^2]\right)$ (for the explicit construction of $P^\Theta_{\epsilon,\Delta}(x - c)$, see Ref. [21]). For visual intuition, we illustrate this polynomial approximation in Fig. 6.

Note that the polynomial $P^\Theta_{\epsilon,\Delta}(x)$ (i.e., with $c = 0$) has odd parity and is bounded in magnitude by 1 and so it may be implemented via the QSVT (i.e., it obeys the conditions on Poly$(a) = \langle m|U_{\vec\phi}|\psi_0\rangle$ discussed in Sec. II A and Appendix A). We use this scheme to polynomially approximate the sign function, e.g., $\Theta^{(\mathrm{SV})}(V) \approx (P^\Theta_{\epsilon,\Delta})^{(\mathrm{SV})}(V)$, and subsequently apply it to the initial uniform superposition over N quantum states. Then, upon applying the predetermined QSVT sequence, this yields $|m\rangle$ in the register with probability $p = |P^\Theta_{\epsilon,\Delta}(a)|^2$.

We can now ask which values of Δ and ϵ we should choose for our intended behavior. Because we desire that $a = 1/\sqrt{N}$ be mapped to a value greater than $1 - \epsilon$, we require that $1/\sqrt{N} \ge \Delta/2$ or, equivalently, $\Delta = \mathcal{O}(1/\sqrt{N})$. Next, note that upon applying $(P^\Theta_{\epsilon,\Delta})^{(\mathrm{SV})}(V)$
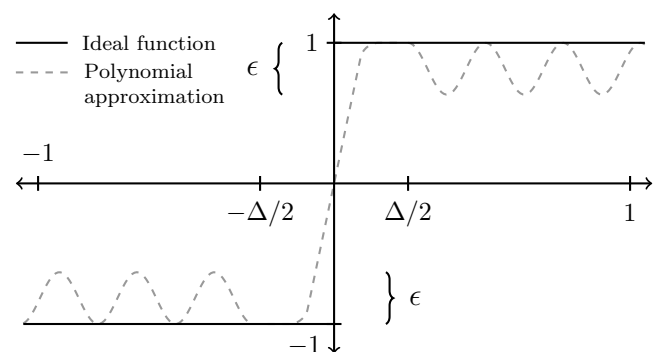

FIG. 6. An illustration of $\Theta(x)$ (solid) and its polynomial approximation, $P^\Theta_{\epsilon,\Delta}(x)$ (dashed). Note how $P^\Theta_{\epsilon,\Delta}(x)$ stays within ϵ of the sign function outside of the region $(\Delta/2, \Delta/2)$, while deviating from $\Theta(x)$ within this region. The shifted version of this function is constructed in much the same way.

Input: Access to a controlled version of the oracle U which bit-flips an auxiliary qubit when given an unknown target state $|m\rangle \in \{|j\rangle, j \in [N]\}$ and acts trivially otherwise, an error tolerance δ, and a $\Delta \leq 2/\sqrt{N}$.

Output: The flagged state $|m\rangle$.

Runtime : $\mathcal{O}\left(\sqrt{N}\log\left(1/\delta\right)\right)$ queries to controlled-U, equivalently C_ΠNOT, $C_{\Pi'}$NOT, and $e^{-i\phi Z}$ gates and a single auxiliary qubit. Succeeds with probability at least $1 - \delta$.

Procedure:

1 Use QSVT to construct the operator $(P_{\delta/2,\Delta}^\Theta)^{(\mathrm{SV})}(V)$, where $V = I$ block encodes $a = 1/\sqrt{N}$ in its $|m\rangle\langle\psi_0|$ element.

2 Apply $(P_{\delta/2,\Delta}^\Theta)^{(\mathrm{SV})}(V)$ to the uniform superposition. With high probability, $|m\rangle$ remains in the register and can be determined by measuring in the computational basis.

Algorithm 1. Unstructured Search by QSVT.

to the uniform superposition, the amplitude of the desired state, $|m\rangle$, is at least $1 - \epsilon$. Thus, upon measuring these qubits in the computational basis, the probability of the resultant state being $|m\rangle$ is at least $(1 - \epsilon)^2 \geq 1 - 2\epsilon$. Therefore, if we want this procedure to succeed with probability at least $1 - \delta$, we set $\epsilon = \delta/2$.

Under these choices, the polynomial $P_{\epsilon,\Delta}^\Theta(x)$ is of degree $d = \mathcal{O}\left[(1/\Delta)\log(1/\epsilon)\right] = \mathcal{O}[\sqrt{N}\log(1/\delta)]$, such that the run time of this algorithm is $\mathcal{O}[\sqrt{N}\log(1/\delta)]$. Note that we again have $\mathcal{O}(\sqrt{N})$ scaling and thus maintain the quadratic speed-up for the search problem, while circumventing Grover search convergence issues. We summarize this algorithm for search by the QSVT in Algorithm 1.

Finally, we remark that this algorithm employs a simple block encoding (i.e., the identity) of $a = 1/\sqrt{N}$ with nontrivial singular vectors (i.e., $|\psi_0\rangle$ and $|m\rangle$). In this setting, our desired transformation would not have been possible had we used the quantum eigenvalue transformation of Sec. II C, as the eigenvalues of V are necessarily 1 and the input and output spaces are identical. Hence, this instance illustrates the advantage of the QSVT over the quantum eigenvalue transformation, as the QSVT can achieve polynomial transformations between different input and output spaces.

IV. THE EIGENVALUE THRESHOLD PROBLEM BY QSVT

Beyond the determination of a desired state, as is the object in unstructured search covered in Algorithm 1 in

Sec. III, a separate major intent of many quantum algorithms is to determine whether a certain property holds of a given quantum system. Through the lens of the QSVT, we show in this section that these two algorithmic goals are similar: here, rather than wishing to prepare a special state (e.g., the marked state for unstructured search), we now wish to encode some underlying unknown information about a quantum system simply into a state that we can then measure.

This linkage between search and property testing is made apparent by considering the *eigenvalue threshold problem*. In the setup of this problem, we are given access to a Hamiltonian $\mathcal{H}$ and would like to determine if $\mathcal{H}$ has any eigenvalues λ that are less than some chosen threshold λ_{th}. Realistically, this must be determined to within some precision Δ_λ, so we are guaranteed that either the Hamiltonian has at least one eigenvalue $\lambda \leq \lambda_{\mathrm{th}} - \Delta_\lambda$ or that all of its eigenvalues obey $\lambda \geq \lambda_{\mathrm{th}} + \Delta_\lambda$. The eigenvalue threshold problem seeks to distinguish these cases.

To solve this problem, one is also provided a state $|\psi\rangle$ that has reasonable overlap with the low-energy subspace (spanned by the eigenvectors with eigenvalues $\lambda \leq \lambda_{\mathrm{th}} - \Delta_\lambda$) *if it exists*. Mathematically, we represent this condition as

$$\|\Pi_{\leq\lambda_{\mathrm{th}}-\Delta_\lambda}|\psi\rangle\| \geq \zeta = \Omega(1), \qquad (54)$$

where $\Pi_{\leq\lambda_{\mathrm{th}}-\Delta_\lambda}$ is a projector onto the (possibly nonexistent) low-eigenvalue subspace mentioned above.

To solve this problem with the QSVT, we need to construct a block encoding of the Hamiltonian $\mathcal{H}$. However, such a block encoding can only be constructed if $\|\mathcal{H}\| \leq 1$, so we instead must determine an $\alpha \geq \|\mathcal{H}\|$ and construct a unitary block encoding of $\mathcal{H}/\alpha$. In general, doing so may be nontrivial because the determination of such an α requires prior knowledge about $\mathcal{H}$. We discuss this drawback in Sec. VII. Fortunately, however, for a large class of Hamiltonians, such as sparse Hamiltonians and linear combinations of unitaries, one can determine a sufficient α and construct a block encoding of $\mathcal{H}/\alpha$ [8,12]. With this rescaled block encoding, one can equivalently imagine that our goal is to determine if $\mathcal{H}/\alpha$ has any eigenvalues less than $\lambda_{\mathrm{th}}/\alpha$ to within a precision Δ_λ/α.

Moreover, as the eigenvalue threshold problem deals with eigenvalues, it is most straightforwardly approached with a quantum eigenvalue transform. However, in accordance with the theme of this tutorial, we may equivalently solve this problem with the QSVT if the eigenvalues of $\mathcal{H}$ are positive, such that the singular values are equal to the eigenvalues. If this is not the case, we may instead use the block encoding of $\mathcal{H}/\alpha$ and the circuit in Fig. 7 to construct a block encoding of the positive definite matrix $\frac{1}{2}(\mathcal{H}/\alpha + I)$. Under this modification, the new goal would be to determine if $\frac{1}{2}(\mathcal{H}/\alpha + I)$ has eigenvalues less than
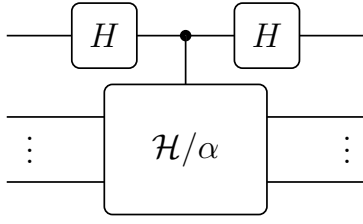
FIG. 7. A simple circuit for block encoding $\frac{1}{2}(\mathcal{H}/\alpha + I)$, the projectors for which are, following QSVT conventions, $|0\rangle\langle0| \otimes \Pi$ and $|0\rangle\langle0| \otimes \tilde{\Pi}$.

$\frac{1}{2}(\lambda_{\text{th}}/\alpha + 1)$ to within a precision Δ_λ/α, which may be achieved with the QSVT. In the remainder of this section, we assume that this issue is alleviated and specialize to the case of distinguishing the threshold $\lambda_{\text{th}}/\alpha$.

With those concerns out of the way, how might this problem be solved with the QSVT? A fruitful approach, which we follow, is to apply the QSVT to the Hamiltonian with a function that filters out large eigenvalues. While one might suspect that we need a function that locates eigenvalues below the threshold $\lambda_{\text{th}}/\alpha$, such as the eigenstate filtering function of Ref. [41], we can actually employ a much simpler construction. All we need is a function that behaves markedly different on eigenvalues greater than and less than $\lambda_{\text{th}}/\alpha$ and such a function may be constructed using the technology from Sec. III.

A good choice for such a function is $\Theta(\lambda_{\text{th}}/\alpha - x)$, which maps eigenvalues less than $\lambda_{\text{th}}/\alpha$ to 1 and larger eigenvalues to -1. One could imagine approximating this function by $P^{\Theta}_{\epsilon,\Delta}(\lambda_{\text{th}}/\alpha - x)$ but there is a catch: this polynomial does not have definite parity and thus does not satisfy the constraints of Theorem 1. Instead, we need to consider a symmetrized version with even parity, the *eigenvalue threshold polynomial*:

$$P^{\text{ET}}_{\epsilon,\Delta,\lambda_{\text{th}}/\alpha}(x) := \frac{1}{1+\frac{\epsilon}{4}}\left[-1+\frac{\epsilon}{4}+P^{\Theta}_{\frac{\epsilon}{2},\Delta}\left(\frac{\lambda_{\text{th}}}{\alpha}-x\right)\right.$$
$$\left. + P^{\Theta}_{\frac{\epsilon}{2},\Delta}\left(\frac{\lambda_{\text{th}}}{\alpha}+x\right)\right]. \quad (55)$$

The factors of $\epsilon/4$ ensure that $P^{\text{ET}}_{\epsilon,\Delta,\lambda_{\text{th}}/\alpha}(x)$ has the following desired properties, which are easily seen via the triangle inequality:

(1) $|P^{\text{ET}}_{\epsilon,\Delta,\lambda_{\text{th}}/\alpha}(x)| \leq 1$ for $x \in [0,1]$.
(2) $|\Theta(\lambda_{\text{th}}/\alpha - x) - P^{\text{ET}}_{\epsilon,\Delta,\lambda_{\text{th}}/\alpha}(x)| \leq \epsilon$
for $x \in [0,1] \setminus (\lambda_{\text{th}}/\alpha - \Delta/2, \lambda_{\text{th}}/\alpha + \Delta/2)$.

So, $P^{\text{ET}}_{\epsilon,\Delta,\lambda_{\text{th}}/\alpha}(x)$ behaves as $P^{\Theta}_{\epsilon,\Delta}(\lambda_{\text{th}}/\alpha - x)$ for $x \geq 0$ (the only relevant range for singular values). In addition, $P^{\text{ET}}_{\epsilon,\Delta,\lambda_{\text{th}}/\alpha}(x)$ has definite parity and magnitude bounded by 1, so it may be implemented through the QSVT [again, it obeys the conditions on Poly$(a) = \langle+|U_{\vec\phi}|+\rangle$ discussed

in Sec. II A and Appendix A]. Thus, this polynomial may be used to distinguish between eigenvalues less than $\lambda_{\text{th}}/\alpha - \Delta/2$ and greater than $\lambda_{\text{th}}/\alpha + \Delta/2$, which naturally indicates a precision $\Delta_\lambda/\alpha = \Delta/2$.

To see how we may employ this technology, note that we may express $|\psi\rangle$ in the eigenbasis as

$$|\psi\rangle = \sum_\lambda c_\lambda|\lambda\rangle = \sum_{\lambda \leq \lambda_{\text{th}}} c_\lambda|\lambda\rangle + \sum_{\lambda > \lambda_{\text{th}}} c_\lambda|\lambda\rangle, \quad (56)$$

where $\sum_{\lambda \leq \lambda_{\text{th}}} |c_\lambda|^2 \geq \zeta^2$ by the guarantee mentioned above. Next, consider using the QSVT to develop $(P^{\text{ET}}_{\epsilon,\Delta})^{(\text{SV})}(\mathcal{H}/\alpha)$ and apply it to $|\psi\rangle$ controlled by an ancilla qubit in the state $|+\rangle$, as in the circuit of Fig. 8. After applying a Hadamard gate to this ancilla qubit, the final (unnormalized) state of the system is

$$\frac{1}{2}\left(|0\rangle\sum_\lambda c_\lambda[1 + P^{\text{ET}}_{\epsilon,\Delta}(\lambda)]|\lambda\rangle\right.$$
$$\left. + |1\rangle\sum_\lambda c_\lambda[1 - P^{\text{ET}}_{\epsilon,\Delta}(\lambda)]|\lambda\rangle\right). \quad (57)$$

We can solve the eigenvalue threshold problem by measuring the ancilla qubit in the computational basis. Observe that the probability of measuring the ancilla qubit in $|0\rangle$ is

$$p(0) = \frac{1}{2}\frac{\sum |c_\lambda|^2[1 + P^{\text{ET}}_{\epsilon,\Delta,\lambda_{\text{th}}/\alpha}(\lambda)]^2}{\sum |c_\lambda|^2\{1 + [P^{\text{ET}}_{\epsilon,\Delta,\lambda_{\text{th}}/\alpha}(\lambda)]^2\}}. \quad (58)$$

If all eigenvalues obey $\lambda > \lambda_{\text{th}} + \Delta_\lambda$, then $-1 \leq P^{\text{ET}}_{\epsilon,\Delta,\lambda_{\text{th}}/\alpha}(\lambda) \leq -1 + \epsilon$ and so

$$p(0| \lambda > \lambda_{\text{th}} + \Delta_\lambda) \leq \frac{\epsilon^2}{2[1 + (-1 + \epsilon)^2]} \leq \frac{1}{2}\epsilon^2. \quad (59)$$

This implies that it is unlikely to measure $|0\rangle$ if all eigenvalues obey $\lambda > \lambda_{\text{th}} + \Delta_\lambda$. On the other hand, if there exists an eigenvalue that obeys $\lambda < \lambda_{\text{th}} - \Delta_\lambda$, then $1 - \epsilon \leq P^{\text{ET}}_{\epsilon,\Delta,\lambda_{\text{th}}/\alpha}(\lambda < \lambda_{\text{th}} - \Delta_\lambda) \leq 1$ and $-1 \leq P^{\text{ET}}_{\epsilon,\Delta,\lambda_{\text{th}}/\alpha}$
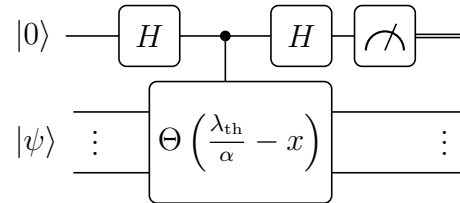

FIG. 8. The circuit that, upon measuring the auxiliary qubit, can be used to solve the eigenvalue threshold problem with high probability as in Algorithm 2.

$(\lambda > \lambda_{\text{th}} + \Delta_\lambda) \leq -1 + \epsilon$, which implies that

$$p\left(0 \mid \exists \lambda < \lambda_{\text{th}} - \Delta_\lambda\right) \geq \frac{\zeta^2(2-\epsilon)^2}{2(1+1)} \geq \zeta^2(1-\epsilon). \quad (60)$$

This indicates that, for reasonably sized ζ, we are more likely to measure $|0\rangle$ if there exists an eigenvalue $\lambda < \lambda_{\text{th}} - \Delta_\lambda$.

With this construction, the eigenvalue threshold problem is reduced to distinguishing between two Bernoulli distributions: one with mean $\leq \frac{1}{2}\epsilon^2$ and one with mean $\geq \zeta^2(1-\epsilon)$. To ensure that these means are well separated, we desire $\zeta^2(1-\epsilon) > \frac{1}{2}\epsilon^2$. Since $\epsilon \leq \sqrt{2/e\pi} \approx 0.48$ by the construction of $P_{\epsilon,\Delta}^\Theta(x)$ in Ref. [21], this may be enforced by choosing an $\epsilon < \zeta$ [i.e., $\epsilon = \mathcal{O}(\zeta)$]; for instance, $\epsilon = 1/4\zeta$ is a sufficient choice, which we make use of in the following. We may then distinguish between these distributions with high probability by performing repeated measurements. This is quantified by the following lemma:

Lemma 5: *Suppose that we have two Bernoulli distributions X_0, X_1 that return elements in $\{0, 1\}$ and that we have expected values $0 \leq a < b \leq 1$, respectively. Given a distribution X that is either X_0 or X_1, we may determine the identity of X with confidence greater than $1 - \delta$ for $\delta > 0$ with $\mathcal{O}\{[b + a/(b-a)^2]\log(1/\delta)\}$ samples of X.*

Proof. To distinguish between these distributions, let us take n samples X_i and guess that X is the distribution the mean of which is closest to the empirical mean of the samples. Suppose that $X = X_0$ and thus has mean a (the case $X = X_1$ may be treated analogously). We are interested in computing the probability of mistaking this distribution for X_1, namely

$$P(\text{error}) = P\left(\sum_{i=1}^n X_i \geq n(a+b)/2\right)$$
$$= P\left(\sum_{i=1}^n X_i \geq n[a + (b-a)/2]\right). \quad (61)$$

To bound this error probability, we may invoke the well-known multiplicative Chernoff bound, with mean $\mu = \mathbb{E}(\sum_{i=1}^n X_i) = na$ and relative difference from the mean ε,

$$\mu + n(b-a)/2 = \mu(1+\varepsilon) \implies \varepsilon = \frac{b-a}{2a}. \quad (62)$$

Employing the multiplicative Chernoff bound, we may ensure that the error probability is less than some chosen δ

as

$$P\left(\sum_{i=1}^n X_i \geq na(1+\varepsilon)\right) \leq e^{-\varepsilon^2 na/3} \leq \delta, \quad (63)$$

which in turn requires $n \geq (3/a\varepsilon^2)\log(1/\delta)$ or, equivalently, $n \geq [12a/(b-a)^2]\log(1/\delta) = \mathcal{O}\{[a/(b-a)^2]\log(1/\delta)\}$.

Performing a similar calculation for the hypothesis $X = X_1$ yields $n = \mathcal{O}\{[b/(b-a)^2]\log(1/\delta)\}$. Summing these results over equal priors over $X \in \{X_0, X_1\}$ provides the scaling $n \geq [6(a+b)/(b-a)^2]\log(1/\delta) = \mathcal{O}\{[a+b/(b-a)^2]\log(1/\delta)\}$, as claimed in the lemma statement. ∎

Applying this lemma to our scenario in the eigenvalue threshold problem, we have $1/b - a \leq 1/\zeta^2(1-\epsilon) - \epsilon^2/2 = \mathcal{O}(1/\zeta^2)$. For instance, with our choice $\epsilon = \zeta/4$, we have $1/b - a < 3/2\zeta^2$. We then have $a + b/(b-a)^2 = (b-a) + 2a/(b-a)^2 = \mathcal{O}(1/\zeta^2 + \zeta^2/\zeta^4) = \mathcal{O}(1/\zeta^2)$. Therefore, if we repeat the aforementioned measurement procedure $\mathcal{O}\left[(1/\zeta^2)\log(1/\delta)\right]$ times, we can correctly distinguish between the two distributions and solve the eigenvalue threshold problem with probability at least $1 - \delta$. As each repetition performs the

---

Input: Access to a Hamiltonian, $\mathcal{H}$, and an $\alpha \geq \|\mathcal{H}\|$.

Output: Whether $\mathcal{H}$ has at least one eigenvalue $\lambda < \lambda_{\text{th}} - \Delta_\lambda$, or if all eigenvalues obey $\lambda > \lambda_{\text{th}} + \Delta_\lambda$.

Runtime : $\mathcal{O}\left(\frac{\alpha}{\Delta_\lambda \zeta^2}\log(1/\zeta)\log(1/\delta)\right)$ uses of a unitary block encoding of $\mathcal{H}/\alpha$ and its inverse, and C_ΠNOT, $C_{\tilde{\Pi}}$NOT and single qubit phase gates. Succeeds with probability at least $1 - \delta$.

Procedure:
1 Prepare a unitary block encoding of $\mathcal{H}/\alpha$.
2 **for** $i = 1$ *to* $\lceil \frac{315}{32\zeta^2}\log(1/\delta)\rceil$ **do**
3   Use QSVT to apply $\left(P_{\frac{\zeta}{4}, \frac{2\Delta_\lambda}{\alpha}, \frac{\lambda_{\text{th}}}{\alpha}}^{\text{ET}}\right)^{(\text{SV})}(\mathcal{H}/\alpha)$ to $|\psi\rangle$, controlled by an ancilla qubit in the state $|+\rangle$ (see Fig. 8).
4   Apply a Hadamard to the ancilla qubit and measure it in the computational basis.
5 If the fraction of $|0\rangle$'s measured is closer to $\zeta^2(1-\zeta/4)$ than to $\frac{\zeta^2}{32}$, then there exists an eigenvalue $\lambda \leq \lambda_{\text{th}} - \Delta_\lambda$ with high probability. Otherwise, all eigenvalues obey $\lambda \geq \lambda_{\text{th}} + \Delta_\lambda$ with high probability.

Algorithm 2. The Eigenvalue Threshold Problem by the QSVT.

QSVT with a polynomial of degree $\mathcal{O}\left[(1/\Delta)\log(1/\epsilon)\right] = \mathcal{O}\left[(\alpha/\Delta_\lambda)\log(1/\zeta)\right]$, this overall algorithm queries the Hamiltonian $\mathcal{O}\left[(\alpha/\Delta_\lambda\zeta^2)\log(1/\zeta)\log(1/\delta)\right]$ times. For consistency, we note that this $1/\zeta^2$ dependence is suboptimal and can be improved to $1/\zeta$ using Lemma 7 of Ref. [42]. We summarize the eigenvalue threshold problem by the QSVT in Algorithm 2. Here, we note that the upper limit on the index of the for loop comes from the condition $n \geq [6(a+b)/(b-a)^2]\log(1/\delta)$ according to Lemma 5, with the choice $\epsilon = \zeta/4$.

V. PHASE ESTIMATION BY QSVT

Another salient problem that admits a quantum advantage is *phase estimation*, which is as follows: given access to a unitary oracle U and an eigenvector $|u\rangle$ such that $U|u\rangle = e^{2\pi i\varphi}|u\rangle$, determine φ. Here, we approach this problem by integrating a feedback procedure inspired by Kitaev's algorithm [20,43], with the technology developed in solving the eigenvalue threshold problem in Sec. IV. We first provide a sketch of the algorithm and then address a few caveats in its construction. Finally, we present a concrete formulation of QSVT-based phase estimation, analyze its capabilities, and demonstrate how the quantum Fourier transform naturally emerges from this construction.

A. Intuition

To motivate iterative phase estimation by QSVT, recall that Kitaev's algorithm for phase estimation [20,43] may be reinterpreted as a semiclassical feedback process [44,45]. One may view this feedback process as incorporating a classical parameter $\theta \in [0,1]$, initialized at $\theta = 0$, such that through a series of controlled-U^j calls and a single-qubit measurements, each of which updates θ, one obtains a value $\theta \approx \varphi$ at the end of the algorithm.

Our approach to phase estimation by QSVT is heavily inspired by this feedback procedure. Just as in Kitaev's algorithm, we introduce a parameter $\theta \in [0,1]$, initialized at 0, that is updated throughout the algorithm and, by the end of the procedure, the final value of θ will approximate φ. We sketch our algorithm in the following section.

Before proceeding, we note that a very similar QSVT-based phase estimation algorithm has independently been discovered in Ref. [17]. A primary goal of that work is to maintain coherence and, accordingly, the tutorial presents a thorough performance analysis of the authors' algorithm, including the derivation of constant factors as well as precise use of the diamond norm. Reference [17] also extends their QSVT-based phase estimation algorithm to coherent energy and amplitude estimation.

1. Sketch of the algorithm

To understand our algorithm, let us use the binary-fraction representation of φ and assume that φ is an m-bit

number: $\varphi = 0.\varphi_1\varphi_2\cdots\varphi_m$ for some finite m (in this representation, $\varphi = \sum_j 2^{-j}\varphi_j$). We show how to deal with the $m = \infty$ case later.

In the QSVT phase estimation algorithm, φ is determined bit by bit through a series of m iterations, beginning with the least significant bit, φ_m, and proceeding down to the most significant bit, φ_1. In each iteration, we construct a matrix the singular values of which encode the least significant bits of φ as well as the current value of θ. By applying the QSVT to this matrix, conditioned on an ancilla qubit, and subsequently measuring the ancilla qubit, we determine a single bit of φ and update θ accordingly. If each iteration succeeds, then $\theta = \varphi$ at the end of the algorithm.

To see how this may be achieved, consider the matrix $A_j(\theta) := \frac{1}{2}(I + e^{-2\pi i\theta}U^{2^j})$, which has a singular value

$$\sigma^j := \left|\cos\left[\pi(2^j\varphi - \theta)\right]\right|, \qquad (64)$$

with corresponding right singular vector $|v^j\rangle = |u\rangle$ and left singular vector $|w^j\rangle = e^{i\pi(2^j\varphi - \theta)}|u\rangle$. Because $\exp(\pi i2^j\varphi) = \exp(2\pi i2^{j-1}0.\varphi_1\cdots\varphi_m) = \exp(\pi i0.\varphi_{j+1}\cdots\varphi_m)$, σ^j may be reexpressed as

$$\sigma^j = \left|\cos\left[\pi(0.\varphi_{j+1}\varphi_{j+2}\cdots\varphi_m - \theta)\right]\right|. \qquad (65)$$

This singular value is used to extract the bits of φ as follows. Consider the first iteration of the QSVT phase estimation algorithm, where we begin with $j = m - 1$ (and decrement j down to $j = 0$). Because θ is initialized at 0, the singular value of interest is

$$\sigma^{m-1} = \left|\cos\left[\pi(0.\varphi_m)\right]\right| = \begin{cases} 1 & \varphi_m = 0 \\ 0 & \varphi_m = 1, \end{cases} \qquad (66)$$

from which φ_m may be evaluated as $\varphi_m = \frac{1}{2}\left[1 + \Theta\left(1/\sqrt{2} - \sigma^{m-1}\right)\right]$. This expression provides a pathway to extract φ_m, akin to the procedure used in the eigenvalue threshold problem: apply the QSVT to $A_{m-1}(\theta)$ with the target function $\Theta\left(1/\sqrt{2} - x\right)$ [46] and employ the circuit in Fig. 8, the measurement result of which is φ_m. After measurement, we update θ by setting the first bit of θ to $\theta_1 = \varphi_m$ (again using the binary-fraction representation $\theta = 0.\theta_1\theta_2\cdots$), such that $\theta = 0.\varphi_m$, which completes the $j = m - 1$ iteration.

We then move to the next iteration, wherein we decrement $j = m - 1 \leftarrow m - 2$, map $\theta \leftarrow \theta/2 = 0.0\varphi_m$, and construct $A_j(\theta) = A_{m-2}(0.0\varphi_m)$, which has singular value

$$\sigma^{m-2} = \left|\cos\left[\pi(0.\varphi_{m-1}\varphi_m - 0.0\varphi_m)\right]\right|$$
$$= \left|\cos\left[\pi(0.\varphi_{m-1})\right]\right|. \qquad (67)$$

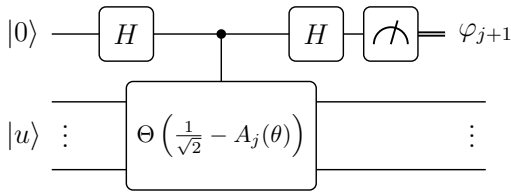Note that the updated value of θ allows for a convenient

FIG. 9. The quantum circuit used to evaluate φ_{j+1} at each iteration of the phase estimation-by-QSVT procedure.

cancellation of φ_m. As this relation is identical to Eq. (66), φ_{m-1} may be determined by employing the QSVT procedure mentioned above and again executing the circuit in Fig 9. Upon obtaining the measurement result of this circuit, we set $\theta_1 = \varphi_{m-1}$.

It is clear that this procedure may be repeated to determine each bit of φ, such that at the end of the $j = 0$ iteration, one obtains $\theta = \varphi$, which solves the phase estimation problem. Ultimately, this procedure succeeds because θ encodes the least significant bits of φ, from which the next bit of φ may be extracted by cleverly transforming the singular values of $A_j(\theta)$. Accordingly, this algorithm may be seen as a binary search through the bits of φ, a key idea to which we return in Sec. V D.

At this stage, however, one may object to this algorithm sketch, as it requires m iterations, and it could be the case that m is unknown or prohibitively large. This is not an issue, as we prove in Appendix B 1 that if we start at $j = n - 1$ for some positive integer n, then we have (with a minor procedural modification needed for $n < m$):

Theorem 6: *If $n \geq m$, and one can exactly implement the sign function through the QSVT, then at the end of this procedure, $\theta = \varphi$.*

and

Theorem 7: *If $n < m$ (including the case $m = \infty$) and one can exactly implement the sign function through the QSVT, then at the end of this procedure, $|\theta - \varphi| \leq 2^{-n-1}$.*

As evidenced by these theorems, which are easily proven by induction, this procedure necessarily outputs an n-bit approximation to φ and is the essence of the algorithm for phase estimation by QSVT. However, in spite of the simple appearance of this algorithm, a few issues must be addressed before presenting its concrete formulation.

2. Caveats

Here, we address a few caveats in the above algorithm sketch. First, as described thus far, this procedure applies the QSVT to the matrix $A_j(\theta)$, which requires a unitary block encoding of $A_j(\theta)$. Fortunately, this encoding can
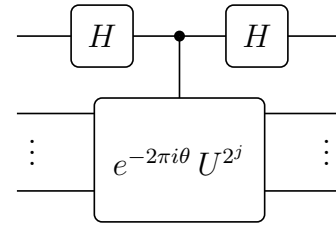

FIG. 10. A quantum circuit that achieves a unitary block encoding of $A_j(\theta)$, which we denote by $W_j(\theta)$ and incorporate into Algorithm 3.

be easily constructed as the circuit in Fig. 10, which we denote by $W_j(\theta)$. A simple calculation of Fig. 10 indicates that $W_j(\theta)$ has the desired form:

$$W_j(\theta) := \frac{1}{2} \begin{pmatrix} I + e^{-2\pi i \theta} U^{2^j} & I - e^{-2\pi i \theta} U^{2^j} \\ I - e^{-2\pi i \theta} U^{2^j} & I + e^{-2\pi i \theta} U^{2^j} \end{pmatrix}$$
$$= \begin{pmatrix} A_j(\theta) & \cdot \\ \cdot & \cdot \end{pmatrix}, \tag{68}$$

with corresponding projection operators $\Pi = \tilde{\Pi} = |0\rangle\langle 0| \otimes I$.

Another concern with the aforementioned procedure is that the sign function, $\Theta\left(1/\sqrt{2} - x\right)$, may seem overcomplicated and unnecessary, as the same results could be attained with a simpler function, such as $1 - x$. However, the sign function is crucial when one desires merely an n-bit approximation to φ, which is necessary when m is unknown or prohibitively large, as is often the case in practice. In this situation, the use of the sign function (the exact sign function, rather than an approximation) will ultimately set θ equal to the n-bit number closest to φ. For example, if $0.\varphi_m\varphi_{m+1}\cdots = 0.011\cdots$ at the first iteration, then despite the fact that $\varphi_m = 0$, the use of the sign function will set $\theta_1 = 1$ because 0.1 is the best 1-bit approximation to $0.011\cdots$.

Lastly, the most significant problem with this procedure as it currently stands is that the sign function is not a polynomial and therefore cannot be implemented exactly in the QSVT. Instead, $\Theta\left(1/\sqrt{2} - x\right)$ must be approximated by a polynomial, a good choice for which is $P_{\epsilon,\Delta}^{\Theta}\left(1/\sqrt{2} - x\right)$ from Sec. III. However, as in Sec. IV, in order to ensure that the target polynomial has definite parity, we must use a symmetrized polynomial, the *phase estimation polynomial*:

$$P_{\epsilon,\Delta}^{\text{PE}}(x) := \frac{1}{1 + \frac{\epsilon}{4}} \left[-1 + \frac{\epsilon}{4} + P_{\frac{\epsilon}{2},\Delta}^{\Theta}\left(\frac{1}{\sqrt{2}} - x\right) \right.$$
$$\left. + P_{\frac{\epsilon}{2},\Delta}^{\Theta}\left(\frac{1}{\sqrt{2}} + x\right) \right]. \tag{69}$$

Like the eigenvalue threshold polynomial, $P_{\epsilon,\Delta}^{\text{PE}}(x)$ is even, is bounded in magnitude by 1, and behaves as

$P_{\epsilon,\Delta}^{\Theta}\left(1/\sqrt{2} - x\right)$ for $x \geq 0$, which is the only relevant range for singular values.

This approximation comes with the potential for error, as an iteration of this procedure may now fail with a nonzero probability, where failure is defined as measuring an incorrect bit of an approximation to φ (an approximation that suffers error $\leq 2^{-n-1}$), which results in an inaccurate θ such that $|\theta - \varphi| \geq 2^{-n-1}$. Therefore, we reformulate our goal probabilistically as follows: obtain an n-bit approximation to φ with probability at least $1 - \delta$.

The probability of failure is dictated by the values of Δ and ϵ. In particular, for a nonzero Δ, note that if $\sigma^j \in \left[1/\sqrt{2} - \Delta/2, \ 1/\sqrt{2} + \Delta/2\right]$, where $P_{\epsilon,\Delta}^{\text{PE}}(x)$ is not a good approximation to the sign function, then an error may occur. Fortunately, if Δ is made sufficiently small, this error mode does not induce significant errors:

Theorem 8: *If we choose Δ such that*

$$\Delta < 2\left[\cos\left(\frac{3\pi}{16}\right) - \frac{1}{\sqrt{2}}\right] \approx 0.25, \qquad (70)$$

then $\sigma^j \in \left[1/\sqrt{2} - \Delta/2, 1/\sqrt{2} + \Delta/2\right]$ is only possible at the $(j = n - 1)$th iteration, such that an error due to Δ can only occur at the jth iteration. If an error is made at this iteration, then at the end of the algorithm, then $|\theta - \varphi| < 1/2^n$, assuming no errors are made at later iterations.

This Theorem, proven in Appendix B 2, indicates that the error caused by Δ is at most 2^{-n} when we obey this bound, which we assume is obeyed in the following.

Next, a nonzero ϵ can also induce errors. Because $P_{\epsilon,\Delta}^{\text{PE}}(x) \epsilon$ approximates $\Theta\left(1/\sqrt{2} - x\right)$ outside of $\left[1/\sqrt{2} - \Delta/2, 1/\sqrt{2} + \Delta/2\right]$, the failure probability of the measurement in Fig. 9 goes as $\mathcal{O}(\epsilon^2)$ and this type of error may occur at any iteration. To study this quantitatively, consider iteration $j < n - 1$, such that $\sigma^j \notin \left[1/\sqrt{2} - \Delta/2, 1/\sqrt{2} + \Delta/2\right]$, and focus on the scenario $\sigma^j < 1/\sqrt{2} - \Delta/2$, in which case $P_{\epsilon,\Delta}^{\text{PE}}(\sigma^j) \in [-1, -1 + \epsilon]$ and $\theta_1 = 0$ is the ideal measurement (the case in which $\sigma^j > 1/\sqrt{2} + \Delta/2$ is analogous). By evaluating the circuit in Fig. 9, with $P_{\epsilon,\Delta}^{\text{PE}}(\sigma^j)$ in place of the sign function, we find that the final (unnormalized) state of the ancilla qubit is

$$\frac{1}{2}\left\{\left[1 - P_{\epsilon,\Delta}^{\text{PE}}(\sigma^j)\right]|0\rangle + \left[1 + P_{\epsilon,\Delta}^{\text{PE}}(\sigma^j)\right]|1\rangle\right\}, \qquad (71)$$

such that the probability of failure (measuring $|1\rangle$) at this iteration is

---

Input: An oracle that performs a controlled-U^j operation, an eigenstate $|u\rangle$ of U with eigenvalue $e^{2\pi i \varphi}$, and $n + 1$ ancilla qubits (or 1 ancilla qubit that is reused $(n + 1)$ times). Also, an $\epsilon \leq \sqrt{2\delta/(n+1)}$ and a $\Delta < 2\left(\cos\left(\frac{3\pi}{16}\right) - \frac{1}{\sqrt{2}}\right) \approx 0.25$.

Output: A θ such that $|\theta - \varphi| \leq \frac{1}{2^n}$.

Runtime : $\mathcal{O}\left(n\log\left(\frac{n}{\delta}\right)\right)$ queries to the controlled-U^j oracle. Succeeds with with probability $\geq 1 - \delta$.

Procedure:

1   Initialize $\theta = 0$.
2   Apply a Hadamard gate to each auxiliary qubit.
3   **for** $j = n - 1$ *down to* 0 **do**
4     $\theta \leftarrow \theta/2$ (equivalently, $\theta = 0.\theta_1\theta_2... \leftarrow 0.0\theta_1\theta_2...$)
5   Construct a block encoding of $A_j(\theta) := \frac{1}{2}(I + e^{-2\pi i \theta}U^{2^j})$ as per Figure 10.
6   Use QSVT to apply the operator $(P_{\epsilon,\Delta}^{\text{PE}})^{(\text{SV})}(A_j(\theta))$ to $|u\rangle$, controlled on an ancilla qubit, where $P_{\epsilon,\Delta}^{\text{PE}}(x)$ is defined in Eq. (69).
7   Apply a Hadamard gate to the ancilla qubit, and measure it in the computational basis.
8   Set the first bit of θ (i.e. θ_1) equal to the result of this measurement.
9   With $j = 0$, repeat lines 5-8 of the above loop.
10   Set the first bit of θ (i.e. θ_0) equal to the result of the measurement.

Algorithm 3. Phase Estimation by QSVT.

$$p_{\text{fail}} = \frac{\left[1 + P_{\epsilon,\Delta}^{\text{PE}}(\sigma^j)\right]^2}{2\left\{1 + \left[P_{\epsilon,\Delta}^{\text{PE}}(\sigma^j)\right]^2\right\}} \leq \frac{\epsilon^2}{2\left[1 + (-1 + \epsilon)^2\right]} \leq \frac{1}{2}\epsilon^2 \qquad (72)$$

for $\epsilon \leq 1$.

What ϵ is sufficient for our purposes? As our procedure consists of n iterations, we desire that each iteration fails with probability no greater than δ/n, such that the overall algorithm succeeds with probability at least $1 - \delta$ by the union bound. From the above inequality, this can be enforced by the choice $\epsilon \leq \sqrt{2\delta/n}$. Therefore, with these conditions on Δ and ϵ, this algorithm sketch succeeds with probability $\geq 1 - \delta$ at determining a θ such that $|\theta - \varphi| < 1/2^n$, as desired.

B. The complete algorithm

After supplying intuition and addressing potential issues, we now present the complete algorithm for phase estimation by QSVT in Algorithm 3. We depict the circuit for this algorithm in Fig. 11, which illustrates its resemblance to the inverse quantum Fourier transform, a connection on which we elaborate in Sec. V D. We further unpack the details of the phase estimation-by-QSVT

circuit in Fig. 12; here, the operator R_ℓ is defined as

$$R_\ell := \begin{bmatrix} 1 & 0 \\ 0 & e^{2\pi i/2^\ell} \end{bmatrix}, \tag{73}$$

which is a z rotation, up to a global phase.

Note that we include the additional lines 9 and 10 in the algorithm to account for the bit in the ones place of θ (i.e., θ_0), which is needed when $n < m$ such that θ is an approximation to φ. Typically, $\theta_0 = 0$, but $\theta_0 = 1$ is possible in the scenario that 1.0 is a good approximation to φ (for example, rounding 0.11111 to two binary decimals yields 1.00). As a result of this additional iteration, we must now require $\epsilon \leq \sqrt{2\delta/(n+1)}$ to ensure that the algorithm succeeds with probability at least $1 - \delta$.

Also observe that Algorithm 3 consists of $\mathcal{O}(n)$ iterations, each of which performs an instance of the QSVT with a polynomial of degree $\mathcal{O}\left[(1/\Delta)\log(\frac{1}{\epsilon})\right] = \mathcal{O}\left[\log(\frac{n}{\delta})\right]$ degree polynomial, so phase estimation by QSVT queries the controlled-U^j oracle $\mathcal{O}\left[n\log(n/\delta)\right]$ times. In addition, the multiplicative factor $1/\Delta$ is not particularly prohibitive because $\Delta = \mathcal{O}(1)$ need only satisfy $\Delta < 2\left(\cos(3\pi/16) - 1/\sqrt{2}\right) \approx 0.25$.

While this $\mathcal{O}(n\log n)$ query complexity may appear to provide a speed-up over conventional phase estimation, the gate count of which scales as $\mathcal{O}(n^2)$, we note that phase estimation by QSVT requires $\mathcal{O}(n\log n)$ queries to the controlled-U^j oracle, whereas conventional phase estimation requires only $\mathcal{O}(n)$ queries [18]. Nonetheless, we believe that this logarithmic factor could be removed by integrating phase estimation by QSVT with a more sophisticated phase estimation protocol, such as the inference procedures of Refs. [47,48], which already attain speed-ups over conventional phase estimation. Indeed, the QSVT-based phase estimation algorithm of Ref. [17] requires only $\mathcal{O}(n)$ queries to the oracle, which is achieved by using a procedure very similar to

Algorithm 3 but varying the degree of the QSVT polynomial at each iteration (schematically, by increasing Δ at each iteration).

Likewise, phase estimation by QSVT may be modified to be more applicable to specialized scenarios, such as if one is restricted in the polynomials that may be implemented through the QSVT or if one has prior knowledge about φ. For example, suppose that one cannot implement $\epsilon \leq \sqrt{2\delta/(n+1)}$ with certainty and so, instead, must choose $\epsilon = \mathcal{O}(1)$. To alleviate this difficulty, one could repeat the measurement in line 7 $\mathcal{O}[\log(n/\delta)]$ times and set θ_1 equal to the majority vote of the measurement results, which yields an accurate value of θ with probability $\geq 1 - \delta$. Similarly, suppose that the constraint on Δ cannot be implemented. Then, again using repeated measurements, if σ^{n-1} is very close to $1/\sqrt{2}$, the majority of the measurement results may not reflect the correct choice of θ_1. But with high probability, the measurement results will be ambiguous, signaling that σ^{n-1} is near $1/\sqrt{2}$. In this event, one could move to an even higher iteration—say, some $j > n-1$—where it is likely that σ^j is not near $1/\sqrt{2}$ and φ_{j+1} is easily determined (and if σ^j is again near $1/\sqrt{2}$, one could again move to a higher iteration $j' > j$). With this bit correctly determined, one may proceed through the rest of the iterations and attain a good estimate of φ.

C. Applications to factoring and beyond

Here, we discuss how phase estimation by QSVT may be applied to the factoring problem and used for robust phase estimation.

1. Factoring

Phase estimation by QSVT may be straightforwardly applied to the factoring problem. Recall that in the factoring problem, the oracle U behaves as $U|u\rangle = |xu(\bmod N)\rangle$ for some $x < N$ and has eigenvalues $e^{2\pi is/r}$, where r is the order of U [i.e., $x^r = 1(\bmod N)$] and $s \in \{0, 1, \dots, r-1\}$
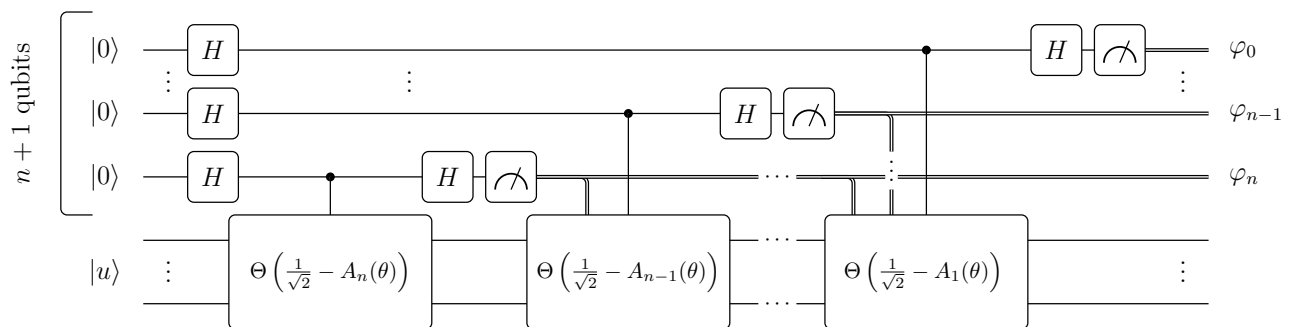

FIG. 11. An abstract overview of the circuit performing phase estimation through the QSVT (Algorithm 3). The double lines indicate that measurement results are fed forward (via the parameter θ) to all future controlled QSVT operations, where this adaptive process is unpacked in Fig. 12. In essence, the circuit systematically computes the least significant bit of an unknown quantum phase, adaptively bit shifts this phase, and repeats. Note the similarity of this operation to the inverse quantum Fourier transform, a connection on which we elaborate in Sec. V D.
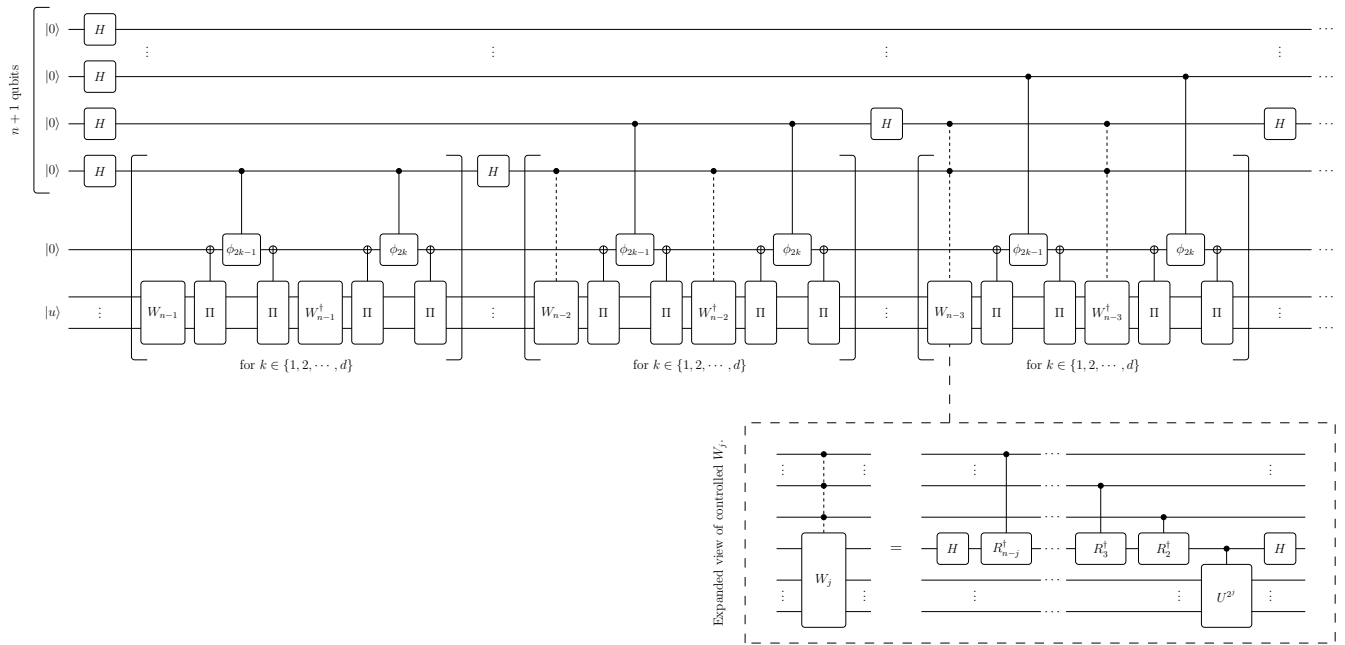
FIG. 12. A quantum circuit for performing phase estimation using the QSVT, following Algorithm 3. The dotted controlled operators, denoted by W_j at step j, represent a composite gadget (unwrapped in the inset block) that depends on the states of $(n - j - 2)$ controlling qubits above it. Here, controlled-ϕ_j gates represent controlled-Z rotations by angle ϕ_j. Additionally, this figure depicts only the even-d case of Theorem 4. This circuit terminates by measuring all auxiliary qubits in the computational basis. Also depicted is an expanded view of a subroutine in the circuit depicted for phase estimation. Note that the W_j operator is the unitary that block encodes A_j, as depicted in Fig. 10. Note also that each R_ℓ operator is defined as in Eq. (73). The controlled R_ℓ operations serve to adaptively zero out the currently estimated phase to the $(n - j - 2)$th bit.

[18]. The goal is to determine some eigenphase s/r, from which r may be determined by the continued fractions algorithm and a factor of N may be extracted. In addition, in the factoring problem, one does not have direct access to an eigenstate $|u_s\rangle$ but, instead, one can prepare a uniform superposition of eigenstates $1/\sqrt{r} \sum_s |u_s\rangle$.

First, consider employing the phase estimation-by-QSVT algorithm but replacing $|u\rangle$ with the superposition $1/\sqrt{r} \sum_s |u_s\rangle$. With this modification, the phase estimation algorithm begins with the superposition state $1/\sqrt{r} \sum_s |u_s\rangle$ and converges to a single eigenstate at the end of the algorithm. To see this, note that each measurement of an ancilla qubit will restrict the state to be a superposition over eigenstates the eigenvalues of which are consistent with the measurement results thus far (i.e., the eigenphases the least significant bits of which agree with the measurement results). Because the eigenvalues are not degenerate, the state at the end of the algorithm will be an eigenstate—say, $|u_s\rangle$—the eigenphase of which is $\theta \approx s/r$, to which the continued fractions algorithm may be applied to determine r.

How accurate must the approximation $\theta \approx s/r$ be? Note that if our estimate θ obeys

$$\left| \frac{s}{r} - \theta \right| \leq \frac{1}{2r^2}, \tag{74}$$

then we can determine the fraction s/r by applying the continued fractions algorithm to θ [18]. We can ensure that this condition is met by selecting $n = 2 \log_2(N) + 1$, such that $|s/r - \theta| \leq 2^{-n} = 1/2N^2 \leq 1/2r^2$, as desired. Thus, subject to these conditions, phase estimation by QSVT may be used to factor N in time $\mathcal{O}\left[n \log\left(n/\delta\right)\right] = \mathcal{O}\left[\log N \log\left(\log N/\delta\right)\right]$.

2. Robust phase estimation

Phase estimation by QSVT comes in handy when U cannot be implemented reliably. For instance, suppose that U^{2^j} can only be implemented approximately, such that the error in U^{2^j} may be interpreted as an additive error in the quantity $0.\varphi_{j+1}\varphi_{j+2}\cdots\varphi_m$ of Eq. (65) (i.e., the erroneous singular value is $\tilde{\sigma}^j = |\cos[\pi(0.\varphi_{j+1}\varphi_{j+2}\cdots + \varepsilon - \theta)]|$, where ε is the additive error). If Δ is made sufficiently small, then these errors are not large enough to induce an incorrect measurement result with high probability and so these errors can be tolerated.

This robustness to error can be quantified via the analysis in Appendix B 2. Formally, if we choose a small enough Δ such that $1/4 - 1/\pi \arccos(1/\sqrt{2} + \Delta/2) < \gamma$ for some $\gamma > 0$, then we can tolerate additive errors in $0.\varphi_{j+1}\varphi_{j+2}\cdots$ of magnitude $< 1/8 - 3\gamma/2$. In this sense, phase estimation by QSVT provides protection against

noise of this form, which is a nice improvement over conventional phase estimation.

D. Emergent quantum Fourier transform

An integral component of Algorithm 3 is the cascading of multiple QSVT sequences, using the results from one instance to control subsequent instances. In its simplest case, this cascading structure reduces to the celebrated quantum Fourier transform (QFT) and generalizes the QFT in more sophisticated constructions. We elaborate on this key connection in this subsection.

To spot this connection, observe that circuit of Fig. 11 resembles that of the (inverse) quantum Fourier transform, which becomes especially apparent in the limit of a length-1 QSVT sequence. From a high level, this arises because Algorithm 3 may be viewed as a binary search through each bit of φ, where a bit of φ is determined at each iteration. Similarly, the (inverse) quantum Fourier transform may be viewed as a binary search through the bits defining an input state, where each qubit stores a bit of the input state at the end of the computation. Denoting the input state by $|x\rangle = |x_1 x_2 \cdots x_n\rangle$, where $x = 0.x_1 x_2 \cdots x_n$, recall that the action of the inverse quantum Fourier transform is to map

$$\frac{1}{2^{n/2}} \sum_{k=0}^{2^n-1} e^{2\pi i x k} |k\rangle = \frac{1}{2^{n/2}} \left(|0\rangle + e^{2\pi i 0.x_n}|1\rangle \right)$$
$$\otimes \left(|0\rangle + e^{2\pi i 0.x_{n-1}x_n}|1\rangle \right) \otimes \cdots \left(|0\rangle + e^{2\pi i 0.x_1 x_2 \cdots x_n}|1\rangle \right)$$
$$\mapsto |x_1 x_2 \cdots x_n\rangle = |x\rangle. \tag{75}$$

Similar to Algorithm 3, this mapping is achieved by applying Hadamards and controlled rotations to extract a bit of x from each qubit, leading to the correspondence seen here.

Let us explicitly show how the QFT naturally emerges from this construction. For the sake of pedagogical clarity, we switch to performing the QSVT by projecting into the $|0\rangle\langle 0|$ block of the QSVT sequence, as in the $(W_x, S_z, \langle 0| \cdot |0\rangle)$-QSP convention of Appendix A, which makes the connection to the QFT more apparent. This change is permissible because the threshold function, an even extension of $\Theta\left(1/\sqrt{2} - x\right)$, can be well approximated by a polynomial in the more restricted class of polynomials of this convention, as presented in Appendix D 7.

Under these conditions, let us make use of the block encoding of Eq. (68) (Fig. 10), where $\Pi = \tilde{\Pi} = |0\rangle\langle 0| \otimes I$. Then, a single instance of the signal-rotation operator (the block encoding) followed by the signal-processing rotation operator (the controlled phase shift), which is iterated in the QSVT, may be fully realized as in the circuit on the left side of Fig. 13. In this depiction, the top qubit is used to implement the projector-controlled phase shift as per Fig. 3 and the middle qubit is the block-encoding qubit used to access the encoding of $A_j(\theta)$ as per Fig. 10.

Evaluation of this circuit allows simplification to the right side of the figure, where we are now able to ignore the top qubit in favor of just the block-encoding qubit, an identity that follows from the simple choice of projector. A block-encoding qubit can also double as one of the ancilla qubits used to read out a bit of φ, as shown below. Let us refer to this type of qubit as a phase-readout qubit.

To make this identification explicit and draw a parallel with the QFT, let us also assume that φ is an $(m = 3)$-bit number. Then, employing a length-d QSVT sequence, the resulting phase estimation circuit may be schematically expressed as in Fig. 14. Here, each phase-readout qubit doubles as a block-encoding qubit [49], which is crucial to the emergence of the QFT, and is indeed a valid restructuring. In particular, instead of measuring each phase-readout qubit, updating θ, and then applying a controlled θ rotation at the next iteration, as in Algorithm 3, here we directly apply rotations controlled by the phase-readout qubit, making intermediate QSP-measurement steps unnecessary.

We implement this with the rotation operations R_ℓ defined in Eq. (73). Similar to Fig. 12, our construction employs controlled R_ℓ's to implement the following behavior: if a phase-readout qubit is $|0\rangle$, then it does not contribute to θ and no phase shift is applied; alternatively, if the phase-readout qubit is $|1\rangle$, then it contributes to θ and the corresponding phase shift is applied. Moreover, we arrange the R_ℓ's in a pattern that implicitly performs the $\theta \leftarrow \theta/2$ operation after each iteration; specifically, if ancilla qubit k controls an R_ℓ at one iteration, then at the next iteration ancilla qubit k controls an $R_{\ell+1}$. It may be easily verified that this procedure correctly encodes $A_j(\theta) = \frac{1}{2}(I + e^{-2\pi i \theta} U^{2^j})$ at each iteration and is entirely equivalent to the formalism of Algorithm 3, thus justifying the recycling of phase-readout qubits as block-encoding qubits.

Further, let the QSVT sequence length be simply $d = 1$, giving the quantum circuit depicted in Fig. 15. This allows us to further simplify the circuit, without changing any circuit elements, by observing that the controlled-U^{2^j} operations commute with the R_ℓ operations, which are z rotations. Slide the first three Hadamard gates over to the far left and gather all the remaining gates on the control qubits on the far right, dropping the signal-rotation
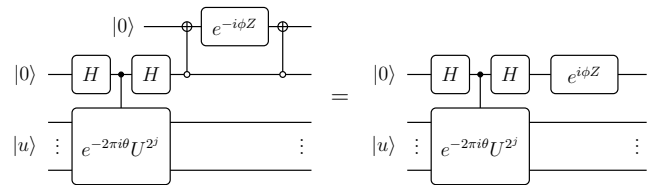

FIG. 13. The simplification of QSP phase control for the case when the block-encoded operator has an accessible control qubit.
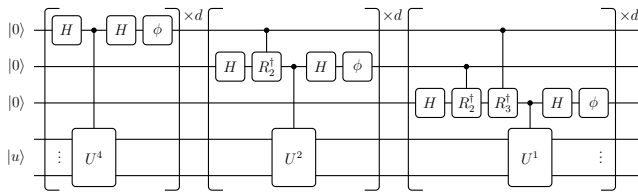
FIG. 14. An illustration of three-qubit quantum phase estimation using QSP.

FIG. 16. An illustration of three-qubit quantum phase estimation using QSP, in the simplified case when $d = 1$, with gates slid along wires to highlight the three-qubit inverse QFT circuit which emerges (gates in dotted rectangular box).

operations ϕ, which are inconsequential at this level of simplification.

This systematic simplification results in the quantum circuit shown in Fig. 16, where the inverse QFT circuit emerges naturally as the set of gates following the controlled-U^{2^j} gates. This is the standard quantum phase estimation circuit.

As this pattern persists to higher orders, we see that quantum phase estimation emerges from the cascaded QSVT (or QSP) employed to iteratively determine bits of the phase of an eigenphase. The QSVT construction of this transform allows for a richer variety of transformations; however, including a trade-off between the accuracy of each iterative step (by increasing d), the number of exponentiated powers U to employ, the number of qubits to use in the transform, and more.

VI. FUNCTION-EVALUATION PROBLEMS BY QSVT

Another useful application of the QSVT is to evaluate a function of a matrix, which we term *function-evaluation problems*. Schematically, if $f(x)$ is the function of interest, such that we wish to evaluate $f(A)$, then we could imagine solving this problem by employing the QSVT with a polynomial $P(x)$ that approximates $f(x)$. While these problems are generally more approachable with a quantum eigenvalue transform, they are still straightforward to solve with the QSVT.

Here, we summarize prominent function-evaluation problems, most notably Hamiltonian simulation and matrix inversion. Our discussion summarizes results from Ref. [12], wherein the full details of these procedures can be found.
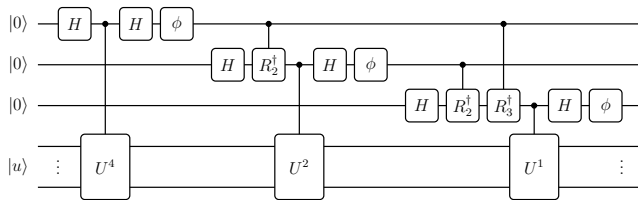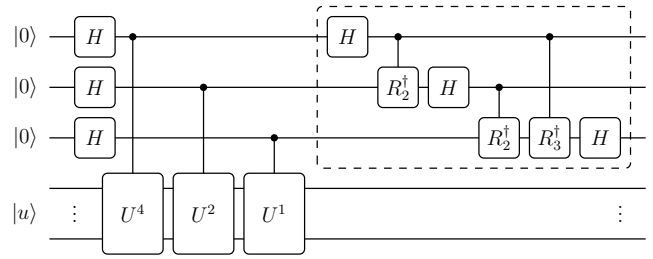

FIG. 15. An illustration of three-qubit quantum phase estimation using QSP, in the simplified case when $d = 1$.

A. Hamiltonian simulation by QSVT

A motivating goal of quantum computation is to simulate the time evolution of a state under a Hamiltonian, a problem known as Hamiltonian simulation. That is, for a Hamiltonian $\mathcal{H}$ and some time t, we would like to approximate the time evolution operator $e^{-i\mathcal{H}t}$, which is evidently a function-evaluation problem with the function $f(x) = e^{-ixt}$.

In the setup of this problem, we assume access to $\mathcal{H}$, of which we desire a unitary block encoding such that we may solve this problem with the QSVT. However, as we discuss in Sec. IV, such a unitary block encoding is only realizable if $\|\mathcal{H}\| \leq 1$. In general, then, we instead determine an $\alpha \geq \|\mathcal{H}\|$ and construct a unitary block encoding of $\mathcal{H}/\alpha$. Again, this requires some prior knowledge about $\mathcal{H}$, a drawback on which we elaborate in Sec. VII, but fortunately such a block encoding can be achieved for a large class of Hamiltonians [8,12].

With this rescaled block encoding, one can equivalently imagine that our goal is to simulate the time evolution of a system under the rescaled Hamiltonian H/α for a time $t\alpha$. This equivalence holds because the corresponding time-evolution operators are identical: $e^{-i(\mathcal{H}/\alpha)(\alpha t)} = e^{-i\mathcal{H}t}$.

How might this problem be solved with the QSVT? Naively, one may try to employ QSVT with a polynomial approximation to e^{-ixt} (here, we view t as a parameter, not a variable). However, because the exponential function does not have definite parity, this function does not satisfy the constraints on Poly(a)= $\langle + |U_{\vec{\phi}}| + \rangle$ discussed in Sec. II A and Appendix A). To circumvent this issue, one can instead apply the QSVT twice—once with an even polynomial approximation to $\cos(xt)$ and once with an odd polynomial approximation to $\sin(xt)$, both of which have definite parities. Then, using the circuit illustrated in Fig. 17, one can sum together the results of these two QSVT executions to obtain $\cos^{(\mathrm{SV})}(\mathcal{H}t) - i\sin^{(\mathrm{SV})}(\mathcal{H}t) = e^{-i\mathcal{H}t}$, as desired.

However, note that the above relation only holds if the eigenvalues of $\mathcal{H}$ are positive, such that the singular values are equal to the eigenvalues. As we discuss in Sec. IV, if this is not the case, we may instead use the

block encoding of $\mathcal{H}/\alpha$ and a circuit analogous to Fig. 10 to construct a block encoding of the positive definite matrix $\frac{1}{2}\left(\frac{\mathcal{H}}{\alpha}+I\right)$. Applying the aforementioned QSVT procedure to this matrix for a time $2\alpha t$ produces a block encoding of $e^{-i\mathcal{H}t}$ up to a global phase. In the remainder of this section, we assume that this issue is alleviated.

Returning to the QSVT scheme, we note that in Ref. [12], Gilyén *et al.* approximate the functions $\cos(xt)$ and $\sin(xt)$ by polynomials using the Jacobi-Anger expansion:

$$\cos(xt) = J_0(t) + 2\sum_{k=1}^{\infty}(-1)^k J_{2k}(t) T_{2k}(x) \quad (76)$$

$$\sin(xt) = 2\sum_{k=0}^{\infty}(-1)^k J_{2k+1}(t) T_{2k+1}(x) \quad (77)$$

where $J_i(x)$ is a Bessel function of order i and $T_i(x)$ is a Chebyshev polynomial of order i. One can attain ϵ approximations to $\cos(xt)$ and $\sin(xt)$ by truncating these expressions at a sufficiently large index k'. The necessary truncation index k' may be determined by a function $r(t, \epsilon)$, which is defined implicitly as

$$\epsilon = \left(\frac{|t|}{r(t,\epsilon)}\right)^{r(t,\epsilon)} \quad \text{such that } r(t,\epsilon) \in (t,\infty) \quad (78)$$

and scales asymptotically as

$$r(t,\epsilon) = \Theta\left(|t| + \frac{\log(1/\epsilon)}{\log\left(e + \frac{\log(1/\epsilon)}{|t|}\right)}\right). \quad (79)$$

In particular, truncating Eqs. (76) and (77) at $k' = \left\lfloor \frac{1}{2}r\left(e/2|t|, 5/4\epsilon\right) \right\rfloor$ yields ϵ approximations to $\cos(xt)$ and $\sin(xt)$, respectively, where $0 < \epsilon < 1/e$. Because $T_i(x)$ is a polynomial of degree i with definite parity, these
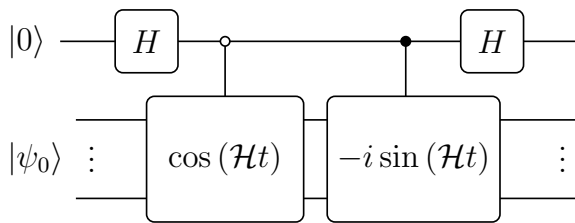

FIG. 17. One quantum circuit that can be used to construct the time evolution operator $\cos(\mathcal{H}t) - i\sin(\mathcal{H}t) = e^{-i\mathcal{H}t}$ and apply it to $|\psi_0\rangle$, for use in Hamiltonian simulation as described in Algorithm 4. Note that the correct evolution of the input state $|\psi_0\rangle$ is achieved only upon postselection of the auxiliary qubit in the $|0\rangle$ state. One can achieve this with either repetition or fixed-point amplitude amplification, similar to projecting into the desired block of the QSVT sequence operator as discussed at the end of Sec. II D.

approximations are polynomials of degree $2k'$ and $2k' + 1$, respectively, with the correct even or odd parity. Let us denote these polynomials by $P_\epsilon^{\cos}(x;t)$ and $P_\epsilon^{\sin}(x;t)$.

Lastly, because cosine and sine are bounded in magnitude by 1, these ϵ approximations only obey $|P_\epsilon^{\cos}(x;t)|, |P_\epsilon^{\sin}(x;t)| \leq 1+\epsilon$. However, we need the target polynomials to necessarily be bounded in magnitude by 1 in order to be implemented through the QSVT. As in Ref. [23], we can fix this by rescaling the polynomials by a factor of $1/1+\epsilon$, at the expense of increasing the error of these approximations to 2ϵ. This can be seen with the triangle inequality as $\left|(1/1+\epsilon)P_\epsilon^{\cos}(x;t) - \cos(xt)\right| \leq 1/1 + \epsilon\left[\left|P_\epsilon^{\cos}(x;t) - \cos(xt)\right| + |\epsilon\cos(xt)|\right] \leq 1/1+\epsilon$ $(\epsilon + \epsilon) \leq 2\epsilon$ and similarly for $P_\epsilon^{\sin}(x;t)$.

To determine the complexity of this Hamiltonian simulation algorithm, first recall that our effective goal is to simulate the rescaled Hamiltonian $\mathcal{H}/\alpha$ for time αt. In addition, note that the truncations of the Jacobi-Anger expansion used in this procedure should be $\epsilon/4$ approximate such that, when rescaled by $1/1 + \epsilon/4$, they are $\epsilon/2$ approximations to $\cos(xt)$ and $\sin(xt)$. With this choice, the sum of these approximations, which is the approximation to e^{-ixt}, is ϵ approximate by the triangle inequality. Incorporating these conditions, we see that this QSVT-based Hamiltonian simulation algorithm prepares an ϵ-approximate block encoding of $e^{-i\mathcal{H}t}$ and queries U a total number of times:

$$2k' + 2k' + 1 = 4 \cdot \left\lfloor \frac{1}{2}r\left(\frac{e}{2}\alpha|t|, \frac{5}{4}\frac{\epsilon}{4}\right)\right\rfloor + 1$$

$$= \Theta\left(\alpha|t| + \frac{\log(1/\epsilon)}{\log\left(e + \frac{\log(1/\epsilon)}{\alpha|t|}\right)}\right). \quad (80)$$

In comparing this query complexity with results quoted in the literature, α may be replaced with $\|\mathcal{H}\|$.

This complexity has state-of-the-art scaling in t and ϵ for Hamiltonian simulation: it is linear in t, logarithmic in ϵ, and additive in these two terms. As such, it provides a significant improvement over other algorithms [8,12,23]. We summarize Hamiltonian simulation by the QSVT in Algorithm 4.

B. Matrix inversion by QSVT

Another straightforward, yet widely applicable, function-evaluation problem is that of *matrix inversion*. That is, given access to a square matrix A, one wishes to construct an approximation to A^{-1}. Harrow *et al.* have presented a quantum algorithm for this problem in the case that A is Hermitian [15]. In their eponymous algorithm, they prepare the state $A^{-1}|b\rangle$, which provides a quantum solution to the linear system $A|x\rangle = |b\rangle$.

Let us now look at this problem through the lens of the QSVT. Suppose that we have an $N \times N$ matrix A with

---

Input: Access to a Hamiltonian $\mathcal{H}$, a desired time t, an error tolerance ϵ, and an $\alpha \geq \|\mathcal{H}\|$.

Output: A block encoded ϵ-approximation of $e^{-i\mathcal{H}t}$.

Runtime $: \Theta\left(\alpha|t| + \frac{\log(1/\epsilon)}{\log\left(e + \frac{\log(1/\epsilon)}{\alpha|t|}\right)}\right)$ queries to (the encoding of) $\mathcal{H}/\alpha$.

Procedure:

1 Prepare a unitary block encoding of $\mathcal{H}/\alpha$.

2 Apply QSVT to this encoding twice, using the polynomials $\frac{1}{1+\epsilon/4}P^{\cos}_{\epsilon/4}(x;t)$ and $\frac{1}{1+\epsilon/4}P^{\sin}_{\epsilon/4}(x;t)$, where $P^{\cos}_{\epsilon/4}(x;t)$ and $P^{\sin}_{\epsilon/4}(x;t)$ are obtained by truncating the series in Eqs. (76) and (77), respectively, at index $k' = \left\lfloor \frac{1}{2}r\left(\frac{e}{2}\alpha|t|, \frac{5}{4}\frac{\epsilon}{4}\right)\right\rfloor$.

3 With the results of the above QSVT executions, which approximate $\cos^{(\mathrm{SV})}(\mathcal{H}t)$ and $\sin^{(\mathrm{SV})}(\mathcal{H}t)$, respectively, run the circuit in Figure 17.

Algorithm 4. Hamiltonian Simulation by QSVT.

singular value decomposition $A = W_\Sigma \Sigma V_\Sigma^\dagger$, where Σ contains the singular values along its diagonal. As per the setup of the HHL algorithm, we also assume that the singular values of A obey $\sigma_i \in [1/\kappa, 1]$ for some finite condition number $\kappa \geq 1$ (if not, A may be rescaled to obey this condition). As the singular values are nonzero, the inverse of A exists and may be obtained as $A^{-1} = V_\Sigma \Sigma^{-1} W_\Sigma^\dagger$, where Σ^{-1} contains the reciprocals of the singular values along its diagonal. Because $A^\dagger = V_\Sigma \Sigma W_\Sigma^\dagger$, this can be reexpressed as $A^{-1} = f^{(\mathrm{SV})}(A^\dagger)$, where $f(x) = 1/x$, indicating that matrix inversion is a function-evaluation problem with $f(x) = 1/x$.

Upon this realization, it is clear how to apply the QSVT to matrix inversion: find an odd polynomial $P(x)$ that approximates $f(x) = 1/x$ over the range of singular values of A and employ the QSVT to construct $P^{(\mathrm{SV})}(A^\dagger)$, which approximates A^{-1}. Finding a good polynomial $P(x)$ is tricky because of the discontinuity in $1/x$, but it can indeed be done. In addition, this procedure requires that one can construct a unitary block encoding of A, which is feasible because $\|A\| \leq 1$ as per the assumption that $\sigma_i \leq 1$. Such a block encoding can indeed be achieved for a variety of matrices relevant to physics [8,12] (but again, we discuss some caveats of doing so in Sec. VII).

Moreover, because we require that the polynomial $P(x)$ be bounded in magnitude by 1 such that it can be implemented through the QSVT, we cannot necessarily use $P(x) \approx 1/x$ as our target function, since $1/\sigma_i \geq 1$ in general. To overcome this challenge, let us instead seek a polynomial approximation to a function that behaves as $1/2\kappa 1/x$ over the range $[-1, 1] \setminus [-1/\kappa, 1/\kappa]$, which

will invert each singular value and is bounded in magnitude by $\frac{1}{2}$ over this range (we use the multiplicative factor $1/2\kappa$ instead of $1/\kappa$ to avoid the need to rescale by $1/1 + \epsilon$ when we ϵ approximate this function, which is done in Sec. VI A). This procedure will output an approximation of $(1/2\kappa)A^{-1}$ and, because κ is *a priori* known, this multiplicative factor is not prohibitive to calculations. However, due to this multiplicative factor, we now desire a $\epsilon/2\kappa$ approximation to $(1/2\kappa)A^{-1}$, from which we effectively attain an ϵ approximation to A^{-1}.

The appropriate polynomial for matrix inversion is thus an $\epsilon/2\kappa$ approximation to $1/2\kappa 1/x$. In Appendix C, we demonstrate how to construct such a polynomial. While the construction is a bit involved, in essence it is a product of a polynomial approximation to $1/x$ over a restricted range (constructed as a sum of Chebyshev polynomials) and a polynomial approximation to a rectangular function [constructed as a sum of the sign function approximations, $P^\Theta_{\epsilon,\Delta}(x)$].

We term this polynomial the *matrix-inversion polynomial*, denoted by $P^{\mathrm{MI}}_{\epsilon,\kappa}(x)$, and defer its rigorous definition to Appendix C. In addition to being an $\epsilon/2\kappa$ approximation to $(1/2\kappa)1/x$, $P^{\mathrm{MI}}_{\epsilon,\kappa}(x)$ has odd parity and is bounded in magnitude by 1 for $x \in [-1, 1]$. Hence, the matrix-inversion polynomial may be implemented through the QSVT.

Moreover, we also show in Appendix C that $P^{\mathrm{MI}}_{\epsilon,\kappa}(x)$ has degree

$$d = \mathcal{O}[\kappa \log(\kappa/\epsilon)]. \quad (81)$$

As the QSVT requires $\mathcal{O}(d)$ calls to the block encoding, we see that matrix inversion by QSVT has complexity $\mathcal{O}[\kappa \log(\kappa/\epsilon)]$. This is an improvement over the conventional HHL algorithm, which has run time $\mathcal{O}[\kappa^2 \log(N)/\epsilon]$. It is quite impressive that the QSVT algorithm provides an large improvement in the scaling with κ/ϵ, although similar results have been achieved with non-QSVT methods [50]. In addition, the HHL algorithm uses a sparse Hamiltonian simulation subroutine with target Hamiltonian A, resulting in the $\log(N)$ term in its complexity, whereas the QSVT algorithm does not use such a subroutine and thus N dependence is absent from its complexity (however, constructing the necessary block encoding of A may scale with N). We summarize matrix inversion by QSVT in Algorithm 5.

Moreover, like the HHL algorithm, matrix inversion by QSVT may be used to solve the linear system of equations $A|x\rangle = |b\rangle$, by applying the block encoding of $(1/2\kappa)A^{-1}$ to $|b\rangle$, which yields an ϵ approximation to $A^{-1}|b\rangle$ upon rescaling by 2κ. As discussed at the end of Sec. II D, this procedure requires that we project into the desired block of the QSVT sequence operator, which may be performed with little overhead.

Lastly, we note that this result can be further extended. With some minor adjustments, this QSVT-based algorithm can be adapted to prepare the pseudoinverse of a rectangular matrix, which is useful in various machine learning contexts [12].

VII. DISCUSSION

In this tutorial, we present how the quantum singular value transformation encapsulates the three major quantum algorithms. Paralleling Ref. [12], we construct QSVT-based algorithms for search, Hamiltonian simulation, and the quantum linear systems problem. Toward this end, we also derive QSVT-based algorithms for the eigenvalue threshold problem and phase estimation. Moreover, the utility of the QSVT is not entirely enumerated here—further applications of the QSVT to the quantum OR lemma, quantum machine learning, quantum walks, fractional query implementation, and Gibbs state preparation appear in the literature [12].

It is insightful that the QSVT encompasses such a broad spectrum of problems. Effectively, the QSVT provides a series of dials (i.e., a well-defined parametrization) that can be turned to transform from one algorithm to another. In addition, when there is sufficient structure inherent to the problem of interest, the resulting algorithm often becomes more efficient. Consequently, the QSVT provides one lens through which to analyze the source of quantum advantage and make concrete the somewhat vague trade-off conjectured between problem structure and quantum algorithmic efficiency (while maintaining a significant gap between optimal classical and quantum performance). In aggregate, these constructions suggest that it is wise to continue to search for new quantum algorithms in the setting of the QSVT.

There is ample room for future research in this area. Notably, various quantum algorithms have not yet been constructed from QSVT-based subroutines, such as variational algorithms like the variational quantum eigensolver [51] or the quantum approximate optimization algorithm [52]. It would be fascinating to see if the QSVT can encompass, or perhaps even enhance, these hybrid quantum algorithms as well. This work also begets the question of how else one might tweak the parameters of the QSVT to create novel algorithms or extend previously known algorithms to novel noise settings. As the QSVT is intuitive and flexible, there is likely a large class of problems that are amenable to analysis by QSVT and that admit a significant quantum advantage.

Moreover, note that there is a significant caveat in the use of the QSVT, arising from the requirement of block encodings. In a typical implementation of the QSVT on a quantum computer, we may imagine that the matrix to which we would like to apply a transform—say, A—is provided in a quantum random access memory (QRAM), from which we may straightforwardly construct a block encoding of $A/\|A\|_F$ [12]. One could then apply the QSVT to this encoding with a run time linear in $\|A\|_F$, similar to how the run times of the eigenvalue threshold algorithm and the Hamiltonian simulation algorithm of this tutorial are linear in α. However, assuming that one has a classical computer with sampling and query access to A, as a classical analog of having A loaded into QRAM, then one can acquire access to a singular value transformation of A by executing a classical (not quantum) algorithm [53]; impressively, this classical algorithm has a run-time polynomial in $\|A\|_F$. This polynomial run time suggests that QRAM-based QSVT cannot always achieve an exponential speed-up over classical algorithms but can still attain a significant polynomial speed-up. Although this challenge could be an Achilles' heel for application of the QSVT to general and unstructured problems, clearly the QSVT is still of interest for speeding up solution of problems with structure, as illustrated by quantum factoring. Also, while every QRAM essentially provides a block-encoded matrix, there are ways to block encode matrices that do not require a QRAM; this is good, given the fact that QRAM implementations generally face a number of practical challenges in their realization, e.g., requiring a number of ancillary qubits that grows linearly with the number of items stored (see, e.g., Ref. [54] and references therein).

It is also interesting to note that while physics has developed significant insight into the role of eigenvalues, appreciation of singular values has lagged. For example, eigenvalues are the bread and butter of quantum systems, as energies for eigenstates of the Hamiltonian and as stability points for stochastic systems. In contrast, singular values apparently play few starring roles in physics. One of the few examples arises in the study of entanglement, where the singular value decomposition is the underlying construct behind the Schmidt decomposition of a bipartite quantum state.

Why are there so few prominent roles for singular values in physics? Maybe it is because physics is drawn to closed

---

Input: Access to A, an error tolerance ϵ, and a condition number $\kappa \geq 1/(\min_i \sigma_i)$

Output: A block encoded $\frac{\epsilon}{2\kappa}$-approximation of $\frac{\kappa}{2} A^{-1}$, which is effectively equivalent to an ϵ-approximation to A^{-1}.

Runtime : $\mathcal{O}\left(\kappa \log\left(\frac{\kappa}{\epsilon}\right)\right)$ queries to (a block encoding of) $A^{\dagger}$.

Procedure:

1 Prepare a unitary block encoding of $A^{\dagger}$.

2 Apply QSVT to this block encoding to compute $(P_{\epsilon,\kappa}^{\mathrm{MI}})^{(\mathrm{SV})}(A^{\dagger})$, where the polynomial $P_{\epsilon,\kappa}^{\mathrm{MI}}(x)$ is defined in Eq. (C6) of Appendix C.

Algorithm 5.   Matrix Inversion by QSVT.

Hamiltonian systems (think: square matrices), whereas singular value decompositions arise mostly in studies of subsystems (think: nonsquare matrices), where the input and output dimensions may be different. As discussed in Sec. I, such disparate dimensions also arise naturally in computation. And, indeed, singular value decompositions play a prominent role in modern computation, especially in machine learning, where they are the basis for principal component analysis, model reduction, collaborative filtering, and more. As quantum information and computer science continue to grow into a unified field, perhaps it is not surprising that singular value decompositions—and singular value transformations—are emerging as a unifying bridge.

APPENDIX A: QSP CONVENTIONS

A quantum signal-processing construction may be entirely determined by four constituents:

(1) The *signal operator* W (sometimes also called the signal unitary)
(2) The *phase angles* $\vec{\phi} = (\phi_0, \phi_1, \ldots, \phi_d)$
(3) The *signal processing* operator $S(\phi)$, constructing using $\phi \in \vec{\phi}$
(4) The *signal basis* M, in which the desired polynomial is obtained

In its most basic form, QSP is performed by interleaving W with $S(\phi)$ operations followed by a projective measurement in the basis M.

The signal operator W is signal dependent and constant throughout the sequence; the signal-processing operator S is parametrized by a sequence of phases $\vec{\phi} \in \mathbb{R}^{d+1}$, which are chosen based on the desired output function. The exact form of W and S, along with the choice of measurement basis M, determine the family of achievable output functions. More specifically, M may be not a basis for a complete vector space but just the basis for a subspace identifying a specific desired polynomial output function.

For example, we may specify $M = \langle +| \cdot |+\rangle$, when the measurement is to select the $|+\rangle$ outcome, with the QSP sequence starting off with the control qubit in the $|+\rangle$ state. Thus, we may refer to any particular QSP construction as being a (W, S, M)-QSP convention.

In this appendix, we discuss common (W, S, M)-QSP conventions in the literature for W, S, and M, and present relationships between the QSP phase angles $\vec{\phi}$. Specifically, we elaborate on the "Wx," reflection, and "Wz" conventions.

1. Wx convention for QSP

One common convention is to choose the signal operator W to be an x rotation in the Bloch sphere,

$$W_x(\theta) := e^{i\frac{\theta}{2}X} = \begin{bmatrix} a & i\sqrt{1-a^2} \\ i\sqrt{1-a^2} & a \end{bmatrix}, \quad (A1)$$

where, compared with Eq. 1, we introduce the additional x subscript for clarity. Under this convention, one also chooses the signal-processing rotation to be a z rotation, $S_z(\phi) = e^{i\phi Z}$. Theorem 1 describes the family of unitaries that are achievable in this convention.

Typically, we are not concerned with the achievable unitaries but, rather, with the achievable functions that can be computed in a subsystem. If we choose to project out the $\langle 0| \cdot |0\rangle$ element, the output is $\text{Poly}(a) = \langle 0|U_{\vec{\phi}}|0\rangle = P(a)$. Here, the choice of Poly is restricted not only by the conditions on P of Theorem 1 but also the additional constraints below, which ensure that a polynomial Q exists satisfying the conditions of Theorem 1.

Theorem 9: $((W_x, S_z, \langle 0| \cdot |0\rangle)\text{-QSP})$. *A QSP phase sequence $\vec{\phi} \in \mathbb{R}^{d+1}$ exists,*

$$\text{Poly}(a) = \langle 0|e^{i\phi_0 Z} \prod_{k=1}^{d} W_x(a)e^{i\phi_k Z}|0\rangle, \quad (A2)$$

for $a \in [-1, 1]$, and for any polynomial $\text{Poly} \in \mathbb{C}[a]$ if and only if the following conditions hold:

(i) $\deg(\text{Poly}) \leq d$
(ii) *Poly has parity d mod 2*
(iii) $\forall a \in [-1, 1]$, $|\text{Poly}(a)| \leq 1$
(iv) $\forall a \in (-\infty, -1] \cup [1, \infty)$, $|\text{Poly}(a)| \geq 1$
(v) *if d is even, then $\forall a \in \mathbb{R}$, $\text{Poly}(ia)\text{Poly}^*(ia) \geq 1$*

The family of achievable polynomials in this case is significantly limited, since the projective measurement is performed in the same basis $M = \{|0\rangle, |1\rangle\}$ as the signal-processing operations. For example, this immediately limits us to polynomial functions such that $|\text{Poly}(\pm 1)| = 1$.

Properties (iv) and (v) are quite restrictive, especially when approximating real functions. In fact, none of polynomial approximations described in this tutorial satisfy these last two constraints. Allowing a small but nonzero imaginary part, it is possible to find QSP phases in the $(W_x, S_z, \langle 0| \cdot |0\rangle)$ convention approximating real functions that satisfy $|f(\pm 1)| = 1$; an example of this is the phase estimation function plotted in Sec. D 7.

Since we are often interested in constructing real polynomials, choosing a different signal basis ends up being much more useful. In particular, when $M = \{|+\rangle, |-\rangle\}$, then we may employ this theorem for real polynomials:

Theorem 10: $[(W_x, S_z, \langle +| \cdot |+\rangle)$-QSP]. A QSP phase sequence $\vec{\phi} \in \mathbb{R}^{d+1}$ exists such that

$$Poly(a) = \langle +|e^{i\phi_0 Z} \prod_{k=1}^{d} W_x(a)e^{i\phi_k Z}|+\rangle \quad \text{(A3)}$$

for $a \in [-1, 1]$, and for any real polynomial $Poly \in \mathbb{R}[a]$ if and only if the following conditions hold:

   (i) $deg(Poly) \leq d$
   (ii) $Poly$ has parity $d \bmod 2$
   (iii) $\forall a \in [-1, 1], |Poly(a)| \leq 1$

This QSP convention is expressive enough for all of the polynomials considered in this tutorial and is used in Refs. [7,12]. The proofs of Theorems 9 and 10 are given in Ref. [12].

2. Reflection convention for QSP

Another common convention is to choose the signal operator W to be a reflection [as in Eq. (12)]:

$$R(a) := \begin{bmatrix} a & \sqrt{1-a^2} \\ \sqrt{1-a^2} & -a \end{bmatrix}. \quad \text{(A4)}$$

The reflection operator is preferred in some cases, as it has the added benefit of being Hermitian, which can simplify proof constructions and, in particular, equations such as Eq. (34). Given a phase sequence $\vec{\phi} \in \mathbb{R}^{d+1}$, we can find a $\vec{\phi}' \in \mathbb{R}^{d+1}$ such that

$$e^{i\phi_0 Z} \prod_{k=1}^{d} R(a)e^{i\phi_k} = e^{i\phi_0' Z} \prod_{k=1}^{d} W_x(a)e^{i\phi_k'}. \quad \text{(A5)}$$

Using the relationship of Eq. (14), this can be accomplished by choosing $\phi_0 = \phi_0' + (2d-1)\pi/4$, $\phi_d = \phi_d' - \pi/4$, and $\phi_k = \phi_k' - \pi/2$ for $k \in 1, \ldots, d-1$. Therefore, these two conventions are equivalent regardless of the final measurement basis.

3. Wz convention for QSP

Theorem 10 can also be understood through its relationship to the convention used in Ref. [34]. Here, the authors define QSP with the signal operator W being a z rotation,

$$W_z(\theta) = e^{i\frac{\theta}{2}Z} = \begin{bmatrix} w & 0 \\ 0 & w^{-1} \end{bmatrix}, \quad \text{(A6)}$$

where $w := e^{i\theta/2}$, and the signal-processing operator is an x rotation, $S_x(\phi) = e^{i\phi X}$. Furthermore, in this convention, it is typical to choose the signal basis as being $M = \{|0\rangle, |1\rangle\}$.

In our notation, this convention is written as $(W_z, S_x, \langle 0| \cdot |0\rangle)$-QSP. This convention is equivalent to $(W_x, S_z, \langle +| \cdot |+\rangle)$-QSP and is equally expressive, which can easily be seen since

$$\langle +|e^{i\phi_0 Z} \prod_{k=1}^{d} W_x(\theta)e^{i\phi_k Z}|+\rangle = \langle 0|e^{i\phi_0 X} \prod_{k=1}^{d} W_z(\theta)e^{i\phi_k X}|0\rangle. \quad \text{(A7)}$$

The Laurent polynomial formulation of $(W_z, S_x, \langle 0| \cdot |0\rangle)$-QSP of Ref. [34] can be related to our formulation as follows:

$$e^{i\phi_0 X} \prod_{k=1}^{d} W_z(\theta)e^{i\phi_k X} \quad \text{(A8)}$$

$$=: \begin{bmatrix} F(w) & iG(w) \\ iG(w^{-1}) & F(w^{-1}) \end{bmatrix} \quad \text{(A9)}$$

$$= He^{i\phi_0 Z} \prod_{k=1}^{d} W_x(\theta)e^{i\phi_k Z}H \quad \text{(A10)}$$

$$= H \begin{bmatrix} P(a) & iQ(a)\sqrt{1-a^2} \\ iQ^*(a)\sqrt{1-a^2} & P^*(a) \end{bmatrix} H, \quad \text{(A11)}$$

for complex polynomials $P, Q \in \mathbb{C}[a]$ and real Laurent polynomials $F, G \in \mathbb{R}[w, w^{-1}]$ with parity $d \bmod 2$.

Explicitly,

$$f_0 = \text{Re}[p_0], \quad g_0 = \text{Im}[p_0] \quad \text{(A12)}$$

and, for $k > 0$,

$$f_k = \frac{1}{2}\text{Re}[p_k + q_k], \quad f_{-k} = \frac{1}{2}\text{Re}[p_k - q_k], \quad \text{(A13)}$$

$$g_k = \frac{1}{2}\text{Im}[p_k - q_k], \quad g_{-k} = \frac{1}{2}\text{Im}[p_k + q_k], \quad \text{(A14)}$$

where the coefficients for P and Q are given in the Chebyshev bases

$$P(a) \equiv \sum_{k=0}^{d} p_k T_k(a), \qquad (A15)$$

$$Q(a) \equiv \sum_{k=0}^{d} q_k U_{k-1}(a) \qquad (A16)$$

and the coefficients for the F and G are given in the standard basis

$$F(w) \equiv \sum_{k=-d}^{d} f_k w^k, \qquad G(w) \equiv \sum_{k=-d}^{d} g_k w^k. \qquad (A17)$$

The requirement for unitarity in Eq. (A9) is equivalent to

$$F(w)F(w^{-1}) + G(w)G(w^{-1}) = 1. \qquad (A18)$$

A numerically stable method for computing phases for a given $F(w)$ in the $(W_z, S_x, \langle 0| \cdot |0\rangle)$ convention is discussed further in Ref. [34] and can also be seen as a constructive proof of Theorem 10.

APPENDIX B: PROOFS ABOUT PHASE ESTIMATION BY QSVT

In this appendix, we prove the theorems used in the development of phase estimation by QSVT in Sec. V.

1. Theorems 6 and 7

Here, we prove Theorems 6 and 7 from Sec. V A 1:

Theorem 6: *If $n \geq m$, and one has the ability to implement the sign function exactly through the QSVT, then at the end of the algorithm sketch of Sec. 1, $\theta = \varphi$.*

and

Theorem 7: *If $n < m$ (including the case $m = \infty$), and one has the ability to implement the sign function exactly through the QSVT, then at the end of the algorithm sketch of Sec. V A 1 (with a minor modification discussed below), $|\theta - \varphi| \leq 2^{-n-1}$.*

The minor modification needed for $n < m$ is an additional $j = 0$ iteration used in the complete phase estimation-by-QSVT algorithm.

In the setup of these proofs, we suppose that φ is an m-bit number and use the notation $0.\varphi_{[j:]} := 0.\varphi_j \varphi_{j+1} \cdots \varphi_m$ to denote a contiguous string of binary digits (this string being infinite in the case $m = \infty$). We also assume that we can implement the sign function exactly in the QSVT (which is unrealistic, but is addressed in Sec. V A 2). Hence, the procedure that we discuss in these proofs is effectively Algorithm 3 modulo the error analysis.

a. $n \geq m$

If $n \geq m$, then append φ with $m - n$ 0's after its mth binary decimal, such that $\varphi = 0.\varphi_1 \varphi_2 \cdots \varphi_m 00 \cdots 0$. This effectively makes φ an n-bit number without changing its numerical value. We now invoke the following lemma, which is proven by induction:

Lemma 11: *If $\varphi = 0.\varphi_1 \varphi_2 \cdots \varphi_n$ is an n-bit number, then at the end of the iteration $j \geq 0$, $\theta = 0.\varphi_{j+1} \varphi_{j+2} \cdots \varphi_n = 0.\varphi_{[j+1:]}$.*

Proof. The proof proceeds by induction.
Base Case: The base case is the $j = n - 1$ iteration, at which $\theta = 0$. Using Eq. (65), we see that $A_j(\theta) = A_{n-1}(0)$ has singular value

$$\sigma^{n-1} = \big| \cos \big[\pi (0.\varphi_n - 0) \big] \big| = \begin{cases} 1, & \text{if } \varphi_n = 0, \\ 0, & \text{if } \varphi_n = 1, \end{cases} \qquad (B1)$$

and so $\Theta\left(1/\sqrt{2} - \sigma^{n-1}\right) = 1 - 2\varphi_n$. Therefore, after we apply $\Theta^{(SV)}\left[1/\sqrt{2} - A_{n-1}(0)\right]$ to $|u\rangle$, controlled on the ancilla qubit, the state of the ancilla qubit is $1/\sqrt{2}\Big[|0\rangle + (1 - 2\varphi_n)|1\rangle\Big]$. The final Hadamard gate brings the ancilla qubit to the state

$$(1 - \varphi_n)|0\rangle + \varphi_n|1\rangle = \begin{cases} |0\rangle, & \text{if } \varphi_n = 0, \\ |1\rangle, & \text{if } \varphi_n = 1, \end{cases} \qquad (B2)$$

and therefore the measurement of the ancilla qubit will yield φ_n. The final step of the iteration sets $\theta_1 = \varphi_n$, such that $\theta = 0.\varphi_n$, as claimed.
General Case: Proceeding with induction, assume that this claim is true at the end of iteration $j + 1$. That is, $\theta = 0.\varphi_{j+2} \varphi_{j+3} \cdots \varphi_n = 0.\varphi_{[j+2:]}$ at the end of iteration $j + 1$. Then, at the start of iteration j, $\theta \leftarrow \theta/2 = 0.0\varphi_{j+2}\varphi_{j+3} \cdots \varphi_n$. Again using Eq. (65), we see that $A_j(\theta)$ has singular value

$$\sigma^j = \big| \cos \big[\pi(0.\varphi_{j+1}\varphi_{j+2} \cdots \varphi_n - 0.0\varphi_{j+2}\varphi_{j+3} \cdots \varphi_n) \big] \big|$$

$$= \Big| \cos \Big(\frac{\pi}{2}\varphi_{j+1}\Big) \Big| = \begin{cases} 1, & \text{if } \varphi_{j+1} = 0, \\ 0, & \text{if } \varphi_{j+1} = 1, \end{cases} \qquad (B3)$$

and so $\Theta\left(1/\sqrt{2} - \sigma^j\right) = 1 - 2\varphi_{j+1}$. This is identical to the base case and we ultimately find that the measurement of the auxiliary qubit yields φ_{j+1}, such that we set $\theta_1 = \varphi_{j+1}$. So at the end of this iteration, we have $\theta = 0.\varphi_{j+1}\varphi_{j+2} \cdots \varphi_n = \varphi_{[j+1:]}$, as desired. ∎

Considering the $j = 0$ iteration, we see that the final output of the algorithm is $\theta = 0.\varphi_1\varphi_2 \cdots \varphi_n = \varphi$, which completes the proof of Theorem 6.

b. $n < m$

If $n < m$, this algorithm produces an n-bit approximation to φ that suffers error at most $1/2^{n+1}$. To show this, we first prove the following lemma.

Lemma 12: *If φ is an $m > n$ bit number (including the case $m = \infty$), then at the end of iteration $j \geq 0$, $\theta = 0.\tilde{\varphi}_{j+1}\tilde{\varphi}_{j+2}\cdots\tilde{\varphi}_n = 0.\tilde{\varphi}_{[j+1:]}$, such that $|\tilde{r}_j.\tilde{\varphi}_{[j+1:]} - 0.\varphi_{[j+1:]}| \leq 2^{j-n-1}$, where $\tilde{r}_j.\tilde{\varphi}_{[j+1:]}$ is attained by rounding $0.\varphi_{[j+1:]}$ to $n-j$ binary decimals and $\tilde{r}^j$ is an additional bit carried over by this rounding.*

Proof. As before, the proof proceeds by induction, albeit longer than the $n \geq m$ proof but no more complicated. One minor difference is that we still write expressions such as $\varphi = 0.\varphi_1\varphi_2\cdots\varphi_m$ to indicate the binary expansion of φ, where it is to be understood that this will really be an infinite sequence in the case $m = \infty$.

Base Case: The base case is the $j = n - 1$ iteration, at which, $\theta = 0$. Using Eq. (65), we see that $A_j(\theta) = A_{n-1}(0)$ has singular value

$$
\begin{aligned}
\sigma^{n-1} &= \left| \cos\left[\pi(0.\varphi_n\varphi_{n+1}\cdots\varphi_m - 0)\right]\right| \\
&= \left| \cos\left[\frac{\pi}{2}\left(\varphi_n + \frac{1}{2}\varphi_{n+1} + \sum_{i=2}^{m-n}\frac{1}{2^i}\varphi_{n+i}\right)\right]\right|. \quad \text{(B4)}
\end{aligned}
$$

We are not concerned with this exact value, but instead with the quantity $\Theta(1/\sqrt{2} - \sigma^{n-1})$, which is dictated by the values of φ_n and φ_{n+1}. In particular, the sum in Eq. (B4) is bounded above by $\frac{1}{2}$, so it cannot affect $\Theta(1/\sqrt{2} - \sigma^{n-1})$ by itself. Let us look at each possible value of φ_n and φ_{n+1}.

First, if $\varphi_{n+1} = 0$ (regardless of the value of φ_n), then this scenario is similar to the proof of Theorem 6 and we ultimately set $\theta_1 = \varphi_n$. In addition, rounding $0.\varphi_n\varphi_{n+1}\cdots\varphi_m = 0.\varphi_n 0\varphi_{n+2}\cdots\varphi_m$ to $n - j = 1$ decimal places yields $\tilde{r}_{n-1}.\tilde{\varphi}_n = 0.\varphi_n$, so indeed $\theta = 0.\varphi_n = 0.\tilde{\varphi}_n$. We then have that

$$
\begin{aligned}
|\tilde{r}_{j-1}.\tilde{\varphi}_j - 0.\varphi_{[j+1:]}| &= \sum_{i=2}^{m-n}\frac{1}{2^{i+1}}\varphi_{n+i} \\
&\leq \frac{1}{4} = 2^{j-n-1}, \quad \text{(B5)}
\end{aligned}
$$

as desired.

On the other hand, if $\varphi_{n+1} = 1$ and $\varphi_n = 0$, then $\Theta(1/\sqrt{2} - \sigma^{n-1}) = -1$, which results in setting $\theta_1 = 1$. Likewise, rounding $0.\varphi_n\varphi_{n+1}\cdots\varphi_m = 0.01\varphi_{n+2}\cdots\varphi_m$ to $n - j = 1$ decimal places yields $\tilde{r}_{n-1}.\tilde{\varphi}_n = 0.1$, so indeed

$\theta = 0.1 = 0.\tilde{\varphi}_n$. We then have that

$$
\begin{aligned}
|\tilde{r}_j.\tilde{\varphi}_{j+1} - 0.\varphi_{[j+1:]}| &= \left|\frac{1}{2} - \frac{1}{4} - \sum_{i=2}^{m-n}\frac{1}{2^{i+1}}\varphi_{n+i}\right| \\
&= \left|\frac{1}{4} - \sum_{i=2}^{m-n}\frac{1}{2^{i+1}}\varphi_{n+i}\right| \leq \frac{1}{4} = 2^{j-n-1}, \quad \text{(B6)}
\end{aligned}
$$

as desired.

Lastly, if $\varphi_{n+1} = 1$ and $\varphi_n = 1$, then $\Theta(1/\sqrt{2} - \sigma^{n-1}) = 1$, which results in setting $\theta_1 = 0$. In addition, rounding $0.\varphi_n\varphi_{n+1}\cdots\varphi_m = 0.11\varphi_{n+2}\cdots\varphi_m$ to $n - j = 1$ decimals yields $\tilde{r}_{n-1}.\tilde{\varphi}_n = 1.0$, so indeed $\theta = 0.0 = 0.\tilde{\varphi}_n$. We then have that

$$
|\tilde{r}_j.\tilde{\varphi}_{j+1} - 0.\varphi_{[j+1:]}| = \left|1 - \frac{1}{2} - \frac{1}{4} - \sum_{i=2}^{m-n}\frac{1}{2^{i+1}}\varphi_{n+i}\right|
$$

$$
\left|\frac{1}{4} - \sum_{i=2}^{m-n}\frac{1}{2^{i+1}}\varphi_{n+i}\right| \leq \frac{1}{4} = 2^{j-n-1}, \quad \text{(B7)}
$$

as desired.

General Case: Proceeding now to the general case, assume that this claim is true at the end of iteration $j + 1$ and all prior iterations. That is, $\theta = 0.\tilde{\varphi}_{j+2}\tilde{\varphi}_{j+3}\cdots\tilde{\varphi}_n$ at the end of iteration $j + 1$. Then, at the start of iteration j, $\theta \leftarrow \theta/2 = 0.0\tilde{\varphi}_{j+2}\tilde{\varphi}_{j+3}\cdots\tilde{\varphi}_n$. Again using Eq. (65), we see that $A_j(\theta)$ has singular value

$$
\begin{aligned}
\sigma^j &= \left|\cos\left[\pi(0.\varphi_{j+1}\varphi_{j+2}\cdots\varphi_m - \theta)\right]\right| \\
&= \left|\cos\left[\frac{\pi}{2}\left(\varphi_{j+1} + \frac{1}{2}(\varphi_{j+2} - \tilde{\varphi}_{j+2})\right.\right.\right. \\
&\quad \left.\left.\left. + \sum_{i=3}^{n-j}\frac{1}{2^{i-1}}(\varphi_{j+i} - \tilde{\varphi}_{j+i}) + \sum_{i=n-j+1}^{m-j}\frac{1}{2^{i-1}}\varphi_{n+i}\right)\right]\right|.
\end{aligned}
$$
(B8)

As before, we are interested in $\Theta(1/\sqrt{2} - \sigma^j)$, which, as we see below, is dictated by the values of φ_{j+1}, φ_{j+2}, and $\tilde{\varphi}_{j+2}$. Let us now look at the possible cases of these values.

First, if $\tilde{\varphi}_{j+2} = \varphi_{j+2}$, then $\Theta(1/\sqrt{2} - \sigma^j)$ is dictated entirely by φ_{j+1} and so we ultimately set $\theta_1 = \varphi_{j+1}$. In addition, the condition $\tilde{\varphi}_{j+2} = \varphi_{j+2}$ implies that no rounding occurs in the $j + 1$ iteration, so $\tilde{r}_{j+1} = 0$ and $\tilde{r}_j.\tilde{\varphi}_{[j+1:]} = 0.\varphi_{j+1}\varphi_{j+2}\tilde{\varphi}_{j+3}\cdots\tilde{\varphi}_n$ is the rounded value of $0.\varphi_{[j+1:]}$. Therefore, we indeed have $\theta = 0.\tilde{\varphi}_{j+1}\tilde{\varphi}_{j+2}\cdots\tilde{\varphi}_n$, and also

$$
\begin{aligned}
|\tilde{r}_j.\tilde{\varphi}_{j+1} - 0.\varphi_{j+1}| &= \frac{1}{2}|\tilde{r}_{j+1}.\tilde{\varphi}_{j+2} - 0.\varphi_{j+2}| \\
&\leq \frac{1}{2}2^{(j+1)-n-1} = 2^{j-n-1}, \quad \text{(B9)}
\end{aligned}
$$

where we use the inductive hypothesis.

On the other hand, consider $\tilde{\varphi}_{j+2} = 1$ and $\varphi_{j+2} = 0$. This condition implies that, at the $j+1$ iteration, $0.\varphi_{[j+2:]} = 0.0\varphi_{j+3}\cdots\varphi_m$ is rounded to $\tilde{r}_{j+1}.\tilde{\varphi}_{[j+2:]} = 0.1\tilde{\varphi}_{j+3}\cdots\tilde{\varphi}_n$, which requires that $\varphi_{j+3} = 1 = \varphi_{j+4} = \cdots\varphi_{n+1}$ and $\tilde{\varphi}_{j+3} = 0 = \tilde{\varphi}_{j+4} = \cdots = \tilde{\varphi}_n$. Using these values, we see that $\tilde{r}_j.\tilde{\varphi}_{[j+1:]} = 0.\varphi_{j+1}\tilde{\varphi}_{j+2}\cdots\tilde{\varphi}_n = 0.\varphi_{j+1}100\cdots0$ is the rounded value of φ^j. In addition, inputting all of these values into $\Theta\left(1/\sqrt{2} - \sigma^j\right)$, we find that we will ultimately set $\theta_1 = \varphi_{j+1}$. Therefore, we indeed have $\theta = 0.\tilde{\varphi}_{j+1}\tilde{\varphi}_{j+2}\cdots\tilde{\varphi}_n$, and again

$$|\tilde{r}_j.\tilde{\varphi}_{j+1} - 0.\varphi_{j+1}| = \frac{1}{2}|\tilde{r}_{j+1}.\tilde{\varphi}_{j+2} - 0.\varphi_{j+2}|$$
$$\leq \frac{1}{2}2^{(j+1)-n-1} = 2^{j-n-1}, \quad \text{(B10)}$$

where we again use the inductive hypothesis.

Finally, consider $\tilde{\varphi}_{j+2} = 0$ and $\varphi_{j+2} = 1$. This condition implies that, at the $j+1$ iteration, $0.\varphi_{[j+2:]} = 0.1\varphi_{j+3}\cdots\varphi_m$ is rounded to $\tilde{r}_{j+1}.\tilde{\varphi}_{[j+2:]} = 1.0\tilde{\varphi}_{j+3}\cdots\tilde{\varphi}_n$, which requires that $\varphi_{j+3} = 1 = \varphi_{j+4} = \cdots\varphi_{n+1}$ and $\tilde{\varphi}_{j+3} = 0 = \tilde{\varphi}_{j+4} = \cdots = \tilde{\varphi}_n$. Thus, if $\varphi_{j+1} = 0$, then $\tilde{r}_j.\tilde{\varphi}_{[j+1:]} = 0.100\cdots.0$ is the rounded value of $0.\varphi_{[j+1:]} = 0.011\cdots1\varphi_{n+2}\cdots\varphi_m$. Inputting these values into $\Theta\left(1/\sqrt{2} - \sigma^j\right)$, we find that we will ultimately set $\theta_1 = 1 = \tilde{\varphi}_{j+1} = 1 - \varphi_{j+1}$. Therefore, we indeed have $\theta = 0.\tilde{\varphi}_{j+1}\tilde{\varphi}_{j+2}\cdots\tilde{\varphi}_n$, and also

$$|\tilde{r}_j.\tilde{\varphi}_{j+1} - 0.\varphi_{j+1}| = |0.100\cdots0 - 0.011\cdots1\varphi_{n+2}\cdots\varphi_m|$$

$$= \left|\frac{1}{2} - \sum_{i=2}^{n-j+1}\frac{1}{2^i} - \sum_{i=n-j+2}^{m-j}\frac{1}{2^i}\varphi_{j+i}\right|$$

$$= \left|\frac{1}{2^{n-j+1}} - \sum_{i=n-j+2}^{m-j}\frac{1}{2^i}\varphi_{j+i}\right| \leq 2^{j-n-1}.$$
(B11)

Similarly, if $\varphi_{j+1} = 1$, then $\tilde{r}_j.\tilde{\varphi}_{[j+1:]} = 1.00\cdots0$ is the rounded value of $0.\varphi_{[j+1:]} = 0.11\cdots1\varphi_{n+2}\cdots\varphi_m$. Inputting these values into $\Theta\left(1/\sqrt{2} - \sigma^j\right)$, we will ultimately set $\theta_1 = 0 = \tilde{\varphi}_{j+1} = 1 - \varphi_{j+1}$. Therefore, we indeed have $\theta = 0.\tilde{\varphi}_{j+1}\tilde{\varphi}_{j+2}\cdots\tilde{\varphi}_n$, and also

$$|\tilde{r}_j.\tilde{\varphi}_{j+1} - 0.\varphi_{j+1}| = |1.00\cdots0 - 0.11\cdots1\varphi_{n+2}\cdots\varphi_m|$$

$$= \left|1 - \sum_{i=1}^{n-j+1}\frac{1}{2^i} - \sum_{i=n-j+2}^{m-j}\frac{1}{2^i}\varphi_{j+i}\right|$$

$$= \left|\frac{1}{2^{n-j+1}} - \sum_{i=n-j+2}^{m-j}\frac{1}{2^i}\varphi_{j+i}\right| \leq 2^{j-n-1}.$$
(B12)

This proves the general case and the entire proof is complete. ∎

Furthermore, we must discuss the additional $j = 0$ iteration, which is just like that of the phase estimation-by-QSVT algorithm. The analysis of this iteration is identical to the case above, just with the modification that $\varphi_0 = 0$ because $\varphi < 1$. Ultimately, we find that this step will usually output $\theta_0 = 0$, such that $\theta = 0.\tilde{\varphi}_{[1:]}$. However, if $\varphi_1 = 1 = \varphi_2 = \cdots = \varphi_{n+1}$ and $\tilde{\varphi}_1 = 0 = \tilde{\varphi}_2 = \cdots = \tilde{\varphi}_n$, then this step will output $\theta_0 = 1$, such that $\theta = 1.00\cdots0$. This accounts for the possibility that 1.0 is the best approximation to φ. For instance, the best two-decimal approximation to $\varphi = 0.1110101$ is $\theta = 1.00$. In either case, θ still satisfies $|\theta - \varphi| \leq 2^{-n-1}$, as Lemma 12 dictates at $j = 0$.

Finally, we mention one last caveat. It is possible that $\sigma^j = 1/\sqrt{2}$, in which case the sign function is 0 and we are equally likely to measure 0 or 1. However, this is not a problem, as the condition $\sigma^j = 1/\sqrt{2}$ implies that $|0.\varphi_{[j+1:]} - 0.0\tilde{\varphi}_{[j+2:]}| = 0.1 = \frac{1}{2}$, such that both $\varphi_{j+1} = 0$ and $\varphi_{j+1} = 1$ are equally accurate approximations, and so the inequality in Theorem 7 is still obeyed. This is analogous to conventional rounding, wherein one could round 0.5 to either 0 or 1 without changing the accuracy of the rounding. With these concerns alleviated, the proof of Theorem 7 is complete.

2. Theorem 8

Here, we prove Theorem 8 from Sec. V A 2:

Theorem 8: *If we choose Δ such that*

$$\Delta < 2\left[\cos\left(\frac{3\pi}{16}\right) - \frac{1}{\sqrt{2}}\right] \approx 0.25, \quad \text{(B13)}$$

then an error due to Δ can only occur at the $j = n - 1$ iteration. If an error is made at this iteration, then at the end of the algorithm, $|\theta - \varphi| < 1/2^n$, assuming that no errors are made at later iterations.

As discussed in Sec. V A2, we assume that we perform the QSVT with a function that behaves as $P_{\epsilon,\Delta}^{\Theta}\left(1/\sqrt{2} - x\right)$ (for $x \geq 0$), which approximates the sign function. Recall that this approximation fails in the region $[1/\sqrt{2} - \Delta/2, 1/\sqrt{2} + \Delta/2]$, which we dub "the Δ region." We show that the adverse effects of a finite-sized Δ region can be mitigated by choosing a sufficiently small Δ.

Throughout this section, we again use the notation $0.\varphi_{[j:]} := 0.\varphi_j\varphi_{j+1}\cdots$ to denote a string of contiguous binary digits. We also use the definition of $\tilde{r}_j$ from Sec. B1 b.

To demonstrate our claim, recall that $\sigma^j = |\cos[\pi(0.\varphi_{[j+1:]} - \theta)]|$ as per Eq. (73). Whether or not σ^j is inside of the Δ region is dictated by the value of $|0.\varphi_{[j+1:]} - \theta|$. In particular, in order for σ^j to be inside
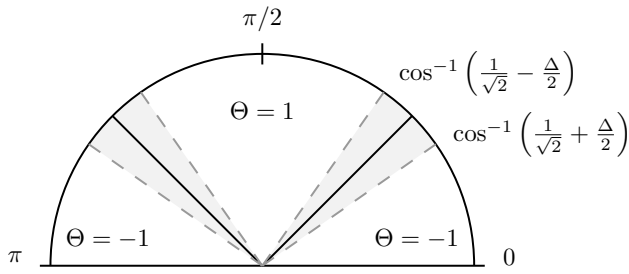
FIG. 18.   A depiction of a finite-sized Δ region and the resulting values of Θ. The shaded regions indicate where the application of the QSVT sequence returns an indeterminate measurement result (equivalently, where the sign function is poorly approximated).

of the Δ region, we require $|\cos[\pi(0.\varphi_{[j+1:]} - \theta)]| \in \left[1/\sqrt{2} - \Delta/2, 1/\sqrt{2} + \Delta/2\right]$ or, equivalently,

$$\pi|0.\varphi_{[j+1:]} - \theta|$$
$$\in \left[\arccos\left(\frac{1}{\sqrt{2}} + \frac{\Delta}{2}\right), \arccos\left(\frac{1}{\sqrt{2}} - \frac{\Delta}{2}\right)\right]$$
$$\cup \left[\arccos\left(-\frac{1}{\sqrt{2}} + \frac{\Delta}{2}\right), \arccos\left(-\frac{1}{\sqrt{2}} - \frac{\Delta}{2}\right)\right]. \tag{B14}$$

We depict this condition graphically in Fig. 18, which is a useful illustration for understanding the proof of this theorem.

As we show below, if we choose Δ such that σ^j is within the Δ region only if $|0.\varphi_{[j+1:]} - \theta| \in (1/4 - 1/16, 1/4 + 1/16) \cup (3/4 - 1/16, 3/4 + 1/16)$, then it is only possible for σ^j to be within the Δ region at iteration $j = n - 1$. In order to enforce this constraint on the Δ region, we require that

$$\left|\frac{1}{\pi}\arccos\left(\frac{1}{\sqrt{2}} \pm \frac{\Delta}{2}\right) - \frac{1}{4}\right| < \frac{1}{16}. \tag{B15}$$

As it turns out, the $+$ condition is actually more stringent, so we have the following lemma:

Lemma 13: *If we choose Δ such that*

$$\left[\frac{1}{4} - \frac{1}{\pi}\arccos\left(\frac{1}{\sqrt{2}} + \frac{\Delta}{2}\right)\right] < \frac{1}{16}, \tag{B16}$$

then σ^j can only be inside of the Δ region at iteration $j = n - 1$.

Proof. To prove this, we begin by analyzing the possible values of σ^j. We assume that, if σ^j is outside of the Δ region, then we can correctly determine θ_1 with

high probability using an appropriate value of ϵ as in Sec. 2. In addition, note that our restriction on Δ implies that, in order for σ^j to be inside of the Δ region, we must have $|0.\varphi_{[j+1:]} - \theta| \in (1/4 - 1/16, \ 1/4 + 1/16) \cup (3/4 - 1/16, \ 3/4 + 1/16)$.

First, let $j = n - 1$ and suppose that we can correctly determine θ_1 (either σ^j is outside of the Δ region, or σ^j is inside of the Δ region and we get lucky). Next, proceed to iteration $j = n - 2$. If $\tilde{r}_{n-1} = 0$ at the previous iteration, then Lemma 12 implies that

$$|0.\varphi_{[n-1:]} - \theta| = |0.\varphi_{[n-1:]} - 0.\tilde{r}_{n-1}\tilde{\varphi}_{n:}|$$
$$\in \left[0.\varphi_{n-1} - \frac{1}{2}2^{n-1-n-1}, \ 0.\varphi_{n-1}\right.$$
$$\left. + \frac{1}{2}2^{n-1-n-1}\right]$$
$$= \left[0.\varphi_{n-1} - \frac{1}{8}, \ 0.\varphi_{n-1} + \frac{1}{8}\right]. \tag{B17}$$

For either possible value of φ_{n-1}, the restriction on Δ implies that the corresponding value of σ^{n-2} is not within the Δ region and so we can correctly determine θ_1 with high probability at this iteration.

On the other hand, if $\tilde{r}_{n-1} = 1$, then $0.\varphi_{[n:]} > 0.11$ and $\theta = 0$. At iteration $j = n - 2$, we have

$$|0.\varphi_{[n-1:]} - \theta| = 0.\varphi_{[n-1:]} > 0.\varphi_{n-1}11. \tag{B18}$$

Again, because $0.011 = 1/4 + 1/8 > 1/4 + 1/16$, the corresponding value of σ^{n-2} is not within the Δ region for either possible value of φ_{n-1} and so we can correctly determine θ_1 with high probability at this iteration.

By inductively following this logic to further iterations, we see that if we can correctly determine θ_1 at iteration $j = n - 1$, then the subsequent values of σ^j will not be in the Δ region and the corresponding values of θ_1 can be correctly determined with high probability.

Next, again let $j = n - 1$ and now suppose that σ^j is inside of the Δ region, such that we choose an incorrect value for θ_1. First, consider the case in which $1/4 < 0.\varphi_{[n:]} < 1/4 + 1/16$ and we make an error by setting $\theta_1 = 0$. Then at iteration $j = n - 2$,

$$|0.\varphi_{[n-1:]} - \theta| = 0.\varphi_{n-1}\varphi_{[n:]}. \tag{B19}$$

This is bounded below by $0.\varphi_{n-1} + 1/8$ and above by $0.\varphi_{n-1} + 1/8 + 1/32$, so σ^{n-2} does not fall within the Δ region. Hence, despite our initial error, the correct value for θ_1 may be determined with high probability at the next iteration. As per the result of the previous paragraphs, this indicates that, with high probability, σ^j will not be inside the Δ region at later iterations.

Next, consider the case in which $1/4 - 1/16 < 0.\varphi_{[n:]} < 1/4$ and we make an error by setting $\theta_1 = 1$. Then at iteration $j = n - 2$,

$$0.\varphi_{[n-1:]} - 0.01 = 0.\varphi_{n-1}\varphi_{[n:]} - 0.01. \qquad \text{(B20)}$$

This is bounded below by $0.\varphi_{n-1} - 1/8 - 1/32$ and above by $0.\varphi_{n-1} - 1/8$, so again σ^{n-2} does not fall within the Δ region.

Finally, we note that the cases in which $3/4 < 0.\varphi_{[n:]} < 3/4 + 1/16$ and $3/4 - 1/16 < 0.\varphi_{[n:]} < 3/4$ are analogous to the two cases illustrated above and we ultimately find that σ^{n-2} is not within the Δ region. $\blacksquare$

This theorem tells us that if Δ is made sufficiently small, then we are only plagued by the Δ region at iteration $j = n - 1$. At all other iterations, σ^j will not be within the Δ region and we can correctly determine θ_1 with high probability. Thus, if at iteration $j = n - 1$, σ^{n-1} is within the Δ region and we do make an error, then our approximation of $0.\varphi_{[n:]}$ is incorrect by some amount $< 1/4 + 1/16 < 1/2$, so the overall error in our estimate of φ will be $< 1/2^n$ (with high probability).

Lastly, if we rearrange the inequality in Lemma 13, then we see that the necessary restriction on Δ must be

$$\Delta < 2\left[\cos\left(\frac{3\pi}{16}\right) - \frac{1}{\sqrt{2}}\right] \approx 0.25. \qquad \text{(B21)}$$

So if we satisfy this constraint, then we can we can guarantee that the algorithm will succeed with high probability and suffer error $< 1/2^n$. This proves Theorem 8.

As a corollary, suppose more generally that we choose Δ such that $\left[1/4 - 1/\pi \arccos(1/\sqrt{2} + \Delta/2)\right] < \gamma$ for some $\gamma > 0$. In order for σ^j to be inside of the Δ region under this constraint, we must have $|0.\varphi_{[j+1:]} - \theta| \in (1/4 - \gamma, \ 1/4 + \gamma) \cup (3/4 - \gamma, \ 3/4 + \gamma)$. Then, suppose that σ^{n-1} is inside the Δ region such that $1/4 < 0.\varphi_{[n:]} < 1/4 + \gamma$ and we make an error by setting $\theta_1 = 0$. Then, at iteration $j = n - 2$,

$$|0.\varphi_{[n-1:]} - \theta| = 0.\varphi_{n-1}\varphi_{[n:]}, \qquad \text{(B22)}$$

which is bounded below by $0.\varphi_{n-1} + 1/8$ and above by $0.\varphi_{[n-1:]} + 1/8 + \gamma/2$. Hence this quantity is necessarily a distance $1/4 - \gamma - (1/8 + \gamma/2) = 1/8 - 3\gamma/2$ from the Δ region and this distance increases at later iterations. Identical bounds hold for the other cases in which σ^{n-1} is in the Δ region. In this sense, the quantity $0.\varphi_{[j+1:]}$ can suffer an additive error of magnitude $< 1/8 - 3\gamma/2$ and the phase estimation algorithm will still output a θ such that $|\varphi - \theta| < 1/2^n$ with high probability, as claimed in Sec. V C 2.

APPENDIX C: CONSTRUCTION OF THE MATRIX-INVERSION POLYNOMIAL

As we describe in Sec. VI B, in order to invert a matrix with the QSVT, we desire an $\epsilon/2\kappa$ approximation to $(1/2\kappa)1/x$, where κ is the condition number of the matrix to be inverted. Gilyén *et al.* design such a polynomial by first noting that the function

$$g_{\epsilon,\kappa}(x) = \frac{1 - (1 - x^2)^b}{x} \qquad \text{(C1)}$$

provides a good approximation to $1/x$ over the range $x \in [-1, 1] \setminus [-1/\kappa, 1/\kappa]$ for large b. In particular, for $0 < \epsilon < \frac{1}{2}$, $g_{\epsilon,\kappa}(x)$ ϵ approximates $1/x$ over the range $x \in [-1, 1] \setminus [-1/\kappa, 1/\kappa]$ for $b(\epsilon, \kappa) = \lceil \kappa^2 \log(\kappa/\epsilon) \rceil$ [12,50].

Next, although $g_{\epsilon,\kappa}(x)$ is not a polynomial, it can be ϵ approximated over the range $x \in [-1, 1]$ by the polynomial

$$P_{2\epsilon,\kappa}^{1/x}(x) = 4\sum_{j=0}^{D}(-1)^j\left[2^{-2b}\sum_{i=j+1}^{b}\binom{2b}{b+i}\right]T_{2j+1}(x),$$
$$\text{(C2)}$$

where $T_i(x)$ is the Chebyshev polynomial of order i and $D(\epsilon, \kappa) = \left\lceil \sqrt{b(\epsilon, \kappa)\log[4b(\epsilon, \kappa)/\epsilon]} \right\rceil = \mathcal{O}[\kappa\log(\kappa/\epsilon)]$ is the degree of this polynomial [12]. In addition, by the triangle inequality, $P_{2\epsilon,\kappa}^{1/x}(x)$ is a 2ϵ approximation to $1/x$ for $x \in [-1, 1] \setminus [-1/\kappa, 1/\kappa]$; hence the subscript 2ϵ. To provide intuition, we illustrate this polynomial in Fig. 19.

At this stage, we may suspect that our candidate polynomial is $1/2\kappa P_{\epsilon/2,2\kappa}^{1/x}(x)$, which ϵ approximates $(1/2\kappa)1/x$ for $x \in [-1, 1] \setminus [-1/2\kappa, 1/2\kappa]$. In addition, for $\epsilon < 1/2$, this polynomial is necessarily bounded in magnitude by 1 for $x \in [-1, 1] \setminus [-1/\kappa, 1/\kappa]$, which is easily seen via the triangle inequality. Unfortunately, however, this candidate polynomial is not necessarily bounded for $x \in [-1/2\kappa, 1/2\kappa]$. In particular, the approximation of Eq. (C2) obeys [12,50]

$$\max_{x\in[-1,1]}|P_{\epsilon,\kappa}^{1/x}(x)| \leq 4D(\epsilon,\kappa) = \mathcal{O}[\kappa\log(\kappa/\epsilon)], \qquad \text{(C3)}$$

so our candidate polynomial is only bounded in magnitude by $(1/2\kappa)4D(\epsilon/2, 2\kappa) = \mathcal{O}[\log(\kappa/\epsilon)]$, which is not necessarily 1.

Therefore, to enforce that the magnitude of the candidate polynomial be bounded, we may multiply it by an even function that is close to 1 for $x \in [-1, 1] \setminus [-1/\kappa, 1/\kappa]$, and close to 0 for $x \in [-1/2\kappa, 1/2\kappa]$, letting the range $x \in [-1/\kappa, -1/2\kappa] \cup [1/2\kappa, 1/\kappa]$ be a transition region between these two values. Such a rectangular function may be polynomially approximated by a linear combination of
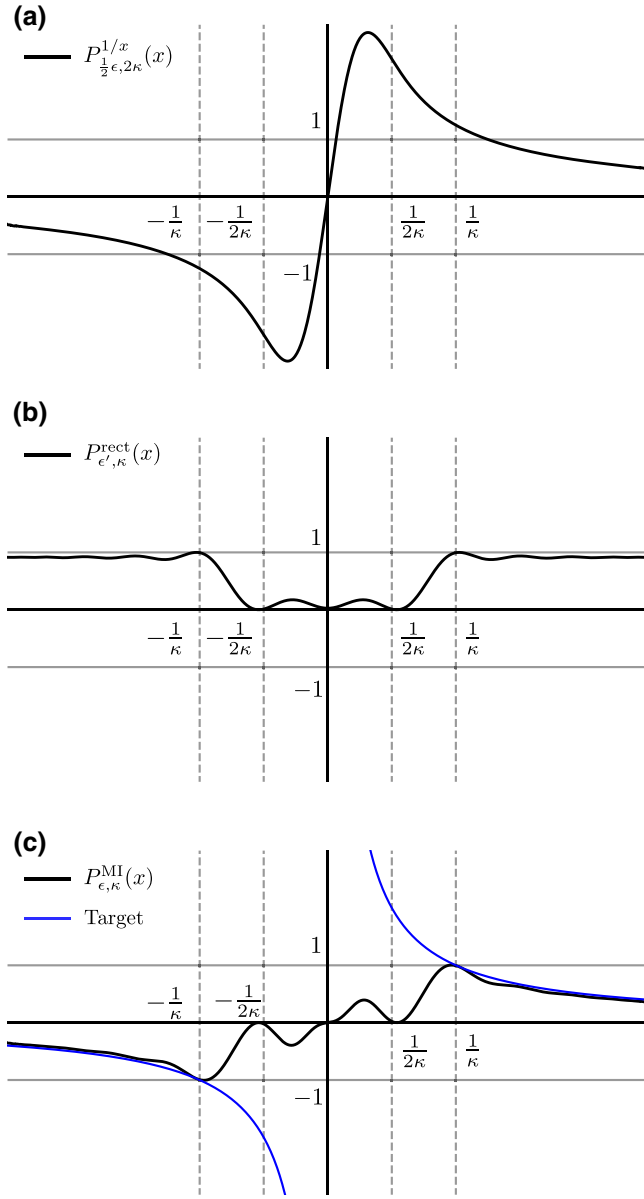
(a)

(b)

(c)

FIG. 19. A sketch of (a) $P^{1/x}_{\frac{1}{2}\epsilon,2\kappa}(x)$, (b) $P^{\text{rect}}_{\epsilon',\kappa}(x)$, and (c) $P^{\text{MI}}_{\epsilon,\kappa}(x)$ for $\kappa = 2.5$. The resulting polynomial, $P^{\text{MI}}_{\epsilon,\kappa}(x)$, is a degree-77 approximation to the inverse function.

the step function approximations of Sec. III:

$$
\begin{aligned}
P^{\text{rect}}_{\epsilon,\kappa}(x) := \frac{1}{1+\frac{\epsilon}{2}} \Bigg\{ 1 + \frac{1}{2} \Bigg[P^{\Theta}_{\epsilon,\frac{1}{4\kappa}} \left(x - \frac{3}{4\kappa} \right) \\
+ P^{\Theta}_{\epsilon,\frac{1}{4\kappa}} \left(-x - \frac{3}{4\kappa} \right) \Bigg] \Bigg\},
\end{aligned}
\tag{C4}
$$

which is easily seen to obey

$$
P^{\text{rect}}_{\epsilon,\kappa}(x) \in [1-\epsilon, 1] \ \forall x \in [-1, 1] \setminus \left[\frac{-1}{\kappa}, \frac{1}{\kappa} \right]
$$

$$
P^{\text{rect}}_{\epsilon,\kappa}(x) \in [0, \epsilon] \ \forall x \in \left[\frac{-1}{2\kappa}, \frac{1}{2\kappa} \right]
\tag{C5}
$$

and has even degree $\mathcal{O}[\kappa \log(1/\epsilon)]$. In the in-between region $x \in [-1/\kappa, -1/2\kappa] \cup [1/2\kappa, 1/\kappa]$, $P^{\text{rect}}_{\epsilon,\kappa}(x)$ transitions between values close to 0 and close to 1, remaining bounded in magnitude by 1 throughout. We illustrate the behavior of this polynomial in Fig. 19.

Therefore, our target polynomial is the *matrix-inversion polynomial*:

$$
P^{\text{MI}}_{\epsilon,\kappa}(x) := \frac{1}{2\kappa} P^{1/x}_{\frac{1}{2}\epsilon,2\kappa}(x) P^{\text{rect}}_{\epsilon',\kappa}(x),
\tag{C6}
$$

where $\epsilon' = \min[2\epsilon/5\kappa, \kappa/2D(\epsilon/4, 2\kappa)] = \mathcal{O}(\epsilon/\kappa)$. We illustrate this polynomial in Fig. 19. In defining ϵ', the term $2\epsilon/5\kappa$ ensures that this polynomial is an $\epsilon/2\kappa$ approximation to $(1/2\kappa)1/x$ over the range of possible singular values and the term $\kappa/2D(\epsilon/4, 2\kappa)$ ensures that $P^{\text{MI}}_{\epsilon,\kappa}(x)$ is bounded in magnitude by 1 for $x \in [-1/\kappa, 1/\kappa]$. Indeed, over the range $x \in [-1, 1] \setminus [-1/\kappa, 1/\kappa]$, $P^{\text{MI}}_{\epsilon,\kappa}(x)$ is an $\epsilon/2\kappa$ approximation to $1/2\kappa 1/x$:

$$
\begin{aligned}
\left| P^{\text{MI}}_{\epsilon,\kappa}(x) - \frac{1}{2\kappa}\frac{1}{x} \right| &\le \frac{1}{2\kappa} \left| P^{1/x}_{\frac{1}{2}\epsilon,2\kappa}(x)(1-\epsilon') - \frac{1}{x} \right| \\
&\le \frac{1}{2\kappa} \left(\left| P^{1/x}_{\frac{1}{2}\epsilon,2\kappa}(x) - \frac{1}{x} \right| + \epsilon' \left| P^{1/x}_{\frac{1}{2}\epsilon,2\kappa}(x) \right| \right) \\
&\le \frac{1}{2\kappa} \left[\frac{\epsilon}{2} + \frac{2\epsilon}{5\kappa} \left(\kappa + \frac{\epsilon}{2} \right) \right] \\
&\le \frac{1}{2\kappa} \left(\frac{\epsilon}{2} + \frac{2\epsilon}{5} + \frac{\epsilon}{10} \right) = \frac{\epsilon}{2\kappa},
\end{aligned}
\tag{C7}
$$

where we use the (loose) bound $\epsilon/\kappa < \frac{1}{2}$. Likewise, $P^{\text{MI}}_{\epsilon,\kappa}(x)$ is bounded above by 1 for $x \in [-1/2\kappa, 1/2\kappa]$:

$$
\left| P^{\text{MI}}_{\epsilon,\kappa}(x) \right| \le \frac{1}{2\kappa} 4D(\epsilon/4, 2\kappa)\epsilon' \le 1.
\tag{C8}
$$

Similarly, for the in-between region $x \in [-1/\kappa, -1/2\kappa] \cup [1/2\kappa, 1/\kappa]$, both components of $P^{\text{MI}}_{\epsilon,\kappa}(x)$ are bounded in magnitude by 1, so $|P^{\text{MI}}_{\epsilon,\kappa}(x)| \le 1$.

Finally, it is easy to compute the degree of $P^{\text{MI}}_{\epsilon,\kappa}(x)$, which is the sum of the degrees of $P^{1/x}_{\frac{1}{2}\epsilon,2\kappa}(x)$ and $P^{\text{rect}}_{\epsilon',\kappa}(x)$:

$$
d = \mathcal{O}[\kappa \log(\kappa/\epsilon) + \kappa \log(1/\epsilon')] = \mathcal{O}[\kappa \log(\kappa/\epsilon)].
\tag{C9}
$$

APPENDIX D: QSP PHASE ANGLE-SEQUENCE EXAMPLES

Presented in this appendix are some explicit polynomials and corresponding QSP phase angles, for functions that are useful in quantum signal-processing applications. These are not necessarily the optimal polynomials, nor the best QSP phase angles, but they are pedagogically clear starting points. Unless otherwise specified, all QSP phases are given in the $(W_x, S_z, \langle + | \cdot | + \rangle)$-QSP convention. All the code for generating these phase angles is available in the PYQSP open-source repository on GitHub [55].

1. Oblivious amplitude amplification

For fixed-point search or oblivious amplitude amplification, it is desired to make a polynomial that maps a as close as possible to 1, for a wide range of small values of a, starting as close to $a = 0$ as possible.

One sequence of phases that accomplishes this optimally, with error $1 - \delta^2$, is given for $k = 0, 1, \ldots, d - 1$ by

$$\phi_{2k} = \alpha_{d-k-1}, \tag{D1}$$

$$\phi_{2k+1} = \alpha_{k+1}, \tag{D2}$$

where

$$\alpha_k = -\cot^{-1}\left(\sqrt{1 - \gamma^2} \tan \frac{2\pi(k+1)}{L}\right) \tag{D3}$$

and $L = 2d + 1$ and $\gamma^{-1} = T_{1/L}(1/\delta) = \cosh^{-1}(1/\delta)/\cosh(1/L)$ [37].

For example, for $d = 10$ and $\delta = 0.5$, the QSP phase angles are as follows:

```
pyqsp --plot-positive-only --plot-probability
--plot-tight-y --plot-npts=400
--seqargs=10,0.5 --plot fpsearch

[-1.58023603 -1.55147987 -1.6009483  -1.52812171
 -1.62884337 -1.49242141 -1.67885248 -1.41255145
 -1.8386054  -0.87463828 -0.87463828 -1.8386054
 -1.41255145 -1.67885248 -1.49242141 -1.62884337
 -1.52812171 -1.6009483  -1.55147987 -1.58023603]
```

The corresponding response function is shown in Fig. 20.

2. Sign function

The sign function $\Theta(a)$ has a number of applications for quantum signal processing. As discussed in Sec. III, a robust polynomial approximation of it can be obtained using the error function

$$\Theta(a) \approx \mathrm{erf}(ka), \tag{D4}$$

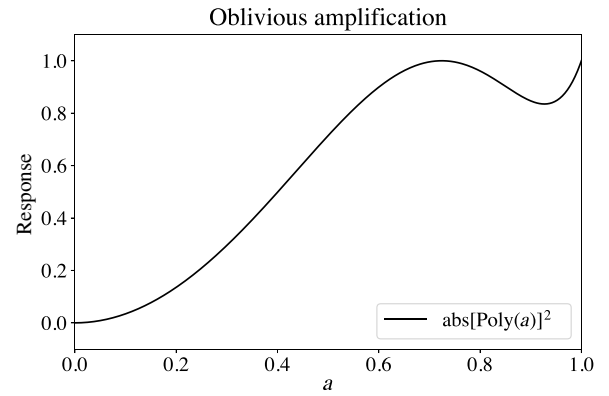where k is a (large) scaling factor.

FIG. 20. The transition probability using the polynomial for oblivious amplification for $d = 10$ and $\delta = 0.5$.

For example, the QSP phase angles obtained using a degree-$(d = 19)$ approximation with $k = 10$ are as follows:

```
pyqsp --plot-real-only --plot-npts=400
--seqargs=19,10 --plot poly_sign

[0.01558127 -0.01805798  0.05705643 -0.01661832
 0.16163773  0.09379074 -2.62342885  0.49168481
 0.92403822 -0.09696846 -0.09696846  0.92403822
 0.49168481 -2.62342885  0.09379074  0.16163773
-0.01661832  0.05705643 -0.01805798  1.5863776]
```

The corresponding response function is shown in Fig. 21. For $a > 0$, this sign function approximation may be employed as a step function, e.g., for oblivious amplitude amplification.

3. Matrix inversion using $1/a$

The $1/a$ function is useful for computing Moore-Penrose pseudoinverses of matrices using the quantum singular value transform. As discussed in Sec. VI B, Chebyshev polynomials may be employed to approximate this function [50].
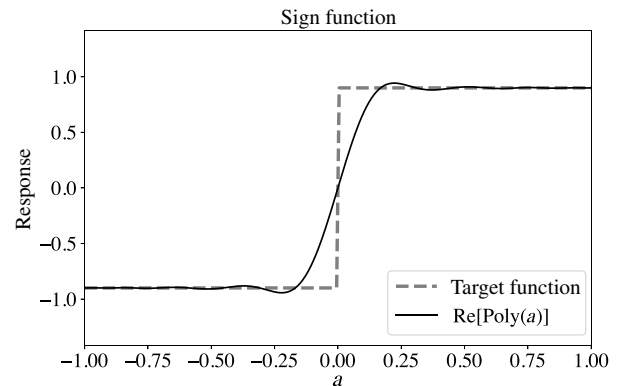

FIG. 21. The response function for the polynomial approximation to the sign function with $d = 19$ and $k = 10$.
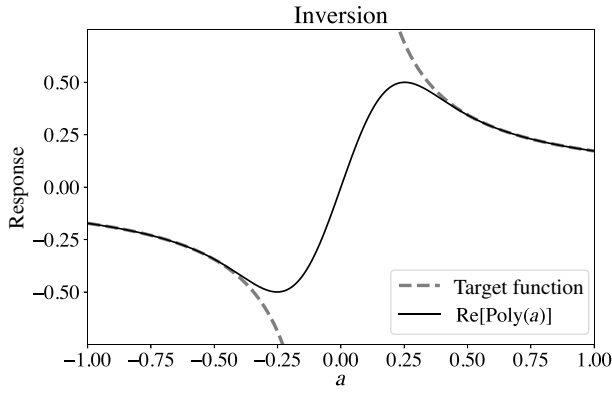
FIG. 22. The response function for the polynomial approximation to the inverse function with $\kappa = 3$ and $\epsilon = 0.3$.

FIG. 23. The response function for the polynomial approximation to the cosine function with $t = 5$ and $\epsilon = 0.1$.

For example, with $\kappa = 3$ and $\epsilon = 0.3$, a set of QSP phase angles for this polynomial is as follows:

```
pyqsp --plot-real-only --plot-npts=400
--seqargs=3,0.3 --plot invert

[-0.27237279 -1.8808697    2.19755533 -0.860515
  0.84659086  0.62794236 -0.69688032 -0.62874403
  0.7406656   0.44483992 -0.60489363 -0.60489363
  0.44483992  0.7406656  -0.62874403 -0.69688032
  0.62794236  0.84659086 -0.860515    -0.94403733
  1.26072295  1.29842354]
```

The corresponding response function is shown in Fig. 22.

4. Cosine and sine functions for Hamiltonian simulation

For Hamiltonian simulation, we seek an approximation to e^{-iat}. As discussed in Sec. VI A, this can be accomplished using the Jacobi-Anger approximations of $\cos(at)$ and $\sin(at)$ of Eqs. (76) and (77). The approximation is chosen to sufficient degree so as to bound the error to $\epsilon > 0$ in the region $a \in [-1, 1]$.

For example, with $t = 5$ and $\epsilon = 0.1$, a set of QSP phase angles for the approximation to $\cos(at)$ is as follows:

```
pyqsp  --plot-real-only --plot-npts=400
--seqargs=10,0.1 --plot hamsim

[-1.70932079 -0.05312746  2.12066859 -0.83307065
 -0.50074601  0.40728859  0.32838472  0.9142489
 -2.81320793  0.40728859 -0.50074601  2.30852201
 -1.02092406 -0.05312746  3.00306819]
```

The corresponding response function is shown in Fig. 23. Similarly, the QSP phases for the approximation to $\sin(at)$ are as follows:

```
[-1.63276817  0.20550406 -0.84198335  0.39732059
 -0.26820613  2.41324245  0.04662674 -2.02847501
  1.11311765  0.04662674 -0.72835021 -0.26820613
  0.39732059 -0.84198335  0.20550406 -0.06197184]
```
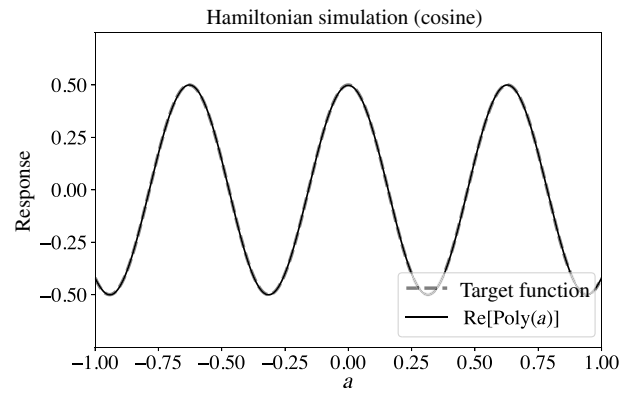
The corresponding response function is shown in Fig. 24.

5. Threshold function

Distinguishing eigenvalues and singular values may be performed using a step function, which we take to be $\text{step}(a - 1/2)$ for illustrative purposes. This may be polynomially approximated using a Taylor series expansion of

$$\text{step}(a - 1/2)$$
$$\approx \frac{1}{2}\left\{\text{erf}\left[k(a + 1/2)\right] - \text{erf}\left[k(a - 1/2)\right]\right\}, \quad \text{(D5)}$$

which becomes a good approximation for large k.

For example, with $k = 10$ and using a degree $d = 18$ Taylor series, a set of QSP phase angles for this polynomial is as follows:
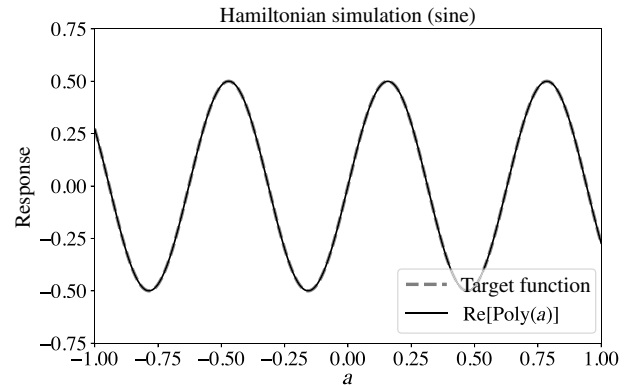

FIG. 24. The response function for the polynomial approximation to the sine function with $t = 5$ and $\epsilon = 0.1$.
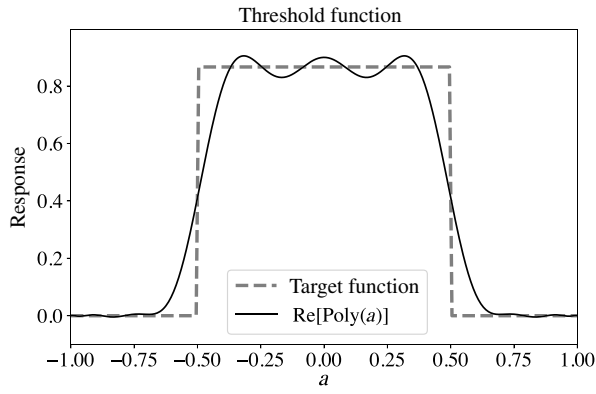
FIG. 25. The response function for the polynomial approximation to the threshold function with $k = 10$ and $d = 18$.

```
pyqsp --plot-real-only --plot-npts=400
--seqargs=18,10 --plot poly_thresh
```

```
[0.73930816 -0.69010006 -0.63972139 -0.47754554
 0.81797049  0.09205065 -0.87660105  0.13460844
 0.23892207  1.32216648 -2.90267058  0.13460844
 2.2649916   0.09205065 -2.32362216  2.66404712
-0.63972139 -0.69010006  2.31010449]
```

The corresponding response function is shown in Fig. 25.

Note that this is an even function of a but it may be used just in the region $a \geq 0$, e.g., to distinguish singular values that are above or below $1/2$. It can be made as sharp as desired by increasing k and the degree of the polynomial.

6. Linear amplitude amplification

Linear amplitude amplification is a subroutine that is useful for a number of quantum algorithms including simulation. The goal is to multiply inputs by a constant $1/2\Gamma$ for $\Gamma \in (0, 1/2)$. As is usual for QSP, the absolute value of the output must bounded by 1 and therefore we seek a polynomial approximation that performs the linear amplification only for small inputs. We can obtain a suitable approximation by truncating the Taylor expansion of

$$
\text{linear\_amplification}(a, \Gamma)
$$

$$
\approx \frac{a}{2\Gamma} \times \frac{1}{2}\left\{\text{erf}\big[k(a + 2\Gamma)\big] - \text{erf}\big[k(a - 2\Gamma)\big]\right\}, \quad \text{(D6)}
$$

where k is chosen to obtain the desired accuracy within the region $[-\Gamma, \Gamma]$. This approximation is described in further detail in Ref. [21].

For example, with $\Gamma = 0.25$ and using a degree $d = 19$ Taylor series, a set of QSP phase angles for this polynomial is as follows:
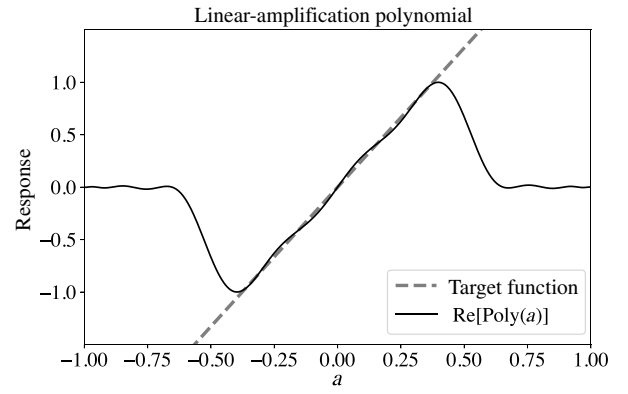

FIG. 26. The response function for the linear amplification polynomial with $\Gamma = 0.25$ and $d = 19$.

```
pyqsp --plot-real-only --plot-npts=400
--seqargs=19,0.25 --plot poly_linear_amp
```

```
[0.07658557 -0.01961714 -0.10257913  0.08269406
 0.16291683  0.43552219 -2.62323892  2.61402960
 0.02001814 -2.21710253  0.92449012  0.02001814
-0.52756304  0.51835372  0.43552219  0.16291683
 0.08269406 -0.10257913 -0.0196171 -1.49421074]
```

The corresponding response function is shown in Fig. 26.

7. Phase estimation polynomial

Similar to the threshold function is the phase estimation polynomial of Eq. (69), used in Sec. V.

For example, with $\Delta = 10$ and using a degree $d = 18$ Taylor series, a set of QSP phase angles for this polynomial in the $(W_x, S_z, \langle + | \cdot | + \rangle)$-QSP convention is as follows:

```
pyqsp --plot-real-only --plot-npts=400
--seqargs=18,10 --plot poly_phase
```

```
[-2.69295576  0.92644177 -2.47601161 -2.95790072
 -3.07710363  2.40352005  2.38432687 -3.0712802
 -2.54668308 -0.87407521  0.59490957 -3.0712802
 -0.75726578  2.40352005 -3.07710363 -2.95790072
 -2.47601161  0.92644177  2.01943322]
```

The corresponding response function is shown in Fig. 27.

A set of QSP phases for this polynomial can also be given in the $(W_x, S_z, \langle 0 | \cdot | 0 \rangle)$-QSP convention; use of this convention clarifies the reduction to the quantum Fourier transform presented in Sec. V D:

```
pyqsp --polydeg 16 --measurement="z"
--func="-1+np.sign(1/np.sqrt(2)-x)+
np.sign(1/np.sqrt(2)+x)"  --plot polyfunc
```

```
[0.6744825   2.4248297   2.7351234   2.7316442
 0.0127715   3.915519    2.3178308  -0.00533221
 2.3088486   2.36385     2.3181891   1.585311
 2.411246    0.4094785  -0.40136954  0.7154387
 1.8687413]
```

The corresponding response function is shown in Fig. 28. Note that we are no longer using the real polynomial approximation of Eq. (69), since it does not satisfy the conditions of Theorem 9; because of this, there is a small nonzero imaginary response. The QSP phase angles for this example are generated using an optimization algorithm.

8. Eigenstate filtering

As in the eigenvalue threshold problem of Sec. IV, suppose that $\mathcal{H}$ is a Hermitian matrix with an eigenvalue λ that is known to be separated from other eigenvalues by a gap $\Delta_\lambda > 0$ and the problem is to create, using QSP, a projection operator onto the eigenspace corresponding to λ. Lin and Tong [41] show that the degree-$(d = 2k)$ polynomial

$$f_k(x, \Delta_\lambda) = \frac{T_k\left(-1 + 2\frac{x^2 - \Delta_\lambda^2}{1 - \Delta_\lambda^2}\right)}{T_k\left(-1 + 2\frac{-\Delta_\lambda^2}{1 - \Delta_\lambda^2}\right)}, \qquad (\text{D7})$$

known as the "eigenstate filtering function," is an optimal polynomial for filtering out the unwanted information from all other eigenstates.

For example, with $\delta = 0.3$ and using a degree-$(d = 30)$ Taylor series, a set of QSP phase angles for this polynomial is as follows:

```
pyqsp --plot-positive-only --plot-real-only
--plot-tight-y --seqargs 30,0.3 --plot efilter

[-2.22655153   2.26982696  -0.76378956   0.07418111
  0.25458387   0.5916072    0.30309002   0.10101557
 -0.12943648  -1.00141425   0.60368299  -2.2897962
 -0.04337353   0.28364185   2.28161478  -0.61804648
 -0.85997787   0.28364185  -0.04337353   0.85179646
  0.60368299  -1.00141425  -0.12943648   0.10101557
  0.30309002   0.5916072    0.25458387   0.07418111
 -0.76378956  -0.87176569   2.48583745]
```
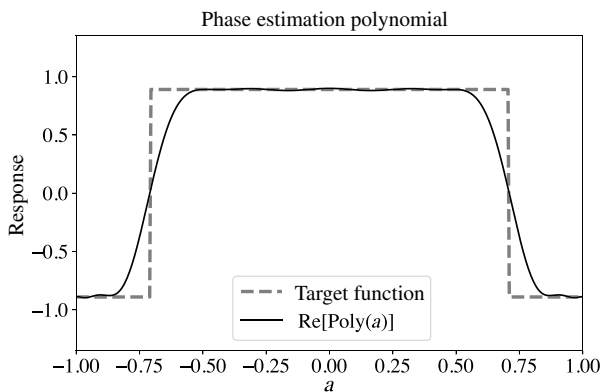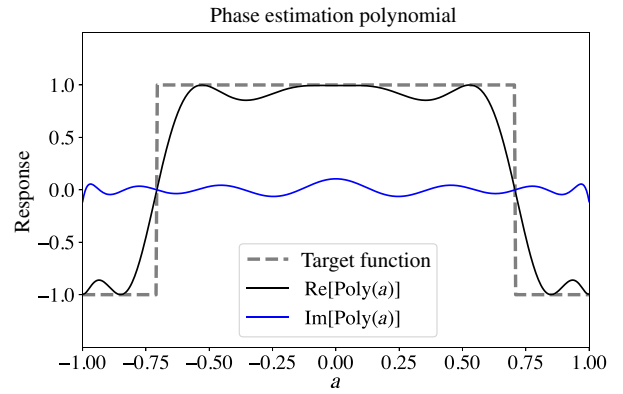

FIG. 28. The response function for a degree-18 polynomial approximation to the phase estimation function in the $(W_x, S_z, \langle 0| \cdot |0\rangle)$ convention.

which produces this response function for $a > 0$. The corresponding response function is shown in Fig. 29. This is a better threshold function than the one presented in Sec. 5 and the threshold can be located where desired by changing Δ_λ.

9. Gibbs distribution

Given $\mathcal{H} = \sum_\lambda \lambda |\lambda\rangle\langle\lambda|$, its corresponding Gibbs distribution state $\rho(\beta) = (1/Z)e^{-\beta\mathcal{H}}$ is the density matrix

$$\rho(\beta) = \frac{1}{Z}\sum_\lambda e^{-\beta\lambda}|\lambda\rangle\langle\lambda|. \qquad (\text{D8})$$

An approximation to $e^{-\beta a}$ is useful for obtaining ρ using QSP. To ensure that the function has definite parity, we choose a polynomial approximation to $e^{-\beta|a|}$.

For example, with $\beta = 3.5$ and using a degree $d = 20$ Taylor series, a set of QSP phase angles for this polynomial is as follows:

FIG. 27. The response function for the real phase estimation polynomial with $\Delta = 10$ and $d = 18$ in the $(W_x, S_z, \langle +| \cdot |+\rangle)$-QSP convention.
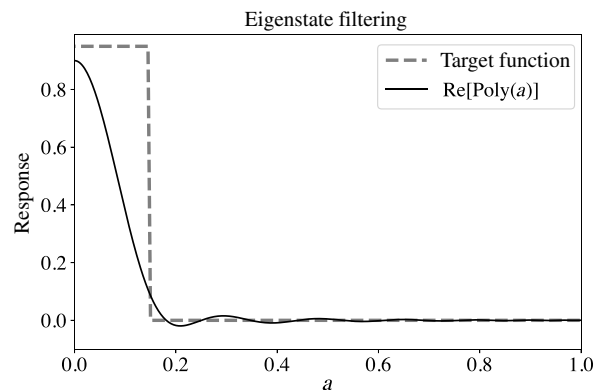

FIG. 29. The response function for the polynomial approximation to the eigenstate filtering function with $\delta = 0.3$ and $d = 30$.
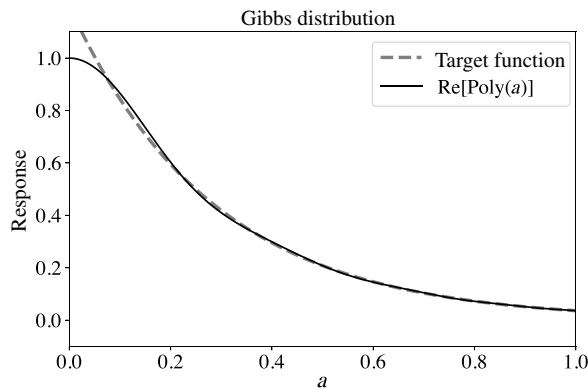
FIG. 30. The response function for the polynomial approximation to the Gibbs distribution for $a > 0$ with $\beta = 3.5$ using a degree-($d = 20$) polynomial. The response is scaled to maximize Poly(a). Note that the approximation deviates from the target near $a = 0$, as the symmetrized version of the Gibbs distribution is nonanalytic about that point.

```
pyqsp --plot-positive-only --plot-real-only
--plot-tight-y --seqargs=20,3.5 --plot gibbs

[-0.0311925   0.15173154 -0.42846816  0.59591
 -0.41539264 -0.16200557  0.12112529 -0.09068282
 -0.92154011 -0.88213549  1.0199175  -0.88213549
  2.22005254 -0.09068282  0.12112529 -0.16200557
 -0.41539264  0.59591     -0.42846816 -2.98986111
  1.53960383]
```

The corresponding response function is shown in Fig. 30.

10. ReLU

The "rectified linear unit" activation function, $\text{ReLU}(x) := \max(0, x)$, is popular in machine learning and QSP is a natural framework to employ for realizing such nonlinear activation functions for quantum machine learning. A
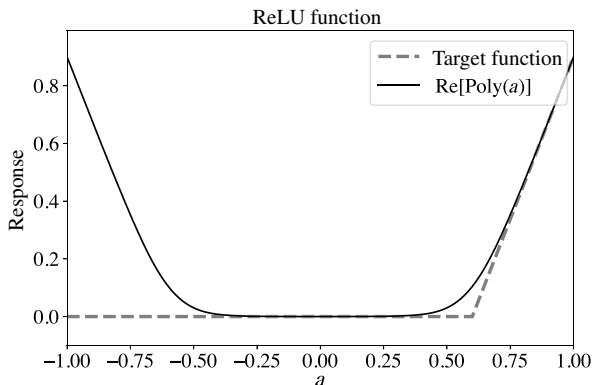

FIG. 31. The response function for the polynomial approximation to the ReLU function with $\delta = 0.6$ and $\Delta = 15$.

common differentiable approximation of the ReLU function is the SOFTPLUS function, which is made into an even function in this version:

$$f(a) = \frac{\ln\left(1 + e^{\Delta(|a|-\delta)}\right)}{\Delta},\qquad \text{(D9)}$$

where Δ parametrizes the steepness and δ the offset of the threshold from 0.

For example, with $\delta = 0.6$ and $\Delta = 15$ and using a degree-20 Taylor series, a set of QSP phase angles for this polynomial is as follows:

```
pyqsp --plot-real-only --plot-tight-y
--seqargs=20,0.6,15 --plot relu

[0.5830891  -0.50867554  0.45797704 -1.83149903
 0.20084092 -0.11936587  0.97960177  0.53415547
-0.9957325  -0.9362886   1.24987957 -0.9362886
-0.9957325   0.53415547  0.97960177 -0.11936587
 0.20084092  1.31009363 -2.68361561  2.63291712
-0.98770723]
```

The corresponding response function is shown in Fig. 31.

[1] C. H. Bennett, Time/space trade-offs for reversible computation, SIAM J. Computing **18**, 766 (1989).

[2] E. Fredkin and T. Toffoli, Conservative logic, Int. J. Theor. Phys. **21**, 219 (1982).

[3] P. W. Shor, in *Proceedings 35th annual symposium on foundations of computer science* (IEEE, 1994), p. 124.

[4] L. K. Grover, in *Proceedings of the twenty-eighth annual ACM Symposium on Theory of Computing* (1996), p. 212, ArXiv:quant-ph/9605043.

[5] R. P. Feynman, Simulating physics with computers, Int. J. Theor. Phys **21**, 467 (1982).

[6] S. Lloyd, Universal quantum simulators, Science **273**, 1073 (1996).

[7] G. H. Low, T. J. Yoder, and I. L. Chuang, Methodology of Resonant Equiangular Composite Quantum Gates, Phys. Rev. X **6**, 041067 (2016). ArXiv:1603.03996

[8] G. H. Low and I. L. Chuang, Hamiltonian simulation by qubitization, Quantum 3, 163 (2019). ArXiv:1610.06546

[9] R. Freeman and M. J. Minch, *Spin Choreography: Basic Steps in High Resolution NMR* (Oxford University Press, Oxford, 1998).

[10] M. H. Levitt, Composite pulses, Prog. Nucl. Magn. Reson. Spectrosc. **18**, 61 (1986).

[11] S. Wimperis, Broadband, narrowband, and passband composite pulses for use in advanced NMR experiments, J. Magn. Reson., Ser. A **109**, 221 (1994).

[12] A. Gilyén, Y. Su, G. H. Low, and N. Wiebe, Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics, Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing (2019), ArXiv:1806.01838.

[13] S. Lloyd, B. T. Kiani, D. R. Arvidsson-Shukur, S. Bosch, G. De Palma, W. M. Kaminsky, Z.-W. Liu, and M. Marvian,

Hamiltonian singular value transformation and inverse block encoding (2021), ArXiv:2104.01410.

[14] Y. Dong, X. Meng, K. B. Whaley, and L. Lin, Efficient phase factor evaluation in quantum signal processing (2020), ArXiv:2002.11649.

[15] A. W. Harrow, A. Hassidim, and S. Lloyd, Quantum Algorithm for Linear Systems of Equations, Phys. Rev. Lett. **103**, 150502 (2009). ArXiv:0811.3171

[16] G. Brassard, P. Høyer, and A. Tapp, Quantum counting, Lecture Notes in Computer Science, 820 (1998), ArXiv:quant-ph/9805082.

[17] P. Rall, Faster coherent quantum algorithms for phase, energy, and amplitude estimation (2021), ArXiv:2103.09717.

[18] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, UK, 2010).

[19] J. Preskill, *Lecture notes for physics 229: Quantum information and computation* (1998).

[20] A. Y. Kitaev, A. Shen, M. N. Vyalyi, and M. N. Vyalyi, *Classical and quantum computation*, number 47 (American Mathematical Soc., 2002).

[21] G. H. Low, *Quantum signal processing by single-qubit dynamics*, Ph.D. thesis, Department of Physics, Massachusetts Institute of Technology (2017).

[22] A. Gilyén, *Quantum singular value transformation & its algorithmic applications*, Ph.D. thesis, Institute for Logic, Language and Computation, University of Amsterdam: Institute for Logic, Language and Computation (2019).

[23] G. H. Low and I. L. Chuang, Optimal Hamiltonian Simulation by Quantum Signal Processing, Phys. Rev. Lett. **118**, 010501 (2017). ArXiv:1606.02685

[24] The convention presented in Eq. (3) may be called the Wx convention. For other conventions, see Appendix A.

[25] L. M. K. Vandersypen and I. L. Chuang, NMR techniques for quantum control and computation, Rev. Mod. Phys. **76**, 1037 (2005). arXiv:quant-ph/0404064

[26] G. Wolfowicz and J. J. Morton, Pulse techniques for quantum information processing, eMagRes **5**, 1515 (2016).

[27] X. Wang, L. S. Bishop, J. Kestner, E. Barnes, K. Sun, and S. D. Sarma, Composite pulses for robust universal control of singlet-triplet qubits, Nat. Commun. **3**, 1 (2012).

[28] C. D. Aiello, M. Hirose, and P. Cappellaro, Composite-pulse magnetometry with a solid-state quantum sensor, Nat. Commun. **4**, 1 (2013).

[29] E. Mount, C. Kabytayev, S. Crain, R. Harper, S.-Y. Baek, G. Vrijsen, S. T. Flammia, K. R. Brown, P. Maunz, and J. Kim, Error compensation of single-qubit gates in a surface-electrode ion trap using composite pulses, Phys. Rev. A **92**, 060301(R) (2015). arXiv:1504.01440

[30] Y. Tomita, J. T. Merrill, and K. R. Brown, Multi-qubit compensation sequences, New J. Phys. **12**, 015002 (2010). arXiv:0908.2593

[31] S. S. Ivanov and N. V. Vitanov, High-fidelity local addressing of trapped ions and atoms by composite sequences of laser pulses, Opt. Lett. **36**, 1275 (2011). ArXiv:1106.0272

[32] J. M. Chow, J. M. Gambetta, A. W. Cross, S. T. Merkel, C. Rigetti, and M. Steffen, Microwave-activated conditional-phase gate for superconducting qubits, New J. Phys. **15**, 115012 (2013). ArXiv:1307.2594

[33] J. Haah, Product decomposition of periodic functions in quantum signal processing Quantum 3, 190 (2019). ArXiv:1806.10236

[34] R. Chao, D. Ding, A. Gilyén, C. Huang, and M. Szegedy, Finding angles for quantum signal processing with machine precision (2020), ArXiv:2003.02831.

[35] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. Brandao, and D. A. Buell *et al.*, Quantum supremacy using a programmable superconducting processor, Nature **574**, 505 (2019).

[36] L. Lin, Optimization based approach for quantum signal processing and its energy landscape (2021), Simons Institute for the Theory of Computing: Quantum Wave in Computing Reunion.

[37] T. J. Yoder, G. H. Low, and I. L. Chuang, Fixed-Point Quantum Search with an Optimal Number of Queries, Phys. Rev. Lett. 113, 210501 (2014). ArXiv:1409.3305

[38] Note that even for degenerate eigenspaces, the linearity of the QSP sequence ensures qubitization.

[39] R. A. Horn and C. R. Johnson, *Topics in Matrix Analysis* (Cambridge University Press, Cambridge, UK, 1991).

[40] W. J. Huggins, J. Lee, U. Baek, B. O'Gorman, and K. B. Whaley, A non-orthogonal variational quantum eigensolver, New J. Phys. 22, 073009 (2020). ArXiv:1909.09114

[41] L. Lin and Y. Tong, Optimal polynomial based quantum eigenstate filtering with application to solving quantum linear systems, Quantum 4, 361 (2020). ArXiv:1910.14596

[42] L. Lin and Y. Tong, Near-Optimal ground state preparation, Quantum 4, 372 (2020). ArXiv:2002.12508

[43] A. Y. Kitaev, Quantum measurements and the Abelian stabilizer problem (1995), ArXiv:quant-ph/9511026.

[44] S. Parker and M. B. Plenio, Efficient Factorization with a Single Pure Qubit and log *n* Mixed Qubits, Phys. Rev. Lett. 85, 3049 (2000). ArXiv:quant-ph/0001066

[45] T. Monz, D. Nigg, E. A. Martinez, M. F. Brandl, P. Schindler, R. Rines, S. X. Wang, I. L. Chuang, and R. Blatt, Realization of a scalable Shor algorithm, Science 351, 1068 (2016). ArXiv:1507.08852

[46] The sign function is discontinuous and cannot be implemented exactly through the QSVT. We deal with this caveat shortly, but for now we assume it can be implemented exactly.

[47] K. M. Svore, M. B. Hastings, and M. Freedman, Faster phase estimation (2013), ArXiv:1304.0741.

[48] N. Wiebe and C. Granade, Efficient Bayesian Phase Estimation, Phys. Rev. Lett. 117, 010503 (2016). ArXiv:1508.00869

[49] Note that we do not include an additional $(m + 1)^{\text{th}}$ phase-readout qubit for the bit in the ones place of φ, because no approximation is needed if we know that $m = 3$.

[50] A. M. Childs, R. Kothari, and R. D. Somma, Quantum algorithm for systems of linear equations with exponentially improved dependence on precision, SIAM J. Computing 46, 1920 (2017). ArXiv:1511.02306

[51] A. Peruzzo, J. McClean, P. Shadbolt, M.-H. Yung, X.-Q. Zhou, P. J. Love, A. Aspuru-Guzik, and J. L. O'Brien,

A variational eigenvalue solver on a photonic quantum processor, Nat. Commun. **5**, 1 (2014).

[52] E. Farhi, J. Goldstone, and S. Gutmann, A quantum approximate optimization algorithm (2014), ArXiv:1411. 4028.

[53] N.-H. Chia, A. Gilyén, T. Li, H.-H. Lin, E. Tang, and C. Wang, Sampling-based sublinear low-rank matrix arithmetic framework for dequantizing quantum machine learn-ing, Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing (2020), ArXiv:1910. 06151.

[54] C. T. Hann, G. Lee, S. Girvin, and L. Jiang, Resilience of quantum random access memory to generic noise, PRX Quantum 2, 020311 (2021). ArXiv:2012. 05340

[55] https://github.com/ichuang/pyqsp.