Generalized Quantum Signal Processing

Danial Motlagh*
Department of Computer Science, University of Toronto, Toronto Ontario, Canada

Nathan Wiebe
Department of Computer Science, University of Toronto, Toronto Ontario, Canada
Pacific Northwest National Laboratory, Richland Washington, USA
and Canadian Institute for Advanced Research, Toronto Ontario, Canada

(Received 21 January 2024; accepted 30 May 2024; published 28 June 2024)

Quantum signal processing (QSP) and quantum singular value transformation (QSVT) currently stand as the most efficient techniques for implementing functions of block-encoded matrices, a central task that lies at the heart of most prominent quantum algorithms. However, current QSP approaches face several challenges, such as the restrictions imposed on the family of achievable polynomials and the difficulty of calculating the required phase angles for specific transformations. In this paper, we present a generalized quantum signal processing (GQSP) approach, employing general SU(2) rotations as our signal-processing operators, rather than relying solely on rotations in a single basis. Our approach lifts all practical restrictions on the family of achievable transformations, with the sole remaining condition being that $|P| \leq 1$, a restriction necessary due to the unitary nature of quantum computation. Furthermore, GQSP provides a straightforward recursive formula for determining the rotation angles needed to construct the polynomials in cases where P and Q are known. In cases where only P is known, we provide an efficient optimization algorithm capable of identifying in under a minute of GPU time, a corresponding Q for polynomials of degree on the order of 10^7. We further illustrate GQSP simplifies QSP-based strategies for Hamiltonian simulation, offer an optimal solution to the ϵ-approximate fractional query problem that requires $\mathcal{O}((1/\delta) + \log(1/\epsilon))$ queries to perform where $\mathcal{O}(1/\delta)$ is a proved lower bound, and introduces novel approaches for implementing bosonic operators. Moreover, we propose a novel framework for the implementation of normal matrices, demonstrating its applicability through synthesis of diagonal matrices, as well as the development of a new algorithm for convolution through synthesis of circulant matrices using only $\mathcal{O}(d \log N + \log^2 N)$ 1 and 2-qubit gates for a filter of lengths d.

DOI: 10.1103/PRXQuantum.5.020368

I. INTRODUCTION

In recent years, significant advancements in quantum algorithm theory have revealed a powerful, overarching insight: the majority of prominent quantum algorithms, such as Hamiltonian simulation [1–4], quantum search [5–7], factoring [8], and quantum walks [7,9], can be fundamentally reduced to the central task of implementing a matrix function of a Hamiltonian, $f(H)$, this was shown in Refs. [10,11]. Over the years, numerous techniques have been developed for constructing such functions, including phase-estimation methods such as the HHL algorithm [12], linear combination of unitaries (LCU) [4], and quantum signal processing (QSP) [2,13–17]. Among these, QSP stands out as the most versatile approach to date, capable of approximating a wide range of matrix functions through eigenvalue or singular value transformations of H, while requiring a minimal number of ancilla qubits. The basic idea of QSP is to build a polynomial approximation of the desired function by assuming oracular access to a unitary U, which encodes H. QSP has been demonstrated to yield asymptotically optimal Hamiltonian simulation algorithms [2]. QSP's applicability was then further extended to the case of nonsquare matrices by the QSVT technique [11]. Together, QSP and QSVT have provided an abstract formalism that facilitates the efficient implementation of a wide range of linear algebraic operations and transformations, creating a new language with greater expressivity for the construction of quantum algorithms.

Despite its success, QSP still suffers from a number of severe limitations. Most notable of them being the restrictions it places on the family of polynomials that can be built using it. For instance, in order to build an arbitrary

*Contact author: dani.mhn.2001@gmail.com

polynomial using the original QSP one has to split the polynomial into four parts by first breaking the polynomial into its real and imaginary parts, and then further breaking down each of those into their respective even and odd components. One would then combine all parts together using linear combination of unitaries (LCU) [4], and furthermore perform a variant of amplitude amplification [18] on the resulting circuit. This requires that we triple the number of operations needed to perform Hamiltonian simulation relative to what we would need if we could implement Hamiltonian dynamics directly. Lastly, while the original QSP guarantees the existence of a set of parameters leading to polynomials satisfying the specified restrictions, finding those parameters has proved to be a complicated (if computationally efficient) task [14,19]. In this paper, we present an algorithm, which we term generalized quantum signal processing (GQSP), that overcomes all the above limitations leading to a more general framework for the central task of implementing functions of Hamiltonians.

Our GQSP algorithm provides us with the ability to build polynomials of unitary matrices using a single ancilla qubit with the only restriction being that its norm is not greater than 1 on the complex unit circle as demonstrated in Corollary 5. Since GQSP does not restrict us to a fixed parity for the polynomial, it gives us the ability to approximate functions of a Hamiltonian H without the need for LCU that appears in some applications of QSP [11]. Furthermore, in Sec. IV we demonstrate that computing the rotation angles required to build a given polynomial is much more efficient and conceptually simpler within this framework. This extends the scope of QSP along two different axes. We then utilize this technique in Sec. V to develop a conceptually simplified formulation of qubitization-based Hamiltonian simulation, and give a provably optimal algorithm for the fractional query problem as special cases of what we call phase functions. In Sec. VI we introduce a powerful idea that extends the concept of Fourier decomposition to normal matrices using polynomials of a special unitary matrix. More precisely, we will demonstrate that any normal matrix can be written as a polynomial of the root of unity unitary in its eigenbasis. We then explore the utility of this idea by giving a general framework for synthesizing diagonal and convolution matrices. But first, we give a review of the original quantum signal-processing algorithm in Sec. II, which sets the stage for our method introduced in Sec. III.

II. REVIEW OF QUANTUM SIGNAL PROCESSING

The quantum signal-processing algorithm constructs a function of $\mathcal{H}$ by interleaving applications of a signal operator with signal-processing operators (Fig. 1). The signal operator encodes $\mathcal{H}$ using controlled applications of $U = e^{i\mathcal{H}}$, where an ancillary qubit acts as the control. The signal-processing operations, on the other hand, consist of single-qubit operations performed on the ancillary qubit. There are two conventions that are commonly used for the formulation of quantum signal processing. One where the signal operator is expressed in the σ_x basis and the signal processing is done in the σ_z basis, and another where the signal operator is expressed in the σ_z basis and the signal processing is done in the σ_x basis. The signal operator in the σ_z basis is defined as

$$A_{\sigma_z} := \begin{bmatrix} e^{i\mathcal{H}} & 0 \\ 0 & e^{-i\mathcal{H}} \end{bmatrix}. \tag{1}$$

We can express the signal operator in the σ_x basis by applying a Hadamard gate to both sides of the ancilla qubit:

$$A_{\sigma_x} := \begin{bmatrix} \cos(\mathcal{H}) & i\sin(\mathcal{H}) \\ i\sin(\mathcal{H}) & \cos(\mathcal{H}) \end{bmatrix}. \tag{2}$$

This shows that, at a high level, both of these formalisms are conceptually identical although there are distinctions between the forms of the polynomials that can be constructed in both bases. The results of QSP are often stated in the A_{σ_x} formalism as follows.

Theorem 1 (Quantum signal processing in σ_x basis). $\forall d \in \mathbb{N}$ and $\vec{\phi} \in \mathbb{R}^{d+1}$:

$$\begin{bmatrix} P(\cos(\mathcal{H})) & -Q(\cos(\mathcal{H}))^\dagger \sin(\mathcal{H}) \\ Q(\cos(\mathcal{H}))\sin(\mathcal{H}) & P(\cos(\mathcal{H}))^\dagger \end{bmatrix} = \left(\prod_{j=1}^{d} R_z(\phi_j) A_{\sigma_x} \right) R_z(\phi_0)$$
$$\Longleftrightarrow$$

(1) $P, Q \in \mathbb{C}[x]$ such that $\deg(P) \leq d$ and $\deg(Q) \leq d-1$.
(2) $\text{Parity}(P) = d \mod 2$ and $\text{parity}(Q) = (d-1) \mod 2$.
(3) $\forall x \in [-1,1], |P(x)|^2 + (1-x^2)|Q(x)|^2 = 1$.

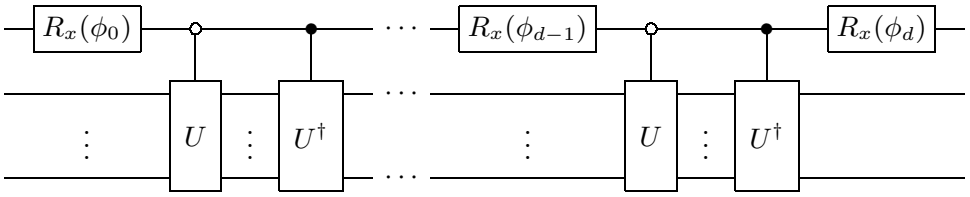

FIG. 1. Quantum circuit for the original QSP approach where the angles ϕ are chosen to enact a polynomial transformation of U.

In practice, one often only cares about building a particular polynomial P, so the question becomes, for what Ps does a Q exist such that they satisfy the conditions of Theorem 1? It turns out the restrictions above can be quite limiting. For example, the third condition requires $|P(\pm 1)| = 1$. This limitation can be overcome by applying Hadamards on the ancilla before and after the QSP sequence, effectively converting into the A_{σ_z} formalism. In this case, it can be shown that we can implement any real polynomial with parity d mod 2 such that $\deg(P) \leq d$, and the following.

Theorem 2 (Quantum signal processing in σ_z basis). For a polynomial P, if

(1) $P \in \mathbb{R}[x]$ such that $\deg(P) \leq d$.
(2) $\text{Parity}(P) = d \mod 2$.
(3) $\forall x \in \mathbb{R}, |P(e^{ix})|^2 \leq 1$.

Then $\exists \vec{\phi} \in \mathbb{R}^{d+1}$ such that,

$$\begin{bmatrix} P(e^{i\mathcal{H}}) & -Q(e^{i\mathcal{H}})^\dagger \\ Q(e^{i\mathcal{H}}) & P(e^{i\mathcal{H}})^\dagger \end{bmatrix} = \left(\prod_{j=1}^{d} R_x(\phi_j)\, A_{\sigma_z} \right) R_x(\phi_0).$$

The above formalism indeed mitigates some previously discussed limitations. However, the requirement that the polynomial P must be real and possess definite parity necessitates the combination of four separate instances of QSP using linear combination of unitaries (LCU) to construct an arbitrary complex polynomial with indefinite parity. In the following section, we address these constraints with a novel approach. We propose a more generalized formulation of quantum signal processing by allowing our signal processing operators to be arbitrary SU(2) rotations, effectively lifting the "realness" condition on the polynomials. Furthermore, we eliminate the parity restriction by simplifying our signal operator. Combined together, our reformulation successfully eliminates all practical restrictions on P, which we will elaborate on and demonstrate in the next section.

III. GENERALIZED QUANTUM SIGNAL PROCESSING

Our generalized quantum signal processing approach seeks to remove the limitations inherent in traditional QSP, thereby providing a more powerful mechanism for constructing functions of Hamiltonians. Our approach generalizes QSP by using SU(2) rotations rather than X rotations on the control ancilla similar to Ref. [20,21]; however, our approach eschews the need to use $U^\dagger$ and avoids the use of Laurent polynomials in the analysis. Specifically, we use the convention that the signal operator is a 0-controlled application of $U = e^{i\mathcal{H}}$:

$$A = \begin{bmatrix} U & 0 \\ 0 & \mathbb{I} \end{bmatrix}. \tag{3}$$

Next, instead of performing rotations in a single basis, we allow for signal-processing operations to be arbitrary SU(2) rotations of the ancillary qubit:

$$R(\theta, \phi, \lambda) = \begin{bmatrix} e^{i(\lambda+\phi)} \cos(\theta) & e^{i\phi} \sin(\theta) \\ e^{i\lambda} \sin(\theta) & -\cos(\theta) \end{bmatrix} \otimes \mathbb{I}. \tag{4}$$

Then by interleaving the above two operations as demonstrated in Fig. 2, we can block-encode polynomial transformations of the unitary matrix U without further assumptions. Note that below we have no assumptions made on the parity of the polynomial transformations

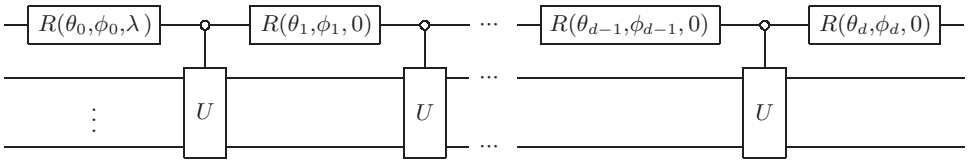

FIG. 2. Quantum circuit for GQSP wherein the angles θ and ϕ are chosen to enact a polynomial transformation of U. Here $R(\theta, \phi, \lambda)$ is a general single qubit rotation as specified in Eq. (4).

unlike those made by Theorem 1. Lastly, notice that we only need to specify a single λ since we can absorb the other instances into the instance of ϕ before them.

Theorem 3 (Generalized quantum signal processing). $\forall d \in \mathbb{N}, \exists \vec{\theta}, \vec{\phi} \in \mathbb{R}^{d+1}, \lambda \in \mathbb{R}$ such that,

$$\begin{bmatrix} P(U) & . \\ Q(U) & . \end{bmatrix} = \left(\prod_{j=1}^{d} R(\theta_j, \phi_j, 0) A \right) R(\theta_0, \phi_0, \lambda)$$
$$\iff$$

(1) $P, Q \in \mathbb{C}[x]$ and $\deg(P), \deg(Q) \leq d$.
(2) $\forall x \in \mathbb{R}, |P(e^{ix})|^2 + |Q(e^{ix})|^2 = 1$.

Proof. Both directions are proven by induction on d. To show the forward direction, we need to show inductively that for all $\{\theta_j\}$ and $\{\phi_j\}$ and d both of the claimed restrictions are met. The base case of $d = 0$ is trivial since

$$R(\theta_0, \phi_0, \lambda) = \begin{bmatrix} e^{i(\lambda+\phi)} \cos(\theta)\mathbb{I} & e^{i\phi} \sin(\theta)\mathbb{I} \\ e^{i\lambda} \sin(\theta)\mathbb{I} & -\cos(\theta)\mathbb{I} \end{bmatrix}. \quad (5)$$

Next let $d \in \mathbb{N}_{>0}$ and assume the induction hypothesis holds for $d-1$ producing $\hat{P}(U)$ and $\hat{Q}(U)$. Then,

$$\begin{bmatrix} P(U) & . \\ Q(U) & . \end{bmatrix} = \begin{bmatrix} e^{i\phi} \cos(\theta) U & e^{i\phi} \sin(\theta)\mathbb{I} \\ \sin(\theta) U & -\cos(\theta)\mathbb{I} \end{bmatrix} \begin{bmatrix} \hat{P}(U) & . \\ \hat{Q}(U) & . \end{bmatrix}. \quad (6)$$

Thus,

$$P(U) = e^{i\phi}(\cos(\theta) U \hat{P}(U) + \sin(\theta)\hat{Q}(U)), \quad (7)$$

$$Q(U) = \sin(\theta) U \hat{P}(U) - \cos(\theta)\hat{Q}(U). \quad (8)$$

Since $\hat{P}$ and $\hat{Q}$ have degree $\leq d-1$, then P and Q must have degree $\leq d$. Furthermore, given the fact that all operations are unitary, property (2) trivially holds. This completes the proof for the forward direction.

The proof for the reverse direction involves showing by induction that any pair of polynomials satisfying the conditions of Theorem 3 can be constructed through an appropriate choice of rotation angles for the single-qubit unitaries in Fig. 2. The base case of $d = 0$ is again trivial since P and Q would both be constants whose norm squared adds to 1. Any such pair can be written as $P = e^{i(\lambda+\phi)} \cos(\theta), Q = e^{i\phi} \sin(\theta)$ and implemented by $R(\theta_0, \phi_0, \lambda)$. Next let $d \in \mathbb{N}_{>0}$, assume P and Q satisfy conditions (1) and (2) from Theorem 3, and the induction hypothesis holds for $d-1$. Specifically, we assume from our induction hypothesis that for any $\hat{P}$ and $\hat{Q}$ of degree at most $d-1$ satisfying $|\hat{P}(e^{ix})|^2 + |\hat{Q}(e^{ix})|^2 = 1$, there exists $\vec{\theta}, \vec{\phi} \in \mathbb{R}^d, \lambda \in \mathbb{R}$ such that

$$\begin{bmatrix} \hat{P}(U) & . \\ \hat{Q}(U) & . \end{bmatrix} = \left(\prod_{j=1}^{d-1} R(\theta_j, \phi_j, 0) A \right) R(\theta_0, \phi_0, \lambda). \quad (9)$$

Thus, it suffices to find θ_d and ϕ_d such that $\hat{P}$ and $\hat{Q}$ are of degree at most $d-1$ satisfying $|\hat{P}(e^{ix})|^2 + |\hat{Q}(e^{ix})|^2 = 1$, where $\hat{P}$ and $\hat{Q}$ are given by

$$\begin{bmatrix} \hat{P}(U) & . \\ \hat{Q}(U) & . \end{bmatrix} = A^\dagger R(\theta_d, \phi_d, 0)^\dagger \begin{bmatrix} P(U) & . \\ Q(U) & . \end{bmatrix}. \quad (10)$$

Then,

$$\hat{P}(U) = e^{-i\phi_d} \cos(\theta_d) U^\dagger P(U) + \sin(\theta_d) U^\dagger Q(U), \quad (11)$$

$$\hat{Q}(U) = e^{-i\phi_d} \sin(\theta_d) P(U) - \cos(\theta_d) Q(U). \quad (12)$$

First notice that regardless of our choice of θ_d and ϕ_d, we have

$$\begin{aligned} \|\hat{P}(e^{ix})\|^2 + \|\hat{Q}(e^{ix})\|^2 &= \hat{P}(e^{ix})\hat{P}^*(e^{ix}) + \hat{Q}(e^{ix})\hat{Q}^*(e^{ix}) \\ &= (\cos^2(\theta_d) + \sin^2(\theta_d))\|P(e^{ix})\|^2 \\ &\quad + (\cos^2(\theta_d) + \sin^2(\theta_d))\|Q(e^{ix})\|^2 \\ &= \|P(e^{ix})\|^2 + \|Q(e^{ix})\|^2 = 1. \end{aligned} \quad (13)$$

Hence, the norm condition is again satisfied trivially by the unitary nature of our operations. Thus, it remains to show $\hat{P}$ and $\hat{Q}$ are of degree at most $d-1$.

Let a_i and b_i be the coefficients of the i-degree term in P and Q, respectively. By our assumption, we have that the target polynomials P and Q satisfy the norm condition $\|P(e^{ix})\|^2 + \|Q(e^{ix})\|^2 = P(e^{ix})P^*(e^{ix}) + Q(e^{ix})Q^*(e^{ix}) = 1$:

$$\left(\sum_{n=0}^{d} a_n e^{inx} \right) \left(\sum_{n=0}^{d} a_n^* e^{-inx} \right) + \left(\sum_{n=0}^{d} b_n e^{inx} \right) \left(\sum_{n=0}^{d} b_n^* e^{-inx} \right) = 1. \quad (14)$$

Next, we need to argue about the structure of the coefficients in this expansion. To begin we note that the inner product between the two parts from P satisfies

$$\left(\sum_{n=0}^{d} a_n e^{inx} \right) \left(\sum_{m=0}^{d} a_m^* e^{-imx} \right) = \|\vec{a}\|^2 + \sum_{n} \sum_{m \neq n} a_n a_m^* e^{i(n-m)x}. \quad (15)$$

Repeating the same expansion for the b coefficients leads to

$$1 = \|\vec{a}\|^2 + \|\vec{b}\|^2 + \sum_n \sum_{m\neq n} a_n a_m^* e^{i(n-m)x} + b_n b_m^* e^{i(n-m)x}. \tag{16}$$

We then note that the x-dependent sum must generically be zero because of the fact that the exponentials form an orthogonal function basis as seen by the Fourier transform as noted by the fact that

$$\frac{1}{2\pi}\int_{-\infty}^{\infty} e^{i(p-q)x}\mathrm{d}x = \delta(p-q), \tag{17}$$

where δ is the Dirac-δ function. Thus because the functions form an orthogonal function space that does not contain the constant function, we cannot form the constant function out of a linear combination of nonconstant exponentials, and the only way that we can satisfy Eq. (16) for all real-valued x is if

$$\sum_{n=0}^{d} (\|a_n\|^2 + \|b_n\|^2) = 1 \tag{18}$$

and

$$\forall i \in \mathbb{Z},\ i \neq 0, \quad \sum_n a_n a_{n-i}^* = -\sum_n b_n b_{n-i}^*. \tag{19}$$

Let $s, s' \geq 0$ be the smallest degrees present in P and Q, respectively, then notice that Eq. (19) implies $\deg(P) - s = \deg(Q) - s'$ since otherwise setting $i = \max(\deg(P) - s, \deg(Q) - s')$ would result in one side of Eq. (19) being 0 and the other being a nonzero value. It is easy to see when $\deg(P) > \deg(Q)$ or $\deg(P) < \deg(Q)$ setting θ_d equal to 0 or $\pi/2$, respectively, will result in the desired $\hat{P}$ and $\hat{Q}$. Therefore, without loss of generality, we assume $\deg(P) = \deg(Q) = d$, which also implies $s = s'$. Hence using Eq. (19) we get

$$a_d a_s^* = -b_d b_s^*. \tag{20}$$

Next, we choose $\theta_d = \tan^{-1}(|b_d|/|a_d|)$ and $\phi_d = \mathrm{Arg}(a_d/b_d)$. Then by Eq. (20)

$$e^{i\phi_d}\frac{\cos(\theta_d)}{\sin(\theta_d)} = \frac{a_d}{b_d} = -\frac{b_s^*}{a_s^*}. \tag{21}$$

Then it is easy to see after substituting our choices of θ_d and ϕ_d into Eqs. (11) and (12) the s-degree terms of P and Q in $\hat{P}$ cancel (no negative degree terms are introduced) and d-degree terms of P and Q in $\hat{Q}$ also cancel. Thus, $\deg(\hat{P}) = \deg(\hat{Q}) = d - 1$ as desired. ■

Theorem 3 specifies the necessary and sufficient conditions for constructible pairs of polynomials P and Q using our method. However, in practice, one often only cares about building a particular polynomial P, so the question becomes "For what Ps does a Q exist such that they satisfy the conditions of Theorem 3?"

In the following theorem we show that, as long as $|P|^2 \leq 1$ on the complex unit circle, there exists such a Q.

Theorem 4. $\forall P \in \mathbb{C}[x]$, we have

$$\forall x \in \mathbb{R}, \quad |P(e^{ix})|^2 \leq 1$$
$$\iff$$

$\exists Q \in \mathbb{C}[x]$ with $\deg(Q) = \deg(P)$ such that

$$\forall x \in \mathbb{R}, \quad |P(e^{ix})|^2 + |Q(e^{ix})|^2 = 1.$$

Before proving Theorem 4, notice that this leads to the following corollary, which states that we can build any arbitrary (appropriately scaled) polynomial P of U using our technique.

Corollary 5. $\forall P \in \mathbb{C}[x]$, with $\deg(P) = d$ if

$$\forall x \in \mathbb{R}, \quad |P(e^{ix})|^2 \leq 1.$$

Then $\exists \vec{\theta}, \vec{\phi} \in \mathbb{R}^{d+1}, \exists \lambda \in \mathbb{R}$ such that

$$\begin{bmatrix} P(U) & \cdot \\ \cdot & \cdot \end{bmatrix} = \left(\prod_{j=1}^{d} R(\theta_j, \phi_j, 0)A\right) R(\theta_0, \phi_0, \lambda).$$

Corollary 5 is one of the most significant results in this paper, as it clearly establishes the superior expressivity of our method over traditional QSP when contrasted with Theorem 2.

Proof of Theorem 4. $\Longrightarrow$: Let $P \in \mathbb{C}[x]$ with $|P|^2 \leq 1$ on $\mathbb{T}$, and define

$$T(\theta) = |P(e^{i\theta})|^2. \tag{22}$$

Then T is a non-negative trigonometric polynomial with terms from $-\deg(P)$ to $\deg(P)$ that satisfies $T \leq 1$. Define $H = 1 - T$ as another non-negative trigonometric polynomial of degree $\deg(P)$. It remains to show there exists Q a polynomial of degree $\deg(P)$ such that $H(\theta) = |Q(e^{i\theta})|^2$.

Let $d = \deg(P)$, then we can rewrite H as

$$H(\theta) = e^{-id\theta} R(e^{i\theta}). \tag{23}$$

For some polynomial R of degree $2d$. Since $H = H^*$, we have for all $|z| = 1$:

$$z^{2d} R^*(z) = z^{2d} R^*\left(\frac{1}{z^*}\right) = R(z). \tag{24}$$

Using the identity theorem for analytic functions, this equality must hold for all $z \in \mathbb{C} - \{0\}$, making R a self-inversive polynomial. Therefore, roots of R are either on the unit circle, or grouped in pairs $(w_i, (1/w_i^*))$ for $|w_i| > 1$. Notice $(e^{i\theta} - w_i)(e^{i\theta} - (1/w_i^*)) = -(e^{i\theta}/w_i^*)\|e^{i\theta} - w_i\|^2$, so by grouping the roots $(w_i, (1/w_i^*))$ together, we get

$$R(e^{i\theta}) = c(-1)^m e^{im\theta} \prod_{k=1}^{m} \frac{1}{w_k^*} \prod_{k=1}^{m} \|e^{i\theta} - w_k\|^2 \times \prod_{k=1}^{2d-2m} (e^{i\theta} - e^{i\theta_k}). \tag{25}$$

Here m is the number of such pairs. We can then write

$$H(\theta) = e^{-id\theta} R(e^{i\theta}) = |G(e^{i\theta})|^2 \hat{H}(\theta), \tag{26}$$

where G is a polynomial of degree m and $\hat{H}$ is a nonnegative trigonometric polynomial with roots only on the unit circle, that maps the unit circle into $[0, \infty)$. Just like H, we can rewrite $\hat{H}$ as

$$\hat{H}(\theta) = e^{i(d-m)\theta} \hat{R}(e^{i\theta}). \tag{27}$$

For some polynomial $\hat{R}$ of degree $2d - 2m$. Extending $\hat{H}$ to an analytic function on a small annulus including the unit circle, we can use the local form of the analytic function near the zeros. This implies a curve passing through a zero of $\hat{R}$ is sent either to a curve through 0, or a line segment ending at 0 around a small neighborhood of the zero depending on the parity of the zero. However, since $\hat{H}$ has all roots on the unit circle, then the unit circle itself passes through all zeros and is mapped into $[0, \infty)$, which implies all zeros have even multiplicity. Hence, we can rewrite $\hat{R} = \hat{G}^2$ for some polynomial $\hat{G}$ of degree $d - m$. Then on the complex unit circle, we have

$$\hat{H}(\theta) = |\hat{H}(\theta)| = |e^{i(d-m)} \hat{R}(e^{i\theta})| = |\hat{G}(e^{i\theta})|^2. \tag{28}$$

Thus,

$$H(\theta) = |G(e^{i\theta})|^2 \cdot |\hat{G}(e^{i\theta})|^2 = |(G\hat{G})(e^{i\theta})|^2. \tag{29}$$

Then letting $Q = G\hat{G}$ we have

$$H(\theta) = |Q(e^{i\theta})|^2 \Longrightarrow |P(e^{i\theta})|^2 + |Q(e^{i\theta})|^2 = 1 \tag{30}$$

and $\deg(Q) = \deg(G) + \deg(\hat{G}) = m + d - m = \deg(P)$, which proves the forward direction of the proof. The other direction is trivial since $|Q(e^{i\theta})|^2 \geq 0$. ■

So far we have only discussed the construction of polynomials of positive degree, however, the framework presented here can also be used to implement polynomials with any ratio of positive and negative degrees by making use of a secondary signal operator, which we will define as

$$A' = (|0\rangle\langle 0| \otimes I) + (|1\rangle\langle 1| \otimes U^\dagger) = \begin{bmatrix} \mathbb{I} & 0 \\ 0 & U^\dagger \end{bmatrix}. \tag{31}$$

Particularly, if the set of rotation angles $\vec{\theta}, \vec{\phi}$, and λ lead to the polynomials $P(x) = \sum_{n=0}^{d} a_n e^{inx}$ and $Q(x) = \sum_{n=0}^{d} b_n e^{inx}$, using our original signal operator A, then we can construct polynomials $P'(x) = e^{-ikx} P(x) = \sum_{n=-k}^{d-k} a_{n+k} e^{inx}$ and $Q'(x) = e^{-ikx} Q(x) = \sum_{n=-k}^{d-k} b_{n+k} e^{inx}$ for $k \leq d$, using the same set of rotation angles by replacing any k instances of A with the complementary signal operator A'.

Theorem 6 (Polynomials with negative powers). $\forall d, k \in \mathbb{N}, \forall \vec{\theta}, \vec{\phi} \in \mathbb{R}^{d+1}, \lambda \in \mathbb{R}$ and $k \leq d$ we have

$$\begin{bmatrix} P'(U) & . \\ Q'(U) & . \end{bmatrix} = \left[\prod_{j=1}^{k} R(\theta_{d-k+j}, \phi_{d-k+j}, 0) A'\right] \times \left[\prod_{j=1}^{d-k} R(\theta_j, \phi_j, 0) A\right] R(\theta_0, \phi_0, \lambda).$$

If and only if

$$\begin{bmatrix} P(U) & . \\ Q(U) & . \end{bmatrix} = \left[\prod_{j=1}^{d} R(\theta_j, \phi_j, 0) A\right] R(\theta_0, \phi_0, \lambda). \tag{32}$$

For $P'(e^{ix}) = e^{-ikx} P(e^{ix})$ and $Q'(e^{ix}) = e^{-ikx} Q(e^{ix})$

Proof of Theorem 6. First notice

$$A' = \left(\mathbb{I} \otimes U^\dagger\right) A. \tag{33}$$

Hence we have

$$\begin{aligned}\begin{bmatrix} P'(U) & . \\ Q'(U) & . \end{bmatrix} &= \left[\prod_{j=1}^{k} R(\theta_{d-k+j}, \phi_{d-k+j}, 0) A'\right] \\ &\times \left[\prod_{j=1}^{d-k} R(\theta_j, \phi_j, 0) A\right] R(\theta_0, \phi_0, \lambda) \\ &= \left[\prod_{j=1}^{k} R(\theta_{d-k+j}, \phi_{d-k+j}, 0) \left(\mathbb{I} \otimes U^{\dagger}\right) A\right] \\ &\times \left[\prod_{j=1}^{d-k} R(\theta_j, \phi_j, 0) A\right] R(\theta_0, \phi_0, \lambda). \quad (34)\end{aligned}$$

Since $\left(\mathbb{I} \otimes U^{\dagger}\right)$ commutes with both A and $R(\theta, \phi, \lambda)$ we then have

$$\begin{aligned}\begin{bmatrix} P'(U) & . \\ Q'(U) & . \end{bmatrix} &= \left(\mathbb{I} \otimes U^{-k}\right) \left[\prod_{j=1}^{d} R(\theta_j, \phi_j, 0) A\right] R(\theta_0, \phi_0, \lambda) \\ &= \left(\mathbb{I} \otimes U^{-k}\right) \begin{bmatrix} P(U) & . \\ Q(U) & . \end{bmatrix}. \quad (35)\end{aligned}$$

■

In this section, we gave a detailed description of our generalized QSP technique and proved its first advantage over traditional QSP. Namely, we showed that our method overcomes the restrictions placed on the family of polynomials that can be built using QSP, which eliminates the need for LCU and oblivious amplitude amplification (OAA). In the next section, we will demonstrate the second advantage of our formalism over traditional QSP, which is, the computation of the rotation angles required to build a given polynomial is much more efficient and conceptually simpler within our framework.

IV. CALCULATING ROTATION ANGLES

In this section, we first provide a simple method to calculate the parameters of our algorithm given desired polynomials P and Q. This is a significant advance because existing approaches for computing the rotation angles in QSP are quite involved. This approach, on the other hand, is elementary and as a result, allows the approach in this paper to be more easily understood by students and more easily deployed in practice. We then provide an extremely efficient optimization algorithm for finding a corresponding Q given a desired P since in practice one is often only interested in implementing a particular polynomial P. We also provide astonishing numerical results that showcase the efficiency of our optimization method. Notably, we are able to find Q for randomly chosen polynomials P of up to $2^{24} \sim 16.8$ million degrees in less than 40 s on an A100 GPU. This is a remarkable advancement given that previous methods have only been able to find rotation angles for polynomials of up approximately 10^4 degrees [14].

Suppose P and Q are given to us as a $2 \times d$ matrix S where the first row contains the coefficients of P and the second row contains coefficients of Q. Now we will use the induction step in the proof of Theorem 3 to formulate a simple recursive algorithm by setting $\theta_d = \tan^{-1}(|b_d|/|a_d|)$ and $\phi_d = \mathrm{Arg}(a_d/b_d)$, calculating $\hat{P}$ and $\hat{Q}$, and calling the function recursively. Calculating $\hat{P}$ and $\hat{Q}$ in this matrix representation corresponds to multiplying S by $R(\theta_d, \phi_d, 0)$, and shifting the top row of the matrix, which would correspond to multiplying by $A^{\dagger}$ in our original formulation. The pseudocode for this algorithm is shown in Algorithm 1. As we can see, this algorithm is very simple and easy to understand, however, the algorithm assumes P and Q satisfy the requirements of Theorem 3, which is not a trivial task.

In Theorem 4 we showed that as long as $\forall t \in \mathbb{R}, |P(e^{it})|^2 \leq 1$, there exists a Q such that together with P they satisfy the conditions of Theorem 3. One way to find such a Q is through root finding by following the idea laid out in the proof of Theorem 4. However, root finding algorithms can be computationally expensive for polynomials of large degree, so here we lay out a simple and cheap optimization method for finding Q. Let $P(e^{it}) = \sum_{n=0}^{d} a_n e^{int}$, and $Q(e^{it}) = \sum_{n=0}^{d} b_n e^{int}$. In other words, P and Q are the discrete-time Fourier transformations of the coefficients $\{a_n\}_n$ and $\{b_n\}_n$. To be more rigorous P and Q are the Fourier transforms of impulse trains:

$$P(e^{it}) = \mathcal{F}\left\{\sum_{n=-\infty}^{\infty} a_n \cdot \delta\left(t - \frac{n}{2\pi}\right)\right\}, \quad (36)$$

ALGORITHM 1. Pseudocode for the computation of the phase factors in GQSP.

$a_d, b_d = S[0][d], S[1][d]$
$\theta_d = \tan^{-1}\left(\frac{|b_d|}{|a_d|}\right)$
$\phi_d = \mathrm{Arg}\left(\frac{a_d}{b_d}\right)$

if $d = 0$ **then**
  $\lambda = \mathrm{Arg}(b_d)$
  return $(\theta_0, \phi_0, \lambda)$
end if

$S = R(\theta_d, \phi_d, 0)^{\dagger} \cdot S$
$\hat{S} = [S[0][1:d], S[1][0:d-1]]$
$\vec{\theta}_{d-1}, \vec{\phi}_{d-1}, \lambda = \mathrm{ComputeParameters}(\hat{S}, d-1)$

return $(\vec{\theta}_{d-1}, \theta_d), (\vec{\phi}_{d-1}, \phi_d), \lambda$

$$Q(e^{it}) = \mathcal{F}\left\{\sum_{n=-\infty}^{\infty} b_n \cdot \delta\left(t - \frac{n}{2\pi}\right)\right\}. \tag{37}$$

Here $a_n = b_n = 0$ for $n > d$ or $n < 0$. The conditions of Theorem 3 also require

$$1 = P(e^{it})P^*(e^{it}) + Q(e^{it})Q^*(e^{it}). \tag{38}$$

Then taking the inverse Fourier transform of both sides of the above equation, and applying the convolution theorem we get

$$\begin{aligned}\delta = &\left(\sum_{n=-\infty}^{\infty} a_n \cdot \delta\left(t - \frac{n}{2\pi}\right)\right) \star \left(\sum_{n=-\infty}^{\infty} a_n^* \cdot \delta\left(t + \frac{n}{2\pi}\right)\right), \\ &+ \left(\sum_{n=-\infty}^{\infty} b_n \cdot \delta\left(t - \frac{n}{2\pi}\right)\right) \star \left(\sum_{n=-\infty}^{\infty} b_n^* \cdot \delta\left(t + \frac{n}{2\pi}\right)\right),\end{aligned} \tag{39}$$

where δ is the unit impulse, and $\star$ is the convolution operator. The above implies that if we are given P and Q as arrays of coefficients the following must hold:

$$\begin{aligned}&[a_0 \cdots a_d] \star [a_d^* \cdots a_0^*] + [b_0 \cdots b_d] \star [b_d^* \cdots b_0^*] \\ &\quad = [0 \cdots 0, 1, 0 \cdots 0].\end{aligned} \tag{40}$$

Here $[0 \cdots 0, 1, 0 \cdots 0]$ is an array of length $2d + 1$ with 1 at the center and d zeros on each side. Hence by letting $\vec{a} = [a_0, a_1, \ldots, a_d]$, $\vec{b} = [b_0, b_1, \ldots, b_d]$, and $\vec{\delta} = [0 \cdots 0, 1, 0 \cdots 0]$ we have

$$\vec{a} \star \text{reverse}(\vec{a})^* + \vec{b} \star \text{reverse}(\vec{b})^* = \vec{\delta}. \tag{41}$$

Here reverse$(\cdot)$ refers to the reversal of an array wherein the last element is first and vice versa. Therefore, we can set up our optimization problem as

$$\text{argmin}_{\vec{b}} \|\vec{a} \star \text{reverse}(\vec{a})^* + \vec{b} \star \text{reverse}(\vec{b})^* - \vec{\delta}\|^2. \tag{42}$$

The above objective function can be evaluated in time $\mathcal{O}(d \log d)$ using FFT-based convolution algorithms. This almost linear runtime of the objective function makes the optimization extremely efficient, as demonstrated in Fig. 3. Notably, it lets us find the coefficients of Q for random polynomials of up to degree $2^{24} \sim 1.68 \times 10^7$ in less than 40 s on an A100 GPU. This is a significant improvement over the degree 10 000 polynomials that can be achieved using existing QSP approaches via the state-of-the-art techniques of Ref. [14]. For completeness, we have provided our code (our code for finding the complementary polynomial Q is available at Ref. [22]) for the above optimization algorithm used to generate Fig. 3.

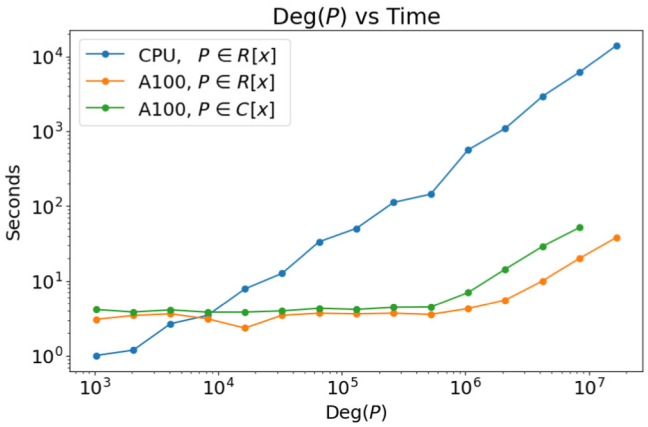

FIG. 3. Computational time required to complete the optimization as a function of the degree of the polynomial on CPU vs GPU. We can see that even on a CPU, we have an almost linear scaling of computational resources due to the $\mathcal{O}(n \log n)$ scaling of FFT-based convolution. Furthermore, the plot showcases the superior scalability of GPUs for the optimization problem.

V. PHASE FUNCTIONS:

A major application of quantum signal processing involves constructing phase functions of input unitaries. Specifically, these approaches give a way to implement a transformation of an input phase via

$$U = e^{iH} \Longrightarrow e^{if(H)}. \tag{43}$$

This sort of transformation is the root of the exponential improvements in accuracy found for the quantum linear system algorithm and also is the central idea behind fixed-point amplitude amplification as well as recent QSP-based approaches to error mitigation. The following theorem will provide us with the building blocks to achieve such transformations in the phase.

This idea is widely used within simulation and other areas [2,5,11]; however our approach provides a substantial simplification in cases where the Fourier series used (equivalently a Laurent polynomial in the phase [10]) is of mixed parity. We provide a host of examples of this reasoning below and show how our generalization of phase estimation can be used to simplify the solution to the following problems.

A. Application to simulation

The first and most obvious application of our framework is Hamiltonian simulation with the same query complexity as that of qubitization [3]. As a first step, we need to note that the function that we wish to implement is the exponential of a cosine. The complexity of this is given below

Theorem 7. Given a unitary matrix $U = e^{iH}$, we can implement an ϵ approximation of $e^{it \sin H}$ and $e^{it \cos H}$ for

$t \in \mathbb{R}$ using $\mathcal{O}(t + \log((1/\epsilon))/\log\log(1/\epsilon))$ controlled-U and 2-qubit operations while using a single ancillary qubit.

Proof. This can be easily implemented through the use of the Jacobi-Anger expansions:

$$e^{it\cos\theta} = \sum_{n=-\infty}^{\infty} i^n J_n(t) e^{in\theta}, \tag{44}$$

$$e^{it\sin\theta} = \sum_{n=-\infty}^{\infty} J_n(t) e^{in\theta}, \tag{45}$$

where $J_n(x)$ is the n^{th} Bessel function of the first kind. The coefficients of the infinite sum are known to decrease exponentially fast, specifically

$$J_n(t) \in \Theta\left(\frac{\left(\frac{t}{2}\right)^n}{n!}\right) \subseteq \mathcal{O}\left(\frac{\left(\frac{et}{2}\right)^n}{n^{n+1/2}}\right) \subseteq \mathcal{O}\left(\left(\frac{et}{2n}\right)^n\right). \tag{46}$$

Thus for any fixed $t \geq 0$ we can choose $n' \geq t$ in which case the function is exponentially decreasing for all $n \geq n'$. Then given that we choose n' to lead to exponential decay we can find for any $\epsilon > 0$ that there exists $n' \in \mathcal{O}(t + \log(1/\epsilon)/\log\log(1/\epsilon))$ such that for all $n \geq n'$ $J_n(t) \leq \epsilon$. Allowing us to build $e^{it\sin H}$ and $e^{it\cos H}$ with accuracy ϵ with a polynomial of only $\mathcal{O}(t + \log(1/\epsilon)/\log\log(1/\epsilon))$ degree. ■

Corollary 8. Let $H = \sum_j \alpha_j U_j$ be a Hermitian matrix where $\alpha_j \geq 0$ and $U_j \in U(2^n)$. Further, let $\texttt{PREPARE}: |0\rangle \mapsto \sum_j \sqrt{\alpha_j}|j\rangle/\alpha$ and $\texttt{SELECT}|j\rangle|\psi\rangle = |j\rangle U_j|\psi\rangle$ be unitary matrices in $U(2^m)$ and $U(2^{n+m})$ and finally let $W = -(1 - 2\texttt{PREPARE}|0\rangle\langle 0|\texttt{PREPARE}^\dagger)\texttt{SELECT}$. Under these assumptions we can, for any $t > 0$ and $\epsilon > 0$, construct a unitary matrix $V \in U(2^{n+m+1})$ that block encodes e^{-iHt} within error ϵ using $\mathcal{O}(\alpha t + \log(1/\epsilon)/\log\log(1/\epsilon))$ applications of W.

Proof. Existing work [2,11] has shown that for each eigenvector $|\lambda_j\rangle$ of H such that $H|\lambda_j\rangle = \lambda_j|\lambda_j\rangle$ there exists two eigenvectors of W that can be conveniently expressed within the span of $\texttt{PREPARE}|0\rangle|\lambda_j\rangle$ and $W\texttt{PREPARE}|0\rangle|\lambda_j\rangle$ such that we define the orthogonal component to $\texttt{PREPARE}|0\rangle|\lambda_j\rangle$ to be $|\phi_j\rangle$ and also define the action of W restricted to this two-dimensional subspace is W_{λ_j} (taking the former vector to correspond to $|0\rangle$ and the latter to $|1\rangle$)

$$W_{\lambda_j} = \begin{bmatrix} \frac{\lambda_j}{\alpha} & \sqrt{1-\lambda_j^2/\alpha^2} \\ -\sqrt{1-\lambda_j^2/\alpha^2} & \frac{\lambda_j}{\alpha} \end{bmatrix}. \tag{47}$$

Next note that for this unitary we have that the eigenvalues of this operation are $e^{\pm i\arccos(\lambda_j/\alpha)}$ and let us denote the eigenvectors of this to be $|\phi_j^{\pm}\rangle$. Then setting $A = |0\rangle\langle 0| \otimes W + |1\rangle\langle 1| \otimes I^{\otimes n}$ we can apply Corollary 5 to perform for any degree d polynomial P such that $\max \|P(U)\| \leq 1$

$$W_{\lambda_j} \mapsto \begin{bmatrix} P(W_{\lambda_j}) & \cdot \\ \cdot & \cdot \end{bmatrix}. \tag{48}$$

In our case we wish to choose $P(U) = e^{-i\alpha tH}$. We can then use the result of Theorem 7 to find an expansion within error ϵ using $\mathcal{O}(\alpha t + \log(1/\epsilon)/\log\log(1/\epsilon))$ queries to the oracle W. We can then see that by making this transformation we can map

$$\begin{aligned} W_{\lambda_j} &\mapsto \begin{bmatrix} e^{-i\alpha t\cos(\arccos(\lambda_j/\alpha))} & 0 \\ 0 & e^{-i\alpha t\cos(-\arccos(\lambda_j/\alpha))} \end{bmatrix} \\ &= \begin{bmatrix} e^{-i\lambda_j t} & 0 \\ 0 & e^{-i\lambda_j t} \end{bmatrix} := F_{\lambda_j}. \end{aligned} \tag{49}$$

Now let us consider the input vector $\texttt{PREPARE}|0\rangle|\lambda_j\rangle = a|\phi_j^+\rangle + b|\phi_j^-\rangle$. Then we have that the transformed block encoding obeys

$$\begin{aligned} &(\langle 0| \otimes I)F_{\lambda_j}(\texttt{PREPARE}|0\rangle|\lambda_j\rangle) \\ &\quad = ae^{-i\lambda_j t}|\phi_j^+\rangle + be^{-i\lambda_j t}|\phi_j^-\rangle \\ &\quad = e^{-i\lambda_j t}\texttt{PREPARE}|0\rangle|\lambda_j\rangle, \end{aligned} \tag{50}$$

which is logically equivalent under local isometries to the evolution of the system. As this operation works for any two-dimensional matrix W_{λ_j} it is straightforward to note that the same procedure must also work by linearity on $\bigoplus_j W_{\lambda_j}$ as well. Specifically, let $F = \oplus_j F_{\lambda_j}$ be the operation that we get when applying the above transformation to the entire Hilbert space of H and the ancillary qubit. Let $|\psi\rangle = \sum_j a_j|\lambda_j\rangle$ then

$$\begin{aligned} F\,\texttt{PREPARE}|0\rangle|\psi\rangle &= \sum_j F_{\lambda_j}\texttt{PREPARE}|0\rangle|\lambda_j\rangle \\ &= \texttt{PREPARE}|0\rangle e^{-iHt}|\psi\rangle. \end{aligned} \tag{51}$$

In order to estimate the cost of this procedure, let us consider the number of queries made to PREPARE and SELECT. An invocation of the W requires three queries. From Theorem 7 we can perform the singular value transformation with a number of applications of W that is in $\mathcal{O}(\alpha t + \log(1/\epsilon)/\log\log(1/\epsilon))$. The claimed result then follows by multiplication of this cost by the $\mathcal{O}(1)$ cost of implementing W. ■

Next we use the fact that this polynomial can be constructed using Theorem 6 we see that a degree $\mathcal{O}(\log(1/\epsilon))$ polynomial can be implemented using $\mathcal{O}(\log(1/\epsilon))$ controlled unitary operations and as for our circuits there are

$\mathcal{O}(1)$ two-qubit gate applied per controlled unitary and thus the result holds. The above algorithm is known to be optimal as improvements upon this scaling would lead to an algorithm that could compute the parity function using fewer than $\mathcal{O}(N)$ queries to the underlying quantum bits [1]. Importantly, however, our approach can be used to simplify the logic of Hamiltonian simulation by removing the need to combine polynomials of different parity.

Next, we will use $e^{it\sin H}$ and $e^{it\cos H}$ as our building blocks to build other phase functions. To this aim, we invoke one of the main technical results of Ref. [[23], Lemma 37] about approximating smooth functions by Fourier series.

Theorem 9. Let $\delta,\epsilon\in(0,1)$ and $f:\mathbb{R}\mapsto\mathbb{C}$ such that $|f(x)-\sum_{k=0}^{K}a_k x^k|\le(\epsilon/4)$ for all $x\in[-1+\delta,1-\delta]$. Then $\exists c\in\mathbb{C}^{2M+1}$ such that

$$|f(x)-\sum_{m=-M}^{M}c_m e^{\frac{i\pi m}{2}x}|\le\epsilon \tag{52}$$

for all $x\in[-1+\delta,1-\delta]$, where $M=\max(2\lceil\log(4\|a\|_1/\epsilon)1/\delta\rceil,0)$.

A notable property of the prior result is that the bounds on the Fourier series do not depend on the degrees of the polynomials terms. This can however be expected since the terms that have large degree make negligible contributions due to the restricted domain $x\in[-1+\delta,1-\delta]$, and therefore we can drop them without loss of accuracy. Note that these results have been traditionally used in conjunction with LCU methods to implement arbitrary functions of a generator [11]. However, here the main difference between the LCU results and ours is that our approach does not require scaling by the inverse 1-norm of the coefficients $\vec{c}$ or the additional polylogarithmic memory. The 1-norm improvement of our algorithm is because LCU is a more general algorithm that builds a linear combination of arbitrary unitaries and not just the powers of the same unitary, because of this LCU puts more restrictions on the probability of success. As a straightforward example of this approach, let us consider the case of fractional queries to a unitary matrix.

B. Application to fractional queries

Fractional queries are a generalization of the following problem listed by Scott Aaronson as one of "The ten most annoying questions in quantum computing": given a unitary U, can we implement $\sqrt{U}=e^{i(H/2)}$? [24] Or more generally $U^t=e^{itH}$ for $t\in(0,1)$? Sheridan *et al.* [25] first gave an algorithm to implement the fractional power of a unitary matrix that runs in $\mathcal{O}(\max(1/\delta,1/\epsilon)\log(1/\epsilon))$, which was then improved upon exponentially in terms of the error dependence by Ref. [11] to run in $\mathcal{O}(1/\delta\log(1/\epsilon))$. Sheridan *et al.* [25] also proved a lower bound on this problem, which shows that the δ dependence of this algorithm is actually optimal. While the optimal lower bound in ϵ is not known, here we provide an improvement of the algorithm, which leads to an additive logarithmic factor in $1/\epsilon$ rather than a multiplicative logarithmic factor. Hence, our algorithm here not only improves upon previous techniques, but it is in fact provably optimal up to double logarithmic factors.

Theorem 10. Let $U=e^{iH}$ be a unitary operator, let $t,\epsilon\in(0,1)$, and assume $\sigma_{\min}(H)\ge\delta$ where $\sigma_{\min}(H)$ is the minimum singular value of H. We can then implement an ϵ approximation of $U^t=e^{itH}$ with $\mathcal{O}\left((1/\delta)+\log(1/\epsilon)\right)$ uses of controlled-U and 2-qubit gates using $\mathcal{O}(\log(1/\delta))$ ancilla qubits. Further, no algorithm is possible that solves the algorithm using $o(1/\delta)$ queries.

Proof. First notice that for $x\in[0,2\pi]$:

$$e^{itx}=\begin{cases}e^{it\arccos(\cos(x))}, & 0\le x\le\pi\\ e^{it(2\pi-\arccos(\cos(x)))}, & \pi<x<2\pi\end{cases}. \tag{53}$$

We hope to use Theorem 9 to build $e^{it\arccos(y)}$, with $y=\cos(x)$ and make the appropriate phase shifts in the second half of the domain to build e^{itx}. However, notice that $y=\cos(x)\in[-1,1]$, which blows up the number of terms in Theorem 9. So the idea here is to write a more fine-grained piecewise representation of e^{itx} such that $y\in[-1/\sqrt{2},1/\sqrt{2}]$, hence getting rid of the δ dependence in Theorem 9, so the approximation part of the algorithm runs in $\mathcal{O}(\log(1/\epsilon))$. We would still need to distinguish the states at the branch-cut, which depends on δ, however, this is a separate part of the algorithm and that is why we end up with a $\mathcal{O}\left((1/\delta)+\log(1/\epsilon)\right)$ runtime instead of the previous best algorithm, which achieves a $\mathcal{O}\left((1/\delta)\log(1/\epsilon)\right)$. We can write this more fine-grained piecewise representation of e^{itx} as

$$e^{itx}=\begin{cases}e^{it\arcsin(\sin(x))}, & 0\le x\le\frac{\pi}{4}\\ e^{it\arccos(\cos(x))}, & \frac{\pi}{4}\le x\le\frac{3\pi}{4}\\ e^{it(\pi-\arcsin(\sin(x)))}, & \frac{3\pi}{4}\le x\le\frac{5\pi}{4}\\ e^{it(2\pi-\arccos(\cos(x)))}, & \frac{5\pi}{4}\le x\le\frac{7\pi}{4}\\ e^{it(2\pi+\arcsin(\sin(x)))}, & \frac{7\pi}{4}\le x<2\pi\end{cases}. \tag{54}$$

In order to implement an ϵ approximation of $U^t=e^{itH}$, we first perform a δ-precise phase estimation, and use the first three qubits storing the phase-estimation results to figure out which 8^{th} of the unit circle we are in, and apply the corresponding transformation based on Eq. (54).

Since by assumption the smallest eigenvalue of H in terms of norm is $\geq \delta$, no eigenvector will be miss categorized at the branch cut. Furthermore, it does not matter if we miss categorize the eigenvalues at the other boundary points since the piecewise function in Eq. (54) is continuous at all other boundaries, and thus applying the neighboring piece will still result in a correct transformation. Then since a δ-precise phase estimation has query complexity $\mathcal{O}(1/\delta)$, it remains to perform an ϵ approximations of $e^{it\arcsin(\sin(x))}$ and $e^{it\arccos(\cos(x))}$ using $\mathcal{O}(\log(1/\epsilon))$ controlled-U and 2-qubit gates.

First let us look at the Taylor series of $e^{it\arcsin(x)}$. One can see that the 1-norm of the coefficients of the Taylor series of $t\arcsin(x)$ is $|t|\arcsin(1) = |t|(\pi/2)$. Therefore, for $t \in [-2/\pi, 2/\pi]$ we get that the 1-norm of the Taylor series of $e^{it\arcsin(x)}$ is $\leq e^1 = e$. Also notice that $\|\sin(x)\|$ and $\|\cos(x)\|$ are both $\leq \frac{1}{\sqrt{2}}$ on the domains where they are being used in Eq. (54), hence eliminating the dependence on δ for Theorem 9. Therefore, for $t \in [-2/\pi, 2/\pi]$, we can write

$$\left| e^{it\arcsin(\sin(x))} - \sum_{m=-M}^{M} c_m e^{\frac{i\pi m}{2}\sin(x)} \right| \leq \epsilon. \tag{55}$$

For $M \in \mathcal{O}(\log(1/\epsilon))$. Hence it suffices to ϵ/M approximate each term $e^{(i\pi m/2)\sin(x)}$, which by Theorem 7 requires a polynomial of degree at most $\mathcal{O}(M + \log(M/\epsilon)) \sim \mathcal{O}(\log(1/\epsilon) + \log(\log(1/\epsilon)/\epsilon)) \subseteq \mathcal{O}(\log(1/\epsilon))$. Notice that the sum of all the polynomials can be written as a single polynomial, hence at the end we only require a single polynomial of degree $\mathcal{O}(\log(1/\epsilon))$. Furthermore, $\arccos(x) = (\pi/2) - \arcsin(x)$, thus building $e^{it\arccos(\cos(x))}$ will have the same runtime as $e^{it\arcsin(\sin(x))}$. Then since For $t \in [-1, 1]$, U^t can then be implemented by applying $U^{t/2}$ twice, $U^t = e^{itH}$ can be implemented with complexity $\mathcal{O}((1/\delta) + \log(1/\epsilon))$ for all $t \in [-1, 1]$.

To show optimality, it is known from Ref. [25] that $o(1/\delta)$ scaling is impossible for fractional queries. This shows that the only remaining question is whether the ϵ scaling is optimal. ■

C. Application to square roots/bosonic simulation

Next let us consider building a series expansion for the square root phase function of a unitary operator. Without loss of generality, every unitary matrix can be expressed as e^{iH} for Hermitian H and our aim is to examine as an example a method for constructing $e^{it\sqrt{H}}$. The reasons why we would want to do this are many, but the simplest example involves implementing the exponential of a bosonic operator of the form $e^{i(a^\dagger + a)t}$ where $a = \sum_{j\geq 0}\sqrt{j}\,|j+1\rangle\langle j|$. Conventionally this operator is implemented using an arithmetic circuit [26–28], which although polylogarithmic in depth [29], involves substantial constant factors and requires many ancillary qubits to carry out the reversible circuitry needed for the logic.

Theorem 11. Let $U = e^{iH}$ be a finite-dimensional unitary operator, let $\delta, \epsilon \in (0, 1)$, $t \in \mathbb{R}$, and assume $\sigma_{\min}(H) \geq \delta$ where $\sigma_{\min}(H)$ is the minimum singular value of H. Then we can implement an ϵ approximation of $e^{it\sqrt{H}}$ using $\mathcal{O}(1/\delta\,(\log(1/\epsilon) + |t|))$ uses of controlled-U and 2-qubit gates using $\mathcal{O}(\log 1/\delta)$ ancilla qubits.

Proof. In order to build $e^{it\sqrt{H}}$, we will build $e^{i\sqrt{2\pi}t\sqrt{y+1}}$ using Theorem 9 and sub in $y = (x/2\pi) - 1$ to ensure $y \in [-1 + \mathcal{O}(\delta), 1 - \mathcal{O}(\delta)]$. Subbing this into Eq. (52) we get

$$|e^{it\sqrt{x}} - \sum_{m=-M}^{M} c_m e^{\frac{i\pi m}{2}(\frac{x}{2\pi}-1)}| \leq \epsilon. \tag{56}$$

This is equivalent to the following:

$$|e^{it\sqrt{x}} - \sum_{m=-M}^{M} c_m e^{\frac{ixm}{4}} e^{\frac{-i\pi m}{2}}| \leq \epsilon. \tag{57}$$

Absorbing the constant into the coefficient and rewriting the sum gives us

$$\sum_{m=-M}^{M} c'_m e^{\frac{ixm}{4}} = \sum_{j=0}^{3} e^{\frac{ixj}{4}} \sum_{k=-M/4}^{M/4} (c'_{4k+j}) e^{ikx}. \tag{58}$$

The result of Corollary 8 shows that an ϵ/M approximation of $e^{iHj/4}$ can be built using a polynomial of degree $\log(M/\epsilon)$ of U and an initial δ-precise phase estimation. Thus, we can build the above approximation using a polynomial of degree $\mathcal{O}(\log(M/\epsilon) + M)$ and an initial δ-precise phase estimation. Since

$$M \in \mathcal{O}\left(\frac{1}{\delta}\left(\log\left(\frac{1}{\epsilon}\right) + \log(\|a\|_1)\right)\right). \tag{59}$$

We can implement $e^{it\sqrt{H}}$ using $\mathcal{O}(M)$ uses of controlled-U and 2-qubit gates and $\mathcal{O}(\log 1/\delta)$ ancilla qubits. So it remains to show $\log(\|a\|_1) \in \mathcal{O}(|t|)$.

Let $f(x) = e^{it\sqrt{x+1}} = \sum_{n=0}^{\infty} a_n x^n$, and $g(x) = f'(x) = (it/2\sqrt{x+1})e^{it\sqrt{x+1}} = \sum_{n=0}^{\infty} b_n x^n$. So we have that $a_{n+1} = (b_n/n+1)$. Next, we will use the Hardy inequality from Ref. [30], which states that if $g(z) = \sum_{n=0}^{\infty} b_n z^n \in H^1$, then

$$\sum_{n=0}^{\infty} \frac{|b_n|}{n+1} \leq \frac{1}{2} \sup_{0<r<1} \int_0^{2\pi} |g(re^{ix})|\,dx. \tag{60}$$

So we get that

$$\sum_{n=0}^{\infty}|a_n| = |e^{it}| + \sum_{n=0}^{\infty}\frac{|b_n|}{n+1} \leq 1 + \frac{1}{2}\int_0^{2\pi}|g(e^{ix})|\,dx. \tag{61}$$

Notice,

$$\frac{1}{2}\int_0^{2\pi}|g(e^{ix})|\,dx = \frac{1}{2}\int_0^{2\pi}\left|\frac{ite^{it}\sqrt{e^{ix}+1}}{2\sqrt{e^{ix}+1}}\right|dx. \tag{62}$$

Since $1+e^{i\theta} = 2\cos(\theta/2)\,e^{i\theta/2}$, then $\sqrt{1+e^{i\theta}} = \sqrt{2\cos(\theta/2)}e^{i\theta/4}$. We have that for $\theta \leq \pi$

$$\Re\left(it\sqrt{2\cos\left(\frac{\theta}{2}\right)}e^{i\theta/4}\right) = -t\sin\left(\frac{\theta}{4}\right)\sqrt{2\cos\left(\frac{\theta}{2}\right)} \tag{63}$$

and similarly for $\theta \geq \pi$

$$\Re\left(it\sqrt{2\cos\left(\frac{\theta}{2}\right)}e^{i\theta/4}\right) = -t\cos\left(\frac{\theta}{4}\right)\sqrt{-2\cos\left(\frac{\theta}{2}\right)}. \tag{64}$$

Thus,

$$\frac{1}{2}\int_0^{\pi}|g(e^{ix})|\,dx \leq \frac{|t|}{4}\int_0^{\pi}\frac{e^{-t\sin(\theta/4)\sqrt{2\cos\left(\frac{\theta}{2}\right)}}}{\sqrt{2\cos\frac{\theta}{2}}}d\theta \tag{65}$$

$e^{-t\sin(\theta/4)\sqrt{2\cos(\theta/2)}}$ has a maximum of $e^{\frac{|t|}{2}}$. Therefore,

$$\frac{1}{2}\int_0^{\pi}|g(e^{ix})|\,dx \leq \frac{|t|e^{\frac{|t|}{2}}}{4}\int_0^{\pi}\frac{d\theta}{\sqrt{2\cos\frac{\theta}{2}}} \tag{66}$$

The integral is a constant (approximately 3.7), thus,

$$\frac{1}{2}\int_0^{\pi}|g(e^{ix})|\,dx \leq |t|e^{\frac{|t|}{2}}. \tag{67}$$

Repeating the same argument for $\theta \geq \pi$ we see that

$$\frac{1}{2}\int_{\pi}^{2\pi}|g(e^{ix})|\,dx \leq |t|e^{\frac{|t|}{2}}, \tag{68}$$

$$\sum_{n=0}^{\infty}|a_n| \leq 1 + 2|t|e^{\frac{|t|}{2}}. \tag{69}$$

Then from the monotonicity of the logarithm function,

$$\log\left(\sum_n|a_n|\right) \leq \log(1+2|t|e^{|t|/2}) \in \mathcal{O}(|t|). \tag{70}$$

Our result then follows by combining Eqs. (70) and (59) ■

In this section, we explored applications of our generalized quantum signal processing framework within the context of phase functions to implement unitary operators. We gave a conceptually simplified formulation of the Hamiltonian simulation algorithm within the qubitization formalism with a factor 2 reduction in the number of queries to the walk operator in cases where independent queries to U and $U^\dagger$ are needed (although in many simulation examples both can be implemented using a single query [31]). We further proposed an enhanced and provably optimal algorithm for implementing fractional queries of unitary operators. Lastly, we gave a novel approach for implementing square root phase functions of unitary matrices with applications in the implementation of bosonic operators. In the next section, we apply our method to implementing nonunitary operations by extending the concept of Fourier decomposition to normal matrices.

VI. NORMAL MATRIX FACTORIES

In this section, we extend the concept of Fourier decomposition to normal matrices by utilizing polynomials of a special unitary matrix. Leveraging this insight, we demonstrate that any normal matrix can be written as a polynomial of the root of unity unitary in its eigenbasis. This result will then be used as a basis for synthesizing normal matrices in various contexts. We will begin by discussing the synthesis of diagonal matrices, demonstrating that this can be accomplished with a relatively low number of single and two-qubit gates. We then further apply this method to create convolution matrices and discuss the application of our approach to implementing convolution operators and solving systems of equations involving convolution.

Given $N \in \mathbb{N}$, we define $\omega_N = e^{i2\pi/N}$ to be a root of unity. This mathematical construct provides a key building block in defining our main subject, the root of unity matrix U_{ω_λ}, in relation to a given basis set. Let us consider an orthonormal basis $\{|\lambda_j\rangle\}_{j=0}^{N-1}$. With respect to this basis, we define the root of unity matrix as

$$U_{\omega_\lambda} = \sum_{j=0}^{N-1}\omega_N^j|\lambda_j\rangle\langle\lambda_j|. \tag{71}$$

Note that U_{ω_λ} is a unitary and diagonal matrix in the basis $\{|\lambda_j\rangle\}_{j=0}^{N-1}$. In an analogy with the complex exponential function $e^{i\theta}$, we will investigate the set of operators $\{U_{\omega_\lambda}^n\}_{n=0}^{N-1}$. Our goal is to demonstrate that this set constitutes a complete orthonormal operator basis for the space of all normal operators in the basis $\{|\lambda_j\rangle\}_{j=0}^{N-1}$. This fact can easily be seen to follow from the properties of the discrete Fourier transform, but a proof is given below for completeness.

Lemma 12. Given an $N \times N$ normal matrix $\mathcal{A} = \sum_{j=0}^{N-1} \lambda_j |\lambda_j\rangle\langle\lambda_j|$ for $\lambda_j \in \mathbb{C}$, $\mathcal{A}$ can be written as

$$\mathcal{A} = \sum_{n=0}^{N-1} c_n U_{\omega_\lambda}^n,$$

where $c_n = \langle U_{\omega_\lambda}^n, \mathcal{A}\rangle = (1/N)\mathrm{Tr}(\mathcal{A}\, U_{\omega_\lambda}^{-n})$.

Proof. First, note that the inner product between two powers of U_{ω_λ} satisfies the orthogonality property by the properties of Fourier sums:

$$\langle U_{\omega_\lambda}^n, U_\omega^m\rangle = \left(\sum_j \omega_N^{j(n-m)}\right)/N = \delta_{m,n}. \tag{72}$$

This implies that

$$\begin{aligned}\sum_n c_n U_{\omega_\lambda}^n &= N^{-1}\sum_{n,j}\lambda_j\,\omega_N^{-nj}\sum_k \omega_N^{kn}|\lambda_k\rangle\langle\lambda_k| \\ &= \sum_{jk}\delta_{jk}\lambda_j|\lambda_k\rangle\langle\lambda_k| = \mathcal{A}.\end{aligned} \tag{73}$$

Thus the powers of the matrix U_{ω_λ} form a complete operator basis for normal matrices in the $\{|\lambda_j\rangle\}_{j=0}^{N-1}$ basis. As the basis set is orthonormal in this space, it, therefore, is a complete orthonormal operator basis as claimed. ■

Building upon the foundational results established in Lemma 12, we are now equipped to introduce a framework for synthesizing normal matrices. We first show this in the computational basis for implementing diagonal matrices, and then extending the result to arbitrary bases. In particular, we will demonstrate how we can utilize this framework in the quantum Fourier basis to implement convolution operators.

A. Synthesizing diagonal matrices

In the computational basis, the construction of the root of unity matrix U_{ω_λ} is straightforward and can be achieved using $\mathcal{O}(\log(N))$ many single-qubit gates, providing an efficient method to build the operator (Fig. 4). This method's efficiency highlights the computational advantages of our framework and further establishes the value of our approach in the context of quantum information processing. From this point forward, we will denote U_ω as shorthand for U_{ω_λ} in the computational basis. In mathematical terms, this gives us

$$U_\omega = \sum_{j=0}^{N-1}\omega_N^j|j\rangle\langle j|, \tag{74}$$

where $|j\rangle$ is the binary bitstring representation of integer j. This succinct notation allows us to concisely express

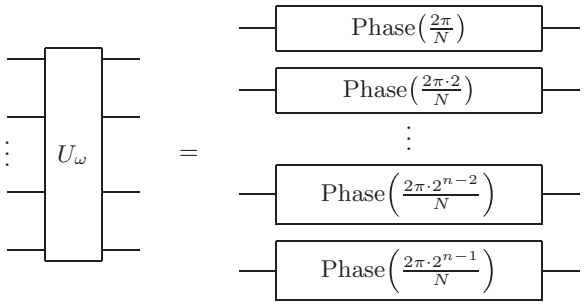

FIG. 4. Quantum circuit implementing U_ω for a system of $n = \log N$ qubits. On the left, a multiqubit gate U_ω is represented. This gate is equivalent to the application of phase gates on each qubit, as shown on the right. The phase angle doubles with each increase in the qubit's index. Here, the phase gate is defined as $\mathrm{phase}(\theta) = \left[\begin{smallmatrix}1 & 0\\ 0 & e^{i\theta}\end{smallmatrix}\right]$.

the operator in terms of the computational basis states. Lemma 12 leads to the following theorem for the synthesis of diagonal matrices.

Theorem 13. Given an $N \times N$ diagonal matrix $\mathcal{A} = P(U_\omega)$, where $\deg(P) = d$, and $|P|^2 \le c$ on on the complex unit circle, we can implement $\mathcal{A}/\sqrt{c}$ using $\mathcal{O}(d\log N)$ 1- and 2-qubit gates.

Proof. By Lemma 12, we know that $\mathcal{A}$ can be written as

$$\mathcal{A} = \sum_{n=0}^{N-1}\alpha_n U_\omega^n. \tag{75}$$

If $|P|^2 \le c$, then $\left|P/\sqrt{c}\right|^2 \le 1$. Thus, as shown by Corollary 5, it is possible to find specific parameters $\vec{\theta}, \vec{\phi} \in \mathbb{R}^{d+1}$ and $\lambda \in \mathbb{R}$ to plug into our generalized quantum signal processing framework, such that it results in

$$\begin{bmatrix} \dfrac{\mathcal{A}}{\sqrt{c}} & \cdot \\ \cdot & \cdot \end{bmatrix}. \tag{76}$$

Therefore, we can implement $\mathcal{A}/\sqrt{c}$ using $\mathcal{O}(d)$ applications of controlled U_ω. And as U_ω can be implemented using $\mathcal{O}(\log(N))$ many single-qubit gates, it follows that $\mathcal{A}/\sqrt{c}$ can be implemented using $\mathcal{O}(d\log N)$ 1- and 2-qubit gates. ■

This theorem paves the way for an ensuing corollary, showcasing a specific application of the above to the construction of Bit Functions. This procedure can be seen as a generalization of methods developed in Ref. [32], where only implementations of unitary bit functions are considered.

Corollary 14 (Bit functions). Given a function $f(x) = \sum_{n=0}^{d} \alpha_n e^{ix2\pi/N}$ where $|f|^2 \leq 1$, we can implement the $N \times N$ diagonal matrix $\mathcal{A}$ such that

$$\mathcal{A}|x\rangle = f(x)|x\rangle$$

using $\mathcal{O}(d \log N)$ 1- and 2-qubit gates.

Upon establishing the feasibility of synthesizing diagonal matrices using a Fourier decomposition into polynomials of U_ω, we expand our perspective beyond this initial construct. In what follows, we show that the construction of normal matrices using our method is not confined solely to the computational basis. Indeed, given a change of basis matrix, the same principle can be effectively used to synthesize matrices in any other bases.

Theorem 15. Given an $N \times N$ normal matrix $\mathcal{A} = P(U_{\omega_\lambda}) = \sum_{n=0}^{d} \alpha_n U_{\omega_\lambda}^n$, diagonalized by unitary matrices $\mathcal{Q}$ and $\mathcal{Q}^\dagger$, where implementing $\mathcal{Q}$ requires $\mathcal{O}(\chi)$ 1- and 2-qubit gates, $\deg(P) = d$, and $|P|^2 \leq c$ on the complex unit circle, we can implement $\mathcal{A}/\sqrt{c}$ using $\mathcal{O}(d \log N + \chi)$ 1 and 2-qubit gates.

Proof. Utilizing Theorem 13 we first build the polynomial in the computational basis:

$$\frac{1}{\sqrt{c}} \sum_{n=0}^{d} \alpha_n U_\omega^n \tag{77}$$

using $\mathcal{O}(d \log N)$ 1- and 2-qubit gates, and then apply the change of basis matrices to get

$$\begin{aligned} \frac{1}{\sqrt{c}} \mathcal{Q}^\dagger \Big(\sum_{n=0}^{d} \alpha_n U_\omega^n\Big) \mathcal{Q} &= \frac{1}{\sqrt{c}} \sum_{n=0}^{d} \alpha_n \mathcal{Q}^\dagger U_\omega^n \mathcal{Q} \\ &= \frac{1}{\sqrt{c}} \sum_{n=0}^{d} \alpha_n U_{\omega_\lambda}^n \\ &= \frac{\mathcal{A}}{\sqrt{c}}. \end{aligned} \tag{78}$$

Then since implementing $\mathcal{Q}$ requires $\mathcal{O}(\chi)$ 1- and 2-qubit gates, our total cost will be $\mathcal{O}(d \log N + \chi)$. ■

We now give an example of this in the Fourier basis (i.e., $\mathcal{Q} = \mathrm{QFT}$) to build convolution operators.

B. Synthesizing convolution matrices

In this subsection, we explore the synthesis of convolution matrices, which play a critical role in signal processing, image filtering, and numerous other applications. Recent work has considered the application of linear combinations of unitaries to implement circulant matrices [33] but QSP-based methods have not yet been fully explored. Our approach to synthesizing circulant matrices is based on GQSP and uses, in particular, the relationship between these matrices and circular convolutions. Specifically, a circulant matrix can be completely defined by a single vector, $\vec{c}$. This vector forms the first column of the matrix, and the remaining columns are each cyclic permutations of $\vec{c}$, each with an offset equal to the respective column index. A circulant matrix is shown below:

$$\begin{bmatrix} c_0 & c_{N-1} & \cdots & c_2 & c_1 \\ c_1 & c_0 & c_{N-1} & & c_2 \\ \vdots & c_1 & c_0 & \ddots & \vdots \\ c_{N-2} & & \ddots & \ddots & c_{N-1} \\ c_{N-1} & c_{N-2} & \cdots & c_1 & c_0 \end{bmatrix}. \tag{79}$$

For a circulant matrix C, given by the vector $|c\rangle = \sum_{j=0}^{N-1} c_j |j\rangle$, and another vector $|\psi\rangle = \sum_{j=0}^{N-1} x_j |j\rangle$, we can perform a convolution of $|\psi\rangle$ by $|c\rangle$ simply by multiplying $|\psi\rangle$ with the matrix C. This operation is critical for many signal processing applications:

$$C|\psi\rangle = \sum_{j=0}^{N-1} (\psi \star c)_j |j\rangle. \tag{80}$$

The resulting $(\psi \star c)_j$ of the convolution is defined as

$$(\psi \star c)_j = \sum_{k=0}^{N-1} c_k x_{[j-k \mod N]}. \tag{81}$$

A well-known characteristic of circulant matrices is that they can be defined by an associated polynomial of the cyclic permutation matrix $\mathcal{P}$. This polynomial association is particularly beneficial when constructing circuits for quantum operations, as it allows for a straightforward definition and synthesis of the required matrix:

$$C = c_0 I + c_1 \mathcal{P} + c_2 \mathcal{P}^2 + \cdots + c_{n-1} \mathcal{P}^{n-1}. \tag{82}$$

The cyclic permutation matrix $\mathcal{P}$ is defined as

$$\mathcal{P} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & 0 & 0 \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix}. \tag{83}$$

We can equivalently express the cyclic permutation matrix $\mathcal{P}$ as

$$\mathcal{P} = \sum_{j=0}^{N-1} |j + 1 \bmod N\rangle\langle j|. \tag{84}$$

The operator $\mathcal{P}$ is a cyclic adder which can be diagonalized using the quantum Fourier transform (QFT):

$$\mathcal{P} = \mathrm{QFT}^\dagger U_\omega \mathrm{QFT}. \tag{85}$$

Then by Theorem 15, we get the following lemma, which provides bounds on the circuit size needed to construct the circulant matrix in a 1- and 2-qubit gate library. Note that the dependence on an error target ϵ is absent here because of the assumed form of the polynomial decomposition and also because rotation synthesis is not needed in this (continuous) gate set.

Lemma 16. Given an $N \times N$ circulant matrix $C = \sum_{n=0}^{d} c_n \mathcal{P}^n$, we can build C (normalized) using only $\mathcal{O}(d \log N + \log^2 N)$ 1 and 2-qubit gates.

This lemma enables us to reach a valuable conclusion:

Theorem 17. Given $|\psi\rangle = \sum_{j=0}^{N-1} x_j |j\rangle$ and a filter $F = \{a_k\}_{k=-d}^{d}$, we can convolve ψ with F:

$$|\psi \star F\rangle = \sum_{m=0}^{N-1} (\psi \star F)_m |m\rangle,$$

where $(\psi \star F)_m = \sum_{k=-d}^{d} a_k x_{[m-k \mod N]}$, by using only $\mathcal{O}(d \log N + \log^2 N)$ 1- and 2-qubit gates.

A powerful application of the previous result is solving systems of equations that are expressed as discrete convolutions. For instance, consider a system of equations represented as

$$x \star c = b. \tag{86}$$

Such equations are significant because we often have a known filter function that is applied to an input and we would like to have an efficient method for inverting such a transformation to find the input that is partially obscured by the convolution. To see how this can be attained, we can see from the discrete convolution theorem that the original convolutional equation can be re-expressed as a linear system via is equivalent to

$$Cx = b, \tag{87}$$

where C is a circulant matrix. If C is invertible, we can then build C using Lemma 16 and invert it using the quantum matrix inversion algorithm to solve the system of equations. The cost of doing so within error ϵ is then $\tilde{\mathcal{O}}(d^2\kappa^2 \log^4(N) \log(1/\epsilon))$, where κ is the condition number of the circulant matrix C using the inversion method of Ref. [34]. Our approach allows us then to directly synthesize such matrices through the polynomial series definition of circulant matrices after diagonalization through the quantum Fourier transform.

VII. CONCLUSION

This paper introduces a substantial advancement to quantum signal processing—the generalized quantum signal-processing (GQSP) method. Unlike traditional QSP frameworks, our method employs a pair of rotations instead of solely relying on either Z or X rotations for signal-processing operations. This strategic modification enables us to move beyond the limitations of the original QSP framework.

Another essential contribution of our GQSP method is the significant simplification it offers in the computation of phase angles compared to existing methods. In instances where both P and Q are known, we introduce a straightforward recursive formula for the angles. This substantial pedagogical improvement addresses one of the key challenges in teaching QSP methods, as the traditional techniques for finding the polynomial function in QSP can be difficult to convey. Our approach simplifies this complex aspect of QSP, making it much more accessible. Additionally, we introduced an efficient optimization algorithm for computing phase angles when only P is known, but Q is not. Our tests showed that our method can compute polynomials of degree greater than 10^7 in under a minute, an impressive computational efficiency when compared to the 10^4 degree polynomials that can be achieved using existing QSP approaches via the state-of-the-art techniques of Ref. [14].

In this paper, we explored several applications of our GQSP methodology. We presented an optimized algorithm for quantum fractional queries, along with a simplified technique for performing Hamiltonian simulation using qubitization. We proposed methods for calculating phase functions, such as the square root, and unveiled new methodologies for synthesizing circulant matrices and performing convolution operations.

As we look forward, our work reveals several potential areas for further exploration. A primary question is how to adapt the principles of our approach to multivariable QSP [15]. Also, our focus on QSP suggests the potential of applying similar concepts to quantum singular value transformation for transforming block-encoded nonsquare matrices. In essence, this paper represents a significant progression in the QSP and QSVT framework. The exploration and understanding of the broad range of opportunities offered by these techniques promise to be a primary focus of research in quantum algorithms in the years to come.

ACKNOWLEDGMENTS

N.W. would like to acknowledge funding for this work from Google Inc. This material is based upon work supported by the U.S. Department of Energy, Office of Science, National Quantum Information Science

Research Centers, Co-design Center for Quantum Advantage (C2QA) under Contract No. DE-SC0012704 (PNNL FWP 76274). We would like to thank Dominic Berry for useful feedback.

[1] Dominic W. Berry, Andrew M. Childs, and Robin Kothari, in *2015 IEEE 56th annual symposium on foundations of computer science* (IEEE, 2015), p. 792.
[2] Guang Hao Low and Isaac L. Chuang, Optimal hamiltonian simulation by quantum signal processing, Phys. Rev. Lett. **118**, 010501 (2017).
[3] Guang Hao Low and Isaac L. Chuang, Hamiltonian simulation by qubitization, Quantum **3**, 163 (2019).
[4] Andrew M. Childs and Nathan Wiebe, Hamiltonian simulation using linear combinations of unitary operations, Quantum Inf. Comput. **12**, 901 (2012).
[5] Theodore J. Yoder, Guang Hao Low, and Isaac L. Chuang, Fixed-point quantum search with an optimal number of queries, Phys. Rev. Lett. **113**, 210501 (2014).
[6] Lov K. Grover, Fixed-point quantum search, Phys. Rev. Lett. **95**, 150501 (2005).
[7] Fré déric Magniez, Ashwin Nayak, Jérémie Roland, and Miklos Santha, Search via quantum walk, SIAM J. Comput. **40**, 142 (2011).
[8] P. W. Shor, in *Proceedings 35th Annual Symposium on Foundations of Computer Science* (IEEE Press, Los Alamitos, 1994), p. 124.
[9] M. Szegedy, in *45th Annual IEEE Symposium on Foundations of Computer Science* (IEEE Press, 2004), p. 32.
[10] John M. Martyn, Zane M. Rossi, Andrew K. Tan, and Isaac L. Chuang, Grand unification of quantum algorithms, PRX Quantum **2**, 040203 (2021).
[11] András Gilyén, Yuan Su, Guang Hao Low, and Nathan Wiebe, in *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2019, (Association for Computing Machinery, New York, NY, USA, 2019) p. 193.
[12] Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd, Quantum algorithm for linear systems of equations, Phys. Rev. Lett. **103**, 150502 (2009).
[13] Yulong Dong, Lin Lin, and Yu Tong, Ground-state preparation and energy estimation on early fault-tolerant quantum computers via quantum eigenvalue transformation of unitary matrices, PRX Quantum **3**, 040305 (2022).
[14] Yulong Dong, Xiang Meng, K. Birgitta Whaley, and Lin Lin, Efficient phase-factor evaluation in quantum signal processing, Phys. Rev. A **103**, 042419 (2021).
[15] Zane M. Rossi and Isaac L. Chuang, Multivariable quantum signal processing (M-QSP): Prophecies of the two-headed oracle, Quantum **6**, 811 (2022).
[16] Zane M. Rossi, Victor M. Bastidas, William J. Munro, and Isaac L. Chuang, Quantum signal processing with continuous variables, arXiv:2304.14383.
[17] Xin Wang, Youle Wang, Zhan Yu, and Lei Zhang, Quantum phase processing: Transform and extract eigen-information of quantum systems, arXiv:2209.14278.
[18] Dominic W. Berry, Andrew M. Childs, Richard Cleve, Robin Kothari, and Rolando D. Somma, in *Proceedings of the forty-sixth Annual ACM Symposium on Theory of Computing* (ACM, New York, NY, USA, 2014).
[19] Jeongwan Haah, Product decomposition of periodic functions in quantum signal processing, Quantum **3**, 190 (2019).
[20] Thais de Lima Silva, Lucas Borges, and Leandro Aolita, Fourier-based quantum signal processing, arXiv:2206.02826.
[21] Zhan Yu, Hongshun Yao, Mujin Li, and Xin Wang, Power and limitations of single-qubit native quantum neural networks, Adv. Neural Inf. Process. Syst. **35**, 27810 (2022).
[22] https://github.com/Danimhn/GQSP-Code.
[23] Joran Van Apeldoorn, András Gilyén, Sander Gribling, and Ronald de Wolf, Quantum SDP-solvers: Better upper and lower bounds, Quantum **4**, 230 (2020).
[24] Scott Aaronson, The ten most annoying questions in quantum computing. https://www.scottaaronson.com/blog/?p=112, 2006.
[25] L. Sheridan, D. Maslov, and M. Mosca, Approximating fractional time quantum evolution, J. Phys. A: Math. Theor. **42**, 185302 (2009).
[26] Katherine L. Brown, William J. Munro, and Vivien M. Kendon, Using quantum computers for quantum simulation, Entropy **12**, 2268 (2010).
[27] Rolando D. Somma, Quantum simulations of one dimensional quantum systems, arXiv:1503.06319.
[28] Alexander F. Shaw, Pavel Lougovski, Jesse R. Stryker, and Nathan Wiebe, Quantum algorithms for simulating the lattice Schwinger model, Quantum **4**, 306 (2020).
[29] Mathias Soeken, Martin Roetteler, Nathan Wiebe, and Giovanni De Micheli, in *Proceedings of the 54th Annual Design Automation Conference 2017* (Association for Computing Machinery, New York, NY, United States, 2017), p. 1.
[30] Peter L. Duren, *Theory of H p Spaces* (Academic Press, 1970).
[31] Ryan Babbush, Craig Gidney, Dominic W. Berry, Nathan Wiebe, Jarrod McClean, Alexandru Paler, Austin Fowler, and Hartmut Neven, Encoding electronic spectra in quantum circuits with linear T complexity, Phys. Rev. X **8**, 041015 (2018).
[32] Jonathan Welch, Daniel Greenbaum, Sarah Mostame, and Alan Aspuru-Guzik, Efficient quantum circuits for diagonal unitaries without ancillas, New J. Phys. **16**, 033040 (2014).
[33] S. S. Zhou and J. B. Wang, Efficient quantum circuits for dense circulant and circulant like operators, R. Soc. Open Sci. **4**, 160906 (2017).
[34] Andrew M. Childs, Robin Kothari, and Rolando D. Somma, Quantum algorithm for systems of linear equations with exponentially improved dependence on precision, SIAM J. Comput. **46**, 1920 (2017).